



UNIVERSIDAD DE MURCIA

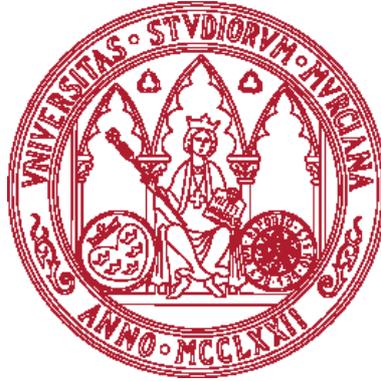
FACULTAD DE INFORMÁTICA

A Novel, Identity-Based Network Architecture
for the Future Internet

Una Arquitectura de Red basada en Identidad
para la Internet del Futuro

D. Pedro Martínez Juliá

2015



Universidad de Murcia
Facultad de Informática

**UNA ARQUITECTURA DE RED
BASADA EN IDENTIDAD
PARA LA INTERNET DEL FUTURO**

TESIS DOCTORAL

Presentada por:
Pedro Martínez Juliá

Supervisada por:
Dr. Antono F. Skármeta Gómez

Murcia, Septiembre de 2015

Resumen

El objetivo de la creación de Internet fue permitir la comunicación entre computadoras remotas sin monopolizar la infraestructura de comunicaciones subyacente e incrementando efectivamente la confianza y la tolerancia a fallos de las redes, lo que supuso un gran avance en las comunicaciones digitales. En su origen Internet estaba enfocada a una pequeña comunidad, pero hoy en día se ha convertido en una red mundial y en la infraestructura central que soporta nuestras vidas digitales. De esta forma, Internet es el lugar virtual donde nos reunimos, donde almacenamos nuestra información, donde vamos a buscar información de cualquier índole, la infraestructura a la cual nos conectamos y, sin remedio, el cuello de botella de nuestras comunicaciones.

En la actualidad cada vez más personas, artefactos digitales y objetos de muchos tipos (cosas) están conectados a Internet, cumpliendo con el objetivo de *todo continuamente conectado*, pero el diseño original de Internet no contempló esta evolución y, por esa razón, han surgido diversos problemas, tanto en el nivel de red como en el nivel de interconexión. Además, esas entidades constituyen la masa crítica de las redes, tanto del presente como del futuro, lo que requiere nuevas funcionalidades que son difíciles de conseguir con el modelo de red que actualmente se utiliza tanto en las redes de acceso como en Internet. Esta situación ha motivado la revisión de la arquitectura de Internet, potenciada además por los requisitos impuestos por una nueva e inmensa área de servicios, dispositivos e información, que han incrementado la presión por un cambio en el modelo de red y una evolución hacia la Internet del Futuro (Future Internet, FI).

Para resolver parte de esos problemas, en el seno de la comunidad investigadora han surgido propuestas para la separación de identificadores y localizadores, pero este modelo no es lo suficientemente completo como para tratar las entidades móviles de la misma forma que se tratan las entidades estáticas (hosts). La respuesta a este problema ha sido la propuesta de arquitecturas completas, las llamadas de *borrón y cuenta nueva*, las cuales proponen rehacer la arquitectura desde sus cimientos, previniendo que los problemas se manifiesten en lugar de parchearlos. La mayoría de estas propuestas ofrecen capacidades prometedoras, pero su impacto ha sido limitado porque requieren desechar la arquitectura

actual antes de ser desplegadas. Además, ninguna de ellas provee una solución integral que refleje en la red las interacciones solicitadas por personas y objetos. Por ejemplo, no hay ninguna solución para soportar los objetos reales que participan en las comunicaciones (personas, servicios, dispositivos, etc.) protegiendo al mismo tiempo la información sobre ellos y sus correspondientes contextos.

Con el trabajo realizado en la presente tesis hemos abordado estos problemas mediante la introducción del concepto de *identidad*, entendida como un conjunto de atributos, en las capas intermedias de la red. Nuestra propuesta radica en utilizar una capa de identidad que permita a los objetos de red (network objects) controlar su participación en el mundo digital, creando un panorama, en la esfera de las comunicaciones, que es singular, único y optimizado para cada entidad de red, dando lugar a un *plano de identidad* que permita a las entidades encontrarse siguiendo un modelo *identidad-a-identidad*.

Muchos pueden pensar que si permitimos a la red o a los servicios tener una comprensión mejor de las identidades (completas o partes de ellas) relacionadas con las entidades implicadas en las comunicaciones, se perderá su privacidad. Sin embargo, nuestra propuesta hace lo contrario: reforzar la privacidad mediante el control de la información a la que tienen acceso las entidades, servicios y elementos de red. Nuestra propuesta contempla la privacidad horizontal, mediante la que se protegen los datos entre servicios, y la privacidad vertical, que maneja la privacidad entre la red y los servicios o dentro de un mismo servicio. Todo esto incrementa la seguridad y privacidad globales del sistema, lo cual provee al usuario de un nivel superior de control, soportado por los mecanismos necesarios para hacer cumplir sus preferencias.

Partiendo de un nuevo modelo de red *identidad-a-identidad*, el objetivo final de nuestra propuesta es la creación de una arquitectura de comunicación digital escalable que refleje la estructura del mundo real: las personas hablan con personas, con objetos, los objetos entre sí y, en general, las entidades, representadas por identidades digitales, se comunican las unas con las otras. Para ello se ha tenido en cuenta que la arquitectura necesita considerar todos los aspectos de las comunicaciones, a saber, “con quién estoy hablando”, la confianza que tengo, o incluso la localización donde está teniendo lugar la comunicación. Estos atributos afectan tanto a la seguridad como a la privacidad y están relacionados con la identidad de los participantes.

Desde un punto de vista analítico, Internet es un grafo global de elementos y redes interconectadas. Cada red es a su vez un grafo y cada elemento (nodo) es un vértice que se conecta con otros elementos mediante aristas (enlaces de red), salvo los nodos extremos, que se conectan con un único elemento intermedio de la red. Esta organización se ha utilizado como premisa para definir los protocolos de red y transporte actuales, sin considerar

ningún dinamismo en la misma. Sin embargo, dicha premisa ya no es válida debido a la existencia de dispositivos móviles conectados a Internet que pueden cambiar con frecuencia su punto de unión a la red. Además, la característica de conexión múltiple (multi-homing), ampliamente presente hoy en día, ha permitido que los nodos extremos tengan más de una arista, cosa que también ocurre en el caso de los dispositivos móviles mientras se mueven entre dos redes. Todo esto nos ha llevado a la definición de *multihoming dinámico* (conexión múltiple dinámica), concepto que considera que multihoming y movilidad son dos aspectos de la misma cosa, presentando los mismos problemas y requisitos.

En nuestro trabajo de investigación hemos ahondado también en la discusión acerca de la separación de identificadores y localizadores, extrayendo los requisitos más importantes y generalizando este concepto en la dirección de *desacoplar los procesos de identificación y localización*. Manteniendo el mismo objetivo, separar la identificación de los elementos del direccionamiento de hosts en la red, la arquitectura propuesta, a diferencia de las anteriores, enfatiza el proceso de identificación de forma independiente a los mecanismos subyacentes que se utilizan para direccionar los mensajes en la red. Este mecanismo pretende además resolver el problema actual de escalabilidad presente en Internet ya que mediante la provisión de un mecanismo separado para la identificación de los extremos de comunicación sin contar con los identificadores o localizadores subyacentes (direcciones IP), estos últimos se pueden estructurar para mejorar las tablas de encaminamiento, utilizando, por ejemplo, una distribución geográfica de direcciones de red (o direccionamiento geográfico en general).

Está claro que dotar a la red de una función de identificación independiente, coherente y basada en identidades impone un requisito crucial, a saber, obtener los identificadores/localizadores utilizados en la red subyacente a partir de identidades. Éste es un reto con alto grado de paralelismo con el proceso de resolución incluido en las propuestas existentes para la separación de identificadores y localizadores. Sin embargo, ninguna de estas soluciones ha sido ampliamente aceptada por lo que en la presente tesis proponemos además mejorar la arquitectura de red con un nuevo sistema de resolución independiente, basado en identidad, altamente escalable y eficiente, que haga efectivo el desacoplamiento de los procesos de identificación y localización.

Nuestro diseño también se ve influenciado por la necesidad de un mecanismo integrado para el descubrimiento de entidades en FI. Actualmente, la mayoría de operaciones de búsqueda y descubrimiento deben realizarlas humanos, a través de motores de búsqueda bien conocidos, lo que no es nada adecuado para las funciones de la capa de red. Una solución apropiada para el descubrimiento en FI debe proveer mecanismos para encontrar cualquier objeto de red (contenido, dispositivos, personas, etc.) a partir de una descripción parcial del mismo. Un identificador o nombre no es suficiente ya que no representa las

propiedades reales de un objeto de red, hecho que se torna más crítico debido al incremento de datos, información, personas, máquinas y, en general, al gran número de *objetos de red* conectados a Internet. Como los mecanismos actuales para la descripción, búsqueda y descubrimiento de los objetos de red están basados en sus nombres, generalmente nombres de dominio (incluyendo *full-qualified domain names*, FQDNs), cuando no se conoce el nombre, las entidades tienen que contactar con una tercera parte que mantiene una lista de objetos de red disponibles y realiza una búsqueda según el criterio indicado. Esto requiere involucrar a personas en la mayoría de los casos, lo que demuestra que los mecanismos de red actuales presentan una falta clara de funcionalidad. Considerando ésta una característica importante, la función de identificación del *plano de identidad* definido en nuestra propuesta incluye capacidades de búsqueda y descubrimiento.

Con el fin de proveer los bloques funcionales apropiados, que incluyan las funciones necesarias que se han comentado hasta ahora, partiendo de la base de dotar al sistema de la capacidad de evolución y manteniendo desacoplados los procesos de identificación y localización, hemos definido los objetivos principales de la tesis de la siguiente forma:

Diseñar, analizar y validar una arquitectura de red altamente modular y evolutiva para la Internet del Futuro, que abstraiga los extremos de comunicación desde hosts hasta objetos de red arbitrarios mediante el uso de sus identidades (conjuntos de atributos) como mecanismo que les permita descubrir, direccionar y comunicarse entre ellos de forma dinámica, inequívoca, segura y privada, en un entorno heterogéneo y cambiante.

En esta definición recogemos el dinamismo de la red, un aspecto clave de las tendencias actuales sobre servicios y elementos de red (dispositivos e infraestructuras). Además incidimos en la necesidad de abstraer los extremos de la red para conseguir flexibilidad y granularidad, y consideramos que las identidades, concebidas como conjuntos de atributos (tal y como se define en ITU-T X.1250), son las mejores candidatas para una identificación inequívoca. Para alcanzar estos objetivos hemos diseñado un conjunto de bloques funcionales que trabajan conjuntamente, formando una arquitectura de red completa, o por separado, para proveer cualidades adicionales a otras arquitecturas. Estos componentes son el Domain Trusted Entity Infrastructure (DTEi), el Identity-Based Network Protocol (IBNP), el Ontology-based Distributed Information Network (ODIN), y el Identity-Based Control Plane (IBCP). Aunque estos bloques funcionales están estrechamente relacionados entre sí y acoplados hasta cierto punto, mantienen su independencia para proveer su función de red sin necesidad de desplegar otro componente.

El DTEi es una infraestructura que permite a los objetos de red encontrarse unos a otros en términos de identidad. Cada dominio de red o administrativo desplegará un nodo del DTEi que le permitirá mantener una relación de confianza con los objetos de su dominio, de forma que el DTEi pueda intermediar en las operaciones de control y así garantizar que se realizan de forma segura, cumpliendo las políticas establecidas por cada objeto de red e incluyendo la gestión segura y privada de sus sesiones de comunicación. Para conseguir un sistema totalmente descentralizado, a la vez que lo desacoplamos de la red subyacente, la infraestructura se construye como una red superpuesta (overlay network) formada por todos los nodos del DTEi.

Para complementar el DTEi hemos diseñado el IBNP. Este bloque funcional consiste en un protocolo para el plano de control y un protocolo sencillo para el plano de datos, habiendo sido ambos concebidos para permitir a los objetos de red hablar unos con otros en términos de sus identidades correspondientes, haciendo innecesarios los identificadores fijos o localizadores (como direcciones IP) para establecer sus sesiones de comunicación, que serán únicamente utilizados en la red subyacente una vez establecida la sesión. Aunque hemos diseñado estos protocolos para interactuar con el DTEi, también pueden ser utilizados por separado para que los objetos de red gestionen sus sesiones con otros objetos sin necesidad de la intermediación del DTEi, siendo la instanciación del protocolo de red subyacente la única diferencia entre ambos usos. Además, estos protocolos soportan entornos y contextos cambiantes y heterogéneos de forma implícita y transparente, lo que incluye el soporte de tecnologías subyacentes diferentes, así como la multiplicidad de redes (multihoming) y la dinamicidad en las conexiones de los objetos de red (movilidad), características que aunamos en el concepto de *multihoming dinámico*.

Por otro lado, hemos diseñado ODIN para ofrecer funciones de búsqueda y descubrimiento basadas en identidad. Este mecanismo permite encontrar objetos de red mediante términos abstractos, independientemente de su localización actual o su estado dinámico, a la vez que se incrementa la granularidad de los extremos de comunicación. Las búsquedas se realizan mediante una *identidad parcial* que consiste en un conjunto incompleto de atributos acerca de la identidad que se quiere encontrar. Además, la estructura de ODIN refuerza la privacidad de los objetos de red previniendo el trazado de sus operaciones y aportando seguridad sin necesidad de mecanismos externos. Este componente incluye una ontología, utilizada para describir las propiedades de los objetos de red y exponer las capacidades que *ofrecen*. Al igual que el DTEi, ODIN se construye sobre una red superpuesta (overlay network) formada por la unión de todos los nodos, los cuales pueden alojarse junto a los nodos del DTEi y así intercambiar, de forma segura y eficiente, atributos de las identidades que gestionan.

Integrando las funciones ofrecidas por DTEi, IBNP, y ODIN construimos una arquitectura completa que lleva el concepto de *identidad* a la red, permitiendo a los objetos de red jugar roles diferentes dependiendo de sus necesidades particulares. Esta característica nos conduce hasta un nuevo modelo de comunicaciones que abstrae el concepto de extremo de comunicación, desligándolo del *host* para incrementar la granularidad y así permitir la existencia de objetos de red en diferentes niveles de abstracción. Además, estos objetos se pueden instanciar en redes subyacentes heterogéneas, utilizando su identidad para representarlos. No obstante, para facilitar la integración de estas funciones con otras arquitecturas necesitamos un componente que se una al plano de control de la red, por lo que hemos diseñado IBCP.

El objetivo de IBCP es encapsular las cualidades provistas por nuestra arquitectura para poder integrarlas con otras arquitecturas de red sin necesidad de cambiar su modelo de operación. Así, IBCP ofrece un conjunto de funciones y operaciones de control, derivadas de DTEi y ODIN, que pueden ser utilizadas por los objetos de red y por cualquier elemento intermedio (e.g. switches y gateways) para direccionar las entidades de red en base a su identidad, en vez de utilizar identificadores fijos, como direcciones IP o MAC. Aunque nuestra instanciación de IBCP está especialmente diseñada para trabajar conjuntamente con soluciones SDN (Software Defined Networking), se puede integrar con cualquier otro plano de control que permita la intercepción y manipulación de sus operaciones de control.

Como hemos visto, mediante estos bloques funcionales, la arquitectura que proponemos en esta tesis permite la identificación de objetos de red de forma flexible, escalable, inequívoca y sin ambigüedades. Además, nuestra arquitectura permite a los objetos de red reaccionar de forma dinámica a los cambios en la red, operando de forma efectiva en entornos cambiantes, incluso cuando las redes subyacentes están fuera de su control. Por último, cabe destacar que esta arquitectura cubre las funciones necesarias que deben ser provistas por cualquier arquitectura de red dirigida hacia FI, habiendo además considerado en el diseño de los bloques funcionales propuestos en esta tesis, la relación de dichas funciones con los requisitos extraídos de nuevos servicios y aplicaciones de red.

Para finalizar, hemos validado las distintas soluciones diseñadas por medio de modelos analíticos y prototipos implementados sobre plataformas reales, evaluando sus funcionalidades, seguridad, factibilidad y rendimiento. Además, hemos integrado las funciones que ofrece nuestra arquitectura con otras arquitecturas de red, particularmente HIMALIS y MOFI, siendo ambas arquitecturas destacadas para FI que a su vez representan dos visiones diferentes de la evolución de la red. Dichas integraciones nos han permitido demostrar la utilidad de nuestros diseños, estableciendo interrelaciones sólidas e identificando tópicos abiertos de investigación para el futuro.

Agradecimientos

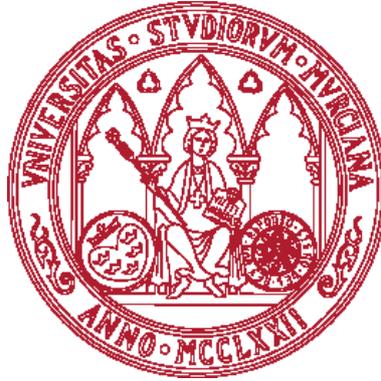
Dedico esta página a agradecer a toda la gente e instituciones que, de alguna forma, han ayudado, soportado, influenciado, y contribuido en el desarrollo de esta tesis.

Ante todo doy las gracias a Antonio, mi director de tesis, por confiar en mis habilidades durante todos estos días, cuando incluso yo desconfiaba de mi mismo. Sin su apoyo esto no habría sido posible. Su visión única del trabajo duro, *apretando pero no ahogando*, ha sido mi inspiración continua.

Aprovecho la oportunidad para agradecer a Ved P. Kafle su inestimable colaboración, así como a Hiroaki Harai por haberme acogido en el *Network Architecture Laboratory* del *Photonic Network Research Institute* del NICT, Tokyo, Japón, durante mi estancia de tres meses en el verano de 2014. Tengo gratas memorias de ellos y del resto del equipo del NICT, que me hicieron sentir como en casa. Además, estoy agradecido a Serge Fdida y al equipo del *Laboratoire d'Informatique de Paris 6* (LIP6) de la *Université Pierre et Marie Curie* (UPMC), Paris, Francia, por acogerme también durante otra estancia de tres meses en la primavera de 2015. También guardo gratas memorias de ellos.

Por supuesto, el trabajo llevado a cabo en esta tesis no podría haberse realizado sin el soporte económico de instituciones y programas de financiación. Éstas son, sin orden especial, el Ministerio de Educación de España, el cual ha financiado la beca que me ha permitido desarrollar esta tesis bajo el programa FPU (AP2010-0576); la Fundación Séneca, que ha apoyado parte de nuestros esfuerzos de investigación por medio del Programa para Grupos de Excelencia Investigadora (04552/GERM/06); el Ministerio de Economía y Competitividad de España, que también ha financiado nuestros esfuerzos de investigación a través del programa Explora (TIN2013-50477-EXP); y a la Comisión Europea, que ha financiado parte de nuestro trabajo por medio de varios proyectos dentro del 7th Programa Marco, entre los que destacan GN3 de GÉANT, OpenLab (INFO-ICT-287581) y SmartFIRE (611165).

Finalmente, estoy especialmente agradecido a mis padres y al resto de mi familia por aceptar el poco contacto que hemos tenido a lo largo de este tiempo, así como el ánimo que me han aportado durante toda mi vida. Entre ellos tengo un lugar especial para Laura, mi pareja, mi compañera eterna, quien ha tenido que soportar todos mis momentos duros, y es la base de mi felicidad. Gracias Laura.



University of Murcia
Faculty of Computer Science

**A NOVEL, IDENTITY-BASED
NETWORK ARCHITECTURE
FOR THE FUTURE INTERNET**

PHD THESIS

Author:

Pedro Martínez Juliá

Thesis Advisor:

Dr. Antonio F. Skármeta Gómez

Murcia, September 2015

Abstract

The inception of the Internet had the objective to enable distant computers to communicate each other without monopolizing the underlying communication infrastructure, effectively increasing the reliability and fault tolerance of networks, and therefore being a huge advance in digital communications. At that time, the Internet was targeting a small community but today it has become a worldwide network and the central infrastructure that supports our digital lives. The Internet is the virtual place where we meet, where we store our information, where we go to find information of any kind, the infrastructure to which one plugs to, and, inevitably, the bottleneck of our communications.

Nowadays, more and more people, digital artifacts, and objects of many kinds (things) are connected to the Internet, following the concept of *everything continuously connected*. But the original design of the Internet did not contemplate this evolution, so several problems have arisen, both at networking and internetworking levels. Moreover, those elements represent the critical mass of the current and future networks, requiring new functionality that is hard to achieve with the current network model, used in both access networks and the Internet as a whole. This has led to a revision of the Internet architecture. A new and huge area of services, devices, and information is waiting for the Internet to answer to their requirements, so increasing the pressure for a change in the network model and an evolution towards the Future Internet (FI).

One of the answers from the research community to overcome part of these problems has been to separate identifiers and locators (also called locator/identifier split). But this scheme is not complete enough to cover the requirements to support mobile elements in the same way static elements (hosts) are supported. To address this challenge, complete architecture proposals, the so-called *clean-slate* architectures, aim to rework the architecture from the basement. Instead of patching the problems, these proposals try to prevent them from manifesting themselves in the first place. Although most of them offer promising capabilities, the fact that they require to jettison the current architecture before they can be deployed reduces their actual impact. Moreover, there is no proposal that provides an integral solution that reflects in the network the solicited interactions of persons

and objects. For instance, supporting the real objects that participate into communications (people, services, devices, etc.) while at the same time protecting information about them and their contexts has no solution yet.

With this thesis we try to address those problems by introducing the concept of *identity*, seen as a set of attributes, to the middle layers of the network. We propose to use an identity layer that enables network entities (objects) to control their involvement in the digital world, creating a view in the communication sphere which is singular, unique, and optimized for every network entity, thus building an *identity plane* that allows entities to address each other in an *identity-to-identity* approach.

Some claim that allowing the network or services to have a better understanding (all or parts of) the identities of the entities involved in communications, they will be losing their privacy. Our proposal does the opposite: by controlling and setting boundaries on what entities, services, and networks can access, privacy is enhanced. Our proposal deals not only with horizontal privacy, where you protect data across services, but also cover vertical privacy, where your privacy is handled between the network and services or within the same service. This increases the overall security and privacy of the system, and the user is empowered with a higher level of control, supported by the necessary mechanisms to enforce their preferences.

Beginning with a new *identity-to-identity* network model, our proposal for the FI is to create a scalable digital communication architecture that mirrors the structure of the real world: people talk to people, objects, objects between themselves and, in general, entities denoted by digital identities communicating with each other. We take into account that, when entities communicate, we need to consider all aspects of communications, such as “to who I am talking to”, the trust I have, or even the location where this is taking place. These attributes affect security and privacy and are related to the identity of the participants.

From an analytical point of view, the Internet is built as a global graph of interconnected elements and networks, where each network is built as a graph, each element (node) is a vertex of the graph connected with other elements by edges (network links), and end nodes are connected to just a single intermediate network element. This view has been used as a premise to define the current network and transport protocols, without considering any dynamism on the structure. However, such view is no longer valid. The multi-homing feature that is widely present today has enabled end nodes to be attached to two or more different vertices of the network graph. Moreover, the rise in mobile devices connected to the Internet has broken the static view of the network graph. Mobile devices change their point of attachment to the network very often and, even worse, during the handover processes, they may have two simultaneous points of attachment to the network. This leads

us to the definition of *dynamic multihoming* concept, with which we represent that both multihoming and mobility are two aspects of the same thing, exposing the same problems and requirements.

During our research we also delve into the discussion about the separation of identifiers and locators. However, we extract the crucial requirements behind such approach and generalize the target as *decoupling identification and location processes*. The objective is the same, to separate node identification from host addressing in the network, but provides more emphasis in the identification process itself, regardless of the underlying mechanisms to address information among network entities. Moreover, this mechanism also intends to resolve the general routing scalability problem found in the current Internet. By providing a separate mechanism to identify network nodes without relying on underlying identifiers or locators (e.g. IP addresses), those underlying identifiers/locators may be structured to improve routing table scalability by, for example, geographically distributing the network addresses.

It is clear that the enabling the network with a separated and coherent identification function based on identities imposes a new crucial requirement — the mapping from such identities to the particular identifiers and locators used by the entities they represent. This challenge has a high degree of parallelism to the mapping performed in typical solutions for separating identifiers from locators. However, none of them has been widely accepted by the network community. Therefore, in this thesis we also propose to enhance the network architecture with a new, identity-based, highly scalable, and efficient mapping system to make effective the decoupling of identification and location processes.

The need for a better discovery mechanism for the FI has also influenced our design. Today, most search and discovery operations are usually performed by humans through well-known search engines. It is not well suited for network layer functions. Moreover, a proper discovery approach for the FI should provide mechanisms to find a network object of any kind (content, device, person, etc.) from a vague or near complete description. A plain identifier or name is not enough since it cannot represent the actual properties of a network object. This is exacerbated by the increasing amount of data, information items, people, machines, and, in general, a large number of *network objects* connected to the Internet. Current mechanisms for the description, search, and discovery of network objects are based on their names, generally being domain names and typically being full-qualified domain names. When the name is unknown, an entity has to contact to a third party that holds a list of available network objects and perform a search for some criteria, which most times involves humans, denoting a clear lack in the functionality offered by current

network mechanisms. Therefore, our proposal includes an identification function into the *identity plane* with discovery and search capabilities.

Targeting the provision of proper functional blocks with the necessary functions stated so far, and with the basis established in the ability of evolution as well as the decoupling of identification and location, we define the main objectives of this thesis as follows:

To design, analyze, and validate a highly modularized and evolvable network architecture for the Future Internet that abstracts the communication endpoints from hosts to arbitrary network objects by using their identities, seen as attribute sets, as the mechanism to enable them to dynamically discover, address, and communicate each other in an unequivocal, secure, and private way over an heterogeneous and changing environment.

Within this definition we highly embrace the dynamism of the network, as it is a key aspect of current trends in both network elements (devices and infrastructure) and network services. We also remark the necessity of abstracting network endpoints, achieving flexibility and granularity, and we consider that identities, been conceived as attribute sets (as defined by ITU-T X.1250), are the best candidates to identify unequivocally. To achieve such objectives we designed a set of functional blocks that are able to work together, forming a complete network architecture, or separately to provide some additional qualities to other architectures. These components are the Domain Trusted Entity Infrastructure (DTEi), the Identity-Based Network Protocol (IBNP), the Ontology-based Distributed Information Network (ODIN), and the Identity-Based Control Plane (IBCP). Although these functional blocks are closely related to each other, as well as coupled to some extent, they retain their independence so each of them is able to provide its target network function without needing to fully deploy any other.

The DTEi is a network infrastructure that enables network objects to reach each other in terms of their identities. Different DTEi nodes are deployed-on and managed-by each network or administrative domain. The objects pertaining to each domain have a trust relation with their DTEi node, so they intermediate in control operations to ensure they are addressed in a secure manner under the policies established by each network object, including the secure and private management of their communication sessions. In order to achieve a totally decentralized system while unlinking it from the underlying network, the infrastructure is built as an overlay network, which is formed by all DTEi nodes.

To complement the DTEi, we designed the IBNP. This functional block consists on a control plane protocol and a simple data plane protocol. They are designed to enable

network objects to talk to each other in terms of their identities, so they do not need fixed identifiers or locators (e.g. IP addresses) to establish their communication sessions, although identifiers or locators will be used in the underlying network once the session is established. These protocols are designed to interact with the DTEi but they can also be used by network objects to manage their sessions with other objects without the intermediation of the DTEi. The only difference would be the instantiation of the protocol into the underlying network. Being identity-based means that these protocols provide implicit and transparent support for changing and heterogeneous contexts and environments, including the support for different underlying technologies as well as the multiplicity of underlying networks (multi-homing) and the dynamicity of the relations of a network object with them (mobility), which we defined here as *dynamic multihoming*.

We then designed ODIN to empower the other components with identity-based search and discovery functions. ODIN is an infrastructure and mechanism to enable network objects to find each other in abstract terms, independently of their current location or dynamical state, while increasing the granularity of the endpoints. This is achieved by allowing network objects to find other objects by providing a *partial identity*, which consists in an incomplete set of attributes about the target identity. It also ensures the privacy of network objects by preventing the traceability of network operations, and achieving the overall security of the system without requiring external mechanisms. This component includes an ontology to describe the properties of network objects as well as exposing the capabilities they *offer*. It is also built on top of an overlay network formed by all ODIN nodes. Each node can be hosted together with the DTEi node, so they can securely exchange the attributes of the identities they manage separately.

Integrating the functions offered by the DTEi, IBNP, and ODIN we built a complete architecture that enlarges the *identity* concept in the network, allowing objects to play different roles depending on their particular intentions. This also ended up with a new communication model that extends the endpoint concept, which today resides in the *host*, by increasing the granularity degree of the network in order to allow the existence of network objects residing into different levels of abstraction, represented by their identities, and which can be embedded (instantiated) onto heterogeneous underlying networks. However, in order to bring these qualities to other network architectures we required another component that attaches to the control plane of the network, so we designed IBCP.

The objective of IBCP is to encapsulate the identity-based functions and offer them as a complement of other network architecture without requiring to change its operation model. It offers a set of control functions, derived from the operations provided by the DTEi and ODIN, that can be used by network objects as well as network intermediate

elements (e.g. switches and gateways) to address network entities by their identity, instead of fixed identifiers like IP or MAC addresses. Although our instantiation of IBCP is especially designed to work with Software Defined Networking (SDN) approaches, it could be integrated with any other control plane that allows the interception and manipulation of control operations.

With these functional blocks, the architecture we propose with this thesis enables flexible and scalable identification of network objects. It is based around the *identity* concept and targeting the provision of unambiguous and unequivocal identification of network objects. Moreover, it allows network objects to dynamically react to the changes in the network, for the architecture to effectively work on changing environments, even though the underlying networks are outside its control. In particular, the architecture we defined also covers the necessary functions that should be provided by network architectures targeting the FI. During the design of the functional blocks we have also considered the relation of such functions with the requirements extracted from new network and application services.

Finally, we validated the designed approaches by means of analytical models and prototype implementations, evaluating their functionality, security, feasibility and performance. Moreover, we have integrated the functions provided by the architecture we designed with other network architectures, particularly HIMALIS and MOFI, which are outstanding architectures for the FI, representing two different visions of the evolution of the network. These integrations allowed us to demonstrate the usefulness of our research results, establishing strong interrelations and identifying open research topics for the future.

Acknowledgements

This page is dedicated to thank all people and institutions that have somehow helped, supported, influenced, and contributed to the development of this thesis.

First of all, I am grateful to Antonio, my supervisor, for being confident on my abilities during all these years, when even I had not much confidence in myself. Without his support this would not have been possible. His unique understanding of hard work, *tightening* but not *choking*, has been my continuous inspiration.

I take this opportunity to also thank Ved P. Kafle for his invaluable collaboration as well as Hiroaki Harai for having hosted me at the Network Architecture Laboratory of the Photonic Network Research Institute of NICT, Tokyo, Japan, during my three-month stay in the summer of 2014. I have great memories from them and the rest of the research team at NICT, who made me feel at home. Moreover, I am grateful to Serge Fdida and the team at the Laboratoire d'Informatique de Paris 6 (LIP6) of the Université Pierre et Marie Curie (UPMC), Paris, France, for also hosting me during my other three-month stay in the spring of 2015. I will also keep great memories of them.

Of course, the work carried out in this thesis could not be done without the support given by funding institutions and programs. They are, in no special order, the Ministry of Education of Spain, which particularly granted me for the development of this thesis under the FPU program (grant AP2010-0576); the Séneca Foundation, which supported part of our research efforts through the Program for Research Groups of Excellence (grant 04552/GERM/06); the Ministry of Economy and Competitiveness of Spain, which also supported our research efforts through the Explora program (grant TIN2013-50477-EXP); and of course the European Commission, which has supported our research work through many projects under the Seventh Framework Programme, outstanding GÉANT's GN3 project, OpenLab (grant INFISO-ICT-287581) and SmartFIRE (grant 611165).

Finally, I am especially grateful to my parents and the rest of my family for their understanding in the little contact we had along this time as well as the encouragement they have given me throughout my life. Among them, I have a special place for Laura, my partner, my eternal companion, who has had to endure all my hard moments, and the foundation of my happiness. Thank you Laura.

Contents

List of Figures	xxviii
List of Tables	xxix
1 Introduction	1
1.1 Contextualization	1
1.1.1 Functions of Network Architectures	3
1.1.2 Relation of Network Architectures and Services	5
1.1.3 Decoupling Identification and Location	7
1.2 Motivation and Problem Statement	9
1.3 Objectives of This Thesis	12
1.4 Main Contributions	16
1.5 Related Publications	17
1.6 Thesis Structure	21
2 Background	23
2.1 State of the Art	23
2.1.1 Evolution and Fixes of Current IP Architecture	24
2.1.2 Separation of Identifier and Locator	25
2.1.3 Overlay Network Architectures	27
2.1.4 Clean-Slate Architectures	28
2.1.5 Information Centric Networking (ICN) Architectures	30
2.1.6 Architectures for Search and Discovery	32
2.2 Gap Analysis	34
2.3 Identity Management, Discovery, Authentication and Authorization	38
2.4 Towards an Identity Plane Using Attribute-Based Discovery	41
3 Initial Steps: An Overlay Approach	45
3.1 Introduction	46
3.2 Initial Architecture Proposal	47
3.2.1 Identity and Identifier	49
3.2.2 Authentication and Integrity	49
3.2.3 Overlay Network Routing	50

CONTENTS

3.2.4	Underlying Network Routing	50
3.2.5	Overlay/Underlying Network Convergence	51
3.2.6	Messages	51
3.3	Evaluation and Results	52
3.3.1	Evaluation Network Topologies	53
3.3.2	Original Algorithm	55
3.3.3	First Optimization	58
3.3.4	Second Optimization	60
3.3.5	Third Optimization	61
3.3.6	Fourth Optimization	63
3.3.7	Algorithm and Optimizations Comparison	65
3.4	Conclusions	71
4	Digging Deeper: Evolving Functional Blocks	73
4.1	Introduction	74
4.2	Domain Trusted Entity Infrastructure	76
4.2.1	Architecture Overview	77
4.2.2	Identities and Identifiers	79
4.2.3	Authentication and Integrity	80
4.2.4	Convergence of the Overlay and Underlying Networks	80
4.2.5	Security of the Infrastructure	81
4.2.6	Provided Security	82
4.3	Identity-Based Network Protocol	83
4.3.1	Network Model Shift	83
4.3.2	Architecture Overview	84
4.3.3	Protocol Description	86
4.4	Network Object Discovery	90
4.4.1	Objective	90
4.4.2	Architecture Overview	91
4.5	Ontology for Network Object Discovery	94
4.5.1	Requirements and Design Principles	95
4.5.2	Ontology-based Discovery Protocol	97
4.6	Identity-Based Control Plane	101
4.6.1	Architecture Overview	103
4.7	Security Analysis	107
4.7.1	Analysis of IBNP	108
4.7.2	Analysis of IBCP	110
4.8	Performance Evaluation	111
4.8.1	Evaluation of IBNP	111
4.8.2	Evaluation of ODIN	119
4.8.3	Evaluation of the Ontology	124
4.8.4	Evaluation of IBCP	134
4.9	Conclusions	142

5	Final Architecture: Beyond the Separation of Identifiers and Locators	147
5.1	Introduction	148
5.2	Identity-Based Network Architecture (INA)	150
5.2.1	Communication Sessions	152
5.2.2	Integrated Node Discovery	152
5.2.3	Mobility and Multi-homing Support	155
5.2.4	Interfaces for New and Legacy Applications	156
5.2.5	Integrated Security	156
5.2.6	Main Benefits of Using Identities	158
5.2.7	Heterogeneous Underlying Network	159
5.3	Mapping Identities to Communication Sessions	160
5.3.1	Independent Interactions	162
5.3.2	Delegated Negotiations	163
5.4	Mapping Sessions to the Underlying Network	164
5.5	Architecture Analysis	166
5.5.1	Application Scenario	166
5.5.2	Comparison with Other Approaches	170
5.5.3	Performance Analysis	172
5.5.4	Experimentation Results	174
5.5.5	Evaluation of the Instantiation on Top of CCN	179
5.5.6	Evaluation of the Integration with HIMALIS	184
5.5.7	Discussion	189
5.6	Conclusions	190
6	Integration with Other Network Architectures	193
6.1	Introduction	194
6.2	Integration with the HIMALIS Network Architecture	195
6.2.1	Architecture Overview	198
6.2.2	Integration Proposal	200
6.2.3	Message Exchanges	203
6.2.4	Attaching the Identity-Based Control Plane.	207
6.2.5	Evaluation of the Integrated Architecture	208
6.2.6	Evaluation of the Integrated Control Plane	210
6.2.7	Experimentation Framework and Testbed	218
6.3	Integration with the MOFI Network Architecture	225
6.3.1	Architecture Overview	226
6.3.2	Secure Identification (SEID) over MOFI	227
6.3.3	Analysis	230
6.4	Conclusions	233

7	Conclusions and Future Work	237
7.1	Main Contributions	238
7.1.1	Design Approaches to Meet the Requirements	238
7.1.2	Features and Functional Blocks	240
7.1.3	Integration with Other Architecture Proposals	242
7.2	Future Work	243
7.2.1	Continuous Integration with New Network Architectures	244
7.2.2	Tighten Relations with Software Defined Networking	245
7.2.3	Evolution Towards the Internet of Everything	246
	Bibliography	248
A	Side Extension: Gateway Overlay Network	265
A.1	Introduction	265
A.2	Related Work	267
A.3	Proposed Approach	267
A.3.1	Security	269
A.3.2	Scalability, Performance, and Efficiency	270
A.3.3	Mobility by Default and Multihoming	270
A.3.4	Generic Interconnection Mechanism	271
A.4	Instantiation Example	271
A.5	Scalability Analysis	274
A.5.1	Classical Interconnection Approach	275
A.5.2	Proposed Interconnection Approach	275
A.5.3	Results	277
A.6	Conclusions	279
B	The GAIA Experimentation Infrastructure	281
B.1	Infrastructure Description	281
C	Schemas for the Network Object Discovery Functions	285
D	Protocols Represented in HLPSP for AVISPA	291
E	Source Code for the Formal Verification of HIMALIS with Alloy	297
F	Autonomic Network Management	303
F.1	Introduction	303
F.2	Future Identity-Based Network Management	305
F.3	Current Autonomic Management Proposals	307
F.4	The Proposed Solution	308
F.4.1	Autonomic Network Manager	310
F.5	Evaluation and Results	313
F.6	Conclusions	316

List of Figures

1.1	Hype cycle for emerging technologies (Gartner, 2014).	2
1.2	Protocol pile of the Internet.	10
1.3	Overview of the networking objective.	13
1.4	Restructured and simplified network layers.	16
2.1	Analysis of architecture proposals for the Future Internet.	36
3.1	First message exchange.	48
3.2	Route shortcut.	50
3.3	First evaluation network topology.	54
3.4	Second evaluation network topology.	55
3.5	Third evaluation network topology.	56
3.6	Average penalty hops comparison.	65
3.7	Maximum penalty hops comparison.	66
3.8	Average underlying hops comparison.	66
3.9	Maximum underlying hops comparison.	67
3.10	Average overlay hops comparison.	68
3.11	Maximum overlay hops comparison.	68
3.12	Median underlying hops comparison.	69
3.13	Median penalty hops comparison.	69
3.14	Accuracy of the algorithms.	70
3.15	Example of maximum length path.	72
4.1	Global view of the objective and the functional blocks.	75
4.2	Overview of the general architecture.	77
4.3	Architecture instantiation on top of Chord.	78
4.4	Identity-Based Network Protocol.	87
4.5	Overview of the lookup operation in ODIN.	92
4.6	Overview of the proposed ontology.	99
4.7	Overview of the Identity-Based Control Plane (IBCP).	104
4.8	Mobility management: handover message exchanges.	107
4.9	Protocol represented in <i>Alice and Bob</i> notation.	109
4.10	Protocol for mobility support in <i>Alice and Bob</i> notation.	110

LIST OF FIGURES

4.11	Bob's XRDS document.	113
4.12	Authentication request and response.	114
4.13	XRDS request and response.	115
4.14	Alice's first data message exchange with Bob.	115
4.15	Validation request and response.	116
4.16	Performance comparison (milliseconds per message).	117
4.17	Performance comparison of our protocol over CCN with raw CCN.	117
4.18	Performance comparison of our protocol over XMPP with raw XMPP.	118
4.19	Comparison of the overhead of our protocol on CCN and XMPP.	118
4.20	CDF of routing table size of main nodes.	121
4.21	CDF of routing table size of nodes from registered objects.	122
4.22	CDF of time spent in object registration.	123
4.23	CDF of time spent in object lookups.	124
4.24	Lookup precision of the overlay network with 5375 nodes.	125
4.25	Lookup precision of the overlay network with 11750 nodes.	126
4.26	Lookup precision of the overlay network with 23500 nodes.	127
4.27	Lookup precision of the overlay network with 35250 nodes.	128
4.28	Description of the experiment with the proposed ontology.	130
4.29	CDF of the time spent in each discovery operation for DNS-SD and ODP.	133
4.30	Scenarios used to build the mobility analysis models for the comparison of HIP, LISP, and our approach (THIS).	137
4.31	Analysis results, variation of α	140
4.32	Analysis results, variation of N_h	141
4.33	Analysis results, variation of N_d	142
4.34	Analysis results, variation of $T_{GW \rightarrow GW}$, $T_{DTE \rightarrow GW}$, $T_{GW \rightarrow MS}$, and $T_{GW \rightarrow RVS}$	143
5.1	Overview of the final architecture.	151
5.2	Process/Identity Discovery.	154
5.3	Starting a session without involving impersonation elements.	157
5.4	Starting a session involving impersonation elements.	158
5.5	Native and adaptation layers with mappings.	165
5.6	Application Scenario: Mobility and granularity in a cloud environment.	166
5.7	Performance cost results for the variation of the number of nodes in the overlay network forming the DHT.	174
5.8	Session start time for different number of attributes.	176
5.9	Total time elapsed in the announcement and the two consecutive session starts.	178
5.10	Fraction of the discoveries that needed a retry.	179
5.11	Relation between the number of peers in the routing table and the discovery time.	180
5.12	Experimentation environment.	181
5.13	Comparison of the overhead evolution when increasing the message exchanges for INA running on top of CCN and raw CCN.	182

5.14	Time spent on the two-way test on each step.	183
5.15	Overhead evolution for stacked steps.	184
5.16	Average time evolution in milliseconds (top) and message loss percentage (bottom).	185
5.17	Excerpt of the evolution of message loss for the different tests that show the turning point where the message loss leaves the 0%, that is around 400 concurrent messages.	186
5.18	Evolution of the difference of the average exchange time of INA-HIM and IP.	187
5.19	Evolution of the difference between the message loss of INA-HIM and IP, and between our architecture during a mobility event and IP.	188
5.20	Evolution of the solution behavior as the mobile node moves through all domains while receiving messages at a rate of 500 msgs/s.	189
6.1	HIMALIS network architecture overview.	199
6.2	HIMALIS network architecture overview (updated).	200
6.3	Integrated architecture.	201
6.4	Integrated architecture remarking mobility and identities.	202
6.5	Messages exchanged to start a session.	204
6.6	Messages exchanged after a movement (handover).	206
6.7	Integrated control plane with HIMALIS.	207
6.8	Evaluation environment for our architecture integrated with HIMALIS.	210
6.9	General topology of the evaluation network.	212
6.10	Network topology with the HIMALIS elements.	213
6.11	Network topology with the controller.	214
6.12	Streaming interleave time for scenario A, hard handover.	216
6.13	Streaming Interleave time for scenario A, soft handover.	217
6.14	Handover time for scenario A, hard handover.	218
6.15	Handover time for scenario A, soft handover.	219
6.16	Handover time for scenario B, hard handover.	220
6.17	Handover time for scenario B, soft handover.	221
6.18	Handover time for scenario C, hard handover.	222
6.19	Handover time for scenario C, soft handover.	223
6.20	Experimentation testbed.	224
6.21	Session information obtained from the viewer component.	224
6.22	Overview of global/local communications in MOFI.	227
6.23	Instantiation of our architecture for secure identification.	228
6.24	Overview of the integrated DTEi and message exchanges.	229
6.25	Total cost of MOFI and MOFI+SEID for the variation of $N_{\text{Host/AR}}$	233
6.26	Total cost of MOFI and MOFI+SEID for the variation of N_{AR}	234
A.1	Overview of the general networking operation.	269
A.2	Protocol overview: Initiator communicates with Target.	272
A.3	Analysis results, variation of α	277

LIST OF FIGURES

A.4	Analysis results, variation of N_p	278
A.5	Analysis results, variation of N_a	279
A.6	Analysis results, variation of N_{GW}	280
B.1	GAIA extended infrastructure with the living labs and deployments.	282
C.1	Ontology schema in Turtle.	286
C.2	Description of network object 1, color printer.	287
C.3	Description of network object 2, information item.	287
C.4	Description of network object 3, black and white printer.	287
C.5	Description of network object 4, color scanner.	288
C.6	Query (SPARQL) to retrieve the information object description.	288
C.7	Query (SPARQL) to retrieve the printer object description.	288
C.8	Query response in JSON of the information object.	289
C.9	Query response in JSON of the printer object.	289
C.10	Protocol schema for Avro in JSON.	289
F.1	Overview of the integrated architecture.	309
F.2	Architecture of the management component, the ANM.	311
F.3	Example scenario: Interaction with the AutoBAHN service.	313
F.4	Performance overview: Average and total processing times.	315
F.5	Scalability overview.	316

List of Tables

2.1	Summary of target qualities for the final architecture.	44
3.1	Routing evaluation results, original algorithm, first topology.	57
3.2	Routing evaluation results, original algorithm, second topology.	57
3.3	Routing evaluation results, original algorithm, third topology.	58
3.4	Routing evaluation results, first optimization, first topology.	58
3.5	Routing evaluation results, first optimization, second topology.	59
3.6	Routing evaluation results, first optimization, third topology.	59
3.7	Routing evaluation results, second optimization, first topology.	60
3.8	Routing evaluation results, second optimization, second topology.	61
3.9	Routing evaluation results, second optimization, third topology.	61
3.10	Routing evaluation results, third optimization, first topology.	62
3.11	Routing evaluation results, third optimization, second topology.	62
3.12	Routing evaluation results, third optimization, third topology.	63
3.13	Routing evaluation results, fourth optimization, first topology.	63
3.14	Routing evaluation results, fourth optimization, second topology.	64
3.15	Routing evaluation results, fourth optimization, third topology.	64
3.16	Penalty percentile table.	70
4.1	Packets of the single iteration experiment with ODP.	132
4.2	Packets of the single iteration experiment with DNS-SD.	132
4.3	Parameter values and ranges.	140
5.1	Comparison of solutions for decoupling identification and location.	171
6.1	Simulation parameter values.	214
7.1	Summary of requirements and how they are met.	239
7.2	Summary of features and functional blocks of the final architecture.	241
A.1	Parameter values and ranges.	277
F.1	Performance results (milliseconds).	314

LIST OF TABLES

Chapter 1

Introduction

The present document describes and discusses the work carried out and the results obtained during the research towards the design of an architecture for the Future Internet (FI) that moves network endpoints from hosts and IP addresses to identities, seen as attribute sets, thus raiding the role in the network of the actual entities behind communications. The general objective is to go beyond the simple separation of locators and identifiers in order to provide the necessary functions to build a flexible, scalable, and secure infrastructure to base communications in future networks.

In this chapter we give a brief contextualization of the topic, outlining the global trends in the digital society and the consequent impact in the network architectures of the future and the evolution of the Internet. Then we introduce the functions of network architectures to establish the boundaries of our work, its relation to both network and application services, and the necessity of decoupling identification and location functions, which has a broader objective than the mere separation of identifiers and locators.

After that we discuss the motivation of this thesis, outlining the problems found in the current Internet that we want to resolve, and describing the specific objectives we want reach during the development of the thesis. Finally, we describe the main contributions of this thesis to the state of the art, the publications achieved in the wake of it, and the structure of the remainder of this document.

1.1 Contextualization

The Internet emerged as a support infrastructure to allow people communicate freely when computers were becoming the preferred tool to handle information about people, public, enterprise and medical records, as well as physical objects (*things*). Suddenly there was no

1. Introduction

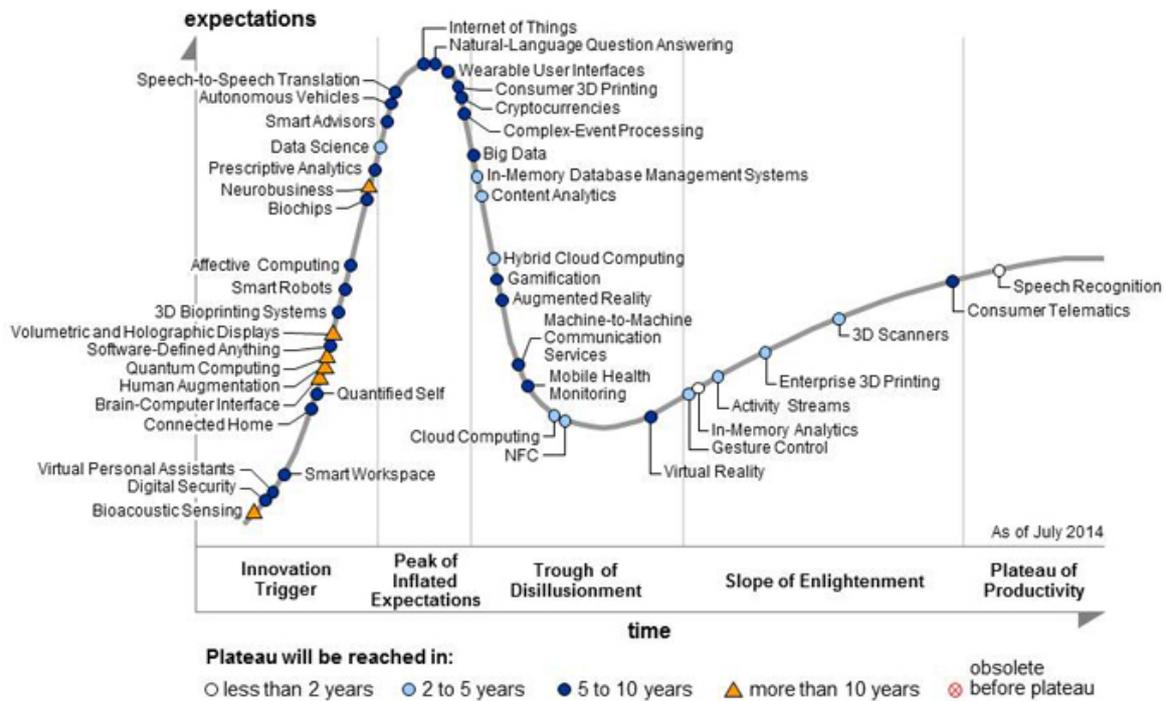


Figure 1.1: Hype cycle for emerging technologies (Gartner, 2014).

barrier to what type of information one could find on this Internet. The Internet became the ruler of all business, social, educational, cultural and global issues. While we see this Internet as an abstract entity to where one *plugs-in*, means are lacking to scale its services to the billions of users and digital entities the Internet currently and in the near future will host.

According to Gartner’s hype cycle for emerging technologies, as shown in Figure 1.1 ¹, *Digital Business* “is the first post-nexus stage on the road map and focuses on the convergence of people, business and things”. This builds the basement of the integration of people into the network, together and at the same level than any other object or device.

Therefore, all entities connected to the network become *network objects* that should be able to communicate each other by means of their *digital identity* and in a secure way. In terms of the Internet itself and the services it brings to the society, this trend is also translated into the need for more social content and associated services, more integration of devices in our daily lives, more communication, more bandwidth, and more power.

With the present thesis we propose to build an architecture to step forward into such direction, abstracting and moving network endpoints from the typical machine, known as

¹<http://www.gartner.com/newsroom/id/2819918>

host, to an higher and broader level, the identity. This means that networks can be formed by any kind of *network object*, whatever its nature is. This also means that endpoints are more granular, meaning that network endpoints are no more associated to the size, location, or computational power of the entities that communicate, and thus reflecting the reality found in the physical and logical world.

This objective is supported by the lack to evolve of the Internet to overcome new challenges and host new services. Over time, its communication model has exposed many problems later adopted as challenges for the Future Internet (FI) as shown by [1] from the general perspective and later by [2] from the routing perspective. Therefore, the original design of the Internet lacks some key capabilities, such as the decoupling of identification and location, scalable mobility support, and, of course, the integrated security and privacy.

On the one hand, there are many patch-on solutions that try to overcome those lacks within the current Internet model, as we outline in Chapter 2, but they do not cover all requirements and as a single/integrated solution. Here is one of the starting points of this thesis, continue where other architectures have left towards an architecture that covers such requirements in a flexible, generic, and scalable manner.

On the other hand, the task of modeling the Internet around the identity concept is not easy. However, there are several initiatives and solutions that have already established strong support for identities in the application layer (e.g. the SWIFT project [3]). This thesis tries to somehow continue the work started by those initiatives, and takes into account the concepts and mechanisms they manage in order to extend them to the network.

1.1.1 Functions of Network Architectures

A network architecture is defined as a set of functional blocks that interwork together to achieve a common goal: Enabling network entities to communicate each other without needing to know the exact details of the underlying communications mechanisms and under certain pre-known conditions. Each functional block is, in turn, a set of active/passive elements and protocols that provide an independent function within the global view of the architecture.

To this extent, different network architectures will provide different sets of capabilities and communication conditions to the entities attached to them. Moreover, network architectures targeting different objectives may be able to coexist and even cooperate. However, with the network model found in the current Internet, such coexistence is limited, as all architectures should be tied to IP as the only networking protocol.

1. Introduction

Since the limitation of the current Internet to adopt different protocols is a major drawback, it has partially broken one of its most sacred design principles: *The principle of constant change* [4]. Anyway, it is clear that any network architecture for the Internet should follow a set of written or non-written design principles [5], so they can be integrated, cooperate, or, at least, have a specific level of quality.

Therefore, the functions provided by the network architecture and the resulting functional blocks will be determined by those design principles. This way, when viewed as a whole and in addition to the widely known capabilities and the specific capabilities of its objective, the functions that every network architecture targeting the Future Internet should provide are:

- Unambiguous and unequivocal identification: Whatever mechanism is used to identify entities involved in communications should be unambiguous, so two entities can be easily differentiated each other and the identification of an entity will be reliable enough to subject subsequent operations to such process. This function just targets the avoidance of typical misidentification problems and inefficiencies so it does not ensure total security but will make attackers to employ much effort in their intend to break the network.
- Unattended communication management: The existence of heterogeneous underlying network infrastructures is adding complexity to current higher layer networks. Even IP has not been able to cope with some of such complexity, leaving some operations to applications and other elements of the network. A network architecture have to completely exploit the possibilities offered by underlying infrastructures without requiring external actions from other elements, including their heterogeneity and dynamicity.
- Extended endpoint management: Although typical network architectures only consider a *host* as a network endpoint, it is no longer the case for the networked elements that require flexible connectivity, such as in dynamic distributed services and devices as found in the Internet of Things. The architecture should provide a mechanism to manage endpoints of different levels, being hosts, software, persons, etc. so any element may take part into communications.
- Entity and protocol polymorphism: Both entities involved in communications and subsequent protocols they follow should be abstracted from specificities as much as possible, having different *faces* that will be used to differentiate the interaction points

to different components and elements of the network. The same entity or protocol will have a different meaning to different elements. This function should be transversal in the network architecture, as it favors its ability to evolve and integration with other architectures.

- Supporting dynamic multihoming: The static and single point-of-attachment to the network is no longer the case in current and future networks. Latest network entities can be attached to multiple networks simultaneously, joining and leaving networks as they see appropriate. This is sometimes called *multihoming* and others *mobility* but it is essentially the same function and thus the architecture only has to support it in order to address those operations.

Although these functions are not the only ones required to build a full working network architecture, they are hardly covered by any current network architecture or proposal, as we will discuss in Chapter 2, and they are the most important to face the complexities exposed by the Future Internet. Moreover, from them we can derive the minimum set of functional blocks and protocols that will compose a network architecture.

Each of the aforementioned functions will be encapsulated into a separate functional block. Some of them will be also reflected into a specific control protocol, into some operations in the overall control protocol, and even into the main protocol of the data plane if required by the specific objectives of the architecture. Therefore, there should be control operations to perform the unambiguous identification, to hide the complexities of underlying networks, and so on.

However, the mere existence of one of such functions will affect all functional blocks. For instance, the provision of unambiguous identification functions will affect how polymorphism is managed and how dynamic multihoming will be addressed. Anyway, all functions still have to pursue the same common goal: To enable and facilitate networked entities (*network objects*) to communicate each other on heterogeneous and dynamic environments, as it is the case with new trends in communications and it has been highly envisioned for the networks of the future.

1.1.2 Relation of Network Architectures and Services

We have already stated the functions that should be provided by a network architecture in order to fit with the communication requirements of current and future networks, specially the Internet, but we have to emphasize the objective behind such requirements and the

1. Introduction

objective behind the construction of a network architecture itself. The final goal is to facilitate and favor the provision of new network and application services [6,7].

Current networks, such as the Internet, are effectively associated with the easy access to information, the way to get (and consume) contents of different kinds and from different sources, the mechanism used by people to stay connected with other people building social networks, and with the general ability to communicate. At this point, most users are not concerned about how the network works, as these services are delivered without their intervention. Therefore, network architectures must be designed from the actual requirements of their users and services.

Future network services and applications will be based on new business models [8] and they will have different requirements, especially related to the evolution of the network and its ability to be adapted to new scenarios. This has a high impact on how network architectures should be designed. For instance, new services will be created from the functions offered by other services [9], but all of them will live on the same underlying network architecture.

Therefore, the network architecture should be designed in a way that enables and favors a set of qualities highly valued, and sometimes effectively required, by new services. This can be reflected in the following new approaches and visions:

- Supporting different kinds of objects beyond the resources: Traditional objects found in networks were associated to resources of different types. However, new objects are no longer easily classified into them. For instance, new objects are found in context, interaction instances, and dynamic states. Services are being designed to support these into their service models but they are more and more requiring new functionality from the underlying network architecture.
- Enlarging the role of services and users: In a typical client/server scenario there are strict roles played by clients and servers. Clients consume what servers offer. New models are totally different. Clients can deliver services to other entities, including servers. Servers are new consumers of services from other servers or clients. In general, all elements have become entities that interact in order to instantiate a service, independently of the direction of such service is produced.
- Customization: There is a clear trend in services of the last decade towards the personalization and customization of their interactions with their consumers. Therefore, different communication instances will have different contexts and environmental constraints, which will finally result into different service instances.

This has to be taken into account while designing a network architecture in order to allow the existence of such differentiations.

- Awareness: New trends in service models are including different kinds of meta-information from the environment of their users, such as situation, location, and context. Current networks do not provide enough support for this, so applications are adding complexity to their operations. Therefore, proper support from the network architecture is required so these models are feasible or even possible.
- Security, privacy, and identity protection: Nowadays, the Internet has become the place where we live our digital lives, relying into services of different natures to support them. However, this means that we are disclosing huge amounts of information regarding our activities and our beings. This information should be secured and, when desired, kept private, out of the reach of undesired entities. This cannot be done without explicit support from the underlying network, as it has to transfer such information among distant points, crossing different elements in the middle that can break those desired properties by just watching the metadata of the communications.
- Seamless mobility: Mobility is a topic that is being considered for the top-most layers of network. Current applications and services are unnecessarily managing the location of the entities they serve in order to allow them to move to other locations. This is because of the lack of current networks to provide good support for mobility and thus has a huge impact on how network architectures should be designed.

These qualities place strong requirements to the underlying network architecture, since without explicit support it will be difficult and inefficient, driving the complexity to users and services, which is highly undesired because the limitations this imposes to the final capabilities offered by those services.

1.1.3 Decoupling Identification and Location

One of the key aspects in latest research for network architectures has been the separation of identifiers from locators [10]. This has its roots in the mixed role that current IP addresses play in the network: They act as locators, used to deliver information from point A to point B in the network. But they also act as identifiers, used by all layers from the network to the application to identify *who* is *who* in the network.

1. Introduction

The need to separate identifiers is to allow network entities to change their location, effectively changing their locator (IP address), without breaking communications. It has been really difficult to adapt current networks to support this capability. There are some proposals to provide some kind of separation of identifiers and locator for IP and other proposals to resolve the problem from the beginning, as we review in Chapter 2, but none of them resolves the actual problem with all the functions mentioned above.

This overloaded/dual semantics of IP addresses as identifier and locator makes difficult to design efficient solutions for mobility, multihoming, renumbering, and security because such solutions require the provision to dynamically change device locators at the network layer without changing identifiers at the transport and upper layers. In general, this overloading has limited the flexibility of the current Internet architecture.

Moreover, the problem is emphasized when considering the capability of an entity, network, organization, etc. to change its topological connectivity with respect to the remainder of the Internet without losing their existing sessions. The existing mechanisms, which are included as patches inside the current Internet model, are based on the renumbering of the mobile device and sometimes using a tunnel to keep using old addresses. The approaches using this mobility model restrict the dynamism because the existing network and transport sessions may be broken due to excessive handover time. Also, it introduces new security and privacy problems because the elements of a foreign network will be involved in the operation of a device. With privacy problem we mean the impossibility for an entity to decide what information reveal and to whom. This includes both operations, identity information, and content.

Some may argue that it is not necessary to separate identifiers from locators, since current IP addresses are actually identifiers in the *network graph* and the problem is that current internetwork structure and addressing name spaces are not properly formed to facilitate the resolution of the problems that appear when a network object can move between networks. However, there already exist some solutions based on the Border Gateway Protocol (BGP) that allow the same IP address to be present in different networks. The problem is that the BGP solution is not scalable and totally unfeasible to manage every object (or even every IP address) of the Internet.

Nevertheless, in this thesis we claim that the rationale behind the separation of identifiers from locators is not just to support mobility and add scalability to the network, but to provide proper support for *identification* and *location* mechanisms. Identifying an entity is different from locating it. Identification is essential to establish reliable communications while the importance of location relies in the qualities of the specific network architecture. Some architectures may not need specific locations, such as those

that route data based on flat labels [11]. However, all architectures require proper identification of network entities. Otherwise direct, entity-to-entity communications would be impossible.

Finally, although mobility is usually considered separate from multi-homing, as we stated above while introducing the functions of network architectures, both capabilities are actually the same. Mobility can be considered as a dynamic multi-homing, and it is better modeled this way when designing the network architecture. This way, with the decoupled identification functions, a device can be attached to multiple networks simultaneously or dynamically, actually moving from a network to another (handover), and its reachability or communications in general will not be altered.

1.2 Motivation and Problem Statement

On its inception, the Internet was designed as a common infrastructure to let distant computers communicate with a packet-based model that prevents the occupation of communication lines and gains robustness thanks to its routing/switching mechanisms. This view culminated in a communications model based on the conversation of two machines (peers) identified by their location dependant addresses. Today, this scenario still represents the model followed by most, if not all, communications performed through the Internet.

Moreover, the current model of the Internet has not been able to consider the evolution of network requirements, let alone adapt its behavior to them. At the end, as shown in Figure 1.2 ², the Internet has adopted the widely known hourglass-shaped architecture. This puts IP in the middle of everything, so everything works on top of IP and IP works on top of everything. Even there is support for IP-over-IP.

Although the approach of placing IP as the standard mechanism that frames every other mechanism to allow them interoperate is not bad, its intrinsic strictness has effected into its lack of evolution, and hence the ossification of the Internet. More flexibility is required in the architecture, specially in the networking layer, to cover the functions introduced in Section 1.1. Designing a mechanism to provide such flexibility is the major motivation of this thesis.

On the other hand, the original Internet design did not include security mechanisms. They have been introduced over time at different layers but, however, the basic security and privacy requirements at network layer are not met because it is not mandatory and a

²<http://www.trilogy-project.org>

1. Introduction

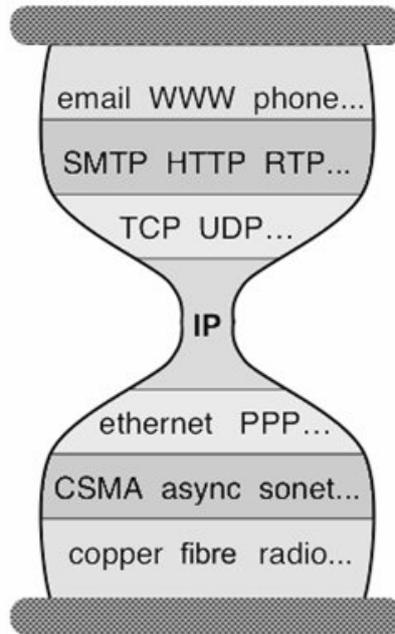


Figure 1.2: Protocol pile of the Internet.

network entity can not control its security when communicating with other network entity which do not forms part of the same security domain. Also, more and more services and applications in the Internet are turning to enforce security mechanisms, which supports integrated security as a key challenge for the Future Internet.

To this extent we have stated that the IP architecture is inherently inflexible and insecure, but they are not all the problems. Clearly following the design principles we discussed at the beginning of this chapter, we found that, derived from those two major problems, the IP architecture has been unable to support the dynamicity of current network environments, specially reflected in its lack to support dynamic multihoming, which includes mobility. Moreover, it is actually impossible to unequivocally associate an IP address with an entity, which is the reason other upper layer techniques have been used to perform the identification processes.

Regarding endpoints, the current Internet has fixed endpoints established in the so called *host*. Only two hosts can communicate to each other. Even though new advancements in computing and networks have enabled many *things* to be shaped as a host, it is still impossible to consider abstract or granular entities as hosts like, for example, *groups of hosts* or *smaller/bigger entities behind hosts* as communication endpoints. This has a high impact in the objects that can be involved in communications, avoiding

1.2 Motivation and Problem Statement

independent processes or smart low-power devices to have direct networking capabilities, and not providing enough *polymorphism* in the network.

That said, and being consistent with the aforementioned functions that network architectures should provide in order to break the current impasse and making the Internet more flexible to facilitate and encourage the creation of new services, we have identified a set of specific requirements that should be addressed by the network architecture of the Future Internet. They are listed as follows:

- (R1) *Flexibility*. As stated above, the current Internet model is ossified. This is blocking the creation of new network services and makes application services more complex, as they have to deal with network issues. The network architecture should provide enough flexibility to upper layers, so they can dynamically adapt it to specific user and service requirements.
- (R2) *Adaptability*. Current networks, and especially the Internet, are in continuous evolution and their new capabilities should be exploited without the implementation of intrusive (and sometimes weird) mechanisms that add unneeded complexity to the final service.
- (R3) *Granularity*. The *host* concept is too strict to represent current trends in networked devices, software, services, etc. We generalize them as *network objects*. Some network objects may live *within* other network objects (embedded) or even behind network objects (gateways). The network architecture should support these scenarios without requiring artificial artifacts, avoiding the added complexity.
- (R4) *Dynamicity*. The current Internet model depends on some stability and most mechanisms rely on static assumptions (e.g. gateway IP addresses and DNS entries). However, those assumptions are less and less valid, as properties of network objects and their environment are no longer stable or static. Moreover, the size and power of network objects can vary from very small/weak to very large/powerful. The future network requires to consider such dynamicity in all aspects so it can accept new style of workloads.
- (R5) *Heterogeneity*. In parallel with the granularity and dynamicity, new networks, specially underlying networks, are no longer homogeneous. There are numerous technologies involved in them and they have different properties. The Future Internet must consider such heterogeneity instead of just relying on IP and forcing them to implement IP as their default interaction interface.

1. Introduction

(R6) *Integrated Security.* Nowadays, security is a central concern in any network environment. There are several mechanisms to secure network communications, but all of them are provided separately, as an additional architecture or as an optional alteration of the protocol. However, the current network model imposes a limit to implement such security and thus the most basic security mechanisms should be integrated into the network architecture.

(R7) *Privacy.* Although it is sometimes included within the security scope, the privacy of network communications is gaining more and more attention in the current digital society. In the same manner with the integrated security, the network architecture should provide some basic mechanism to provide enough level of privacy so that upper layer services and applications can effectively achieve full privacy in their communications.

Unfortunately, as supported by the results of our initial analysis, discussed in Chapter 2, there is no architecture that fully covers these requirements. Most current architectures and proposals do not consider those requirements into their core, and they just rely into third party additions to implement them.

1.3 Objectives of This Thesis

Most architecture proposals for the Future Internet have targeted into improved performance, scalability, and pragmatism. Moreover, being influenced from and influential to the current SDN movement, almost all of them provide some support for separation of control and data planes. However, as we discuss deeply in Chapter 2, current architectures and proposals lack to some extent in all requirements stated in the previous section. Those limitations have encompassed the research and design of a new network architecture for the Future Internet and they have established the reasoning for the inception of this thesis and its main research objectives.

Although it is widely known and accepted that there is a need for a new architecture for the Internet to overcome current problems [12], designing a clean-slate architecture is not appropriate. Clean-slate architectures are based in the complete jettisoning of the current architecture before they are deployed, which implies too much effort and a *switch off* of the current services, and ends in lack of interest and support from the network research and engineering communities. Therefore, in this thesis we support the definition of modularized functional blocks that can be used together or integrated with other architectures, including the current one.

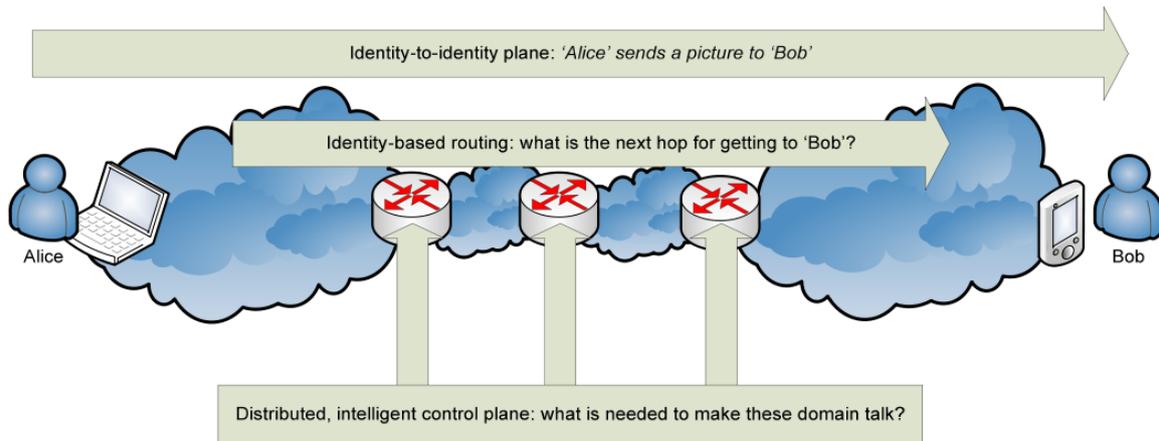


Figure 1.3: Overview of the networking objective.

On the other hand, also reflected into the objectives of the SDN movement, the need for the network to be more flexible and dynamic in response to the continuous changes in the environment and context has imposed an added difficulty to network design and implementation. Current mechanisms used to identify network flows are limited by the existence of *identifiers* that are fixed to some extent and do not comprise the specific aspects of network entities that are required to fine-tune the behavior of the network to their requirements. A new mechanism is needed to identify both network entities and their associated flows, in order to provide such flexibility to the network.

Following these major design principles, as shown in Figure 1.3, the main view of this thesis is to achieve an abstract view of the Internet that places identities of networked entities in the middle of the network to achieve broad communications controlled by *who* is the peer involved instead of *where* it is.

Evolution is a key feature that lacks in current Internet and is not considered in most proposals. Therefore, another key point that guides the design of the architecture defined in this thesis, and the functional blocks of which it is composed, is the ability of evolution of both the initial network architecture, probably tied to the current network model, and any future architecture. This will ensure the accomplishment of one of the key requirements stated above, the *adaptability* requirement, but it will be extended to also cover the architecture found in the current Internet, so it can be evolved towards the objective architecture.

The starting point of our research towards the design of our target architecture has been the separation of identifiers and locators (aka. id/loc split). It is a key design compromise that must be considered from the beginning or it will be difficult for the final design

1. Introduction

to achieve some of its key features. However, we enlarge such feature to get improved *decoupling of identification and location*, which targets a broader objective. This means that we will not just define a new network layer based on identifiers that complements an existing or new network layer based on locators. In contrast, we define complete functions for the identification of network nodes, even if there is no differentiation of *identifiers* and *locators* in the data plane of the network.

Targeting the provision of proper functional blocks with the necessary functions stated so far, and with the basis established in the ability of evolution as well as the decoupling of identification and location, we define the main objectives of this thesis as follows:

To design, analyze, and validate a highly modularized and evolvable network architecture for the Future Internet that abstracts the communication endpoints from hosts to arbitrary network objects by using their identities, seen as attribute sets, as the mechanism to enable them to dynamically discover, address, and communicate each other in an unequivocal, secure, and private way over an heterogeneous and changing environment.

Within this definition we highly embrace the dynamism of the network, as it is a key aspect of current trends in both network elements (devices and infrastructure) and network services. We also remark the necessity of abstracting network endpoints, achieving flexibility and granularity, and we consider that identities, been conceived as attribute sets (as defined by ITU-T X.1250 [13]), are the best candidates to identify unequivocally.

In particular, based on the requirements stated in the previous section, and considering both the functions of network architectures and their relations to network services, we nail down the development of this thesis towards the following set of objectives:

- (O1) To design a new communication model that extends the endpoint concept by increasing the granularity degree of the network in order to allow the existence of network objects residing into different levels of abstraction, represented by their identities, and which can be embedded (instantiated) onto heterogeneous underlying networks.
- (O2) To design an architecture that enables flexible identification of network objects around the *identity* concept, seen as a set of attributes, and targeting the provision of unambiguous and unequivocal identification of network objects.
- (O3) To enhance the designed architecture with support for network objects to dynamically adapt the resulting behavior of the network, effectively obtaining unattended

communication management, for the network to work properly on changing environments, even though the underlying networks are outside their control.

- (O4) To enable network objects to find each other in abstract terms, independently of their current location or dynamical state, for empowering the granularity and ensuring their privacy, by preventing the traceability of network operations, and achieving the overall security of the system without requiring external mechanisms.
- (O5) To enlarge the *identity* concept used in the new network architecture with support for network objects to play different roles with different features depending on their intentions or policies, providing the possibility to have different *faces* for different workloads.
- (O6) To include transparent support for changing and heterogeneous contexts, including the support for different underlying technologies, the multiplicity of underlying networks (multi-homing), and the dynamicity of the relations of a network object with them (mobility).
- (O7) To validate the designed approaches by means of analytical models and prototype implementations, evaluating their functionality, security, feasibility and performance.

In order to approach the new communications model stated in these objectives we will start by forgetting the layering defined in current networks, specially the layering defined in IP. In its place, as depicted in Figure 1.4, we will define a simple and abstract model that reforms the strict model of layers to flexible and interoperable layering around the guidance of an identity layer. The new model abstracts all mechanisms that allow the communication of data between two points and encloses them as underlying *transports*. It also abstracts all devices and applications that make use of the network as *network objects*.

Considering this new model, it is easy to cover the requirements stated in Section 1.2 by introducing new mechanisms enclosed in separated functional blocks that talk to each other through the identity plane in order to achieve the overall control of network communications and instances (sessions). Moreover, it will be an integral part of the final architecture, as it can be embedded and instantiated onto different underlying networks, depending on the specific requirements imposed by the target communications.

1. Introduction

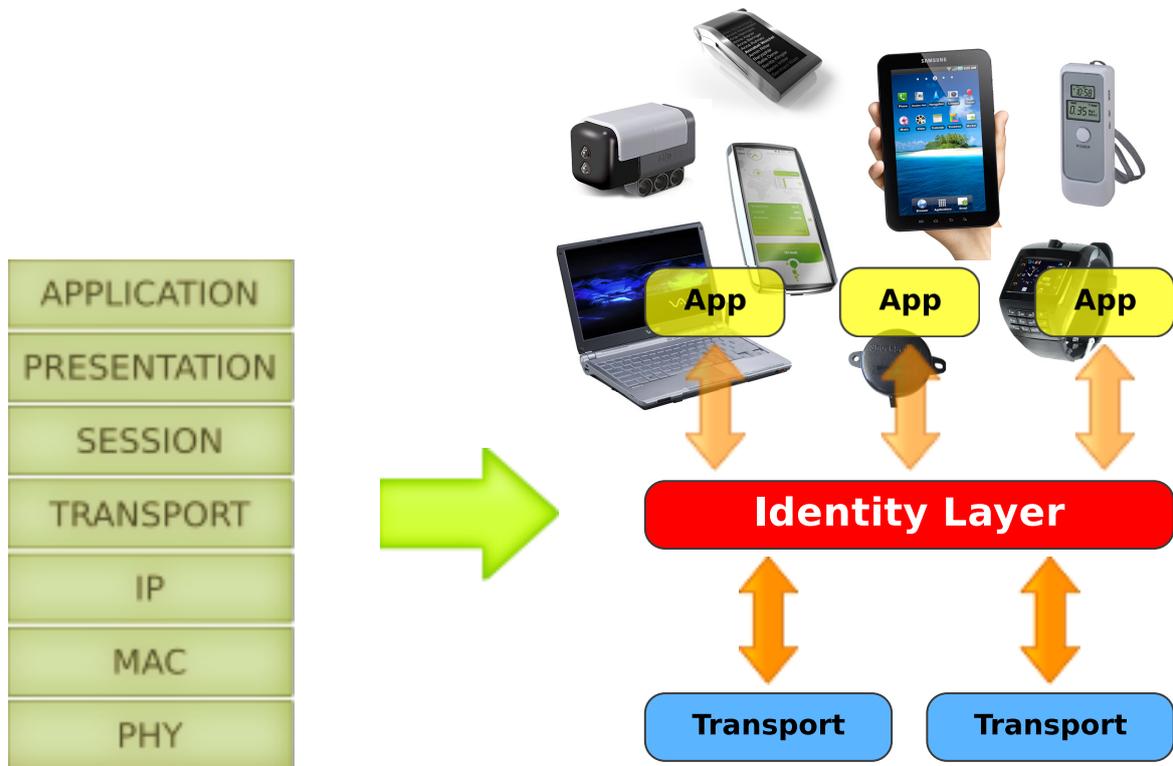


Figure 1.4: Restructured and simplified network layers.

1.4 Main Contributions

In order to accomplish the objectives described above, in this thesis we propose the design of a set of functional blocks that isolate and encapsulate the necessary functions to meet the requirements stated in Section 1.2, as we describe in Chapter 4. Then we show how to integrate them to form a final architecture, as we describe in Chapter 5. This allows us to advance the state of the art in some aspects, as described below.

Therefore, the major contribution of this thesis is the definition, design, and validation of a network model that uses identities (attribute sets) as communication endpoints. This model is supported by the necessary architectural components, effectively enclosed in functional blocks, to bring its benefits to network entities on top of different underlying network architectures. The central element is the Domain Trusted Entity Infrastructure (DTEi), as we introduce in Section 4.2, whose singular design is another key contribution of this thesis.

In order to instantiate communications using such model we require the design of a protocol that covers all communication operations, from the discovery of network objects

(entities) using their identity to the management of sessions, including support and management for dynamic multi-homing (mobility). This is the Identity-Based Network Protocol (IBNP) and forms the second main contribution of this thesis, complementing the definition of the DTEi and being integrated with it.

The actual data plane used in communications derived from the inclusion of the DTEi and the IBNP depends on the capabilities offered by the underlying network. Some underlying networks, such as IP or HIMALIS [14], allow the direct communication between two entities, so our approach has a direct embedding/translation into the network. However, other networks, such as CCN [15], do not allow direct communications, so we need to provide an intermediate adaptation layer. Anyway, sessions are managed by building an overlay network on top of the specific underlying network to which communications are embedded. Throughout this document we demonstrate the feasibility of this approach, which also forms another of our main contributions.

Modeling the resulting features for them to be integrated with any network architecture, including current or future architectures, is another of our main contributions. We achieve this objective by defining an Identity-Based Control Plane (IBCP) that encloses the aforementioned functions as a separate plane to the general control and data planes. This plane also makes extensive use of the overlay network technology to facilitate the integration.

From the definition of the IBCP we are able to integrate the main functions defined by our architecture with other network architectures, mainly HIMALIS and MOFI [16]. Achieving these integrations validates the usefulness of our approaches to other research works targeting the Future Internet at the time they suppose the collection of enormous feedback to our research. Such integrations form our last main contribution. However, the integration work still has some points to be improved and we will continue it in the future, as we describe in Chapter 7.

1.5 Related Publications

The research work carried out during the development of this thesis has led to different publications in conferences and scientific journals. The most relevant contributions are presented below in chronological order.

1. Introduction

Articles in Indexed Journals (JCR)

- P. Martinez-Julia, A. F. Gomez-Skarmeta, V. P. Kafle, and M. Inoue. **Secure and Robust Framework for ID/Locator Mapping System.** *IEICE Transactions on Information and Systems*, vol. E95-D, no. 1, pp. 108–116, 2012.
This article describes the initial approach to integrate the objective functions of this thesis with the HIMALIS network architecture.
- P. Martinez-Julia and A. F. Gomez-Skarmeta. **Using identities to achieve enhanced privacy in future content delivery networks.** *Computers and Electrical Engineering*, vol. 38, no. 2, pp. 346–355, 2012.
This article describes how to model the defined functions in this thesis to provide their benefits to content delivery networks through the information-centric networking scheme.
- P. Martinez-Julia and A. F. Gomez-Skarmeta. **A Novel Identity-based Network Architecture for Next Generation Internet.** *Journal of Universal Computer Science*, vol. 18, no. 12, pp. 1643–1661, 2012.
This article describes the control protocol defined in this thesis (IBNP) and demonstrates its security properties and performance.
- P. Martinez-Julia and A. F. Skarmeta. **Beyond the separation of identifier and locator: Building an identity-based overlay network architecture for the Future Internet.** *Computer Networks*, vol. 57, no. 10, pp. 2280–2300, 2013.
This article describes the initial shot of the final architecture and its validation.
- P. Martinez-Julia, A. F. Skarmeta, and A. Galis. **Towards a Secure Network Virtualization Architecture for the Future Internet.** *The Future Internet, Lecture Notes in Computer Science*, vol. 7858, pp. 141–152, 2013.
This article describes a full-fledged virtualization architecture for the Future Internet that integrates the identity-based functions proposed in this thesis as part of the mechanisms of the control plane.
- P. Martinez-Julia and A. F. Gomez-Skarmeta. **Empowering Security and Mobility in Future Networks with an Identity-Based Control Plane.** *IEICE Transactions on Communications*, vol. E97-B, no. 12, pp. 2571–2582, 2014.
This article describes the modeling of the functions described in this thesis to build the identity-based control plane that is then used to integrate the features provided by the final architecture with other network architectures.

Conference Contributions

- A. F. Gomez-Skarmeta, P. Martinez-Julia, J. Girao, and A. Sarma. **Identity Based Architecture for Secure Communication in Future Internet**. *In the 6th ACM Workshop on Digital Identity Management, 2010*.
This paper presented the roots and an overview of the objectives established in this thesis.
- P. Martinez-Julia and A. F. Gomez-Skarmeta. **Secure Identity-to-Identity Communications over Content-Centric Networking**. *In the 4th IEEE International Conference on Internet Multimedia Systems Architecture and Applications (IMSAA) 2010*.
This paper presents the instantiation of the initial steps of the architecture and protocol on top of CCN.
- P. Martinez-Julia, A. F. Gomez-Skarmeta, J. Giaro, and A. Sarma. **Protecting Digital Identities in Future Networks**. *In the Future Network and Mobile Summit (FNMS) 2011*.
This paper presents the initial mechanisms used to achieve network privacy by protecting the identity of communication parties.
- P. Martinez-Julia and A. F. Gomez-Skarmeta. **A Novel Architecture to Achieve Security and Mobility in Coalition Networks**. *In the RTO Information Systems Technology Panel (IST) Symposium, RTO-MP-IST-099 – Emerged/Emerging “Disruptive” Technologies (E2DT), 2011*.
This paper presents how to model the proposed approach to bring security to highly dynamic networks, such as coalition networks used in army.
- P. Martinez-Julia and A. F. Skarmeta. **A Lightweight and Identity-Based Network Architecture for the Internet of Things**. *In the Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS) 2012*.
This paper presents how to achieve proper communications in the Internet of Things using the architecture proposed in this thesis.
- P. Martinez-Julia, A. F. Skarmeta, V. P. Kaffle. **Research and Experimentation With the HIMALIS Network Architecture for Future Internet**. *In the Future Network and Mobile Summit (FNMS) 2012*.

1. Introduction

This paper presents the initial experimentation results obtained from the implementation of the HIMALIS network architecture.

- P. Martinez-Julia, A. F. Skarmeta, H. Y. Jung, and S. J. Koh. **Evaluating Secure Identification in the Mobile Oriented Future Internet (MOFI) Architecture.** *In the Future Network and Mobile Summit (FNMS) 2012.*

This paper presents the integration of the architecture proposed in this thesis with the MOFI network architecture to provide it with secure, identity-based capabilities.

- P. Martinez-Julia and A. F. Skarmeta. **An Overlay Network Approach to Find Objects in the Future Internet.** *In the Future Network and Mobile Summit (FNMS) 2013.*

This paper presents the discovery mechanism included in the architecture proposed in this thesis, which demonstrate how to find objects from partial identities.

- P. Martinez-Julia, V. P. Kaffle, and A. F. Skarmeta. **Integrating an Identity-Based Control Plane with the HIMALIS Network Architecture.** *In the IEEE Conference on Network Softwarization (NetSoft) 2015.*

This paper presents the integration of the identity-based control plane designed in this thesis with the HIMALIS network architecture.

Other Relevant Publications

- D. Papadimitriou, T. Zahariadis, P. Martinez-Julia, I. Papafili, V. Morreale, F. Torelli, B. Sales, and P. Demeester. **Design Principles for the Future Internet Architecture.** *The Future Internet*, Springer, pp. 55–67, 2012.

This book chapter describes the results of the joint research on the most important design principles for the architectures targeting the Future Internet.

- P. Martinez-Julia, A. J. Jara, and A. F. Skarmeta. **GAIA Extended Research Infrastructure: Sensing, Connecting, and Processing the Real World.** *In the TridentCom Conference, 2012.*

This paper presented the GAIA testbed used in the evaluation of the architecture proposed in this thesis.

- A. Gavras, A. Bak, G. Biczók, P. Gajowniczek, A. Gulyás, H. Hrasnica, P. Martinez-Julia, F. Németh, C. Papagianni, S. Papavassiliou, M. Pilarski, A. Skarmeta. **Heterogeneous Testbeds, Tools and Experiments – Measurement**

Requirements Perspective. *Measurement Methodology and Tools, Lecture Notes in Computer Science*, vol. 7586, pp. 139–158, 2013.

This article discusses the research methodology and introduces the experimentation tools we have used for the evaluation of the present thesis.

- P. Martinez-Julia, A. Marin Cerezueta, and A. F. Gomez-Skarmeta. **A Service Oriented Architecture for Basic Autonomic Network Management.** *In the IEEE Symposium on Computers and Communications (ISCC) 2010.*

This paper presented the foundation of an architecture for integrating network management with our architecture, as described in Appendix F.

- P. Martinez-Julia, D. R. Lopez, and A. F. Gomez-Skarmeta. **The GEMBus Framework and its Autonomic Computing Services.** *In the International Symposium on Applications and the Interenet Workshops (MidArch) 2010.*

This paper presented the framework used to support the autonomic management services used in the integration described above.

- Y. Demchenko, C. Ngo, P. Martínez-Julia, E. Torroglosa, M. Grammatikou, J. Jofre, S. Gheorghiu, J. A. Garcia-Espin, A. D. Perez-Morales, and C. Laat. **GEMBus Based Services Composition Platform for Cloud PaaS.** *Service-Oriented and Cloud Computing, Lecture Notes in Computer Science*, vol. 7592, pp. 32–47, 2012.

This article discusses the provision of cloud services with the network management approach we are integrating with our architecture.

- E. Torroglosa-García, A. D. Pérez-Morales, P. Martinez-Julia, and D. R. Lopez. **Integration of the OAuth and Web Service family security standards.** *Computer Networks*, vol. 57, no. 10, pp. 2233–2249, 2013.

This article describes the integration of OAuth and WS-security family, which are good candidates to interconnect security in application and network layers.

1.6 Thesis Structure

The remainder of this document is organized as follows:

First, in Chapter 2 we provide a brief description and analysis of the most relevant proposals to overcome the limitations found in the current Internet. This analysis considers the most important aspects of such architectures that try to meet the challenges exposed for the Future Internet. Then, we establish the roots of the architecture we will

1. Introduction

design, specially reflected in the need for an *identity plane* that allows network entities to communicate in terms of their digital identity.

In Chapter 3 we provide a view of the initial steps taken during the development of this thesis in order to achieve our objectives. Instead of searching each functional block independently, we face the problem in a top-down approach, by first achieving an architecture that allows network objects (fine-grained entities) to communicate each other in terms of a plain identifier, dropping IP addresses to lower layers.

In Chapter 4 we delve into the problem, researching different functions that are required to build the full identity-based architecture, evaluating their qualities, and encapsulating them into separated functional blocks. This begins with the definition of a trusted entity that will be instantiated into each network or administrative domain. Such entity will allow network objects to reach each other, in a secure manner, without requiring to know the underlying communication mechanisms.

Then we define a protocol to make effective the qualities of the mentioned functional block and thus allow network objects to establish secure network sessions. We then define a mechanism used to find entities in terms of their identities (attribute sets) instead of using just domain names or plain identifiers. This mechanism is then completed with an ontology to represent the information coherently. Finally, this chapter defines an identity-based control plane that makes use of the other functional blocks but provides their functions to other network architectures as a complement of their corresponding control planes.

In Chapter 5 we put everything together to build the final architecture, which forms the main result of this thesis and the baseline to integrate it into future approaches. In this chapter we delve into the stated problems and discuss which mechanism of the final architecture is used to resolve them. We also include a complete analysis of the qualities of such architecture.

In Chapter 6 we discuss how to integrate the resulting architecture with the HIMALIS and MOFI architectures respectively. They describe which functions they have in common and the points to attach the functional blocks designed in this thesis. These integrations suppose other of the main contributions of this thesis.

Finally, in Chapter 7 we present conclusions of this thesis and discuss the open issues found during the research phase and left to future work.

Chapter 2

Background

The architecture supporting the Internet is ossified, as demonstrated by several research results and its disability to address new service and user requirements from its core. Those mark the roots of the so called Future Internet (FI), which supposes a step forward in the architecture design of the core to offer more flexibility and robustness. However, redefining the network architecture of the Internet is not an easy task. There are several architecture proposals and research results that support different ways of overcoming the current problems to form the basement of the FI. In this chapter we walk through the most important and influent proposals, analyzing their benefits and drawbacks to state the baseline for the research work carried out in this thesis.

In Section 2.1 we walk through the main problems found in the current Internet architecture and describe the most outstanding proposals to overcome them. We continue with the analysis of capabilities in Section 2.2. Then, in Section 2.3 we establish the requirements that defend the role of identities in the network. Finally, in Section 2.4 we describe the basis of the work carried out in this thesis, outlining the objectives derived from the lacks exposed by current architectures and proposals.

2.1 State of the Art

The main objective of the Internet is to allow distant networks to communicate each other so the network objects connected to them (machines, persons, etc.) can seamlessly reach each other and perform their communication activities. This brings up to the top both the discovery of network objects and the routing of messages from an origin to a destination. Therefore, behind any architecture proposal for the Future Internet there should be a strong definition of routing and discovery [2].

2. Background

Moreover, the search towards the Future Internet has exposed many other challenges, the most important are: [1, 12] Energy efficient communications; separation of identity and address; location awareness; explicit support for client-server traffic and distributed services; person-to-person communication; incorporated security; control, management, and data plane separation; resource consumption isolation; symmetric/asymmetric protocols; and guaranteed policy-based quality of service.

Furthermore, in current networks, host autonomy and security are achieved through the use of Network Address Translation (NAT), which decouples routing inside an isolated network, or even an autonomous system, and the Internet [17]. NATs are a make-shift solution relating a few public with multiple private addresses, which has implications and impacts for certain applications and protocols, especially because NAT breaks the nature of end-to-end communications.

There are several proposals that try to address these problems. Most of them are organized towards the same research and/or experimentation objective. In Europe, with the Future Internet Assembly (FIA) [18] and the Future Internet Research and Experimentation (FIRE) [19]; in USA, with the Future Internet Design (FIND) [20] and GENI [21]; in South Korea, with the Future Internet Forum [22]; and in Japan with the AKARI project [23]. All these efforts have a common goal: designing and developing a future Internet that provides dependable, ubiquitous computing and communication facilities to society for future innovation. Below we discuss the most prominent proposals and how they try to resolve the limitations of the current Internet architecture, classified according to their type.

2.1.1 Evolution and Fixes of Current IP Architecture

We start describing the proposals to overcome the problems by complementing, evolving, or fixing the current networking and inter-networking architecture.

First, in TRIAD [24], the authors propose an IPv4 NAT-based architecture in which Full Qualified Domain Names (FQDNs) are used as identities, being mapped directly to next hops. Here, routing uses the Name-Based Routing Protocol (NBRP) [25], which works by distributing name suffix reachability among realms, much like the Border Gateway Protocol (BGP) distributes address prefix reachability among autonomous systems. TRIAD is forced to use resolution to reach objects that are outside their home realm, thus preventing its routing model to have proper scalability on fully general, mobile entities. Moreover, this architecture relies on locators following closely the Domain Name System (DNS) hierarchy.

As strict layering may be a culprit for the underperformance of the current Internet architecture, the authors in [26] propose to avoid layering violations present in the TCP/IP architecture by eliminating layers themselves. They present a Role-Based Architecture (RBA) sitting on top of IP, thus replacing the current transport layer (i.e. TCP) and part of the application layer. Despite its qualities are promising, the proposal never reached a certain maturity level; even a brief cost/benefit analysis is left for future work. However, adding an architected support for roles to the Internet remains to be an interesting concept.

2.1.2 Separation of Identifier and Locator

Instead of fixing the problems with complementary mechanisms, there is some consensus in the research community in that some problems related to end-to-end communications may be resolved by separating the location and identifier of a network node. This approach is followed by two outstanding architectures, HIP [27] and LISP [28, 29], as well as other derivatives and totally new proposals.

The Host Identity Protocol (HIP) introduces cryptographic host identifiers forming a new global name space as a new intermediate layer between the IP and transport layers. It decouples the endpoint identifier and locator enabling the transport on host identifiers and routing on IP addressing that serve as pure locators. Although this seems a good solution, it presents many problems to be deployed because of the intrinsic meaning of identifiers and, in general, its weak solution to all requested capabilities for the Future Internet.

On the other hand, the Locator/ID Separation Protocol (LISP) is a routing-based solution using map-and-encap at border routers. The upstream IP address of border routers is used as Routing Locators (RLOCs) of hosts residing in the local domain to perform inter-domain routing, while intra-domain routing is performed using conventional IP addresses also referred to as Endpoint Identifiers (EIDs). These identifiers (EIDs) can potentially be associated with a group of RLOCs to support multi-homing. LISP tunnels data packets between source and destination RLOCs using the LISP database, which contains the RLOC/EID relation for each local domain to interpret packets. It also uses an on demand cache to select the destination RLOC for sending packets towards specific destinations. As the main drawbacks exposed by this architecture we highlight the need of maintaining a LISP database, because it raises scalability concerns, and that the creation of an on demand cache to route packets among different domains is not clearly defined.

As a HIP derivative, the BLIND architecture [30] enhances HIP with security and identity protection, as well as location privacy by introducing new forwarding agents. Even

2. Background

though this is an interesting architecture from the security point of view, it inherits the same problems that HIP has.

Also being similar to HIP, the Routing Architecture for the Next Generation Internet (RANGI) [31] introduces a new layer called *NodeID*, which is used for transport related communications instead of IP addresses. RANGI adopts a hierarchical structure of host identifies employing special IPv6 addresses. The main benefit of RANGI is scalability but its main drawback is the complexity related with fundamental modifications into the IP addressing scheme and the DNS entry to store the required mappings.

The Node Identity Internetworking Architecture (NodeID) [32] proposal from the EU FP6 *Ambient Networks* project is also similar to HIP and LISP and introduces an architecture based on separation of locator, address, and administrative domains. Nevertheless, it is a network layer solution based on a locator/identifier split approach, and thus did not address session or identity management at all. Although the work carried out in the present thesis can benefit from some aspects of NodeID, it is by no means sufficient to solve the problem we are addressing. Also, this architecture suffers from scalability problems related to inter-domain routing; a process based on routing tags, which contain domain information about the location of a concrete node identifier.

Defined within the AKARI project, which is being carried out by the National Institute of Information and Communication Technology (NICT) of Japan, we find the HIMALIS architecture [14]. It proposes a complete architecture that shares features with HIP and LISP but targets on a new identification scheme, different from IP and capable to support sensor networks and the Internet of Things (IoT). The major benefits of HIMALIS reside in their simple targets of heterogeneity and mobility. It provides the necessary functional blocks to complement the current Internet architecture and allow it to evolve to a more functional network. However, there are still functions not covered by this architecture, specially those related to address the identities of network elements and manage communications according to them. Despite this limitation, the approach proposed by HIMALIS has some common targets with our objective, so we integrate some of the functions we have defined to the HIMALIS architecture in order to improve it, as shown in Section 6.2.

The Mobile Oriented Future Internet (MOFI) [33] proposes an architecture that considers, from the beginning, that network elements (hosts) can move across networks while also considering the existence of current IP networks as backbone for communicating separated edge networks. In contrast, it proposes new control and data planes for those edge networks to optimize mobility. The simplicity of this architecture and the organization of the functions provided in separated functional blocks (components) make it very interesting

for adopting new functions into the Internet. However, it does not address the security or discovery problems. As our proposal specially targets these problems we also consider to integrate some functions of our architecture with MOFI, as shown in Section 6.3.

Finally, the ILNP architecture [34] approaches the id/loc split by proposing a modification to the current DNS to use it to resolve names to identifiers and identifiers to addresses. It also proposes the incorporation of the identifiers as part of the IPv6 addresses used in network interactions. This proposal currently is being studied by the Internet Research Task Force (IRTF). Although this is a very lightweight and promising approach, it is host-centric and does not provide fine granularity or flexible naming. Also, the DNS infrastructure is not prepared to support the highly dynamic workloads required by the FI.

2.1.3 Overlay Network Architectures

A totally different scheme to overcome the limitations of the current network architectures, including the Internet, is by means of overlay networks. They are built in top of other architectures, which includes both current and future architectures. They use the resources offered by those architectures to build a totally different (overlying) architecture with many benefits and the only drawback of the initially degraded efficiency or performance in comparison with the underlying network they use.

The flagship overlay network of the research community can be found in Chord [35]. It describes a decentralized lookup service for mapping keys to nodes, much like a churn-tolerant Distributed Hash Table (DHT), which can be used for routing on flat identifiers (labels). Chord operates over a ring structure with fairly good performance in simulations. Nodes have fixed-size and arbitrary identifiers and they are sorted in a ring. To communicate a message to a node, the origin has to send the message to its *next* node in the ring, which will forward the message to its next, and so on to reach the destination. This is improved by using the so called *finger table*, which stores pointers to exponentially distant nodes, so messages are sent to the *closest* node to the destination, instead of the next one. This implies that, at the end, messages can reach their destination in an order of $\log_2 N$, where N is the number of nodes in the network and coincides with the bitwise length of the identifier. This upper limit is rigid and very interesting in terms of efficiency. Moreover, there are many derivatives of Chord proposing high improvements, such as LPRS [36], which makes the performance of Chord comparable in number of hops to its underlying network. However Chord has problems to recover from ring partitioning

2. Background

and lacks security, as it is not its main target. Anyway, Chord is a good basis to build a network architecture, as described below.

In order to make Chord a complete network architecture, ROFL [11, 37] defines a routing scheme for semantic-free flat labels. These labels may be self-certifying identifiers, an interesting choice for entity identifiers in an architecture targeting the Future Internet. It is based on previous work done by Chord, but also in Virtual Ring Routing [38] and Canon [39]. It supports three classes of nodes: routers, stable nodes, ephemeral nodes; and claims to manage correctly both the intra- and inter-domain cases. However, performance is not remarkable, and it also lacks traffic control. Concerning security, we can find [40] that introduces the necessary architecture, protocols, and models to build a secure policy-based overlay network for multi-domain scenarios.

Another influential derivative of Chord is Kademia [41], which has an highly improved routing protocol and algorithm based on the algorithm used in Chord. Both use the same type of identifiers for the nodes but Kademia proposes a different metric to measure the distance among them. While Chord uses the numeric distance between node identifiers, Kademia uses the so called *XOR metric*, with which the distance between two nodes is calculated by applying XOR to their identifiers. This way, the distance between two nodes is reciprocal, and the resulting model is more efficient. Another difference is that Kademia uses an iterative protocol, being the *caller* node the responsible of managing all the communications, while Chord relies in the *next* node to the destination to do so. Nonetheless, this is the routing protocol used to build the DHT behind the BitTorrent network, so it has proven its ability to scale up to several millions of nodes as well as its stability over time. We have found this protocol very interesting to form part of our architecture, as we describe in Chapter 4.

Currently, DHTs are mainly used in relation to peer-to-peer (P2P) protocols, even though there are proposals to apply those principles to other approaches [42]. There are many different and interesting P2P architectures [43] with their own capabilities. DHTs can also be used in Loc/ID separation proposals, such as in LISP-DHT [44] that proposes a DHT-based architecture to implement the ID-location mapping for LISP. There are approaches to build hierarchical DHTs [45], proposing to build DHTs in top of structured overlay networks.

2.1.4 Clean-Slate Architectures

As a different and disruptive family of approaches, clean-slate architectures propose to totally jettison the current Internet architecture and build from the beginning a well

designed architecture. They have their benefits and drawbacks but are worth to explore as they are current and strong that will provide valuable benefits, in terms of functions or models, to the Future Internet.

MILSA [46] and Enhanced-MILSA [47] explore a novel, clean-slate architecture for the Internet, based on the principle of identifier and locator split but with advanced capabilities. They make an explicit distinction between realms (organizational areas, which may form an explicit hierarchy) and zones (network connectivity areas). Realms are considered trust domains, that is, entities inside the same sub-realm are assumed to all trust each other; zones on the other hand have the requirement that their addresses must be strictly topologically aggregated. Management is made by special entities named RZBSs (Realm-Zone Bridging Servers, introduced in MILSA), acting much like registering proxies in SIP [48] but with advanced functionality. They take care of translating identifiers to locators and are the transition points between realms and zones. Regarding routing, MILSA relies on a stable layout of the RZBS, which must be pre-configured. Then, DNS is used for resolving RZBS names (but not for other nodes). These proposals also differentiate between *Provider Independent* (PI) and *Provider Aggregatable* (PA) addresses. PI is the address given by the organization and refers to the identifier of a peer, while PA is the address given by the ISP and refers to the location of a peer. A PI address, called Hierarchical URI-like Identifier (HUI), is divided in two parts: a flat part for inter-host trust and a hierarchical part for inter-realm trust.

EMILSA also proposes to use a Hierarchical Code Based Locator Structure [49] as PA addresses, that can be set in certain form for backward compatibility while the whole architecture is not completely deployed. Moreover, it is designed to support HUI-based multicast and *manycast* traffic, designating a HUI to the multicast group that is mapped to an RZBS. Manycast is used to deliver packets to a peer connected in different locations. These features, together with the previously described, make MILSA support for an integrated service model, so it can be integrated with different services, such as using the HUI as user identifier. The main drawback of these proposals has been found in their scarce security mechanisms. The authors claim that their design favours policy enforcement, but no specific policy framework is presented with them. Moreover, being clean-slate reduces their impact, so although their proposals are interesting, they are difficult to translate into the real network.

Trilogy [50] aims to establish a new Internet architecture combining resource control, reachability and socio-economic issues. It proposes multipath transport to enable a more robust Internet combined with multipath congestion control to handle efficiently fluctuations in capacity, while the congestion volume charging is performed from a

2. Background

socio-economic perspective. Considering NodeID, Trilogy is more concerned with scalability adopting the LISP perspective, while the identity is only accounted for the congestion each user cause in the network. However, security, mobility, and multi-homing are not part of Trilogy, which are vital elements for the work carried out in this thesis for the design of the architecture.

ANA [51] has developed a novel network architecture with autonomic attributes that enable flexibility in forming network nodes and networks in a dynamic manner. ANA introduces content-based communication and routing; a scheme that use an additional routing overlay to the conventional location based routing, which identifies path in a distributed fashion based on the content of communication. Such an approach has similarities with our proposal as they identify paths based on intention, but our proposal advances this approach much further adding security, Quality of Service (QoS) and mobility.

From the 4WARD FP7 European project [52], we found the concept of Generic Paths (GP) in the routing process, which aims to extent routing horizontally, introducing multipath and inter-domain specific extensions and vertically combining routing with physical and transport layers. The use of such GPs is to support various flows across different domains, technologies and QoS demands. The main lack of GPs is to encounter the content and identity into the routing flows or otherwise sessions, which has certain drawbacks in terms of QoS and security, while the whole approach is not scalable since addressing, is still overloaded combining both identity and location. Moreover, GPs are designed to be used by other upper-layer architectures with other mechanisms to build complete network architectures.

2.1.5 Information Centric Networking (ICN) Architectures

Besides the architecture approaches presented in previous subsections, the Information Centric Networking (ICN) approach has been a disruptive but well accepted and supported trend for future networks. It breaks with current node-centric networking by claiming that the majority of data communications in the Internet are held to request or deliver content from a provider to a consumer, so they have an information-centric nature. Today, information is hosted by specific nodes and the only way for retrieving it is via establishing an end-to-end communication with them. Recent research efforts towards developing architectures for the Future Internet have adopted a totally different point of view to address network operations by raising the role of information to the center of communications. In these approaches, the network does not connect nodes with links and

processes via end-to-end connections, as is the case in the current Internet, but connects information consumers with the information they request, that is provided in a decoupled way by producers and distributors. This opens up new opportunities for accommodating, for example, sporadically connected nodes efficiently. However, it is not considering the identities of network nodes in any way.

The EU-funded projects 4WARD [52] and SAIL [53], its successor, defined and motivated the introduction of a new Future Internet architectural paradigm called Network of Information (NetInf) [54] that extends the concept of identifier/locator split with another level of indirection and decouples self-certifiable objects from their storage location/s. NetInf distinguishes between Data Objects (DO), always encoded in a particular scheme (bit-pattern), and Information Objects (IO), i.e., information at a level above particular encodings. For example, much of the web content found today is semantically related, but this relationship is not represented in any way in the Internet. Similarly, relationships between copies of the same content are not represented. By employing bindings between IOs and DOs, information can be unambiguously replicated and located in the network. Moreover, since objects are self-certifiable, users can effectively employ access patterns that are reminiscent of any-casting and obtain the object that is *closer* (in network terms) to their access device.

Another pair of EU-funded projects, PSIRP [55] and PURSUIT [56], which is its successor, also approach ICN but from a totally different point of view. They propose a publish/subscribe scheme for the Internet, where information consumers *subscribe* to the information they want and information providers *publish* that information. In contrast with NetInf, this solution also approaches the information (data) delivery process and propose different identification schemes for information, subscribers, publishers, and intermediate nodes.

The Data-Oriented Network Architecture (DONA) [57] is a communications architecture that replaces DNS names with self-certifying flat names and a name-based anycast primitive above the current Internet. Instead of certifying the content, DONA certifies the publishers and labels the data. Also, the data can not be dynamically generated, it must be first registered in the trusted resolution handlers (RHs). Once the content requested by a client is found, it is delivered using IP routing. The security in DONA is achieved by content and provider validation.

The Content-Centric Networking (CCN) approach [15] proposes an architecture similar to PSIRP/PURSUIT but enlarging intermediate content-centric elements to all intermediate elements in the network (i.e. switches and routers). In CCN, information consumers tell the network their *interest* for certain information item (content object) and

2. Background

then the network operates to provide it. Information producers do not need to react to consumers, they just drop the information to the network, as the network will deal with it. This approach can be seen as a globally distributed cache, that fits perfectly in Content Delivery Network (CDN) scenarios. The problem with this architecture is that also leave aside the identity of communication parties, so it does not consider endpoint security or privacy.

In summary, the work carried out by the present thesis is effectively within this new area of research as the architectural emphasis is moving away from nodes and network interfaces, towards higher level abstractions, such as *information* or *content* (NetInf, PURSUIT, CCN) and now also *identity*, as proposed by the present thesis. We plan to capitalize on them while further developing the concepts of identity management and identity-based sessions. Taking this into account, we have considered ICN architectures, specially CCN, in our work and have some interactions with it, as discussed in Chapter 4.

2.1.6 Architectures for Search and Discovery

In this subsection we give some hints about the best known mechanisms for search and discovery. We first consider the well-known Domain Name System (DNS) [58]. It has evolved over the years to cover the basic needs of the name lookup operations of the Internet but its query flexibility is not enough to cover the requirements of the FI, remarking its lack to support high-rate of updates. To achieve query flexibility, the straight alternative to DNS is the Lightweight Directory Access Protocol (LDAP) [59]. It provides a mechanism to query on hierarchically organized information, which is subject to predefined schemas. However, its evolution possibilities are coped by its update-rate limitations and static linkage between its nodes.

Providing high scalability degree in the storage as well as flexible support for query and update operations, we find the DHTs introduced above. When they are deployed for search and discovery, they use their totally decentralized system to store data objects for easy and quick access (query) and update (store). As DHTs are built on top of overlay networks, the information items are spread and identified with unique keys. As stated above, the DHT of BitTorrent, which is based on Kademlia, is one of the most widespread and widely used.

That said, overlay networks and DHTs are well suited to form the basement of a proper discovery mechanism, as we can see in the Overlay Management Backbone (OMB) approach [42], but they only provide data storage and retrieval functions. To add proper schema evolution to the information/content discovery we also need some strong description

mechanism, such as the Resource Description Framework (RDF) [60]. It provides a flexible and extensible mechanism to define network objects. Information in RDF is represented as triples (subject, predicate, and object). The subject identifies the information object, a predicate identifies a relation type, and an object identifies a property of the described object. To ensure semantic consistency of the definitions, their formulation is guided by schemas, called ontologies and vocabularies, which are able to evolve and thus be adapted to new requirements.

Combining a DHT mechanism with RDF we find, for example, RDFPeers [61]. It proposes to store each RDF triple in three separate nodes of the DHT, using keys derived from the subject, the predicate, and the object. Then, it propose to use an adapted version of RDQL [62] to perform the queries, retrieve the matching triples, and computing the query result. The main problems of this approach are that it consumes a lot of storage space and that it is not efficient for simple searches. Moreover, SPARQL [63] has superseded RDQL as the *de facto* query language for RDF, providing a more coherent and simple search mechanism.

On the other hand we have more specific discovery protocols for the network layer. The most wide-spread are Bluetooth SDP [64], UPnP [65], Salutation [66], SLP [67], River (heir of Jini) [68], and mDNS/DNS-SD [69, 70]. In [71] we find a survey that describes the peculiarities of each, together with other global discovery methods based on overlay networks. Moreover, in [72] we find an analysis of the protocols and concludes that none of them have the necessary expressiveness to cover the discovery requirements for the Future Internet. Then, it defines a basic ontology to overcome this problem but does not meet with the required simplicity and genericity so it can not properly describe any type of network object.

As mentioned above, using overlay networks to build discovery services is a different and emerging approach. GloServ [73] proposes to use the CAN [74] overlay network for service discovery and also demonstrates an example ontology for travelling problems. It is centered on the destination of the travel but is complimented with activity, accommodation, and location. However, it is tied to travel activities and not well suited for general discovery of arbitrary network objects.

From the technologies developed for the evolution of the World Wide Web we can find strong and supported initiatives and proposals to apply semantics and thus ontologies to the data stored and linked on the Web. That is the case of the Linked Data [75], the basement of the *Web of Data*, an overlay network formed by the data and its relations. It is an initiative to describe the existing data on the web with RDF by following different ontologies. It illustrates how a (big) cloud of publishers are following the initiative (4.7

2. Background

billion of RDF triples, interlinked by around 142 million RDF links). This represents a frame to demonstrate the benefits of the adoption of ontologies and semantics in the network object discovery.

Many European projects have approached the application of semantics and ontologies to network discovery. We can highlight SPITFIRE [76]. It proposes a *Web of Sensors* that uses RDF to describe the information and SPARQL to query it. It also proposes a complete ontology for sensor environments and reuses other previously defined ontologies. Although it is near to be prepared for general use, it is tied to sensor networks and lacks proper descriptions for general network objects. Finally, being among the ICN architectures introduced above, NetInf also combines a hierarchical DHT with RDF. It provides a mechanism to model and organize linked information, hence its name. NetInf handles content metadata as information objects (IOs) with the nuances of the linkage among them.

2.2 Gap Analysis

In this section we present a qualitative analysis of the most outstanding and complete proposals from those described in the previous section in order to find their strengths and weaknesses so we can clearly identify the gap that should be filled by our proposal. Moreover, this analysis has been guided by the specific requirements mentioned at the beginning of the present Chapter, and those that we address in the work carried out in this thesis.

During the analysis we have noted how research trends are increasingly centered around the approaches providing the separation of identifiers and locators, from its first statement in [77] to [33] – although a solid, detailed solution addressing the issues arising from it, in a way which can be proposed as a candidate for the Future Internet, still remains to be found. However, considering the isolated properties of the different architectures we can find complementary functional blocks that can provide enough qualities to meet with the requirements we discussed above. In order to be sure that the analysis is meaningful and that independent elements do not blur our conclusion, we have only considered proposals that claim to:

- propose a reasonably complete architecture;
- separate effectively the node locators and their identifiers;

- eliminate the problems present in the current Internet derived from having IP addresses everywhere.

This yields the following candidates: HIP, LISP, HIMALIS, MOFI, ROFL, EMILSA, NetInf, PURSUIT, and CCN. For these, we have evaluated their strengths in several aspects, using the following parameters:

- how much architected support for policies they have;
- how scalable they are;
- how independent they are of the current DNS resolution scheme and IP layout;
- how pragmatic they are, as opposed to purely theoretical approaches;
- how secure they are;
- how much separation they manage to do between control and data flows;
- how deployable and manageable they are, as opposed to requiring much pre-configuration;
- how well they perform.

The results of evaluating these parameters are summarized in Figure 2.1. It is clear to see that all approaches lack on security and many of them also lack on policy support. Specifically, even though TRIAD is a fairly complete solution, based on IPv4 and, in principle, quite deployable and scalable, it lacks however an explicit policy framework, and is too dependent on IP addresses being topologically aggregated, and also on nodes following closely the DNS hierarchy.

This deficiency is addressed in HIP and LISP, as also seen in their independent figures, but they also lack in security, policy support, and independence of IP. Both HIMALIS and MOFI have similar qualities each other, as depicted in their independent plots, providing good level of independence of IP and separation of control and data planes. They also have similarities with HIP and LISP, specially in the lack on security and policy support. As a totally different architecture, ROFL also provides independence of IP, together with pragmatism and scalability. Although it improves in security, it neglects the performance and policy support.

Being a clean-slate architecture, EMILSA has a good score in policy management, as well as in achieving a good separation between control and data planes. Its main drawbacks

2. Background

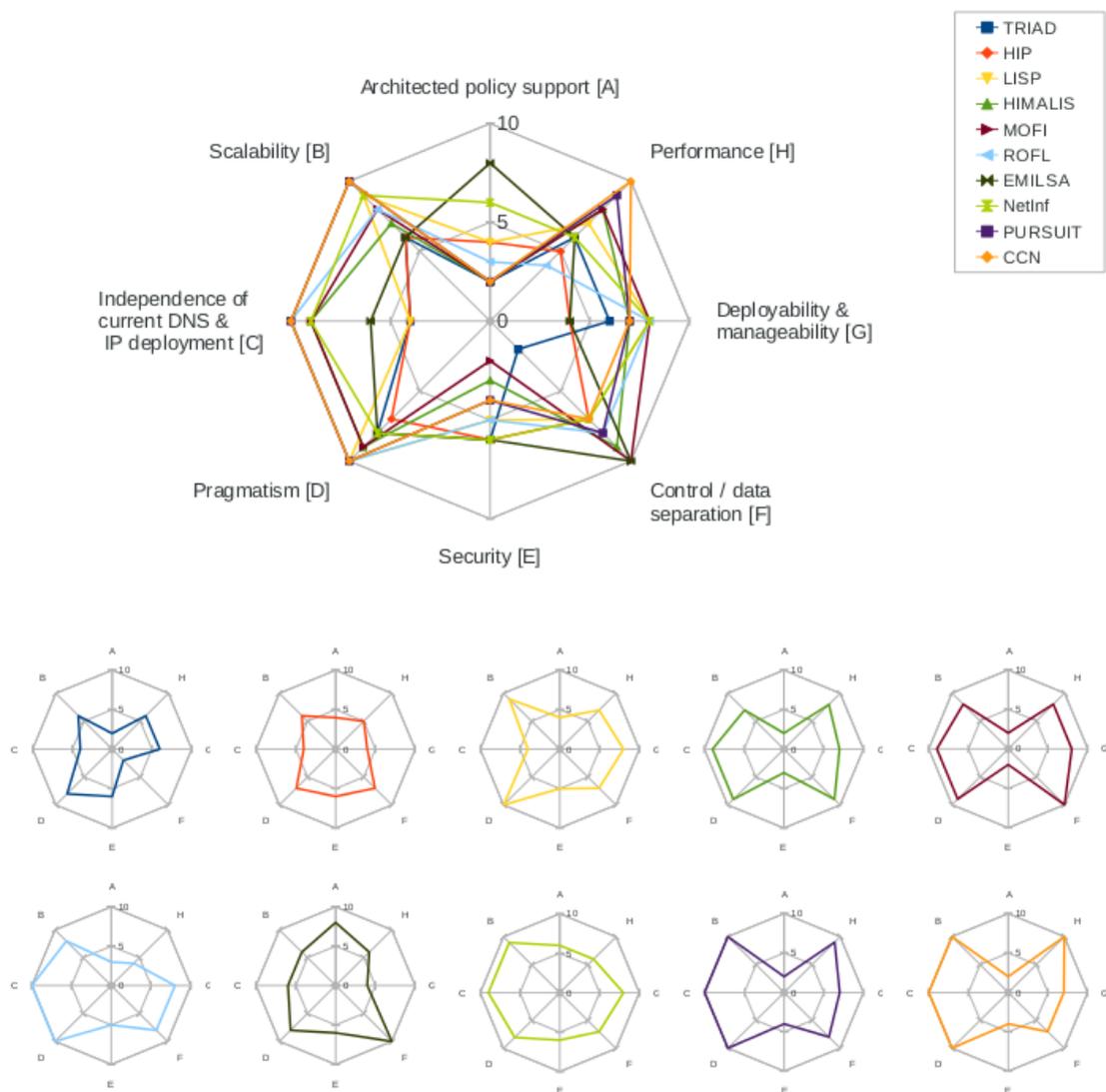


Figure 2.1: Analysis of architecture proposals for the Future Internet.

are the need for pre-configuration (placement, population, and management of directory services) and the little security services supported by the architecture. Even though the final three architectures from the analysis are information-centric, there is a clear separation from NetInf and PURSUIT/CCN. NetInf has some advancements in security and policy support, being one of the most complete architectures when considering all aspects. In contrast, PURSUIT and CCN, which are very similar, have many improvements in performance, scalability, pragmatism, and independence of IP but, as many other architectures commented above, they lack in security and policy support.

The results of this analysis can be summarized as follows: there is a significant area (gap) to be covered regarding architecture proposals for the Future Internet; namely, we find that adding qualities such as control/data separation or architectural policy support causes a low score in the ability to deploy and manage the solution. Security is also affected by this, although we believe this is purely coincidental as there is no good reason for it. Scalability is allegedly good in all the proposals analyzed here, although there should be more research efforts in improving it, along with the independence from the current DNS resolution and IP aggregation.

The concept of decoupling identity and location aims to separate the routing process, which is related to locators and IP addresses from higher layers, by introducing node identifiers as an intermediate layer for security, mobility and multi-homing purposes. To deliver such separation, a new architecture needs to address node identifiers in every network element. It also has to provide methods to maintain associations between identifiers and locators, so they enable successful sessions. The means of developing a separate location and identifier-based routing scheme is realized though a new node identifier layer mainly used for transport or higher layers, while routing is still performed using conventional IP addressing. Although each layer serves a different purpose, an association or mapping is essential to provide location information. Such mapping should differ from the map-and-encap router-based solution introduced by LISP, which still uses IP addresses for both identifiers and locators.

Although not included in the analysis because it does not provide a complete network architecture, in the internetworking architecture proposed by NodeID, the location and identity split is applied on every network element providing an integral security solution, smooth interconnection among heterogeneous domains along with mobility and multi-homing support. Nodes resident to a particular domain need to register their routing tag along a sequence of NodeID routers that inter-connect different domains to a common static core network. A routing tag resolution system at the core maintains an association of routing tags and individual NodeID routers belonging to domains directly connected with the core. This raises scalability issues related to inter-domain routing particularly in advertising and maintaining an accurate id-location association as well as on the location of maintaining such information. New ways to scale the identity location separation should explore a hierarchical structure of host identifies like the one proposed in [31] or introduce network identifiers to serve as an association of the NodeID with the network of residence. An interesting issue is to provide methods to aggregate the association between NodeIDs and locators and investigate the impact of network topology in defining a protocol and

2. Background

architecture to perform identity routing. This will be one of the main topics of the work carried out in this thesis.

Besides scalability issues, the architecture proposed in this thesis will enhance security and communication flexibility. In particular, the discovery process will combine the identity of the network object (i.e. host, user, thing, etc.) in the process of selecting paths based on identity routing, while considering the need of the user providing different degrees of security or preferring certain interfaces and communications ports. Also the issue of mobility requires further investigation in this particular context of path discovery with traffic engineering and multi-path support being critical options of the proposed routing scheme bringing together security with QoS on establishing paths. New features of combining user identity with node identifiers to perform the routing operations and their impact on establishing flexible paths for certain sessions with the option of pausing and re-starting a communication will be also studied.

2.3 Identity Management, Discovery, Authentication and Authorization

In this section we discuss the background of the present work regarding identity management, the discovery of network objects, and the authentication and authorization mechanisms used in the network. First we throw the question:

What is a digital identity?

One can find many different answers to this question. There are several definitions and variants. Perhaps the most general definition is the one from a new draft ITU-T standard (X.1250) on global identity management [13], which states that identity is the *Representation of an entity (or group of entities) in the form of one or more information elements which allow the entity(s) to be uniquely recognized within a context to the extent that is necessary (for the relevant applications)*. This definition is so general that it lacks precision of whose identity we are talking about (who or what is an entity?) and what data are we talking about (what is an information element?). However, it clearly exposes that an entity can be any object, including persons, machines, services, things, etc.

When identifying people, the information elements are restricted to Personal Identifying (or Personally Identifiable) Information (PII), which is *the information pertaining to any living person which makes it possible to identify such individual (including the information capable of identifying a person when combined with other information even*

2.3 Identity Management, Discovery, Authentication and Authorization

if the information does not clearly identify the person) [78]. We can consider that PII is simply the attributes of a person, such as: their hair colour, sound of their voice, height, name, qualifications, past actions, reputation, medical records, phone number etc. You might think that hair colour is not PII and is not a digital identity as it is too generic, but if we had a rule that stated that ginger haired people are granted a 10% discount at Ginger's hairdressing salon, then hair colour alone would be sufficient identity information to allow a person to be uniquely recognised within a context to the extent that is necessary (for the relevant applications).

That said, even something as generic as hair colour can be included inside an identity. On the other hand, when identifying service providers, the attributes that are of most interest are business related ones, such as their VAT registration number, their postal address and telephone number, their credit rating, their logo, and their reputation. To summarise, we can say that a person's or SP's (digital) identity comprises a set of attributes, and only a subset of these attributes are necessary to allow the entity to be sufficiently recognised within a given context. We will deepen into this assumption throughout the development of this thesis.

Current state-of-the-art federated identity management systems are based on the SAMLv2 [79], Liberty Alliance [80], or Shibboleth [81] models to provide infrastructure support for identity management, authentication, and authorization services. Anyway we should clarify first the difference between an identifier and an identity in the context of this work area. An identifier is usually a series of digits and/or characters that is used to uniquely identify an entity within one domain or system at certain moment of time. Therefore, two entities (objects, users) within the same system cannot have the same identifier, at least at the same time. This way, an identifier, such as an international phone number, is a rather special type of identity attribute, since no two users can share the same identifier at the same time, whilst they may have other identity attributes in common, such as hair colour.

Moreover, an identifier is tightly bound to the system or domain in which it is defined; it usually cannot be meaningfully moved between domains, unlike the other identity attributes. Indeed, different domains can use the same identifier to identify different users. An identifier is only one of the identity attributes that comprise an entity's digital identity within a system. Different service providers know different subsets of attributes for the same identity, but each provider will have its own identifier which uniquely identifies this entity within their system. Likewise different entities know different subsets of attributes from the identity of a service provider. For example, a bank has a different set of identity attributes in the network to those it presents to the public. An individual object whose

2. Background

identity is distributed throughout many systems will therefore have multiple identifiers. For example, a person will have a passport number, login identifier, social security number, email address, etc. These attributes are each unique within their own domains. Some systems may store the identifiers from remote domains as well as their own. For privacy (and other) reasons, entities are typically wary about releasing their identifiers to third parties, since these can uniquely identify them, whereas their other identity attributes, such as age, typically cannot.

An attribute assertion is a key concept of identity management. It is a claim made by some entity (the asserter) that a particular entity (subject) possesses a particular attribute. Any entity can assert any attribute about any other entity, but not all attribute assertions are true or believable. Ultimately it is the decision of the relying party which attribute assertions to trust and which to not trust. In order to be widely trusted, attributes should to be conferred on entities (or asserted) by authoritative sources. Whilst some entities may be trusted in some situations to assert some of their own identity attributes (e.g. a person stating his/her favourite drink), they certainly wont be trusted in all situations to assert all of their own identity attributes (e.g. qualifications or criminal record of a person).

In the same way, service providers are not trusted to make all assertions about themselves (otherwise auditors, regulators and consumer watchdogs would not be needed). Thus different authoritative sources are usually responsible for asserting different attributes about entities. For example, the university in which one person obtained his/her graduation is the authoritative source of his/her degree attribute, the postal service is the authoritative source of postal addresses, the tax authorities for VAT numbers, etc. These authoritative sources are also known as Attribute Authorities (AAs).

An identity provider (IdP) is an attribute authority combined with an authentication service to authenticate its users. An IdP can authenticate a user and then issue an attribute assertion about the user. Attribute assertions typically have to be digitally signed to ensure their integrity and authenticity. A digitally signed attribute assertion is an authorization credential. Authorization credentials are different from authentication credentials. Authentication credentials allow users to identify themselves via an identifier, which is an attribute that use to be static but can vary on demand. In contrast, authorization credentials allow users to identify themselves via a chosen subset of their identity attributes, without necessarily revealing their actual (whole) identity or their identifiers.

In general, from this topic we have extracted for the development of the present thesis that once each party to a communication has identified the other from a subset of their identity attributes, each may then authorize the other to perform some function.

Each party may attach their own identifier to this particular communication, so that the communications can be restored if it unexpectedly breaks or is temporarily suspended. Thus, here we propose to use identity attributes for discovery, identification, and authorization purposes, and identifiers for uniquely identifying a particular communication session.

2.4 Towards an Identity Plane Using Attribute-Based Discovery

Through the work carried out during the present thesis we want to go one step further than securing content, and start talking about *securing identity*. We propose to use the identity as the end point of the communications and to use it as an enabler of the characterization of the communication process like security or fine-tuning the desired addressing. For example, we propose to allow explicitly requesting (or rejecting) anycast/manycast delivery, limiting the final recipients based on some criteria based on identity attributes, etc.

Today we have Skype, GTalk, MSN accounts in addition to cell phone numbers plus VoIP providers. All of them use domain specific unique endpoint identifiers which actually refer to the same entity (specially the person behind the communications device) and can be seen as attributes of its identity. However, during a communication, a network object wants to reach other particular object (or objects) irrespective of the underlying technology. It should be possible to say, make a phone call to an identity you know, without knowing the phone number of that identity. In that sense this means to design how to develop a complete, end-to-end identity plane based on the identity and the attributes associated to it, with special attention to issues such as the separation between connectivity and administrative domains. This plane must integrate all the needed logic for implementing the identity-to-identity approach we propose in this thesis.

The use of an appropriate discovery service could allow the communicating object to be provided with the most-suitable underlying technology at a particular moment, or redirect it to intermediate elements that enable such communication under the specified parameters. The discovery service should be able to determine if the remote entity (using its identity) is connected to the requested service at the moment, or if its device is on (it is reachable), or it have a web site [54]. We propose to build a discovery service based on entity attributes so other entities will be able to identify entities and look them up based on their identity attributes. The discovery service will be appropriately secured so that entities can privacy protect their identity attributes and set policies for who should be able

2. Background

to *see* which attributes. Whilst persons may wish to protect their phone numbers, web sites may wish to make their URLs public. Our model will be unified so that entities at each layer of our communications architecture can store their identities in the discovery service, so that other entities will be able to discover them.

An entity that wishes to contact a service provider may first look them up in the discovery service, or may be directed to its web site by a previous communication, or even be directed to a spoof site by a phishing attack. The object establishes the communication with the service provider, which now confirms its identity not via a simple TLS certificate as now is done, but via a set of attributes properly asserted by trusted third parties. For example, if the web site owns to a bank, it may display logos that reveal, on clicking, an assertion from the postal service stating its postal address, an assertion from the tax authorities asserting its VAT number, one from Companies House providing its registered company name, an assertion from Dun and Bradstreet saying that it is triple A rated, and one from the Trademark Registrar asserting its trademarks.

In this way network objects provide plenty of evidence about their identity, allowing other objects to recognise and confirm it. This process of course can be fully automated, so that the object provides the assertions automatically to other objects when requested to do so, and some relying and fully trusted entity can check their validity and only alert the caller entity when it is unable to perform the validation. Because the assertions are freshly minted and digitally signed by their authoritative sources, it will be extremely difficult for an attacker to spoof any identity and thus global security is enhanced.

It is worth to mention that all the assertions contain the domain name of the entity they are asserting as well as its public key. The service provider will sign a message to the user to prove it is the rightful owner of the private/public key pair. It may also provide a self signed assertion during the initiation of communications providing its public key. This allows entities who now trust this service provider to get its public key certificate and incorporate it to their local key store as a root of trust, after which they will be able to validate assertions made by this organisation about other ones.

A common practical example can be depicted as follows: A bank could make an assertion via one of its business customer web sites that they bank with it. In this way, the network of organisations that user learn to trust can grow organically. A potential customer no longer has to trust a single SSL certificate signed by a CA before transacting with a service provider; it can validate the various logos on the web site to see what other authoritative sources say about this web site. In the current Internet trust is constantly deteriorating due to the weakness of the current SSL certification infrastructure – hence the recent provision of Extended Validation Certificates to try to counteract this trend.

2.4 Towards an Identity Plane Using Attribute-Based Discovery

Deteriorating trust limits the Internet's potential for growth as a pervasive infrastructure. The model we propose allows trust to actually grow organically thereby increasing the potential of the Internet to connect all things together.

Similarly an entity can provide assertions to the endpoint of the communication (lets assume for simplicity another entity) about itself. The entity does this by selecting its identity providers and asking them to create attribute assertions about it. For example, the entity may present to the service provider an assertion from its bank saying that it is a customer with a positive credit balance, or from a credit card company saying that she possesses this particular credit card, and from the Postal Service stating her postal address. The important thing to note is that the entity never presents its authentication credentials to the service provider, nor is it ever duped into communicating to a spoofed entity of its IdP, since it chooses which IdPs it wishes to use and which attribute assertions are sent to the service provider. In order to link all these assertions together, the entity must hold an asymmetric key pair, to which each of the attribute assertions are linked. In this way the service provider can validate that all the assertions refer to the holder of the same private key, and the entity can sign a message to prove it is the holder of such private key.

Once two objects (user and service provider) have performed mutual authentication, an identity-based session is created by the infrastructure, and it is given an identifier which uniquely identifies it and its context. The session now becomes a new identity in the discovery service, enabling it to be subsequently searched for. Both parties give this session a local nickname, e.g. the user calls it *bank query about my overdraft* and the bank calls it *customer X calling* for ease of reference. As the context of the communication changes, details about the session can be updated in the discovery service. The users may update the nicknames to reflect this new context. For example, as the user navigates through the bank's site to refine his query about his overdraft, the bank may rename the session *customer X query about bank charges*. Now, if the session is closed or broken for any reason, either party may re-establish it by asking the infrastructure to reconnect them to their nicknamed session. Once the session has been re-established it is obvious to both parties what the communication is about, as the correct context has already been established.

In summary, the architecture of the Future Internet has to consider the reality of communications, the actors involved, their intent and their final objective, in order to provide a view of the network that is adapted to the actual needs of network objects. The architecture has to ensure that this adaptation is achieved by the network in an integrated and transparent manner, so complexity is not relayed to network objects. This imposes

2. Background

Table 2.1: Summary of target qualities for the final architecture.

Requirement	Feature
Flexibility	Use <i>identities</i> to get flexible naming, decouple the identification and location processes, and ensure that network functions are agnostic to underlying network technologies
Adaptability	Ensure network functions can be instantiated on top of different types of underlying infrastructures
Granularity	Extend the endpoint concept by removing the dependency on the <i>host</i> concept, ensuring that any (abstract or concrete) object can be an actor in communications
Dynamicity	Enable network structures to change frequently, including identities (dynamic description and flexible discovery), network attachment points (integrated mobility and multi-homing), and respond to changes in the environment
Heterogeneity	Allow multiple underlying networks to coexist, separating control operations from data path instantiation
Integrated Security	Use policies to manage access to all delicate information, including the location of objects
Privacy	Ensure that only the authorized elements are able to know <i>who</i> is behind a communication

a set of specific requirements qualities that should be offered by the components of such architecture, as shown in Table 2.1.

Meeting those requirements imply a major advancement in the architecture of the Future Internet. However, there are other side benefits obtained from the design of the *Identity Plane* that provides the mentioned features. In particular, the clear separation of functional blocks will enable the introduction of some of such features into other network architectures, addressing control operations from the perspective of the real object that is behind communications. In the following chapters we will delve into the design of the architectural elements needed to achieve such requirements.

Chapter 3

Initial Steps: An Overlay Approach

In this chapter we discuss the initial steps performed during the development of the architecture proposed by the current thesis, a proposal to build identity-based networks using overlay network mechanisms. Here we define an architecture that benefits from the widely deployed identity provider infrastructures to associate the unique identifier used in the communication with a real identity, managed by its identity provider.

Apart from the location/identifier separation, the main goal of the initial architecture described in this section is to provide a service-aware protocol that will help services with different aspects of the communication, such as mobility (because it is location independent), the user identity, and others. Therefore, a service can relay part of its current job on the capabilities provided by our architecture, benefiting the separation of concerns and letting services to be concentrated on their businesses. As the overlay network behavior is very important for this architecture, we also present the results of some tests we performed with a particular algorithm to build overlay networks, as well as with certain optimizations of it. With those results we demonstrate the flexibility provided by the overlay network routing algorithm, both in terms of simplicity, evolution capacity, and performance. It encourages us to continue investigating different variations and optimizations.

The remainder of this chapter is organized as follows. First, in Section 3.1 we contextualize the motivation towards the design of our initial architecture. Then, in Section 3.2 we overview the architecture, particularizing its most important aspects and qualities. In Section 3.3 we show the initial results we obtained from the evaluation we performed to different overlay network routing algorithms. Finally, in Section 3.4 we discuss the conclusions of this chapter and introduce the discussion carried out in the following chapters as well as issues left for future work.

3.1 Introduction

This chapter presents an approach to build identity-based networks, mainly targeting the requirements of the Future Internet (FI), a service-based Internet. Thus, the architecture design we propose is service-aware and may benefit all services. It provides certain capabilities implemented in almost all service and application level protocols, so they may be simpler and more reliable. For instance, services may relay some parts of their functionality, such as user identities and identification (authentication), on the protocol proposed in this architecture.

We propose an identity-based architecture that approaches the separation of location and identity by using location addresses in a lower protocol such as IP, specially IPv6, and identifiers in the intermediate protocol, that is used in end-to-end communications through an overlay network. We propose to build a different overlay network for each service-type, so users of different services do not interfere each other. The identifiers are used by one part of the communication (peer) to unambiguously determine “who” is the other part of the communication. Each part of the communication can be a user (person), a machine or a service, so we occasionally call them *peers*. Moreover, we define the way to validate the peer identifiers, so they correspond to the identity the peers pretend to be. A peer may use different identities and they can be associated to different services or service-types. Also, the peer is able to use an anonymous identity to protect its real identity. This architecture inherits all the benefits of the identity management, that is a key aspect of the FI.

Apart from the identity orientation, our architecture is focused in enhancing other communication aspects with the introduction of the overlay network as communication mechanism. This is the most important piece of our architecture because the other parts can be integrated from existing solutions and proposals. As we discuss later, the overlay network is fragile to heterogeneous underlying networks with heterogeneous bandwidths. Although the overlay network can be overridden under certain circumstances to improve communications, it is not always possible and performance problems arise since the start of communication. This performance penalty is their main problem and, therefore, we decided to evaluate a basic algorithm to build overlay networks and a few optimizations. With each algorithm version, we performed some tests in simple scenarios with the purpose of getting an approach of the strengths and weaknesses of that algorithms.

3.2 Initial Architecture Proposal

In this section we show an overview of the initial architecture we proposed to build identity-based networks. It aims to resolve many problems found in today's networks design, thus covering many challenges of the Next Generation Internet (NGI). On the one hand, the architecture we propose follows the common idea of locator/identifier separation but, on the other hand, it aims to provide a solution that adds valuable capabilities to the network, such as integrated user/peer identification, by linking the identifier with an actual identity, its validation (authentication), and other security issues such as encryption. In general, this architecture aims to provide the essential mechanisms that are usually implemented in many upper layer protocols and that are likely to be implemented by the remainder protocols in the near future.

First, we propose to use the same identifiers used in the exchanged messages to determine the identities of the parties of the communication. Also, in order to get a service-aware architecture and protocol, we propose to make an overlay network for each service or service-type, possibly with its customized protocol and routing algorithm, so each application protocol of today's design may have its own overlay network in our current design. For instance, VoIP services will have their own overlay network, optimized for VoIP loads and their users behavior. Therefore, we propose to support different but compatible overlay network implementations. Apart from the optimizations, another benefit of this approach is that the users of certain service are logically grouped and users from other services do not interfere them. Moreover, this approach keeps optimized the overlay network size because each network will be composed only by the users that need to be there. Furthermore, this does not avoid different services to share resources, because we propose to have unique identifiers for each identity in all overlay networks. Finally, this design fits perfectly in scenarios in which users may have a different identity for each service.

To get in touch with the architecture we propose, Figure 3.1 shows necessary interactions and involved elements in the exchange of the first message between two parties, Alice and Bob. In this scenario, Alice sends a request message to Bob, which is offering certain service, whether as a service provider (client-server) or as a peer (P2P). Apart from the two ends of the communication, the figure depicts the necessary infrastructure elements: *Service-type Registry* is used to obtain the entry point to the overlay network of a service, *Overlay Peer Network Entry Point* is used to join the overlay network of the desired service, *Identity Providers Infrastructure* is used to request necessary information about

3. Initial Steps: An Overlay Approach

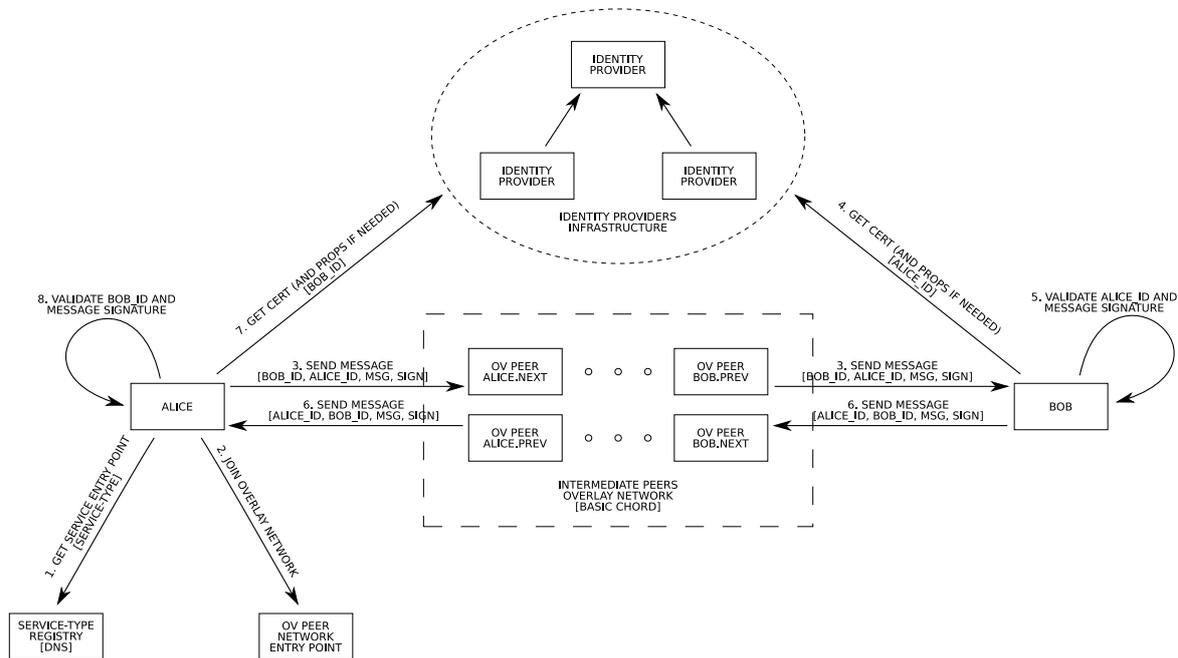


Figure 3.1: First message exchange.

an identity, and *Intermediate Peers* depict the overlay network used in the communication. The process followed in this message exchange is the following:

1. Alice wants to request something to Bob, that offers a service of “service-type”, so it asks *Service-type Registry* the entry point of the overlay network in which “service-type” is offered.
2. Alice sends a join message to the overlay peer obtained in the previous action. Then it is joined to the overlay and, therefore, it knows its neighbors in the overlay network.
3. Alice sends a message to Bob through the first peer of the overlay network. As Bob is already joined to the overlay network, the message goes from one peer to another it is delivered to Bob.
4. Bob requests necessary data from *Identity Providers Infrastructure* to validate Alice’s id and message content. It may also get other information if needed to do its work.
5. Bob validates Alice’s id and message content so now it is sure that the message is sent by Alice and the content is not modified on the way.
6. Bob sends back a response message to Alice, through the overlay network, using Bob’s next node.

7. Alice now gets the necessary Bob's information from *Identity Providers Infrastructure* to validate its response.
8. Alice validates Bob's id and message content and finishes the message exchange.

In the following subsections we show the particularities of this architecture.

3.2.1 Identity and Identifier

The identity of a user represents more than the identifier of a message. It represents one or more profiles with several information items of the user, such as its name, address, credit-card number, security certificate, etc. But the user role is not restricted to real users, it is also played by machines or services, so they have their own identity. User identity is managed by identity providers that keep it secure, under the preferences set by the user. Thus, the identity providers are a very important piece of our initial architecture design, as well as its surrounding technologies such as SAML [79], OpenID [82], and Shibboleth [81].

The identifier is a piece of fixed-size data that identify a user. It can be used to determine who a user is and to obtain information from its identity, managed by its identity provider. In our approach, we decided to start with a friendly name, or well known name, that represents a user or entity. Then we propose to apply a hash algorithm like SHA1 [83] to convert the friendly name into fixed-size data that is used as identifier and as a k key for its identity.

Even though the proposed architecture is heavily based on identity, a user that wants or need anonymity may request an anonymous identity to its identity provider. Also, a user may simply forbid any personal information to be disclosed, so only its hashed identifier can be known. Hence, having identity management infrastructures integrated in the network layer provide many valuable capabilities to the users, applications, services, etc.

3.2.2 Authentication and Integrity

To ensure the identity of the communication peers can be easily verified (authentication) and to ensure the messages are untouched (integrity), we propose to use the user identifier of the source and destination in each message headers and a signature mechanism to sign the whole message. Although it is not recommended, to speed up certain connections, such as the access to public services, peers can avoid identifier verification and even calculation and inclusion of message signatures.

3. Initial Steps: An Overlay Approach

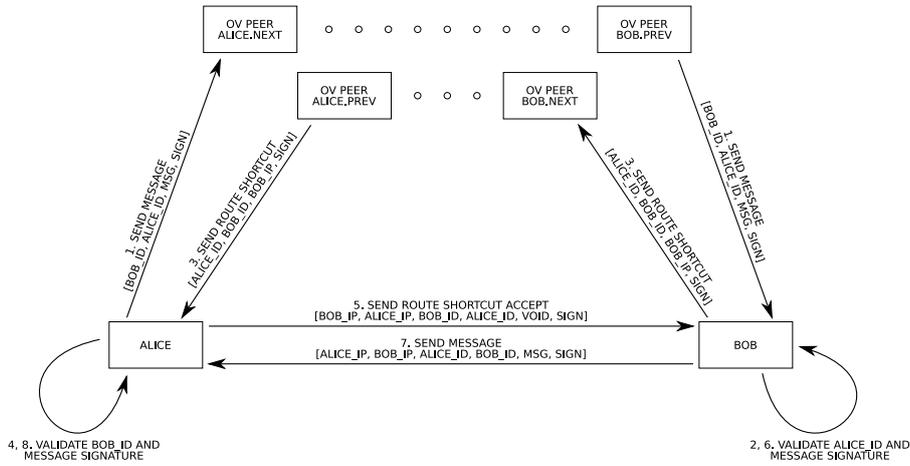


Figure 3.2: Route shortcut.

3.2.3 Overlay Network Routing

Instead of using the overlay network to map identifiers to locations, the architecture we propose uses an overlay network to deliver peer messages. This may impact in communications performance, but we plan to investigate many optimizations to different overlay protocols so they can be used with minimum performance penalty. A key benefit of this approach is that highly changing topologies are inherently supported. Moreover, the actual address and location of a peer is only known by their neighbors, so it adds certain location privacy.

Under certain circumstances and under agreement by all communication parties, location addresses may be used to directly send messages. Figure 3.2 shows the moment in which a communication party decides to use the location addresses (IPs) to their message exchanges. This change improves communication performance, so we let any peer to ask for it whenever it wants.

3.2.4 Underlying Network Routing

The location-based underlying network is desirable to have rigid infrastructure and simple routing. In order to achieve these objectives we propose to organize the address (locator) space hierarchically, using a type of geographic address allocation. This approach prevents excessive growing of routing tables as well as slow routing algorithms. Moreover, also due to the hierarchical address allocation, traffic engineering is better supported.

3.2.5 Overlay/Underlying Network Convergence

In our initial proposal, we considered IPv6 as the underlying network. We propose to use it in many ways. In an very early stage, every device should use the identifier (ID) stack upon the locator (IP) stack. Then, when ID stack is spread to all devices of a network, only the network ingress/egress points must have IP stack. The network will work only with IDs but the messages should be inserted into IP packets if they should cross IP-only networks, as in MILSA AZR or LISP ITR/ETR. Moreover, this opens another benefit to traffic engineering because there can be many paths with different protocols to reach a destination, so the traffic policies may determine the path/protocol to use.

Multihoming is another inherent benefit of using overlay networks. A node that is uniquely identified in the overlay may have allocated many addresses or locators, so a message can be delivered through many different paths. The specific path to deliver the message is determined by simple policies, such as using them in a round-robin fashion or using the underlying network distance to take the decision. Also, the *anycast* mechanism can be implemented using this feature.

Another concern of the network convergence layer is the implementation of the *multicast* capabilities. A simple approach is to define special group identifiers that can be mapped to a group of actual identifiers or identities. As in multicast for IPv6, intermediate routers are responsible of delivering information to all the destinations but avoid the necessity of group management, because they are managed by *multicast identity managers*. Anyway, we should investigate this feature to determine the best solution.

3.2.6 Messages

Message format is a key element of the whole architecture so it may vary over time to support new features. This is an important issue we decided to resolve, support of an extensible message format. In certain aspects, our message design is inspired on HTTP [84] because it supports many features we needed in our message, such as the aforementioned extensibility. To get a more rigid and fixed-size design, some parts of our message header are inspired by IPv6 header design.

First of all, we defined the message format in three parts: header, body, and signature. The header has certain mandatory fields with fixed position and size, such as source and destination identifiers, header length, message type, security modifier, and content length. Then, many other fields of variable size may be added in the header. They are inserted using a key-type-value manner, so they are flexible enough to contain any necessary field.

3. Initial Steps: An Overlay Approach

Certain standardized headers can be included in one field, using a special key and indicating the header-type in the type field. The body of the message contains the payload of the message and has variable size. Finally, the signature is intended to hold the necessary message integrity code.

We defined three message types and three security modifiers. The message types are the following:

- Data messages contain only application data.
- Control messages are intended to change the behavior of the network or a peer, such as the *route shortcut* message or those to establish the signature or encryption algorithm.
- Management messages are used to interact with network infrastructure.

And the security modifiers are the following:

- Plain messages do not contain any signature and are not encrypted.
- Signed messages contain the necessary code to verify its integrity.
- Encrypted messages have the body encrypted and are signed.

3.3 Evaluation and Results

The most complicated and important part of this initial architecture is the overlay network. Without it, the architecture we propose lacks some of its key benefits. Moreover, the overlay network is a delicate piece because its nature could make it unusable in scenarios with many peers connected through heterogeneous performance networks. We researched different overlay network approaches, some of them introduced in Chapter 2, and found the more outstanding approaches in Chord [35] and Kademlia [41]. Initially we chose to explore the possibilities of our approach using Chord because it meets the our requirements, it keeps simplicity, and is well-known among the network research community. Therefore, we implemented a custom simulator in Python to evaluate it and thus get an initial impression of different performance aspects of Chord, a well-known and simple overlay network routing algorithm. Such implementation allowed us to check both overlay network hops and underlying network hops. Then, we applied four simple optimizations to demonstrate that the flexibility of this type of routing algorithm allows us to improve its performance in many ways.

With the overlay network implementations we performed a simple evaluation test that consists in sending a message from each node to every other node. We got the route followed by each message in three ways. First, we got the route followed by the message in the overlay network. Then, we got the route followed by the message in the underlying network. Finally, we got the optimal route using Dijkstra’s algorithm. With all this data we obtained many statistical data: the average, standard deviation, minimum, Q1, median, Q2, and maximum. We also calculated these statistics for the penalty hops that are obtained subtracting the length of the optimal route to the length of the underlying route for each case.

In the following subsection we show the topologies we used to perform the evaluation. Then, we discuss the results obtained with the original and resulting routing algorithms. Finally, we compare the different results and discuss their relations with certain behavior predictions.

3.3.1 Evaluation Network Topologies

In this subsection we show the topologies used in the evaluation. We defined certain scenarios to cover a broad spectrum, from standard local networks to backbone networks.

The first topology, as shown by Figure 3.3, consists in four subnetworks of six nodes. Each subnetwork has all its nodes fully connected and is connected to two other networks through its “router”. Each node is labeled by a letter that differentiates the subnetwork, a dot, and a number for the end peers or two letters for the *switches* (SW) and *routers* (RT). As commented above, the routers of adjacent subnetworks are connected to each other, so each router is connected to all nodes of its subnetwork and to another two nodes. Although all nodes use the same implementation, so they have same capabilities, we decided to label the nodes as the roles they play. The switches are only represented to show that it could be interesting to involve them in the overlay network. This topology let us know an approximation of the behavior of the overlay network in the typical networks found in small-size and medium-size organizations.

The second network topology used in the evaluation, as shown by Figure 3.4, consists in three networks of six fully connected nodes (B, C, D) and a regular backbone composed of six nodes that connects the other networks (A). Although all nodes in the backbone play the router role, we kept the same labels of the previous topology to ensure that the resulting overlay network is the same as in the previous topology. This topology represents a typical scenario in which three small-sized organizations are connected through the same provider, whose routers are connected in a ring with certain shortcuts inside the ring.

3. Initial Steps: An Overlay Approach

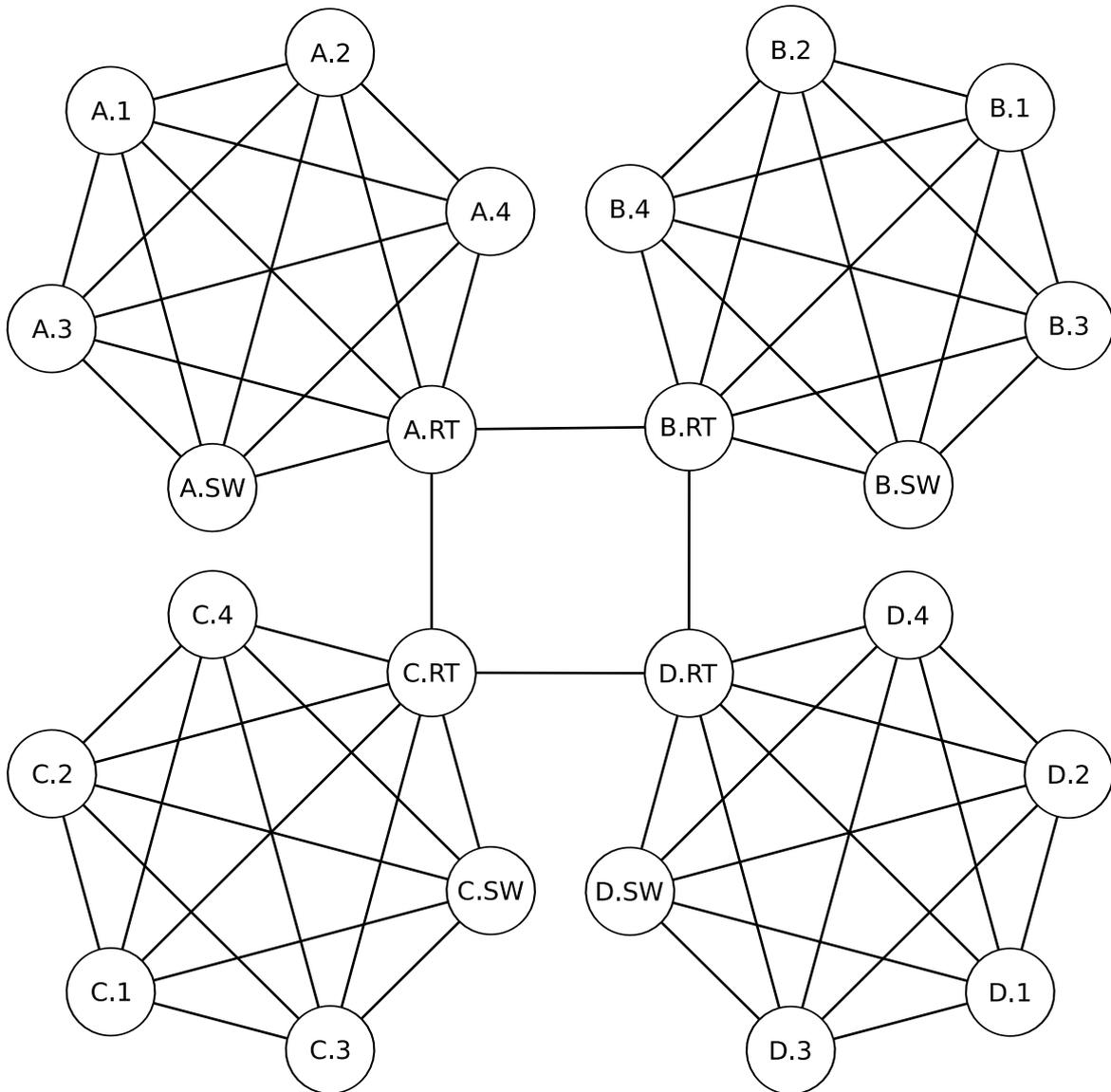


Figure 3.3: First evaluation network topology.

Finally, the third network topology used in the evaluation, as shown by Figure 3.5, is composed of two subnetworks (C and D) with their corresponding routers (B.RT, C.RT) which are connected to two non regular backbone networks with different topologies (A, D). In this topology, as in the previous, we kept the same node labels to retain the overlay network shape. This topology represents the most complex scenario we used in our evaluation. It has two local are fully connected networks (represented in B and C), a typical small-sized backbone (A), and a typical long-distance backbone (D). We intended to show the behavior of the overlay network in a typical network found in Internet.

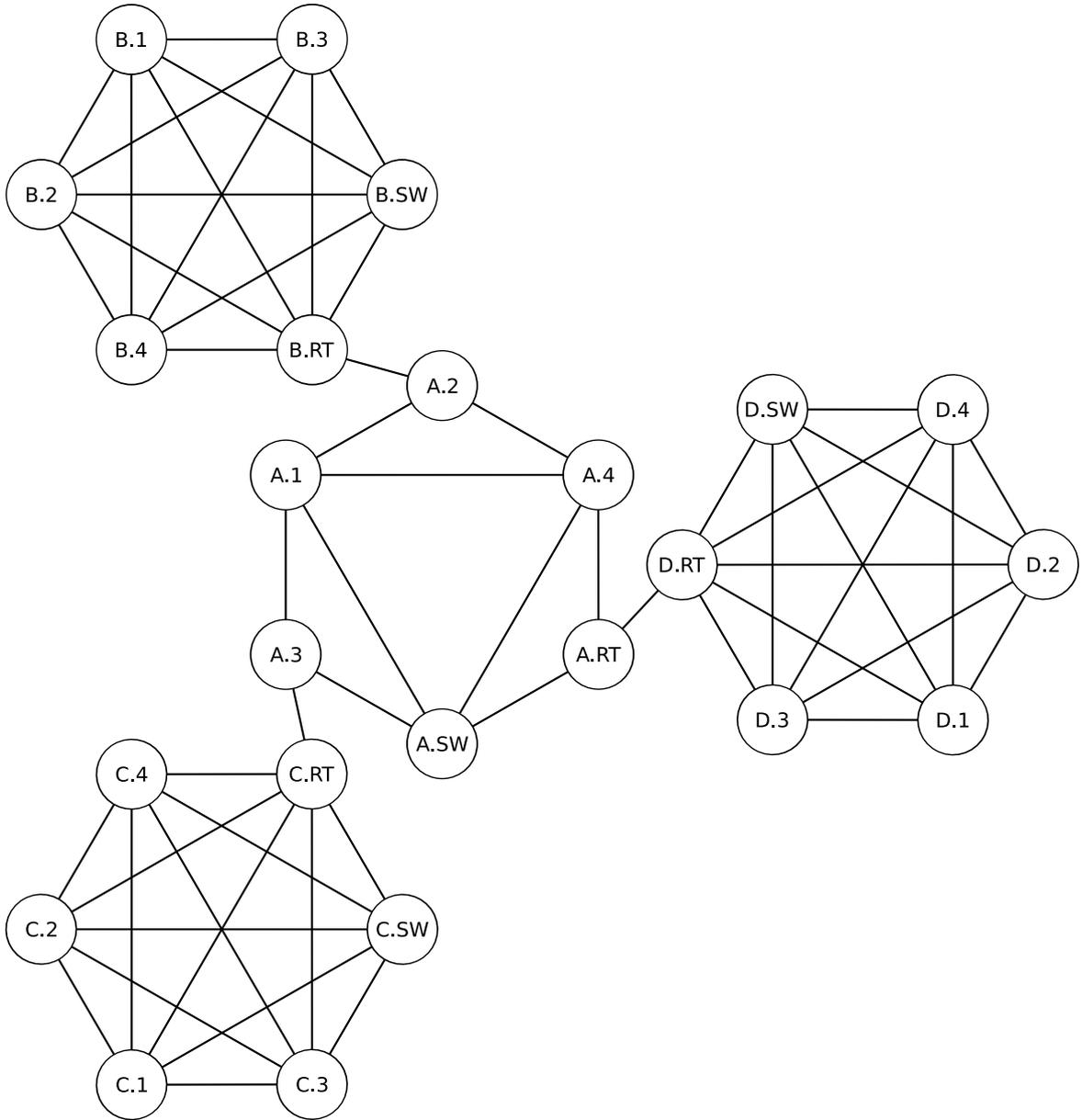


Figure 3.4: Second evaluation network topology.

In the following subsection we discuss the results obtained by running our evaluation with the original algorithm on the topologies described here.

3.3.2 Original Algorithm

As we described above, the original algorithm is based in the mechanism used by Chord to build its overlay network. It consists in building a ring-based topology in which each

3. Initial Steps: An Overlay Approach

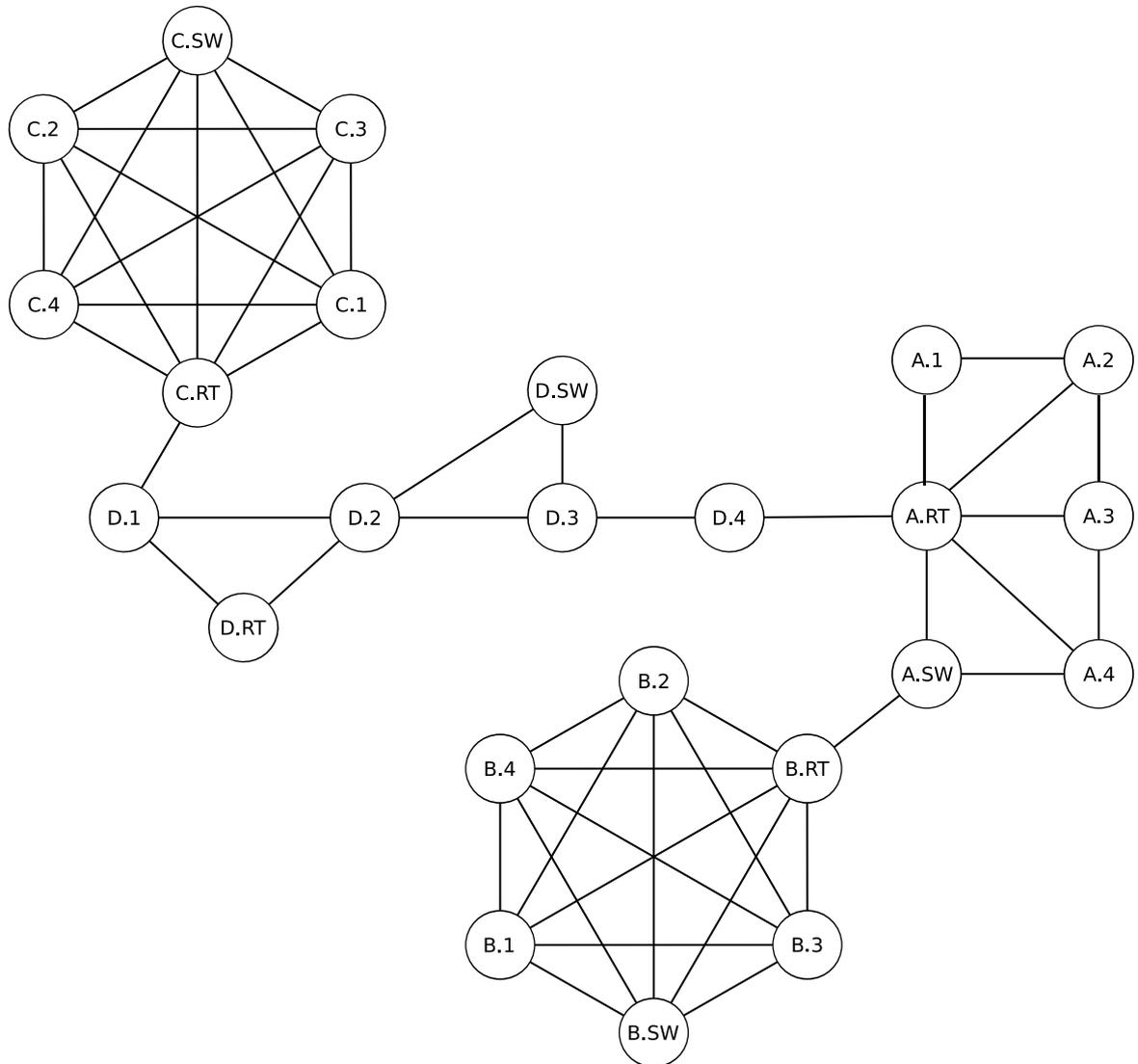


Figure 3.5: Third evaluation network topology.

node is connected to its predecessor and its successor. All nodes are sorted around a ring, using a metric based on their label. For example, nodes can be sorted using the hash value (SHA1) of their labels. This way, observing the ring in clockwise, the predecessor of a node is the node just before and the successor is the node just after it. Chord also proposes to keep a finger table which entries point to nodes beyond the successor, exponentially jumping nodes. To route a message, a node sends the message to the nearest node to the destination from its predecessor, successor and finger table entries. This distance is calculated clockwise using the same metric used to sort the nodes.

3.3 Evaluation and Results

Table 3.1: Routing evaluation results, original algorithm, first topology.

	Avg.	StDev.	Min.	Q1	Med.	Q3	Max.
Overlay network hops	2.77	1.25	1	2	3	4	7
Underlying network hops	6.78	3.36	1	4	6	9	17
Optimal network hops	2.57	1.04	1	2	3	3	4
Penalty hops	4.21	3.38	0	1	4	6	14

Table 3.2: Routing evaluation results, original algorithm, second topology.

	Avg.	StDev.	Min.	Q1	Med.	Q3	Max.
Overlay network hops	2.77	1.25	1	2	3	4	7
Underlying network hops	9.87	5.57	1	6	9	13	26
Optimal network hops	3.70	1.87	1	2	4	6	6
Penalty hops	6.17	5.61	0	1	6	11	25

Table 3.1 shows the results we obtained from the execution of the tests using the original algorithm in the first topology. As shown in the table, even in this simple scenario used for the tests, the underlying network hops are much more than the optimum, being the worst result the difference of the maximum hops, that are 17 hops using the overlay network over 4 hops using an optimal network, which also means a bit more than three times more (325%) of penalty. Another unpleasant result is that, although average penalty hops match perfectly with the subtraction of the optimal hops to the underlying hops, the maximum penalty hops are worst than the subtraction of maximum hops. This means that, in the worst case, using the overlay network may have 14 hops more than the current network, that makes the routing decisions based on optimal paths.

Table 3.2 shows the results we obtained from the tests performed in the second topology, also with the original algorithm. These results keep a similar proportion than the previous, reaching the maximum a 333% more of penalty. It indicates certain level of scalability of the basic overlay routing algorithm. However, the maximum penalty hops value is only at one unit of the maximum underlying network hops, which means that there is at least one path that needs 1 hop and using the overlay network it took 26 hops. This result is not very good for us but we expect to improve it after the optimizations.

Table 3.3 shows the results we obtained running the tests to the original algorithm in the third topology. As in the previous case, these results also keep a similar proportion than the observed in the first case. In this evaluation instance the longest path is 244% longer

3. Initial Steps: An Overlay Approach

Table 3.3: Routing evaluation results, original algorithm, third topology.

	Avg.	StDev.	Min.	Q1	Med.	Q3	Max.
Overlay network hops	2.77	1.25	1	2	3	4	7
Underlying network hops	11.39	6.65	1	6	10	15	31
Optimal network hops	4.34	2.55	1	2	4	7	9
Penalty hops	7.05	6.87	0	1	5	12	30

Table 3.4: Routing evaluation results, first optimization, first topology.

	Avg.	StDev.	Min.	Q1	Med.	Q3	Max.
Overlay network hops	1.99	0.90	1	1	2	3	5
Underlying network hops	4.24	2.57	1	2	4	6	11
Optimal network hops	2.57	1.04	1	2	3	3	4
Penalty hops	1.68	2.05	0	0	1	3	9

than the optimum, so it can reinforce the affirmation that the overlay network has certain type of scalability. Although the topologies are very different, the same thing happened with the longest path. Now, the longest path of the overlay network is of 31 hops and the highest difference between the paths needed by the overlay network and the optimal paths is of 30 hops. Furthermore, these results are almost in the same line than the previous, so we get the same conclusions and expect to improve this behavior with the optimizations done to the algorithm.

3.3.3 First Optimization

In this subsection we comment the first optimization we applied to the routing algorithm used in the overlay network. The optimization is very simple and consists in adding a neighbor table to each node that enumerates the nodes that it may reach in one hop. Then, when the algorithm searches the node to send the message, that is the nearest node to its destination among the nodes it knows. It also takes into account if one of those neighbors is the nearest to the destination, or one of them is just the destination. With this modification we expect to gain performance, reflected in a general reduction of hops and a more stable behavior among the different topologies.

Table 3.4 shows the results obtained in the first topology after the first simple optimization of the algorithm. As we expected, this optimization improved the behavior of

3.3 Evaluation and Results

Table 3.5: Routing evaluation results, first optimization, second topology.

	Avg.	StDev.	Min.	Q1	Med.	Q3	Max.
Overlay network hops	2.01	0.89	1	1	2	3	5
Underlying network hops	5.93	3.89	1	3	6	8	17
Optimal network hops	3.70	1.87	1	2	4	6	6
Penalty hops	2.23	3.23	0	0	1	3	15

Table 3.6: Routing evaluation results, first optimization, third topology.

	Avg.	StDev.	Min.	Q1	Med.	Q3	Max.
Overlay network hops	2.14	1.03	1	1	2	3	6
Underlying network hops	7.24	4.74	1	3	7	10	23
Optimal network hops	4.34	2.55	1	2	4	7	9
Penalty hops	2.90	4.08	0	0	1	4	18

the overlay network, reducing the number of hops in every aspect, including the maximum number of hops. Now, the average path length using the overlay network is only 65% longer than the optimal average path length and the longest path is only 175% longer than the optimal path, so the results are better than with the previous algorithm. Also, it is noticeable that the number of hops of the overlay network itself has been reduced.

Table 3.5 shows the results of the tests of this first optimization with the second topology. Also, as expected, these results are better than those obtained with the original algorithm. Moreover, it is noticeable that in this topology, this algorithm makes the number of hops in the overlay network itself slightly higher than in the first topology but it is still lower than with the original algorithm. The average path length using the overlay network is 60% higher and the maximum path length is 185% higher, therefore, the behavior of this algorithm in the second topology is very similar than in the first topology. This reinforces stability evidence of the routing algorithm.

Table 3.6 shows obtained results from the evaluation of this version of the algorithm in the third topology. In these results, the average number of hops on the overlay network is slightly higher, but still lower than with the original version of the algorithm. Also, the average number of hops in the underlying network is 67% higher than the optimum and the maximum number of hops is 156% higher than the optimum. Both percentages are in the same line as the previous, reinforcing again the stability evidence of this algorithm version. Finally, in this topology, the maximum penalty hops are not so close to the maximum hops

3. Initial Steps: An Overlay Approach

Table 3.7: Routing evaluation results, second optimization, first topology.

	Avg.	StDev.	Min.	Q1	Med.	Q3	Max.
Overlay network hops	1.96	0.90	1	1	2	3	5
Underlying network hops	4.14	2.54	1	2	4	6	11
Optimal network hops	2.57	1.04	1	2	3	3	4
Penalty hops	1.58	2.01	0	0	1	2	9

in the underlying network, so it is an indication of the partial resolution of this problem in this algorithm, which is the worst problems detected in the previous algorithm.

3.3.4 Second Optimization

In this subsection we comment the results obtained with the second optimization applied to the overlay routing algorithm. This second optimization complements the first optimization. It consists in building a routing table on each node that urges it to send all messages to its corresponding router, unless that the message is for one of its neighbors. Also, it urges each router what other router it should use to reach any node. This optimization might be a bit unrealistic because it is too much information to keep in each node, but we can get an approximation of the overlay performance when some routing specific information is taken in account to decide the path to the destination. This approach can be implemented in other ways, for instance it behaves like a hierarchically organized overlay, so it could be a way to get the same results without large routing tables.

Table 3.7 shows the results we obtained in the first topology after applying the second optimization to the algorithm. It is easily noticeable that they are very similar than those from the first optimization with slightly average number of hops. Also, the average path length of the underlying network tests is 61% higher than the optimum and the maximum path length is 175% higher than the optimum. This proportions are in the same line we showed in the previous subsection. Moreover, this algorithm is slightly more stable because almost all penalty hops statistics are very close to 0.

Table 3.8 shows the results obtained from the execution of the algorithm version with the second optimization in the second topology. Above all, they show the same stability of the algorithm than in the first scenario, being almost all penalty hops statistics close to 0. Also, the average number of hops in the underlying network is 50% higher than the optimum, that is the least performance penalty found until now, and the maximum number of hops is 183% higher than the optimum, that is close to the proportions seen

Table 3.8: Routing evaluation results, second optimization, second topology.

	Avg.	StDev.	Min.	Q1	Med.	Q3	Max.
Overlay network hops	1.93	0.85	1	1	2	2	5
Underlying network hops	5.56	3.61	1	2	6	7	17
Optimal network hops	3.70	1.87	1	2	4	6	6
Penalty hops	1.87	2.76	0	0	1	2	14

Table 3.9: Routing evaluation results, second optimization, third topology.

	Avg.	StDev.	Min.	Q1	Med.	Q3	Max.
Overlay network hops	1.99	0.91	1	1	2	3	5
Underlying network hops	6.51	4.24	1	3	7	9	20
Optimal network hops	4.34	2.55	1	2	4	7	9
Penalty hops	2.17	3.47	0	0	1	2	16

after the first optimization. As in all previous results, the worst value is the longest path of the underlying network. This means that, by the moment, it is the most important case we need to improve.

Finally, Table 3.9 shows the results obtained in the tests performed with this optimization in the third topology. As in the last test, the underlying average number of hops is 50% higher than the optimum, but the maximum number of hops is only 122% higher than the optimum, which is, by the moment, the best result in this aspect. Anyway, this is still the weak point of the overlay network. The rest of the results are in the same line as the previous ones, keeping the stability and thus having the penalty hops statistics close to 0.

3.3.5 Third Optimization

The third optimized algorithm is also based on the second algorithm version and consists in making the nodes to keep two finger tables, the one defined by Chord, that points to distant nodes in clockwise, and a new one that points to distant nodes in counterclockwise. Both tables are used to find the nearest node to the destination. With this optimization we expect a huge performance improvement because we are letting a node to know more nodes and, then, the overlay network should need less hops to reach the desired node. Also, selecting nodes in a counterclockwise manner make the most distant nodes to be closer.

3. Initial Steps: An Overlay Approach

Table 3.10: Routing evaluation results, third optimization, first topology.

	Avg.	StDev.	Min.	Q1	Med.	Q3	Max.
Overlay network hops	1.69	0.77	1	1	2	2	4
Underlying network hops	3.71	2.22	1	2	3	5	11
Optimal network hops	2.57	1.04	1	2	3	3	4
Penalty hops	1.14	1.71	0	0	0	2	8

Table 3.11: Routing evaluation results, third optimization, second topology.

	Avg.	StDev.	Min.	Q1	Med.	Q3	Max.
Overlay network hops	1.70	0.76	1	1	2	2	4
Underlying network hops	5.15	3.36	1	2	5	7	17
Optimal network hops	3.70	1.87	1	2	4	6	6
Penalty hops	1.46	2.49	0	0	0	1	14

Table 3.10 shows the results we obtained using the current optimization in the first topology. As we expected, the most significant improvement is in the average number of hops on the overlay network and, therefore, in the underlying network hops. Now, the average underlying network hops is only 44% higher than the optimum, but the maximum number of hops is 175%, what is in the same line of the other tests. This makes us think that the longest paths are not those of the most distant nodes. We must investigate this behavior to identify those paths and possibly make an optimization to eradicate so long paths. Moreover, these are the most stable results we obtained by the moment because of the penalty hops statistics are the closest to 0, in fact, three of the five statistics are 0.

Table 3.11 shows the results obtained in the second topology after the third optimization of the routing algorithm. On it, we can see that the average number of hops in the underlying network is 39% higher than the optimum, which is the best proportion seen by now. Also, we can see that the maximum number of hops is 183% higher than the optimum and is in line with the results obtained during all the tests. This reinforces the idea of investigating how improve those longest paths, because they are confirmed as the weakest point of the overlay network algorithm. In this topology, the penalty hops statistics are even closer to 0, with only two exceptions: 14 for the maximum and 1 for the third quartile (Q3).

Table 3.12 shows the results of the last evaluation test we performed in the third topology with the last optimization. As expected, the percentage of the average number

Table 3.12: Routing evaluation results, third optimization, third topology.

	Avg.	StDev.	Min.	Q1	Med.	Q3	Max.
Overlay network hops	1.72	0.82	1	1	2	2	5
Underlying network hops	6.13	4.07	1	3	6	9	20
Optimal network hops	4.34	2.55	1	2	4	7	9
Penalty hops	1.79	3.29	0	0	0	2	16

Table 3.13: Routing evaluation results, fourth optimization, first topology.

	Avg.	StDev.	Min.	Q1	Med.	Q3	Max.
Overlay network hops	1.50	0.68	1	1	1	2	4
Underlying network hops	3.30	1.96	1	2	3	4	10
Optimal network hops	2.57	1.04	1	2	3	3	4
Penalty hops	0.74	1.40	0	0	0	1	7

of hops in the underlying network over the optimum is slightly higher than in the previous topology (41% vs 39%) but, as with the previous algorithm, the proportion of the maximum number of hops in the underlying network is lower (122% vs 183%). The only reason to explain this is the reduction of the number of links of the third topology compared with the second topology. As also expected, because of the same reason, the average number of hops on the overlay network is slightly higher than in the other topologies. Finally, these results also demonstrate the stability of the optimized algorithm, keeping 3 of 5 penalty hops statistics at the minimum value (0) and one of the remainder two at a closer value (2). This evaluation test also certificate the necessity to investigate the source and possible fix of the longest paths.

3.3.6 Fourth Optimization

Although the previous optimizations used a neighbour table to reduce the path lengths, they did not use that table at the network level but the overlay level. In this optimization we propose to use that information also at the network level, even though it means that every routing decision inspects message IDs and looks for them in their neighbour table, what could not be entirely desirable.

Table 3.13 shows the results obtained in the first topology using the fourth optimized algorithm version. This optimization has improved the overall performance (hop reduction)

3. Initial Steps: An Overlay Approach

Table 3.14: Routing evaluation results, fourth optimization, second topology.

	Avg.	StDev.	Min.	Q1	Med.	Q3	Max.
Overlay network hops	1.63	0.74	1	1	1	2	4
Underlying network hops	4.98	3.30	1	2	5	7	17
Optimal network hops	3.70	1.87	1	2	4	6	6
Penalty hops	1.29	2.39	0	0	0	1	14

Table 3.15: Routing evaluation results, fourth optimization, third topology.

	Avg.	StDev.	Min.	Q1	Med.	Q3	Max.
Overlay network hops	1.59	0.76	1	1	1	2	5
Underlying network hops	5.52	3.57	1	2	5	8	16
Optimal network hops	4.34	2.55	1	2	4	7	9
Penalty hops	1.18	2.48	0	0	0	1	14

in a reasonable proportion. In this topology, the average underlying hops is only a 28% higher than the optimal hops, which is the least performance loss of all the tests we performed. Moreover, the maximum underlying hops are 150% higher than the optimal, which also is the least performance loss of all tests. The penalty hops have been reduced in a significant amount and the penalty statistics are even closer to 0 than in previous tests.

Table 3.14 shows the results obtained with the fourth algorithm in the second topology. As in the previous test, it shows a great improvement of the fourth algorithm. Now, the average underlying network hops is 34% higher than the optimum but the maximum underlying network hops is 183% higher than the optimum. While the former result confirms that the new algorithm has improved performance, the later result is in the same line as the results obtained with the previous algorithms, meaning that the longest paths are still an issue to improve.

Finally, Table 3.15 shows the results we obtained from the execution of the fourth algorithm in the third topology. In this case, the average underlying network hops is only 27% higher than the optimum and the maximum underlying network hops is less than 78% higher than the maximum. These results are the best of all tests we performed, with very significant improvement. Also, these results confirms that, on the one hand, the average underlying hops have been reduced and, on the other hand, the longest paths have been shortened. Even though the maximum path length have become unpredictable, the worst

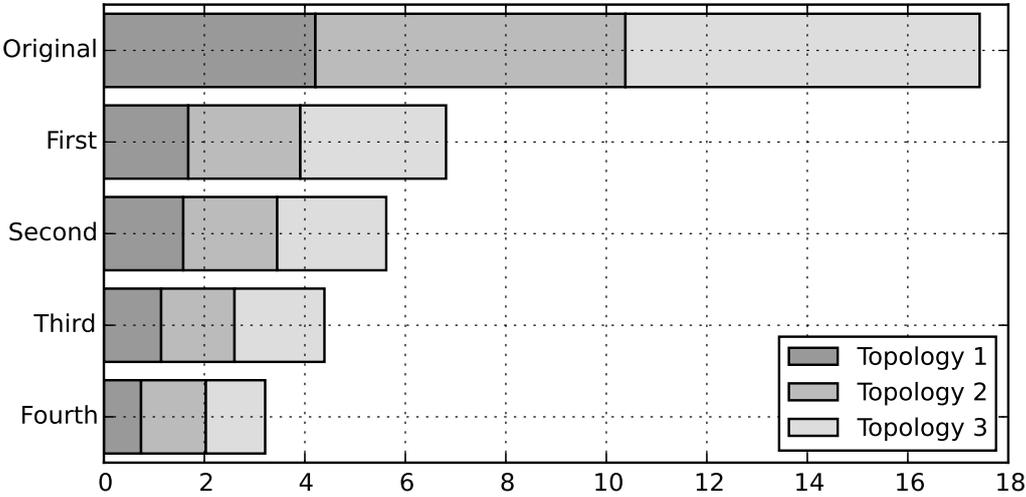


Figure 3.6: Average penalty hops comparison.

result have been in the line of the results obtained in previous tests, only surprising by reducing the maximum path length.

3.3.7 Algorithm and Optimizations Comparison

In this subsection we discuss the overall results introduced in previous subsections. First, we show a comparison of the average penalty hops and maximum penalty hops among the different algorithms and topologies. Then, we show a comparison of the underlying hops. Furthermore, we show a comparison of the average and maximum overlay network hops to get an impression of the improvement obtained in this layer. Finally, we show two figures with the comparisons of the median hops obtained in the underlying and the penalty median hops. Also, we discuss the accuracy of all algorithms used in the tests.

Figures 3.6 and 3.7 show a comparison of the average and maximum penalty hops. As seen in Figure 3.6, the average number of hops have been reduced as the optimizations were applied. The first optimization makes the average number of hops to be reduced to less than the half of the results obtained with the original algorithm. The other optimizations did not get this huge improvement but they still added valuable improvement. A similar behavior can be seen in Figure 3.7, but the improvements in this case are more discreet compared with the previous case. Anyway, the difference of the accumulated maximum penalty hops in the different topologies from the first optimization and the fourth optimization is very noticeable, falling from more than 40 hops to a value near 35 hops.

3. Initial Steps: An Overlay Approach

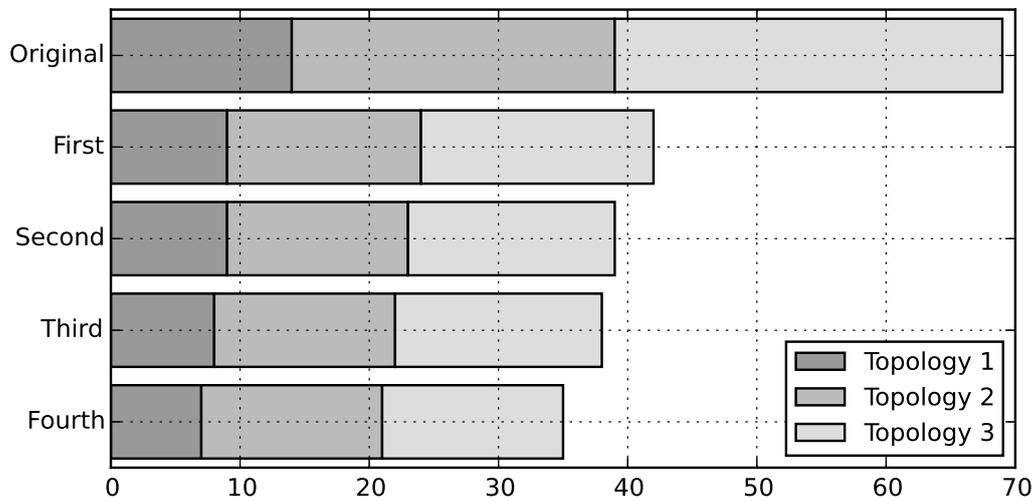


Figure 3.7: Maximum penalty hops comparison.

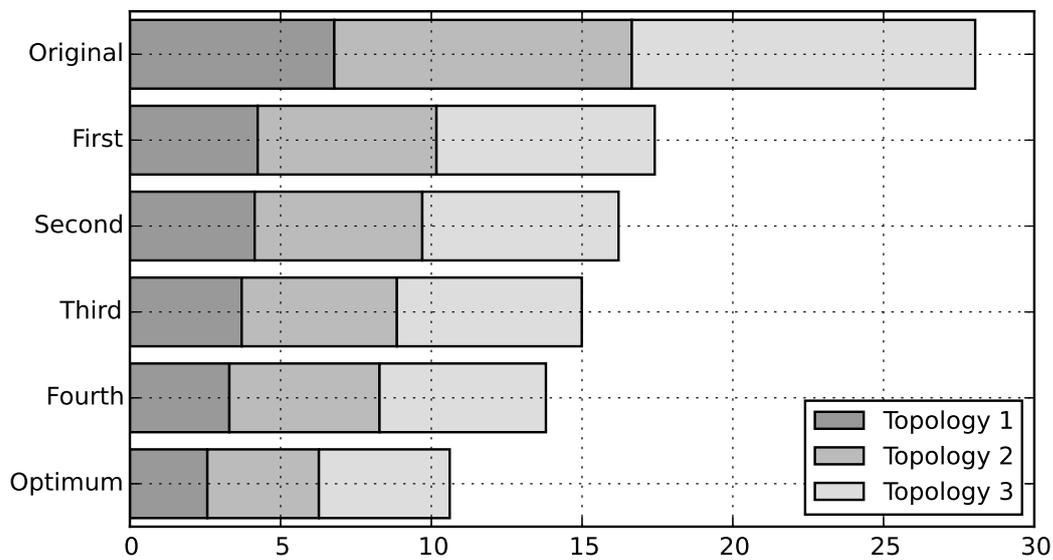


Figure 3.8: Average underlying hops comparison.

Figures 3.8 and 3.9 show the average number of hops seen in the underlying network layer. In this case, all the algorithms are compared among them and with the optimal results. In Figure 3.8 we can appreciate that the latest optimizations are very near the optimum while the original algorithm needs more than two times the hops needed by the

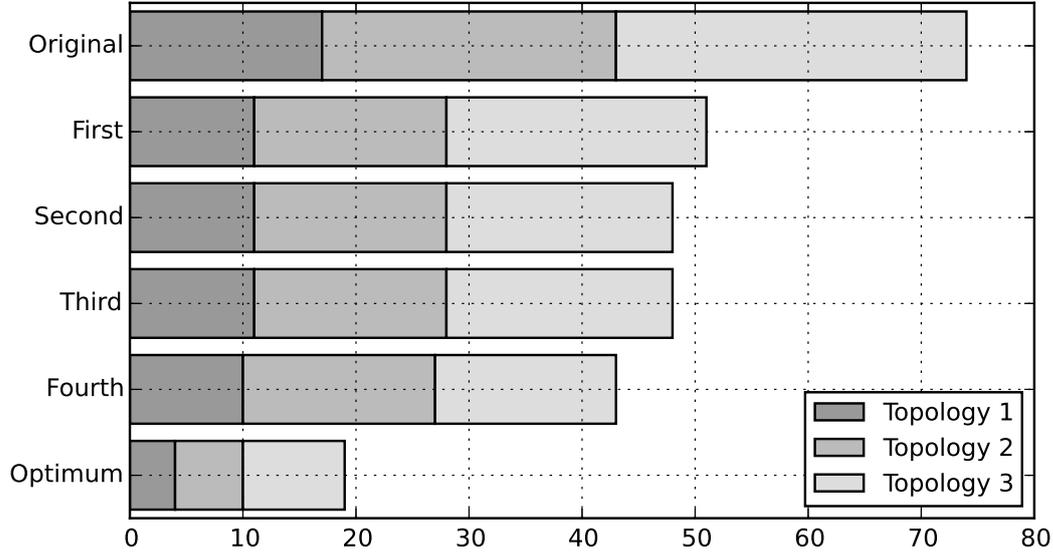


Figure 3.9: Maximum underlying hops comparison.

optimum. Although the different optimizations are very near among them, the difference between the results obtained with the first and fourth optimized versions of the algorithm is close to the difference between the results obtained with the fourth optimized version and the optimum, so the improvements done after the first have achieved the half of the way to reach the optimal hops. As seen in Figure 3.9, the maximum path lengths have different behavior than the average. The performance of the overlay network has been improved with the different optimizations, but it is still very far from the optimum. However, it is closer to the optimum than to the original algorithm. This result also reinforces our recommendation to investigate the different longest paths to see what can be done to get them shorter.

In order to get an approach of the overall behavior of the overlay network layer after the different optimizations, Figures 3.10 and 3.11 also show the added average hops and added maximum hops of the overlay network layer we obtained in the tests with all the topologies. It shows that, even though the improvements have reduced hops of the underlying layer in a respectable amount, in the overlay layer we only achieved a modest reduction. Anyway, we need to consider that one hop in the overlay layer can provoke several hops in the underlying layer, so any reduction is welcome and the improvements we obtained are in that line. In Figure 3.10 we can appreciate that the little reductions have led the number of hops from more than 8 to less than 5, which means an average of more than 3 hops

3. Initial Steps: An Overlay Approach

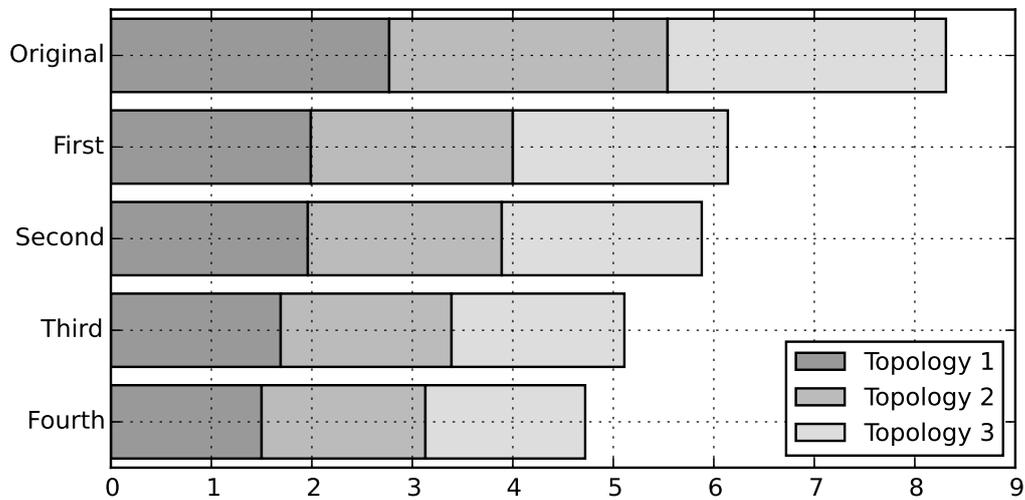


Figure 3.10: Average overlay hops comparison.

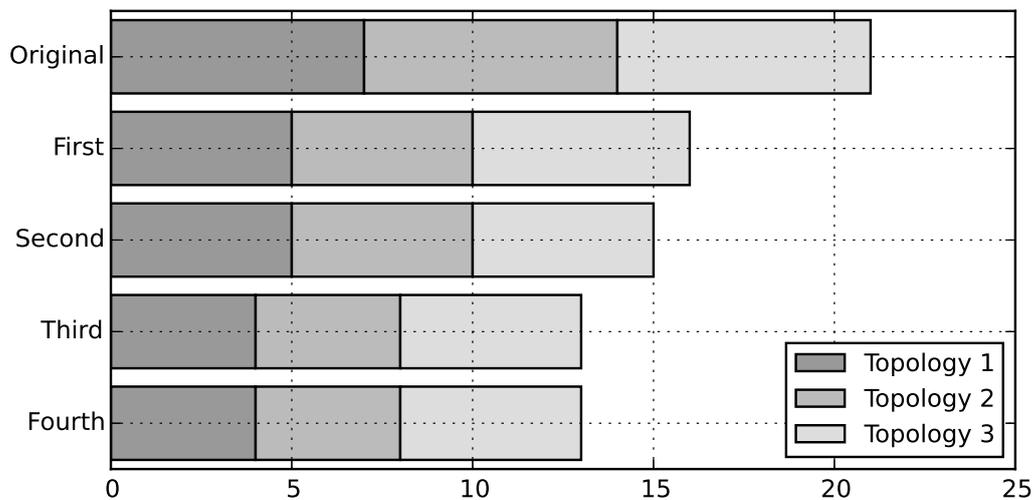


Figure 3.11: Maximum overlay hops comparison.

less. On the other hand, in Figure 3.11 we see a little reduction of the longest paths in the overlay layer. Here, the differences with the original algorithm are more noticeable, but the differences among the results from the optimizations considered are very low.

Figures 3.12 and 3.13 show the median of the hops for the underlying network layer and the penalty hops. As shown by Figure 3.12, we can see that the improvements obtained with the optimization of the algorithm have left the final results very close to the

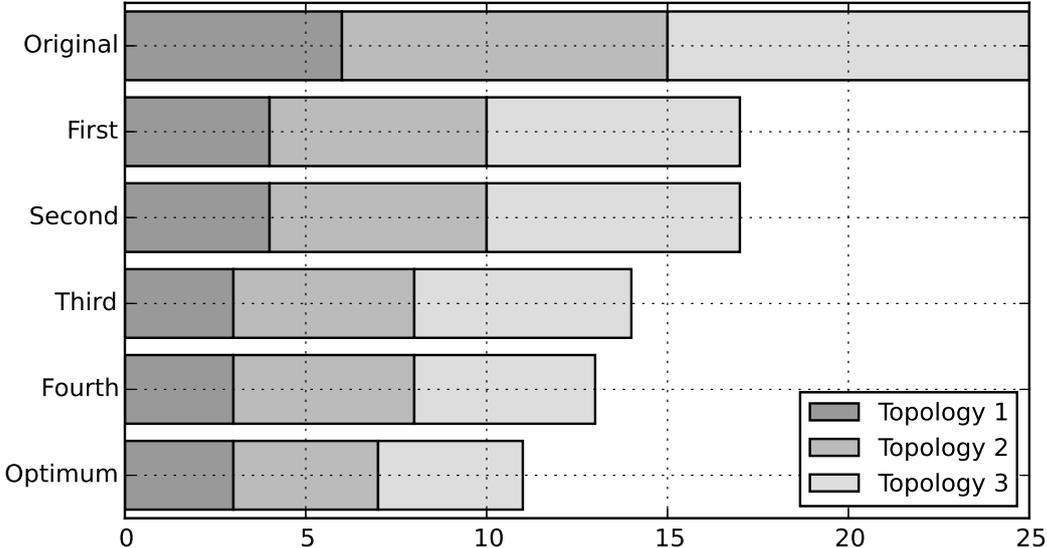


Figure 3.12: Median underlying hops comparison.

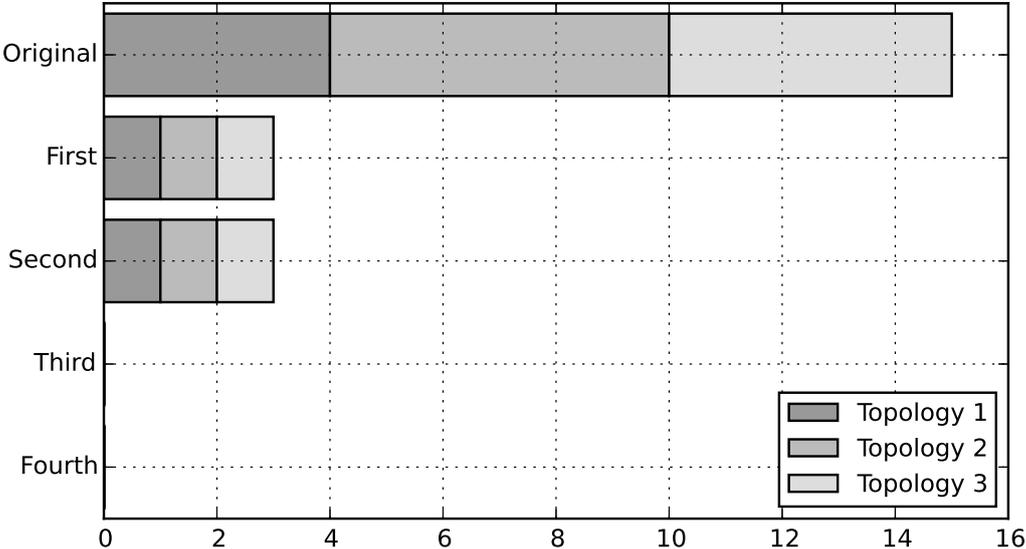


Figure 3.13: Median penalty hops comparison.

optimum. This result, together with the average, indicates that the resulting algorithm of the fourth optimization can be usable in more than the half of the cases. Moreover, in Figure 3.13 we see that the added penalty hops medians of all tests. It shows the huge

3. Initial Steps: An Overlay Approach

Table 3.16: Penalty percentile table.

Algorithm	Topology	0	10	20	30	40	50	60	70	80	90	100
original	first	0.00	0.00	1.00	2.00	3.00	4.00	5.00	6.00	7.00	9.00	14.00
original	second	0.00	0.00	1.00	1.00	3.00	6.00	7.00	9.00	12.00	14.00	25.00
original	third	0.00	0.00	1.00	2.00	3.00	5.00	8.00	11.00	13.00	18.00	30.00
first	first	0.00	0.00	0.00	0.00	0.00	1.00	1.00	2.00	4.00	5.00	9.00
first	second	0.00	0.00	0.00	0.00	0.00	1.00	1.00	2.00	6.00	7.00	15.00
first	third	0.00	0.00	0.00	0.00	0.00	1.00	2.00	3.00	6.00	9.00	18.00
second	first	0.00	0.00	0.00	0.00	0.00	1.00	1.00	2.00	3.00	5.00	9.00
second	second	0.00	0.00	0.00	0.00	0.00	1.00	1.00	2.00	4.00	7.00	14.00
second	third	0.00	0.00	0.00	0.00	0.00	1.00	1.00	2.00	4.00	7.00	16.00
third	first	0.00	0.00	0.00	0.00	0.00	0.00	1.00	1.00	2.00	4.00	8.00
third	second	0.00	0.00	0.00	0.00	0.00	0.00	1.00	1.00	2.00	6.00	14.00
third	third	0.00	0.00	0.00	0.00	0.00	0.00	1.00	1.00	2.00	6.00	16.00
fourth	first	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	1.00	3.00	7.00
fourth	second	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	2.00	6.00	14.00
fourth	third	0.00	0.00	0.00	0.00	0.00	0.00	0.00	1.00	2.00	4.00	14.00

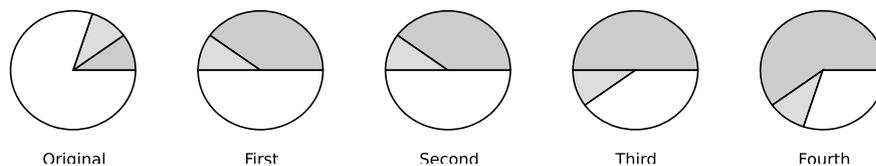


Figure 3.14: Accuracy of the algorithms.

performance improvement from the original algorithm and the first optimized algorithm. It also shows that from the third implementation, the penalty medians are 0, which means that more than half of path lengths were optimum. These results can confirm us that we can concentrate our efforts in reducing the longest paths instead of the shortest.

Finally, Figure 3.14 shows the accuracy of each algorithm. In each pie, darker gray indicates the portion of paths with the same length than the optimum, lighter gray indicates the interval in which the paths can or can not have the same length than the optimum, and white indicates the portion of paths that do not have the optimum length. The portions of the pie are in multiples of 10%. Therefore, the accuracy of the original algorithm is between 10% and 20%, the accuracy of the algorithm with the first optimization is between 40% and 50%, the accuracy of the algorithm with the second optimization is also between 40% and 50%, the accuracy of the algorithm with the third optimization is between 50% and 60%, and the accuracy of algorithm with the fourth optimization is between 60% and 70%.

This information is extracted from Table 3.16 that shows all the 10% percentile of penalty hops for each testing scenario and algorithm.

3.4 Conclusions

In this chapter we presented an initial architecture to build identity-based networks that uses an overlay network to transport the messages among communication participants. Our objective here is to validate the overlay network mechanism as feasible to allow identifier-based communications, ensuring scalability, and achieving enough flexibility to become the basement of our final architecture for the Future Internet. To demonstrate this we have evaluated some routing algorithms based on Chord, applying incremental optimizations to them. The results from the evaluations are very promising, supporting our claims. In most cases, the performance of the overlay network can be close to the performance of the underlying network, so it can be used with little penalty by different functional blocks of the architecture we present in this thesis.

One problem we found in all algorithms, even the optimized algorithms, is that they have certain large paths that can make them unusable. For example, Figure 3.15 shows the path followed to reach the node A.4 from the node B.2 with the fourth optimized version of the algorithm and in the third scenario. The route in the overlay network layer is [A.4, C.SW, B.2], the route in the underlying network layer is [A.4, A.RT, D.4, D.3, D.2, D.1, C.RT, C.SW, C.RT, D.1, D.2, D.3, D.4, A.RT, A.SW, B.RT, B.2], and the optimal route should be [A.4, A.SW, B.RT, B.2]. This path is so long because there is no node between A.4 and C.SW that is nearer to B.2 than C.SW, if so, the code used in some of our optimizations would be executed and the path would be short-circuited, thus reducing its length, occasionally to the minimum length. This case is one example of the cases that require more interaction with the underlying network to improve the performance of the overall system.

To sum up, in the following chapters we will deepen all the aspects discussed in this chapter, deriving our initial architecture to build the necessary functional blocks to achieve the objectives discussed in Section 2.4. Moreover, we will evolve the routing algorithm described here to achieve better reliability and performance. We will replace Chord by Kademia, as it is more stable and provides a stronger and widely-used mechanism to build the overlay network.

Chapter 4

Digging Deeper: Evolving Functional Blocks

In previous chapters we have described the most important requirements for the network architecture of the Future Internet (FI), we have reviewed the most influential approaches, we have stated the missing functions, and we have described an initial architecture that tries to cover the gaps. In this chapter we delve into the problem and present the definition and evaluation of different functional blocks that will be incorporated into the final architecture, as described in Chapter 5.

The remainder of this chapter is organized as follows. First, in Section 4.1 we contextualize and justify the design of the functional blocks that will compose the final architecture proposed within this thesis. Then, in Section 4.2 we introduce the Domain Trusted Entity Infrastructure (DTEi), which is the architectural component that promptly provides the identity-based functions to the network. It is deployed on each network to manage the interactions with the identities of such network and protect their privacy and overall security. As a main requirement of the DTEi, we discuss the design of the Identity-Based Network Protocol (IBNP) in Section 4.3. It specifies the necessary message exchanges to enable different entities contact each other in terms of their identity so they can establish communication sessions.

In Section 4.4 we discuss the design of the approach we have researched to provide an identity-based discovery function to the architecture we are building. It enables network entities to find each other in terms of a partial identity, which is a partial set of the attributes of the objective identity. To get this, the approach we propose makes extensive use of overlay networks, as we have proposed throughout the development of the thesis, because of its scalability and decentralization qualities. In Section 4.5

4. Digging Deeper: Evolving Functional Blocks

we present the ontology used to structure the identities in the discovery functions. In Section 4.6 we describe how to effectively encapsulate the proposed functions into a separate identity-based control plane, which facilitates other network architectures to incorporate the functions provided by our architecture.

Once we have described the design of the functional blocks we proceed to evaluate their main capabilities. In Section 4.7 we analyze the security of IBNP and IBCP to demonstrate they provide the claimed security level. In Section 4.8 we evaluate the performance of the relevant components of the architecture to demonstrate their qualities and feasibility. Finally, in Section 4.9 we conclude this chapter and summarize our contributions so far.

4.1 Introduction

The main outcome expected from the research work carried out during the development of this thesis is the provision of the necessary functional blocks to overpass the ossification of the architecture behind the Internet by increasing the flexibility and granularity of network endpoints and providing effective identification functions with the aim of identifying *network objects* (persons, devices, software, etc.) unequivocally and which operate independently of the underlying network technologies. Thus, in this chapter we discuss the design of the necessary components, protocols, and network mechanisms in general required to achieve such objective.

We claim throughout this thesis that removing the fixed concept of endpoint, which today resides in the *host* component or element and which is represented by its network address (e.g. IP and MAC addresses), and introducing a flexible structure will be an important step towards such objective. We propose to raise the role in the network of *identities*, seen as attribute sets, as effective communication endpoints, being the identification mechanism that will be used by *network objects* to establish their sessions and make communications effective in the network.

Allowing network objects (persons, devices, software, etc.) to reach each other in terms of their identities while respecting their privacy and overall security has been a key and transversal requirement that we have imposed to all the functional blocks. Moreover, respecting the essential qualities of the practical network has been both a design principle and an invariant of our designs. This includes to avoid any mechanism that may interfere with *dynamic multihoming* operations or even provide explicit support for them.

Figure 4.1 depicts the global view of our objective, including the functional blocks we have designed to achieve it. The first functional block we have designed has been the

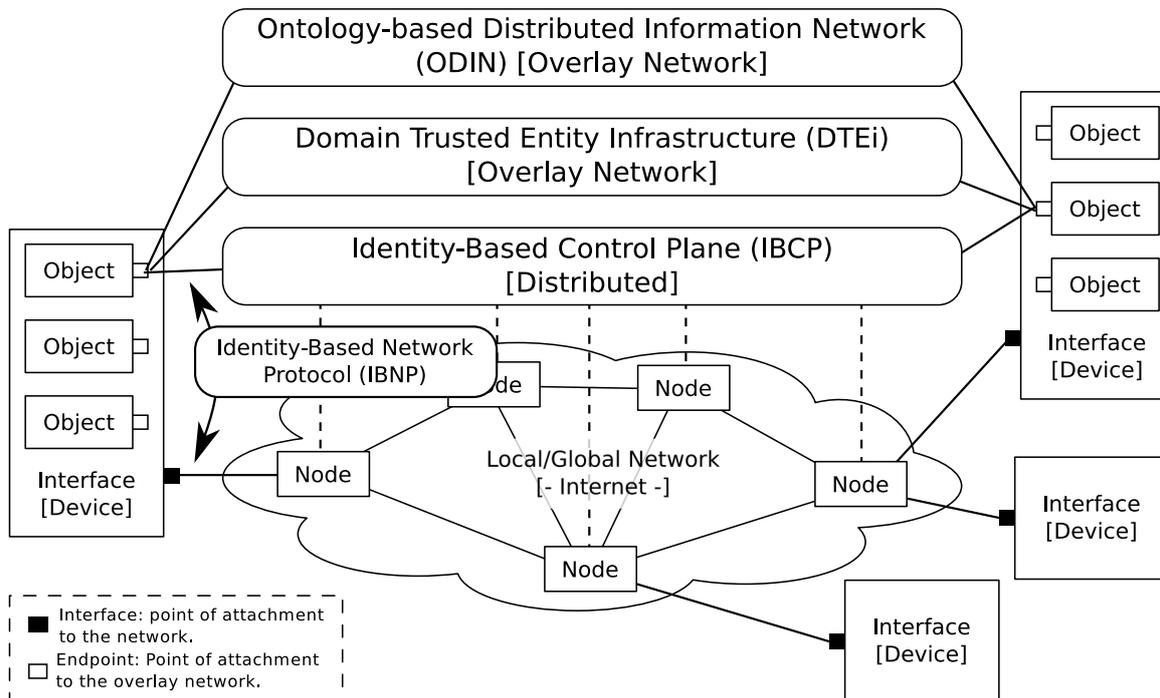


Figure 4.1: Global view of the objective and the functional blocks.

Domain Trusted Entity Infrastructure (DTEi). It formalizes the concepts presented as our initial approach in Chapter 3 but also considers not just the network independence requirement but also identification and security. While in our initial approach we identified network objects by their plain identifiers, effectively separate from underlying network addresses, the DTEi uses full identities to achieve the identification function. Moreover, the DTEi squeezes the possibilities offered by overlay networks to organize the space of network nodes in separate network or administrative domains by introducing a trusted element into each domain that interconnects with the elements of other domains to enable network objects to reach each other in a secure and trusted way.

Once we defined the DTEi we found the need for a proper protocol for network objects to interact each other and with the DTEi, for them establish and manage communication sessions. Therefore we designed the Identity-Based Network Protocol (IBNP). It is the basis for network objects to authenticate into the DTEi, discover the capabilities of other objects, request the establishment of communication sessions, and finally exchange messages with them. This protocol also ensures the secure and private communication among network objects, avoiding to reveal unnecessary or forbidden information about them. Therefore, we also demonstrate that the proposed protocol performs well and provides the required security capabilities.

4. Digging Deeper: Evolving Functional Blocks

Although IBNP includes some support for discovery, it does not define how to perform such discovery. Here is where the Ontology-based Distributed Information Network (ODIN) enters into the game. This functional block enables network objects to find each other in terms of their identity but just providing a vague or incomplete description. We have used the term *partial identity* to differentiate an incomplete attribute set from a full identity that has all attributes of a network object. ODIN also makes heavy use of overlay networks, so its nodes can reside and collaborate with the nodes of the DTEi. However, it also uses the overlay network to build an effective Distributed Hash Table (DHT) for storing the public attributes from network objects that are used by the search function. In order to ensure that network object descriptions are consistent we have designed an ontology, by extending other ontologies. Furthermore, we have evaluated both ODIN and the proposed ontology to demonstrate their qualities and behavior.

Finally, in order to integrate the functional blocks and qualities provided by our architecture with other network architectures we decided to design a control plane that gets the functional blocks already defined, extracts the relevant control operations, and puts them together. Thus we designed the Identity-Based Control Plane (IBCP). Its architecture is parallel to the architecture of the DTEi. In fact, the DTEi is used to provide the infrastructure that supports the control plane but we defined a new style of interaction, modeled to reflect the current trends in separation of control and data planes, especially represented by the Software Defined Networking (SDN) movement. In the following sections we delve the discussion regarding the design of the components and their evaluation.

4.2 Domain Trusted Entity Infrastructure

The basic idea behind the locator/identifier separation in network protocols is to differentiate the location (*where*) from the entity (*who*) that is taking part in the communication. But the architectures following this idea are finally associating an entity with the particular device it is using to communicate, which means that actual identity behind the device is not considered enough and that its identifier is somehow disclosed.

We overcome such limitation by avoiding the manifestation of different problems, such as traceability and overall privacy, and misses the opportunity to involve the actual identity of the different parties in the communication operation. In this section we discuss the design of the Domain Trusted Entity Infrastructure (DTEi), which is integrated into the network in order to build a trusted and scalable communication infrastructure that

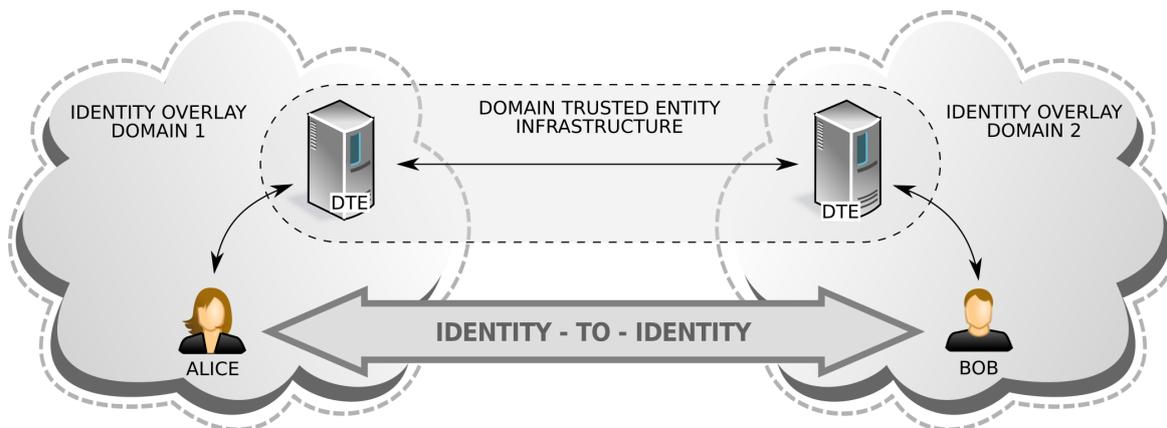


Figure 4.2: Overview of the general architecture.

enhances privacy, prevents network object traceability, and involves them into network control operations. It also provides the basic security measures to make an step forward towards a complete architecture for the FI.

4.2.1 Architecture Overview

As introduced above, with the introduction of the DTEi, the architecture evolution we propose in this section keeps raising the identity concept to be a central element of the network while taking into account inter-domain scenarios and their implications on security. As other aforementioned architectures, we support the common idea of locator/identifier separation but we want to exploit the evolution opportunity to introduce new capabilities like inherent authentication, inherent mobility support, and others.

Figure 4.2 shows the general shape and objective of our architecture. We first depict the DTEi, which is the resulting infrastructure of interconnecting separate trusted nodes residing on separate domains to build an *identity plane* that allows entities to address each other by means of an intuitive, identity-to-identity approach. IP addresses are no longer used identify entities, since they may change dynamically during a session. Second, we propose to use an *evolved routing scheme* supporting routing for identity-based communication. Third, we need an intelligent *control plane* to manage domain-to-domain issues. This intelligence would be distributed across the network to ensure end-to-end communication is made possible by negotiating any agreements needed between pairs of different administrative domains.

As we did with the initial architecture presented in Chapter 3, to implement the control architecture and hence the DTEi, we propose to use an *overlay network* like Chord [35]

4. Digging Deeper: Evolving Functional Blocks

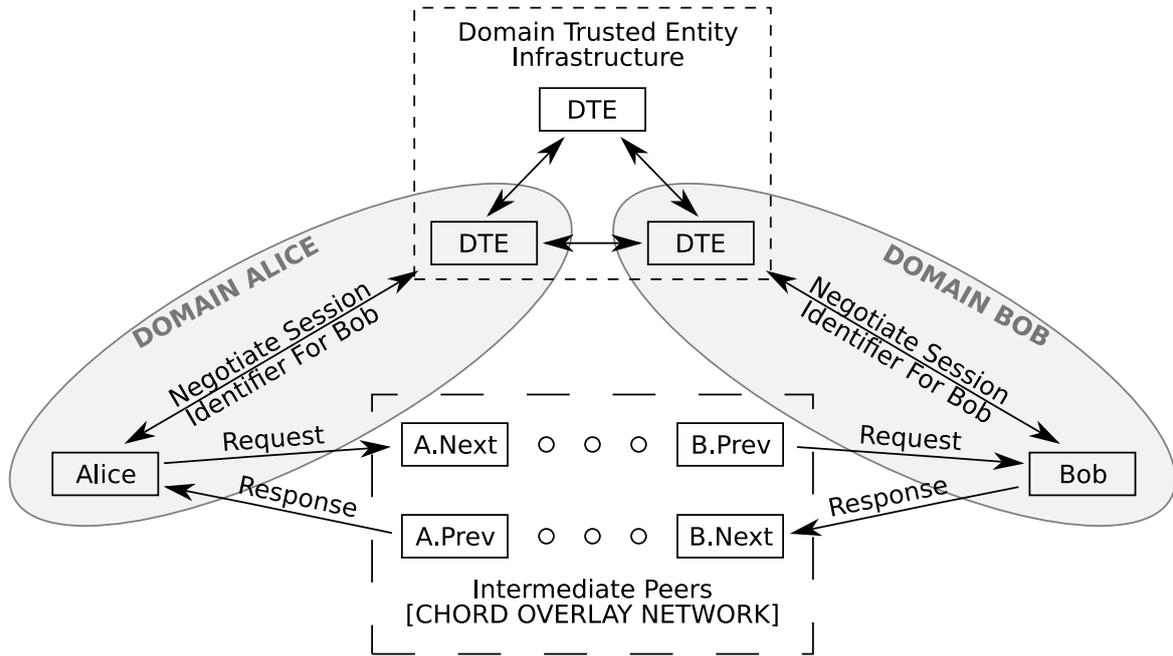


Figure 4.3: Architecture instantiation on top of Chord.

or Kademia [41], since they offer a totally distributed mechanism to support high number of interactions and traffic. With them we can address each entity by its identifier instead of a locator. Then, we divide the whole identity space in many *identity domains*, so each entity belongs to a domain. Then a node of the DTEi is introduced into each domain to hold and protect the identity information of the entities bound to such domain. The DTEi infrastructure is formed by connecting together the nodes of different domains using security mechanisms like cryptographic certificates. Finally, we make the entities to use the DTEi to privately negotiate different identifiers (one for each endpoint) in each different session. This last requirement provides traceability prevention because those identifiers are only known by communication parties and their corresponding nodes of the DTEi, so they can not be associated to the actual entities by others.

We decided to instantiate the DTEi on top of current IP-based networks by using Chord, the overlay routing mechanism. Figure 4.3 shows the shape of the resulting approach. On it we show the interactions needed when Alice (a peer) wants to communicate with Bob (another peer). First, Alice negotiates the session identifier that Bob is going to use in the communication. To do that, Alice requests that operation to its corresponding node of the DTEi, providing its previously generated session identifier, that it is going to use in the conversation. Then, the node sends the same request to the node of the DTEi residing in Bob's domain, that in turn sends it to Bob.

Once Bob receives the message, it generates its session identifier and sends it back to the node of the DTEi in charge of its domain so it can be delivered to Alice. The whole operation is completely ruled by policies, if one of the exchanges is forbidden by a policy, the session is not started. Once Alice receives the identifier that Bob is going to use in the session, it sends its request through the overlay network that is going to deliver it to Bob. Finally, Bob sends back to Alice its response using the same overlay network. Because of the nature of this type of overlay networks, the messages can take different routes and it is hard to trace them. This increases the privacy but the key point is that those session identifiers can not be associated to Alice or Bob.

In contrast with LISP [29] or HIP [85], which were described in Chapter 2, our architecture is completely separated from the underlying network. On the one hand, LISP offers a simple way to improve routing scalability by a map-and-encapsulate mechanism that interacts with network elements and, on the other hand, HIP uses resolution mechanisms to get a locator from an identifier. To do this, our architecture builds an overlay network in top of one or more underlying network architectures that lets it to directly deliver messages (packets) only based on identifiers. Below we outline the qualities of the resulting architecture.

4.2.2 Identities and Identifiers

As considered throughout this thesis, the architecture we propose emphasizes the differentiation of *identity* and *identifier*. In contrast with other interpretations [37], we meet with the ITU-T definition of identity on its X.1250 recommendation as follows: The representation of an entity in the form of one or more information elements which allow the entity(s) to be sufficiently distinguished within context. For identity management purposes, the term identity is understood as contextual identity (subset of attributes), i.e., the variety of attributes is limited by a framework with defined boundary conditions (the context) in which the entity exists and interacts. Thus, each entity is represented by one holistic identity, which comprises all possible information elements characterizing such entity (the attributes). However, this holistic identity is a theoretical issue and eludes any description and practical usage because the number of all possible attributes is indefinite.

On the other hand, also meeting with its ITU-T definition, we consider an identifier to be a piece of fixed-size data that identify something. In a general sense, this architecture uses identifiers to determine the endpoints of communication parties, as well as to obtain information from the identity if permitted. Nevertheless, they are not used to

4. Digging Deeper: Evolving Functional Blocks

unambiguously associate an identity to an object on time, just in certain moment and communication event.

Moreover, as described in [27], “*An Identity refers to the abstract entity that is identified. An Identifier, on the other hand, refers to the concrete bit pattern that is used in the identification process*”. In our approach, an identity represents a person, machine or service that has many attributes, such as its name, address, etc. Each identity is managed by a trusted entity controlling its corresponding home domain that keeps it under the preferences set by its holder. The attributes of the identity may be distributed in different places. Therefore, when a client asks for an attribute, this entity either obtains the attribute and gives it to the client or let the client know how to get it. Thus, the *domain trusted entity* is a very important element in our architecture design.

4.2.3 Authentication and Integrity

To authenticate the peers and to ensure message integrity, we propose to use identifiers themselves and certain signature mechanism to sign the whole message. Therefore, once messages are validated against their signature (integrity), the identity of a peer is recognized from the identifier of the messages (authentication). Although it is not recommended or expected, peers can avoid calculation and verification of message signatures and even the session negotiation to speed up certain communications, such as the access to public services.

The DTEi is also used to validate that the identifier (or identifiers) used by a network object. Thus, any communicating party can be sure that it is talking to the entities it wants to talk without knowing any attribute of the actual identities behind them. Again, this functionality is also controlled by policies, so some entities may decide to forbid the validation. Furthermore, when an entity requires anonymity, it may request an *anonymous identity* whose identifiers can be validated and whose attributes can be requested, but the actual information of the real entity is not disclosed in any manner.

4.2.4 Convergence of the Overlay and Underlying Networks

On the one hand, instead of mapping identifiers to locations, we propose to deliver control messages directly through an overlay network with the necessary optimizations to prevent performance degradation. As we consider that the FI will be formed by many heterogeneous network architectures, we propose to build the overlay network in top of them and, occasionally, make them cooperate.

4.2 Domain Trusted Entity Infrastructure

On the other hand, this architecture needs, in one way or another, a special underlying network infrastructure that is capable to deliver messages using identifiers instead of network addresses. When being instantiated on top of an address-based network architecture, it should allow the reservation of many network addresses from the same device. For instance, both IPv4 and IPv6 supports this feature, but current IPv4 based network infrastructures obstruct and/or forbids this operation, so we can only consider IPv6 as a direct underlying network. Thus, the architecture can be instantiated on top of different underlying network infrastructures and, occasionally, allow the coexistence of multiple underlying architectures that can cooperate to support complex scenarios.

Many other network architectures are better suited for our architecture than current IP infrastructures. First, we have the overlay network protocols used in many DHT infrastructures, as used in Chord because its simplicity and its performance improvements, such as the Lookup-Parasitic Random Sampling (LPRS) [36] and Relaxed-2-Chord [86]. We also considered other interesting overlay networks that may provide performance improvements, such as Kademlia or Cyclone [87]. Then, we have other interesting protocols coming from content-centric or publisher/subscriber network architectures, such as CCN [15] and PSIRP [88], which were analyzed in Chapter 2.

While using an overlay network, the actual address and locator of a peer is only known by their neighbors. Nevertheless, under certain circumstances and under agreement by all parties, the underlying network may be directly used to exchange messages. Also, using an overlay network permits the underlying network to retain its rigid infrastructure and, when it is address-based like IP, the routing may be simplified because its address space can be hierarchically organized, thus preventing excessive growing of routing tables and providing better support to traffic engineering. We consider IPv6 as a transition underlying network protocol, so when ID stack is spread to all devices, messages should be inserted into IP packets when crossing IP-only networks, as in MILSA AZR [46], LISP ITR/ETR [28], or Inter-domain Rendezvous in PSIRP.

4.2.5 Security of the Infrastructure

As the DTEi will be the central element of our final architecture we need to take special care about its security and trust. First, in order to ensure the confidentiality, the communication between entities and the nodes forming the DTEi are encrypted with a key negotiated during the authentication, which can be renewed when necessary. This also applies to the communication between DTEs.

4. Digging Deeper: Evolving Functional Blocks

On the other hand, to ensure a trusted path between pair of nodes of the DTEi, each node has its own public/private key pair that is distributed to the other nodes when it is instantiated in the overlay network. This may be, a priori, a heavy task but it is necessary to ensure the overall security. A dynamic trust model, like a Web of Trust (WoT) or a Public Key Infrastructure (PKI) is then used to maintain those keys and ensure the global security. Putting all together, the DTEi offers the secure and trusted channel required by the architecture and protocol to operate correctly.

4.2.6 Provided Security

Instead of hiding the identity information of an entity, this architecture offers the possibility to access it in a controlled manner. Thus, the DTE is responsible of managing the identity information, so others may ask it to find out if an entity is exactly "who" it claims to be. Also, we can consider that an entity is *authenticated* just by validating that the identifier (or identifiers) it is using belongs to it and ensuring the integrity of the messages exchanged with it, which is done by a signature (or token) field included in the messages. Therefore, our architecture and protocol provides integrated authentication of all communication ends as well as message integrity.

At the networking layer, the privacy of an entity can be violated by guessing the identity that is behind a device/host just by linking/tracing its network operations. In order to prevent this problem, the architecture described here permits arbitrary changes of session identifiers. This capability does not affect communications because they are bound to *identities* instead of *identifiers* or *addresses*. New session identifiers are negotiated through the DTEi, which is a totally secure and trusted channel, so attackers can not follow the data flow to guess what identity is behind a concrete identifier. An open issue for general security could be that with this model an attacker can not be identified, but it is not a new issue as soon as any communication must be negotiated before taking place and the DTEi has (and protects) the binding between identities and identifiers. It is able to log them for subsequent cyber-crime investigations, just as the same way it is done today.

Finally, our architecture proposes and recommends to use an asymmetric encryption mechanism to give confidentiality when needed. It may be inefficient and processor hungry but with obvious benefits over weaker encryption mechanisms: 1) Transmitted information will be kept secret for longer; 2) There is no need to negotiate the security terms, with the speed-up it represents; 3) Fits perfectly and performs much better in publisher/subscriber underlying networks. In the future, processor performance improvements may make those methods much more feasible. This does not prevent our architecture to adopt symmetric

mechanisms and key exchange protocols such as IKEv2 (Internet Key Exchange Version 2) but they are out of the scope of the current work.

4.3 Identity-Based Network Protocol

Continuing the work discussed in the previous section, in this section we show how to complete the proposed DTEi with a proper protocol that allows entities to address each other as well as leveraging entities with identity-based communication sessions. It is worth to remember that the resulting architecture from the work carried out so far already provides complete location independence, has general support for the mobility of communication parties, prevents traceability, and is designed to work over different underlying network infrastructures. Apart from describing the operation of our architecture and protocol, we evaluate its security capabilities and other important aspects.

As a side effect, the resulting architecture inherently supports authentication and mobility of the entities involved in the communication. Moreover, it is designed to be agnostic to any underlying network infrastructure and can be used to enhance them with reduced penalty, which makes it a perfect component to take its features to existing networks without defining a brand new transport layer. We also show the successful verification of the protocol security and demonstrate its feasibility and scalability showing its behavior when instantiated on top of two different underlying architectures. First, we show how to instantiate the architecture in top of the CCN infrastructure, explaining how to build a test application based on the architecture. Second, we show how to instantiate the architecture on top of XMPP. We also compare the solutions to demonstrate the flexibility of our solution.

4.3.1 Network Model Shift

It is clear that the network model used in the Internet is not valid in the long term to support new services and workloads. Therefore, we encourage a shift in the network model by the design of the Identity-Based Network Protocol (IBNP). Although there are several architectures targeting the same objectives than ours, they lack in securing the entities behind communications, which is an important challenge for the FI [1, 12], as we stated in Chapter 2. We also consider them more than important, essential, for the network of the future and thus here is where our newly defined protocol and architecture go into action. These essential capabilities are:

4. Digging Deeper: Evolving Functional Blocks

- Clear protection of privacy with particular focus on traceability prevention.
- Secure identification of the entities that take part of a communication.
- Utilization of roles as communication endpoints, instead of hosts or persons.
- Dynamic management and provisioning of identity attributes in security negotiations.

These capabilities are desirable in any network architecture, specially those trying to overcome the problems of the current Internet, including ICN. This is because, even though a piece of data can be secured through cryptographic mechanisms, the source and the piece of information itself can be easily followed during a communication, so an attacker (or not so attacker) may know what information is accessed by a client, thus violating its privacy.

It has been discussed that most architecture proposals for the FI do not provide mechanisms to prevent the traceability of communications, so the privacy of the communication parties can be violated. Those architectures lack in the integration of security to the network layer and do not separate the actual entities behind the communication from the devices they are using, so an entity is associated-to and identified-by its device. These properties are widely accepted as important challenges for the FI so we consider that the identities of the entities involved in communications should be treated in a special manner and the traceability of their operations should be prevented, thus protecting their privacy.

On the other hand and from the content-centric perspective, the content distribution mechanisms of the Internet have evolved from a centralized model to a modern and massively distributed model that is reflected in current Content Delivery Networks and Peer-to-Peer (P2P) networks. In addition, the definition of new approaches is motivated by the spreading of high quality content of the Web and the augmentation of the bandwidth provisioned at the edge network [89]. Our work here also target content delivery so here we also cover some related proposals.

Once stated the qualities of previous work and the lack of current architectures and proposals to correctly address privacy at the network level, in the following subsections we discuss the required evolution of the architecture we are building in this thesis to fill the mentioned gaps, including the protocol used on it and the evaluation of both.

4.3.2 Architecture Overview

The main contribution of the present section to the overall work carried out in the thesis is the design of a protocol for the architecture integrated with the DTEi that fills the

4.3 Identity-Based Network Protocol

gaps found in current Internet model in terms of loc/id separation, mobility support, and integrated security. These capabilities are delivered in an integrated manner with the concept of secure identity-based end-to-end communications (identity-to-identity) which, by means of the architecture described here, is carried to the network. Thus, apart of loc/id separation, it provides scalable mobility support and integrated security, with special attention to privacy and traceability prevention. In order to achieve this objective we build an identity overlay network whereby entities are addressed by their digital identity, instead of logical address of the device (or host) they use. This overlay network is then divided in many domains of trust which are independent of the actual networks. Each entity is associated with a domain and can have different devices connected to different physical or logical networks at the same time.

To achieve these capabilities, the architecture incorporates many elements and mechanisms. We retake Figure 4.2 from previous section, which shows an overview of the architecture with its main elements, leaving out the lower layer networking infrastructure used by the devices of the communication parties. The most important elements of the architecture are the entities participating in the communication, which can be people, software (services), and hardware (machines), among other things (in reference to the Internet of Things [90]). The DTEi manages the association of entities and identifiers for its domain and permits communication parties to be sure they are *talking* to *who* they want without revealing identity information. This is achieved by just asking the DTEi to validate an identifier against a query of identity attributes, which is an abstract representation of an identity and thus a communication endpoint. It can also be used by other elements to obtain certain identity attributes if allowed by policies. DTEi nodes of different domains are connected forming an infrastructure that supports and protects the identity of the communication parties. Finally, the underlying network infrastructure is used to transmit low-level messages among communication parties.

With the integration of IBNP, communications are established through endpoints that are used in message exchanges and are identified by location independent identifiers. If the underlying network is based on addresses, this architecture requires to allocate many addresses to be associated with the different identifiers which can be dynamically negotiated through the DTEi. Also, it permits to change any endpoint identifier at any time, so the mobility support is inherent and the privacy can be enhanced with arbitrary identifier renegotiation.

At the identity level, the protocol defined here will manage identities and build identifiers with Extensible Resource Identifiers (XRI) and Extensible Resource Descriptor Sequence (XRDS) [91]. XRI is used to build the identifiers and XRDS is used as resolution

4. Digging Deeper: Evolving Functional Blocks

mechanism to dynamically associate the identifiers to an identity and vice-versa. This way, our architecture has a consistent and coherent identifier scheme that may be coupled with existing identity federation architectures, such as OpenID [82], using adaptation mechanisms. Thus, IBNP is able to integrate other identity management technologies like Security Assertion Markup Language (SAML) [79] and collaborate with other systems like Shibboleth [81]. The protocol may rely on the DTEi to store the XRDS documents that belong to its entities and which other entities may request, so they do not have to talk each other directly. The XRDS document describes the services offered by an entity and how they can be contacted, their service endpoints. Therefore, the DTEi plays the role of the XRI/XRDS resolution infrastructure in the OpenID architecture.

4.3.3 Protocol Description

In the previous section we discussed the details of the DTEi and the qualities it offers to the network but, as it just concentrates on the infrastructure and how DTEi nodes interact each other. Here we describe the protocol used by network objects to communicate with other network objects, which is not bound to any underlying network architecture. By other means, both the full architecture researched in this thesis and the protocol specified are so generic that they can be instantiated on top of many network architectures.

When the protocol is tied to the DTEi, it is composed of four main operations: 1) The authentication of entities into their corresponding nodes of the DTEi; 2) The search and discovery of the target/destination entities (communication counterparts) through the DTEi; 3) The establishment of a communication session between communication participants, also through the DTEi; and 4) The direct exchange of data messages without using the DTEi. All control messages (object-to-DTEi-node, and object-to-object) are exchanged through the overlay network, so, as discussed above, they are independent of the addressing and delivery mechanisms of the underlying network architectures.

To describe the protocol operation we use a simple scenario in which two entities start a *conversation*. As shown by Figure 4.4, the scenario is composed of two entities (peers), Alice and Bob, which are from two different domains, domain 1 and 2, with its corresponding node of the DTEi. Alice belongs to domain 1 while Bob belongs to Domain 2. The four main operations of the protocol are detailed below.

First of all, each entity (peer) authenticates into the node of the DTEi controlling its domain and registers the XRDS document that describe its exposed facets, each one with its own identifier based on XRI. Those facets, also called virtual identities, represent the entities during communication acts to protect their actual identities.

4.3 Identity-Based Network Protocol

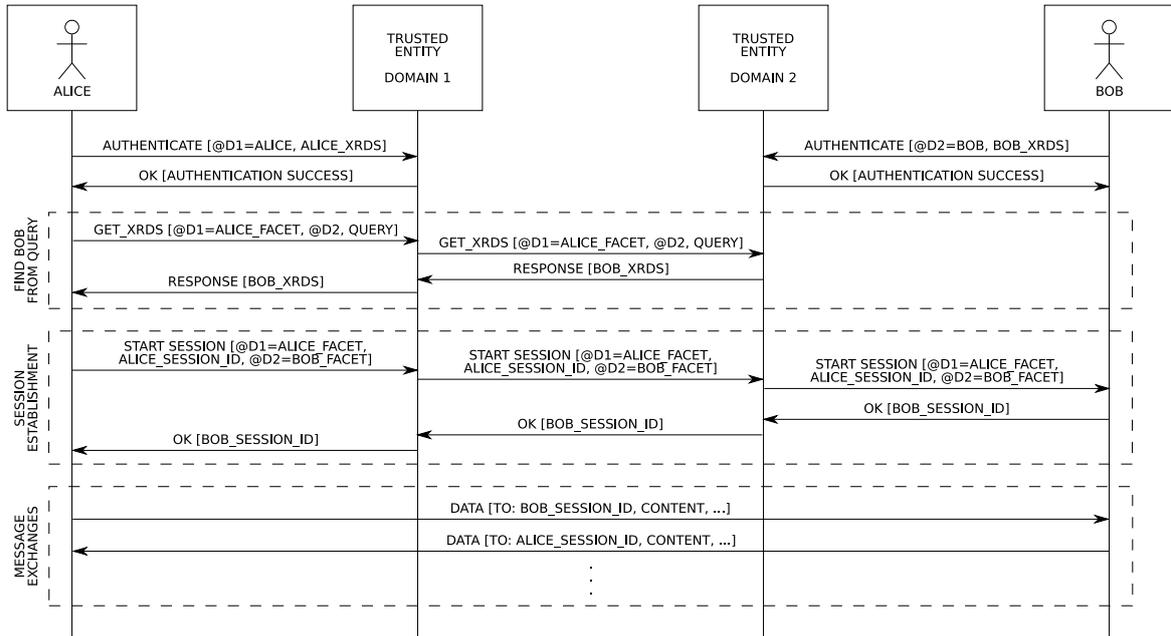


Figure 4.4: Identity-Based Network Protocol.

The integrity and confidentiality of all messages is correspondingly ensured with a signature or encryption mechanism that is independent of the message content. The authentication message (*AUTHENTICATE*) has the client identifier of the entity that is being authenticated encoded in XRI together with the XRDS data that describes the services (facets) offered by the entity. The authentication response (*OK*) only contains the verification result of the identifier (*AUTHENTICATION SUCCESS*). The *GET_XRDS* message contents are the XRI-encoded identifier of the service (facet) used by the initiator (*@D1=ALICE_FACET*), the destination domain also encoded in XRI (*@D2*), and a query to search *Bob* in terms of its identity attributes (e.g. *And(Equals(Attribute("email"), "bob@mail.com"), GreaterThan(Attribute("Age"), 27))*). The query language definition is outside the scope of the discussion carried out in this section. The *RESPONSE* message has a subset of the XRDS document of *Bob* with the services (facets) allowed to the faced used by *Alice* to resolve the query. The *START_SESSION* message content has the facet used by *Alice*, like in the previous message, the session identifier that is going to be used to identify it in the session, and the selected service (facet) from the previously received XRDS document. The response (*OK*) message includes the session identifier to be used by the counterpart (responder) that here is *Bob*. Finally, the *DATA* message contains the destination identifier and the payload. It does not contain the source identifier because

4. Digging Deeper: Evolving Functional Blocks

each identifier is associated with only one session, so from the destination identifier the responder can extract the source identifier.

After the authentication process, and being registered the services offered by each entity, Alice sends a request to the node of the DTE_i controlling its domain with a query to get an XRDS document that describes some facets (specified by the query) of Bob, providing the facets Alice wants to use in the associated subsequent sessions. Then, the node controlling domain 1 will contact with the node controlling domain 2 to forward the query and thus get the response with the XRDS document of Bob which is then sent by the node controlling domain 1 to Alice. The response of the node controlling domain 2 is conditioned to the policies set by Bob to regulate the access to its information.

Once Alice gets the XRDS document from Bob, she knows the identifier of the facet (virtual identity) she wants to communicate with (“@D2=BOB_FACET”), which will be usually associated with some service she wants to consume. Now, to start a session, Alice allocates a new session identifier and sends a *start session* request to the node of the DTE_i controlling its domain, indicating its *source* facet, its session identifier, and the selected facet from Bob. The node controlling domain 1 will forward this request to the node controlling domain 2, which will check the policies to know if the communication is allowed and, if so, will forward the request to Bob. Then, Bob allocates a new session identifier and sends its response to Alice through the DTE_i by sending the response message to the node that controls its domain.

Finally, Alice receives the *OK* from its *start session* request and knows the session identifier used by Bob, so she will proceed to send data messages to Bob through its session identifier (BOB_SESSION_IDENTIFIER) and will expect to receive data responses from Bob through its session identifier (ALICE_SESSION_IDENTIFIER). This is done in this way because a message just includes the *destination* identifier, which represents a source and a destination (session direction), so it improves the protocol efficiency.

We should notice that both Alice and Bob may be instantiated by many elements. For instance, when certain content is going to be delivered to many clients (broadcast), the role of Bob is played by the content source and the role of Alice is played by the multiple clients that will receive the content. In this case, the negotiation is performed by the nodes of the DTE_i controlling the domains of the involved entities. When instantiated on top of architectures like CCN, the role of Bob is also played by the intermediate equipments involved in the content delivery.

Mobility

The protocol defined here, IBNP, has inherent mobility support because communications are bound to identities instead of identifiers or locators (addresses). This means that communications at the application level are totally independent of the network attachment point of the device/host and the mobility effort falls on the network infrastructure.

Regarding actual mobility management, when the architecture is instantiated on top of a locator-independent underlying infrastructure, such as Chord or CCN, the IBNP stack just reports to the underlying infrastructure the new attachment point of the device when it moves from one edge network to another. In contrast, when the architecture is instantiated on top of a locator-dependent underlying infrastructure, the end-nodes collaborate among them using IBNP to establish a new identifiers and new locator (address) to maintain communications but the applications or existing sessions are still unaffected.

Application Message Exchanges

Since our architecture provides endpoint semantics and permits services to have their own identifiers, it may be directly used by applications and services to exchange their messages, reducing the final layers used in communication. For instance, in a SOAP (Simple Object Access Protocol) [92] based application, each layer introduces its own headers and message format so it is difficult to take full communication control from the application layer and also makes it difficult to apply traffic engineering because communication semantics are hidden in upper protocol layers. On the contrary, using our protocol the messages are directly delivered through the network, so it is simpler and traffic engineering may easily consider the application level.

Message Format

Applications in the FI will require extensive use of metadata in their communications, so the message format must be extensible to support an arbitrary number of fields but keeping mandatory the minimum necessary fields, such as the source and destination identifiers, and the content.

Our architecture has defined a flexible message format, as commented above. Thus, applications may include specific headers into network messages so identity infrastructure is able to correctly, securely, and efficiently deliver them. Also, other information may be introduced to be used by endpoints, so applications get a fine control over their messages. Finally, actual messages can be instantiated in many low-level message representations

4. Digging Deeper: Evolving Functional Blocks

that may need specific headers, as name/value/field-separator, JSON (JavaScript Object Notation), XML (Extensible Markup Language), and binary.

4.4 Network Object Discovery

The discovery of network objects (content, services, devices, persons, etc.) is an important challenge to face when researching towards the FI. In this section we give an overview of the challenge regarding network object discovery for the FI, proposing a new discovery architecture (functional block) that provides mechanisms to achieve network object discovery from partial object descriptions, building an overlay network to organize object descriptions and using strong statistical mechanisms to find them from partial descriptions. With a prototype implementation of the architecture we demonstrate experimentally its feasibility and general behavior in different scenarios.

4.4.1 Objective

In the FI, the necessity of a better discovery mechanism increases significantly [93] and thus many challenges have appeared to the search and discovery. Today, the general search and discovery is usually performed through well-known search engines but they require human interaction and are not well suited for network layer operations like found in Machine-to-Machine networking. In such scenarios, the mechanisms are based on simplistic mapping from a *name* to a *value* and this does not cover the diverse requirements of the FI.

A proper discovery approach for the FI should provide mechanisms to find a network object of any kind (content, device, person, etc.) by a vague or near complete description. A plain identifier or name is not enough because it cannot represent the actual properties of a network object.

Based on our knowledge and observing the operation of existing techniques, mainly deployed and proposed architectures for network object discovery, here we claim that it is feasible to achieve robust, scalable, and flexible network object discovery from partial object descriptions (partial identities) by organizing object descriptions in a DHT using a variation of the Kademlia overlay routing algorithm [41], also used in the DTEi and IBNP. We switch from Chord to Kademlia because the former is more suitable for academic research while the latter has been used in production to build DHTs in several solutions, such as *Bit Torrent*. The keys used to store objects will be the resulting bit-string of the *Bloom Filters* [94] that represent the object descriptions as attribute/value sets (identities).

To demonstrate our claims, in this section we discuss our proposal for a new search and discovery architecture called ODIN, standing for Ontology-based Distributed Information Network. It will become a crucial functional block in the final architecture built as a result of this thesis and it supposes three main innovations. First, it makes use of ontologies and semantic techniques to allow network participants to search network objects in a natural way, by shaping their knowledge about a desired object into a partial description. Second, using *Bloom Filters* to represent the identities (attribute sets) it successfully obtains representative identifiers for the identities and thus provides a feasible mechanism to accomplish semantic search in overlay networks using just one key and resolve the queries in just one iteration through the network. Third, by including semantic technologies in the network layer, it allows the discovery architecture to evolve without varying the underlying mechanisms and keeping full compatibility.

It is worth to mention that a similar objective has been targeted by other architectures for the FI, such as NetInf [54], as we discussed in Chapter 2 but they lack in the definition of specific mechanisms to achieve content discoveries from arbitrary attributes or partial descriptions.

4.4.2 Architecture Overview

The Ontology-based Distributed Information Network, in short ODIN, is a global network infrastructure that manages linked descriptions of network objects. Such descriptions are formed by any kind of meta-data about network objects (content, services, devices, persons, etc.). Network object descriptions must follow some ontology, which is indicated within the description. A main (root) ontology is provided with ODIN to serve as skeleton to hold description items from other ontologies. Moreover, an object description is treated as the identity of such object, using the identity definition from ITU-T X.1250 [13]. Furthermore, the architecture accomplishes with the design principles for the FI, as defined in [5], so it will have a strong design basement.

The discovery process used by ODIN receives a partial description and returns a set of descriptions matching it. Both partial and complete descriptions are represented in Turtle [95], which is a serialization format for RDF graphs. The partial description is translated into a partial identity (attribute set), which is in turn used to perform the search in ODIN. The main (root) ontology is restricted to a few and simple RDFS [96] definitions with the objective of minimizing the concepts introduced and encouraging the reuse of already defined ontologies. Our ontology basically defines a new class called *NetworkObject* that inherits from the class *Object* defined by other ontology, DOLCE

4. Digging Deeper: Evolving Functional Blocks

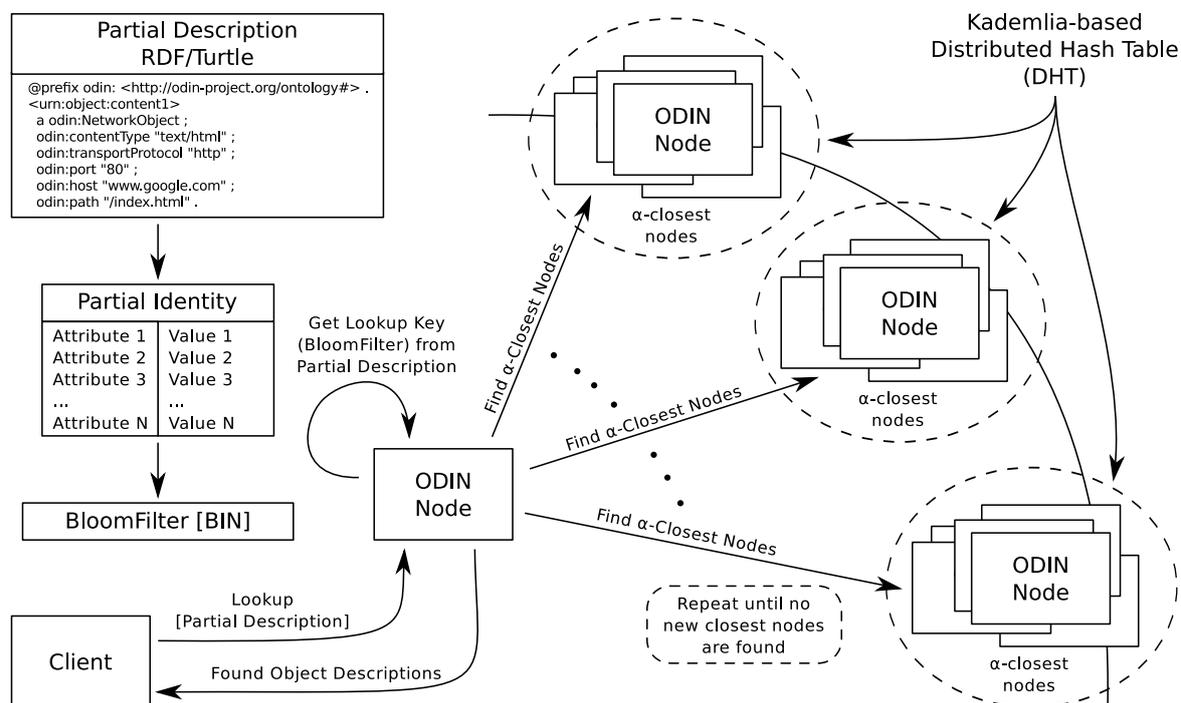


Figure 4.5: Overview of the lookup operation in ODIN.

UltraLite (DUL) [97]. This increases the chance of reusing other concepts from DUL, as well as the interconnection of ODIN objects with objects from other ontologies, such as Friend of a Friend (FOAF) [98] or SPITFIRE [76].

The basement of ODIN is the overlay network and DHT built by ODIN nodes, which are deployed into separate administrative or network domains. This overlay network is built with a variation of Kademlia [41] and it is used to build the DHT which stores the network object descriptions. The key associated to each object description is obtained by building a Bloom Filter [94] with its attributes, so similar object descriptions retain close keys when using a XOR metric as found in Kademlia.

To build the key for an object, we first represent its description as an attribute set (identity). Then, each name/value pair of the identity is added to a Bloom Filter (BF), represented as a 160-bit string and built with 4 different hash functions. As elements inserted to a BF can be queried for presence but not retrieved, the resulting string represents the object identity without revealing any of its attributes.

In the following subsections we describe in detail the publication and lookup/discovery procedures provided within ODIN.

Object Publication and Search Procedures

Clients publish their object descriptions by contacting with their corresponding discovery servers, which are part of the global DHT formed by ODIN. Then, the discovery server will take control of the operation and will ensure that the descriptions are available for other clients to find them. The search procedure is quite similar. Clients send partial descriptions to their corresponding discovery servers, indicating they want to know the complete description, and the discovery servers will return back the complete object descriptions they find.

From the perspective of the discovery servers, both operations are similar. Once they receive the description to register or the partial description to lookup, they obtain the *Bloom Filter* (key) and traverse the DHT to find the α -closest nodes to such key. At this point, the operation differs. If it is a register operation, the discovery server will send the received description to the α -closest nodes. In contrast, when the operation is a search, the discovery server will request all descriptions that match (exactly or partially) with the key obtained from the partial description. Below we discuss in detail the lookup operation.

For instance, as depicted in Figure 4.5, if a publisher (e.g. *Google*) wants to publish a content (e.g. its *index.html* webpage), it has to first build a description for it, based on the ontology detailed below. Then, the ODIN component in the client side will translate the description to an identity, which is a set of attribute/value pairs, and apply the *Bloom Filter* technique to get an associated key that will be used to index such description in the DHT. While the details of the identity will be kept by the node of ODIN assigned to the publisher, the corresponding node in the DHT (more than one if replication is enabled) will have an entry that associates the key with the node that holds the identity. When a requester wants to request such content, it builds a partial description, which is similar to the original description but not complete because a requester may not know all attributes. Then, a similar procedure to the publication is followed to build the partial identity and search key. Finally, the lookup procedure, described below, is followed in order to find the node that holds the details of the desired object. Then, the requester will contact such node and ask for the content.

Lookup Operation Details

The transformation from a partial description to a *Bloom Filter* and the lookup operation, as shown in Figure 4.5, begins when a client sends a lookup request to its discovery server (DS), which is also a node in the overlay network. Such request includes a partial description of the object the client is looking for. The DS uses the partial description to

4. Digging Deeper: Evolving Functional Blocks

get a lookup key follows the procedure described above. Then, the DS queries its own table of nodes to get the k -closest nodes to the key, selects the α -closest and stores them into a temporary list. Each node from this α -closest list is queried to obtain a new list with the k nodes they know are closest to the key. This gives α lists of k nodes. The DS builds a new list with the α closest nodes from those found in the previous lists and goes back to the previous step to get more nodes. The algorithm stops when it does not find new nodes to be included in the α -closest list.

To prevent extra queries, the lookup algorithm has a set with the ids of the already queried nodes, so it only queries the nodes which are not included in this set. It is also used to discard nodes in the stop condition of the algorithm.

Once the DS has obtained the α -closest list, it sends a *get_object_description* request with the lookup key to each node of the list. They return back the closest objects to the key from their storage tables. Finally, the DS returns a list with all these objects to the client. This list is sorted so that the first element is the closest object description to the lookup key, the second element is the second closest, and so on.

4.5 Ontology for Network Object Discovery

The increasing amount of data, information items, people, machines, and, in general, a large number of *network objects* connected to the Internet has exposed many challenges to the research towards the FI. In this environment, the search and discovery of those network objects is a crucial operation that is not covered by currently existing technologies.

Current mechanisms for the description, search, and discovery of network objects are based on their names, generally being domain names and typically being full-qualified domain names. This presupposes that a network entity that wants to interact with a network object, either retrieve the information it holds or just start a conversation with it, needs to previously know its name. When the name is not known, such entity needs to contact to a third entity that holds a list of available network objects and perform a search for some criteria. But this operation involves the end uses, humans, in the search and discovery operation.

Involving humans in the discovery operation denotes a clear lack in the functionality offered by current network mechanisms. Such behavior is, at best, undesirable to meet with the requirements of some scenarios (e.g. machine-to-machine communications) and invalid for many situations (e.g. ad-hoc networks). Moreover, the problem grows with the

introduction of new technologies, such as IoT and ICN, because they cause a huge increase of the number of network objects.

To overcome this problem we investigate the feasibility of using semantics in the search and discovery operations. Therefore, we design an ontology to efficiently organize the concepts used to describe and find network objects. In summary, we claim that using semantics in the discovery operations provides enormous benefits to the process, adding a high degree of expressiveness and facilitating network entities to perform their operation without the support of a human being. This affirmation is initially evidenced by the possibility to describe unlimited complexity in the queries, instead of a plain name, domain name, or a set of fields. This is provided by the linkage of concepts from different contexts.

As introduced in Chapter 2, the basement of current semantics and ontologies is the Resource Description Framework (RDF) [60]. It divides the description of an entity (here, a network object) in separate knowledge items. Each knowledge item is represented as a triple, which is composed of *subject*, *predicate*, and *object*. The subject is the entity that is described, the object is other entity, and the predicate is the relation between both entities. When using RDF to describe entities, a specific schema may be followed. To define the schema we have RDFS [96] and OWL 2 [99]. The former provides the basic schema elements (such as A is a B) and the latter provides complexer elements to build full ontologies. Moreover, to search for some object description residing in a RDF knowledge base we have RDQL [62] and SPARQL [63]. Both are simple query languages to express requested entities and constraints based on subjects, predicates, and objects. As RDF is not a file or code format, it needs specific methods to serialize it to a file. This is the case of Turtle [95], a quite simple and expressive language to represent RDF triples grouped by subjects.

In this section we use the mechanisms described above to design an ontology-based discovery protocol. We then evaluate it and demonstrate its behavior using a prototype implementation running on top of an experimentation infrastructure based on PlanetLab. We also demonstrate through practical experimentation how performs the proposed mechanism in comparison with DNS-SD [70], a global discovery mechanism based on the existing DNS infrastructure and methods. We therefore provide experimentation evidences that the ontology-based mechanism may be used for search and discovery, providing the enormous benefits without a great impact on performance.

4. Digging Deeper: Evolving Functional Blocks

4.5.1 Requirements and Design Principles

Before we design the ontology and protocol for the discovery we need to go around the specific requirements they should meet and the design principles they should follow. We discuss it below.

The specific requirements for the discovery mechanisms in the FI are coming from the necessity to overcome the problems found in the current Internet. They are easily seen by watching the typical operation. For instance, every network operation usually starts with a human using a search engine to find an object to communicate with. As introduced above, the object may be an information item, a person, a service, etc. This step is found in every interaction although many times the search is performed just the first time and the results are cached. However, this indicates a clear lack in the current Internet architecture because this functionality should be provided by the network in a transparent manner.

Derived from the problem stated above, we can state that the most important requirement that the discovery mechanism should meet is the possibility to state object queries in a natural way. This implies that semantics must be taken into account and thus that an ontology should be used in the discovery operation, so we are on track.

Moreover, the information retrieved from any discovery operation (query) must be consistent over time and with the information provided by the creator or publisher. Thus, the consistency is a principle that should be followed by our design.

One problem found in many network technologies is their impossibility to evolve with the changing requirements. However, for a semantically-driven mechanism, the extensibility and possibility of evolution is a requirement. Thus, both the network object description and the discovery protocol should be extensible to cover any future requirement while maintaining backward compatibility.

Additional design principles to follow can be found in [5]. From it we can highlight the simplicity principle, which states that the design should be kept as simple as possible, avoiding the complexity as much as possible. Also, we can highlight the connectionless packet switching principle. This principle states that the protocols should avoid establishing connections to perform their task. Related to the extensibility requirement described above, the polymorphism principle states that network protocols should manage and operate with first class objects in a transparent and abstract manner. Thus, the specific object contents are not treated by the intermediate layers and modules, just by the modules that should handle them. Furthermore, following the resource awareness principle gives additional benefits to the discovery mechanism and the network in general.

Finally, in order to support low-power devices and sensor networks, the discovery protocol should be as much efficient as possible and thus keep the message size to the minimum. This imposes the design to follow the scalability and amplification principle, stating that a proper mechanism for the FI should be prepared for the significant increase of network objects. It also reinforces the design to follow simplicity principle, because they are interconnected.

4.5.2 Ontology-based Discovery Protocol

In this subsection we discuss the design approach we followed to define the ontology used to semantically describe the network objects and the proposed discovery protocol, the Ontology-based Discovery Protocol (ODP), required to effectively use it with ODIN, the architecture introduced in Section 4.4.

As stated throughout this thesis, we define *network objects* as any abstract or concrete object connected to a network of any type. It includes information items, which are usually transferred through the Web, but also persons connected to a communication (intermediate) device and the devices themselves, such as machines, network equipment, etc. That said, a network object represents any entity that can be interoperated through a network. Moreover, the abstraction degree of the network object definition accomplishes the polymorphic design principle.

ODP follows this definition of network object and, therefore, its core ontology grows around the network object concept. Moreover, all types of network objects are treated in the same manner, so the same query mechanism and syntax is used to retrieve the information associated with any object. The differentiation is achieved by the use of the ontology in the description of the objects, as well as in the queries. As the designed ontology can be extended with any other, current and future, ontology, it correctly meets with the extensibility requirement.

Apart from the central network object, the designed ontology includes other concepts. To properly include those concepts we reviewed the concepts used in other ontologies. We discuss it below.

Review of Concepts

Before designing a proper ontology that covers the requirements discussed above we first need to analyze some other ontologies proposed in previous work to be familiar with the typical concepts and extract the lacks they have.

4. Digging Deeper: Evolving Functional Blocks

A basic ontology can be found in [72], it includes the following concepts: Profile, DeviceProfile, PrinterProfile, ScannerProfile, Printer1Profile, and Printer2Profile. These concepts can not be easily used to describe any type of network object, so the ontology we design should cover a wider range of network object types. However, our design should provide the ability to be mapped with other ontologies, including this one.

From the Machine-to-Machine (M2M) domain, the ontology found in [100] models a stack for management of M2M environments and includes the following concepts: Management, NetworkIF, Application Component, Presentation, Session, Transport, Internet, IP, and Link. Moreover, the management service ontology includes the following concepts: Service, Network, Element, Region, VNL, NW-Element, VNL-Element, Topology, and Trajectory. Although this ontology is richer than the previous one, it is too tied to the management view of the network and does not provide a clear form to represent plain network objects.

Finally, a more generic ontology can be found in [101]. It covers general network object description and discovery with specificities of energy efficiency. It includes the following concepts: Energy, ICT Resource, Networking, Computing, Storage, Bandwidth, CPU, Memory, Hard disk, Green, Dirty, Solar, Wind, Hydro, Gas, Oil, and Coal. As in the previous case, this ontology does not provide enough semantics to describe network objects and their relations, it is more tied to some specific aspects of the network objects. Anyway, it could be used to extend the concepts included in our ontology to add those specificities, so we do not need to include them.

Ontology Overview

Having extracted the concepts and lacks of the ontologies discussed above, we have a strong basement to determine the concepts of a new ontology that fits with the network object discovery requirements. It has to provide abstract entities, capabilities, and behavior, as well as the possibility to semantically link them. Moreover, it has to be simple but extensible, so we determined to reuse as much as possible the ontologies offered by other frameworks, such as Friend of a Friend (FOAF) [98], DOLCE UltraLite (DUL) [97], Dublin Core (DC) [102], and SPITFIRE [76].

That said, Figure 4.6 illustrates the designed ontology. It is centered around the network object concept as discussed throughout this thesis. Moreover, it differentiates capabilities and interfaces. A capability is an ability offered by a network object and an interface is a mechanism used by the network object to interact with it. Also, the ontology is extended by

4.5 Ontology for Network Object Discovery

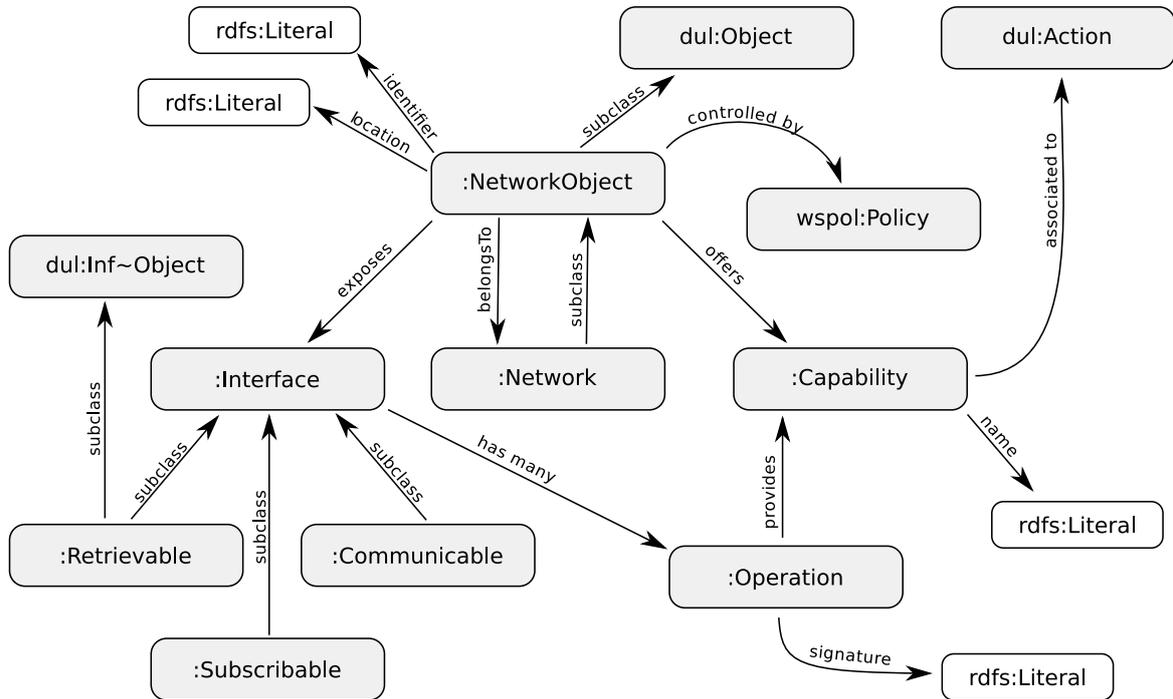


Figure 4.6: Overview of the proposed ontology.

WS-Policy to describe the access control of the network objects. In the ontology, a network itself is a network object and can offer own and aggregated capabilities and interfaces.

As shown in the ontology design, there are three different types of interfaces: retrievable, subscribable, and communicable. The retrievable interface is also a subclass of the Information Object defined by DUL and offers operations to retrieve the information (content) that the network object represents. The subscribable interface is included to cover the active objects which emit events. It offers the necessary operations to let event consumers, which may be also represented as network objects, to receive the updates of the events. Finally, the communicable interface represents a network object with operations to receive and emit messages and is capable to establish conversations.

Although the ontology proposed by SPITFIRE is not directly used, we included the proper equivalences of the network and relation (belongs to) concepts in order to easily support the extension of our ontology with it.

Once we designed the ontology, we represent it in RDF. We decided to use Turtle [95] as representation format. Also, we used RDFS [96] and OWL 2 [99] to represent the relations in a standard way. This schema is fed to an RDF engine, together with the RDF descriptions given by the network objects, and then used to resolve the queries sent by the requesters, which are specified in SPARQL [63]. To interact with the engine and let

4. Digging Deeper: Evolving Functional Blocks

network objects and other entities perform register and discovery operations we designed the protocol described below. Please refer to Appendix C to see the Turtle representation of the ontology, together with the serialization output of the example objects and queries.

Protocol Overview

Following the simplicity design principle, we decided to keep the search and discovery protocol as simple as possible. Thus, it is composed of only four messages: register request, register response, find object request, and find object response. The message format is designed with the minimum necessary fields to accomplish the objective of each message. The common fields found on every message (headers) have a message type, used to differentiate the enclosed message, and a message identifier, used to associate requests and responses.

The register request has a description field to include the description, in Turtle format, of the network object to register. The register response has a response field to return back the result of the register request, which can be *OK* or any error given by the RDF engine. The find object request has a query field to specify the query, in SPARQL, provided by the requester network object or entity. The find object response has a object field that returns back the result of the requested search. It is formatted in JSON and previously simplified to remove repeated values. These repeated values are found when there are many triples that fit with the same query condition.

All message exchanges are instantiated and serialized using the Apache Avro [103] mechanism and then compressed using the widely-known DEFLATE algorithm provided by the zlib library. In this tandem, Avro provides the required extensibility to the protocol while DEFLATE provides a reduced size to permit relatively big messages accomplish the minimum size requirement. The Avro mechanism itself supports the use of DEFLATE, so no extra effort should be dedicated to compress or uncompress the messages. Moreover, an Avro implementation is available for the most popular programming languages, so it is easy to adopt in most platforms. To carry the binary messages, the product of Avro+DEFLATE mechanism, between the client and server, we decided to use UDP as underlying protocol because it is quicker and lighter than TCP and follows the connectionless principle.

To meet the maximum size requirement while keeping the simplicity of the mechanism we differentiate two size limits (profiles): normal and lightweight. When the protocol operates with the normal profile, the maximum size of a message is determined by the maximum size of UDP, generally 65507 bytes (65535 - 8 byte UDP header - 20 byte IP header) when running on top of IPv4 and 65487 (65535 - 8 byte UDP header - 40 byte IP

header) when running on top of IPv6. In contrast, when the protocol operates with the lightweight profile, the maximum size of a message is fixed very low, under 512 bytes. This is a temporary limit because it will be reduced to fit with the maximum transmission units of IoT and sensor networks.

The descriptions of the network objects are meant to be immutable, so they can be spread without violating the consistency principle. Thus, a timestamp and a limited time-to-live is added to each network object description item. When a description should be updated, the owner just needs to send a new register message with the new information and the system will use the timestamp to replace the old information with the new one. The system could also push the updated information to the necessary caches and other elements that previously retrieved it, but this is out of the scope of the present work. The time-to-live is used to remove old information from the system and prevent to fill the system with outdated items. This implies that the description of a network object must be registered (refreshed) before the previous register expires. This is particularly important when using a distributed system, generally an overlay network, to build the discovery mechanism because the time-to-live varies depending on the churn rate of the network. A concrete study to determine the refreshing rate depending on the time-to-live and the churn rate is left for future work.

This protocol is designed to be used in a distributed manner, so no centralized system is in charge of the network object descriptions. However, to demonstrate the feasibility of the proposed ontology we concentrate the efforts in the implementation of the protocol against a centralized system and leave the inclusion of a distributed system for future work.

4.6 Identity-Based Control Plane

Current network technologies, mainly represented by the Internet, have demonstrated little capacity to evolve because of the strict binding of communications to identifiers and locators. While locator namespaces represent the position of communication participants in the graph of a specific protocol, unstructured/plain identifiers represent the position of communications participants in the global network graph. Although they are valid for forwarding packets along communication paths, both views fail to fully represent the actual entities behind communications beyond a simple vertex.

Another problem is that current mobility management schemes do not offer enough protection of security and privacy. When a device moves from one network to another (handover), its security and privacy contexts should be maintained regardless of the place,

4. Digging Deeper: Evolving Functional Blocks

location, or point of attachment to the network. Moreover, security and privacy contexts should be transparently preserved during the handover process, so when an entity moves from one network to another, some infrastructure element should be in charge to maintain its security and privacy, and this element should be trusted. Moving entities will change their properties dynamically, so their virtual identities change.

Apart from the context, when devices can move from network to network it is hard to unequivocally identify them or the entities they represent (persons, services, etc.). As we discussed in previous sections and stated in [104–106], we propose to use digital identities to represent the entities behind communications to identify them securely and privately. We still retain the definition of identity given by ITU-T X.1250 [13], to which an identity is a collection of attributes about an entity. We consider that entities are people, software (services), hardware (machines), things, etc. Thus, identities can act as communication endpoints, like when a person *talks* to a service.

We consider that an evolutionary migration approach has greater expectations of success than *clean-slate* approaches, so here we address the location/identifier separation in an integrated manner, with special attention to security and privacy. Therefore, in this section we take some of the functional blocks defined so far in this thesis, rework them, and join them together to define and evaluate the Identity-Based Control Plane (IBCP). With it we try to resolve the aforementioned problems by abstracting communications from identifiers and locators and by using identities to achieve enhanced security and mobility management operations. This control plane can be then integrated into different network architectures in order to incorporate the features it provides. This facilitates the evolution capacity of those architectures that separate the information transmission concerns (networking, routing), from end-to-end aspects like security and mobility management.

The need for preserving security and privacy contexts and their application regardless of the location, together with the secure identification of network entities, has led us to ensure that IBCP offers enhanced security and mobility management functions for other network architectures to permit their entities to communicate in an *identity-to-identity* manner. Mapping identity attributes among entities and applying them to different security contexts is a complex problem we aim to resolve by means of a network model based on an overlay network.

In summary, the approach we propose provides three main innovations not present in current security and mobility management solutions. First, the management of identity information makes extensive use of ontologies and semantic techniques to allow entities to be represented in a natural way. This facilitates retaining strong security and mobility

management schemes. Second, using the mechanism defined in ODIN to translate identities (attribute sets) into identifiers for referencing identities along the overlay network provides a feasible and efficient mechanism. Third, our approach integrates the functions in a control plane that will be attached to the network layer of other network architectures, so benefiting the separation of concerns and allowing those architectures to evolve, while keeping backwards compatibility.

To demonstrate our claims, we have analyzed the proposed approach from three different views. We have compared it to a well-known protocol with similar capabilities to our proposal, demonstrating the key differences. We have also performed a new security analysis, which complements the one performed in Section 4.3, to demonstrate that the mobility management protocol included in IBCP is secure. Finally, we have performed a detailed performance analysis by modeling our mobility management approach and weighing it against well-known identifier/locator separation protocols. Thus we have obtained strong evidence for its feasibility and for the assertions above.

4.6.1 Architecture Overview

In order to overcome the complexity of maintaining security and privacy contexts, while providing secure identification, we propose to build a control plane that addresses entities by their digital identities instead of their point of attachment to the network. This will complement any current and future network architecture, allowing them to incorporate its qualities by just using it to initiate and manage communications. This benefits the separation of concerns so that the network architecture can be concentrated, among other things, on network traffic routing and underlying mobility management, while the identity-based control plane will resolve the security issues, such as access control, privacy protection, authentication, authorization, etc. Hence, this proposal permits the creation of new architectures with security as a central aspect.

The core element of the IBCP is the DTE_i, as shown in Figure 4.7. Integrating the DTE_i provides a trusted overlay network in which each element is responsible for managing the identities of its own domain but offers restricted operations to other domains. This way, when an entity wants to contact with another entity from other domain, it will use the DTE_i node that controls its own domain, which the entity can directly reach. Thus, the *contact* operation is performed in a trusted, private, and totally secure manner. As this functionality is included in the IBCP, the entities do not need to contact their DTE_i nodes directly; their network protocols will do this for them.

4. Digging Deeper: Evolving Functional Blocks

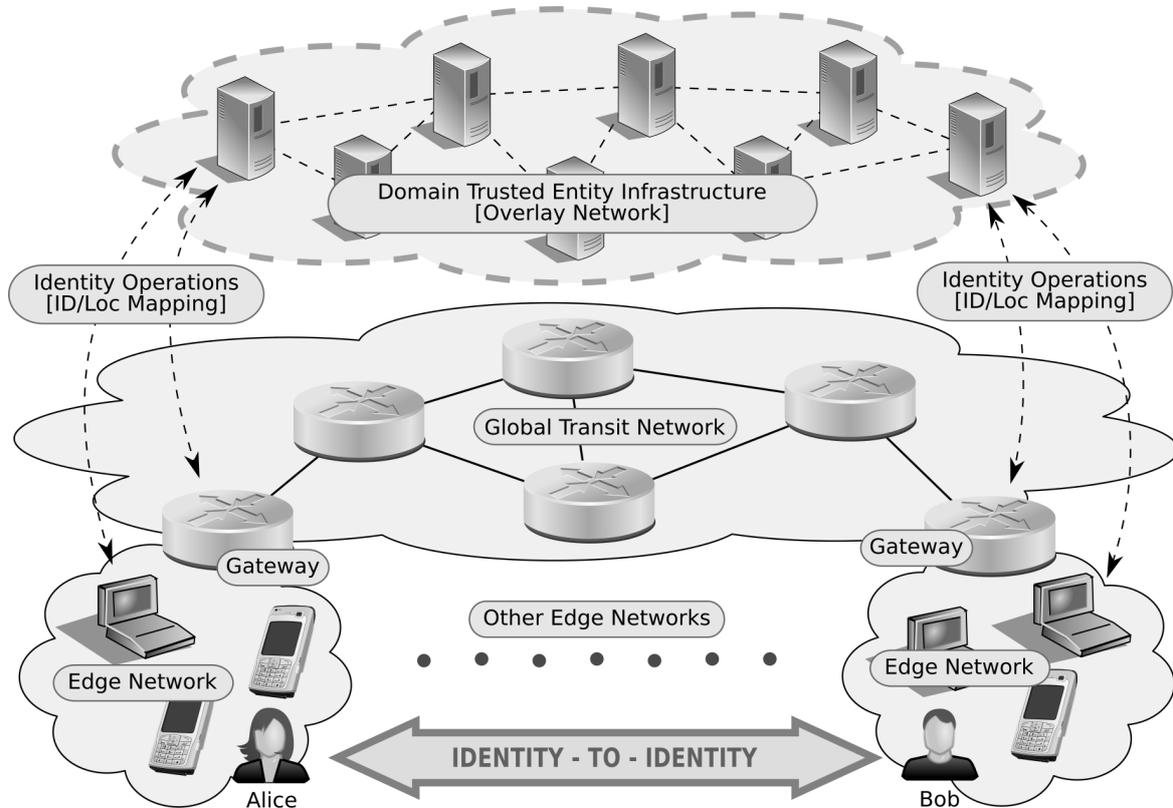


Figure 4.7: Overview of the Identity-Based Control Plane (IBCP).

Once the DTEi is integrated into the IBCP, it will use its inherent trusted and secure features to offer a set of network services that are part of the control plane of the network with which it is integrated. The main services are security negotiations and mobility management. Addressing security and privacy issues from the beginning of communications is a prospective requirement for all future network communications, so it should be offered from the same perspective as other communication operations. Furthermore, mobility has security implications beyond the inner security of the mobility mechanisms because the new network to which a node has moved may not enforce the security policies set by the communication parties.

Role of the DTEi in The Control Plane

As introduced above, the DTEi is responsible for managing the identities of network entities but it will also manage the security associations they establish for their communication, as well as their privacy contexts. These functions are achieved by interconnecting all DTEi nodes to build an overlay network that permits nodes to communicate each other

in a trusted and secure way without requiring external resolution mechanisms or even addresses.

To achieve proper indexing of identities in the overlay network we represent them as sets of attribute/value pairs. Then, we use the functionality provided by ODIN, the *Bloom Filters* [94], to get an indexing key for each identity, so similar identities will have close keys (using XOR metric). As attribute/value pairs cannot be derived from the keys, DTEi nodes are able to reference identities without revealing any attribute.

Each DTEi node or instance manages identities and dynamic identifiers for an administrative or network domain. While identities are used to identify networked entities, dynamic identifiers are just used to identify communication sessions. Identifiers can change over time without breaking communications because the DTEi will be used to inform the entities involved which identifier corresponds to which session. Moreover, the DTEi permits entities to validate those identifiers so they can be sure that they are *talking to who* they want without revealing their actual identities.

The DTEi will protect the privacy of networked entities but, if permitted by policies, it will be able to provide some attributes to other entities. For instance, when tied to address-based networks, the current location of an entity can be revealed to the underlying elements, so they can deliver network traffic to the entity.

Abstracting Endpoints from Identifiers and Locators

One key aspect of the identity-based control plane is that it emphasizes the differentiation of *identity* and *identifier*. It follows the ITU-T X.1250 definition of identity as “the representation of an entity in the form of one or more information elements which allow the entity(s) to be sufficiently distinguished within context”. On the other hand an identifier is a piece of fixed-size data that identify something.

Identities are therefore used to identify entities. They are the endpoints, so underlying identifiers and locators may change during communications. The identity-based control plane will be in charge of negotiating and reporting all changes required to manage communications.

Secure Communications

In general, entities participating in communications are authenticated with their digital identity. This is performed by the IBCP while mediating in session management. But entities are in full control of what they want to reveal and they can use virtual identities

4. Digging Deeper: Evolving Functional Blocks

to achieve anonymity. Moreover, using the elements of the DTEi, the IBCP can reveal identity attributes, but it is totally regulated by the policies set by the entities.

Apart from authentication, the IBCP can provide attribute-based authorization. It supports negotiating access to any resource, even to identity attributes, by means of the value of the attributes of another identity. For example, an identity can have a policy to *talk* only with other identities pertaining to the same domain. The control plane is used to negotiate this and, if it succeeds, permits the communication, reserves a session identifier, and reveals the locator in the underlying network.

With this mechanism, instead of hiding the identity information of an entity, the IBCP offers other entities a controlled access to such information. Thus, it is able to validate identities against specific entities, so other entities may ask it to ensure that an entity is “who” it is claiming to be. Also, we can consider that an entity is *authenticated* just by validating the identifier (or identifiers) it is using and the integrity of the messages exchanged with it, which is achieved by a signature field included in the messages. Therefore, when another architecture integrates the identity-based control plane, it will provide integrated authentication and authorization of communications.

In order to prevent educated guesses of the identity that is behind a host, the architecture permits arbitrary changes of session identifiers. Existing sessions are not affected by the identifier change because they are bound to session identifiers. New session identifiers are negotiated through the IBCP, which is a totally secure and trusted channel, so attackers can not follow the data flow to guess the identity associated to an identifier.

Finally, our approach proposes an asymmetric encryption mechanism to get confidentiality when needed. For instance, Identity Based Encryption (IBE) [107] provides strong security and fits perfectly with our proposal. Those mechanisms have obvious benefits over weaker encryption methods: 1) Transmitted information will be kept secret for longer; 2) They fit and perform much better in publisher/subscriber underlying networks. In the future, processor performance improvements may make those methods much more feasible.

Mobility Management

Mobility operations are sensible to environment changes, both from a functional and security points of view. Therefore, the IBCP includes a mobility management mechanism. It can be used to achieve comprehensive mobility support in other network architectures or just to complement them to ensure the security in their mobility schemes.

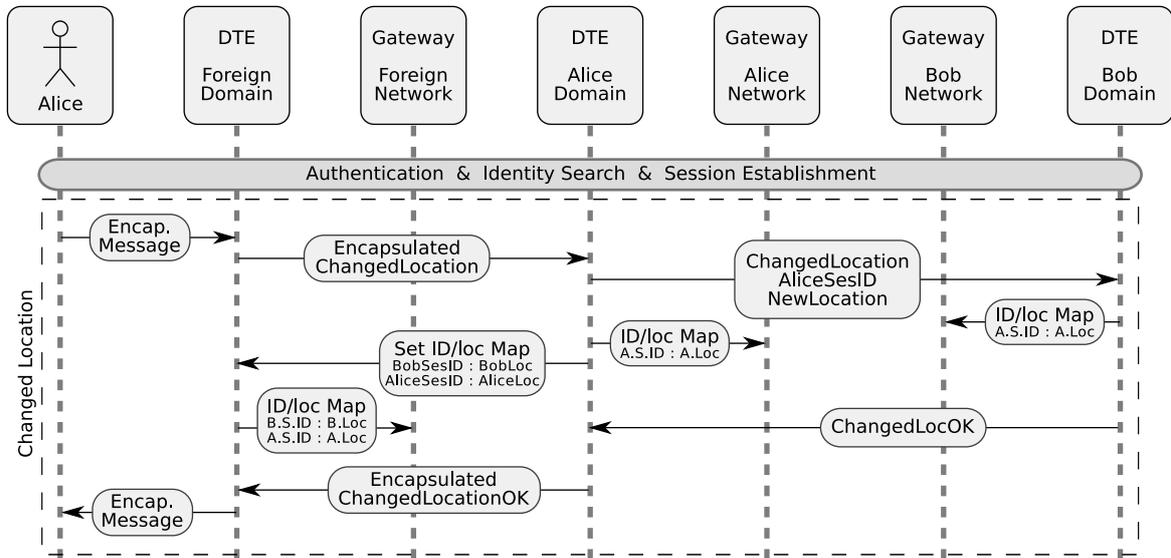


Figure 4.8: Mobility management: handover message exchanges.

As this control plane promotes the use of identifiers to deliver messages without using network addresses or any other location information, entities are able to keep the identifiers they are using even when they change from one underlying network to another. However, the underlying network infrastructures need to know how to deal with the messages exchanged by the entities, so they have to update the mapping between an identifier and the locator of the entity it represents.

When the control plane is integrated into an address-based underlying network, it introduces a *gateway* on each network domain, like in Proxy Mobile IPv6 (PMIPv6) [108] or Hierarchical Mobile IPv6 (HMIPv6) [109]. Behind the gateway there may be one or more entities, so their actual addresses are protected.

As shown in Figure 4.8, once an entity (Alice) has moved to a new domain (Foreign Domain), it uses the IBCP to report its new location for the current session identifier to the DTE_i node of its identity domain (Alice Domain) by sending an encapsulated message to the DTE_i node that controls the foreign domain. Then, the DTE_i node of Alice Domain will report the new location to the DTE_i node of the corresponding entity (Bob Domain) and both send a *ID/loc Map* message to the gateways involved in the communication. As the DTE_i node of Alice Domain does not have rights to set the mapping into the foreign gateway, it will send such message to the DTE_i node of the foreign domain and this node will send the mapping to the foreign gateway. Finally, a confirmation is sent back to the originating entity through the IBCP. As we demonstrate in the following sections, this

4. Digging Deeper: Evolving Functional Blocks

procedure does not add a big overhead to the network because it only requires to update the gateways of the entities with which the mobile node has opened sessions.

4.7 Security Analysis

In this section we analyze the security of the protocols defined within the main functional blocks of the architecture proposed in this thesis. They are IBNP and IBCP. With this analysis we demonstrate their qualities to provide protection of identifiers in IBNP, avoiding external entities to know details about them, as well as the security enforcement in the mobility approach of IBCP.

To strengthen our claims about the good security of the solutions, we decided to perform automated validation of the security aspects of the protocols by means of the AVISPA tool [110]. We have chosen this tool to perform both analyses because its simplicity and strength when analyzing network protocols.

4.7.1 Analysis of IBNP

As one of the main objectives of our architecture is to provide inherent authentication and enhanced privacy to other underlying network architectures, which are security related aspects, we need to be sure that the message exchanges of the proposed protocol are secure. Below we discuss the security requirements and how they are covered by the proposed protocol. The main security requirements to ensure communication privacy are defined as follows:

1. The session identifiers negotiated during the session establishment must be used to authenticate communication participants, which we call Alice and Bob, but represent any pair of abstract entities that can be formed by one or more real entities.
2. The service identifiers, which we call facets but also found as virtual identities in the literature, are used by the entities to start a session and must only be seen by the entities involved in a communication and the nodes of the DTE_i controlling their domains.

The proposed architecture and protocol meets with these requirements by the introduction of the indirection mechanisms through the DTE_i. On the one hand, as the entities are authenticated in their corresponding node of the DTE_i, the negotiation of session identifiers by a pair of nodes of the DTE_i can be used to authenticate the

1	A	->	DTE1	:	{ {AfID.AsID.BfID}_inv(KA) }_KDTE1
2	DTE1	->	DTE2	:	{ {AfID.AsID.BfID}_inv(KDTE1) }_KDTE2
3	DTE2	->	B	:	{ {AfID.AsID.BfID}_inv(KDTE2) }_KB
4	B	->	DTE2	:	{ {BsID}_inv(KB) }_KDTE2
5	DTE2	->	DTE1	:	{ {BsID}_inv(KDTE2) }_KDTE1
6	DTE1	->	A	:	{ {BsID}_inv(KDTE1) }_KA
7	A	->	B	:	BsID
8	B	->	A	:	AsID

Figure 4.9: Protocol represented in *Alice and Bob* notation.

entities associated to them. This operation is secure because the DTE_i has a channel with assured confidentiality by using message encryption and assured integrity by using signature mechanisms. On the other hand, using the DTE_i to query an entity and communicate the service identifiers to be used in a communication prevents any third party entity to know them and violate the privacy of the communication participants.

The extract shown in Figure 4.9 describes the protocol used to start a session between *A* (Alice) and *B* (Bob) through their corresponding nodes of the DTE_i (*DTE1* corresponds to *A* and *DTE2* corresponds to *B*). The parameters *AfID* and *BfID* are the service (facet) identifiers, the parameters *AsID* and *BsID* are the session identifiers, and *KA*, *KB*, *KDTE1*, and *KDTE2* are the public keys of the involved entities. The functions $\{d\}_-(k)$ and $\{d\}_{inv}(k)$ are respectively used to encrypt the message *d* with the public key *k* or the private key that corresponds to the public key *k*. The encryption is used to prevent an attacker to see the session or facet identifiers if it intercepts a message, so facet identifiers can not be linked to session identifiers. Finally, it shows a unencrypted message exchange, which is included to be sure that, after the session is established, an attacker can not link the session identifiers with the entities forming participating in the communication.

First, we formalize the protocol model shown in Figure 4.4 using Alice-Bob (A-B) notation that can be later used to perform its analysis and validation. We are interested to analyze the portion of the protocol started by Alice to communicate with Bob. The resulting notation is shown in Figure 4.9. From this notation we create a full description in High-Level Protocol Specification Language (HLPSL) that is used by the AVISPA tool. The HLPSL code can be seen in Appendix D. On it we define a different role for each entity taking part of the communication and fulfill each role with its specific responsibilities defined above in the notation. Then, we indicate that the analyzer tool should check that *AsID* and *BsID* can be used to authenticate Alice and Bob respectively, and that it should

4. Digging Deeper: Evolving Functional Blocks

```
1 A    -> DTE3 : {{TkADTE1.AsID.Aloc}_inv(KA)}_KDTE1
2 DTE3 -> DTE1 : {{TkDTE13.{{TkADTE1.AsID.Aloc}_inv(KA)}_KDTE1}_inv(KDTE3)}_KDTE1
3 DTE1 -> DTE2 : {{TkDTE12.AsID.Aloc}_inv(KDTE1)}_KDTE2
4 DTE2 -> GW2  : {{AsID.Aloc}_inv(KDTE2)}_KGW2
5 DTE1 -> GW1  : {{AsID.Aloc}_inv(KDTE1)}_KGW1
6 DTE1 -> DTE3 : {{H(TkDTE13).AsID.Aloc.BsID.Bloc}_inv(KDTE1)}_KDTE3
7 DTE3 -> GW3  : {{AsID.Aloc.BsID.Bloc}_inv(KDTE3)}_KGW3
8 DTE2 -> DTE1 : {{H(TkDTE12).OK}_inv(KDTE2)}_KDTE1
9 DTE1 -> DTE3 : {{H(TkDTE13).{{H(TkADTE1).OK}_inv(KDTE1)}_KA}_inv(KDTE1)}_KDTE3
10 DTE3 -> A    : {{H(TkADTE1).OK}_inv(KDTE1)}_KA
```

Figure 4.10: Protocol for mobility support in *Alice and Bob* notation.

check the secrecy of AfID and BfID to be sure that there is a secure channel between Alice and Bob through the DTE_i, so the session identifiers can not be publicly associated to their owners.

Using the HLPSL file obtained from the Alice-and-Bob notation we run the AVISPA tool using the On-the-Fly Model Checker (OFMC) as backend of the analysis. The tool output indicated us a *SAFE* result with *10 nodes* and *9 plies*. We also run the AVISPA tool using the Constraint Logic (CL-AtSe) backend and also got a *SAFE* output with *22 states* analyzed and *7 states* reached. With these results we can be sure that the protocol is secure. That said, we demonstrated that the proposed protocol covers the requirements raised above so our architecture can successfully prevent the traceability of network operations and thus enhance the privacy of communication parties.

4.7.2 Analysis of IBCP

Once demonstrated the security qualities of IBNP we proceed to analyze the security of the mobility scheme provided by IBCP. As mentioned above, we have also used AVISPA as we want to strengthen our claims about the security of the protocols provided by our architecture.

Since AVISPA requires the input file in the High-Level Protocol Specification Language (HLPSL), we first formalize our mobility protocol using Alice-Bob (A-B) notation. As shown in Figure 4.10, we first represent the entities taking part in the protocol. They are: Alice, represented as A; the nodes of the DTE_i corresponding to three domains (home domain as DTE1, correspondent domain as DTE2, and foreign domain as DTE3); and their gateways, represented as GW1/2/3. The function *H* represents a hash and the *inv* function gets a cryptographic private key from a public key. The variables KA, KDTE1, KDTE2, KDTE3, KGW1, KGW2, and KGW3 are the corresponding public keys of the entities. The variables named Tk* are the authentication tokens obtained during

the authentication process and $H(\text{Tk}^*)$ are hashed tokens used to authenticate answers. Finally, AsID.Aloc and BsID.Aloc represent the session identifiers and locators of Alice and Bob.

With this A-B notation we create the HLPSL file, as shown in Appendix D, assigning a different role to each entity and indicating that the analyzer tool should check the secrecy of all tokens (Tk^*), which can be known only by the pair of entities that communicate. We also indicate that the analyzer should use those tokens to authenticate the senders. To keep the simplicity of the process we do not test for replay attacks and we do not include the sequence numbers in the protocol notation, but urge the tool to run two parallel sessions to see if there is any problem with it.

The resulting HLPSL file is used as input for AVISPA to generate the Intermediate Format (IF) that is, in turn, used by the actual analyzers (backends). Apart from OFMC and CL-AtSe we have used in the previous evaluation, to strengthen the analysis we also used the SAT-based Model-Checker (SATMC) and the Tree Automata-based Protocol Analyser (TA4SP). All backends gave *SAFE* as result except the TA4SP, which gave an *INCONCLUSIVE* result due to the nature of the rules. These results demonstrate that the protocol is secure.

4.8 Performance Evaluation

In this section we evaluate the performance of the main functional blocks in order to demonstrate the claims as well as the feasibility of the proposed architecture. First we evaluate IBNP working on top of different underlying networks. Then we evaluate ODIN in terms of its accuracy and the proposed ontology in terms of the overhead it may introduce to simple naming schemes. Finally, we evaluate IBCP and compare its properties to HIP.

4.8.1 Evaluation of IBNP

In order to evaluate IBNP we have built two different proof-of-concept implementations, one made with CCN and the other made with the Extensible Messaging and Presence Protocol (XMPP), both working as lower layer networks. First we use CCN because it is a novel clean-slate architecture that defines both protocol and routing infrastructure, placing the content in the middle of communications. On the contrary, we also use XMPP to show how we can instantiate our architecture on top of an existing network protocol.

With the instantiations we executed a set of experiments to exercise the architecture and protocol working over the mentioned approaches. We then compared these results with

4. Digging Deeper: Evolving Functional Blocks

the results obtained from the execution of raw protocols, without our architecture, so we can get a notion of the performance penalties that can be introduced by our architecture. From the experiments we measure both the time spent in each message exchange and the total time spent in the whole test. Then, with the former measures we calculate the average time spent for each message exchange and with the latter measures we calculate the time spent by each message in terms of the whole application.

Instantiation Over CCN

The Content-Centric Networking (CCN) is a network architecture that defines a content-centric protocol, similar to a publisher/subscriber architecture, where identifiers are used to identify the content that is delivered through the network instead of the communication endpoints. Thus, it does not directly provide the possibility to perform end-to-end message exchanges, therefore, we need to build an adaptation layer to allow direct communications among network nodes.

In CCN, a subscriber declares its interest on some content, which is identified by a URI-like identifier, and waits until that content is available. Then, a publisher *updates* the content with that identifier, sending that content to the intermediate elements that deliver it to all interested subscribers. We built the adaptation layer exploiting this behavior, so each communication party declares its interest on a content identified by its own endpoint identifier. Then, when other party wants to send a message, it just updates the content identified by the destination identifier. This update makes the message to be received by the destination party. We take into account that two consecutive updates may hide one message, so when a message starts a conversation, another specific pair of identifiers are reserved (session identifiers).

After building the adaptation layer we define the message format, as well as the necessary message content types to perform the exchanges shown by Figure 4.4 (authentication request/response, validation request/response, and XRDS request/response). Data messages have no special content type, they are directly sent in the content of plain messages. In this evaluation, both message and content types are defined in XML, so the demonstration messages are quite simple to follow. Because of CCN identifiers are URI-like, we can use directly the XRI identifiers as proposed in our architecture, so it fits perfectly on top of CCN.

Once defined message and content formats we implement the logic of the DTEi responsible of receiving requests and sending back responses for authentication, XRDS, and validation. Then, we instantiate a different node of the DTEi for each domain with

its own configuration. Finally, we build the clients, Alice and Bob, that send and receive those messages defined in the scenario described above.

Instantiation Over XMPP

The Extensible Messaging and Presence Protocol (XMPP) is an application layer protocol widely used nowadays by many messaging infrastructures and we think it could be interesting to see how our architecture can be integrated with it. For instance, we can instantiate our architecture on top of XMPP, as we discuss below, but also we can modify XMPP to run on top of our architecture or, finally, get both architectures working side-by-side to achieve the identity-based authentication and privacy protection mechanisms.

Here we describe how to instantiate a solution that implements our architecture over XMPP. The key functionality offered by XMPP to cover our requirements is the dynamic user-name registration and de-registration, which is used by our architecture to reserve the dynamic session identifiers (user names in XMPP) because here XMPP is acting as lower layer network, and bind them to the identities.

After knowing that our architecture can be easily instantiated over XMPP we define the entities that implement the functionality to run the scenario shown in Figure 4.4. In this case, the role of DTEi nodes is played by XMPP server, so we define an XMPP server for each domain. Then, we build XMPP clients to play the role of Alice and Bob. Finally, we noticed that this layer, used to adapt our architecture to XMPP, is very thin because it fits perfectly with our architecture.

Identity/Underlying Interactions

The first step of the described scenario is the authentication of Alice and Bob in their corresponding nodes of the DTEi. Before this, DTE1 and DTE2 communicate their interest in content identified by “@domain1” and “@domain2” respectively. Then, since Bob acts as *server* it must be waiting for incoming data messages (requests) so must declare its interest in content identified by “@domain2=bob”. Before that, Bob sends an authentication message to the node of the DTEi controlling its domain, updating “@domain2” content with the message shown in Figure 4.12 that contains the XRDS document with Bob service endpoint identifiers as shown in Figure 4.11. Then, the node controlling its domain sends back a success response as shown in Figure 4.12 and Bob waits for messages.

After initialization, once Bob and the nodes of the DTEi controlling both domains are ready for responding the test, Alice starts and authenticates into the node controlling its

4. Digging Deeper: Evolving Functional Blocks

```
1 <?xml version="1.0" encoding="UTF-8"?>
2 <XRDS>
3   <XRD ref="xri://@domain2=bob">
4     <Query>@domain2=bob</Query>
5     <Status ceid="off" cid="verified" code="100"/>
6     <Expires>2018-05-05T00:15:00.000Z</Expires>
7     <ProviderID>xri://@domain2</ProviderID>
8     <Service>
9       <ProviderID>@domain2=bob</ProviderID>
10      <Type>service</Type>
11      <Redirect>@domain2=bob+service</Redirect>
12    </Service>
13  </XRD>
14 </XRDS>
```

Figure 4.11: Bob's XRDS document.

```
1 <message>
2   <source-id>@domain2=bob</source-id>
3   <destination-id>@domain2</destination-id>
4   <content-type>authentication-request</content-type>
5   <content>
6     <authentication-request>
7       <username>bob</username>
8       <password>bobpw</password>
9       <xrds> ... Bob's XRDS document ... </xrds>
10    </authentication-request>
11  </content>
12 </message>
13
14 <message>
15   <source-id>@domain2</source-id>
16   <destination-id>@domain2=bob</destination-id>
17   <content-type>authentication-response</content-type>
18   <content>
19     <authentication-response>
20       <status>OK</status>
21     </authentication-response>
22   </content>
23 </message>
```

Figure 4.12: Authentication request and response.

```

1 <message>
2   <source-id>@domain1=alice</source-id>
3   <destination-id>@domain2</destination-id>
4   <content-type>xrds-request</content-type>
5   <content>
6     <xrds-request>
7       <id>@domain2=bob</id>
8     </xrds-request>
9   </content>
10 </message>
11
12 <message>
13   <source-id>@domain2</source-id>
14   <destination-id>@domain1=alice</destination-id>
15   <content-type>xrds-response</content-type>
16   <content>
17     <xrds-response>
18       <id>@domain2=bob</id>
19       <xrds> ... Bob's XRDS document ... </xrds>
20     </xrds-response>
21   </content>
22 </message>

```

Figure 4.13: XRDS request and response.

domain as Bob did before. Then, Alice sends a request to DTE2 asking for Bob's XRDS document and DTE2 sends it in the response, as shown in Figure 4.13.

Using the XRDS document, Alice knows the endpoint identifier of the service offered by Bob (*@domain2=bob+service*). Then, Alice generates a new identifier to be used only in this communication with Bob (session identifier) applying SHA1 cryptographic function to a concatenation of its service endpoint and a random number, which in this case results in *6cf...244*. Using it as source and Bob's service endpoint identifier as destination, Alice sends its first message (request) to Bob, shown in Figure 4.14. Since Bob is associated with that identifier, it receives this message and validates the source identifier against the DTE1 with the messages shown in Figure 4.15. Then, Bob generates its own session identifier as Alice did before, resulting in *d81...df4*, and sends its response to Alice as shown in Figure 4.14. Alice receives the message and validates Bob's session identifier against DTE2.

Finally, once designed and validated the identifiers, the actors can normally continue with the remaining messages of their session. If, for whatever reason, an actor wants to use a new identifier, it only needs to use it in a message, the other party will validate it and continue again.

4. Digging Deeper: Evolving Functional Blocks

```
1 <message>
2   <source-id>
3     6cf5bc3ef38a8220d14c8bd2a21fdeef9af1d244
4   </source-id>
5   <destination-id>
6     @domain2=bob+service
7   </destination-id>
8   <content-type>data</content-type>
9   <content>ALICE'S REQUEST</content>
10 </message>
11
12 <message>
13   <source-id>
14     d8164dadcbf03c8b0ca25e57c3f9cd562eef6df4
15   </source-id>
16   <destination-id>
17     6cf5bc3ef38a8220d14c8bd2a21fdeef9af1d244
18   </destination-id>
19   <content-type>data</content-type>
20   <content>BOB'S RESPONSE</content>
21 </message>
```

Figure 4.14: Alice's first data message exchange with Bob.

```
1 <message>
2   <source-id>@domain2=bob</source-id>
3   <destination-id>@domain1</destination-id>
4   <content-type>validation-request</content-type>
5   <content>
6     <validation-request>
7       <service-id>@domain1=alice+service</service-id>
8       <session-id>
9         6cf5bc3ef38a8220d14c8bd2a21fdeef9af1d244
10      </session-id>
11     </validation-request>
12   </content>
13 </message>
14
15 <message>
16   <source-id>@domain1</source-id>
17   <destination-id>@domain2=bob</destination-id>
18   <content-type>validation-response</content-type>
19   <content>
20     <validation-response>
21       <status>OK</status>
22     </validation-response>
23   </content>
24 </message>
```

Figure 4.15: Validation request and response.

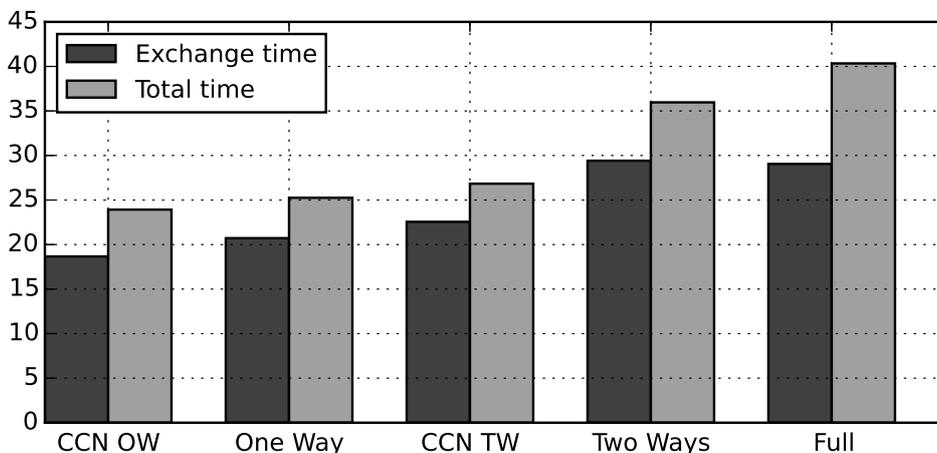


Figure 4.16: Performance comparison (milliseconds per message).

Performance Comparison

To get an approach of the behavior of IBNP we have executed a set of experiments under the scenario described above. With the measurements taken from the experiment we compared the average time spent in message exchanges. Figure 4.16 shows the results for: sending messages only from an emitter to a receiver using raw CCN (“CCN OW”) and our implementation (“One Way”); sending requests and receiving responses using raw CCN (“CCN TW”) and our implementation (“Two Ways”); and exchanging messages using all elements of our architecture (“Full”). For each case, it shows the average time spent on each message exchange (“exchange time”) and the overall time taken by the whole execution divided by the number of exchanges (“total time”), which includes the extra processing.

From these results we extract that the implementation of our architecture takes only a few milliseconds more than raw CCN, due to the extra time needed to process XML formatted messages. The worst case is in the exchange time of the request/response exchange because of two extra steps (XML parsing and encoding), but it only takes over 6-7 milliseconds more than raw CCN. Finally, the extra *total time* observed in the *full* test is due to the authentication, XRDS exchanges, and identifier validation. These operations are negligible for communications with several exchanges but must be minimized for those with few exchanges.

When comparing CCN with XMPP, Figure 4.17 and Figure 4.18 show the results of the evaluation performed with CCN and XMPP respectively, as well as with our architecture

4. Digging Deeper: Evolving Functional Blocks

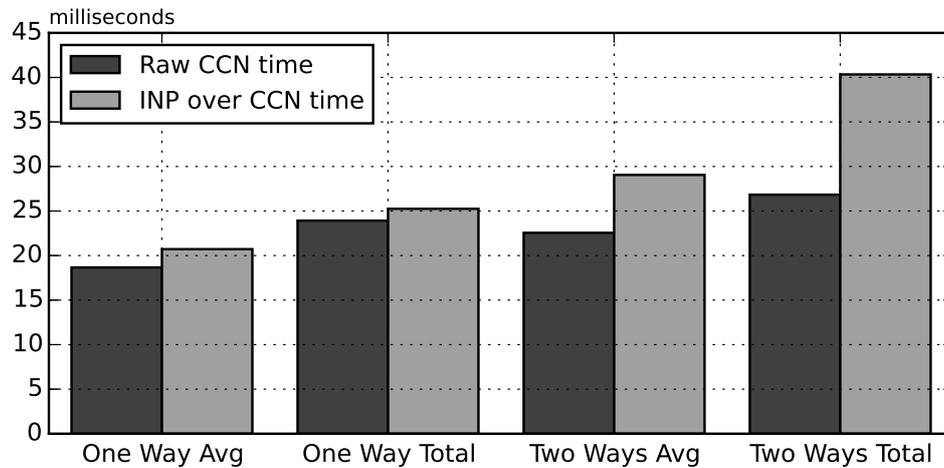


Figure 4.17: Performance comparison of our protocol over CCN with raw CCN.

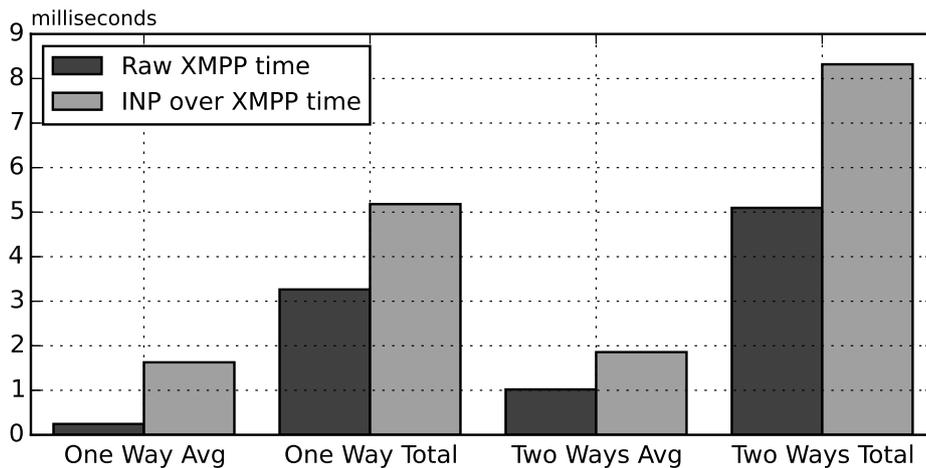


Figure 4.18: Performance comparison of our protocol over XMPP with raw XMPP.

instantiated on top of them. On the plots we can watch the average time spent on each message exchange displayed as “One Way Avg” and “Two Ways Avg”. It also shows the total time taken by the whole execution divided by the number of exchanges, including the extra processing, displayed as “One Way Total” and “Two Ways Total”. One-way results are obtained measuring the time spent in sending messages only from an emitter to a receiver, while two-way results are obtained measuring the time spent in sending requests and receiving responses. The two-way test includes the messages exchanged with the other elements of our architecture.

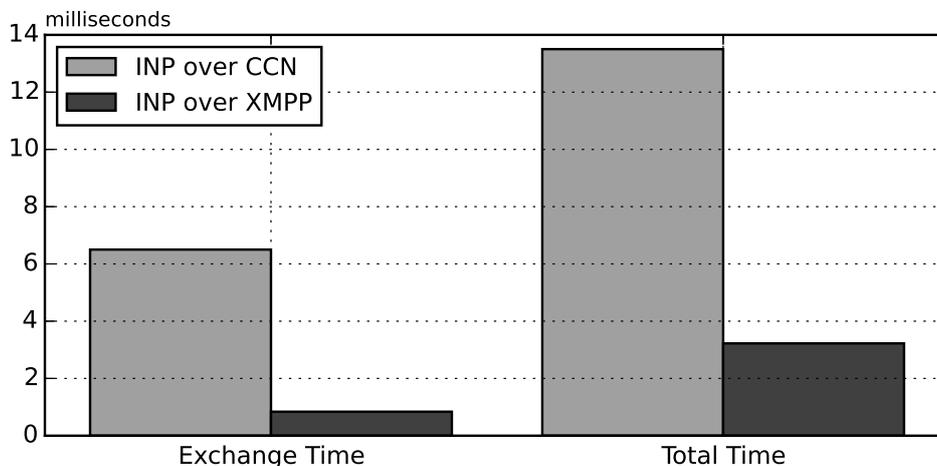


Figure 4.19: Comparison of the overhead of our protocol on CCN and XMPP.

Then, Figure 4.19 shows the comparison of the overhead of our protocol (aka INP) when instantiated over CCN and XMPP for the two-way tests. The overhead is the principal remark of all tests with respect to our architecture. It lets us see the time increased by using our identity based architecture and protocol on top of the other lower layer network architectures. Furthermore, overhead is stabilized below 10 ms in the case of INP over CCN and below 1 ms in the case of INP over XMPP.

Observing the results described above and especially the overhead comparison, we extract that our architecture takes only a few milliseconds (ms) more than raw CCN or XMPP, which is due to the necessary extra time to process JSON formatted messages and certain specific operations of the adaptation layer over each lower layer protocol. The worst case is in the exchange time of the request/response messages because of two extra steps (JSON parsing and encoding) but, as shown in the overhead comparison and reinforced by the overhead evolution, it only takes around 6.5 ms more than raw CCN and less than 1 ms more than raw XMPP. Finally, the extra *total time* observed in the tests is due to the authentication, XRDS exchanges, and identifier validation. Although this extra time is negligible for communications with several exchanges, it must be minimized for those with few exchanges.

4.8.2 Evaluation of ODIN

To evaluate ODIN we built a prototype implementation of the proposed approach together with a discovery server, a client for object registration, and a client for object

4. Digging Deeper: Evolving Functional Blocks

lookup. Then, we defined an experiment and executed it into the GAIA experimentation infrastructure [111], briefly described in Appendix B. It is built by 47 computation nodes with 1 GiB of RAM and 2 Ghz of CPU, interconnected by two switching levels. The first level connects 20 nodes through 100 Mbit Ethernet links while the second level connects the switch of the first level with the remaining computation nodes, all of them through 100 Mbit Ethernet links.

During the experiment we deployed as many ODIN nodes as computing nodes available in the experimentation infrastructure. Then, we deployed one more ODIN node for each registered network object. These nodes run in different threads, so they do not interfere each other in their operation. We executed the experiment for different number of registered objects (127, 250, 500, and 750). This results into different network sizes (5375, 11750, 23500, and 35250), which is one of the most important aspects to analyze in an overlay network.

The lookup clients will send 10 different queries to each discovery server for each number of attributes in the partial description (from 1 to 9). Each operation is repeated 5 times, resulting an average of 15000 lookups for each experiment configuration.

To control the experiment we built a custom framework that permits us to define the application instances that form the experiment and indicate the outputs that should be obtained. The experiment controller resides in a separate node inside the network of the experimentation infrastructure. The experimentation workflow is as follows: 1) The controller contacts all nodes to know which of them are available; 2) The work directories from the selected nodes are prepared; 3) The instance packages are uploaded to the nodes; 4) Packages are extracted to the place they will be run; 5) All instances are launched (some in background, like servers, and others in foreground, like clients); 6) Wait for finish and terminate (kill) all instances; 7) Download the results indicated in the experiment description.

From each configuration of the experiment we obtained different measurements. First, we measured the time spent in registering objects for each operation. Second, we measured the time spent in the lookup operations. Third, we measured the size of the routing table of each node, separated in main nodes and object nodes. Fourth and final, we measured the precision of the lookup operations, differentiating the number of discoveries returning the desired object as as first result, the number of discoveries returning the desired object as non first result, and the number of discoveries that do not return the desired object. The results of these measurements are discussed in the following section.

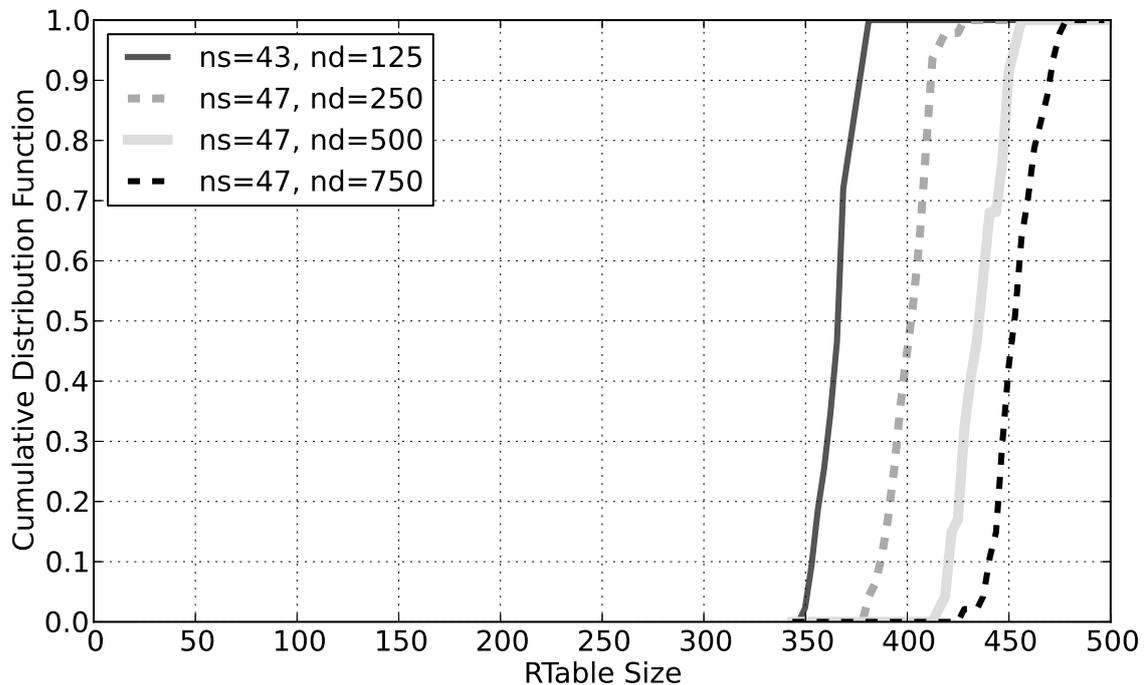


Figure 4.20: CDF of routing table size of main nodes.

Results

Here we discuss the results of the experiments we run to demonstrate the qualities offered by ODIN, as discussed in Section 4.4. The main outcomes of this evaluation are extracted by studying the measurements we obtained from the experiments.

Regarding the routing table size, as shown in Figure 4.20, the routing table sizes of the main nodes increase with the number of nodes in the overlay network but the increase of the number of main nodes is negligible. This is because, as they act as access point for the secondary nodes (object nodes) they have the opportunity to know more objects. However, as the maximum number of nodes in the table is 3200, resulting from the 160 buckets and 20 nodes per bucket, the average size of the tables is very low, not overpassing the 450 nodes per table on average.

The routing table size of the nodes corresponding to the objects registered, as shown in Figure 4.21, is very similar, regardless of the number of nodes registered. This is because those nodes are bootstrapped to the main nodes, their access points, and they only know other nodes when they interact with them. On average, this size is kept around 20, which is the selected bucket size in the experiments. This is because, during the bootstrap, the nodes select k ($= 20$) other nodes to be included on their tables. From this result we can

4. Digging Deeper: Evolving Functional Blocks

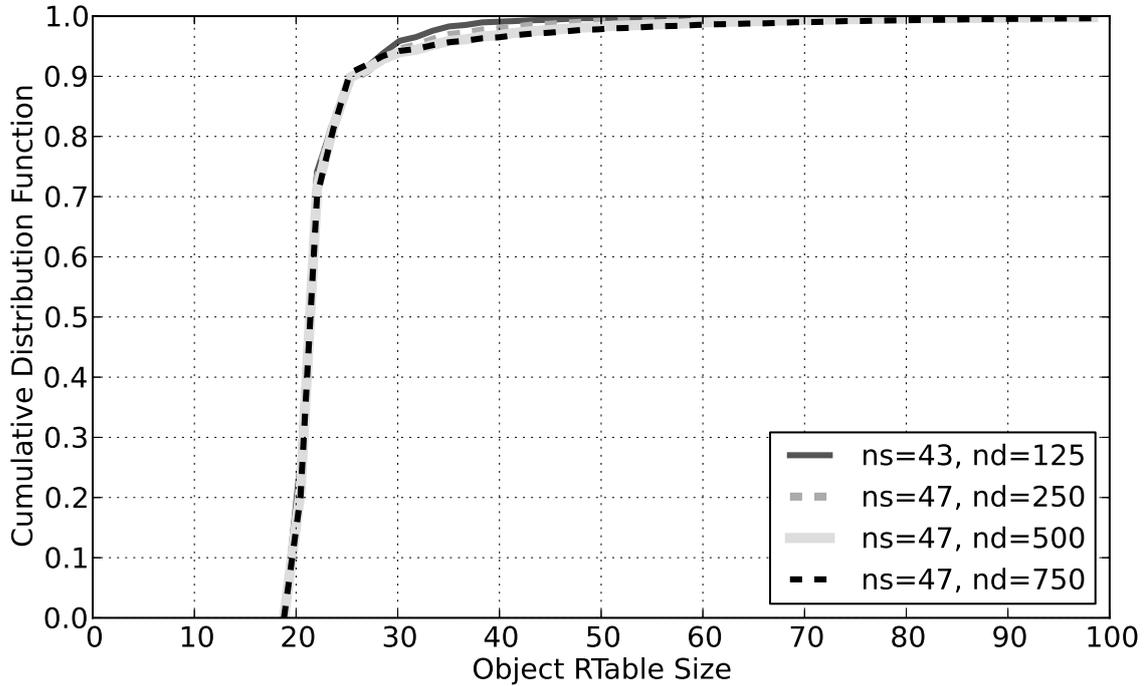


Figure 4.21: CDF of routing table size of nodes from registered objects.

again confirm that it is very far from the maximum number of nodes (3200), so the overlay network can still grow without impacting performance.

Regarding the time spent in registering network objects, as shown in Figure 4.22, there is a slight difference when the overlay network size changes. This difference is because of the lookup operation that each node performs during the bootstrap into the overlay network. Although the median grows from around 100 ms to around 125 ms, the average time grows from around 100 ms to around 170 ms. This difference has been mainly affected by those operations with extreme times, which had errors (packet losses). However, both are reasonable times considering all operations that should be performed to bootstrap a new node and register the network object description in the α -closest nodes of the DHT. Indeed, this demonstrates that the registration time depends on the size of the network but the time is very low and demonstrates the validity of the proposed solution.

Regarding the time spent in the lookup operation, as shown in Figure 4.23, the size of the overlay network has more impact to the lookup than the register operation. This operation is very quick for small networks, with an average of 63 ms and a Q3 of 66 ms for the smallest network, but is also kept in small times when the network grows, having an average of 207 ms and a Q3 of 209 ms when the overlay network is almost 7 times bigger.

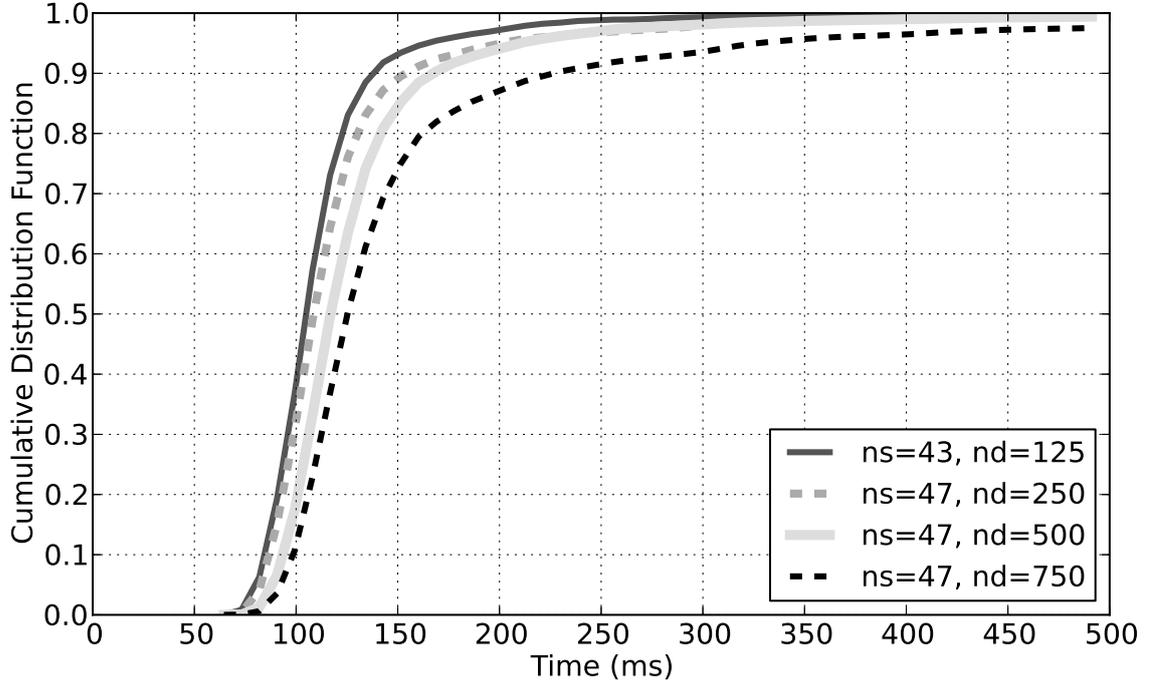


Figure 4.22: CDF of time spent in object registration.

As it is expected, the overlay network size affects to this time because DHT operations have to contact many nodes, having a maximum complexity order of $O(\log n)$ nodes, where n is the size of the network. This is correlated with the measurements obtained because the network size is grown 7 times but the lookup time grows slightly more than 3 times. This result is another indicator of the feasibility and scalability of the proposed solution.

We have also extracted the lookup precision of the operation. This means that, for each number of predicates used in the lookup operation, we have measured proportion of lookups did not result the requested object, the proportion of lookups that resulted the requested object as non first result, and the proportion of lookups that resulted the object as first result.

In Figure 4.24 we show the results regarding the precision of the discovery approach proposed in ODIN. It depicts that when using only 1 or 2 attributes in the partial description used in the lookup, the system does not retrieve the desired object. However, starting from just 3 attributes, 30% of the discovery operations return the desired object, of which less than 10% have it as first result. As expected, the precision grows exponentially with the number of predicates (or attributes) included in the partial description. From 4

4. Digging Deeper: Evolving Functional Blocks

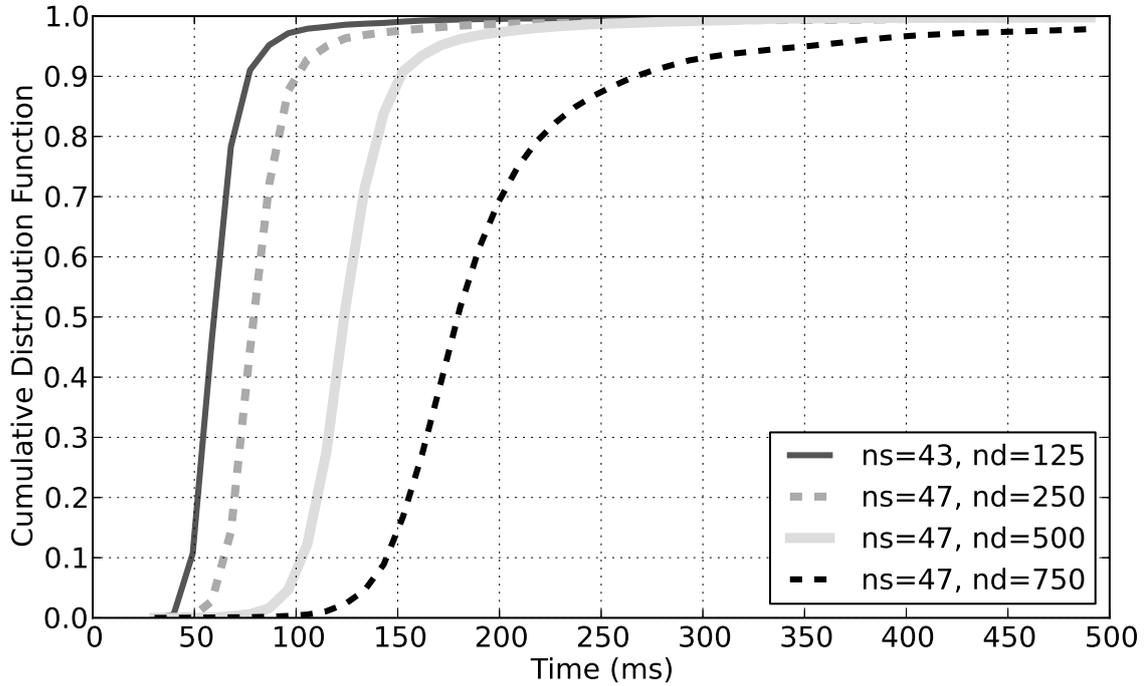


Figure 4.23: CDF of time spent in object lookups.

attributes onwards, almost the 100% of the discoveries return the desired object description, most of them having it as first result.

When increasing the number of nodes forming the overlay network, as shown in Figure 4.25, the precision of the lookup method is almost the same, with the exception found in the reduced proportion of objects found using partial descriptions with 3 and 4 predicates. The same behavior can be observed in following increments of the number of nodes forming the overlay network, as shown in Figure 4.26 and Figure 4.27.

From the precision results we can obviously determine that using a very low number of attributes/predicates is not adequate to obtain the requested network object description from ODIN. However, from 4 predicates onwards, the results are impressive because more than 90% of discoveries found the searched object for all the different network sizes used in the experiments. Although some objects are not found as first result, a client finds what it is looking for, so they are considered as positive results. This result, together with the performance results discussed above, demonstrates the adequacy, and hence the feasibility, of the proposed solution with ODIN to be used as discovery mechanism for the FI.

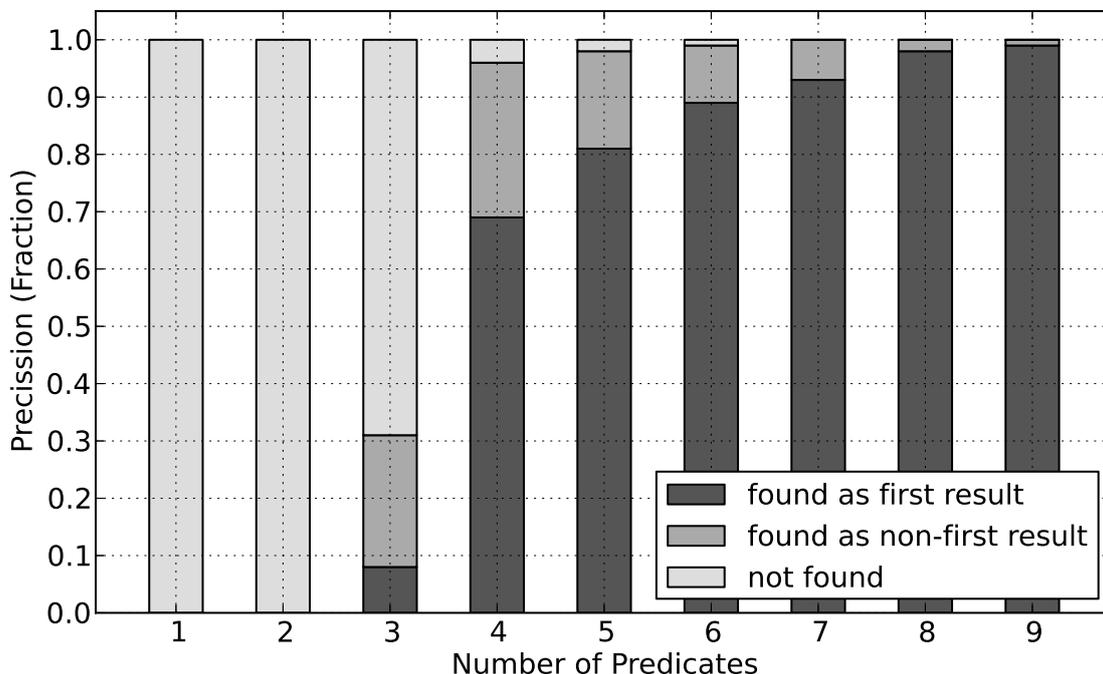


Figure 4.24: Lookup precision of the overlay network with 5375 nodes.

4.8.3 Evaluation of the Ontology

To know whether the proposed ontology and protocol, as described in Section 4.5 benefits to network operations or not we have evaluated them. Thus, here we discuss the procedure and methodology we used to evaluate the proposal. In [112] and [113] we find different proposals to evaluate ontologies. From them, we selected the application-based evaluation method because it shows the behavior of the ontology while running on its target environment.

The application-based evaluation method states that an ontology should help the applications to do their job and proposes to compare the behavior and performance of an application when it is using the proposed ontology and when it is not using it. Therefore, to apply this method to evaluate the proposed approach, we need two different applications to be compared, one with the ontology and other without it. We decided to use an application using the well-known DNS-SD protocol [70] and compare its behavior with a prototype implementation of the proposed protocol and ontology.

This prototype implementation of the discovery server is based on two different modules. The first module is based on RedStore [114] which provides a lightweight RDF storage and query engine and is accessed through a simple REST interface. The second module

4. Digging Deeper: Evolving Functional Blocks

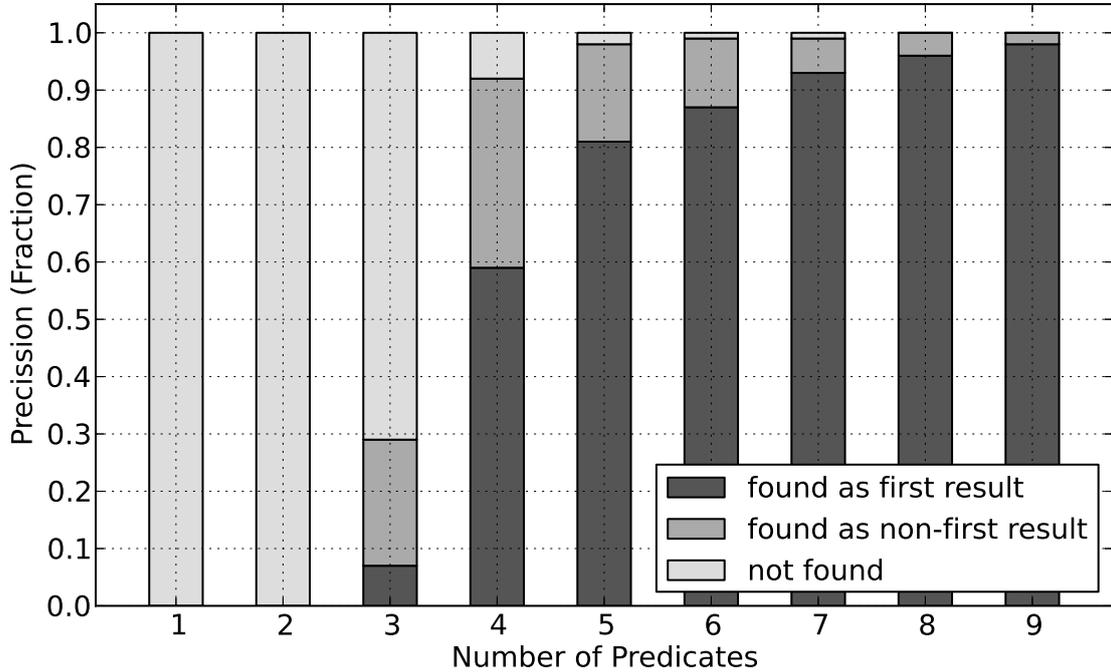


Figure 4.25: Lookup precision of the overlay network with 11750 nodes.

provides the implementation of the protocol, receiving and processing the Avro messages and interacting with the RedStore. Both modules are instantiated in the same node.

The implementation of the discovery clients is straightforward, just using the protocol and the Avro format to interact with the discovery server to register the network object descriptions or query them.

Once we finished the prototype implementation we decided to perform the evaluation in an experimentation environment, so we can get valid results of the behavior of the proposal that can be transferred to the real world. Moreover, this evaluation approach facilitates the reproducibility of the experiments. Thus, we prepared the experimentation process as discussed below.

Experimentation Infrastructure

Before running the experiments we need to select and prepare an experimentation environment, specifically the experimentation infrastructure to be used to run the proposed experiments. We decided to use a private PlanetLab [115] infrastructure we have deployed in our research laboratory. It is built by the same software used in both the global PlanetLab and the PlanetLab Europe branch, so we can use the same experimentation tools

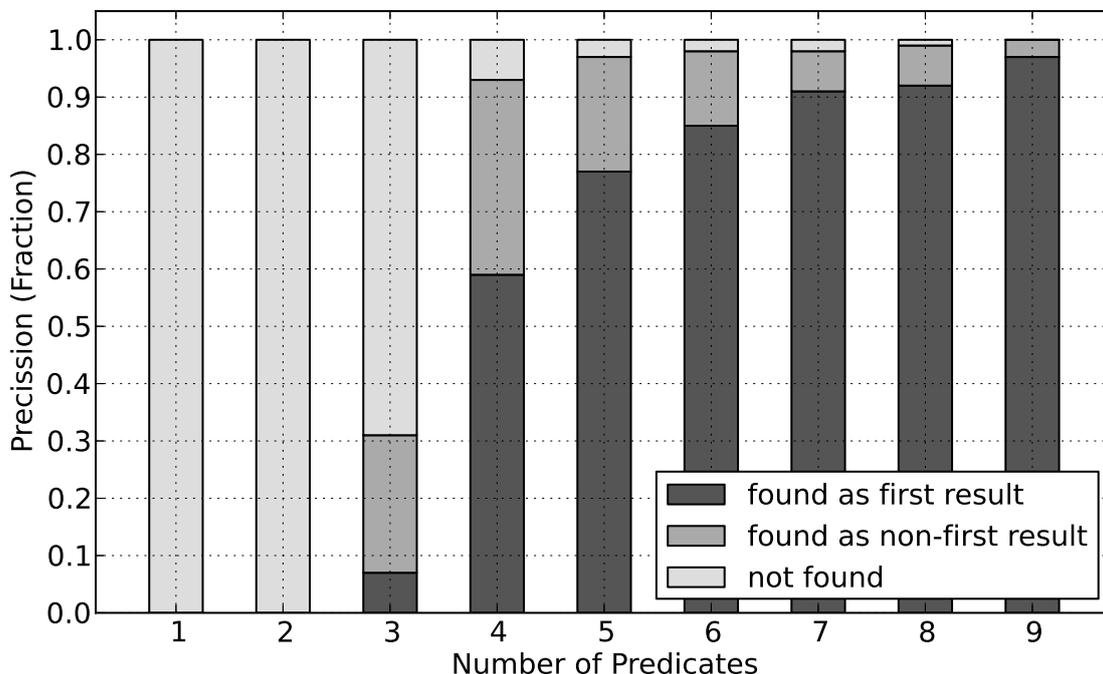


Figure 4.26: Lookup precision of the overlay network with 23500 nodes.

used in those environments and our experiments can be performed in those experimentation infrastructures without requiring any modification.

The PlanetLab deployment on our private experimentation infrastructure is composed of six nodes with 1 GiB of RAM and 2.33 GHz of CPU. The nodes are connected to the same local network (VLAN) by 100 Mbps ethernet links and a Cisco Catalyst 2650 switch. These resources are managed by the MyPLC (My PlanetLab Central) software, openly distributed by PlanetLab. Within this deployment we created a new slice, composed by the six nodes, and assigned it to the experiments discussed here.

Experimentation Control Framework

To facilitate the experiment deployment, execution, result gathering, and final cleaning, we decided to use an experimentation control framework to take control of all the steps in the course of the experiments. We selected the Network Experimentation Programming Interface (NEPI) [116] because it supports, among others, the PlanetLab experimentation environment we have deployed. Moreover, it provides a consistent framework with support for the whole experimentation process.

4. Digging Deeper: Evolving Functional Blocks

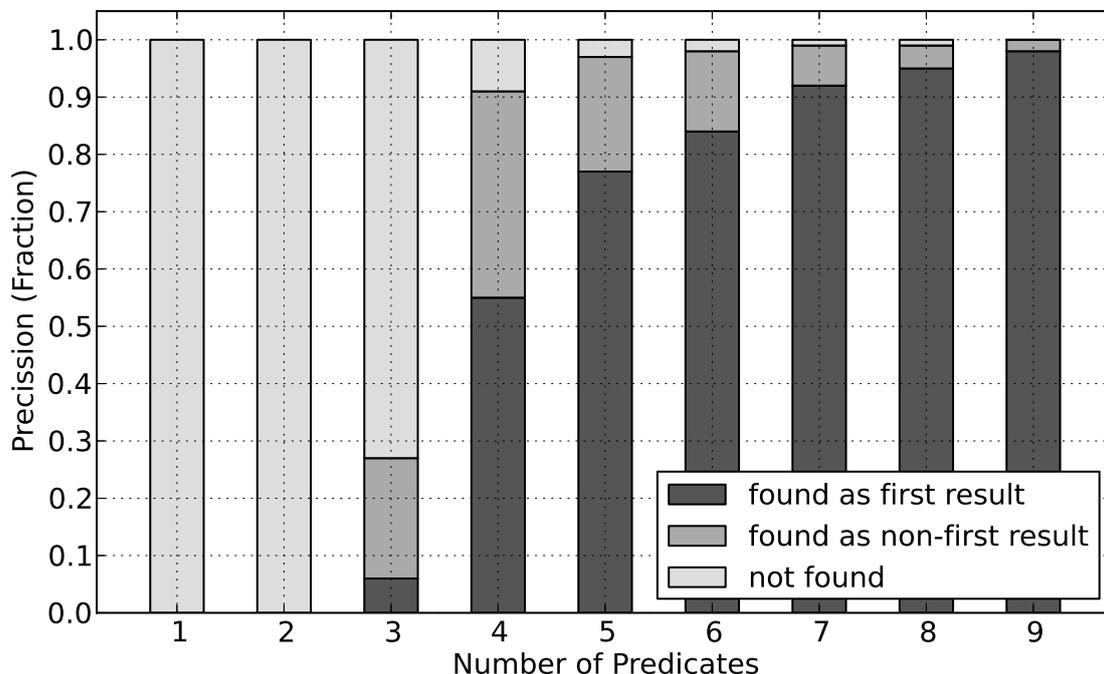


Figure 4.27: Lookup precision of the overlay network with 35250 nodes.

In NEPI, each functional entity involved in an experiment is represented as a node in a graph. Thus, we have nodes that represent the actual network nodes, nodes that represent the network interfaces, nodes that represent the applications deployed on each real node, nodes that represent the real network (Internet), etc. Then, the nodes are instantiated and interconnected (wired) to define the experimentation topology (graph). The NEPI framework gets this graph and it connects to the experimentation infrastructure to select the network nodes to which deploy the applications, set-up the wiring, etc. Finally, the experiments are executed and, when they are finished, NEPI gets back the outputs and measurements.

To build a topology, NEPI provides two different ways. On the one hand, NEPI is accompanied with a GUI to visually build the graph of the experiment by drag-and-drop the graph nodes and create the definitions of the applications, nodes, interfaces, etc. involved in the experiment. On the other hand, it also provides a Python library that offers the node types as classes, so they can be instantiated and interconnected to build the experiment graph topology. We decided to use the Python library because it facilitates the control of the repeating actions.

Experimentation Methodology

The main objective of the experiment execution is to demonstrate that the proposed protocol and ontology can be used as discovery mechanism in replacement of currently existing protocols. To achieve this objective we built a prototype implementation of the proposed mechanism to compare its performance and behavior with a solution based on DNS-SD. We described above the experimentation infrastructure and framework we will use in the experiment and here we discuss the necessary configurations and measurements to demonstrate those claims.

A typical scenario to evaluate the behavior of a discovery process involves some network objects, the resolution (description, query) system, and a client searching for the network objects. Thus, in our experiment we define four different network objects, a discovery server, and a requester client. Each of them is instantiated in a different network node. Moreover, the same scenario is used for both DNS-SD and our prototype, so the results are consistently comparable.

Each experiment execution performs in the following way. First, all the nodes are instantiated with the corresponding software. Second, the network objects register themselves into the discovery server by providing a RDF description that follows the proposed ontology. Third, the discovery server includes the provided descriptions into its RDF database. Fourth, the requester (client) sends a query to the discovery server. Fifth and final, the discovery server resolves the query and responds with the results.

To exercise the capabilities of each technology we created the following network objects:

- A simplex color printer.
- An information object (content), pointing to a website and described by a set of tags.
- A duplex, black-and-white printer.
- A color scanner.

Each client queries the discovery server as many times as necessary to resolve first the information object by providing some tags as the constraints and then the printer with color support by providing this capability as constraint.

From the experiment execution we will obtain the following measurements:

- The number of exchanged packets/messages needed to resolve each network object. This permits to know the operational cost of the solution in number of packets.

4. Digging Deeper: Evolving Functional Blocks

- The size (in bytes) of each message exchange. This permits to know the minimum and maximum packet sizes of the studied protocols.
- The size of all exchanges. This permits us to know the operational cost in bytes.
- The time spent in each message exchange. This permits to extract the individual message cost of each protocol.
- The time spent to obtain the whole network object description. This permits to know the operational cost in time of each protocol.

Below we discuss how to use the prototype implementation together with the control framework to execute the experiments in our experimentation infrastructure and thus obtain these measurements.

Implementation and Execution of the Experiments

To obtain the measurements discussed above we need two different mechanisms. First, the message counting and packet/message size can be obtained by capturing (sniffing) the network traffic flowing among the nodes involved in each operation. Second, the time measurements should be taken from the applications, so we need to instruct them to emit time traces to their standard output.

Once the applications are prepared and before building the experiment definition with the NEPI framework, we need to generate the corresponding deployment packages to both solutions. These packages are used by NEPI to deploy and, when necessary, build the applications on the experimentation nodes. Therefore, for each solution (DNS-SD and ODP), we generate a package containing the discovery server, another package containing the client, and a different package for each network object. Each package is configured with the necessary parameters to execute the planned experiment. For example, the client is configured to wait a few seconds to permit the network objects to be registered before it queries the discovery server.

Now we build the experiment definition in Python using the NEPI library. We first need to instantiate an object that represents the testbed, in this case PlanetLab, and which is then used to create the NEPI experiment definition nodes. Using it, as shown in Figure 4.28, we create the *Internet* NEPI node, which represents the real network connection among the real experimentation nodes of the infrastructure. Both dashed and solid lines represent the interconnection between a pair of NEPI objects. The small boxes are the nodes of the NEPI experiment graph while the big boxes represent the actual

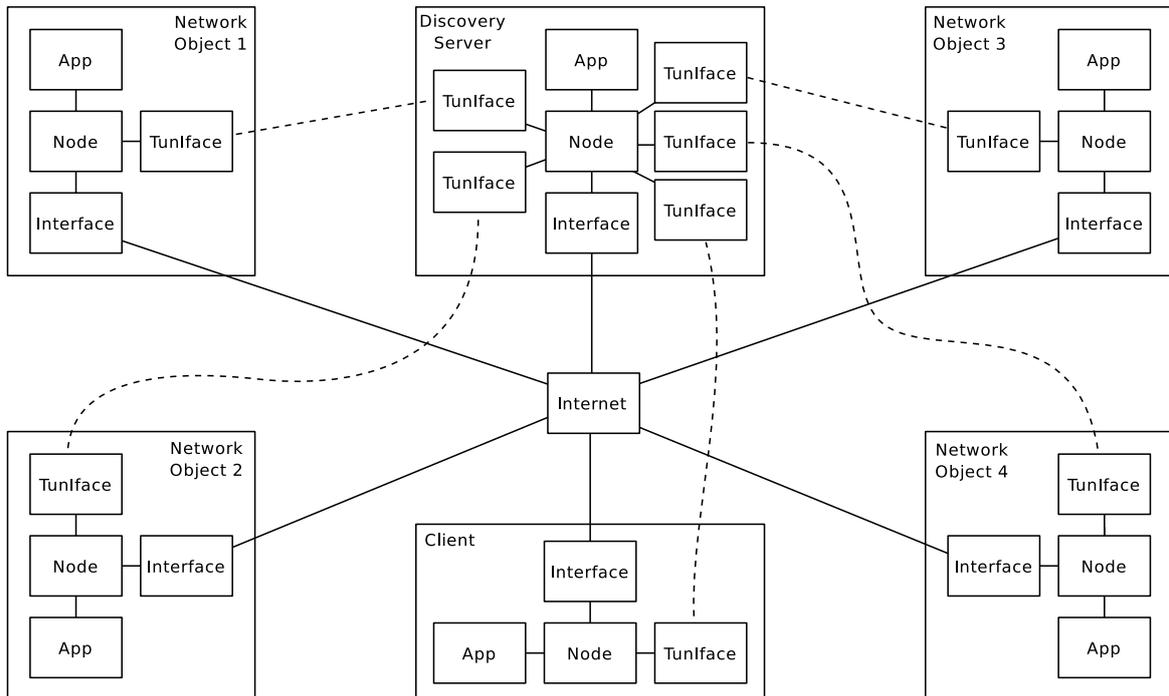


Figure 4.28: Description of the experiment with the proposed ontology.

network nodes participating in the experiment. After that, we create two new NEPI description nodes, representing the node and its interface, for each real node involved in the experiment. Then, we connect the interface description node with the node description node and with the Internet description node. To permit the packet capture, we need to establish tunnels (tun interfaces) between the node of the discovery server and the nodes of the client and network objects. To get this, we create the corresponding NEPI nodes to the tunnels and instruct each of them to grab a *pcap* file with the packets crossing the tunnel. Once we have the nodes and tunnels we create a NEPI node for each application, indicating the deployment package that should be used, as well as the build and execution instructions. Finally, we set the experiment controller of the NEPI description to execute the experiment and retrieve the results obtained from the standard output of the applications and the *pcap* files obtained from the tunnels.

Using the same experiment description in NEPI we execute two different experiments for each solution (DNS-SD and ODP), resulting in four executions. The first experiment of each solution performs one resolution of the information object and one resolution of the color printer, independently of the number of necessary exchanges to obtain them. From each resolution the client obtains a description of the requested network object. The second experiment performs 1000 iterations of the same operations, chaining both

4. Digging Deeper: Evolving Functional Blocks

Table 4.1: Packets of the single iteration experiment with ODP.

Timespan	Src. IP Addr.	Dst. IP Addr.	Size (bytes)
0.000000	10.0.0.2	10.0.0.1	327
0.014673	10.0.0.1	10.0.0.2	150
0.016339	10.0.0.2	10.0.0.1	406
0.027351	10.0.0.1	10.0.0.2	168

Table 4.2: Packets of the single iteration experiment with DNS-SD.

Timespan	Src. IP Addr.	Dst. IP Addr.	Size (bytes)
0.000000	10.0.0.2	10.0.0.1	62
0.002565	10.0.0.1	10.0.0.2	117
0.003094	10.0.0.2	10.0.0.1	71
0.005291	10.0.0.1	10.0.0.2	203
0.005659	10.0.0.2	10.0.0.1	71
0.007754	10.0.0.1	10.0.0.2	132
0.008087	10.0.0.2	10.0.0.1	62
0.009941	10.0.0.1	10.0.0.2	94
0.010348	10.0.0.2	10.0.0.1	76
0.012247	10.0.0.1	10.0.0.2	214
0.012692	10.0.0.2	10.0.0.1	71
0.014458	10.0.0.1	10.0.0.2	149
0.014870	10.0.0.2	10.0.0.1	71
0.016709	10.0.0.1	10.0.0.2	148
0.017092	10.0.0.2	10.0.0.1	71
0.018934	10.0.0.1	10.0.0.2	132
0.019306	10.0.0.2	10.0.0.1	62
0.021174	10.0.0.1	10.0.0.2	94

resolutions (information object and color printer) in each iteration. This will give us more confidence in the time measurements.

Results

Once we executed the experiments and obtained the resulting standard output of the applications and *pcap* files from the tunnels, we processed them to get the measurements discussed above. Then, we compared the results obtained from the DNS-SD experiments and the ODP experiments.

From the results of the single iteration experiments we can know the number of message exchanges used by DNS-SD and ODP. On the one hand, Table 4.1 shows the packets

exchanged between the client and discovery server during the ODP experiment. The upper section includes the messages exchanged to resolve the information object and the lower section includes the messages exchanged to resolve the printer object. It shows that each network object description can be resolved in just two messages. On the other hand, Table 4.2 shows the packets exchanged between the DNS-SD client and server during its corresponding experiment. It shows that the first resolution (information object), upper section of the table, needs 8 messages and the second resolution (printer) needs 10 messages. This is due to the fact that during the second resolution the client finds two different matchable objects (the two printers) and needs to obtain the description for both of them in order to know which object the client really wants.

In these results we can see that the DNS-SD solution requires, at least, 4 times the number of messages required by ODP. Moreover, when there are many services matching the initial query, the DNS-SD solution adds two extra messages. In terms of size, although the individual message size in DNS-SD solution is lower, ranging from 62 to 214 bytes in contrast with the 150-406 bytes of the ODP solution, the final transferred bytes in the whole resolution is higher. The resolution of the information item takes 812 bytes in DNS-SD and 477 bytes in ODP. In addition, the resolution of the printer takes 1088 bytes in DNS-SD and 574 bytes in ODP. That said, ODP outperforms DNS-SD in terms of total size and number of packets.

Now, using the time traces obtained from the measurements performed by the client applications used in the experiments that executed 1000 iterations we extract the statistics (average, median, minimum, and maximum) of the time spent in each operation. Figure 4.29 shows the cumulative distribution function (CDF) of the time spent in the discovery operation of the information item and the proper printer both by DNS-SD and ODP. It demonstrates that, on the median (50% percentile), DNS-SD spends 7.98 milliseconds (ms) to resolve the information item (webpage content) while ODP spends 11.45 ms. Moreover, DNS-SD spends 10.16 ms to resolve the printer object while ODP spends 12.88 ms. In contrast, on average, DNS-SD correspondingly spends 8.43 ms and 10.59 ms to resolve the information object and the printer object, while ODP correspondingly spends 11.36 ms and 12.79 ms to resolve the information object and the printer object. The proximity of the average and median indicates that the behavior of both mechanisms is very stable. Comparing the results, ODP overloads DNS-SD in a little more than 3 ms when DNS-SD only finds one matching element and a little more than 2 ms when DNS-SD finds two matching elements. Therefore, the gap between DNS-SD and ODP is reduced as the number of matching elements increases.

4. Digging Deeper: Evolving Functional Blocks

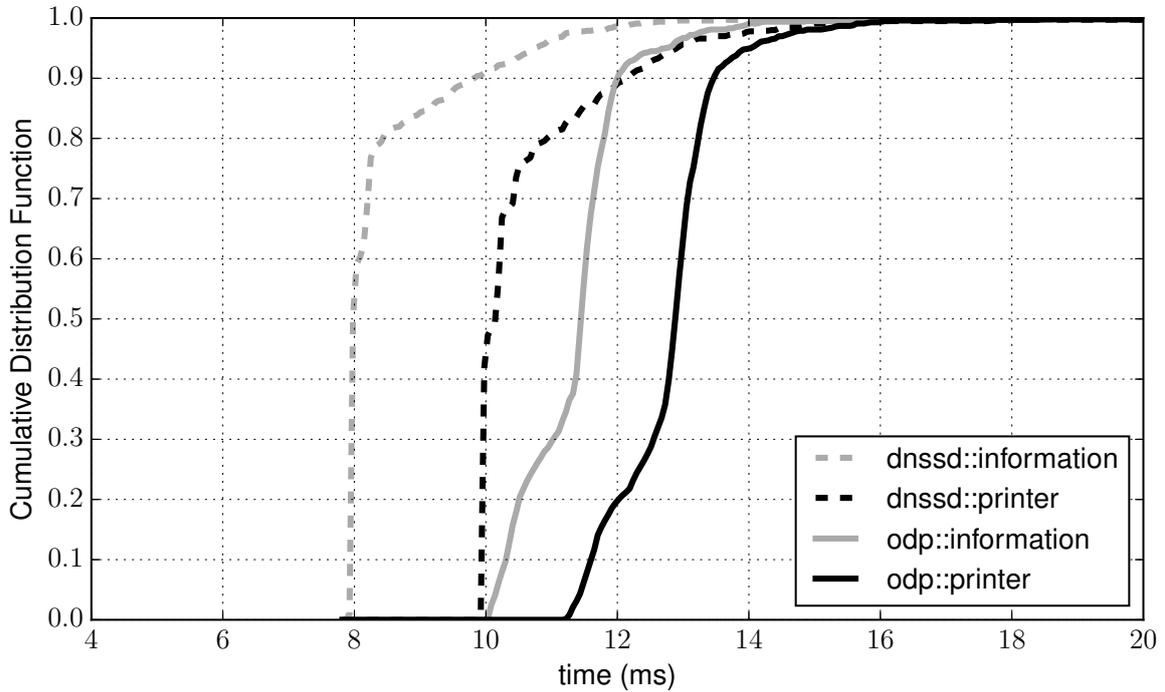


Figure 4.29: CDF of the time spent in each discovery operation for DNS-SD and ODP.

In summary, DNS-SD requires much more exchanges and twice the transferred bytes to resolve the same object but costs less time. This overhead resides in the processing of the SPARQL query but, however, has a little impact on the resolution time. Moreover, in networks with high latency and in cases when there are many matching elements (e.g. many printers), the DNS-SD solution reduces its performance and ODP provides a high benefit. Anyway, the expressiveness provided by ODP may be enough reason to accept the little overhead (3 ms). Finally, as the maximum packet size in DNS-SD is less than ODP, it could fit better than ODP in some scenarios but, as ODP needs less total bytes, it helps to reduce the traffic of the Internet.

4.8.4 Evaluation of IBCP

In this subsection we evaluate IBCP, as described in Section 4.6. We first compare the capabilities of our approach with HIP, as both have some common mechanisms. Then we analyze the mobility operation of the IBCP in comparison with HIP and LISP to show the performance of our approach. Finally, we discuss the experimentation results we have

obtained to demonstrate the performance of the lookup approach used by the IBCP, which is derived from ODIN.

Conceptual Comparison with HIP

As introduced in Chapter 2, HIP provides a mechanism to achieve locator/identifier separation by defining the Host Identity Tag (HIT) as the identifiers and network addresses (normally IP addresses) as locators. It also provides a mobility solution as described in [117]. It states that when a host moves to a new network and obtains a new address, it must send an *HIP UPDATE* packet with a *LOCATOR* parameter indicating the new address to any other party to which it is communicating. Then, this packet is acknowledged and the handover process is finished.

When the communication is established in a secure way through, for example, an ESP tunnel, keys must be negotiated again. Moreover, to support the simultaneous mobility of two hosts that are communicating, HIP proposes its Rendezvous Extension [118]. It states that a rendezvous server (RVS) intermediates in the first message exchanges, so a host that move will update the RVS with its current locator. The RVS will send the message to the destination and then the entities will communicate directly.

In comparison to our proposal, while HIP requires the RVS to globally know all HIT/address mappings, in our architecture, each node of the DTE_i knows only the identifier/locator mappings of its domain. When entities from different domains communicate, the DTE_i nodes of those domains interact to establish the communication and to exchange the necessary identifier/locator mappings. Also, when a host moves to a new network, instead of trusting in peers to directly communicate locator changes to each other, our architecture uses the IBCP, which uses the DTE_i to provide a trusted path to communicate the updated identifier/location mappings.

When our architecture is instantiated together with an address-based network, gateways are used to deal with the identifier/locator resolution, like in Proxy Mobile IPv6 (PMIPv6) [108], so the mobility support in them is also transparent to the entities. In addition, when it is instantiated together with an overlay network, it does not need to update real identifier/locator mappings to the infrastructure, because it is location independent.

That said, our architecture introduces intermediate elements and more message exchanges to provide mobility support but with the great benefit of enhanced trust, privacy, and overall security. Moreover, our architecture provides other advantages over HIP:

4. Digging Deeper: Evolving Functional Blocks

- HIP relays on DNS to resolve names to HITs, HITs to addresses, and to obtain the RVS address when it is used. Instead, our architecture moves the need for a hierarchical DNS infrastructure to a local element (DTEi node). This enhances security and trust.
- Instead of name/address resolutions, our architecture is based on queries to (one or more) identity attributes to resolve the locator. This adds enormous flexibility to entity identification.
- HIP identifiers (HITs) are unique for each host. Our architecture permits to dynamically change identifiers. This prevents traceability and increases privacy by changing identifiers between sessions.
- HIP lacks support for identity or identifier negotiations, so any entity can resolve and try to contact any other entity. Our architecture provides identity-based negotiations to enhance security by preventing unauthorized entities to obtain the identifier/locator mapping.
- Finally, the RVS in HIP is outside the control of the identifier/locator mapping owner. Our architecture stores identifier/locator mappings in the DTEi nodes managing the identity domain of their respective owners. This increases security and improves scalability.

Apart from these differences, as discussed above, our architecture offers other interesting capabilities, such as the consideration of digital identities instead of hosts as communication endpoints.

Mobility Performance Analysis

In this section we analyze the performance of the mobility management approach included in the identity-based control plane and compare it with the mobility approaches from HIP and LISP. To perform this analysis we have built a mathematical model for each approach. Such models represent the cost in milliseconds (ms) of sending a message/packet from one endpoint to another during and after a handover. We have to notice that, from the beginning, we expect our approach to offer less performance than HIP or LISP because it adds extra security operations to ensure the security advantages discussed throughout the section.

To build proper mathematical models, as shown in Figure 4.30, we have defined a common scenario that maps the same elements to specific elements of each approach.

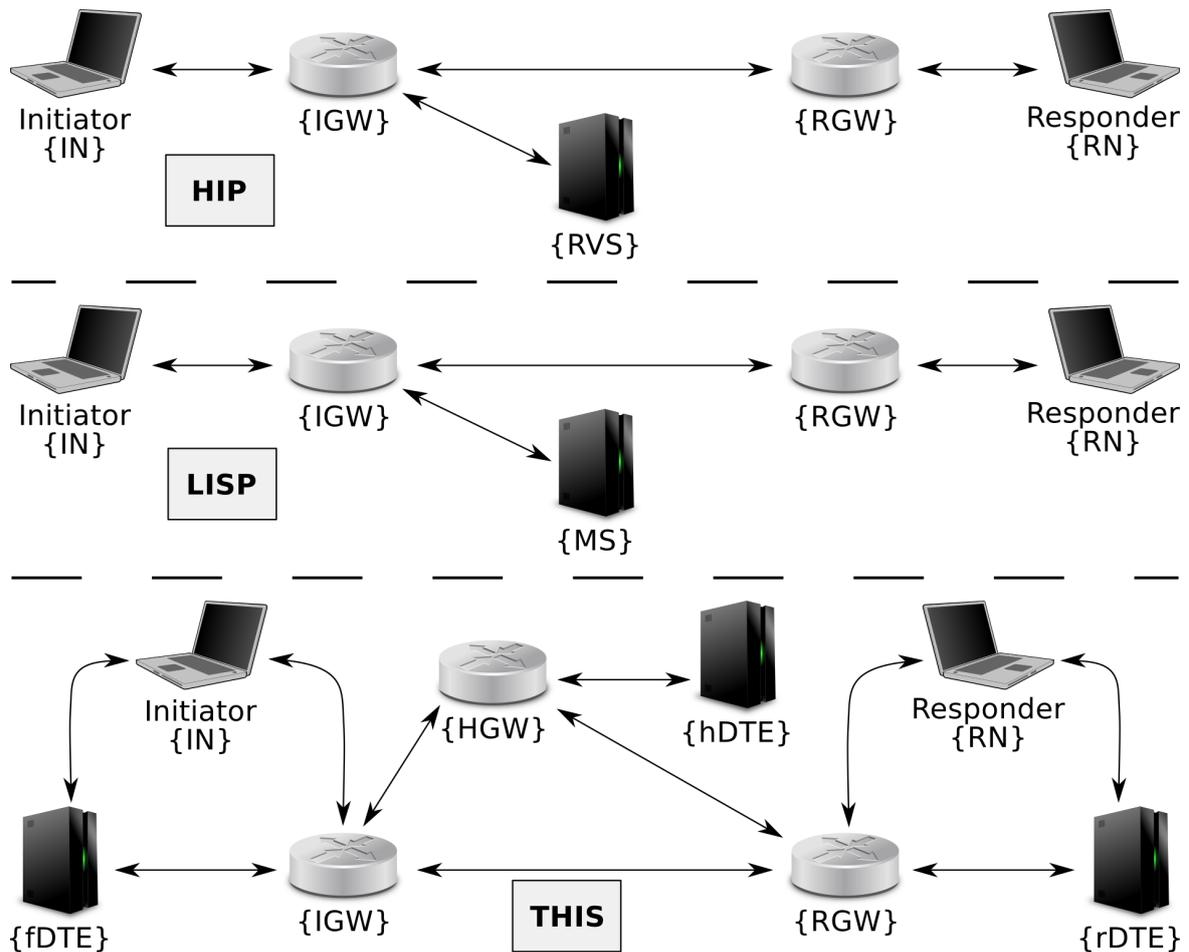


Figure 4.30: Scenarios used to build the mobility analysis models for the comparison of HIP, LISP, and our approach (THIS).

Therefore, we have a scenario with five elements: Initiator Node (IN), Responder Node (RN), Initiator Gateway (IGW), Responder Gateway (RGW), and Rendezvous Infrastructure (RI). These elements have direct mapping to HIP and LISP elements, including the RI, which is mapped to the RVS of HIP and the Mapping System (MS) of LISP.

In our approach (labeled THIS), the RI is mapped to the whole DTEi but since the operations inside it are complex we decided to represent the different DTEi nodes. Therefore, the RI is replaced by the hDTE (home domain), the fDTE (foreign domain), and the rDTE (responder domain). To better represent our approach, we also included the Home Gateway (HGW) into this representation.

4. Digging Deeper: Evolving Functional Blocks

Once we have defined the common scenario, we proceed with the definition of the base parameters used to construct the mathematical models as follows:

- $T_{a \rightarrow b}$: Time (ms) spent to transmit a message/packet from a to b , where a/b can be any of the elements defined above. We consider that $T_{a \rightarrow b} = T_{b \rightarrow a}$.
- N_d : Number of network/administrative domains.
- N_h : Average number of hosts per domain.
- α : Time (ms) spent finding an entry in a hash table per entry in the table.

Using these parameters we have built the equations that represent the intended mathematical models, which are used to calculate the cost (in milliseconds) of the handover and a following message/packet exchange between two nodes of different domains (IN and RN) for HIP (eq. 4.1), LISP (eq. 4.2), and the approach proposed in this section (eq. 4.3).

First we define the equation used for HIP. The handover starts when IN sends an id/loc update message to RN, going through IGW and RGW. Then, RN sends an acknowledgement to IN. To update the RVS, IN sends another id/loc update message to it through the IGW, and the RVS updates the corresponding record and sends an ACK to IN through the IGW. Finally, IN sends the message to RN through IGW and RGW, and RN sends its response to IN through RGW and IGW. The equation results as follows:

$$\begin{aligned}
 C_{\text{HIP}} = & T_{IN \rightarrow IGW} + T_{IGW \rightarrow RGW} + T_{RGW \rightarrow RN} \\
 & + T_{RN \rightarrow RGW} + T_{RGW \rightarrow IGW} + T_{IGW \rightarrow IN} \\
 & + T_{IN \rightarrow IGW} + T_{IGW \rightarrow RVS} + \alpha * N_h * N_d \\
 & + T_{RVS \rightarrow IGW} + T_{IGW \rightarrow IN} \\
 & + T_{IN \rightarrow IGW} + T_{IGW \rightarrow RGW} + T_{RGW \rightarrow RN} \\
 & + T_{RN \rightarrow RGW} + T_{RGW \rightarrow IGW} + T_{IGW \rightarrow IN}
 \end{aligned} \tag{4.1}$$

The handover in LISP requires updating the mappings in the MS so IN sends an id/loc update message to MS through IGW and the MS sends an acknowledgement, also through IGW. To communicate with RN, IN sends a message to IGW. This asks the MS to find the mapping entry and it sends the response to IGW. Then, IGW sends the message to the corresponding RGW which, in turn, sends it to RN. Now, RN sends its response to RGW and this again asks the MS to resolve the locator of IN. Finally, RGW receives the

locator of IN and sends it the response message through IGW. The resulting equation is as follows:

$$\begin{aligned}
C_{\text{LISP}} = & T_{IN \rightarrow IGW} + T_{IGW \rightarrow MS} + \alpha * N_h * N_d \\
& + T_{MS \rightarrow IGW} + T_{IGW \rightarrow IN} \\
& + T_{IN \rightarrow IGW} + T_{IGW \rightarrow MS} + \alpha * N_h * N_d \\
& + T_{MS \rightarrow IGW} + T_{IGW \rightarrow RGW} + T_{RGW \rightarrow RN} \\
& + T_{RN \rightarrow RGW} + T_{RGW \rightarrow MS} + \alpha * N_h * N_d \\
& + T_{MS \rightarrow RGW} + T_{RGW \rightarrow IGW} + T_{IGW \rightarrow IN}
\end{aligned} \tag{4.2}$$

The handover process of the approach proposed in this section, as depicted in Figure 4.8, begins with the initiator node (IN) that has moved to a foreign network and uses the foreign DTEi node (fDTE) and foreign gateway (IGW) to send an encapsulated message to its home DTEi node (hDTE) with the new location for its session ID. The home DTEi node (hDTE) sends the new location to the DTEi node assigned to the responder node (RN) and both of them send in parallel an update loc/id map to the gateways. The home DTEi node (hDTE) also sends the foreign DTEi node (fDTE) an update loc/id map with both IN and RN entries, which in turn sends it to its gateway. Finally, the home DTEi node (hDTE) sends a confirmation to IN through fDTE and the corresponding gateways (HGW and RGW). The process is translated to an equation with the same parameters used above and results in the following equation:

$$\begin{aligned}
C_{\text{THIS}} = & T_{IN \rightarrow fDTE} + T_{fDTE \rightarrow IGW} \\
& + T_{IGW \rightarrow HGW} + T_{HGW \rightarrow hDTE} + \alpha * N_h \\
& + T_{hDTE \rightarrow HGW} + T_{HGW \rightarrow RGW} \\
& + T_{RGW \rightarrow rDTE} + \alpha * N_h \\
& + T_{rDTE \rightarrow RGW} + T_{RGW \rightarrow HGW} \\
& + T_{HGW \rightarrow hDTE} \\
& + T_{hDTE \rightarrow HGW} + T_{HGW \rightarrow IGW} \\
& + T_{IGW \rightarrow fDTE} + T_{fDTE \rightarrow IN} \\
& + T_{IN \rightarrow IGW} + T_{IGW \rightarrow RGW} + T_{RGW \rightarrow RN} \\
& + T_{RN \rightarrow RGW} + T_{RGW \rightarrow IGW} + T_{IGW \rightarrow IN}
\end{aligned} \tag{4.3}$$

4. Digging Deeper: Evolving Functional Blocks

Table 4.3: Parameter values and ranges.

Parameter	Default	Min	Max
α	0.02	0.002	0.04
N_h	50	5	100
N_d	10	2	25
$T_{GW \rightarrow GW} \mid T_{DTE \rightarrow GW}$	5	1	20
$T_{GW \rightarrow MS} \mid T_{GW \rightarrow RVS}$	5	1	20
$T_{N \rightarrow GW} \mid T_{N \rightarrow DTE}$	10	-	-

To facilitate the handling of the models we simplify them, generalizing all gateways (IGW, RGW, HGW) to GW, both endpoints (IN, RN) to N, and all nodes from the DTEi (fDTE, hDTE, rDTE) to a specific DTEi node. Thus, the simplified models are as follows:

$$C_{\text{HIP}} = 10 * T_{N \rightarrow GW} + 4 * T_{GW \rightarrow GW} + 2 * T_{GW \rightarrow RVS} + \alpha * N_h * N_d \quad (4.4)$$

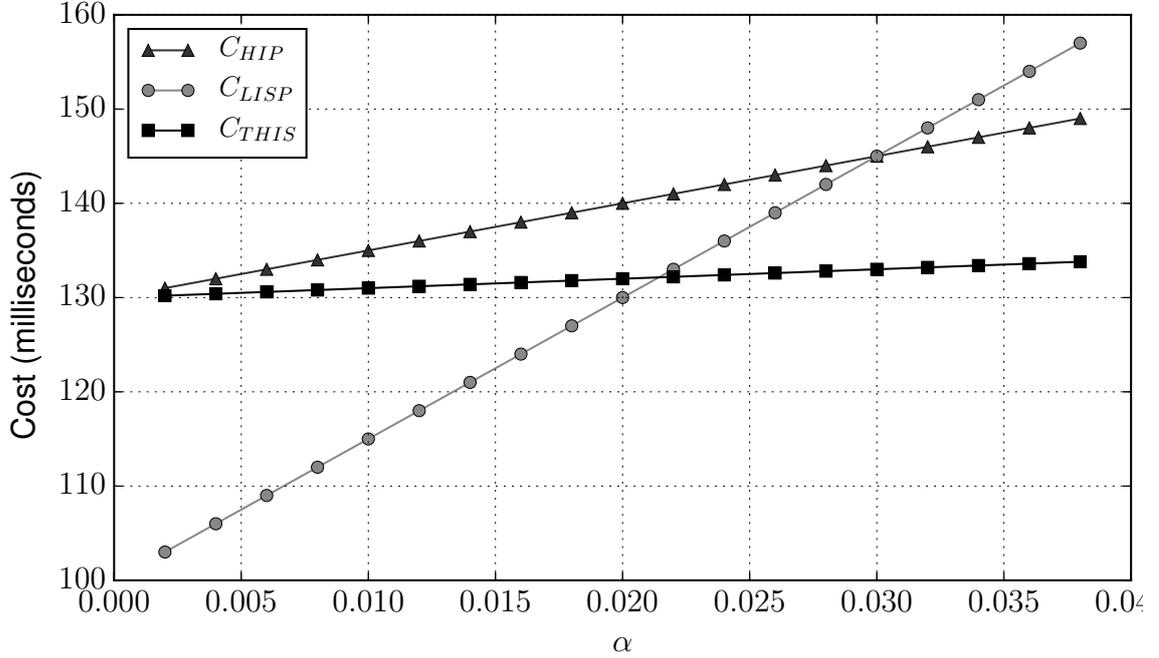
$$C_{\text{LISP}} = 6 * T_{N \rightarrow GW} + 6 * T_{GW \rightarrow MS} + 3 * \alpha * N_h * N_d + 2 * T_{GW \rightarrow GW} \quad (4.5)$$

$$C_{\text{THIS}} = 2 * T_{N \rightarrow DTE} + 8 * T_{DTE \rightarrow GW} + 6 * T_{GW \rightarrow GW} + 2 * \alpha * N_h + 4 * T_{N \rightarrow GW} \quad (4.6)$$

Once the models have been defined, we run them by using the values and ranges shown in Table 4.3 to assign their variables. Default values are set from experience from previous experiments in real networks but in order to obtain valid results we decided to use wide intervals (from *min* to *max*) that include as much real cases as possible. Thus we have a wide spectrum of values for the parameters.

For instance, we set transmission times for each hop to vary from 1 ms, which is very low and is typically found in wired links (1 hop), to 20 ms, which is somewhat high and is typically found in wide networks (10-15 hops). Moreover, these wide intervals emphasize the points where the plots intersect, which are the points of interest from the results, together with the slope of each plot.

With the outputs from the models we have built the plots shown in Figures 4.31, 4.32, 4.33, and 4.34, which show the analysis results for the equations 4.1, 4.2, and 4.3

Figure 4.31: Analysis results, variation of α .

with the parameters taken from Table 4.3. First, Figure 4.31 shows the results for the three approaches when varying the α parameter between the defined range boundaries. It shows that LISP is much better than HIP and our approach for lower values of α , but for $\alpha > 0.025$, our approach improves on the others. Next, Figure 4.32 shows that, for the variation of the average number of hosts per domain, our approach improves on the others from 60 average hosts per domain onwards. Figure 4.33 shows the results for the variation of the number of network or administrative domains, and also shows that our architecture improves on the others when there are more than 12 domains.

As discussed above, the first three architectures have a similar behavior when varying α , N_h , or N_d . In contrast, Figure 4.34, which represents the variation of elapsed time to contact a gateway from other gateway or from a DTEi node, shows a very different picture. It shows that the overhead of our approach over HIP or LISP increases with $T_{GW \rightarrow GW}$. This is the main drawback of our approach for the selected parameters, but it is not a big problem, since the time between GWs is not expected to grow over 5 ms but rather to decrease in the future. However it is something we have to analyze in future iterations of our approach.

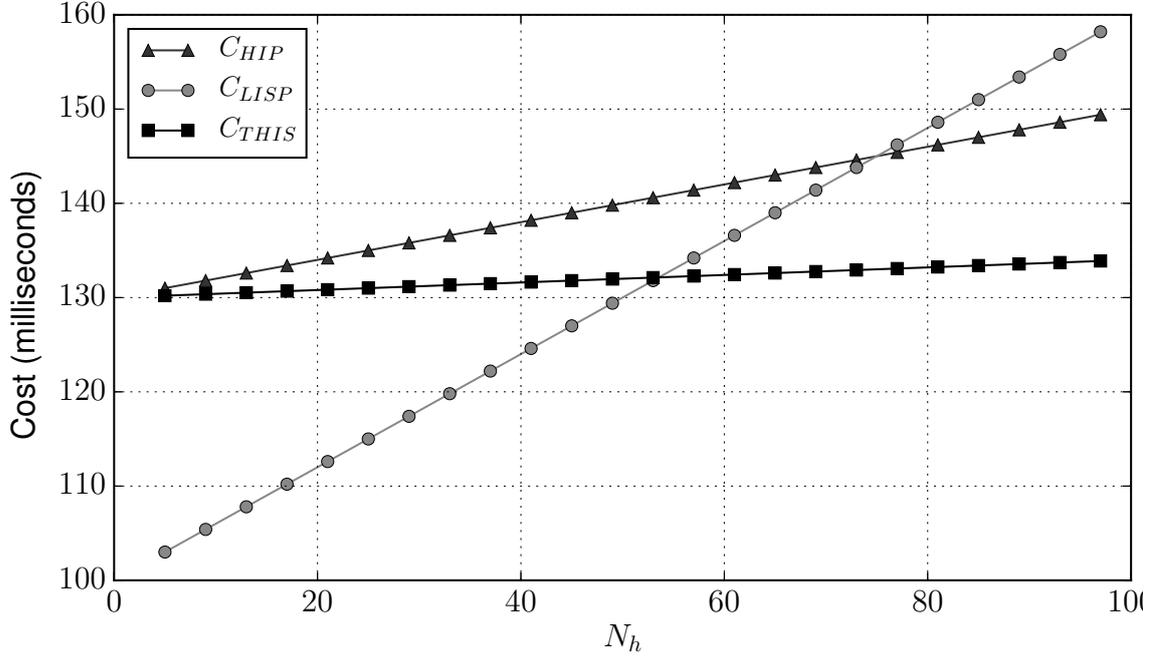
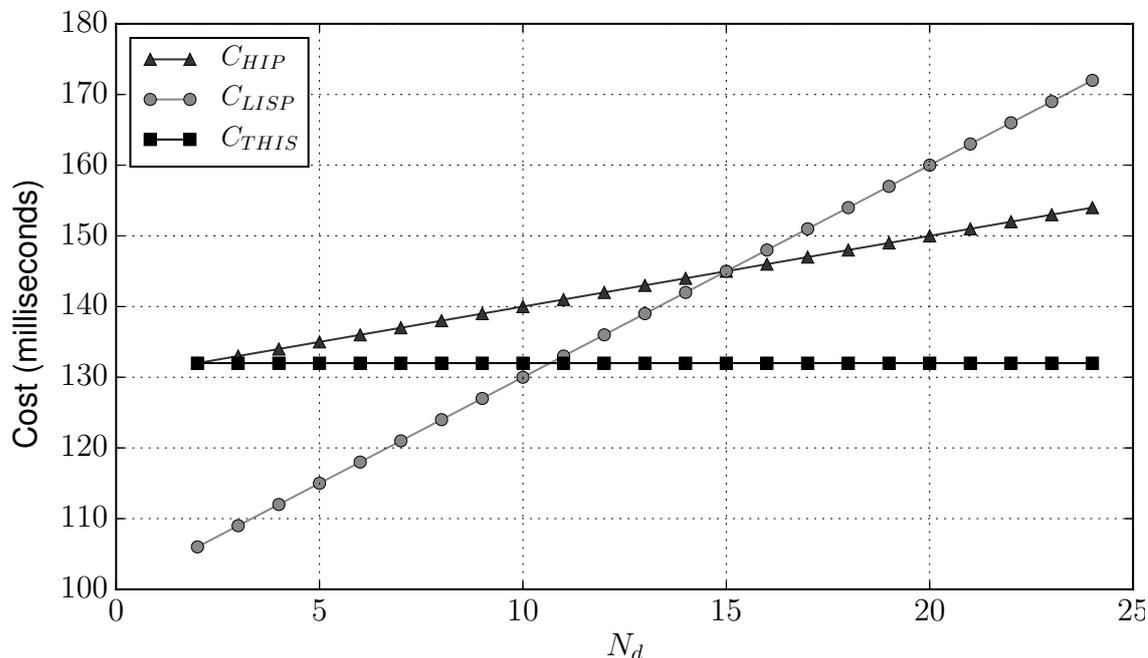


Figure 4.32: Analysis results, variation of N_h .

4.9 Conclusions

The architecture that has been researched throughout this thesis has the objective of enabling *identity-to-identity* communications for the network to mirror the real world and the way people, machines, services, and objects in general communicate with each other. Within this chapter we presented the functional blocks we designed to evolve the initial and rudimentary approach presented in Chapter 3 towards the design and construction of a complete architecture for the FI.

First, we have particularly addressed and discussed how the introduction of a secure and trustworthy infrastructure, the DTEi, built as an overlay approach is key to achieve this goal. This infrastructure will be the core of our proposal and its functions will be also interlinked with other functional blocks so, for instance, we have used the qualities provided by the DTEi to build IBNP, the network protocol that brings practical *identity-to-identity* networking to network objects. Therefore, this protocol reinforces the objective of placing digital identities in the middle of communications. It also keeps overall security and privacy, preventing traceability of the entities that communicate. We have then evaluated the protocol and demonstrated how it can integrate its qualities with other network

Figure 4.33: Analysis results, variation of N_d .

architectures, showing how to enhance the privacy in CCN, which is an outstanding approach that follows the ICN model and targets the FI.

Achieving practical privacy with the introduction of the proposed *identity-to-identity* network model has meant that we have designed IBNP for the provision of essential qualities for the FI. They are summarized as follows:

- The traceability prevention is achieved by the dynamic negotiation of the identifiers used as communication endpoints. This is performed by IBNP and, when available, through the DTEi so the underlying identifiers are only known by the entities involved in the specific communication instance, and they are only valid for such session. For example, in ICN architectures this function is used to dynamically negotiate the content identifier used by content publishers (providers) and subscribers (consumers).
- Using both IBNP and the DTEi, the network elements they introduce will intermediate during the initiation of communication instances (session start) to securely identify the identities taking part of them without needing to reveal the actual identities of the entities involved.

4. Digging Deeper: Evolving Functional Blocks

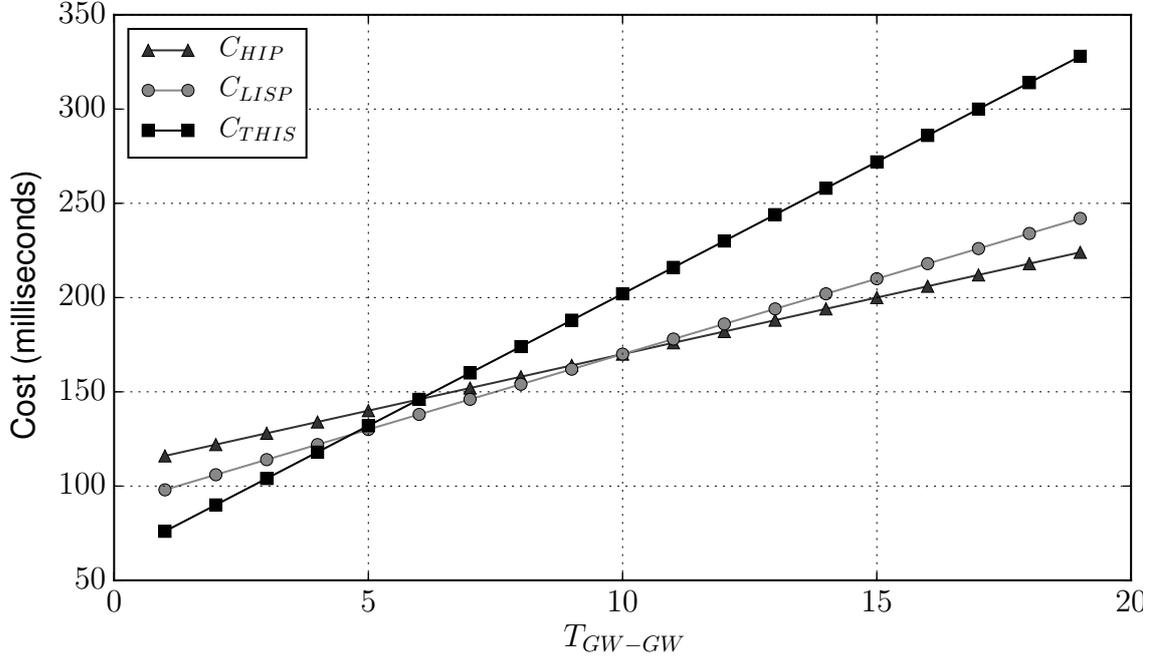


Figure 4.34: Analysis results, variation of $T_{GW \rightarrow GW}$, $T_{DTE \rightarrow GW}$, $T_{GW \rightarrow MS}$, and $T_{GW \rightarrow RVS}$.

- Since the protocol and architecture we are building abstracts entities to virtual identities and places them as communication endpoints, any entity of any type can take part of a communication, including persons, machines, services, groups of entities, etc. We generalize this concept as network objects and their virtual identities enable them to establish role-based endpoints.
- During security negotiations, e.g. during session establishment, IBNP enables any participant to request identity attributes and they will be dynamically provided if permitted by policies.

To ensure these qualities are effectively achieved we formalized the exchanges and verified they are performed securely by using an automated protocol security verification tool. We built a specific model of IBNP, including some interactions with the DTEi, and added the necessary instructions to check that the session identifiers used by communication parties are not disclosed to other entities and that they can not be mapped to the entity they correspond. Moreover, we demonstrated the feasibility of IBNP by evaluating proof-of-concept implementations built on top of CCN and XMPP, whose

results demonstrate the scalability of our proposal, adding less than 10 ms to each message exchange on top of CCN and less than 1 ms when running on top of XMPP.

Using some functions already present in the DTEi and IBNP as basis, we also designed ODIN, a general and global registry and discovery mechanism for network objects that targets the objectives for FI, as mentioned throughout this thesis. It is, of course, based on identities, which represent the descriptions of the network objects. At this time we introduced the concept of *partial identity* which is an incomplete description of a network object. This concept is provided to ODIN by a requester in order to find the target network object.

The main innovations provided by ODIN are as follows. First, ODIN allows network clients to represent their searches in a natural way, by shaping their knowledge about a desired network object into the mentioned *partial identity* structure. Second, ODIN achieves the semantic search and discovery in overlay networks from those partial identities by taking advantage of the properties provided by *Bloom Filters* when applied to attribute sets, which provide representative identifiers for the identities. Third, including semantic technologies like RDF in the network layer allows the discovery architecture to evolve without varying the underlying mechanisms and keeping full compatibility. Furthermore, we have also evaluated ODIN to demonstrate our claims and show that it is feasible to achieve a robust, scalable, and flexible network object discovery mechanism based on partial object descriptions by organizing object identities in a DHT built from a variation of Kademlia and using *Bloom Filters* to obtain the indexing keys for both partial and complete objects descriptions.

In order to ensure the completeness of object descriptions we complemented ODIN with a specifically designed ontology. To do this we first stated the necessary requirements and design principles that both ODIN and the ontology will follow to fit in the FI. We also designed an independent protocol, ODP, that covers such requirements and formalizes the interactions between ODIN and network objects. We also analyzed the feasibility of using semantic queries to discover network objects by evaluating both the ontology and ODP by using a prototype implementation to execute some particularly designed experiments in an experimentation infrastructure built with a well-known framework, MyPLC¹ on top of the GAIA testbed (see Appendix B). Thus, the results we obtained can be easily translated to the real world.

The experiments also demonstrate the behavior of ODP in comparison with DNS-SD. We found that DNS-SD requires much more message exchanges than ODP to resolve any

¹My PlanetLab Central, <https://svn.planet-lab.org/wiki/MyPLC>

4. Digging Deeper: Evolving Functional Blocks

network object, being additionally increased when the initial query matches more than one object. We observed in the experiments that the maximum packet size is less for the DNS-SD solution than the ODP solution but ODP requires less total bytes in each resolution. Also, we demonstrated that the overhead in time of ODP, due to the processing of the SPARQL query, is just over 3 ms for simple queries and is reduced with increasing query complexity. Therefore, we believe that the little performance drawback of ODIN may be negligible when exploiting the enormous benefits it provides, particularly its semantic expressiveness and extensibility.

Deriving the functions designed within the DTEi, IBNP, and ODIN we finally designed IBCP, which is an identity-based control plane that evolves such functions to get a *pluggable* functional block that can be integrated within any network infrastructure, considering both current and future architectures. It provides comprehensive security and integrated mobility management, using an identity-based overlay network to place digital identities in the middle of communications. We have also evaluated IBCP by first comparing it to other architectures that target similar objectives, such as popular security-centric solutions to locator/identifier separation like HIP, demonstrating that our proposal stands out in the following points:

- Identities are used as endpoints, which are derived to dynamic identifiers, instead of static identifiers.
- Entity resolutions are performed in a natural way by representing queries as partial identities instead of strict domain-names.
- Endpoint identifiers can be changed securely and dynamically, improving traceability avoidance.
- Communication sessions are negotiated using participant identities and their authorization policies are enforced, so general security is enhanced.
- Scalability is improved because the identifier/locator mappings are distributed across the overlay network and DHT built by the DTEi and ODIN.

We have also demonstrated the security and performance of IBCP. First, we have formalized the network model proposed by IBCP in order to analyze the security of the proposed mobility management scheme with an automated protocol security analysis tool, as also used to evaluate the security in IBNP. It demonstrates that the handover protocol addressed by the proposed identity-based control plane is secure. Second, we built formal

performance models for IBCP, HIP, and LISP so we could compare them. Although in principle we expected our approach to add significant overhead to HIP and LISP because of its extra security capabilities, the analysis has demonstrated the opposite. The only drawback in our approach appears when message transmission time between gateways increases, which, in fact, is something that is supposed to decrease in the future.

To sum up, the research results discussed throughout this chapter demonstrate that the functional blocks we have designed provide the necessary qualities that meet the requirements we targeted for the FI so they are properly backed up to form the foundation of the final architecture, as we discuss in the following chapter.

4. Digging Deeper: Evolving Functional Blocks

Chapter 5

Final Architecture: Beyond the Separation of Identifiers and Locators

The Internet is continuously evolving from a static, host-based, uniquely attached node model to a mobile, host-free node model with the possibility of multiple attachment points to the network. However, as stated throughout this thesis, the current Internet was not designed to support this type of workload because of its strict addressing mechanism. Thus, the Future Internet promotes the introduction of new architectures that provide the decoupling of identification and location. Moreover, new technologies, like *Cloud Computing* and *Internet of Things*, raise the necessity for a finer granularity of network nodes. Furthermore, the huge number of devices connected to the Internet and their finer granularity impose the necessity of flexible naming and, thus, integrated discovery mechanisms.

In this chapter we join together all functional blocks discussed previously in this document to build an architecture that goes beyond the simple separation of identifiers and locators to effectively decouple the identification and location processes. It is achieved by using identities to identify the network objects, which represent a higher abstraction than network nodes, and moving from a host-to-host to a fine-grained process-to-process view of the network. Together with the mobility and multi-homing support, it also provides integrated discovery, flexible naming, and integrated security features. Finally, we analyze the architecture to discuss its performance and compare it with other (existing) approaches.

The remainder of this chapter is organized as follows. In Section 5.1 we contextualize the capabilities provided by the architecture in the current network environment. In Section 5.2 we describe the final architecture that joins the DTEi, IBNP, and ODIN. Then, in Section 5.3 we discuss the mapping of identities to communication sessions and

5. Final Architecture: Beyond the Separation of Identifiers and Locators

in Section 5.4 we discuss the mechanism used to map those sessions to the underlying network. Finally, in Section 5.5 we analyze the qualities and performance of INA and in Section 5.6 we conclude the chapter.

5.1 Introduction

The Internet, as any other network, is built as a global graph of interconnected networks, each built as a graph of connected nodes. Each node is attached to the network graph as a vertex and connected to other nodes by edges (network links). Thus, the current network and transport protocols are defined to work with this view as a premise, without considering any dynamism.

But this view is no longer valid. The multi-homing feature that is widely present today has made a node to be attached to two or more different vertices of the network graph. Moreover, the rise in mobile devices connected to the Internet has broken the static view of the network graph. Mobile devices change their point of attachment to the network very often and, even worse, during the handover processes, they may have two simultaneous points of attachment to the network.

These new circumstances of the Internet ecosystem, primarily affecting the network layer, has brought back the discussion about the decoupling of identifiers and locators to separate the node identification from host addressing in the network [2]. Moreover, this mechanism also intends to resolve the general routing scalability problem found in the current Internet. By addressing nodes by their identifier, the locators (IP addresses) may be structured to improve the routing table scalability by, for example, geographically distributing the network addresses.

It is clear that the separation of identifiers and locators imposes a new crucial requirement — the mapping from identifiers to locators. This challenge has been approached by many architectures and proposals but none has been widely accepted. Therefore, a new, highly scalable, and widely accepted mapping system is mandatory to achieve the decoupling of identifiers and locators.

There is a totally different problem in the evolution of the Internet. The current Internet model is based on the existence of *hosts* attached to the network, and the hosts form the aforementioned vertices of the network graph. With the proliferation of sensor networks together with the Internet of Things (IoT) [90], the boom of the *cloud computing*, and the virtualization technologies in general, a host is no longer the actual vertex of the network graph. Instead, the hosts may be composed of themselves and many other hosts

residing inside them. Thus, the inner and outer hosts are attached at the same point to the network. However, they may be logically connected to different networks. Therefore, there is no easy and manageable matching between the network graph and the actual network. This problem becomes even worse when mixed with the movement of the inner hosts (or processes) between outer hosts.

As discussed in Section 2.1, most of the common shortcomings of the current proposals are that they are host-based and do not provide flexible naming. In their architectures, the network nodes are hosts, so they are host-tied, and do not consider node granularity or provide mechanisms to facilitate the process mobility or node impersonation. Moreover, none of those proposals provide flexible naming because they use plain identifiers or URI-like naming, and the support for highly dynamic assignment and resolution of identifiers to locators is questionable in most of them.

The solution to this problem also goes through the separation of identifiers and locators but with increased granularity. This way, the underlying network structure is kept consistent. Moreover, the identifier-oriented network mechanisms are responsible for resolving the possible inconsistencies induced by the granularity, multi-homing, and mobility capabilities. This follows the separation of concerns design principle, which is widely applied in computer science. It helps each functional element to be centered on its own functions, so the complexity introduced by the aforementioned capabilities is easily achieved by using different mechanisms for the underlying network and the node identification.

The introduction of the node granularity into the network brings a derivative problem: the increased complexity of the identification procedure. With several millions of nodes connected to the Internet it is very difficult to assign comprehensive names to identify them. Although the address space is hugely increased with the introduction of IPv6, it only opens the space in the underlying network, which is still host-oriented. Today, the naming is resolved by the Domain Name System (DNS), but current applications are still bound to IP addresses, so it can not be used to identify a node reliably. Neither can the current DNS afford to the expected dynamic workloads of the future, primarily caused by the capabilities discussed here. Moreover, as commented above, the naming functions provided by the DNS are simple. They comprise little more than the resolution of a domain name to an IP address. A highly dynamic and flexible naming is necessary for the future.

In this chapter we also discuss the qualities of the final architecture proposed by this thesis. It is the integration of the DTEi, IBNP, and ODIN, which are described in Chapter 4, in order to build the Identity-Based Network Architecture (INA), which goes beyond the decoupling of identifiers and locators to build a network model that decouples

5. Final Architecture: Beyond the Separation of Identifiers and Locators

the actual identity behind a network node from the locators used in the underlying network. With it, each network node is identified by its identity, seen as a collection of attributes, instead of an IP address or a plain identifier. This resolves the decoupling of the location and identification mechanisms. Moreover, it also resolves the flexible naming because network nodes are naturally addressed by using the knowledge an originator node has about a target node. The node granularity is also resolved by the dynamic mapping of identities to identifiers and IP addresses, so the nodes of the identity-based overlay network are stable because their identity barely changes but can be freely moved through the underlying network without impacting their interactions. Therefore, the primary view of the network we propose with the architecture discussed here moves the network interactions from host-to-host to process-to-process.

5.2 Identity-Based Network Architecture (INA)

As deeply discussed throughout the development of the thesis and clearly stated at the beginning of this chapter, finding a coherent mechanism to achieve the separation of identification and location beyond the simple loc/id split is a key challenge for the Future Internet (FI). This comes together with the need for higher node granularity and flexible naming. In Chapter 2 we reviewed the most important proposals to resolve the id/loc split and how they are lacking in the node granularity and flexible naming. In this section we discuss the approach we designed from the integrating the functional blocks defined in previous chapters, the Identity-Based Network Architecture (INA), with the objective of providing an identity-based overlay network to achieve effective decoupling of the identification and location processes, as well as higher level of node granularity and flexible naming.

The key aspect of this architecture is the construction of an overlay network that permits the actual network nodes (processes) to be organized independently of the underlying network. Figure 5.1 shows an overview of the architecture. A host is composed of different processes, that may be sensors that are behind a gateway (the sensor is the process and the gateway is the host), virtual machines (VMs) that are deployed in a host, or, as their name suggests, processes running in a host. While the hosts are attached to the network and the Internet by interfaces identified by the host IP addresses, the processes are attached to the overlay network by endpoints identified by the identities of the processes. To summarise, the network communication moves from a host-to-host view (identification based on IP addresses) to a process-to-process view (identification based on identities).

5.2 Identity-Based Network Architecture (INA)

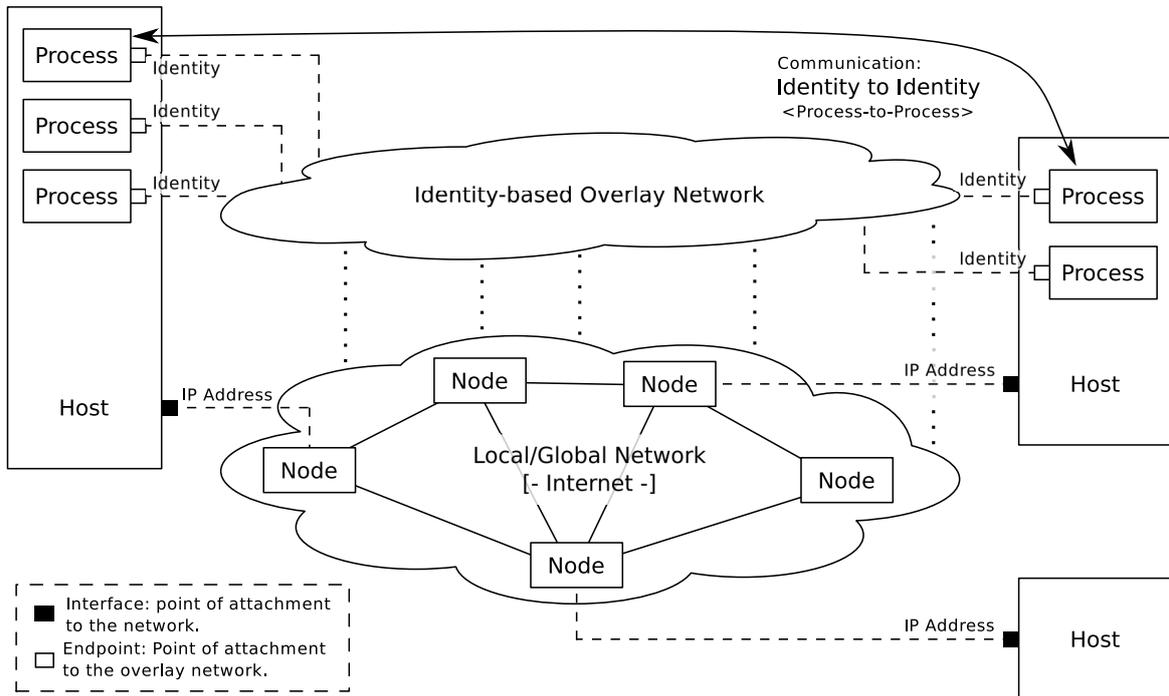


Figure 5.1: Overview of the final architecture.

The mapping between identities and IP addresses, for new applications, is divided in two stages. First, as is discussed in depth in Section 5.3, the identities are resolved to communication sessions. After that, as discussed in depth in Section 5.4, the communication sessions are resolved to IP addresses (network locators). It is worth noting that the underlying network of the architecture is considered to be IPv6 because it is a fact and is already widely deployed. However, both INA and, specifically, the identity-based overlay network may be deployed on top of many different underlying infrastructures by just adjusting the mapping functions of the second stage.

With this architecture we meet the global objectives sustained during the development of this thesis and raised at the beginning of this chapter. The decoupling of identification and location is achieved by using identities to identify processes and locators to exchange information through the underlying network. The flexible naming is also achieved by using identities. An identity, seen as a collection of attributes, is a natural way of identification of network nodes and permits enormous flexibility by describing network nodes with the acquired or preset knowledge items (attributes). In particular cases, the identities may have only one attribute, for example the MAC address of a host when identifying the whole host. Finally, node granularity is achieved by separating the identification of processes from the

5. Final Architecture: Beyond the Separation of Identifiers and Locators

identification of entire hosts and organizing them in the identity-based overlay network. Below we describe the particularities of the architecture.

5.2.1 Communication Sessions

As described in Section 5.3, to begin a communication between two processes (process-to-process), the processes first need to negotiate (begin) a session. This mechanism is resolved by the mapping between identities to sessions. A session is identified by a context identifier, which is a string of 16 bytes to identify a session and context shared by communicating identities. This size is chosen because, as described in Section 5.4, it fits in the tails (suffixes) of two IPv6 addresses, so the mapping between sessions and the underlying network does not add any overhead to the communication. It is not important to ensure the global uniqueness of these identifiers because they are only used in the scope of the identities involved in the communication.

For its part, a context includes a shared description of the session: the involved identities, their locations (IP prefixes), the negotiated network parameters, the security aspects of the communication, and any other information that may be shared in a session. This also includes any application-specific information, so applications do not need to transfer the *headers* of their protocols on each of their exchanged messages. They just need to set their information into the context and the counterparts are automatically notified. This mechanism reduces the meta-information overload of the messages/packets.

When two processes need to exchange different flows, they can be multiplexed on top of the same communication session. This is achieved with an upper-layer protocol, such as TCP or SCTP. However, the specific protocol to achieve it is outside the scope of this thesis.

5.2.2 Integrated Node Discovery

As discussed throughout this chapter, INA binds communications to processes instead of hosts. Thus, the processes are the nodes are interconnected to form the identity-based overlay network. In this overlay network, the nodes are addressed by their identities that should be known, to a certain extent, in order to establish a communication session. This, as described below, is achieved by the integrated identity-based node discovery, but first we give some definitions to describe the discovery mechanism.

First, a virtual identity is a full identity, seen as an attribute set, that can be used to unequivocally identify a node (process) or a set of nodes in the overlay network, although

5.2 Identity-Based Network Architecture (INA)

it may not correspond to a real identity. For instance, a virtual identity may be a reduced version of an existing identity that hides sensible attributes. Also, a virtual identity may be used for anonymity by hiding the name, address, or other attributes, or even be totally invented.

Second, a partial identity is an identity that only has an attribute subset of a virtual or real identity. In general, the partial identities are built by nodes when describing the knowledge (attribute items) they have about the identities of other nodes. Depending on their completeness, the partial identities are usually unable to be used in the precise identification of network nodes.

That said, the discovery mechanism offered by INA starts with a partial identity, which is completed to build a virtual identity. Thus, an originator node builds a partial identity with the knowledge it has about the identity of a target node. From it, the originator node obtains a virtual identity that can be used to unequivocally establish a session.

The functionality required to obtain the virtual identities is provided by a Distributed Hash Table (DHT) because the discovery mechanism should be scalable, highly fault tolerant, and totally decentralized. These properties are required to ensure the independence and self-management of these important network mechanisms. To build the DHT we decided to use an approach based on Kademlia [41] because its overlay network routing mechanism is very simple and efficient, and also because it is currently being used in some of the most important DHTs found in the Internet. As it is a descendant of the widely known Chord [35], it inherits some of Chord's most important features while providing a simpler metric and some improvements.

After the DHT mechanism has been defined, all nodes pertaining to the identity-based overlay network will be connected to the DHT, so they share their public identity information to the other nodes. Each DHT entry is a triple that contains an identifier, an attribute name, and an attribute value. It is stored in the DHT node whose identifier is closest to the result from applying the SHA1 cryptographic function to the attribute value. The SHA1 function gives a value with little clash probability with others and can be used with the XOR metric, as proposed by Kademlia. The identifier provided by the triple is used in the DHT to identify the virtual identity that contains the specified attribute and value. As shown in Figure 5.2, the discovery mechanism follows the process defined by RDFPeers [61] and is divided into four simple steps.

First, the identity-based overlay node receives a *begin session* instruction with a partial identity as the destination of the session. As commented above, this node may be a module integrated into the network node itself (process) or a separated network element.

5. Final Architecture: Beyond the Separation of Identifiers and Locators

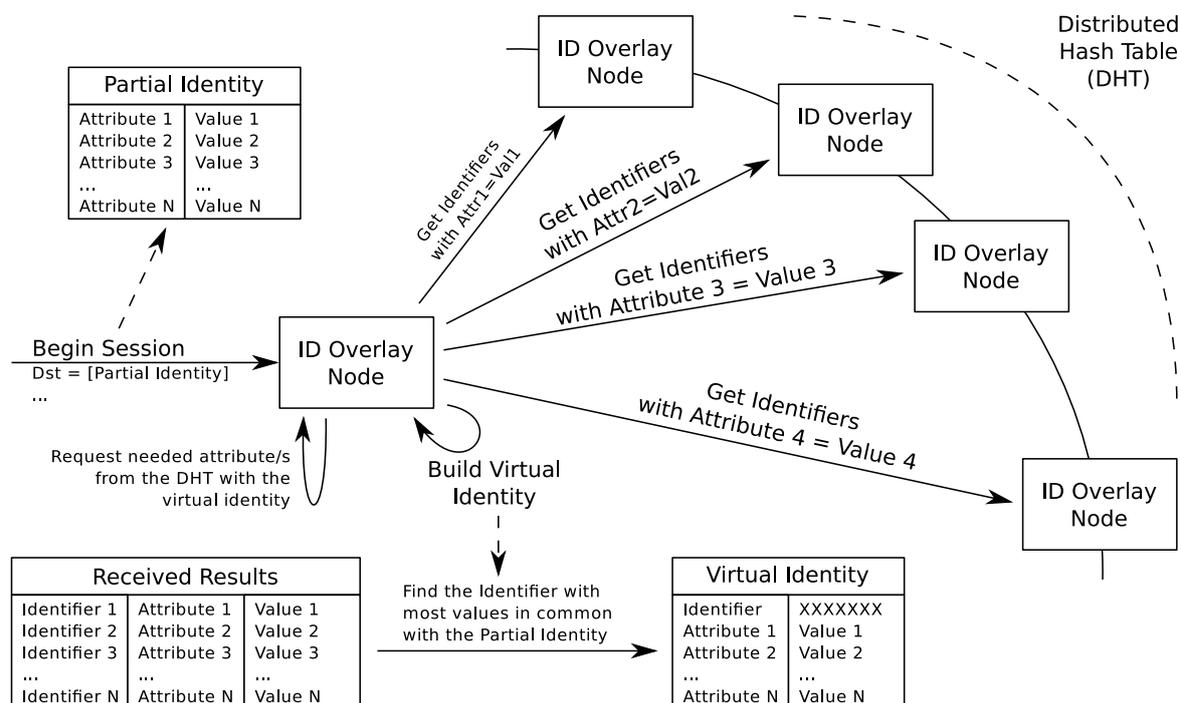


Figure 5.2: Process/Identity Discovery.

Second, the node builds as many identifier requests as necessary to obtain the identifiers associated to the attributes and values specified in the received partial identity. These requests are sent to the overlay node whose identifier is the closest to the result from applying a cryptographic function (SHA1) to the attribute value for each attribute of the partial identity.

Third, after receiving the responses from the other nodes of the DHT, the requesting node intersects all triples (identifier, attribute, value) to obtain the identifier of the requested virtual identity. This corresponds to the virtual identity that best matches with the provided attributes. When matching with more than one virtual identity, because the partial identity does not contain enough attributes to find only one or the requested virtual identities are very similar, the system launches a conflict resolution process, whose definition is outside the scope of this solution, and thus left to be implemented by applications or other higher layer mechanisms. This process may interact with the requester or apply a default policy in order to choose the virtual identity that best fits with the purpose of the requester.

Finally, the requester node requests a whole virtual identity or, at least, the attributes it needs by using the obtainer identifier. This virtual identity now has all the necessary

5.2 Identity-Based Network Architecture (INA)

information to contact with its corresponding node and request new sessions, as discussed in Section 5.3.

We need to note that, during the discovery process, some attributes or virtual identities may not be available, or prohibited. Thus, the requester node should contact directly with the node that holds the *secured* information in order to perform the necessary negotiations to obtain the permission. However, this mechanism is an advanced issue and outside the scope of the network protocol. It should be resolved by higher layer protocols.

To organize the identities in the DHT and build the whole discovery mechanism, the identities are defined as entries in RDF format [60]. Each identity is represented as a set of RDF triples in Turtle [95] and these are used as items to populate the DHT. Thus, each node (the network nodes, processes, or the nodes with the delegated responsibility) will build as many virtual identities as needed, assign a different identifier to each of them, and populate the RDF triples (identifier, attribute, value) to the corresponding node of the DHT. That is, each triple will be stored in the node with the closest identifier to the SHA1 result of its value field.

Finally, all the results obtained from the discovery operation, the virtual identities, may be cached to reduce the actual number of lookups, so it is an important feature. However, we decided not to include caching mechanisms to avoid problems in the protocol. Caches can be implemented by extensions to the architecture or by external entities, such as the applications or other higher layer mechanisms.

5.2.3 Mobility and Multi-homing Support

Two key aspects of the final architecture described here, INA, are the integrated mobility and multi-homing support. They are achieved as a result of the decoupling of the identification and location. For the mobility support, when a node moves, it changes its location but it is still consistently identified by its identity, so it can be reached without using special mechanisms like the Home Agent (HA) and binding update mechanism used in Mobile IP [119]. However, the sessions already started must be updated when a node changes its location. Since this is a change in the context, the context update message will report the new IP prefixes to the interested nodes. As previously discussed in [120], a special control message is sent to the counterparts of all communication sessions in the same way as the session is started, as described in Section 5.3. In Section 5.4 we detail how to obtain the IP addresses from the context identifiers and the IP prefixes.

For the multi-homing support, a process (network graph node in INA) may have different IP prefixes included in its virtual identity. The specific path of the underlying

5. Final Architecture: Beyond the Separation of Identifiers and Locators

network followed by the messages/packets of a communication context is determined by the IP prefix. It is obtained from a selection process that involves the network management and operators. This process is not dealt with here.

5.2.4 Interfaces for New and Legacy Applications

INA provides direct support for newly created applications by offering them a specific Application Programming Interface (API) to gain fine control on the partial identity definition used in the node (process) discovery and the context management. Thus, a new process needs just to define its *endpoint*, the identity associated to it, and connect it to the identity-based overlay network.

To support legacy applications, INA proposes the definition of an adaptation layer that maps IP addresses to context identifiers and backwards. This adaptation layer is offered through a virtual network interface defined in the operating system, so existing applications do not need to be modified to support the new functionalities. Moreover, the adaptation layer will transform DNS requests to identity discovery and session start requests, so each DNS resolution will define a new context identifier and a new virtual IP address (VIP) is sent in response to the application. When the application uses this VIP, the adaptation layer will perform a look-up in the VIP-to-CTX mapping table and find the corresponding context identifier.

Once new and legacy applications have obtained their destination context identifiers, the *Identity & Context Layer* maps them to source and destination network addresses and builds the actual IP packet, as described in Section 5.4.

5.2.5 Integrated Security

As previously discussed in [105, 120] and illustrated in Figure 5.3 and Figure 5.4, INA integrates the security from the beginning during the formation of the identity-based overlay network. While in Figure 5.4, the process becomes a node in the overlay network, by using the impersonation quality of the architecture, as shown in Figure 5.3, the process is not a node of the identity-based overlay network and it uses a trusted entity that is part of the overlay to perform its operations.

The mutual authentication of network nodes is achieved by the validation of the identity attributes. It is achieved by attaching a signature to each attribute, which is emitted by a trusted signing entity. For instance, banking information should be signed by a bank and

5.2 Identity-Based Network Architecture (INA)

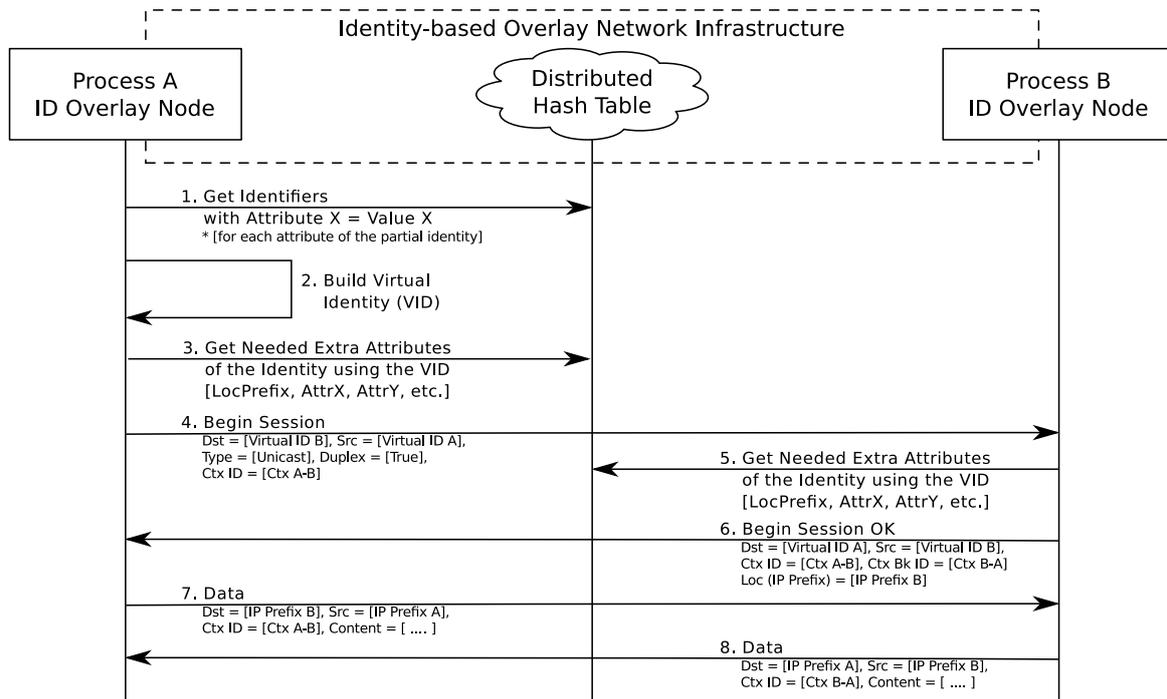


Figure 5.3: Starting a session without involving impersonation elements.

e-mail information should be signed by the corresponding organization. Thus, every node may determine the authentication level and the required strength of the attribute signing.

In turn, the authorization (access control) is also based on a trusted negotiation of the access based on the attributes provided by the identities of the originating and target nodes. Moreover, the access to non-public identity information is regulated by policies defined by the owner of the information. Thus, it is disclosed only to the authorized subjects by using the same attribute-based authorization method.

In addition, INA proposes and recommends to use an asymmetric encryption mechanism when the aim is to gain confidentiality. It may be inefficient and processor hungry but it has some benefits over weaker encryption mechanisms: 1) Transmitted information will be kept secret for longer; 2) There is no need to negotiate the security terms, with the speed-up it represents; 3) It fits perfectly and performs much better in publisher/subscriber underlying networks. In the future, processor performance improvements may make those methods much less processor hungry in comparison with other tasks. This does not prevent adapting and using other weaker encryption mechanisms (primarily symmetric) and key exchanges protocols, such as IKEv2 [121].

Finally, the security and trust in the negotiations is achieved by the inclusion of the intermediary elements, illustrated as *Overlay Node A* and *Overlay Node B* in Figure 5.4

5. Final Architecture: Beyond the Separation of Identifiers and Locators

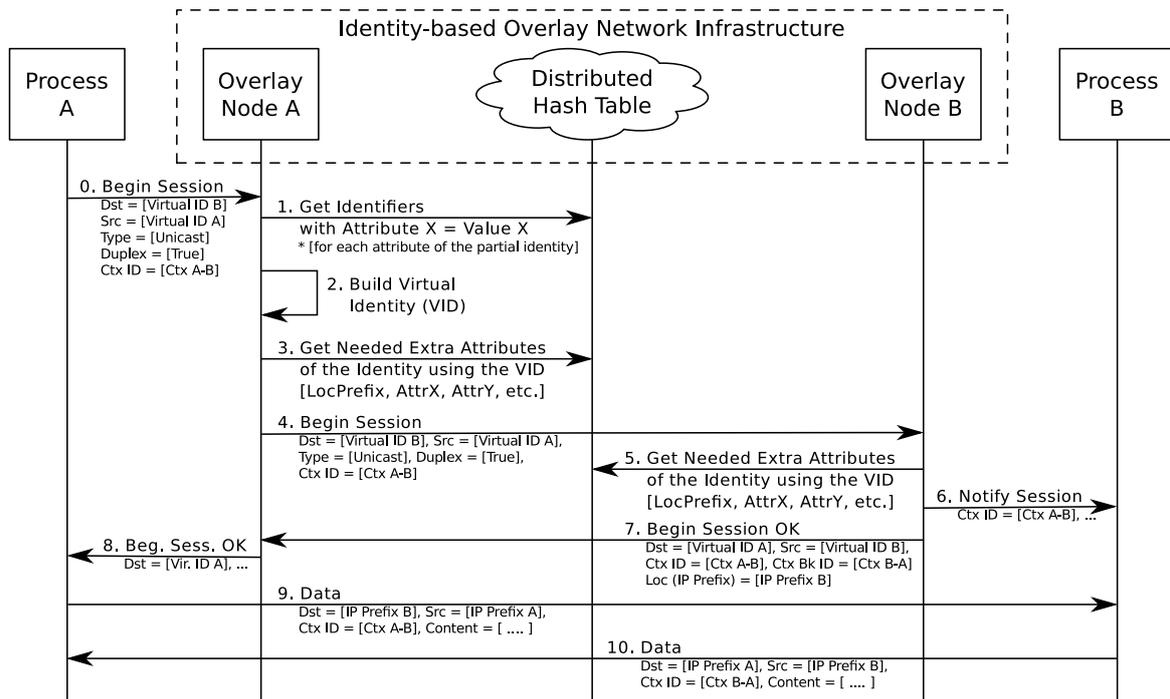


Figure 5.4: Starting a session involving impersonation elements.

and, when exercising their security mediator role, the nodes of the DTEi, as detailed in Chapter 4 and published in [105,120]. These elements are connected to each other to build a globally trusted infrastructure by the pre-exchange of their cryptographic certificates and assuring the confidentiality and authentication of their exchanges by using encryption and signature mechanisms.

5.2.6 Main Benefits of Using Identities

Throughout this chapter we have discussed the role that identities are playing in the node identification mechanism of INA. Below we discuss the main benefits provided for the Future Internet.

First, using identities instead of plain names, addresses, or identifiers, provides a natural form of identifying and addressing network nodes. When a node wants to contact with other node, it only needs to join the knowledge (identity attribute set) it has about the other node and the network architecture will find the way to resolve it to context identifiers and underlying network addresses.

Second, when identifying network nodes by their identity, the architecture provides integrated discovery and (secure) identification. The discovery is no longer a separate

5.2 Identity-Based Network Architecture (INA)

(previous) task. It is integrated in the session establishment stage. When a node finds a virtual identity from a partial identity it is actually performing a background network node discovery to complete the identity attributes and, at the same time by performing a *source* authentication and authorization process. Also, when a node validates the received virtual identity it achieves the secure identification by performing a *target* authentication and authorization process.

Third, by using identities, different network scopes are referenced in the same manner. When a node uses a very precise partial identity to establish its communication it may be referring to a specific individual network node. However, when a node uses a vague partial identity to establish a session, it is referring to multiple network nodes. This may be used to establish multicast or anycast sessions.

Fourth, in contrast with IP addresses, the communications may be achieved in an *identity-to-identity* way without taking into account the intermediate elements. For example, when an e-mail application sends a message to a destination it is actually sending the message to a mail server. By using identities, a message can be sent directly to the destination and, if it is not present, catch it automatically by an intermediate element that is *impersonating* the identity. With domain names and IP addresses this can not be achieved because a network node can not give impersonation rights to other network nodes for a specific identifier (name or address). The secure identification mechanism will rule this functionality, so it does not incur in a security penalty.

Fifth and finally, the identities may be seen as a generalization of current DNS-like domain names or URI-like identifiers. In fact, the domain names or URIs may be found as attributes of the identity, but also being split into different parts to form different attributes of the identity. This feature permits the architecture to inherit, adopt, and interact with any current infrastructure based on these types of identifiers.

5.2.7 Heterogeneous Underlying Network

Although the specific solution presented and analyzed here works on top of IP, particularly IPv6, its design permits it to work on top of many different underlying network architectures. The responses to the challenges imposed by the Future Internet (FI) have been reflected in the appearance of many different architecture proposals. Some of them, like HIMALIS [14], MOFI [33], or CCN [15], have high potential to be widely deployed. Therefore, our view of the FI is that there will be many network architectures and, thus, INA can be instantiated on top of them. Actually, in Sections 4.3 and 5.5.5 we show how to instantiate a functional block of our architecture on top of CCN, in Section 5.5.6 we

5. Final Architecture: Beyond the Separation of Identifiers and Locators

discuss how to integrate this architecture with HIMALIS, in Section 6.2 we delve into this integration, and in Section 6.3 we discuss how to integrate it with MOFI.

As an example of instantiation of INA on top of other approach for FI, as previously discussed in [105], we instantiated the architecture on top of CCN and demonstrated its feasibility through experimental results, as shown in Section 5.5. The specificity of this instantiation is that we designed a simple adapter to provide two-way communication on top of a single-way communication infrastructure. This is because CCN, being an Information Centric Networking (ICN) approach, is not based on IP addresses or any endpoint identifier. Instead, it identifies information, which is requested by the consumers and provided by the producers. The network matches each consumer request with the corresponding producer, so it builds a *path* to deliver the information. Thus, our adapter uses this concept to build identity-based endpoints, which are identified in the same way as the information is identified in CCN. Using the adapter, INA is deployed as discussed throughout the chapter, with the exception of the mapping from sessions to underlying network. This mapping is not necessary because CCN, together with our adapter, is able to deliver the messages using just the context identifiers.

Apart from being instantiated on top of other underlying networks, INA may be integrated within other architectures for the FI and thus provide them with some of its features. For instance, in [120], we propose to integrate one of the functional blocks of INA, particularly the DTEi, within the HIMALIS architecture to provide secure id/loc mappings. In this integration, the discovery and negotiation functions of INA are placed as hostname/identifier/locator resolution infrastructure. Thus, when an HIMALIS node requests the resolution of a hostname, the *edge* element of the DTEi (identity-based overlay network node) receives the request, maps the hostname to a host identifier (session identifier) using the mechanism discussed in Section 5.3, obtains the necessary locators, and reports them to the *gateways*. In HIMALIS, the gateways are the elements which perform the id/loc mapping, so they need to know the (session) identifiers and network locators. Finally, we evaluate this approach in Section 5.5 and demonstrate its feasibility.

5.3 Mapping Identities to Communication Sessions

As deeply discussed in previous sections, the network nodes (processes) need to map their identities to context/session identifiers before they start to exchange messages (communication). When the communication is expected to be duplex, messages flowing to and from a target node, the mapping should return two different identifiers. This is

5.3 Mapping Identities to Communication Sessions

because each identifier is only used to identifying the communication path and context followed by the messages from an origin to a destination.

However, this mechanism is more complex than a simple mapping operation. It involves the actual identity discovery, as described above, to obtain a virtual identity from a partial identity, as well as the *session* negotiation.

As introduced in the previous section, INA defines sessions (contexts) as dynamic associations of identifiers for identities that can be renewed (changed) at any time. It does not follow the session layer concept defined by the Open Systems Interconnection (OSI), which is more tied to current network *connections*. Instead, the session concept proposed here lies in the totally stateless *conversation* that is performed between two or more processes. Therefore, a context identifier (session) does not identify an identity but a specific communication and it can change during the session if the communication parties agree a new context identifier.

That said, the mapping operation that obtains a communication context/session identifier is translated to a negotiation that involves the identity discovery and session negotiation. However, despite its complexity, this process is still considered as a mapping operation from the higher level view of the architecture.

As shown in Figures 5.3 and 5.4, to obtain the session/context identifiers, a network node first needs to negotiate the permissions to establish a communication. This negotiation, discussed in depth in the previous section, involves some attributes of the identities of the nodes taking part of the communication.

The mapping process is quite simple but has two different variations. The first variation only involves the network nodes (processes) which are connected to the identity-based overlay network and form part of the Distributed Hash Table (DHT). The second variation also involves the network nodes (processes) but adds specific overlay nodes that are responsible for building the DHT and performing the negotiations. The utility of the second variation is twofold. First, it permits low-power devices to be connected to the identity-based infrastructure by delegating the negotiation responsibilities to the special elements. Second, as described in the previous section, it permits the construction of a totally trusted infrastructure to perform secure negotiations. Below we describe the stages composing each variation.

5. Final Architecture: Beyond the Separation of Identifiers and Locators

5.3.1 Independent Interactions

The first variation of the mapping operation to begin a session and obtain the context identifiers, as discussed above, just involves the communication nodes and their corresponding identities. The operation is shown in Figure 5.3 and has six steps.

First, the initiator process (Process A), following the discovery operation described in the previous section, defines a partial identity (PID) and, for each of its attribute values, requests the triples (identifier, attribute, name) to the corresponding node of the overlay network and DHT.

Second, with the responses received from the previous step, the initiator process builds a virtual identity (VID) with the already known attributes and the guessed (obtained) identifier.

Third, the initiator process requests from the DHT the extra necessary attributes to actually begin the session and obtain the context identifiers. One of the attributes is the location of the counterpart of the negotiation. It is the entity that will negotiate the access to the destination network node (Process B). In this variation, this locator (IP address) indicates the current locator of "Process B".

Fourth, the initiator process (Process A) sends a begin session message to the destination node by using the previously obtained locator. In this message, the initiator process includes the destination VID, its own VID, which is the identity it will use in the communication, the communication type (unicast, multicast, anycast, multicast), whether the communication will be duplex, and the randomly chosen context identifier that represents the session between the initiator and the destination (Ctx A-B).

Fifth, the destination (responder) process (Process B) receives the session request and gets the necessary extra attributes from the DHT by using the VID provided by the initiator node. Again, an important attribute is the current locator of the counterpart.

Sixth and finally, the destination process (Process B) sends back a session confirmation message to the initiator process. It includes the destination VID, the source VID, the previously defined context identifier (Ctx A-B), a new context identifier (Ctx B-A) to be used in the messages sent from Process B to Process A, and a final locator prefix (IPv6 prefix) to be used in the following messages.

Once operation has been completed, the initiator node starts its data message exchanges by mapping the context identifier to underlying network addresses, as discussed in Section 5.4.

5.3.2 Delegated Negotiations

The second variation of the mapping operation to begin a session and obtain the context identifiers, as discussed above, involves the communication nodes with their corresponding identities and the intermediate elements (trusted overlay nodes). The operation is shown in Figure 5.4 and has nine steps.

First (step 0), the initiator process (Process A) declares its intention to begin a session by sending its corresponding intermediate element (Overlay Node A, IE-A) a begin session message that includes the partial identity of the destination (PID), the virtual identity it wants to use in the session (VID), the communication type, whether it is duplex, and, finally, the randomly selected context identifier used in between the initiator and the destination (Ctx A-B).

Second, IE-A receives the begin session message and, following the discovery operation described in previous section, uses the provided PID to request, for each attribute value, its triples (identifier, attribute, name) from the corresponding node of the overlay network and DHT.

Third, with the received responses of the previous step, IE-A builds a virtual identity (VID) with the already known attributes and the guessed (obtained) identifier.

Fourth, IE-A requests from the DHT the extra necessary attributes to actually begin the session and obtain the context identifiers. One of the attributes is the location of the counterpart of the negotiation. This is the entity that will negotiate the access to the destination network node (Process B). In this variation, this locator (IP address) points to the intermediate element that represents "Process B", which is IE-B.

Fifth, IE-A sends the begin session message to IE-B using the previously obtained locator. In this message, IE-A includes the same fields provided by the initiator process but changes the PID by the obtained VID.

Sixth, IE-B receives the session request and gets the necessary extra attributes from the DHT using the VID provided by IE-A coming from the initiator node. Again, an important attribute is the current locator (IP address) of the counterpart, which in this variation is IE-A. Also, in this step, IE-B generates a context identifier for the way back from the destination to the initiator nodes (Ctx B-A).

Seventh, IE-B notifies the destination node (Process B) that a session has been started and specifies the corresponding context identifiers. If the session is simplex (one way), there is just one identifier to be notified.

Eighth, IE-B sends back a session confirmation message to IE-A. It includes the destination VID, the source VID, the previously defined context identifier (Ctx A-B),

5. Final Architecture: Beyond the Separation of Identifiers and Locators

the newly generated context identifier (Ctx B-A) to be used in the messages sent from Process B to Process A, and a final locator prefix (IPv6 prefix) to be used in the following messages in order to reach Process B.

Ninth, IE-A receives the session confirmation message and sends it to the initiator node (Process A). The message includes the information required to start the message exchange, such as the source and destination VID, to let the node know what VID the message relates to, the context identifiers for both ways (CtxA-B, Ctx B-A), and the current prefix used by Process B.

Once this operation has been completed, the operation continues like in the other variation described above, so the initiator node starts its data message exchanges by mapping the context identifier to underlying network addresses, as discussed in Section 5.4.

5.4 Mapping Sessions to the Underlying Network

In the previous section we described the necessary operation to map identities to communication sessions and thus to context identifiers. That operation dynamically associates a plain identifier to an identity to be used during a specific session, for specific purposes, and for a specific time. However, as described in Section 5.2, the overlay network nodes use the underlying network to perform the actual message exchanges. This network, here is based here on IPv6, requires IP addresses (locators) to route the messages from the originator node (source host) to the target node (destination host). Therefore, the architecture includes the necessary operation to map context identifiers (sessions) to underlying network locators (IP addresses).

In contrast with the mapping from identities to context identifiers, the mapping operation to obtain network locators does not always include special negotiations. The network locators are built from the context identifiers and the IP prefixes obtained from the DHT or the corresponding nodes. However, when requesting the IP prefixes, which are set as attributes of the virtual identities, the owner may need to accomplish certain policies and thus force an attribute-based negotiation. An event will be triggered so the negotiation can be addressed by specific modules or, in general, any listener attached to the event emitters of the architecture.

Once a node has the IP prefixes and the context identifiers, as shown in Figure 5.5, the source and destination addresses are built as follows. First, the context identifier is split into two pieces of 64 bits of size. Second, each piece is used as a suffix to be joined with the provided IP prefixes and so form the source and destination IP addresses. Thus, the joined

5.4 Mapping Sessions to the Underlying Network

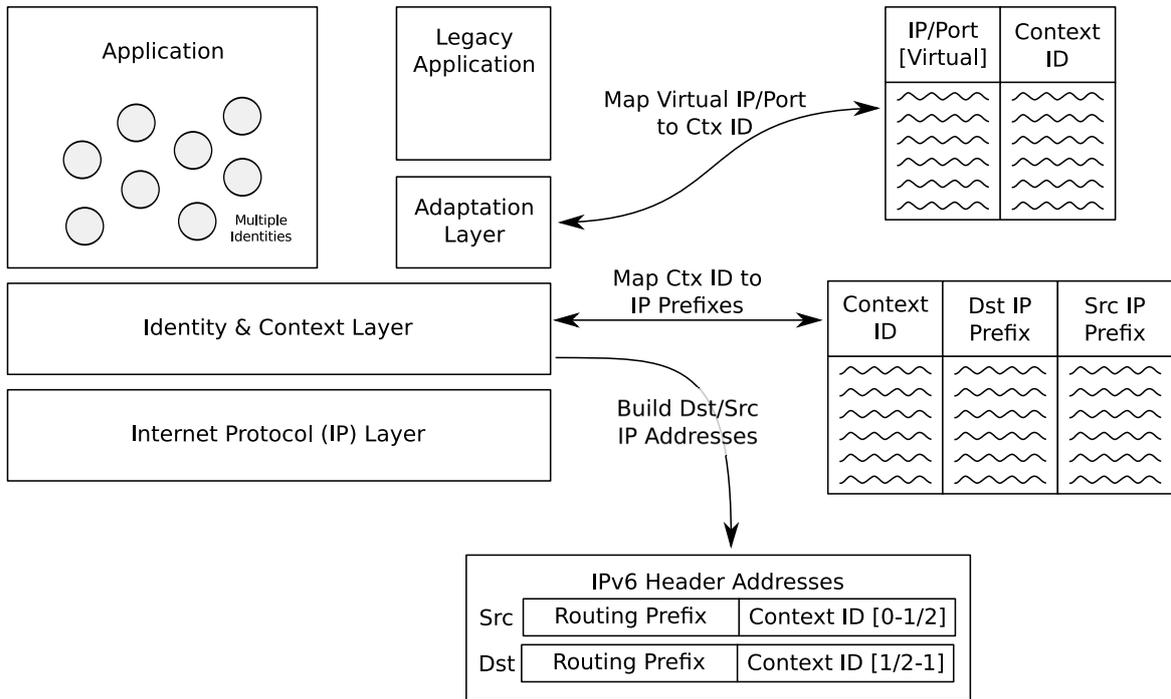


Figure 5.5: Native and adaptation layers with mappings.

IP suffixes hold the context identifier. This method is possible because, in IPv6, the first 64 bits are devoted to *locating* a node in the network (routing) while the last 64 bits are devoted to identifying the specific *interface* of the host. In this way, the messages/packets are routed regardless of the specific suffix used in the IP addresses. Moreover, by using this feature of IPv6, INA permits the session of each message/packet to be identified without adding extra overheads.

To manage the mapping between context identifiers and IP addresses, as shown in Figure 5.5, the *Identity & Context Layer* has a mapping table that stores the relations between the context identifiers and source/destination IP prefixes. This table is updated each time a new context identifier is obtained (because a new session is established) and each time the IP prefixes change (because a node has moved to a different underlying network). Moreover, this table is also used to support a low-level multi-homing feature in both the source and destination sides.

5. Final Architecture: Beyond the Separation of Identifiers and Locators

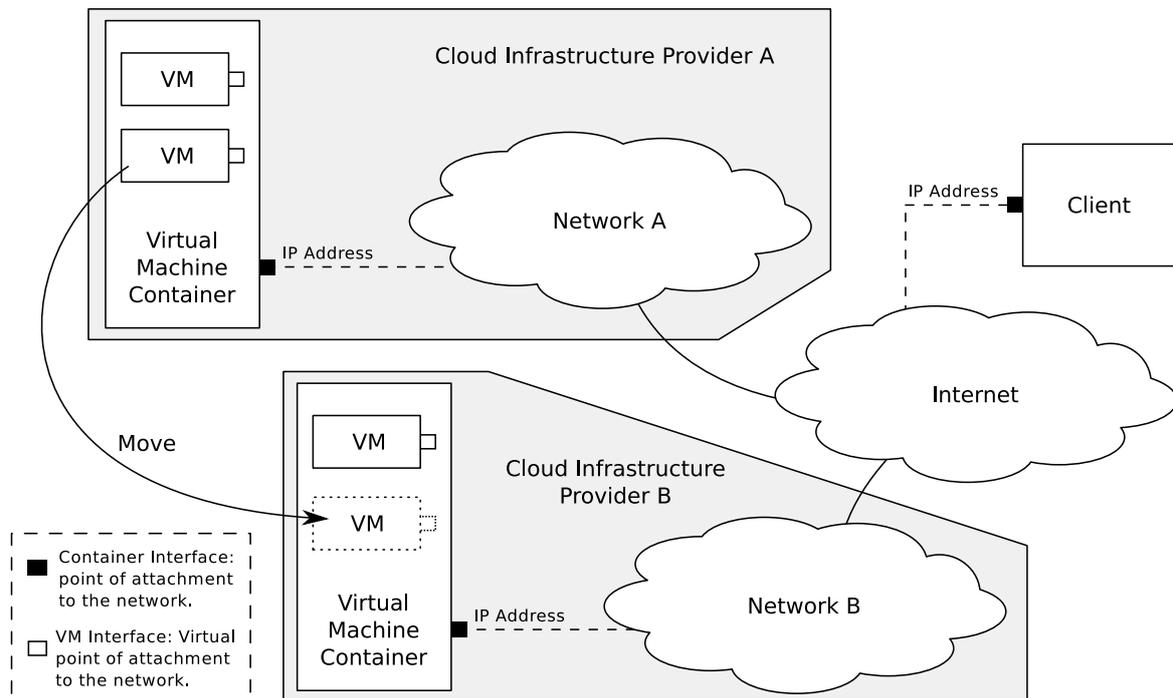


Figure 5.6: Application Scenario: Mobility and granularity in a cloud environment.

5.5 Architecture Analysis

In the previous sections we described the general identity-based architecture approach to decouple the identification and the location of network nodes, the mapping from identities to context identifiers (communication sessions), and the mapping from context identifiers to underlying network locators (IP addresses). In this section we analyze this approach to ascertain its capabilities and discuss how it meets the objectives set out at the beginning of the chapter.

5.5.1 Application Scenario

In the first stage of the analysis we provide an overview of the real utility of INA by discussing how it works in an application scenario. As widely discussed throughout the present document, there are many rising scenarios that current architectures do not cover properly. Among them, we can highlight the network virtualization management, highly interlaced with *Cloud Computing*, a continuously emerging tendency. Here we describe a specific scenario that illustrates how INA improves the behavior of the current IP architecture and the previously discussed identifier/locator separation proposals.

The conditions of this application scenario are shown in Figure 5.6, which illustrates a resource (Virtual Machine, VM) moving between the domains (datacenters) of two different Cloud Infrastructure (IaaS) providers. Each provider has its own network with its own addressing and their networks are connected to the Internet. However, the VM that moves from one provider to the other should have a consistent mechanism for it to be reached through the global network. Using IP addresses at the VM level implies that they should be part of the network of one provider, and this becomes the *home* domain of the VM. Mobile IP could be used, but this has two problems. First, Mobile IP does not provide a real separation between location and identification, as we discussed above. Second, since the initial provider becomes the home domain, the VM that moves to other provider keeps a (maybe contractual) relation with the initial provider, and this is not desirable. We propose to use INA to overcome this limitation. Thus, we also illustrate in the figure one VM container on each domain and the movement operation that moves a VM from the container of the first provider (A) to the container of the second provider (B). Finally, the figure depicts a client connected to the Internet that is accessing to the services of the VM before, during, and after the movement, so the VM needs to be moved without losing reachability or connectivity.

Although the most obvious feature shown in the commented scenario is centered around the mobility and multi-homing support, it is not related just to it. The most important aspect of the scenario is the existence of two different network views. One view is from the VM container, which is attached to the underlying (IP-based) network. The other view is from the VM itself, which is attached to a virtual network that may match or not with the underlying network. This illustrates the aforementioned need to decouple localization and identification. While a network node (VM) may be *located* in a network (the underlying network of the IaaS provider), it may belong to one or more different virtual networks, which are organized and structured following other criteria, determined by the node owner and not by the IaaS provider. Thus, the need to decouple the localization and identification is clearly seen.

However, the first aspect shown by the application scenario discussed here is that it raises an operation that is expected to be frequent in the near future and that is not supported by the current Internet model. Moreover, the direct mobility solutions applied to such model, the Mobile IP [119] and its derivatives [108,109], do not fit with this scenario. They require the continuous collaboration of the *home* domain when a node has moved to a *remote* domain. Also, since they do not really separate location and identification, a moved node still works with the old IP address and the complex adaptation mechanism it involves.

5. Final Architecture: Beyond the Separation of Identifiers and Locators

After discarding the solutions based on Mobile IP, we proceed to analyze how the previously discussed proposals to separate identification and location may fit in the scenario. The first question that rises when applying these technologies is where to apply them. As shown in the scenario, each VM has its own virtual network interface connected to the internal mechanisms of the VM container which, in turn, has a network interface connected to the underlying (real) network. This implies that there exist two different topologies living simultaneously: the topology of the underlying network and the different topologies formed by the virtual network built on its top.

That said, the aforementioned id/loc split solutions may be applied to the VM containers or to the VMs. If one of such solutions is applied to the VM containers, the actual separation and mobility support resides on them, so the nodes hardly benefit from this feature. Also, it does not permit the logical separation of the virtualized networks that connects different VMs and the underlying network that connects the VM containers. Therefore, the id/loc split solutions should be directly applied to the VM.

When applying the id/loc split solutions to the VM themselves we face a new problem. First, in a generalized scenario, the VMs may not have enough independence to implement their own network solution. For instance, if the VM container is based on a lightweight virtualization technology, the network stack is virtualized but shared among the deployed VMs. Also, even if the VM container is based on a heavyweight virtualization technology, the point of attachment to the network is doubled so both the container and the VM are logically attached to the same network. This returns to the previous case, where the id/loc split solution is applied to the VM containers, and we enter an endless circle.

The main problem extracted from this discussion is that the current id/loc split solutions do not provide the *fine granularity* feature that we consider important for the Future Internet, as supported by the research community. This feature would provide to split the solution to deploy the location part in the VM containers and the identification part in the VMs. As discussed in Section 5.2, that is precisely what INA provides and, as seen in the scenario, is a very important feature for the Future Internet.

In any case, there is no direct match between the real elements found in the network (as depicted in Figure 5.6) and the logical mechanisms applied to them. Thus, neither the current Internet model or the proposed id/loc split solutions provide specific definition and separated concepts for the VM containers, the VMs themselves, the virtual networks, and the underlying network. It is very important to separate those elements both conceptually and logically, incorporating specific functionality to each of them.

We do not claim that INA defines the way to cover those conceptual and logical differentiations. The objective of INA is just to provide a mechanism to separate the

concepts of underlying physical or logical equipment and the upper processes, being both elements of the network architecture. A specific architecture design with all conceptual definitions will be treated in future work.

Continuing the application scenario description, as commented above, in this scenario, INA is deployed in both the VMs and the VM containers. The identification part, that forms the identity-based overlay network and build the DHT to perform the identity discovery, is deployed to the VMs. Thus, the mapping between identities and sessions (with their context identifiers) is performed in the upper virtualization layer. However, the location part of the architecture is deployed in the VM containers because they are actually attached to the network. Thus, the mapping between sessions and the underlying network is performed in the lower virtualization layer. This way, when a node moves, just needs to update its context to let other corresponding nodes know its new point of attachment to the underlying network. Also, this removes the unnecessary complexity of maintaining two generic stacks in two unrelated elements, so each element has its own and specialized task in the whole network process.

About the client side, although it may not follow the VM/container relation, in our architecture, it just needs to have the identification mechanisms because the underlying location mapping mechanisms are provided by the infrastructure elements of its network.

Apart from the mobility and the id/loc split, as discussed in Section 5.2, INA provides other features demanded by the Future Internet and not found in current proposals. That is the case of the integrated discovery, flexible naming, and integrated security. As deeply discussed in the following subsection, current id/loc split approaches do not cover these features. Thus, in the discussed scenario, a client must use an external service to find the *description* of the service (VM) it wants to contact. With our approach, a client just needs to form a partial identity with the knowledge it has about the VM and the identification mechanisms (found in the identity-based overlay, which is formed by all nodes, including VMs and clients) will perform the mapping to communication sessions. Also, this provides integrated security because a client may perform an authorization and authorization process during the session negotiation. Finally, as also discussed in Section 5.2, a key point of INA is that no node is reachable by default. It must first map an identity to a session (negotiation) to obtain a context identifier that the destination accepts and, during this process, any party may apply identity-based access control mechanisms, which are very strong in comparison with others based on IP addresses or plain identifiers.

5. Final Architecture: Beyond the Separation of Identifiers and Locators

5.5.2 Comparison with Other Approaches

As stated above, to demonstrate that INA provides a clear advance over the current proposals, we compare its capabilities with those provided by previous work. We choose the most important features which are required for the Future Internet (FI) and missing in the current Internet model. These are detailed below.

First, we analyzed the identification type of the different architectures. As their main purpose is to achieve the separation of identification and location, the mechanism used to identify the nodes in the network is a key aspect of the architecture. Also, the identification type provides a notion of the orientation and organization of the architecture from its identification side.

Second, we analyzed the granularity level. As mentioned throughout this chapter, the granularity is a key aspect for the FI, although most architectures do not take it into account. This feature evaluates the minimum element that forms a network node as three values: low, medium, and high.

Third, we analyzed the mobility support and mechanism used to provide it. It is widely known that mobility is a key aspect of the FI, so we include it in our evaluation. When it is included in the architecture without requiring special and external elements, we have evaluated it as *integrated*.

Fourth, since it is another key feature for the FI, we also analyzed the multi-homing support. In this case we just evaluated whether an architecture provides support for it or not, so the values taken are *yes* and *no*.

Fifth, we analyzed whether the architecture includes integrated discovery or not. The integrated discovery is a key aspect for the FI and one of the most important challenges present today. Again, we have evaluated if the architecture includes integrated discovery support or not, so it takes the *yes* and *no* values (with some exceptions).

Sixth, although it is related to the integrated discovery, we also analyzed the flexible naming support of the architecture. This feature could be evaluated as values of different levels, like the granularity level, but we preferred to use boolean values to determine if the architecture includes support for fully flexible naming or not, so the resulting values are *yes* or *no*.

Seventh and lastly, we finally analyzed whether the architectures include integrated security support or not, so its results are Boolean values (*yes* or *no*), but we allowed the inclusion of the *extensions* value when an extended version of the architecture provides some mechanisms to provide integrated security to the architecture. This does not consider

Table 5.1: Comparison of solutions for decoupling identification and location.

Approach	Identification Type	Granularity Level	Mobility	Multi-homing	Integrated Discovery	Flexible Naming	Integrated Security
HIP	Plain identifier	Low	Rendezvous	No	No	No	No
LISP	IP address	Low	Separated	No	No	No	No
MILSA	URI-like	Low	Integrated	Yes	No	No	No
EMLISA	URI-like	Low	Integrated	Yes	No	No	Yes
HIMALIS	URI-like	Medium	Integrated	Yes	No	No	Extensions
MOFI	Plain identifier	Medium	Integrated	Yes	No	No	Extensions
ILNP	Plain identifier	Low	Integrated	Yes	Only DNS	No	No
INA	Identity	High	Integrated	Yes	Yes	Yes	Yes

the totally unrelated but compatible mechanisms that can be provided as a complement or higher layer security.

That said, in Table 5.1 we show the results of the feature analysis of the different architectures. In the first feature compared, we can appreciate that most proposals use plain or URI-like identifiers. The benefit of URI-like identifiers is that they can be organized to hierarchies, but they do not provide any other benefit to the node identification. In contrast, our proposal uses identities and permits a stronger identification together with an improved organization (including hierarchical organization).

For the second feature, the granularity level, we can see that very few architectures (HIMALIS and MOFI) have progressed from the host-centric approach. Thus, although they still share many similarities with the host-centric approaches, the medium granularity level offered by those architectures is higher than the others, so it denotes a good advance. However, as we stated at the beginning of the chapter, granularity is a very important requirement for the FI, so it is one of our objectives and thus INA provides a much higher granularity level in comparison to the other architectures.

On observing the comparison results of the third feature, the mobility support, we can see how this is a mature requirement. Both HIP and LISP have included mobility support during their evolution, the former by a rendezvous mechanism while the latter by a separated mechanism. Moreover, all other approaches (including INA) have integrated

5. Final Architecture: Beyond the Separation of Identifiers and Locators

the mobility support to the core of the architecture. The same happens to the multi-homing support, even though HIP and LISP have not addressed a proper support for this, it is also a very mature requirement and all other approaches (including INA) already provide support.

The three remaining features, the integrated discovery support, the flexible naming, and the integrated security, are not so mature as the previous ones, so most approaches do not provide them. The last feature, the security support, denotes a widely known feature but many approaches, like HIP and LISP, do not address it at first. However, it is being considered by many architectures, as is the case of MILSA and EMILSA. The latter is defined to cover the security requirements for FI missing in the former. Moreover, both MOFI and HIMALIS are defined without integrated security in the core architecture, but they are evolving to include it through new profiles and extensions. Even though the ILNP approach is relatively newer, it does not cover the integrated security requirement and leaves security to higher layers. This is a step back as denoted by the position of the other architectures regarding integrated security. This is not the case of our architecture, which meets this challenge and includes integrated security through the identity of the communication participants.

Finally, we pay special attention to the integrated discovery and flexible naming. These two features are related and this is reflected in the results of the architecture comparison. It is also a relatively new challenge, and only ILNP tries to provide some mechanisms to achieve integrated discovery. The flexible naming requirement is even, mainly promoted by the rise of the Internet of Things (IoT), so there is no previous architecture that provides this feature. Again, as it is part of the objectives of INA, we use identities to provide both integrated discovery and flexible naming, as discussed in Section 5.2.

5.5.3 Performance Analysis

The final architecture discussed here, being the result of integrating the DTEi, IBNP, and ODIN, builds an identity-based overlay network to let fine grained network nodes (processes) reach each other via their identity instead of an address or plain identifier. However, the actual communication is not affected by this capabilities, because once the identities are mapped to context identifiers and the context identifiers (a session is started) are mapped to underlying network locators (IP addresses), the message/packet exchanges between network nodes is performed directly in the underlying network without adding extra steps or overhead. Therefore, the performance of the architecture is subject to the performance of the two mapping operations.

We should recall the structure of the mapping operations. While the first mapping, from identity to session/context identifier, includes the node discovery and thus requires many interactions, the second mapping, from context identifier to network locator, only requires a simple operation composed of a look-up on a local table (the CTX-to-IP mapping table) and the construction of IP addresses. This way, the small overload introduced by the second mapping operation is negligible, so we center our performance analysis to the first mapping operation.

As commented above and detailed in Section 5.3, the mapping operation that obtains the context identifiers from the identities includes two different functionalities which we analyze separately. The first functionality is the network node discovery, which gets a virtual identity (VID) from a partial identity (PID). This functionality is found in the two variations discussed in Section 5.3. The second functionality is the actual session start, comprising the associated negotiation, and the exchange of context identifiers.

The node discovery, also detailed in Section 5.3, is performed by multiple queries to a Distributed Hash Table (DHT) built with the nodes pertaining to the identity-based overlay network. This operation is not cheap because it involves many messages comprising many network hops and additional calculations. In summary, a numerical upper limit of the cost of this operation can be expressed in the following equation:

$$C = 2 [(M + 1) \log_2(N) T_{\text{INET}}] \quad (5.1)$$

M is the number of requested identifiers (number of attributes in the partial identity), N is the number of nodes forming the overlay network and the DHT, and T_{INET} is the cost (in milliseconds) of sending a message through the Internet. The logarithmic function is applied because it is the number of hops needed to reach a node in the DHT from any other node, as discussed in [35, 41]. Moreover, using the experimentation results presented in [104, 122], this number of hops may be substantially reduced because of many optimizations. Thus, the logarithm is multiplied by three different factors (0.55, 0.45, 0.35) to obtain the actual behavior. This is a good aspect because, even though the global order of the equation is $O(M)$, knowing that the number of attributes in the identities will be much smaller than the number of nodes in the overlay network, the real order is $O(\log_2(N))$; so reducing the impact of the logarithm will substantially improve the general cost.

In Figure 5.7 we show the evolution of the cost, using the equation commented on above, with and without the performance improvements, for the variation of the number

5. Final Architecture: Beyond the Separation of Identifiers and Locators

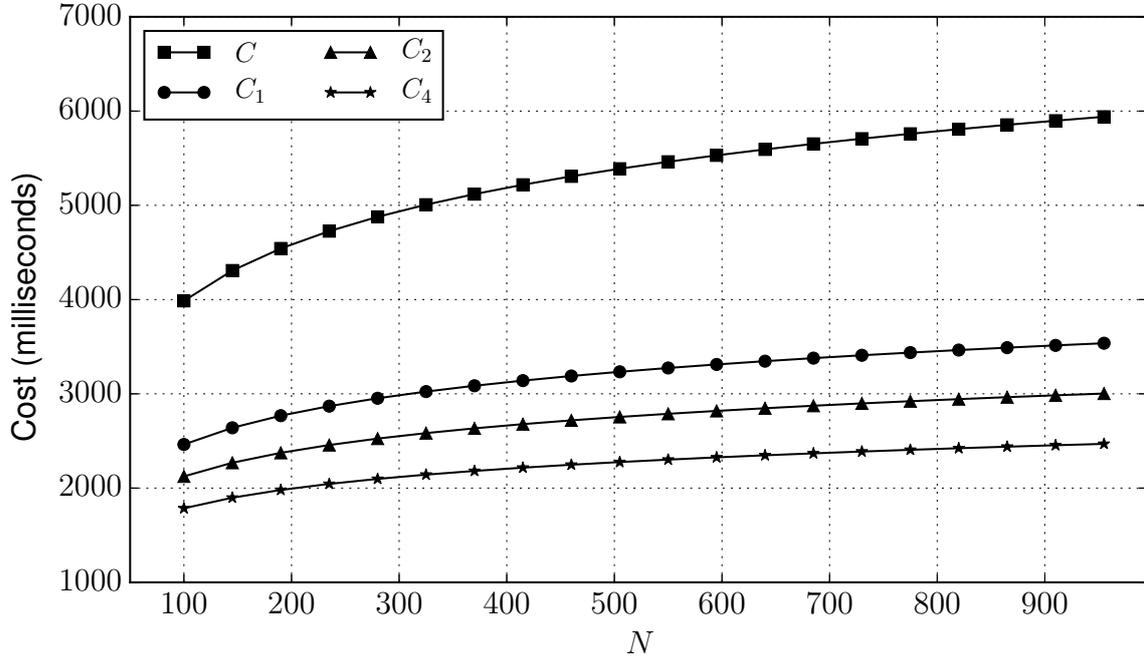


Figure 5.7: Performance cost results for the variation of the number of nodes in the overlay network forming the DHT.

of nodes in the overlay network forming the DHT and with a fixed number of attributes ($M = 5$) and a fixed Internet transfer cost ($T_{\text{INET}} = 50ms$).

Regarding the functionality of the second part, the actual session initiation and negotiation. This functionality also underlines the difference between the two variations commented on above, and adds a few more message exchanges. Although its impact on the performance of the whole architecture may be considerable, it comprises just a few message exchanges so, in the terms we are analyzing here, its impact will be negligible compared to the impact of the identity discovery. Thus, we leave its specific analysis for future work and also compare it with other similar methods to initiate sessions (identifier exchanges).

5.5.4 Experimentation Results

In the previous subsection we discussed the theoretical results for the performance of INA. Now we show the experimental results obtained with a proof-of-concept implementation of the mechanisms proposed with the architecture, as discussed in Sections 5.2, 5.3, and 5.4.

The evaluation environment (testbed) we used is similar to the scenario discussed at the beginning of this section. We deployed our proof-of-concept implementation in various virtual machines (VMs) managed by the Xen hypervisor, so the computing performance is not degraded due to the reserved resources and the paravirtualization model it follows.

The proof-of-concept implementation has the necessary mechanisms to map partial identities to context identifiers and context identifiers to network addresses. The mapping mechanism used to obtain the context identifiers includes the identity discovery and the session initiation. As discussed in Section 5.2, the discovery mechanism included in INA is based on a DHT. To get a good approach to the performance of INA when deployed in real scenarios we decided to use a real world DHT. Thus, in our experiments, we used *Mainline DHT*, the Kademlia-based DHT [41] used by the BitTorrent network. It has millions of concurrently connected nodes (*peers*) and has been deeply studied [123]. It is worth noting that the *Mainline DHT* works on top of IP. It has support for IPv6 but it is not widely deployed because most of its nodes only support IPv4. Thus, the discovery part of our implementation works on top of IPv4.

Since the discovery mechanism is the most sensitive part of the architecture, because it involves the interaction with many nodes to resolve an identity, we paid special attention to its implementation and tried different ways to implement its functionality. As described in Section 5.2 and shown in Figure 5.2, the discovery process asks many nodes of the DHT for the triples (identifier, attribute, value) whose value matches the value of the partial identity. When using the *Mainline DHT*, each node of the DHT that receives the request sends a response with the address/port pairs of the nodes to send the actual identifier request. Thus, there are two chain operations to be performed. After trying different approaches we determined that the best method to use is an event-based method, so each response from the DHT is processed as soon as it is received. The same is applied to the responses coming from the nodes contacted after receiving the DHT response. Finally, as soon as the module receives valid identifiers for all attributes, it discards any further response and returns the requested identifier to the main module.

Another aspect of the discovery is the method used for the DHT node lookup and routing. The work presented in [124] proposes different lookup and routing methods to improve the performance of a Kademlia-based DHT. We decided to use the DHT implementation (source code) provided in that work to implement our discovery mechanism. Moreover, based on the results presented there, we decided to use the NICE + Low RTT, which tries to reduce the lookup latency by continuously refreshing the overlay routing table to remove unreachable *peers* and also replace any *peer* by another with less

5. Final Architecture: Beyond the Separation of Identifiers and Locators

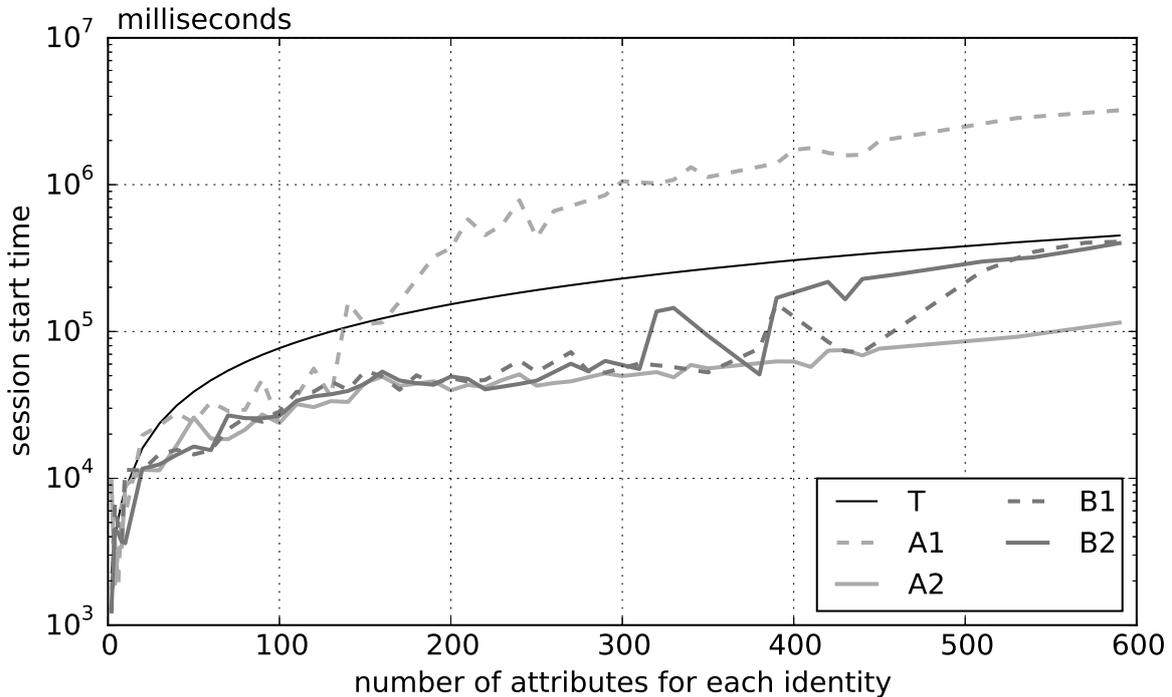


Figure 5.8: Session start time for different number of attributes.

round trip time (RTT). This gives us a good starting point for the performance of our proof-of-concept implementation.

The approach commented on above makes the resolutions as quick as possible, removing any dead time and reducing the latency as much as possible. However, not all nodes will respond to the identifier request because they may fail or because they are using the DHT for other purposes like serving BitTorrent content. Thus, we also considered that some resolutions should be retried.

After implementing the functionality defined by INA, we designed an experiment that consists of announcing many identities with many attributes and starting a session with one of them. We measured the elapsed time of each operation of the experiment, so we can extract an experimental notion of the performance of the architecture. For the identity announcements we have two variations: launching the session start operation (with the identity discovery) without waiting for the confirmation of the identity announcements and waiting for the confirmations before launching the session start operation. We decided to obtain measurements for both variations and analyze the difference in performances. We only varied the number of attributes because varying the number of identities does not affect the performance of the discovery and session start times.

From the results we got Figure 5.8, which shows the session start time for different number of attributes. T represents the theoretical results obtained by applying the equation discussed above. $A1$ and $A2$ are the results for the two consecutive session starts performed after announcing the identities in the DHT but without waiting for confirmation. $B1$ and $B2$ are the results for the two consecutive session starts performed after announcing the identities in the DHT and wait for its confirmation. In it we can see that session start time increases when the number of attributes for each identity increases. This is expected because each *new* attribute in a partial identity supposes more than one exchange in the DHT and the final node that gives the triple. However, the start session time is kept under the theoretical estimation (T) obtained from Equation 5.1 for all results but for $A1$, which represents the results from the first session operation without waiting for identity announcement confirmation. This time is hugely increased because the DHT operations need to wait until the hashes of the identities are stored in the corresponding nodes before giving the responses. This is clearly indicated by $A2$, which is the most stable result because it represents the time elapsed during the session starts launched after the confirmation of all identities. Finally, $B1$ and $B2$ results are similar because they were launched after the announcement confirmation of all identities. These are higher than $A2$ because some peers, which were nearer to the peer that stores the information for the requested identity, have been lost from the routing table after waiting (so long) until all identity announcements are confirmed.

As mentioned above, the operations performed after the announcements of all identities ($B1$ and $B2$) had been confirmed took much less time to complete than the first of the other operations ($A1$). However, as seen in Figure 5.9, which illustrates the time elapsed in the announcements together with the time elapsed in the session starts, the total time of the second approach (B), whose implementation waits for confirmation of all identity announcements, is higher than the first (A), whose implementation does not wait for the identity announcements to be confirmed. This is because the announcement confirmations take a huge time to be received from the DHT. This could be improved but the announcement of such huge number of identity information is a very rare case and, in case it is necessary, should be addressed by other modules or higher layer applications.

Since the DHT is a very dynamic structure built on a very dynamic network — the nodes (peers) join and leave very frequently — some discovery operations fail and need to be retried. Figure 5.10 depicts the fraction of the total discoveries that needed a retry. $A1$ and $A2$ correspond to the results for the two consecutive discoveries (session starts) performed without waiting the confirmation of the identity announcements. $B1$ and $B2$ correspond to the results for the two consecutive discoveries performed after waiting the

5. Final Architecture: Beyond the Separation of Identifiers and Locators

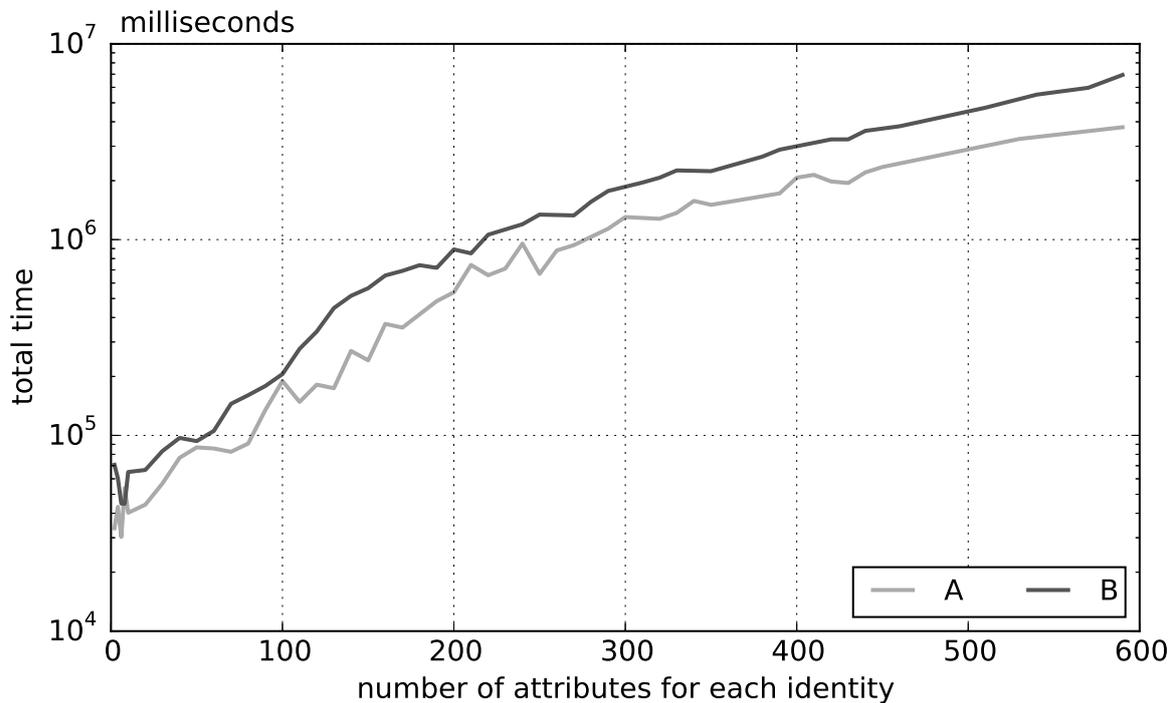


Figure 5.9: Total time elapsed in the announcement and the two consecutive session starts.

confirmation of the identity announcements. These results show that the fraction of the total discovery operations that should be retried is between 30 and 35 percent for the more normal and stable cases (*A2*, *B1*, and *B2*). However, for the more aggressive case (*A1*), it is reduced to less than 15 percent. This is because of the small time elapsed between the announcements and discovery requests. When this time is short, very few nodes have left after announcing the identities, so the DHT is almost the same. This indicates that the time from an announcement and its posterior request should be reduced as much as possible, but without affecting the time needed to resolve the request.

As discussed in the last paragraphs, the peer *join* and *leave* operations (from the DHT) affects the overall performance of the discovery mechanism and thus the session start operation. However, this is also related to the number of peers present in the routing table of the initiator node. Figure 5.11 shows the relation between the number of peers in the routing table and the discovery time. *A1* shows the relations for the discoveries performed without waiting the confirmation of the identity announcements. *Failed* represents the relations for the discoveries that were not complete. *Others* represents the relations for the other cases. As we can see in the figure, the relation between the number of peers present in the routing table and the response times of every operation (independently of

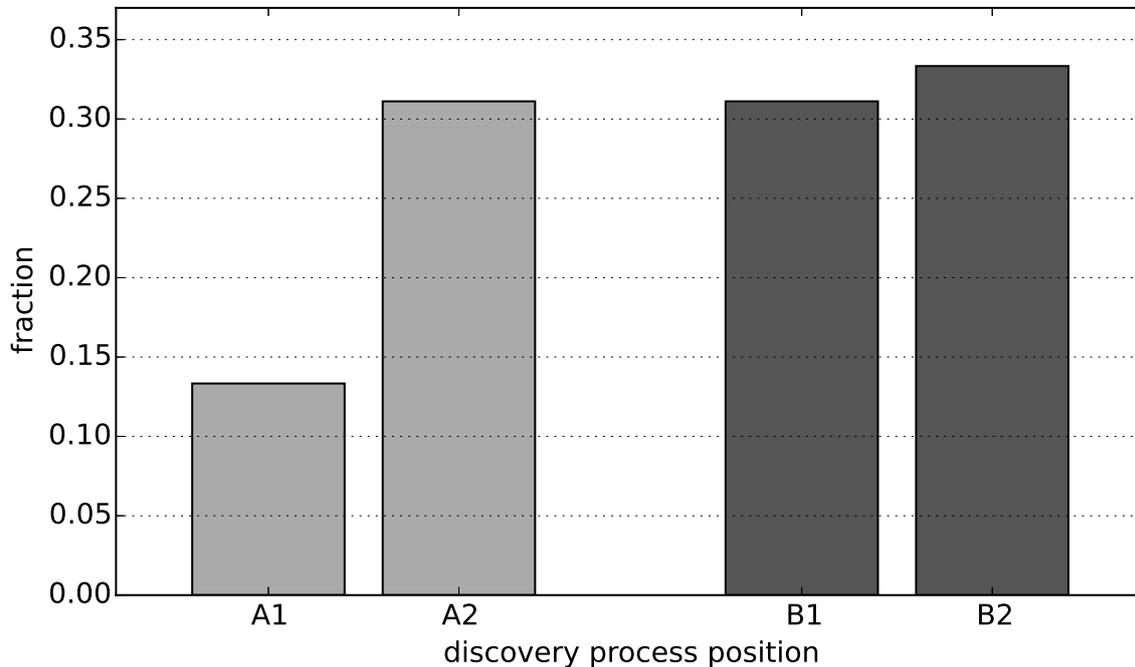


Figure 5.10: Fraction of the discoveries that needed a retry.

the number of attributes to be resolved), the general response time of the discoveries is reduced to less than 10^5 ms in most cases, when the number of peers present in the routing table is bigger than 100. When observing just the *Others* results, which do not include the failed requests or the first discovery requests that do not wait for identity announcements, there was just one operation that lasted more than that time for more than 100 peers in the routing table. Moreover, the results distribution is thinner from that value onwards, so the response times are more stable. Therefore, we can affirm that the response time does not directly depend on the number of requested attributes but on the number of peers found in the routing table.

5.5.5 Evaluation of the Instantiation on Top of CCN

Apart from the analysis and experimentation centered on the discovery mechanism of INA we also evaluated its communication stage when instantiated on top of the Content Centric Networking (CCN) architecture, as we presented in [105], while running in the specific application scenario discussed above. Here we discuss the results obtained from the measurements obtained from the experiment.

5. Final Architecture: Beyond the Separation of Identifiers and Locators

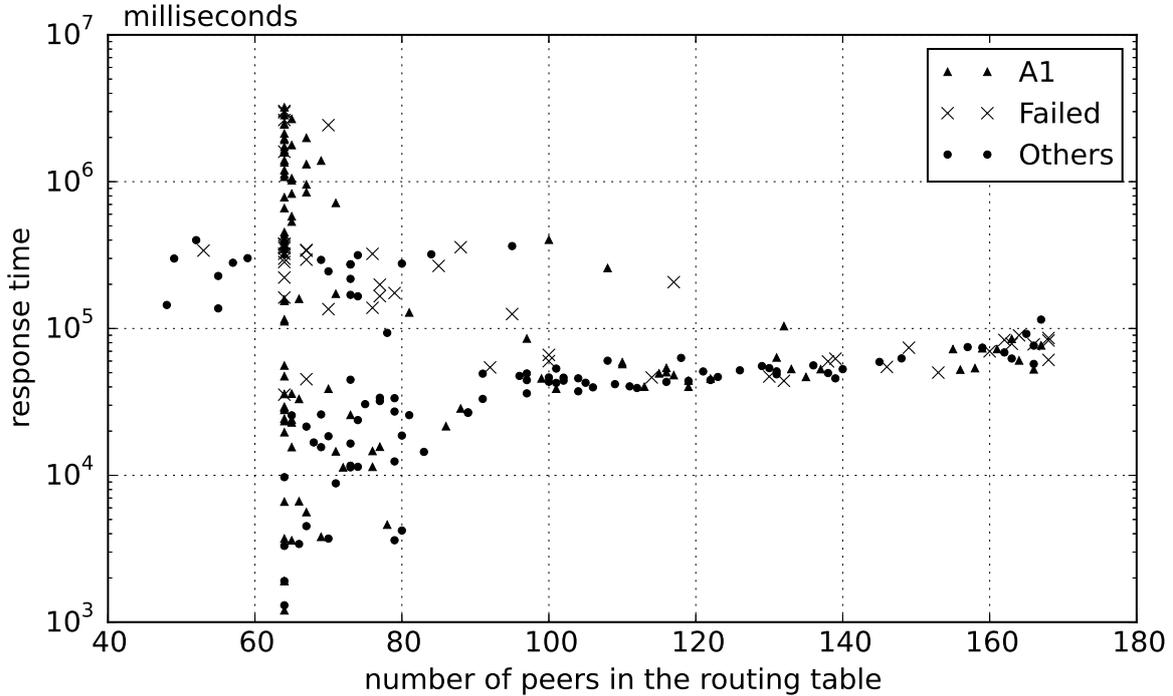


Figure 5.11: Relation between the number of peers in the routing table and the discovery time.

The evaluation has been performed by designing and running an experiment on top of a testbed composed of 6 networks: a virtualized interconnection network which acts as the global transit network defined in the architecture, and 5 edge networks, two of which are real networks and the remaining three are virtual networks. All nodes are Virtual Machines (VMs) running in top of the Xen virtualization technology (paravirtualization), a widespread solution in high performance virtualization environments, like in many Cloud Computing infrastructures. All the equipment, virtual and physical, runs the Linux operating system. The virtualized networks are built using Linux kernel bridges in the host machines and using VTun [125] to build ethernet bridges through TCP/IP connections between separated host machines.

As shown in Figure 5.12, the topology of the experimentation environment has all the elements defined in our architecture (nodes of the DTEi and GWs) together with the client nodes (MN and CN). The edge networks correspond to different network and identity domains. Thus, the MN and CN respectively belong to Domain 1 and Domain 3. As expected, regardless of whether they are connected to the real or virtual networks,

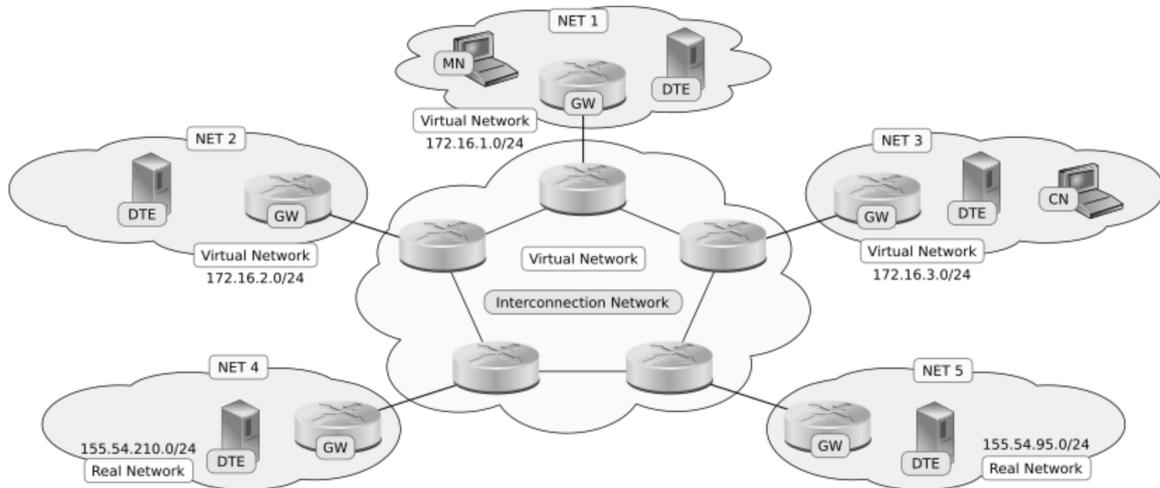


Figure 5.12: Experimentation environment.

the client nodes are configured to route messages through their corresponding GW which is configured to route them through the interconnection network.

In order to see the scalability and overall performance of the solution we first run an experiment to get the average time spent in transmit and receive a single message but at different message rates. We set the MN and CN to exchange messages at rates from 1 msg/s to 16384 msg/s. It is run 30 times for each rate to get the average and the whole solution is run 10 times to get the standard deviation and standard error of the measurements. We also measure the loss rate for each message rate. Finally, we run a IP-based client to get comparable results. After that, we run the experiment again but set the MN to change from its home network to other network after some messages to obtain the loss rate when the node is moving. Finally, we choose a specific message rate and make the MN to sequentially move to all the domains, each 5 seconds in counterclockwise. Thus, we can demonstrate the behavior with many handover events.

To get a notion of the penalties introduced by INA over CCN, we compared its performance to the performance of raw CCN. Thus, in the experiment, we measure both the time spent on each message exchange and the total time spent on the whole test. Then, using the former measures, we calculate the average time spent for each message exchange and with the latter measures we calculate the time spent by each message in terms of the whole application.

First, Figure 5.13 shows the evolution of overhead when increasing the number of messages. The overhead is the principal result of all the experiments with respect to INA. It lets us see the time increased by using our identity based architecture and protocol

5. Final Architecture: Beyond the Separation of Identifiers and Locators

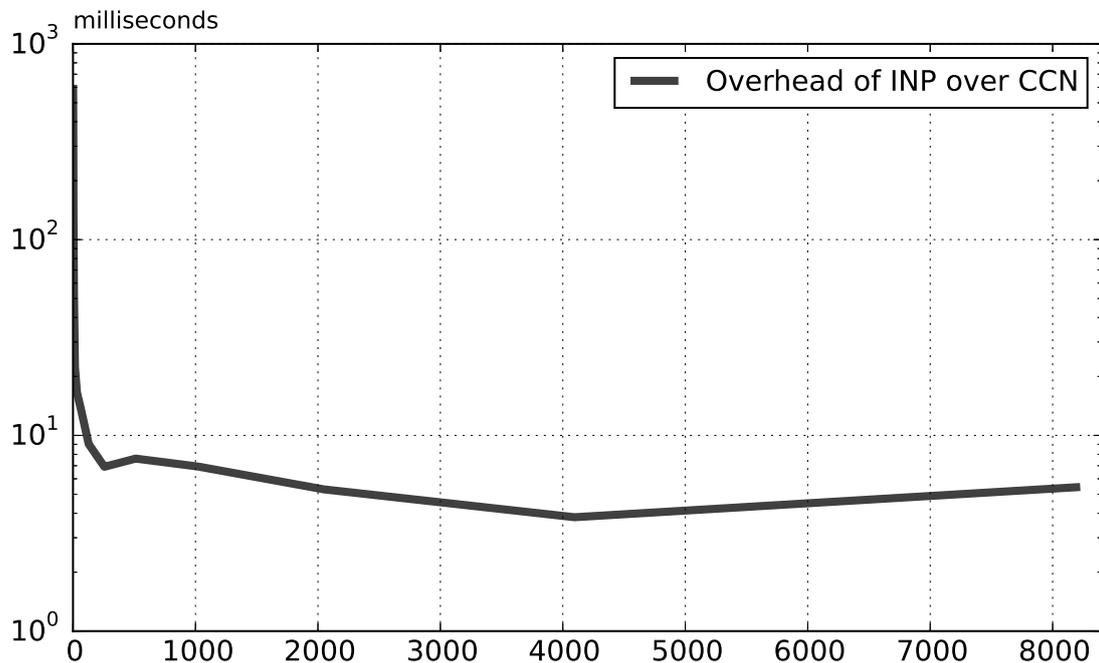


Figure 5.13: Comparison of the overhead evolution when increasing the message exchanges for INA running on top of CCN and raw CCN.

on top of the other lower layer network architectures. We can see that the overhead is bigger when there is a small number of exchanges but it decreases quickly as the number of exchanges increases, and stabilizes below 10 ms.

Figure 5.14 shows the time spent on the different communication steps. It shows the following communication steps: Authentication (*AuthN*), identity search (*Query*), session establishment (*Session*), and exchanging (request + response) 10 messages (*Data x 10*). From this result we get that the authentication, identity search, and session establishment take more time because the first message exchanges in CCN need to establish the *interest* and the corresponding data structures. Still, it takes less than 1800 ms to negotiate a session, so it benefits long sessions but harms short sessions or individual out-of-session messages. Although it is not so very different from the behavior of the current Internet, which also benefits long sessions, in the future we should investigate how to treat short sessions and individual messages in a different way to raise their performance.

Figure 5.15 shows the overhead of each communication step for an increasing number of messages sent in the session for different communication situations: base message exchanges with session establishment (*Session*); base message exchanges with session establishment

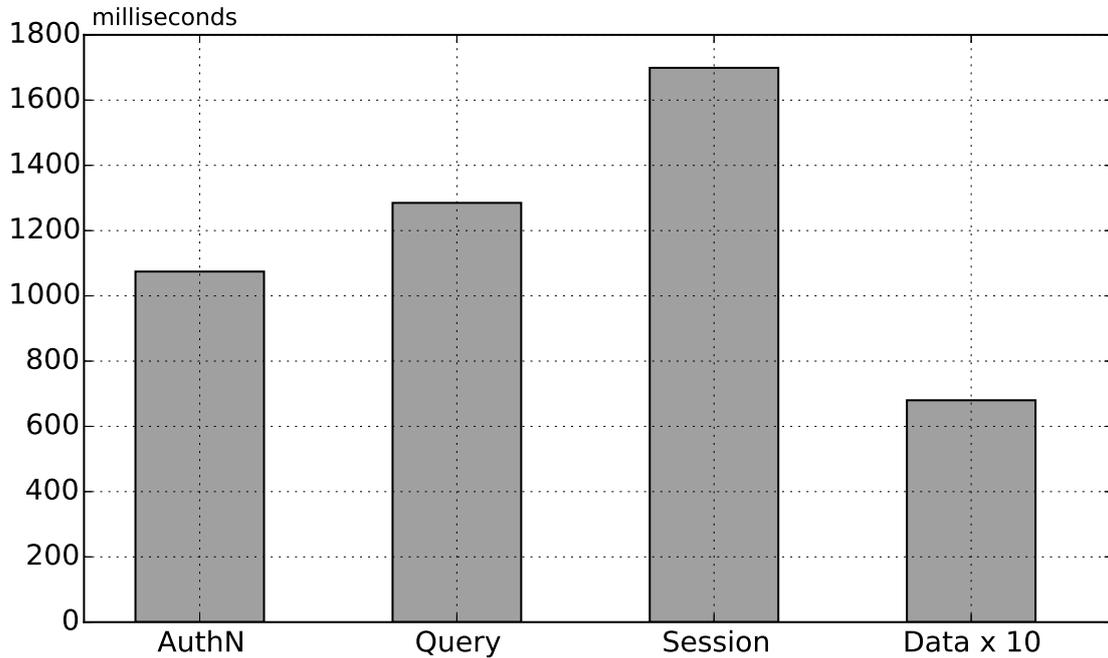


Figure 5.14: Time spent on the two-way test on each step.

and identity query (*Query*); and base message exchanges with session establishment, identity query, and authentication (*AuthN*). This demonstrates that for short sessions the architecture adds a very noticeable overhead but for long sessions it is almost negligible. For instance, for sessions with more than 40 message exchanges, the overhead is less than 100 ms. Thus, as the overhead quickly decreases, we confirm the behavior introduced above. We can also see that, as the three plots are parallel, the new communication steps do not imply a big increase in the overhead compared with each other; they only add overhead to the base case (raw message exchanges).

From the results described above, and especially the overhead comparison, we extract that INA takes only a few milliseconds more than raw CCN, which responds to the necessary extra time to process the INA messages and the specific operations of the adaptation layer, as described in Section 5.2. The worst case is found in the exchange time of the request/response messages because of two extra steps (message parsing and encoding) but, as shown by the overhead evolution chart, its overhead over raw CCN is almost negligible. Finally, the extra *total time* observed in the experiment is due to the authentication, identity description exchanges, and identifier validation. Although this

5. Final Architecture: Beyond the Separation of Identifiers and Locators

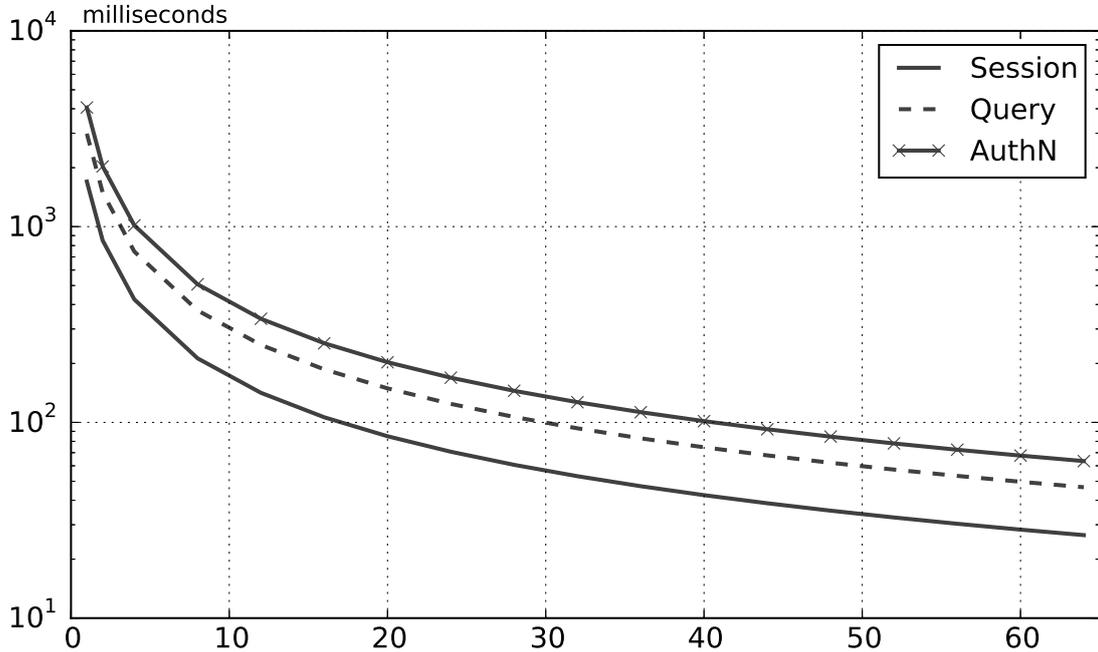


Figure 5.15: Overhead evolution for stacked steps.

extra time is negligible for communications with several exchanges, it must be minimized for those with few exchanges.

5.5.6 Evaluation of the Integration with HIMALIS

Similar to the above, where we show the results of INA instantiated on top of CCN, we also instantiated INA as a complement of HIMALIS and integrated it to run some communication experiments to see the behavior of the architecture in this integration. Thus, with the integration of INA with HIMALIS (INA-HIM), as also discussed in [120], we ran some experiments for the application scenario shown in Figure 5.6 and compared the results obtained with the results discussed above while running INA on top of CCN. Thus, we first show the scalability results and performance comparison between INA-HIM and IP, as well as with raw CCN and with INA on top of CCN. Then we compare the percentage loss for IP, INA-HIM, and INA-HIM when responding to a mobility event.

Figure 5.16 shows the aforementioned results. It shows the average time evolution in milliseconds (top) and message loss percentage (bottom). The former compares the behavior of INA integrated with HIMALIS (INA-HIM) with IP and with INA working

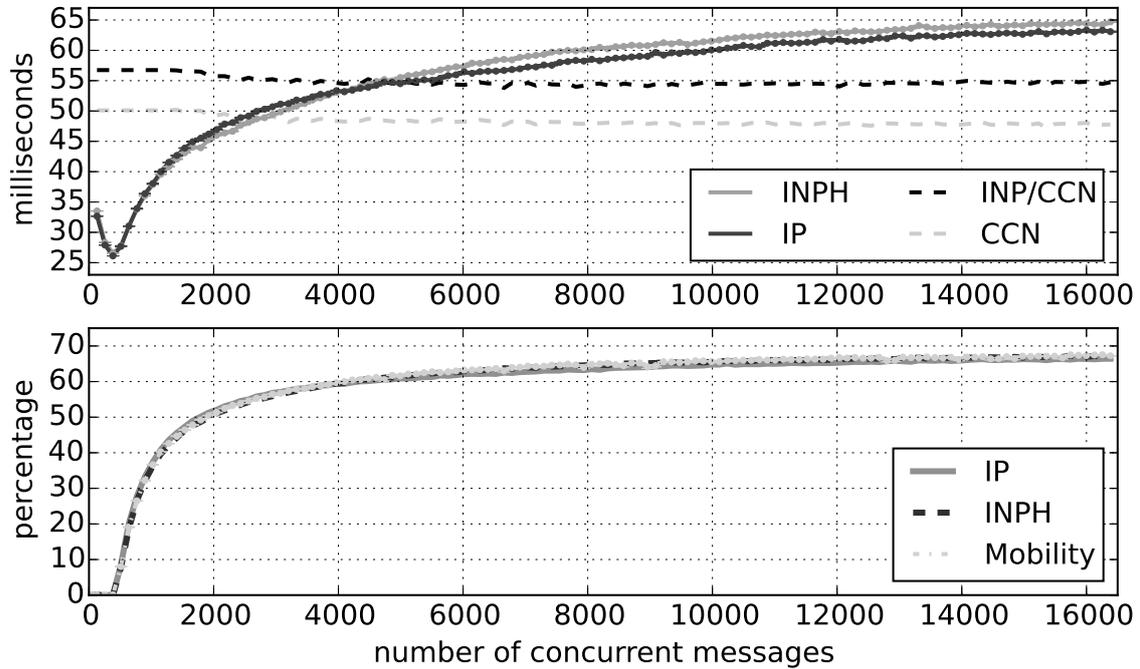


Figure 5.16: Average time evolution in milliseconds (top) and message loss percentage (bottom).

on top of CCN. The latter compares the message loss percentage of IP, INA-HIM, and INA-HIM when responding to a mobility event. As the plots are overlapped, Figure 5.18 and Figure 5.19 show the differences of our architecture with the base case (IP). As we can see, both plots show that INA-HIM is close to the IP approach, either with or without the mobility event.

For the average time per message exchange, we can see that it stays under 35 milliseconds (ms) for rates under 1000 messages per second, but does not surpass 65 ms for huge rates. This demonstrates the good scalability of our solution. About the CCN results, we show that our approach adds a few milliseconds to the raw CCN but it is constant for increasing data rates. This also demonstrates that our architecture scales well. The fact that CCN results are under IP and INA-HIM is due to the CCN nature. CCN does not use any routing machinery or gateway, it broadcasts messages to all interested nodes, so it can deliver more messages per second after it has established the path. For low data rates, CCN is worse because it is implemented in Java and the path establishment from the sender and receiver takes more time than the other solutions.

5. Final Architecture: Beyond the Separation of Identifiers and Locators

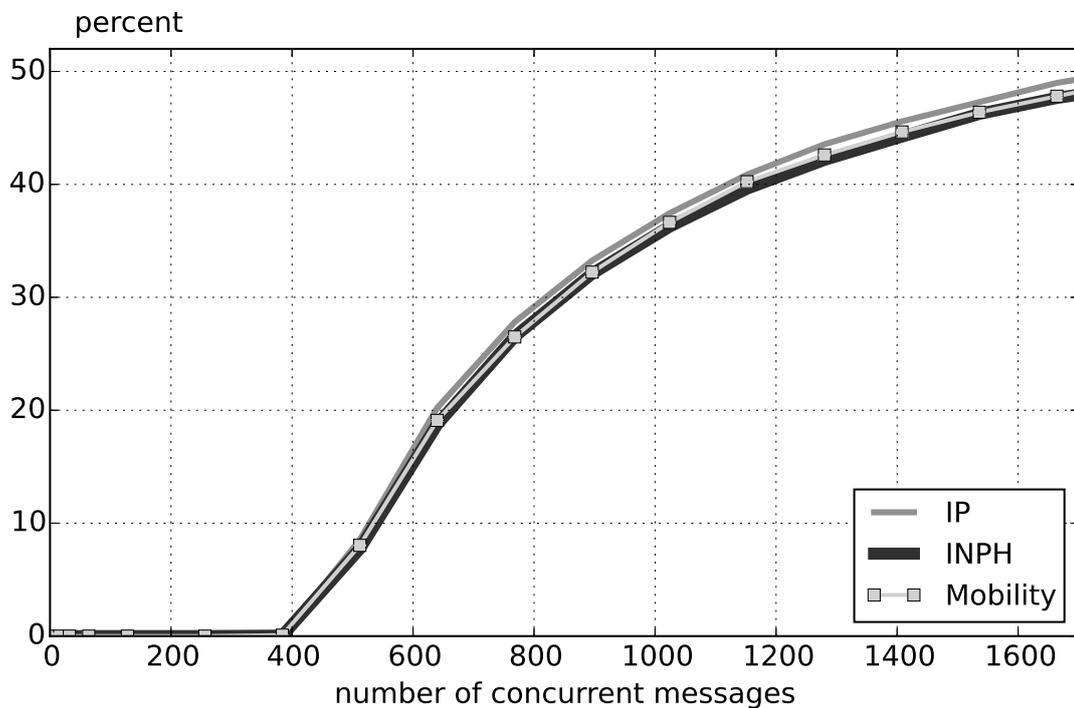


Figure 5.17: Excerpt of the evolution of message loss for the different tests that show the turning point where the message loss leaves the 0%, that is around 400 concurrent messages.

For the percentage loss, in which the three results are together, we see that they do not exceed 70% of message loss even for huge data rates. For rates under 2000 msgs/s, the loss stays under 50% and so on. Figure 5.17 shows the results for the lower rates amplified, to see at which rate the solutions start to lose messages. We can see that no solution loses messages under 400 msgs/s but around 500 msgs/s the loss percentage is still around 10%. We will choose this rate to perform our final experiment.

As the results for INA-HIM and IP are very near in the figure commented above and since we want now see the real separation between them, we now show the differences in different figures and comment them.

Figure 5.18 shows the separation of the average time per message of INA-HIM and IP. It also shows the standard error of the measurements, illustrated as error-bars wrapping the plot. We should note that the plot is saw-shaped because of the enlargement, which is demonstrated because the standard error and, hence, the standard deviation is very low, what gives us high level of confidence in the results. In the plot we can see that INA-HIM does not differ from IP in more than 2.5 milliseconds and that for some data rates it

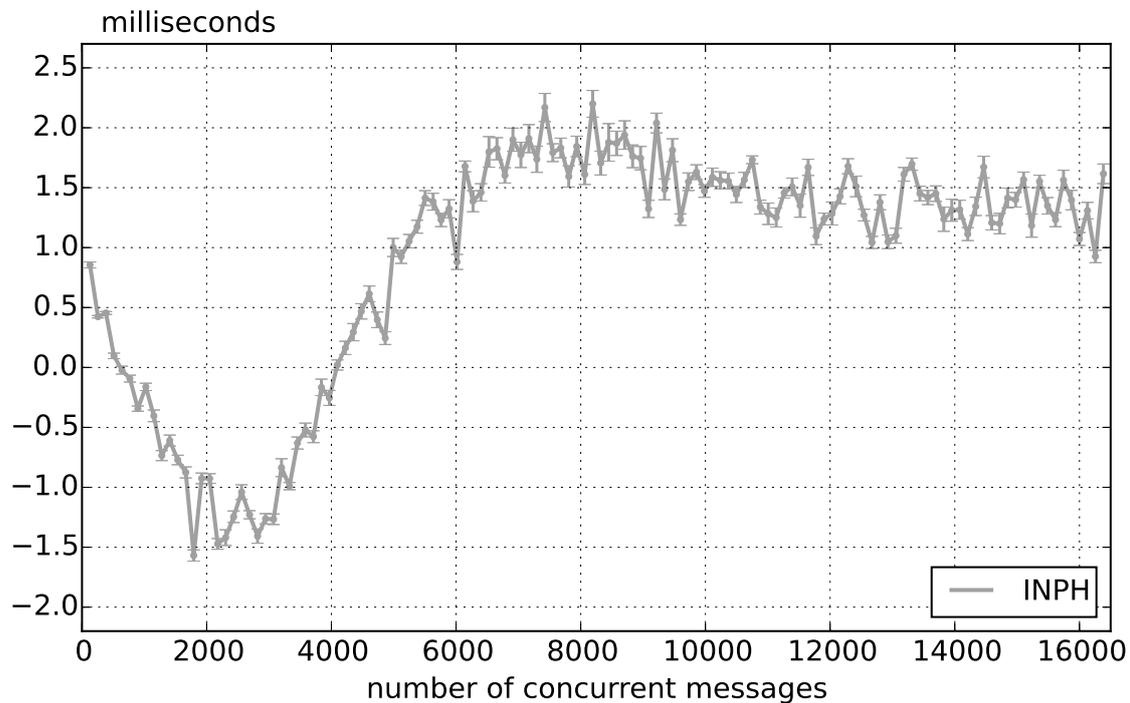


Figure 5.18: Evolution of the difference of the average exchange time of INA-HIM and IP.

spends up to 1.5 milliseconds less per each message exchange. In huge message rates, the difference is stabilized between 1 and 2 milliseconds over IP, which is a very good result taking into account that INA-HIM is implemented on top of UDP. The reason that our approach is better than IP for some rates is that for lower rates it is quicker to resolve an id/loc mapping than a routing table entry.

In parallel to the above figure, Figure 5.19 shows the difference of the loss percentage between INA-HIM and IP, and between INA-HIM during the mobility events and IP. The error-bars represent the standard error of the original measures of the tests with our architecture. Both plots also have the standard error of the percentage loss measurements wrapping the plot lines. As we can see, the results are with high confidence since the standard error is very low, but the non-mobility test has slightly more confidence. Also, the two plots are very similar in their global movings, but the mobility plot is more unstable and has higher standard error. Seeing the results as a whole, our approach is separated around 1.5% of the raw-IP solution, having around 0% and 1.5% more message loss than the raw-IP approach. Like the previous results, these results also validate our identity-based approach against the behavior of the raw-IP solution.

5. Final Architecture: Beyond the Separation of Identifiers and Locators

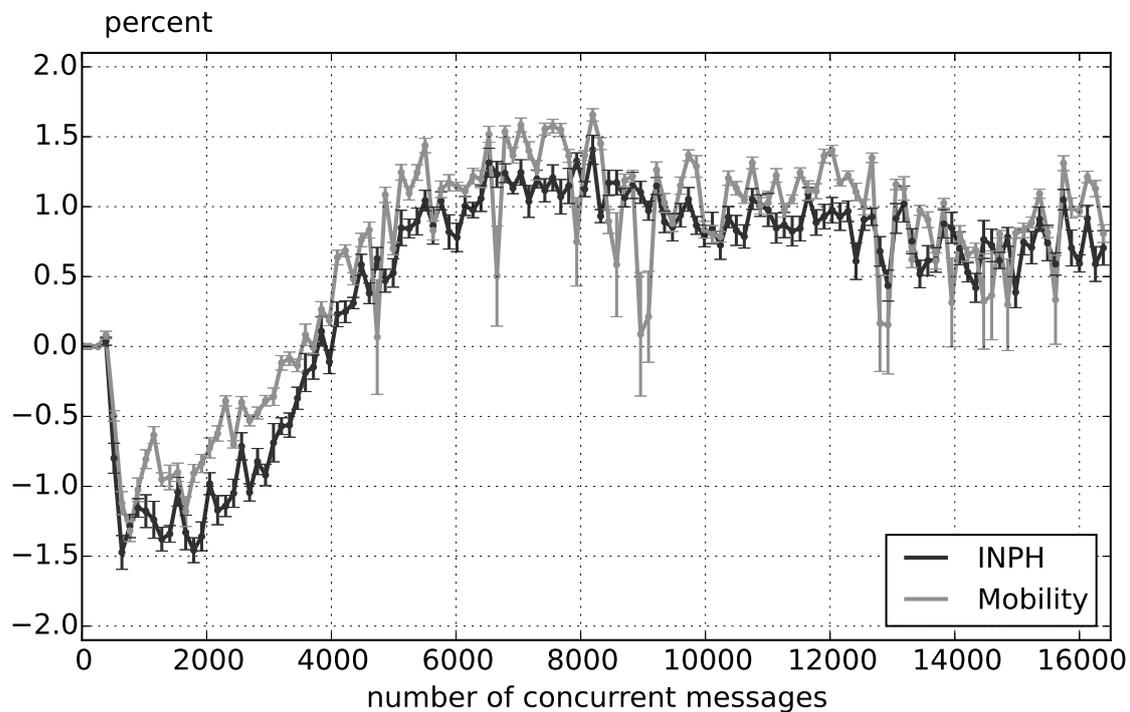


Figure 5.19: Evolution of the difference between the message loss of INA-HIM and IP, and between our architecture during a mobility event and IP.

Finally, we run an experiment with many movements (handovers) with the 500 msgs/s parameter defined from the commented results first. Figure 5.20 shows for each moment of time, starting from 0 seconds and ending at 28 seconds, the net rate of the message exchanges between two nodes. The vertical solid line marks the time when the request is sent and the vertical dashed lines mark start and end of each handover process, whose respective timespan is around 500 ms for all but for the last, which is negligible. Each 5 seconds, one of the nodes changes its location to a foreign domain and in the 25th second returns to its home domain. We can see the moment at which the moved node sends the request and the moments at which it starts and ends the handover between domains. As expected for the 500 msgs/s rate, the net rate is about 10% but during the handover processes it loses more messages. These losses are sometimes compensated, as seen in the following bars close to the handover end events. The handover events are launched every 5 seconds, so they are represented by the vertical dashed lines near the following x labels: 5, 10, 15, 20, and 25. The handover timespan is about 500 ms in all events but the last. The difference is due to the fact that the node has already prepared a network interface for its home domain, so the last handover has an almost negligible timespan.

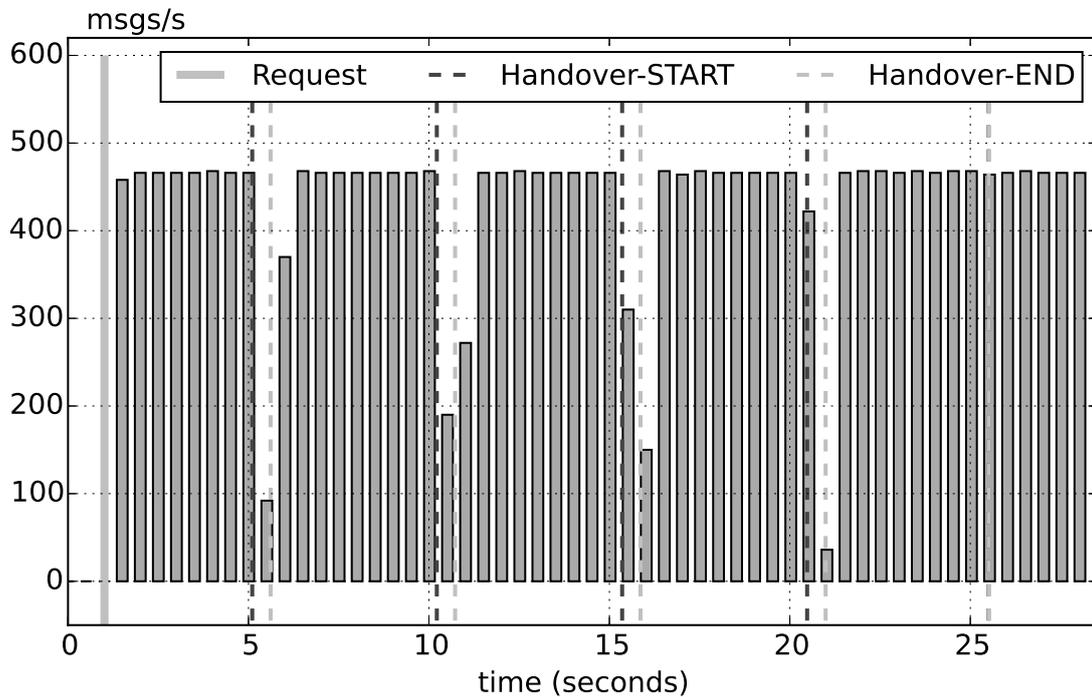


Figure 5.20: Evolution of the solution behavior as the mobile node moves through all domains while receiving messages at a rate of 500 msg/s.

5.5.7 Discussion

As illustrated by the application scenario mentioned at the beginning of this section, INA provides some important benefits that are not covered by the current Internet model or the current proposals for decoupling the identification and location. Moreover, from the feature comparison shown above we demonstrate that the architecture outlined in the current chapter, which is supported by the research results discussed throughout the thesis, includes all the features already provided by previous architectures as well as some features not present in the others. Specifically, the features only provided by INA are the newly defined requirements for the Future Internet (FI), while the other features are defined for long.

Regarding the theoretical performance analysis, we can appreciate how the cost of the discovery stage grows logarithmically with the number of nodes involved in the overlay network, so denoting a good scalability. Moreover, it does not depend at all on the number of existing identities, which is a very important result and aspect provided by INA. However, from the experimental results presented above, we determine that the performance of the architecture somehow depends on the number of attributes used in

5. Final Architecture: Beyond the Separation of Identifiers and Locators

the partial identity to perform the discovery operation, which is the most sensitive part of INA. Anyway, we demonstrated that INA behaves well for low numbers of attributes and the impact is limited when there is a relatively big number of nodes in the routing table.

With our experiments, running in a network with millions of nodes, we demonstrated that, in contrast to our initial supposition, the performance is really independent of the number of attributes, but having more attributes implies that the probability of waiting for longer responses, given by more distant nodes, is increased. This is because the attribute requests are parallelized, so the longest request determines the total time needed to retrieve all information needed from the DHT.

Furthermore, since the discovery results (virtual identities) may be cached, the average impact of the discovery on the global performance of the architecture is reduced. This makes the architecture much more feasible but, since it is always *tricky* to place caches, here we wanted to demonstrate the real performance of the architecture without using them.

From the whole analysis presented throughout this section we also determined that it is essential to have a good routing table and we studied the identity refresh time to reduce the number of retries due to node failures. Also, a deeper analysis of the discovery performance when using different lookup and routing methods, as shown in [124], together with other methods to quickly get a large number of peers in the routing table, could give us an improved method for the discovery operation. Therefore, the study of these aspects together with a detailed analysis of the architecture performance, including the caching and session creation process, is very important and thus taken into account for future work.

Apart from the discovery analysis, from the specific communication analysis, we demonstrate that the overhead added by INA to the network is negligible, because it only adds a few exchanges to the beginning of the communication sessions. This also demonstrates that the most delicate aspect of INA is the discovery stage. Finally, it is worth noting that the results obtained from INA when instantiation on top of CCN and integrated with the HIMALIS architecture are similar to those presented in [105, 120] because the implementation and test-cases are similar.

5.6 Conclusions

In this chapter we have presented the resulting architecture of the research carried out during the development of the thesis. Its main objective, the colophon, is the decoupling the identification and location mechanisms for the Future Internet (FI) that provides specific

capabilities not found in existing approaches. As discussed throughout this document, the FI ecosystem shows new requirements, defined as fine granularity of network nodes, integrated node discovery, and flexible naming. These requirements are derived from the move from host-based network nodes to a more flexible node space, particularly raised by the Internet of Things (IoT), the *cloud computing*, and the network virtualization technologies.

One of the key advancements provided by this thesis is the ability of INA to provide fine granularity to the definition of network node, which is translated to network object, by moving the host-to-host communications to process-to-process communications while defining the mechanism to separate the identification from the location. Furthermore, to identify the network nodes (processes), INA moves from plain or URI-like identifiers to fully fledged identities. Thus, the network nodes also gain the aforementioned flexible naming feature. Our architecture defines two mapping mechanisms, as discussed in Section 5.3 and Section 5.4, to map identities to context identifiers (sessions) and context identifiers to network locators (IP addresses). The former mapping also deals with the integrated node discovery, which also covers an important requirement for the FI.

The key mechanism used to achieve this objective is the DTEi (detailed in Section 4.2), which manages the identities of communication participants in a trusted and secure way. Moreover, in contrast to other architecture proposals for the separation of identification and location, INA, the final approach proposed in this thesis does not need to add an extra layer or extra overhead to the network. It just uses the split definition of IPv6 addresses to represent the location (IP prefixes) and session/context identification (joined source and destination IP address suffixes). Thus, INA does not penalize communications once they have negotiated the context identifiers. Therefore, the performance of INA depends on the node discovery and context identifier negotiation, which are decoupled from the network in the other architectures but integrated in our proposal in order to achieve the extra functionality.

We compared INA with the most important and widely known proposals for decoupling identifiers and locators, showing that our approach offers the same features plus the additional integrated discovery, flexible naming, and integrated security, which are not provided by many of the analyzed approaches. Also, we analyzed the performance of INA. As the data message exchanges do not add extra overhead, the performance of our approach is the same as that found in the current Internet. Also, as its operation just comprises a few message exchanges apart from the network node discovery, we centered the analysis on the discovery mechanism. Thus, we demonstrated its scalability, which is related to the usage of the overlay network and the construction of a Distributed Hash Table (DHT) to

5. Final Architecture: Beyond the Separation of Identifiers and Locators

perform the attribute lookups. Finally, through experimental analysis, we determined that the performance is kept inside a controlled range and scales well, depending more on the longer lookup than on the number of attributes to request.

Chapter 6

Integration with Other Network Architectures

In the previous chapter we have presented the resulting approach of joining together the components designed in the present thesis in order to build a complete network architecture for the Future Internet (FI). So far we have evaluated and validated the outcomes of our research independently but in this chapter we present how we have extended the outcomes of our research results by integrating the network model and functional blocks defined by our architecture into other network architecture proposals for the FI. The chosen architectures are HIMALIS and MOFI. As we introduced in Chapter 2, they are two approaches for the FI with outstanding qualities but with some important lacks we tried to cover with the research performed during the development of this thesis.

This chapter is organized as follows. First, in Section 6.1 we contextualize and justify the integration work. Then, in Section 6.2 we discuss the integration of our architecture with the HIMALIS network architecture, contextualizing it, describing the basis of such architecture, the integration itself, and the evaluation and experimentation results we got to defend the outcomes of the integration. In Section 6.3 we discuss the integration of our architecture with MOFI, giving an overview of SEID together with the target architecture and analyzing the benefits it provides. Finally, in Section 6.4 we conclude this chapter, highlight the outcomes of both integrations, and discuss some points to continue the collaboration work with both HIMALIS and MOFI.

6.1 Introduction

As discussed throughout this thesis, the use of IP addresses as both host identifiers and locators in the protocols that form the current Internet imposes constraints on designing efficient solutions for mobility, multihoming, renumbering, and security. To eliminate such constraints, different approaches have recently proposed to split the network layer into identifier and locator layers for future network architectures, specially the FI. To this respect, among the multiple alternatives, we recall here both HIMALIS [14, 120] and MOFI [33]. They represent two major advances in the research towards the network of the future coming from two different but related fronts. The former has been designed within the AKARI project [23] by the NICT in Japan as part of a much bigger and complete network architecture while the latter has been designed as part of the project with the same name by ETRI in South Korea.

On the one hand, the HIMALIS network architecture proposes to use different sets of values for identifiers and locators and allows the network layer to change locators without requiring the upper layers to change identifiers. However, one of the major challenges of HIMALIS is the design and implementation of a distributed database system to map identifiers to locators. It has to efficiently store, update and provide the up-to-date mapping data to the network elements. In order to provide such function, in this chapter we propose to integrate the functional blocks provided by our architecture with HIMALIS. This way it gains a unified mechanism to get locators from high level identifiers (names) with enhanced security, privacy, and trust, while maintaining all capabilities and full compatibility with the previous DNR, HNR, and IDR infrastructures found in HIMALIS.

Moreover, the raise of Software Defined Networking (SDN) has opened a wide spectrum of new improvements for the network, which can be effectively exploited to enhance future network architectures. We take advantage of the qualities offered by SDN to change the way HIMALIS addresses its control operations. We propose to leverage such control operations through IBCP, which is the *identity-based control plane* we designed within our architecture. Thus, we designed a *controller* that attaches to the north-bound API of SDN in order to deepen the integration with HIMALIS in the way it takes advantage of the global knowledge of the network status and the evolution of the functions provided by the control plane without changing the implementation of end nodes.

On the other hand, with the recent growth of smart phone services, it is envisioned that *mobile* will be the key driver toward FI and, among other outstanding capabilities and similarities with HIMALIS, MOFI has the quality of considering *hosts* to be mobile from the beginning. However, we found some lacks in how MOFI address identification

of network nodes, which is one of the key qualities offered by our architecture. Therefore, in this chapter we also present how to integrate our architecture with MOFI by means of encapsulating the functions offered by our architecture in a new functional block for MOFI, the SEcure IDentification (SEID) functional block. It adds secure identification of communication participants to the MOFI architecture, effectively extending it with new functions without altering the other functions it defines.

To demonstrate our claims and the benefits obtained from the integrations we have evaluated them from different perspectives. First we have built mathematical and simulation models for HIMALIS and MOFI, both with and without adding the new functions. Then, for the cases we considered appropriate to get more empirical evidences, we have built proof-of-concept implementations and executed them in controlled environments.

The proposal and analysis of both integrations demonstrate the benefits and outcomes obtained from the general research work and architecture design carried out in this thesis. It is worth to remark also that we have established strong collaboration relations with those architectures that will be reflected in future work, as discussed in Chapter 7.

6.2 Integration with the HIMALIS Network Architecture

The current Internet is based on two kinds of namespaces: domain names and IP addresses. Internet applications resolve the domain name into an IP address during a communication initialization phase via a Domain Name System (DNS) lookup. The IP address is then used by the networking protocols to identify communication sessions and locate the destination host during data communications.

That said and as we have discussed throughout this thesis, it is widely known that there are some significant problems in the use of IP addresses in the whole host protocol stack. Namely, an IP address is used in the network layer protocols as a locator to forward packets and locate the destination host. The same IP address is also used in the transport and upper layer protocols as the host identifier (IDs). This dual role of IP addresses as host IDs and locators makes difficult to design efficient solutions for mobility, multihoming, renumbering, and security because such solutions require the provision to change locators used at the network layer without changing the host IDs used at the transport, session, and application layers. So, in the search towards the FI, the separation of ID and locator has become one of the most important challenges [1, 12].

6. Integration with Other Network Architectures

We recall and emphasize that some of the limitations found in the Internet have resulted from the use of a single numbering space, i.e. IP addresses, as both host identifiers and locators [77]. In particular, an IP address is used in the network layer protocols as a locator to find the destination host in the network topology. The same IP address is also used in the transport and upper layer protocols as the host identifier (ID). This dual role of IP addresses as host IDs and locators makes the current Internet unable to natively support mobility, multihoming, renumbering, scalable routing, and security because these functionalities require the provision to change locators without changing the host IDs [126].

As we introduced in Chapter 2, there are several proposals to overcome those issues, providing a clean-slate design so that their functionalities are not affected by the inherent limitations of the existing Internet. Here we highlight the Heterogeneity Inclusion and Mobility Adaptation through Locator ID Separation (HIMALIS) [14, 120]. It proposes a comprehensive architecture to address the problems of mobility, multihoming, routing and security as well as supporting heterogeneous protocols in the network layer. The HIMALIS architecture uses distinct sets of values for identifiers and locators and allows the network layer to change locators without requiring the upper layers to change identifiers.

Therefore, the HIMALIS architecture implements ID/locator mapping functions in the end hosts as well as in the edge routers or gateways. A new layer, called identity layer, is introduced between the transport and network layers to execute these functions. The new layer separates the scopes of IDs and locators, i.e. IDs are used in the application and transport layers to identify sockets while locators are used in the network layer to locate destination hosts and forward packets towards them. ID/locator mapping records are required to perform the ID/locator mapping functions. In order to efficiently maintain and retrieve ID/locator mapping records, it introduces three new elements that cooperate with hosts and gateways. The new elements are the Domain Name Registry (DNR), the Host Name Registry (HNR), and the ID Registry (IDR). In summary, the DNR is a name resolution infrastructure used to resolve domain names to the address of the HNR elements. Then, the HNR elements, which in turn are instantiated on edge networks, are responsible of maintaining hostname to ID and locator mapping records. Finally, the IDR is a distributed registry used to maintain the ID/locator mappings and communicate their updates to the routers that connect those edge networks to the global transit network.

In previous work [14], the DNR, HNR and IDR are organized into a hierarchical logical network, which are queried by hosts and gateways in a sequential order. Although the hierarchical structure is good for scalability, it may not optimal for faster updates of records. Therefore, here we explore an alternative architecture by joining the functions provided by the HNR and IDR and deliver them by our architecture, specifically using

6.2 Integration with the HIMALIS Network Architecture

the Domain Trusted Entity Infrastructure (DTEi) introduced in Section 4.2. This way the resulting architecture will be identity-based, gaining enhanced security and robustness without performance penalty. Moreover, the DNR functionality is unnecessary because the elements of the DTEi (joint IDR/HNR) can be reached through the overlay network they form. Additionally, our architecture will provide HIMALIS with some sort of identity management mechanisms, such as checking policies to control the communications or dynamically change of Host IDs to enhance privacy. Also, it provides to network elements the ability to know that an entity is *who* it is pretending to be and, thus, getting inherent and integrated authentication and, when applying policies, the authorization of network operations.

On the other hand, the mechanisms to separate control and data planes in the network as part of the emergence of Software Defined Networking (SDN) approaches has supposed the opening of a wide range of improvement possibilities to network architectures, in terms of efficiency, control, and deployment ability. Exploiting these capabilities will facilitate the adoption of new network architectures that finally benefit the network and its users. This is especially beneficial for architectures targeting next generation networks since SDN mechanisms provide them the perfect companion to their adoption and production deployment. Therefore, we also propose to evolve HIMALIS and adapt it to SDN by means of the IBCP we discussed in Section 4.6, which will be integrated with the control plane of HIMALIS to operate its network elements and thus achieve its mobility and operations in an efficient way. The IBCP is also connected with the SDN controller, so it has more direct relation to the underlying network. This way, HIMALIS may take advantage of the benefits offered by SDN in a simple way, thus increasing its deployment ability while gaining efficiency and exposing more network functions to control plane operators.

This can be achieved thanks to the clear and strict separation of control and data planes promoted by SDN. Such separation allows network engineers and administrators to have full control of network behavior and general operation. On the one hand, this is a key requirement for IT infrastructures to adopt a new architecture, so it is a must for any network architecture that would be widely implemented in the future. On the other hand, the strict separation of control and data planes adds flexibility to the network architecture, so it can be extended to different underlying network infrastructures in the data plane while keeping untouched all its capabilities offered through the control plane. This improves the heterogeneity of the architecture with little impact to its complexity.

Other important aspect of networks is reliability. In general, increasing network efficiency may suppose an impact to its reliability but addressing control operations through the identity-based control plane working on top of the SDN control plane permits to add

6. Integration with Other Network Architectures

efficiency to the network, both in terms of control and data plane, without a negative impact to network reliability. This is achieved by restricting efficiency improvement operations to a single plane at a time, which is greatly facilitated by the capabilities offered by SDN.

In this section we finally demonstrate the benefits of using an identity-based control plane to perform control exchanges in HIMALIS by building and executing simulation models for both HIMALIS and our approach. The results show a valuable improvement of the integration proposed in this chapter.

6.2.1 Architecture Overview

The HIMALIS network architecture has been designed to achieve communications over heterogeneous underlying networks with emphasis in mobile nodes. Its design follows the identifier/locator separation design principles described in [2]. Thus, it includes a new naming scheme for generating host names and identifiers (IDs), which are decoupled from low layer network locators. Host IDs are used as anchor values to allow changes in lower layers without affecting transport or application layers. This way, the architecture provides support for mobility, multihoming, and security functions [120].

Host name resolution to host IDs and network locators is a key aspect of the architecture, and it is provided by the Host Name Registry (HNR). Following the separation of concerns design principle and with the objective of scalability, there is a different HNR instance on each administrative or network domain and each instance is in charge of the mapping entries of its domain.

As hosts have to know which HNR they have to contact, they have to query the Domain Name Registry (DNR) that will resolve the locator of the HNR provided the name of the target host. The DNR infrastructure is hierarchically organized like the current DNS and it only stores the names/locators assigned to HNRs but not the global host names of all nodes. Therefore, the DNR database does not grow as fast as the number of hosts, so the retrieval process for DNR records is kept fast and scalable.

Putting these elements together on the network, as shown in Figure 6.1, the HIMALIS view of the network is mainly composed of edge (or access) networks, gateways, a global transit network, and the unified logical control network. The edge networks provide network access to various end systems or hosts, which are able to use different access technologies and network layer protocols as well as different numbering spaces for locators (called local locators), which are routable only in the respective edge networks. For example, some edge networks may use prefix aggregatable locators, like the current IP addresses, while others may use geographical coordinates given by GPS as locators.

6.2 Integration with the HIMALIS Network Architecture

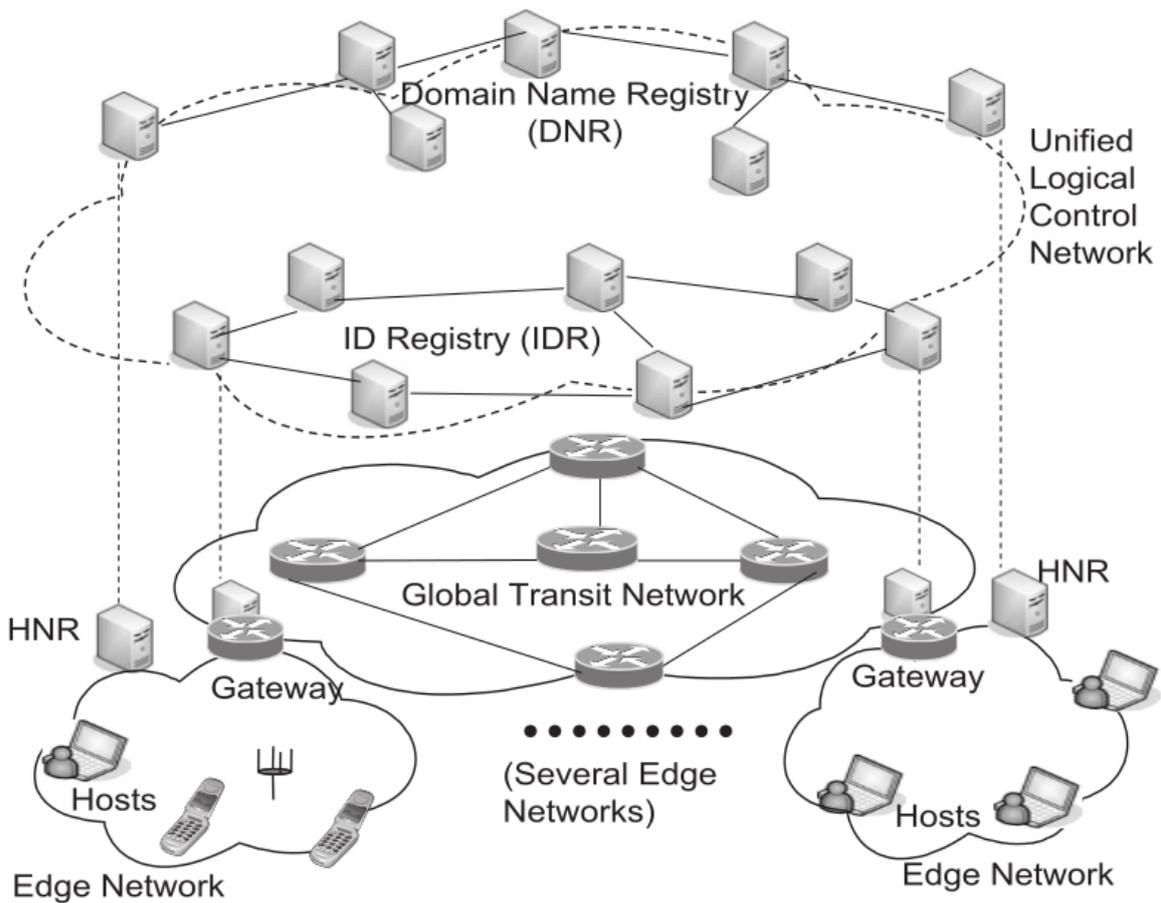


Figure 6.1: HIMALIS network architecture overview.

The global transit network, which would be the collection of service providers' backbone networks, will use a single network layer protocol and single numbering space for global locators that are used to forward packets among edge networks. The gateways located on the border between edge and transit networks translate local locator and network layer protocol into the global locator and network layer protocol for communication expanding across the border.

The unified logical control network contains the HNR, DNR, and IDR. The DNR records are updated by the HNRs, the IDR records are updated by the gateways, and the HNR records are updated by the end hosts. In addition to these registries' logical networks, the unified control network may contain additional logical networks for maintaining and distributing information on authentication, authorization, accounting (AAA), policy profiles, network configuration, QoS control, etc.

6. Integration with Other Network Architectures

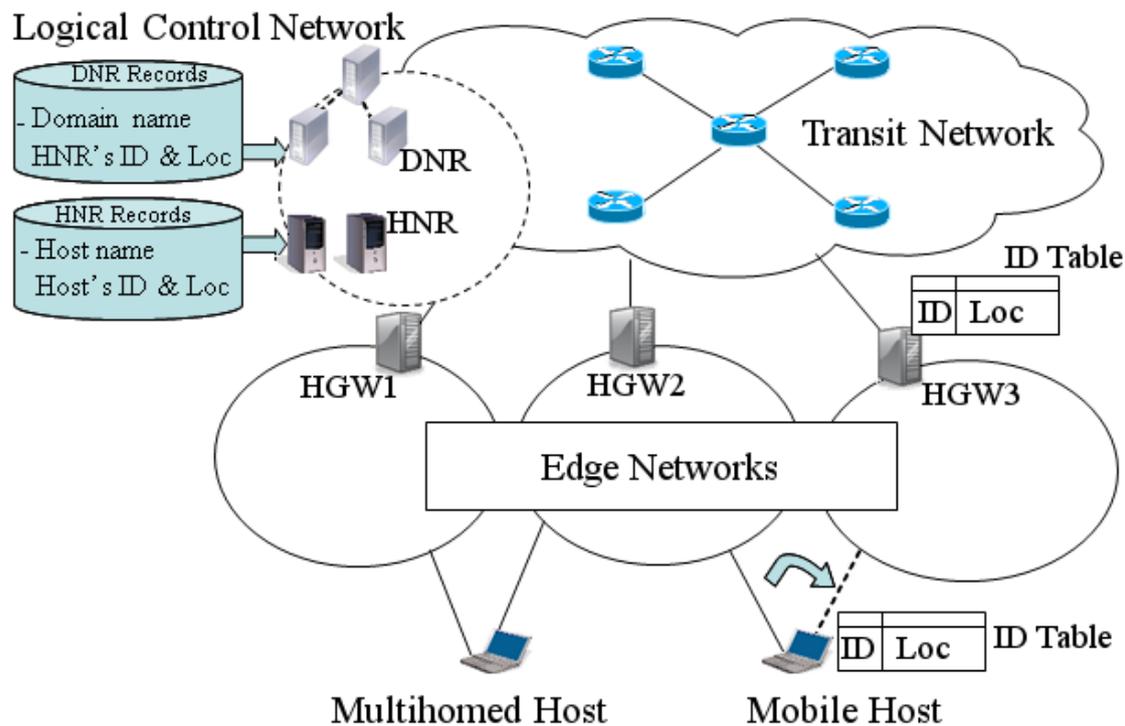


Figure 6.2: HIMALIS network architecture overview (updated).

The mapping information is provided by the gateways. Each gateway collects the mappings to populate the IDR infrastructure. It is achieved by interacting with hosts or observing host behavior, i.e., when a host initiates a session or enters from another edge network to the edge network connected through the gateway. They also store a copy of the mappings and use it to translate protocols or locators. Moreover, they also support fault tolerance. In other words, when a gateway crashes, a substitute gateway of the same edge network can download the mappings stored in the IDR to resume the communications.

Finally, in the latest iterations of the architecture design, as shown in Figure 6.2, the aforementioned elements are complemented by two other elements: the Local Name Server (LNS) and the Authentication Agent/Registrant (AAR). These are introduced to locally store host information (closer to the nodes and gateways) and also enforcing security in network access [127].

6.2.2 Integration Proposal

In this section we detail our view of the profile that uses the functionality provided by the joined IDR/HNR infrastructure mentioned above.

6.2 Integration with the HIMALIS Network Architecture

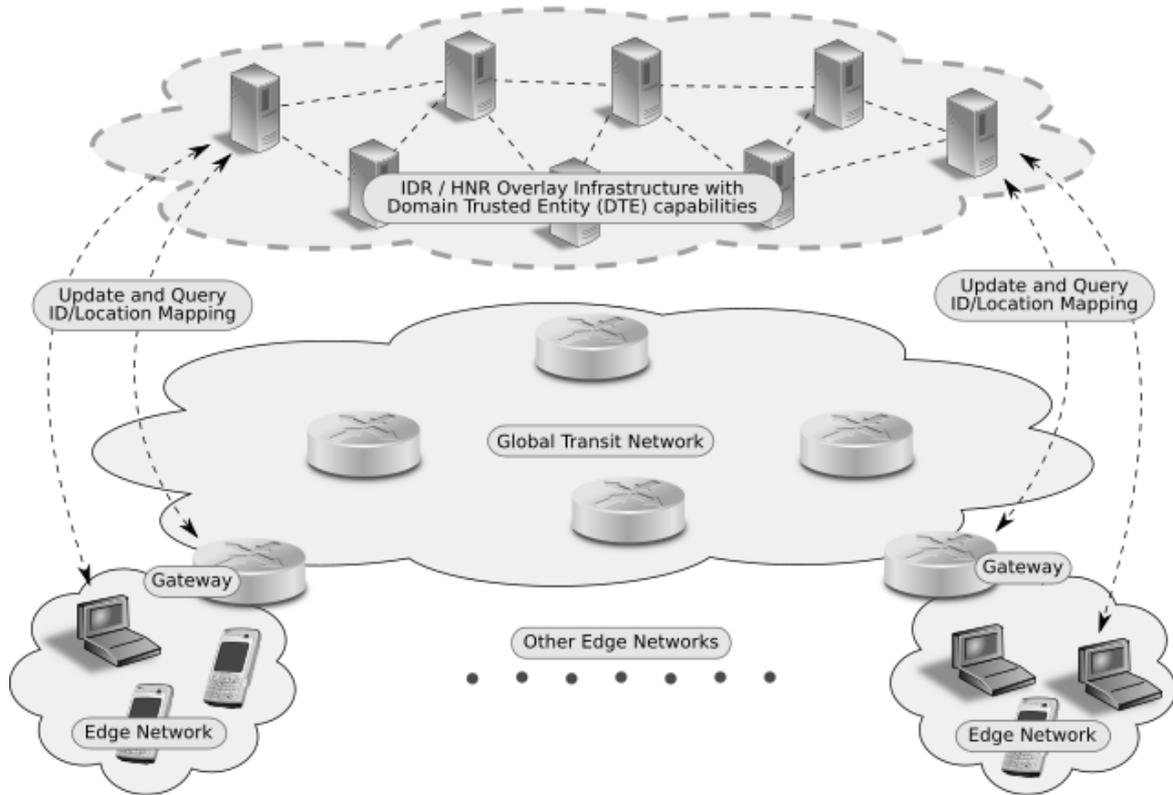


Figure 6.3: Integrated architecture.

First, Figure 6.3 shows the architecture of the profile we propose for HIMALIS. On it we depict the general components defined by HIMALIS, changing the IDR by our IDR/HNR infrastructure and, on purpose, not showing the DNR because we assume that IDR/HNR elements can be reached without needing a previous name resolution. Also we depict with arrows the main interactions between the gateways and the IDR/HNR, as well as the end devices and the IDR/HNR infrastructure.

Each element of the IDR/HNR is responsible of a network domain and therefore it is instantiated (usually) in the primary or main network of the same domain. It is also responsible of the identification domain of the entities belonging to its domain. This identification domain may not match with its physical/logical network domain because of mobility. As introduced above, all IDR/HNR elements of each domain are connected to form an overlay network that is independent of the physical/logical topology. Although overlay networks like Chord may seem inefficient, their simplicity have led to the emergence of many improvements, such as LPRS [36], a derivation of Chord that makes it perform very close to its underlying network. Furthermore, each element of the IDR/HNR

6. Integration with Other Network Architectures

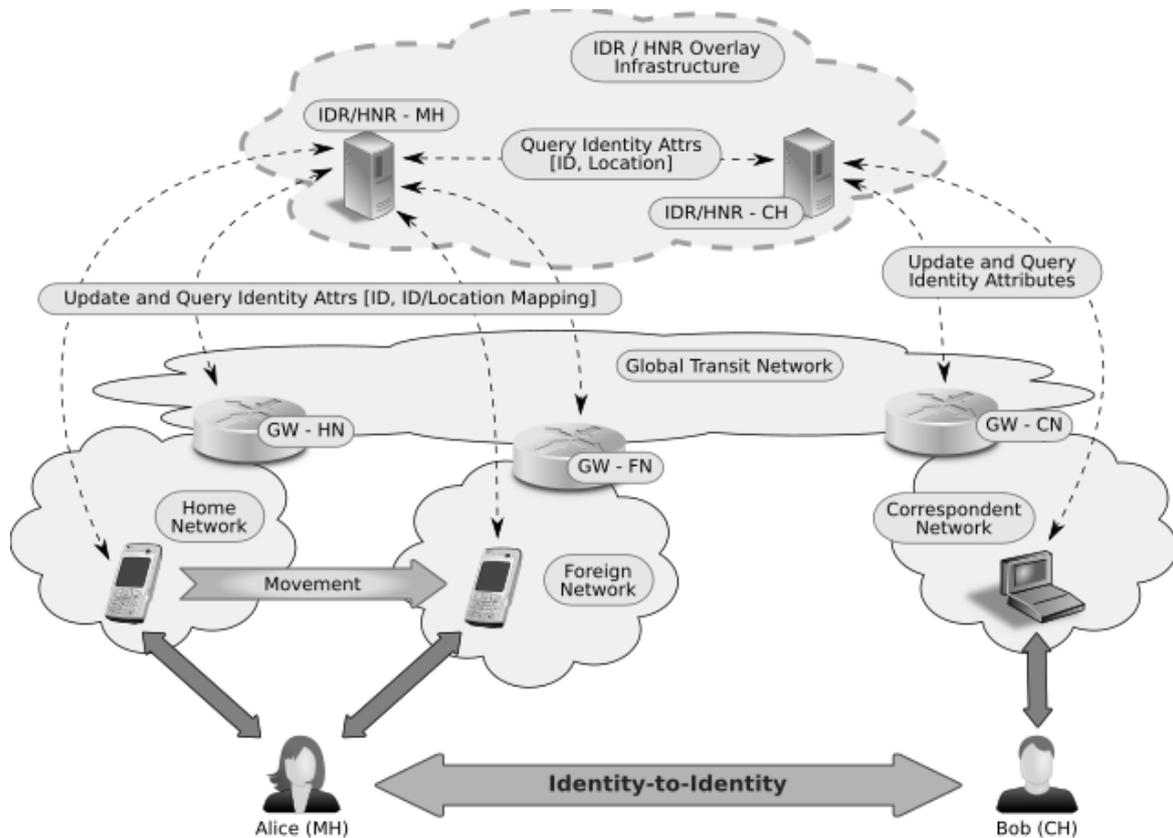


Figure 6.4: Integrated architecture remarking mobility and identities.

infrastructure has certain level of trust with the other elements and has the necessary mechanisms to securely communicate.

Like the HIMALIS architecture, the integrated profile we propose starts with a Host ID that has to be mapped to a locator (address), but now the gateways asks to the joined IDR/HNR infrastructure for those mappings. Also, when necessary, the gateways will update the id/loc mapping information but final devices can also manage that information, because they are the primary authority of their id/loc mapping. Although the mapping is supported by the HostID of the entities taking part of the communication, they are not fixed and can be dynamically generated. In this case, the IDR/HNR infrastructure can be used to resolve the HostID of a communication party from other attributes of its digital identity. To keep security, this process is controlled by policies that determine if the operation can or cannot be done.

Once gateways have received the necessary mappings to get addresses from HostIDs, they keep them on their mapping cache until they expire or new information is pushed by the IDR/HNR. Thus, the IDR/HNR has a close and mutual collaboration with gateways,

6.2 Integration with the HIMALIS Network Architecture

both in an active manner. The final device can also take part of this collaborative process but it is not strictly necessary, thus devices with reduced capabilities (like those in sensor networks) are totally supported by the architecture.

After introducing the whole architecture, Figure 6.4 shows the interactions and elements involved in an expanded scenario in which a device moves from an edge network to another. Here we show how IDR/HNR elements interact to bring id/loc mapping information to the necessary gateways (those involved in the communication). Also, it depicts how the address or location of a device is treated as an attribute of the identity that is behind it, so its access can be controlled by policies, like other identity attributes. The result is that an entity represented by a digital identity can *talk* to other entity represented by another digital identity.

6.2.3 Message Exchanges

In this section we discuss the necessary message exchanges to implement the integrated architecture discussed in the previous section. Thus we detail the specific messages needed to start a session between two entities and the specific messages needed to update the id/loc mappings stored in different network elements when when an entity changes its location (handover).

Before two (or more) entities can start a communication they need to establish a session between (among) them. First of all, the entities (or their devices) must be registered in the IDR/HNR elements of their corresponding domains. Here we call this process “authentication” because, on it, the entities behind the devices confirm their identities and thus the IDR/HNR can authenticate them whenever needed.

After the authentication, the entity (or its device) can communicate with other entities but, before, it must find them. To do it, the requester entity contacts with the IDR/HNR element of its domain and asks it to find the desired destination identity from certain provided information. Here, in the HIMALIS integrated profile, the information may be the HostID but, as commented above, it is not limited to it. Thus, the process can operate with a general query to the attributes of the digital identity, always subject to the policies set by its owner and also taking into account the identity of the requester.

When the search process ends, the requester entity and the IDR/HNR element of its domain receive the necessary information to communicate with the other identity (representing single or multiple entities), such as the *facets* it exposes, which here is like a virtual identity. Although this process is needed to search the descriptors of any identity,

6. Integration with Other Network Architectures

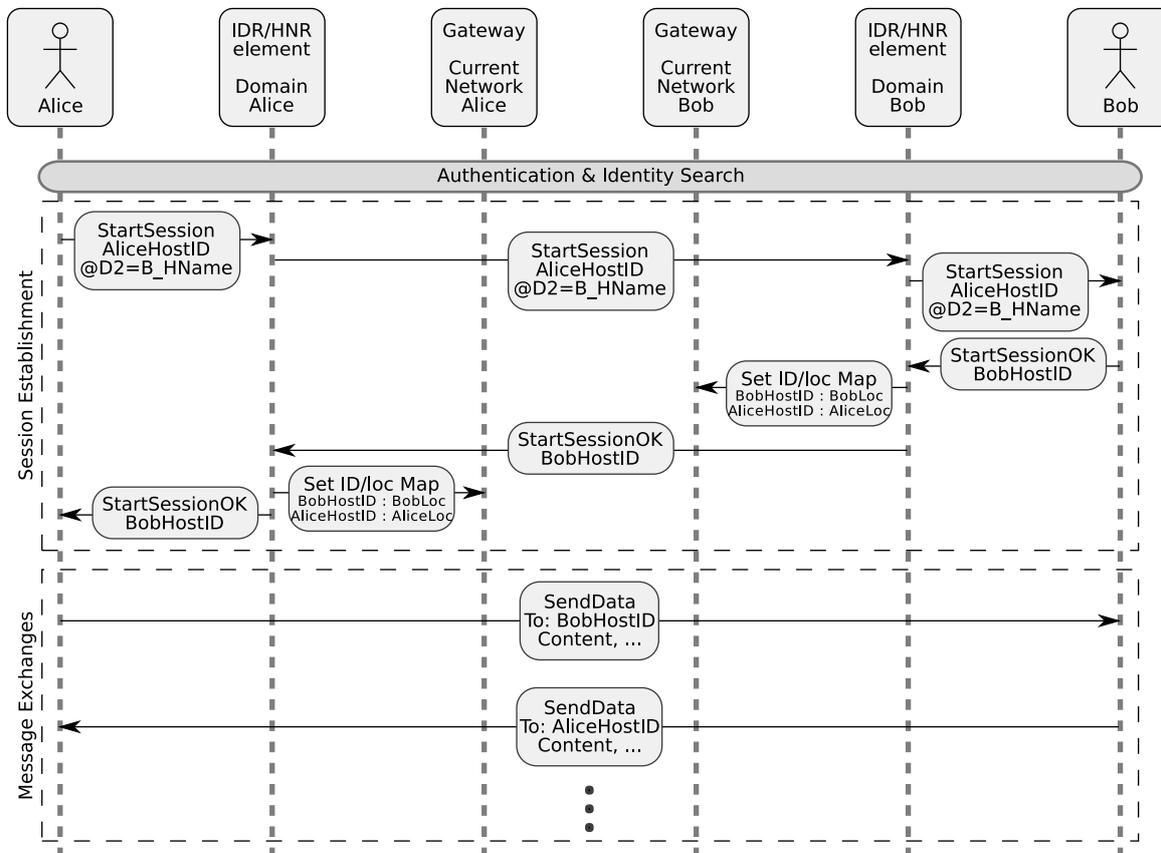


Figure 6.5: Messages exchanged to start a session.

the results can be cached to be used in subsequent communications. Finally, the requester entity selects a *facet* of the destination identity and the session can start.

From this point, the HIMALIS integrated profile proposed in this chapter differs from the process followed by the current HIMALIS architecture. To start a session, generally the first session, instead of let entities communicate by their own, we propose to place the IDR/HNR infrastructure in the middle to negotiate the session attributes, such as the session identifiers used by each endpoint. As the IDR/HNR has a highly secure and reliable way to make this communication, using it to make the initial negotiation strengthens security. Figure 6.5 shows the messages exchanged to start a session between Alice and Bob, which actually represent the devices used by the actual entities. It works as follows:

1. Alice sends to the IDR/HNR element of its domain a message called *StartSession* which contains its HostID and a query to find Bob by means of its hostname, that is seen as an attribute of Bob's identity.

6.2 Integration with the HIMALIS Network Architecture

2. The IDR/HNR element of Alice's domain sends a request to the IDR/HNR element of the other domain (Bob's domain, "D2") with the information provided by Alice. Each IDR/HNR element can reach each other through the overlay network without needing to resolve any name, just using their domain identifier.
3. Now, the IDR/HNR element of "D2" checks the corresponding policies set for Bob's hostname and searches the entity that responds to it. Then it sends the *StartSession* request to Bob because it responds to *Bob*.
4. Bob accepts the session, keeps the HostID of Alice, and sends a *StartSessionOK* message with its HostID to its IDR/HNR element.
5. Once the IDR/HNR element of Bob's domain has the HostIDs and locations of Alice and Bob, it reports to the current gateway to which Bob is connected to set new mappings for the HostIDs with the locations (BobHostID, AliceHostID).
6. After setting the mapping to the gateway, The IDR/HNR element of Bob's domain sends a *StartSessionOK* message to the corresponding IDR/HNR element of Alice's domain.
7. As did the IDR/HNR element of Bob's domain, the IDR/HNR element of Alice's domain sends a message to set the id/loc mappings to the current gateway to which Alice is connected.
8. Finally, the IDR/HNR of Alice's domain sends the *StartSessionOK* to Alice and the session is considered started, so Alice begins to send messages to Bob. These messages are intercepted by the gateway that use the mapping to know the location of Bob, that is where it delivers the messages. The same happens when Bob sends messages to Alice.

Once the session is started we show what happens when one of the entities move to other network. After this, the gateways involved in the communication (old and new) must be updated with the new mappings. Below we comment how this is achieved by our proposal and Figure 6.6 illustrates it. This process operates as follows:

1. After changing its location, Alice sends a *ChangedLoc* message to the IDR/HNR element of its domain indicating its new location.

6. Integration with Other Network Architectures

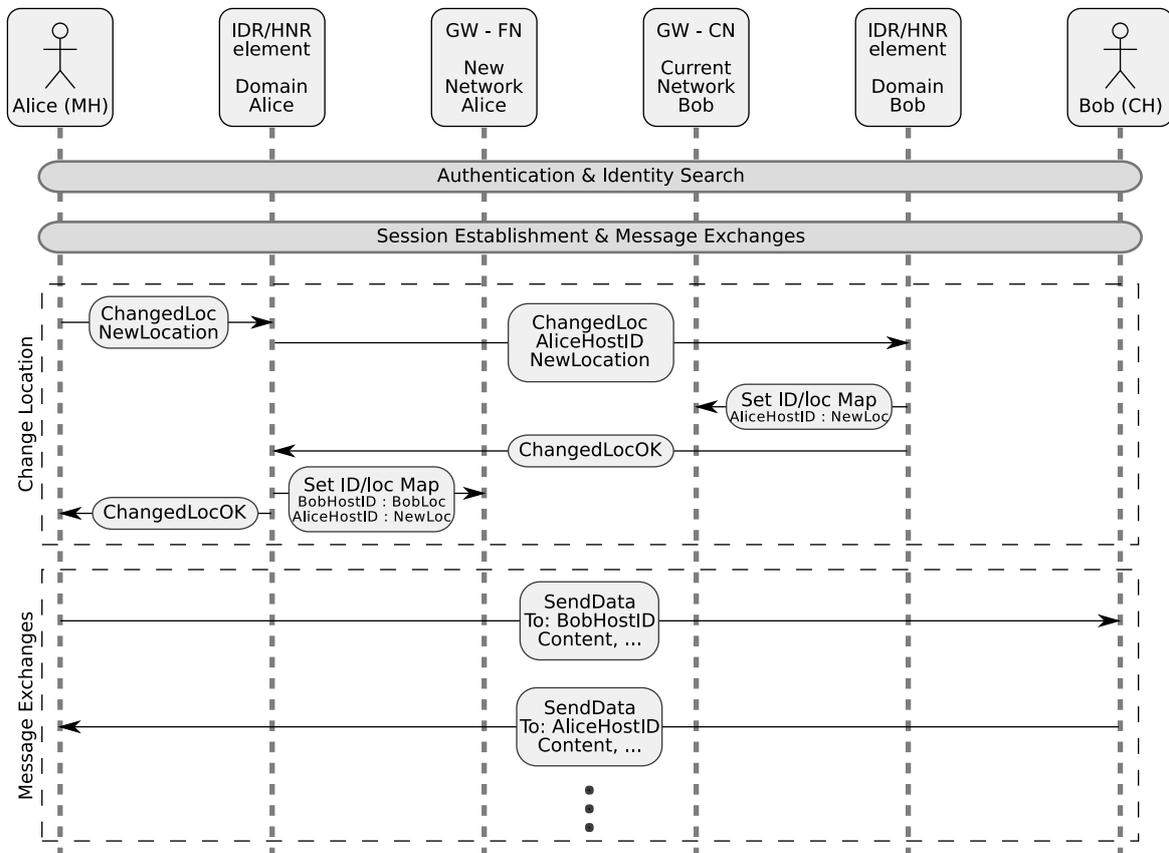


Figure 6.6: Messages exchanged after a movement (handover).

2. Since Alice's IDR/HNR element knows the opened sessions, it sends a *ChangedLoc* message for each session to their corresponding IDR/HNR element, so here it sends a *ChangedLoc* message to the IDR/HNR element of Bob's domain with *AliceHostID* and Alice's new locator.
3. As the IDR/HNR element of Bob's domain knows that Bob's HostID has not changed, it only sends the new mapping of Alice to the gateway to which Bob is connected using the appropriate message.
4. After knowing that the change has taken place, the IDR/HNR element of Bob's domain confirms it sending a *ChangedLocOK* message to the IDR/HNR element of Alice's domain.
5. Now the IDR/HNR element of Alice's domain sends the id/loc mapping of Alice and Bob to the new gateway to which Alice is connected.

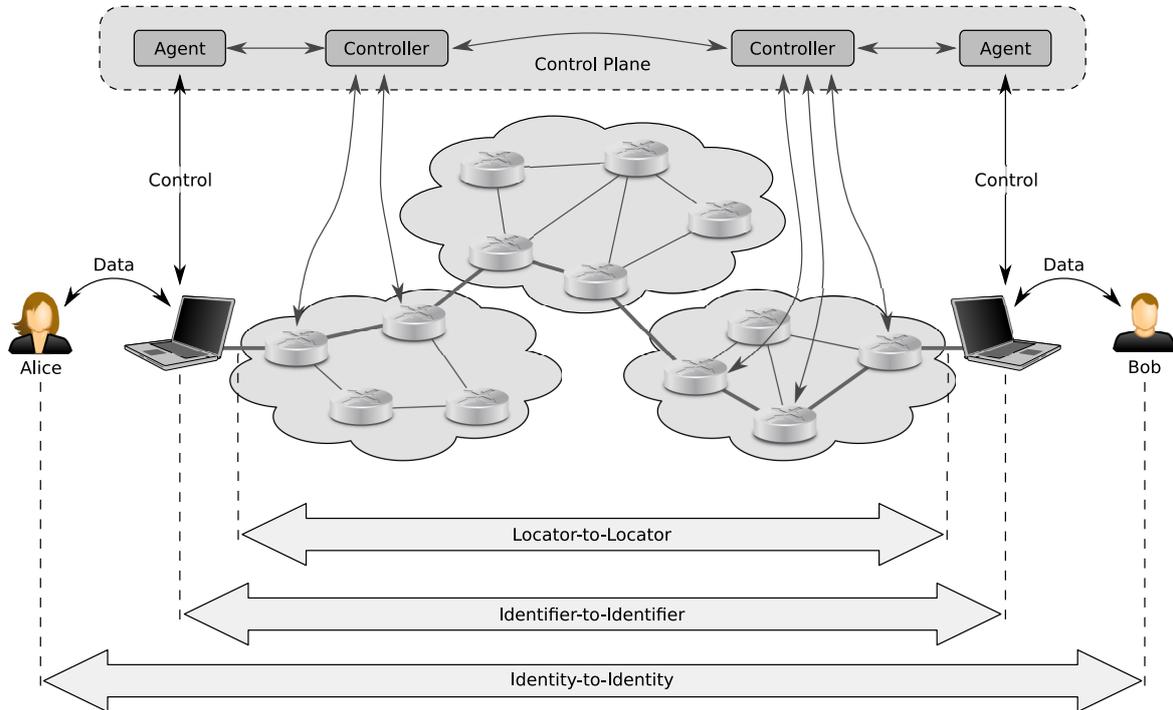


Figure 6.7: Integrated control plane with HIMALIS.

- Finally, the IDR/HNR of Alice’s domain sends the *ChangedLocOK* message to Alice, that now can continue its communication with Bob.

In parallel to this process, after setting the new mappings in the new gateway, the IDR/HNR of Alice’s domain reports back the new location of Alice to the old gateway so it can send its missing messages to Alice’s new location.

6.2.4 Attaching the Identity-Based Control Plane.

Here we describe how to integrate the identity-based control plane of our network architecture with the HIMALIS architecture in order to enhance its flexibility, evolution ability, and handover performance. As a side effect, the resulting architecture will be easily deployable on SDN, which facilitates the deployment of HIMALIS in the network.

The first design decision we have taken to achieve the integration is moving the control functions implemented by HIMALIS elements to the control plane. As shown in Figure 6.7, this is achieved by designing a network controller for HIMALIS (split into agent and controller) and connecting it to the intermediate elements of the underlying network (e.g. switches). Thus, the controller can capture HIMALIS control operations and address them through the control plane, parallel to what is discussed in [128].

6. Integration with Other Network Architectures

HIMALIS elements do not lose their role. They will be kept separated but, as they are tied to the controller, they will increase their cooperation with general network control tasks in order to build a big network knowledge base. It includes host identifiers and locators (upper and lower layer locators as IP addresses and DPID/port of switches). The effects of handover operations are also gathered by the controller so it knows the state of the network in every moment. This can be enlarged with other identity aspects of both network hosts and the objects that stay behind them, so the knowledge base is kept as much complete as possible in order to improve the general behavior of the network.

Concentrating the knowledge about network hosts and their operations improves network efficiency. This is achieved by avoiding superfluous signaling in typical operations, like handover operations, so reducing the required time to complete them. For instance, the controller will notice the mobility event instantly and update the locators in the necessary elements but just using the minimum required exchanges through the control plane, which is not disrupted by the handover. Notifications to interested elements are sorted by their importance, so main elements involved in communications are notified promptly while secondary elements are notified later.

While getting all benefits described above, the integration with the identity-based control plane exposes some new research challenges. First, decentralized controllers have to be coordinated to implement HIMALIS functions the same way as they were centralized. Second, a model to enforce general governance and administrative policies is required to address both inter-domain and intra-domain networking operations. Third and final, a bigger model that includes inter-domain operations is required to analyze the behavior of the proposed solution working with different administrative policies. All these challenges should be addressed in future works.

6.2.5 Evaluation of the Integrated Architecture

Once we have described the architecture design and defined the behavior of the integrated architecture we wanted to get an approach of its performance so we implemented a proof-of-concept solution and used it to evaluate the proposal and demonstrate the behavior of the architecture. In the following subsections we describe the implementation, the testbed, and the evaluation performed with them.

Proof-of-Concept Implementation The implementation of the architecture consists of many base components and a few final applications to perform the evaluation described below. First, we built a library that implements the protocol used by our architecture,

6.2 Integration with the HIMALIS Network Architecture

as well as the clients that use it. We also built a library with an implementation of the Chord [35] overlay routing algorithm used by the elements of our architecture to communicate each other. With these libraries, plus a UDP-based low-level transport layer, we built a module for the instantiation of three network intermediate elements (home, foreign, correspondent) and other module for the two clients (the Mobile Node and the Correspondent Node). We remark that using UDP implies that, when performing the tests, some (or many) messages are going to be lost for high data-rates. Finally, using also the UDP transport, we have implemented a simple gateway module with id/loc mapping support and which is able to receive ID-based messages and transmit them to other gateway or to a client, depending on the location of the client. All components are implemented with the Python language because of its simplicity and flexibility, but the critical modules, like the Gateway, are then built to binary using Cython.

We decided to implement the overlay network following the Chord approach with the only optimization of the finger table that shorts the process of finding far nodes in the overlay. Therefore, although some performance improvements are certainly possible to be applied to the communication across the DTEi included within our architecture, as we show in [122], here we have not focused on this aspect but on the secure identity-to-identity communication and the mobility support of our architecture applied as an id/loc mapping system.

Finally, the client nodes are defined to make a simple communication. The Mobile Node asks for a number of messages, specifying the rate at which she wants them, and the Correspondent Node sends the messages to her. In a mobility scenario it performs the same operations but at certain moment the Mobile Node changes its location (changes its current network). Finally, we built other clients using direct UDP exchanges over IP instead of the Gateways to communicate each other.

Evaluation Environment To test the solution described above we built an evaluation environment that consists of 6 networks: a virtualized interconnection network which acts as the global transit network defined in the architecture, and 5 edge networks, two of which are real networks and the remaining three are virtual networks. All nodes are Virtual Machines (VMs) running in top of the Xen virtualization technology (para-virtualization), a widespread solution in high performance virtualization environments, like in many Cloud Computing infrastructures. All the equipment, virtual and physical, runs the Linux operating system. The virtualized networks are built using Linux kernel bridges in the host machines and using VTun [125] to build ethernet bridges through TCP/IP connections between separated host machines.

6. Integration with Other Network Architectures

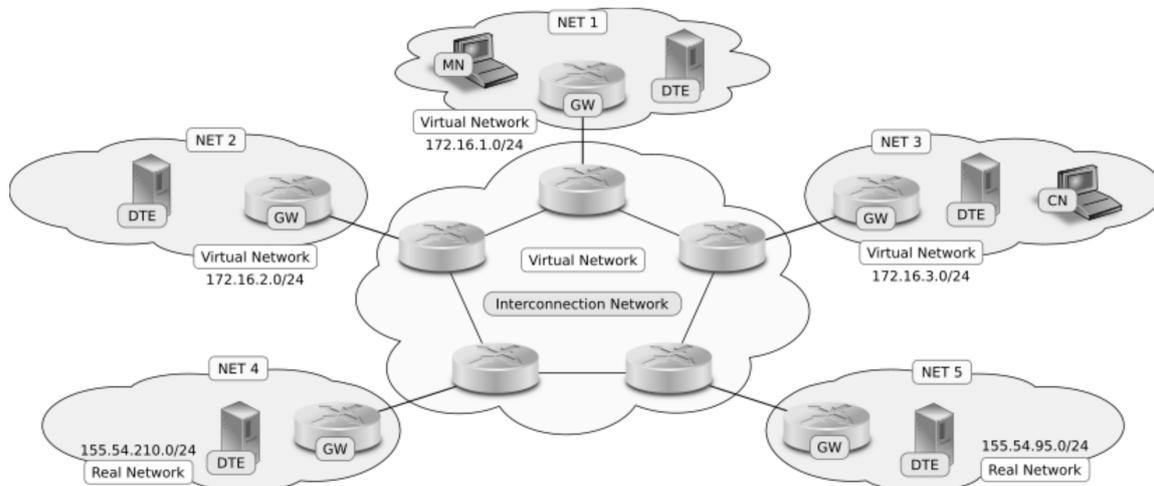


Figure 6.8: Evaluation environment for our architecture integrated with HIMALIS.

Figure 6.8 shows the topology of the evaluation environment with all the elements defined in our architecture (nodes of the DTE_i and GWs), as well as the client nodes (MN and CN). The edge networks correspond to different network and identity domains. Thus, the Mobile Node (MN) and Correspondent Node (CN) respectively belong to Domain 1 and Domain 3. As expected, regardless of whether they are connected to the real or virtual networks, the client nodes are configured to route messages through their corresponding Gateway which is configured to route them through the interconnection network.

As defined in our approach, the DTE_i is built as an overlay network. The names used by its nodes are: “domain1”, “domain2”, “domain3”, “domain4”, and “domain5”. Thus, using the mapping system we implemented in the Chord layer, that uses 16-bit identifiers, they correspond to: “0xEFA2”, “0x804F”, “0x23CC”, “0xBD92”, and “0x26BB”. These identifiers are disperse enough to get message exchanges that cross different domains. For instance, the node of domain 3 needs to cross the node of domain 2 to reach the nodes deployed in domains 1 and 4, but can reach the nodes of domains 2 and 5 directly, in just one hop through the overlay network.

6.2.6 Evaluation of the Integrated Control Plane

To demonstrate the main functional benefit of the proposed solution to the handover behavior and performance of the HIMALIS architecture we have built separate models for plain HIMALIS and the proposed approach. Both models have been designed and constructed following the same methodology so the results can be compared effectively.

6.2 Integration with the HIMALIS Network Architecture

The running models have been constructed with the simulation framework provided by SimPy [129], which has been previously validated by other researchers [130], so the results we obtain are reliable. This framework provides the ability to build discrete event simulators with the Python programming language. It is very clean and simple, so the important parts reside in the models not the framework.

Simulation Models

Before building the specific models for HIMALIS and our proposal (HIMALIS-CTL), we have defined a network model to get a proper and reliable view of the underlying network (e.g. the Internet). Each element comprising the model has been modeled as an object that can be instantiated and connected to other elements. Also, each element have parameters that determine its behavior so simulations can be more realistic (using real world parameters) or can be used to explore specific situations. This model has been used as basement to build the other models.

The model that represents the operation of the HIMALIS architecture as introduced in Section 6.2.1 has been simplified around the functionality provided by the HNR. In fact, it implements most functions of the architecture, so the resulting model is very simple and comparable to the proposed model without losing noticeable fidelity. Therefore, name resolution and security elements are left out of the model but they are not really necessary for the handover evaluation we propose, so the results we obtained are reliable and reproducible in real scenarios with very high probability. Moreover, the comparison of HIMALIS and HIMALIS-CTL will be more fair because only one element is involved in the control operations.

The model for HIMALIS-CTL has parallelism with the model designed for the plain HIMALIS. It includes the same elements than the plain HIMALIS model but redefines their implementation to delegate the functions to the controller and perform control communications through the identity-based control plane. Thus, the implementation of the identifier layer differs. While this model routes signaling operations to the controller via the control plane, the plain model routes signaling operations through the data plane of the network to the destination elements.

Finally, in order to use proper traffic in the simulated scenarios we have defined a set of applications (services and clients) that can be connected to the elements of any of the models. If an element implements the *host* interface, an application can be attached to it.

6. Integration with Other Network Architectures

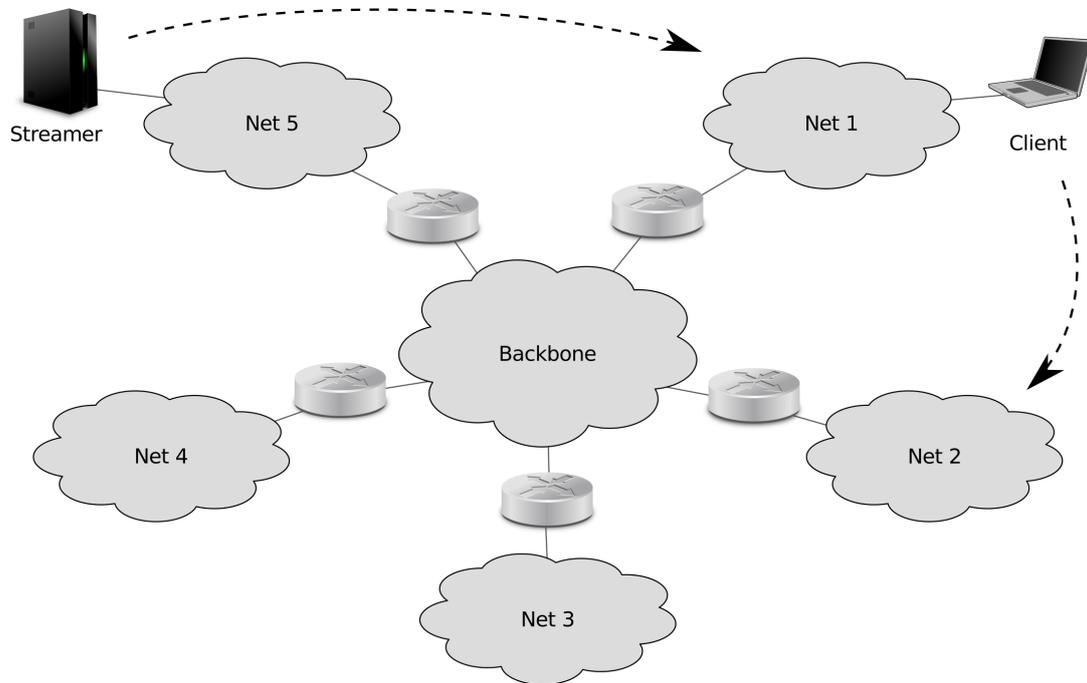


Figure 6.9: General topology of the evaluation network.

Network Topology

To stress the interesting aspects of both approaches during the evaluation of the proposed solution and thus demonstrate its behavior and benefits we have defined a common topology. Apart from using the specific implementation provided by the solution for the elements of the network, the only difference in the topologies is the addition of the specific elements defined by plain HIMALIS and HIMALIS-CTL.

The network topology, as shown in Figure 6.9, is composed of six different networks. Five of them are the edge (or access) networks while the remaining network represents the global interconnection network. The edge networks are attached to the interconnection network by five different gateways, so those edge networks have to cross the interconnection network in order to reach each other.

Moreover, two host elements with their corresponding application elements are instantiated. One tandem of application and host will be dedicated to the streaming server and another tandem will be dedicated to the streaming client. The target of the simulation will be to make the hosts to move to their adjacent networks, as depicted in

6.2 Integration with the HIMALIS Network Architecture

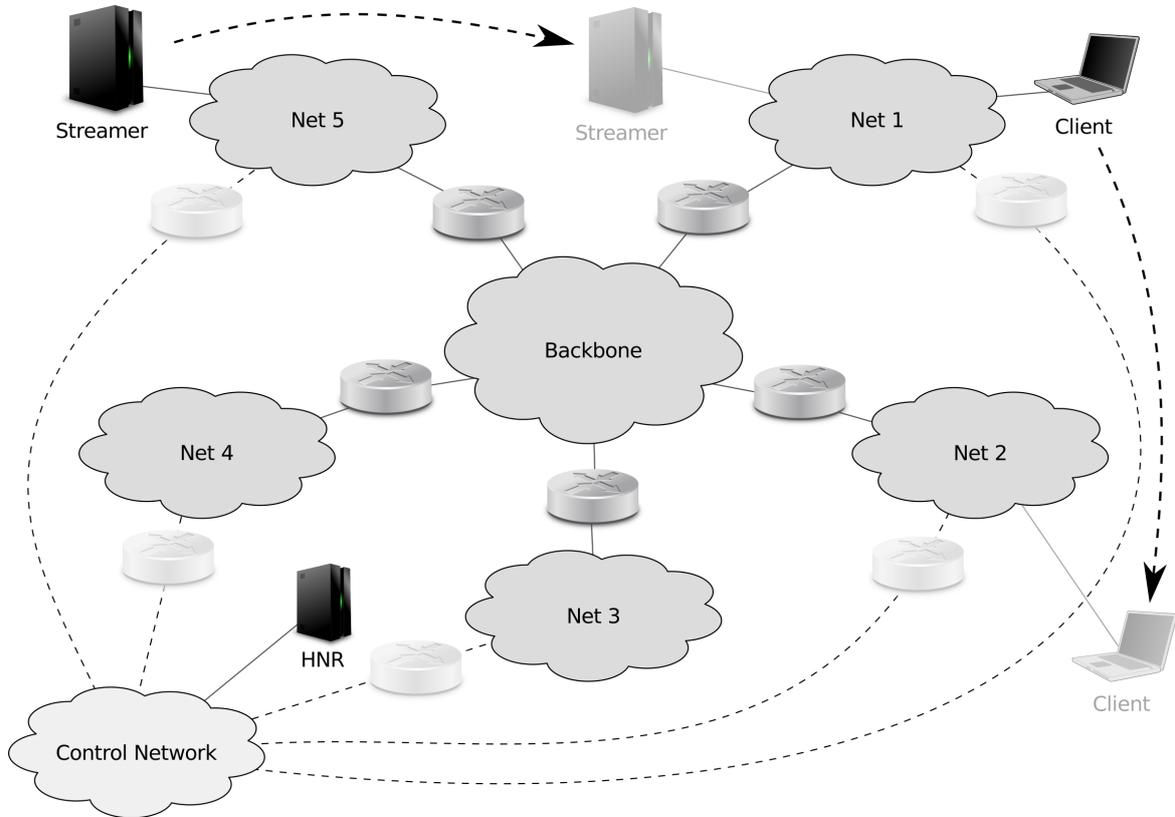


Figure 6.10: Network topology with the HIMALIS elements.

the aforementioned figure. As the general network model does not allow it, this operation will only be performed by the HIMALIS architecture and the approach proposed here.

As mentioned above, to build the topology for the simulations involving the evaluated architectures we need to include the associated elements that have been designed to support heterogeneous networks and mobility, as discussed in the previous section. The resulting topology is shown in Figures 6.10 and 6.11. They show the resulting environment after the introduction of the necessary elements. Also, the implementation of the gateways have been changed to use the one defined by the corresponding architecture. In addition, for the HIMALIS instance, a control network has been added and connected to all edge networks. This will have the instance of the HNR element. In contrast, HIMALIS-CTL will use the control plane.

Regarding the client and server, they have been instantiated from the *host* element included into the simulation models. Therefore, end-nodes (hosts) will have three layers: the application layer that has the same element included in the general model (without any variation), the HIMALIS layer that has the HIMALIS or HIMALIS-CTL host element,

6. Integration with Other Network Architectures

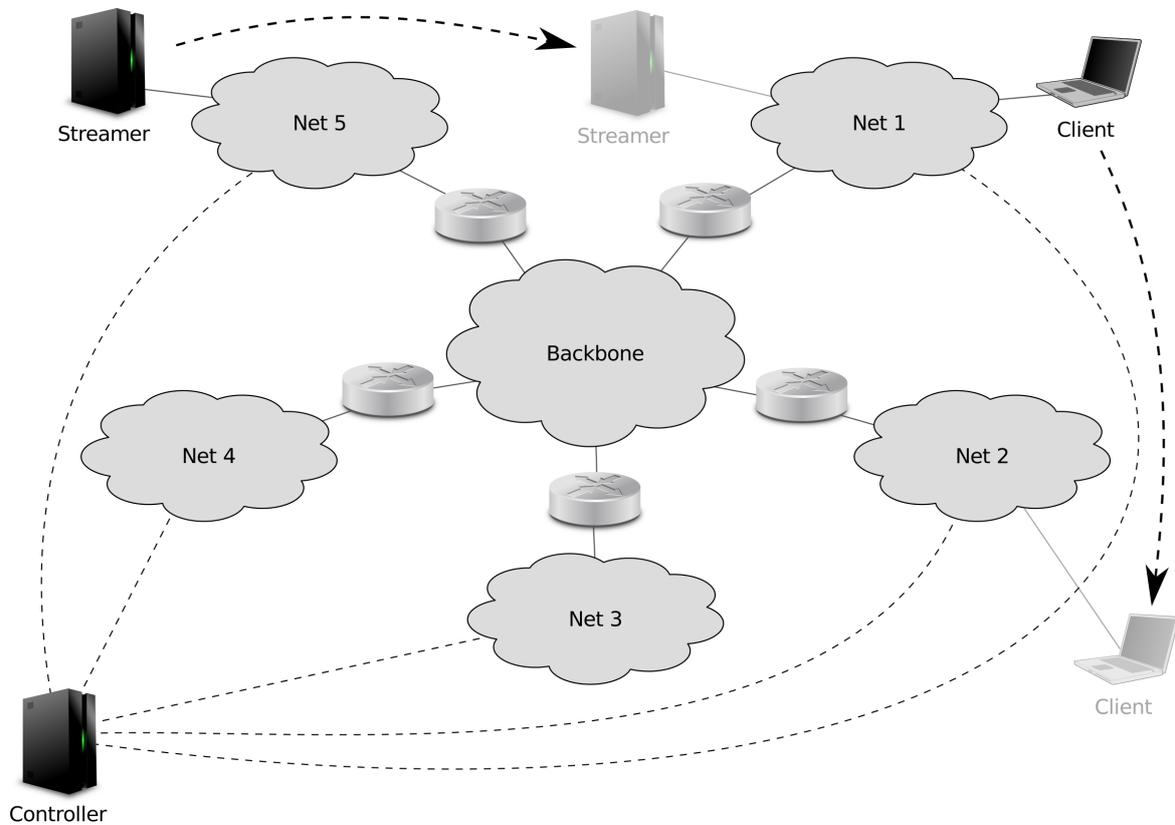


Figure 6.11: Network topology with the controller.

Table 6.1: Simulation parameter values.

Element	Parameter	Value
Backbone Network	latency	0.050 s
Backbone Network	bandwidth	10000 msg/s
Backbone Network	queue size	2500 msgs
Access Networks	latency	0.015 s
Access Networks	bandwidth	4000 msg/s
Access Networks	queue size	1000 msgs
Gateways	processing time	0.001 s
Gateways	queue size	100 msgs
Hosts	rate	4000 msg/s
Hosts	queue size	1000 msgs
Configuration Service	processing time	1.5 s

and the underlying layer that has the plain host and contains the HIMALIS host element. Table 6.1 shows the parameters used for each element of the model to execute the simulation.

Simulation Scenarios

The simulation scenarios are built by instantiating a streaming server in an access network and a client in each of the remaining access networks. Once instantiated, the clients will request the content and start receiving frames from the server. Then, the clients will move to the clockwise adjacent network and wait for some time before continuing to the next one.

The first scenario, scenario A, consists of instantiating only one streamer server with a rate of 1000 msg/s and only one client. Once instantiated, the client will request the content and start receiving frames from the server. Then, the client will move to the clockwise adjacent network and wait for some time before doing it again. In total, the client will complete ten cycles, which results in a total of fifty handover operations for the client. Finally, when the client has completed all its work, the server will do the same and start moving to its adjacent network to complete other ten cycles.

The second scenario, scenario B, consists of the same instantiation and general work details as described for the first scenario but, in this case, a new element is instantiated to add background traffic to all networks. On the one hand, there are four elements to generate background traffic and are instantiated in different edge networks. On the other hand, there is an echo service instantiated in the remaining network. They are static (do not move) so their behavior is consistent.

The third scenario, scenario C, also consists of the same instantiation and general work details as described for the first scenario. However, this scenario introduces three more clients that request the content from the server. All end-hosts, clients and server, are spread across all edge networks and will perform handover operations in the way described for the first scenario. Thus, there will be some time when two clients are connected to the same network, which means that we have slightly different situations taken into account in the evaluation.

In total, each client will complete ten cycles, which results in a total of fifty handover operations per client. Finally, when the clients have completed their cycles, the server will do the same and move to its clockwise adjacent network to complete ten cycles. It is important to notice that this evaluation considers different cases, for example the case in which two clients are connected to the same network for some time.

Moreover, the scenario has two different execution modes: *hard handover* and *soft handover*. On the former, hosts will first break their current links (disconnect their interfaces) and then establish them again (connect those interfaces to a new network).

6. Integration with Other Network Architectures

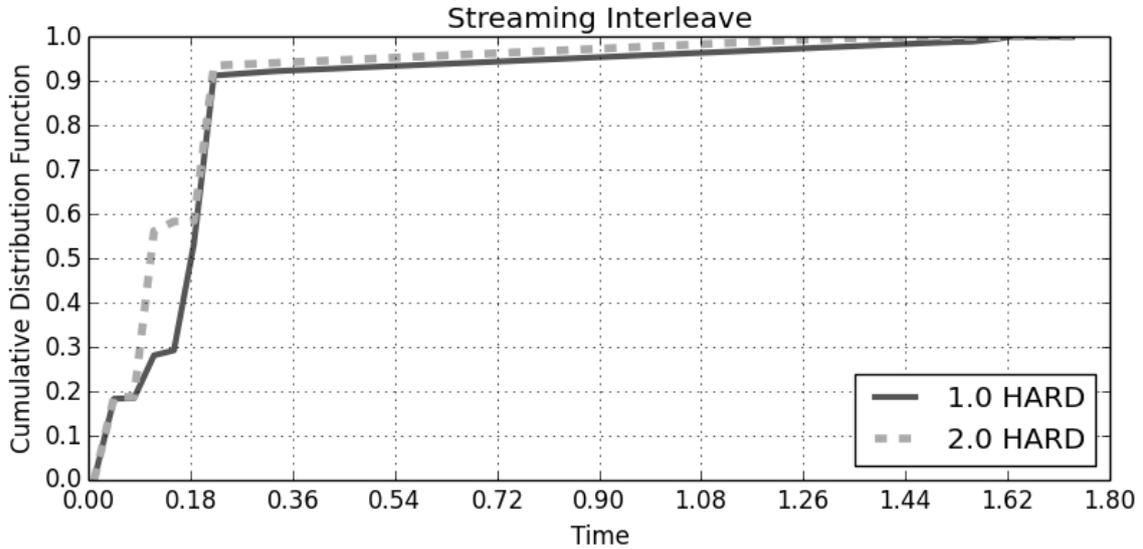


Figure 6.12: Streaming interleave time for scenario A, hard handover.

On the latter, hosts will instantiate and connect a new interface to a new network before disconnecting and destroying the interface that connects it to the previous network.

Results

From each execution of the models we have instructed the simulator to take handover measurements. Client and server handover operations have been differentiated. The simulator takes the measurements from the time (simulation tick) when a host is ordered to “move to the network X” until the time the host receives the final control message confirming that the operation has finished. The time dedicated to the configuration of the host into a new network has been subtracted, so we get only the time introduced by the architecture. Then we have built the corresponding Cumulative Distribution Function (CDF) diagrams that represent with high confidence the comparison between the two approaches.

From the evaluation, frames are supposed to be received at exactly one per second, the separation between them should be 1.0 s. However, the network operation introduces some delay, especially during handovers. Figures 6.12 and 6.13 show the CDF of such delay without taking into account the lost frames, which is a good marker of the quality of the protocol and architecture operations. We can appreciate in them that the proposed integration improves the basic architecture both for hard and soft handovers. For hard handovers, the proposal reduces the delay of more than 50% of frames to 160 ms from the

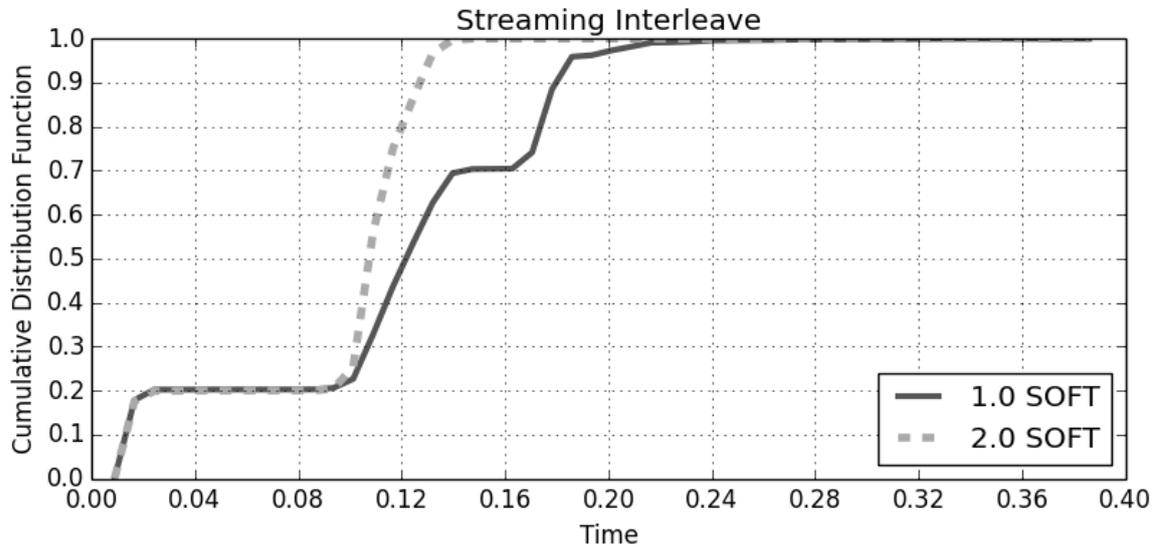


Figure 6.13: Streaming Interleave time for scenario A, soft handover.

180 ms took by the original approach. For soft handover, the proposed integration makes the result much more stable, with 80% operations lasting 120 ms or less while the original approach still requires 180 ms.

As depicted in Figures 6.14 and 6.15, we see that the behavior of the integrated approach (referenced as "2.0") outperforms the original HIMALIS architecture (referenced as "1.0") by a considerable amount of time. More than 90% of client handover operations have required less than 70 ms with the proposed improvements while the original approach requires more than 150 ms for the 80% of operations. In the case of servers, the proposed approach requires less than 100 ms for more than 90% of handover operations while the original approach requires more than 200 ms for more than 90% of handover operations. Clients require less time to complete a handover because servers and their networks are more busy processing the data messages they send, which increase with the number of clients.

The same overall behavior is also shown in scenarios B and C, as depicted in Figures 6.16, 6.17, 6.18, and 6.19. That said, there are 20% of operations for HIMALIS that outperform HIMALIS-CTL. This is because there is a 20% of occasions when the client and server are in the same network, so the handover is sped up. However, HIMALIS-CTL requires almost the same time for all operations to complete because, even though the operation is more complex, it requires less message exchanges between hosts and the control element but all handover operations must control such control element. This is not the

6. Integration with Other Network Architectures

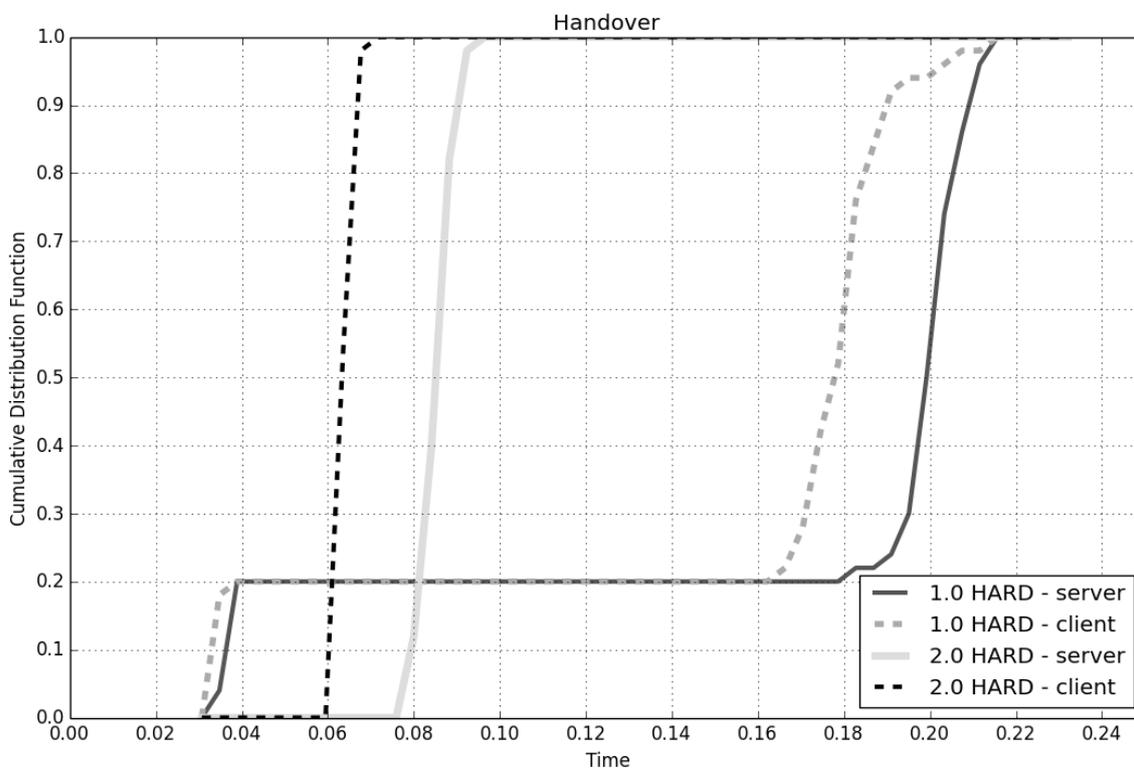


Figure 6.14: Handover time for scenario A, hard handover.

case of the servers because, in the case of plain HIMALIS, they have to perform much more message exchanges than the clients.

As a final remark, HIMALIS-CTL requires an average of 0.06-0.07 seconds (60 to 70 ms) to complete while the best case for the plain HIMALIS architecture require an average from 0.17 to 0.27 seconds (170 to 270 ms). This means that moving the HIMALIS control messages to a separate control plane can really improve handover operations, reducing the time required by the signaling to almost a third of the current required time. This is an important result but it has to be proven also in experimental scenarios with real code, as we discuss in the following section.

6.2.7 Experimentation Framework and Testbed

To experiment with the functionalities defined in previous section, the HIMALIS architecture is accompanied with an experimentation framework that instantiates each component into a deployable software piece. In order to facilitate the deployment, the experimentation framework aggregates the DNR, HNR, and IDR functionalities into a

6.2 Integration with the HIMALIS Network Architecture

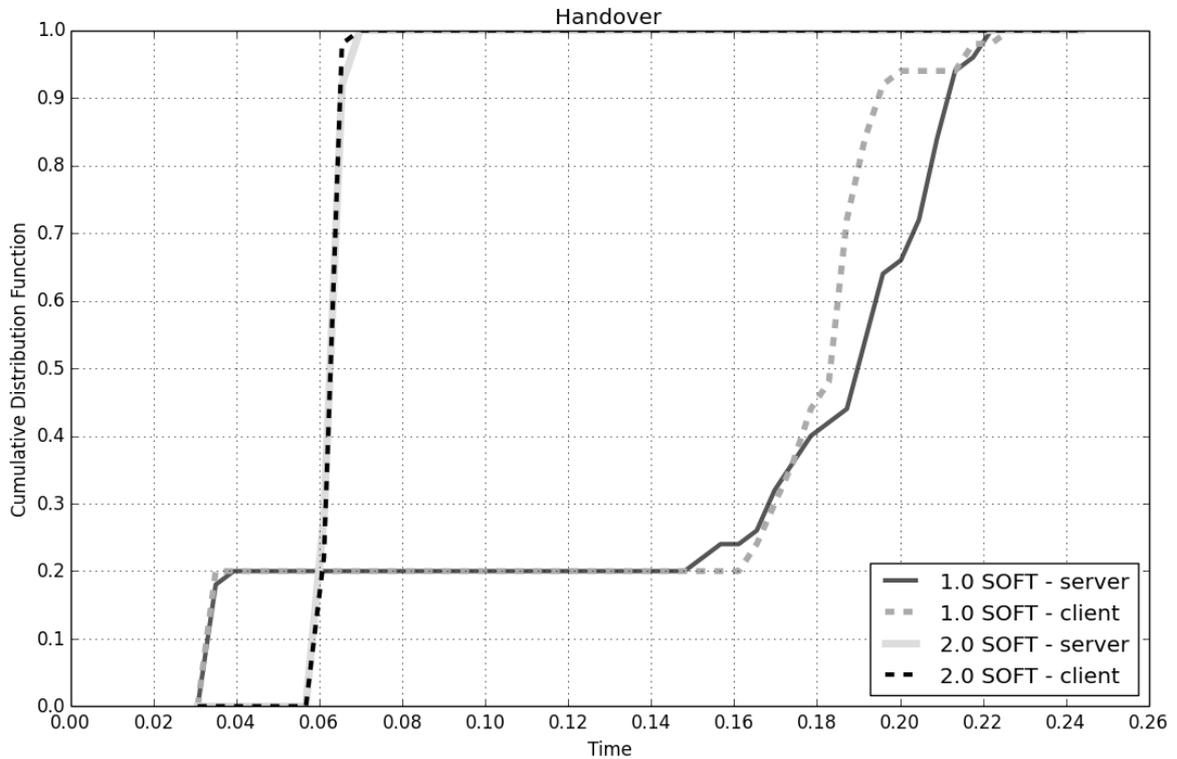


Figure 6.15: Handover time for scenario A, soft handover.

single software component that should be configured to differentiate the final role of the deployed element. The gateway (GW) component is implemented as a different software piece, so it is deployed separately. Finally, the end-host functionality is implemented as a different software component that should be deployed in the end-hosts in order to let them interact with the infrastructure elements.

Using all software components described above, we deployed a simple testbed, shown in Figure 6.20, that lets us experiment the behavior of the proposed HIMALIS architecture. The three infrastructure elements (GWs) are interconnected through an IPv4 network which are not equal but instantiated with different infrastructure software. The common components of all elements are the HNR and the GW. Both edge network 1 (NET 1) and edge network 2 (NET 2) use IPv4, while the edge network 3 (NET 3) uses IPv6. To build the experimentation testbed we need to follow the following steps.

First we need to prepare the equipment of the testbed. In this case, we have three servers with two network interfaces, one is connected to a switch that represents the global transit network while the other is connected to other separate switches that represent each

6. Integration with Other Network Architectures

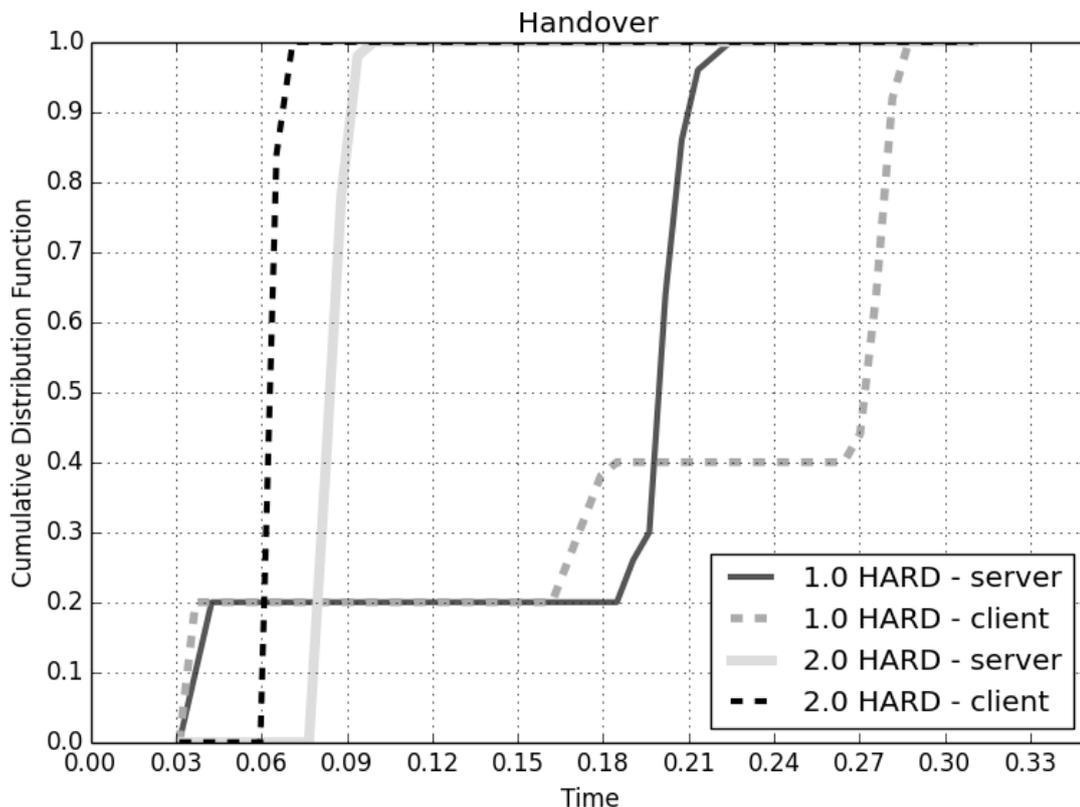


Figure 6.16: Handover time for scenario B, hard handover.

edge network. The end-hosts are two laptops respectively connected to network 1 (the switch connected to server 1) and network 3 (the switch connected to server 3).

Once the physical elements (equipment and cabling) are prepared, we configure the logical elements and underlying network services. Each server has a Dynamic Host Configuration Protocol (DHCP) service. The servers of edge network 1 and 2 assign IPv4 addresses while the DHCP service of edge network 3 assigns IPv6 addresses in conjunction with the Router Advertisement Daemon (radvd) service. With these configurations the three servers are correctly configured as gateways for IPv4/IPv6 underlying networks and then we proceed with the instantiation of the HIMALIS components.

The experimentation framework of HIMALIS includes an installer script that automates the installation of the components. It accepts a parameter to set the type of component to install, so we need to copy the framework code to the all machines and then run the installer script with the corresponding parameters to the role of the machine. As the three infrastructure elements include a GW and at least one of the DNR, HNR and IDR, we

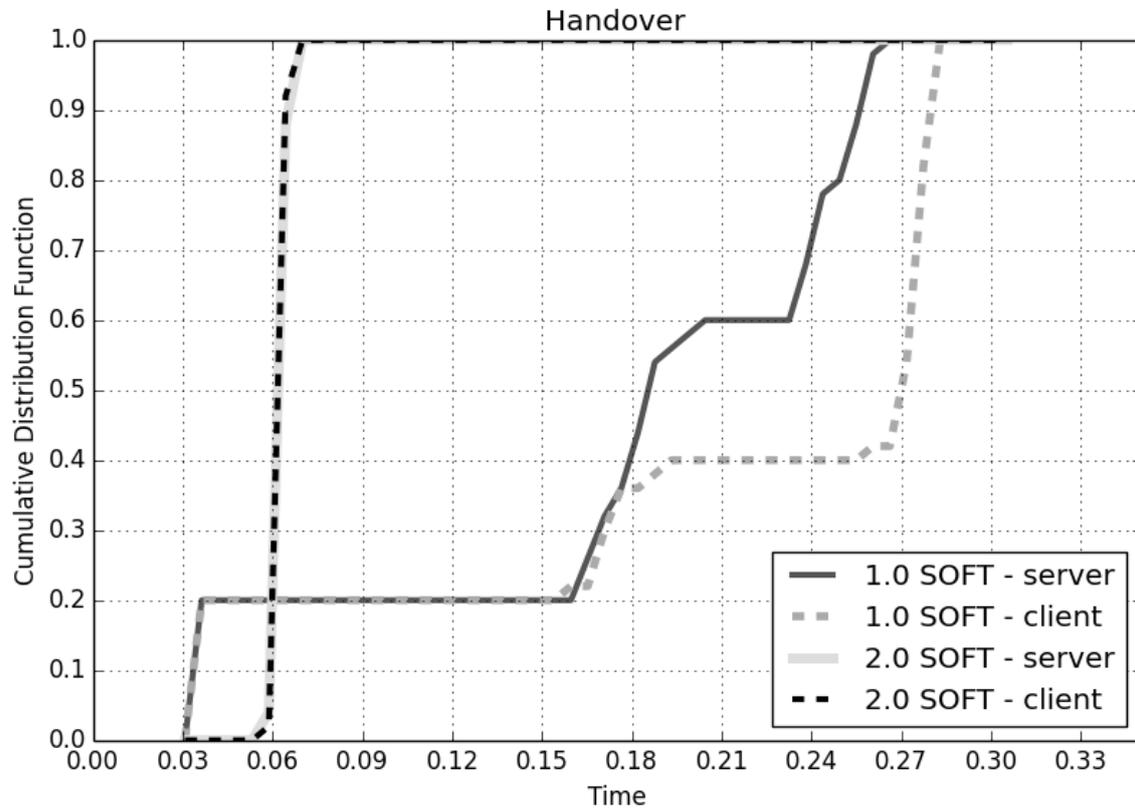


Figure 6.17: Handover time for scenario B, soft handover.

set the installer to install both GW and HNRDNR components to all servers. For the end-hosts, we set the installer to install just that, the host component. Now we proceed to configure the necessary software component of each element as seen in the aforementioned figure that represents the testbed, that is: The server of network 1 has HNR, GW, DNR, and IDR; the server of network 2 has HNR, GW, and viewer; the server of network 3 has HNR, GW; and the hosts has the HOST component.

As each component offers a configuration web interface, to configure the DNR we point a web browser to the appropriate URL and set the domain name managed by that DNR (east.idl.nict.go.jp, center.idl.nict.go.jp, and west.idl.nict.go.jp) and both the unique identifier and IP address of the HNR to which is associated. Then, we need to tell the DNR how to find the other domains so, for each server, we point the web browser to the appropriate URL and set the domain name and IP address of the other DNRs. With this step we have the DNRs configured knowing how to find each other and how to find their corresponding HNR.

6. Integration with Other Network Architectures

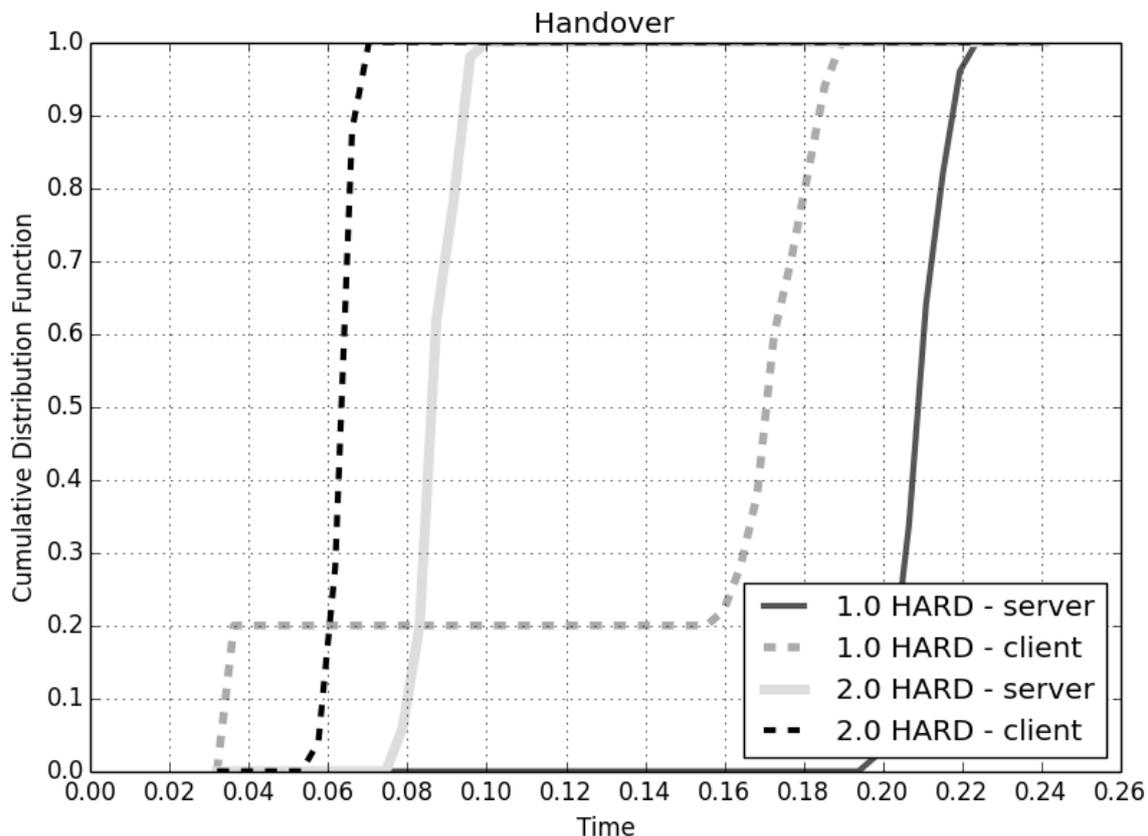


Figure 6.18: Handover time for scenario C, hard handover.

Having configured the DNRs, we continue with the configuration of the GWs so, for each GW, we point the web browser to the appropriate configuration URL and set the corresponding URL of the DNR associated with the GW, the inner network interface used to contact with edge network hosts, and the outer network interface used to reach other domains. At the same time, this configuration is used to inform the IDR elements how to find each DNR.

Finally, we run the client application in end-hosts. It launches a web browser to register the host into the HNR, so we should indicate the host name, its domain name, and its HNR IP address. The HNR will register the host and tell the further association between host name and ID to the DNR. With this step we finished the set-up of the experimentation environment.

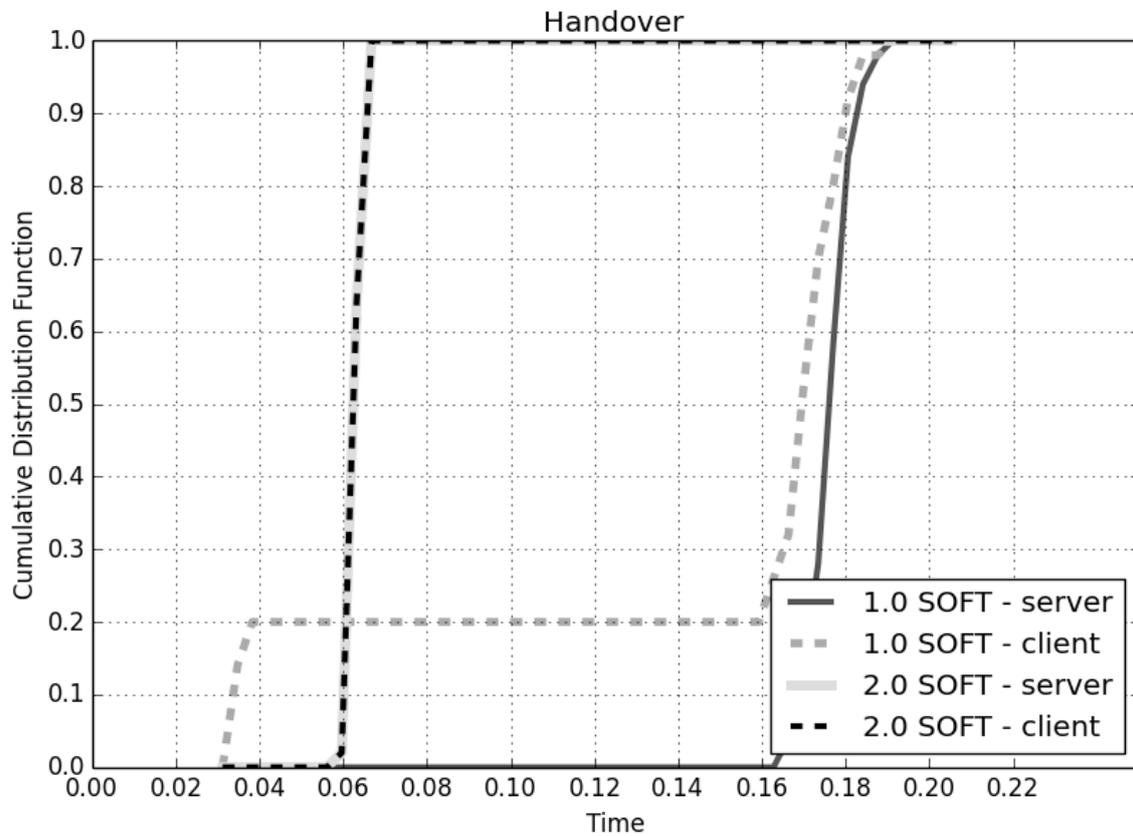


Figure 6.19: Handover time for scenario C, soft handover.

Experimentation Results

The primary purpose of the testbed deployment described in previous section is to experiment with the integration of heterogeneous networks and the handover among them, and thus show the behavior of the HIMALIS architecture in such scenarios.

Using the client application installed in end-hosts we can start a session between them and then transfer a looped video stream from one to the other. This lets us observe how network traffic is exchanged in terms of host name and based on identifiers, which then are resolved into addresses.

After starting the video session between the two end-hosts we can disconnect one of them (Host 3) from its network (NET 3) and then connect it to other network (NET 2). Then, after the DHCP service of the new network gives a new IP address to the host, we can see how the video session continues, so the time to handover is reduced just to 200 ms, apart from the time needed to obtain the underlying network address. This is because the

6. Integration with Other Network Architectures

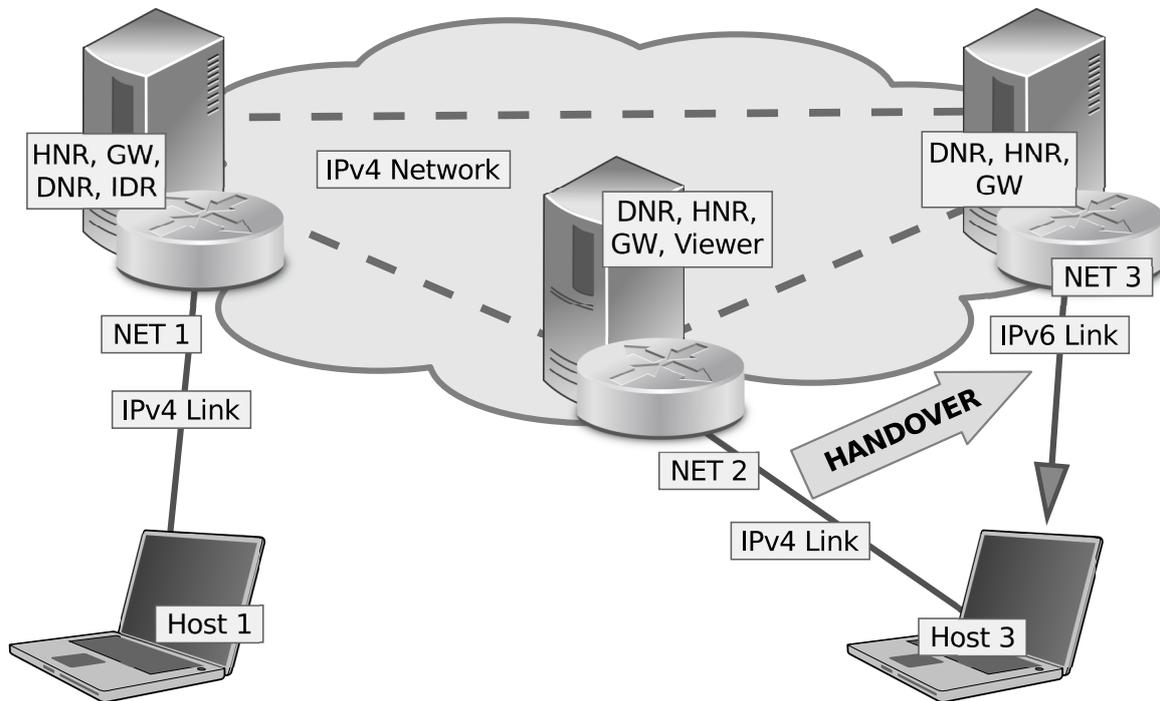


Figure 6.20: Experimentation testbed.

ID/Locator's Session Information






Layers	Parameters								
Application	Global Hostname	host1#east.idl.nict.go.jp				host3#west.idl.nict.go.jp			
Transport	ID	db73:dbef:642d:9529:af60:ad22:196d:e9f2				8906:168f:c1c1:771b:7272:b2c0:d1c6:5283			
Identity	ID	db73:dbef:642d:9529:af60:ad22:196d:e9f2				8906:168f:c1c1:771b:7272:b2c0:d1c6:5283			
Network	Locator	192.168.14.4	192.168.14.1	192.168.1.124	192.168.1.123	2001:db8:1:400::1	2001:db8:1:400:a434:417c:a572:37bd		

Figure 6.21: Session information obtained from the viewer component.

client component receives the connection events and instantly performs the binding update in the corresponding infrastructure elements, without needing to deal with home-agents or something like that. The time spent in the DHCP negotiation should be not considered because in a real deployment of the architecture there will be other quicker methods to obtain underlying locators (addresses).

The handover operation also changed the underlying protocol from IPv4 to IPv6, so we can appreciate that the architecture correctly translates from different underlying

6.3 Integration with the MOFI Network Architecture

protocols. To check this we can use the viewer component installed with the DNR and HNR and obtain the session information. We configured the server 2 to run the viewer, which contacts with the other servers to obtain that information. As we can see in Figure 6.21, which represents the session information after the Host 3 is back to its network, the Host 1 uses an IPv4 locator while the Host 3 uses an IPv6 locator. Also, the infrastructure element of NET 1 needs to use an IPv4 locator in its edge network and the infrastructure element of NET 3 needs to use an IPv6 locator in its edge network. Both infrastructure elements use IPv4 for the infrastructure network (the global transit network). Therefore, we can see how the different network protocols can coexist with the HIMALIS architecture, the only requirement is that all infrastructure elements must share the same underlying network (again, the global transit network).

6.3 Integration with the MOFI Network Architecture

With the wide popularity of smart phones and general mobile devices (sensor networks), the Internet has rapidly changed from fixed-based to mobile-based. This trend is going to dominate the future, with more than 1.6 billion of 2014, exceeding the number of desktop users [131]. The current Internet model, and its patch-on extensions, is fixed-host centric, so the mobile-based environment is a primary factor to be incorporated to the FI.

Being tightly integrated to the current Internet, the Mobile IP (MIP) [132] and its variants does not fit with this mobile-centric view because they still rely on a fixed-host environment. However, as introduced in Section 2.1, some approaches try to mitigate this problem by really providing the separation of identifiers and locators. Among them we highlight HIP and LISP, which also provide some handover mechanisms to support mobility but they still consider the network nodes as fixed hosts. These limitations have caused the appearance of the *clean-slate* approaches we have discussed throughout this thesis. However, they also lack in the specific consideration of mobile nodes by default and in the definition of security and privacy protection, which is starting to be addressed in [120].

On the contrary, as deeply discussed below, the architecture proposed by MOFI [133] is centered around mobile devices, considering mobility the default behavior of the FI. It is primarily built with three functional blocks: 1) host ID and network LOC, 2) global ID-based communication with local LOC-based delivery in the data plane, and 3) search-based distributed mobility control in the control plane. With them, it builds a architecture with complete separation of control and data planes and integrated mobility

6. Integration with Other Network Architectures

and multihoming support. The main limitation we find in this approach is the lack of security. In this chapter, we discuss how to enhance this architecture by the introduction of the secure identification functional block (SEID) that enhances MOFI with identity-based negotiations of networking security aspects, primarily the identification of communication participants (entities), as we addressed in [105].

6.3.1 Architecture Overview

As introduced above, MOFI is designed with three functional blocks. First, the ID-LOC separation with Host Identifier and Network Locator (HINLO) [134] functional block meets with the separation of host identifier from network locator, an essential requirement for FI. In the data plane, the packet delivery is accomplished by Global ID-based communication and Local LOC-based delivery (GIC-LLD) which separates the delivery mechanism of access network (AN) and backbone network. For mobility control, MOFI uses the Search-based Distributed Mobility Control (SDMC) [16] in the control plane, which meets with the FI requirements of separation of control and data delivery functions and the distribution of mobility control.

MOFI basically assumes the host-based communication that is already adopted in current Internet. The Host Identifier and Network Locator (HINLO) functional block is used to support ID/LOC split, in which independent host ID (HID) is employed and separated from locator (LOC).

The mobility control is performed by the Search-based Distributed Mobility Control (SDMC) functional block, which provides search-based and distributed signaling operations based on the distributed mobility management presented in [135]. It naturally implies the separation of control plane from data plane. This search-based mobility control may induce a certain amount of delay for LOC query signaling. However, such an initial signaling delay may be negligible in the session setup phase. In case of handover, the LOC binding will be updated directly between the two concerned ARs for route optimization.

The Global ID Communications and Local LOC Delivery (GIC-LLD) functional block, as depicted in Figure 6.22, provides the possibility for each host to have a global HID with several LOCs used for packet delivery in each network. Each LOC is used locally in the networks, without any assumption on global uniqueness. In GIC-LLD, the packet delivery operations have two stages, i.e. access networks and backbone. On it, communication between two hosts is performed with HIDs, whereas LOCs are used for packet delivery inside the Access Network (AN) and through the Internet backbone. The SDMC control operation is used to find appropriate LOC for each HID in each network.

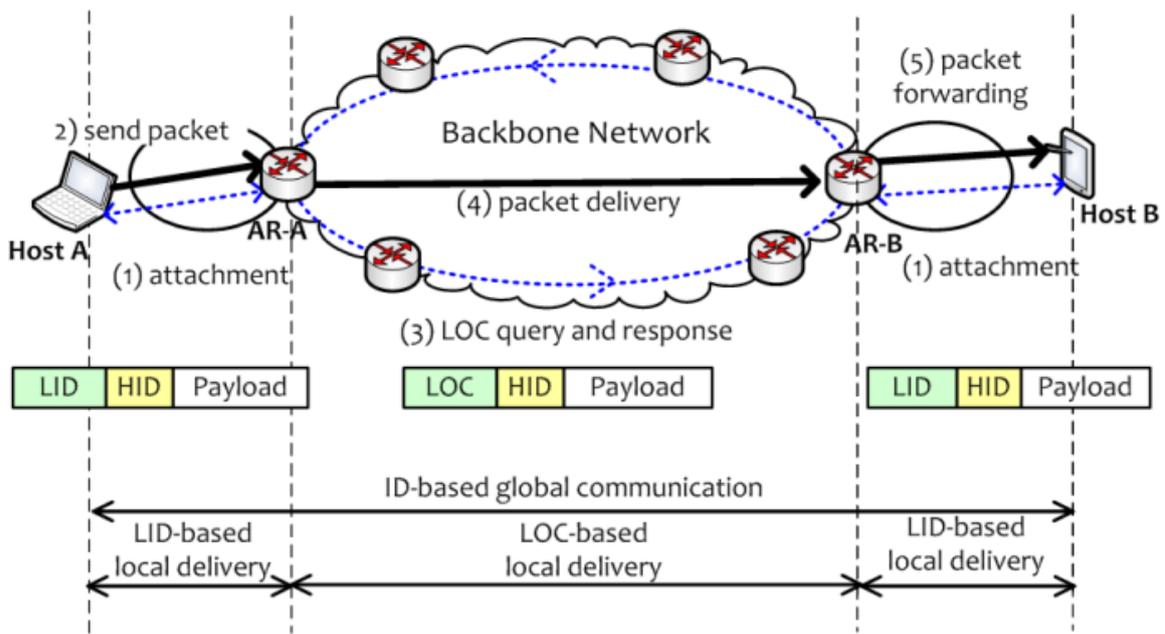


Figure 6.22: Overview of global/local communications in MOFI.

6.3.2 Secure Identification (SEID) over MOFI

Due to the security limitations we find in MOFI we propose to integrate it a new functional block, the secure identification (SEID). It is responsible to securely mediate in the resolution of HIDs from host names (or identities) and the resolution of LOCs from HIDs. With it, networked entities can be securely identified by means of their digital identity but, if specified in their security policies, entities can communicate without revealing any information about their true identity, not even their activities, so they cannot be traceable. This is the normal operation of SEID, so even when communications are not encrypted, it prevents the traceability and provides attribute-based negotiation of sessions to enhance privacy. Also, when desired, an entity may change its HID to prevent session linkability.

The basic mechanism behind SEID is the identity plane, which is provided by the architecture proposed in this thesis. To integrate SEID with the other functional blocks defined by MOFI we chose the DTEi, presented in Chapter 4, as the interconnection point. As shown in Figure 6.23, the network objects connected to a MOFI infrastructure will use the control plane, IBCP, to reach the DTEi for negotiating sessions under the terms of their identities. This instantiation of IBCP can also be used by other elements of the network,

6. Integration with Other Network Architectures

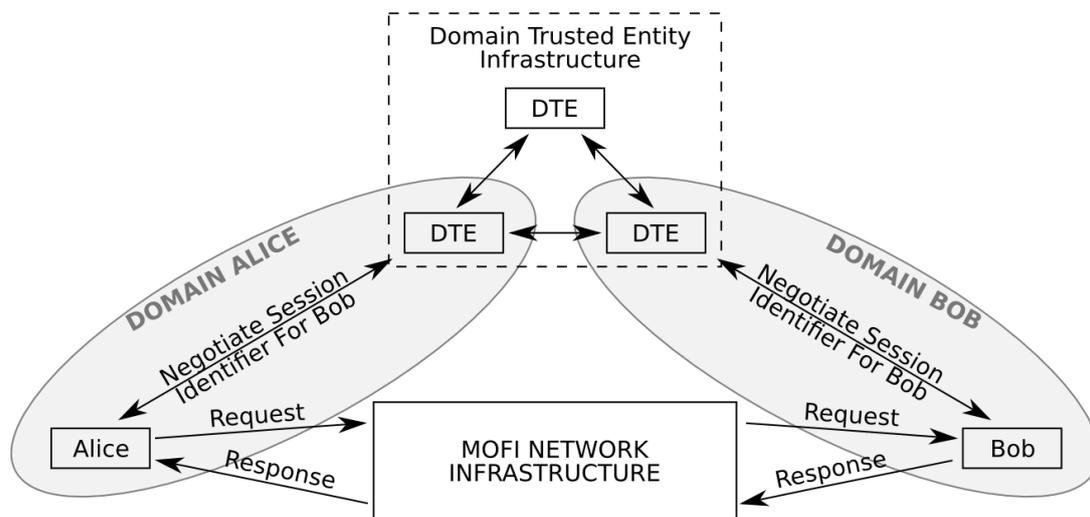


Figure 6.23: Instantiation of our architecture for secure identification.

such as other participants of communications or even infrastructure elements that want to be somehow involved in the session, ensuring security and protecting their identities.

Our architecture proposes to deploy a different intermediate element (node of the DTEi/ODIN) into each network/administrative domain. This fits with the design of MOFI, which places specific control functions in access routers (gateways). In our case, the intermediate node of our architecture that is instantiated in one domain will be in contact with the access router of such domain. This gives MOFI a full coverage of the identity space in a distributed manner to allow its endpoints to securely and privately manage the identity information of the (several) entities taking part of communications.

As introduced above, the functionality provided by our architecture, especially the functions provided by IBCP, is then used to negotiate other aspects of the network, such as the permissions of a communications to cross a network, the establishment of security associations, etc. Figure 6.24 shows the position of the DTEi in relation to MOFI as well as its interactions. On the left it depicts how the DTEi is formed from different elements connected in a overlay ring to form the infrastructure and indicates how to connect it with other MOFI. On the right side it shows the adaptation of IBNP, the identity-based network protocol, to define basic message exchanges used in security negotiations within MOFI. Below we describe the most important aspects of this integration.

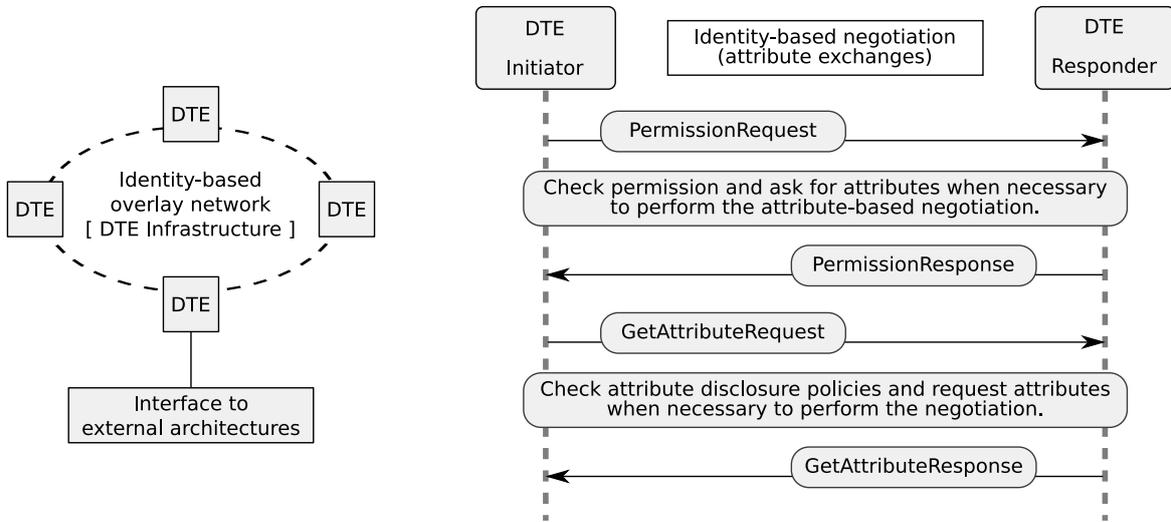


Figure 6.24: Overview of the integrated DTEi and message exchanges.

Overlay Network

As commented throughout this document, our architecture is highly based on overlay networks, especially the DTEi and ODIN. This way, the whole infrastructure is totally distributed and the instances can reach each other without even knowing their underlying identifiers or addresses. There are many ways to build the overlay network, we decided to use a similar routing algorithm to the one found in Chord [35], because its simplicity, and improved with LPRS [36] and Kademia [41].

Operations

The main operations supported by our architecture are, as commented above, the identity-based negotiations provided by IBNP and carried out through IBCP. When instantiating the architecture within MOFI, what we called SEID, we support two types of negotiations: the permission request to access a resource and the attribute request to obtain the value of an attribute from an entity. Both operations can use the attribute query when required by policies, that is, attribute requests from a responder to an initiator can be used anytime during a negotiation. The permission request operation is used to perform an attribute-based authorization procedure and get access to a resource that is represented by an identity while the attribute request is used to perform a plain attribute-based negotiation.

6. Integration with Other Network Architectures

Authentication and Security

The interactions between the intermediate elements of our architecture and the elements defined by MOFI are delicate, so they must be conducted in a secure way and under mutual authentication of the parties taking part of the negotiation. This allows communication participants to be sure that those parties are who claim to be and no other entity can see any exchange of their negotiations.

In order to achieve the mutual authentication in a fast and reliable way we propose to use only one exchange (round-trip) and be based on digital (public key) certificates of the intermediate elements that communicate, specifically the nodes of the DTEi and ODIN. To be sure this is performed in a trusted manner, we propose to trust only in the entities that hold the private key of certificates issued/signed by a trusted certification authority (CA), like in X.509 used by Private Key Infrastructures (PKIs), or certificates signed by itself or by N trusted entities, like in the Web of Trust concept used by PGP. The two models denote the centralized and decentralized (distributed) trust models.

6.3.3 Analysis

To analyze the performance of the Secure Identification (SEID) functional block, we compare the behavior of the plain MOFI architecture with SEID over MOFI. In order to facilitate the analysis we build a mathematical model of both approaches. As the mobility is an essential part of the architecture, the model includes the binding update cost (BUC) together with the packet delivery cost (PDC), so the total cost (TC) is represented as $TC = BUC + PDC$. The importance of the comparison is because, in SEID over MOFI, the ARs need to also update the intermediate elements defined by the control plane defined by our architecture (IBCP), which here are mainly represented by the DTEi. This way, so the BUC of the SEID approach is slightly increased. It adds a new control exchange that logarithmically depends on the number of ARs deployed in the network.

To build the equations that model the communication cost of MOFI and MOFI+SEID, we use the following parameters: 1) P_k is the processing cost for binding update or location lookup at node k , which is proportional to the number of active hosts in the domain. This can apply to P_{AR} and P_{HA} ; 2) T_{a-b} is the transmission cost of a packet between two nodes (a and b) in the network, which is proportional to the number of hops between the concerned two nodes. This can apply to T_{MN-AR} , T_{AR-HA} , and T_{AR-AR} ; 3) α is the unit cost of packet transmission per hop; 4) β is the unit cost of location binding or lookup per database entry; 5) H_{a-b} is the hop count between nodes a and b in the network; 6) N_{AR} is the number of

6.3 Integration with the MOFI Network Architecture

ARs in the network; 7) $N_{\text{Host/AR}}$ is the number of hosts per AR; 8) S_{control} is the size of a control packet; 9) Finally, S_{data} is the size of a data packet.

The SEID functional block differentiates between host DTEi instance and visited DTEi instance. So, there are two different variables to measure: local BUC and remote BUC, whose model equations are as follows:

$$\begin{aligned}
 BUC_{\text{MOFI+SEID-local}} &= \\
 &= S_{\text{control}} \times 2T_{\text{MN-AR}} + S_{\text{control}} \times 2T_{\text{AR-DTE}} + P_{\text{DTE}} \\
 &= S_{\text{control}} \times 2\alpha(H_{\text{MN-AR}} + H_{\text{AR-DTE}}) + \beta(N_{\text{Host/AR}}).
 \end{aligned} \tag{6.1}$$

$$\begin{aligned}
 BUC_{\text{MOFI+SEID-remote}} &= \\
 &= S_{\text{control}} \times 2T_{\text{MN-AR}} + S_{\text{control}} \times 2T_{\text{AR-DTE}} + S_{\text{control}} \times 4T_{\text{DTE-DTE}} + 2P_{\text{DTE}} \\
 &= S_{\text{control}} \times 2\alpha(H_{\text{MN-AR}} + H_{\text{AR-DTE}} + 2[H_{\text{DTE-DTE}} \times \log_2(H_{\text{AR}})]) + 2\beta(N_{\text{Host/AR}}).
 \end{aligned} \tag{6.2}$$

The packet delivery cost of SEID over MOFI also involves the communication through IBCP, which requires to cross the DTEi in order to perform the attribute-based negotiation, which is in turn necessary to enhance privacy and security requirements to the communication, as well as to reveal the actual locator (LOC) of a node. It also differentiates between local and remote operations. Thus, the PDC can be obtained as follows:

$$\begin{aligned}
 PDC_{\text{MOFI+SEID-local}} &= \\
 &= S_{\text{data}} \times T_{\text{CN-AR}} + S_{\text{control}} \times 2T_{\text{AR-DTE}} + S_{\text{control}} \times 4T_{\text{DTE-DTE}} + 2P_{\text{DTE}} + \\
 &+ S_{\text{control}} \times 2T_{\text{AR-DTE}} + S_{\text{data}} \times T_{\text{AR-AR}} + S_{\text{data}} \times T_{\text{MN-AR}} \\
 &= S_{\text{data}} \times \alpha H_{\text{CN-AR}} + S_{\text{control}} \times 2\alpha H_{\text{AR-DTE}} + S_{\text{control}} \times 4\alpha H_{\text{DTE-DTE}} \times \log_2(N_{\text{AR}}) + \\
 &+ 2\beta(N_{\text{Host/AR}}) + S_{\text{control}} \times 2\alpha H_{\text{AR-DTE}} + S_{\text{data}} \times \alpha H_{\text{AR-AR}} + S_{\text{data}} \times \alpha H_{\text{MN-AR}}.
 \end{aligned} \tag{6.3}$$

6. Integration with Other Network Architectures

$$\begin{aligned}
PDC_{\text{MOFI+SEID-remote}} &= \\
&= S_{\text{data}} \times T_{\text{CN-AR}} + S_{\text{control}} \times 2T_{\text{AR-DTE}} + S_{\text{control}} \times 4T_{\text{DTE-DTE}} + 2P_{\text{DTE}} + \\
&+ S_{\text{control}} \times 2T_{\text{DTE-DTE}} + P_{\text{DTE}} + S_{\text{control}} \times 2T_{\text{AR-DTE}} + S_{\text{data}} \times T_{\text{AR-AR}} + S_{\text{data}} \times T_{\text{MN-AR}} \\
&= S_{\text{data}} \times \alpha H_{\text{CN-AR}} + S_{\text{control}} \times 2\alpha H_{\text{AR-DTE}} + S_{\text{control}} \times 4\alpha H_{\text{DTE-DTE}} \times \log_2(N_{\text{AR}}) + \\
&+ 2\beta(N_{\text{Host/AR}}) + S_{\text{control}} \times 2\alpha H_{\text{DTE-DTE}} \times \log_2(N_{\text{AR}}) + \beta(N_{\text{Host/AR}}) + \\
&+ S_{\text{control}} \times 2\alpha H_{\text{AR-DTE}} + S_{\text{data}} \times \alpha H_{\text{AR-AR}} + S_{\text{data}} \times \alpha H_{\text{MN-AR}}.
\end{aligned} \tag{6.4}$$

Based on the above model equations, we have two different variables for the total cost of the security enhancements (SEID) of MOFI, they are:

$$\begin{aligned}
TC_{\text{MOFI+SEID-local}} &= BUC_{\text{MOFI+SEID-local}} + PDC_{\text{MOFI+SEID-local}} \\
TC_{\text{MOFI+SEID-remote}} &= BUC_{\text{MOFI+SEID-remote}} + PDC_{\text{MOFI+SEID-remote}}
\end{aligned} \tag{6.5}$$

Once the above model equations are formulated, we set the parameters to typical values, as used in [134], which are as follows: $N_{\text{Host/AR}} = 100$ (range from 10 to 1000), $N_{\text{AR}} = 10$ (range from 10 to 100), $H_{\text{AR-AR}} = 3$, $H_{\text{DTE-DTE}} = 3$, $\alpha = 0.5ms$, $\beta = 0.2ms$, $H_{\text{MN-AR}} = 1$, $H_{\text{CN-AR}} = 1$, $H_{\text{AR-DTE}} = 1$, $S_{\text{data}} = 1024$ (bytes), and $S_{\text{control}} = 1024$ (bytes).

Exercising the models with these parameters we obtained the results for the total cost of the operation (including mobility) for the variation of $N_{\text{Host/AR}}$ and N_{AR} , which demonstrate the behavior of the MOFI architecture including the proposed enhancement. Figure 6.25 shows that the total cost of communications in both MOFI and MOFI+SEID has a linear dependency on the number of hosts connected to each access router. It also demonstrates that the SEID approach adds a little overhead to MOFI. However, this addition remains constant, independently of the value of the number of hosts per AR, which indicates that SEID keeps the same level of scalability found in the original MOFI.

Regarding the number of access routers (ARs, gateways) present in the whole network, Figure 6.26 shows that the total cost of communications in MOFI has a linear dependency on the number of access routers. However, in contrast with the behavior when varying the number of hosts connected to each access router, adding SEID to MOFI removes the linear dependency that the total cost has on the number of access routers, transforming it to a logarithmic dependency and thus making the solution more scalable. This is because

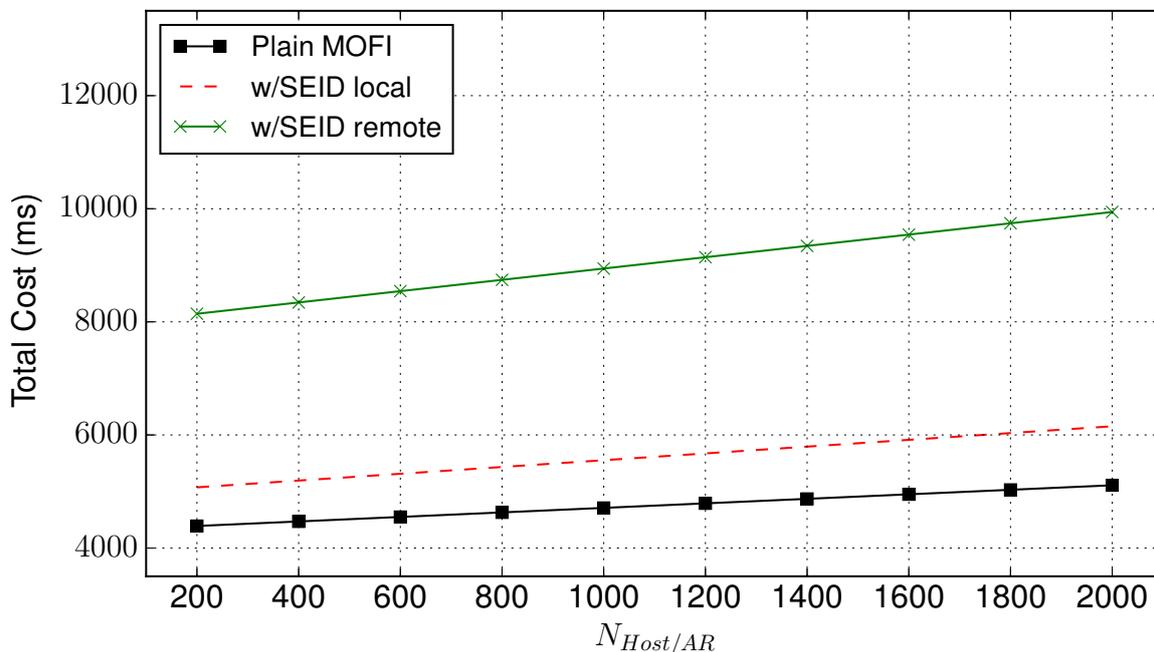


Figure 6.25: Total cost of MOFI and MOFI+SEID for the variation of $N_{Host/AR}$.

MOFI+SEID has a logarithmic growth as it only involves the necessary ARs to build the overlay network of our architecture while MOFI involves all ARs.

6.4 Conclusions

In this chapter we have presented The work carried out to integrate the architecture we propose in this thesis with both HIMALIS and MOFI. Apart from the benefits those architectures obtain from the integration work, it has brought us enormous feedback from the researchers working on them while also establishing a strong collaborative relation with them. We therefore consider these integrations as a mean to demonstrate the benefits that the architecture proposed with this thesis is able to provide to other architectures targeting the FI. Having demonstrated such benefits we can claim that we have validated the designs we propose against other bigger research efforts, also demonstrating its feasibility and interesting qualities for the future of the network.

Especially, with the work presented here we have brought to HIMALIS and MOFI some key mechanisms to enhance their security, privacy, and traceability prevention. We have also demonstrated how their handover mechanisms can be improved by addressing

6. Integration with Other Network Architectures

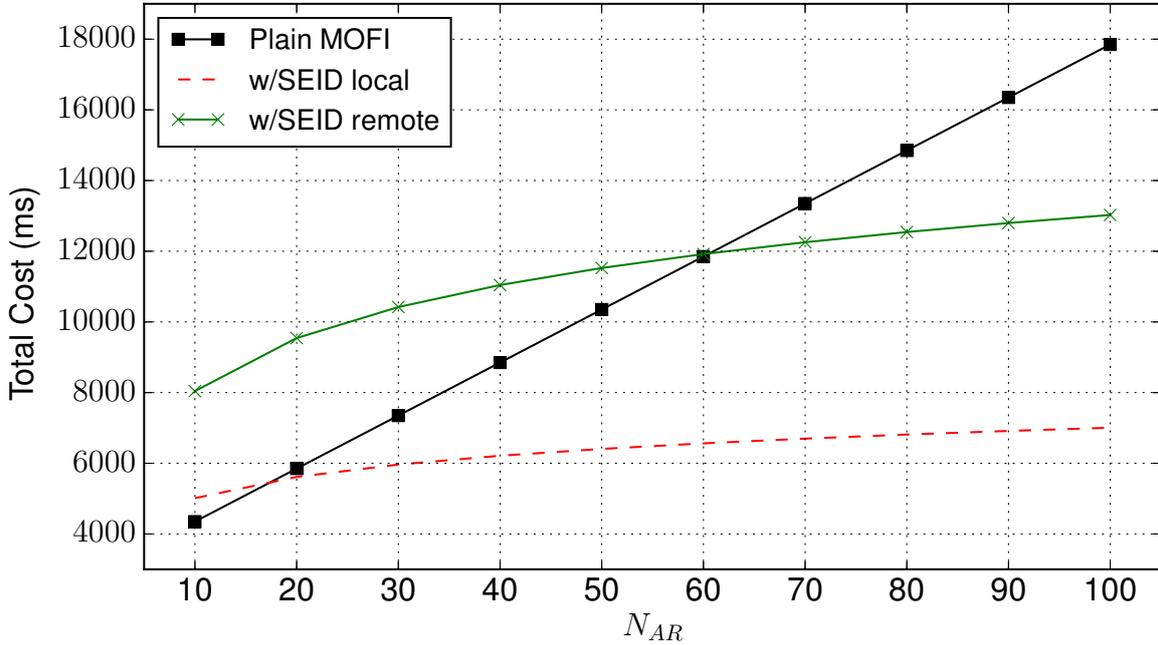


Figure 6.26: Total cost of MOFI and MOFI+SEID for the variation of N_{AR} .

their control operations through our identity-based control plane. To achieve this goal, we proposed to interconnect the elements defined by each architecture with the identity-based architecture we propose with this thesis, as deeply described in previous chapters. Being particularly defined for the HIMALIS architecture, we have also introduced the concept of *controller*, which brings an effective control plane connected to the control plane of our architecture (IBCP) and thus replaces (or complements) the current Host Name Registry (HNR) it defines.

In the integration proposal with the HIMALIS architecture, the elements of our architecture replace the IDR infrastructure of the original HIMALIS and generate a different profile that gets rid of the individual HNR elements and DNR elements of the original architecture, but offering the same functionality through our IBCP, playing the role of the IDR and HNR elements and making the DNR elements totally unnecessary. Moreover, the function of IBCP is the interpretation of the HIMALIS control operations intercepted by the controller in order to get an enhanced view of the network state that supports network intelligence and facilitates the improvement of typical network functions, such as mobility functions. Also, this change of view will facilitate the coupling of HIMALIS with SDN, so it will be easier to deploy in current and future networks.

As discussed in Sections 6.2.2 and 6.2.3 the overall security in communications is managed by the resulting IDR/HNR infrastructure, which function is provided by our architecture. It is achieved by making the *home* node of our architecture deployed on each network domain to be the main authority in the negotiation of network operations involving any of the entities of its domain, such as the resolution from Host ID to locator. Moreover, enhanced privacy is achieved by applying policies defined by the entities to those operations. Our architecture will protect the privacy by revealing information pertaining to an entity (identity) only to the allowed entities. Furthermore, such privacy is also enhanced by supporting the dynamic change of host identifiers, which can be used to prevent network operation linkage to concrete entities.

On the other hand, the integration with MOFI has been performed by adding a new functional block to its architecture, which we called SEID. It provides MOFI with identity-based negotiations of security aspects and abstraction of endpoints. We have also demonstrated the qualities that the integration of our components bring to MOFI, allowing it to scale better when several networks are interconnected. Together with the positive feedback given by MOFI designers to our research, and the active collaboration we established with them, this integration strengthens the global proposal of this thesis and the benefits it provides.

We have demonstrated the feasibility, scalability, and good performance of our integration proposals by evaluating them with theoretical and simulation models, as well as experimental proof-of-concept implementations running on realistic testbeds and with several network and administrative domains. We also demonstrated that the proposed mobility approach is totally transparent to the communication parties and almost transparent in terms of performance and message loss. Finally, we discussed the results that demonstrate the behavior of the architecture with many handover events through many different network domains. From it we can extract that the main obstacle in the handover is the establishment (set-up) of the new network interface when moving to a new network. In some situations, our proposal outperformed the base architectures because of the scalability provided by the mechanisms used in the functional blocks of our architecture.

Finally, we have also introduced the initial steps to build an experimentation testbed for HIMALIS, in order to make effective the proposed approach into its core and evaluate it while running in a real (logical and physical) network. This has been translated to the infrastructure provided by OpenLab [136], which part of the FIRE initiative [19]. During the experimentation, we introduced the main network elements defined by HIMALIS: the HNR, the DNR, and the IDR. We stated that each element targets to a specific operation and scope and justified its necessity. Then, we have experimented with a proof-of-concept

6. Integration with Other Network Architectures

instantiation of the architecture on top of an heterogeneous environment (mixing IPv4 and IPv6). With this, we found that both the handover process and the heterogeneous network integration are seamlessly achieved with it.

Chapter 7

Conclusions and Future Work

In the present document we have described and discussed the results of the research we have carried out to achieve the design of a proper architecture for the Future Internet. First we analyzed current architectures and proposals in order to find out the missing features and general gaps they have when targeting the specific requirements of new and future network and application services. Having those gaps in mind we researched an architecture to meet those requirements.

As we found during our research, there is no single architecture that provides the required capabilities but there are some promising approaches that are worth to consider. However, most outstanding approaches are clean-slate, requiring to jettison the current architecture before being deployed, which makes them less interesting to be adopted by network architects. This has highly influenced our main design decision: to ensure that our proposal begins with the current architecture and provides functional blocks to enable its evolution.

That said, the achievements of this thesis are enclosed in the design of a new functional blocks that can work together forming a full architecture or separately, integrated within other architectures, including current networks, to enhance them with a specific capability. From this we have also researched a redesigned network model for the Internet, as well as an evolution path using overlay networks, which has culminated in the abstract, flexible, and evolvable network model and architecture presented in Chapter 5.

Being an important success assessment, we have also validated the new model against other outstanding network architectures in Chapter 6. To sum up, in this chapter we conclude the discussion, correlating the objectives and major contributions presented in Chapter 1 with the research results discussed throughout this thesis. Finally, we present the

7. Conclusions and Future Work

most relevant research lines we have envisaged for the future work beyond the completion of this thesis.

7.1 Main Contributions

The architecture of the current Internet is highly outdated and there is a clear need for a shift in its network model to overcome the problems raised with the latest trends in networks and services [12]. However, building a new architecture for the Future Internet is not an easy task. Providing a new network model and associated architecture has been targeted by many research works, as discussed in Chapter 2, without too much success.

Our initial assumption to this respect is that the success of most previous architecture proposals has been directly related to their intention to change every piece of the network and their inability to provide a consistent network model that meets the requirements exposed by new services and to support an evolutionary deployment that uses the current architecture as initial point. Where other architectures have failed we think we could succeed.

Instead of providing a clean-slate architecture, in this thesis we have provided a set of functional blocks that can be used by any other architecture, as demonstrated throughout this document. Considering all blocks together we build a complete architecture that uses current architectures as underlying networks, so we provide an highly needed evolutionary way for the current architecture to adopt and support new network and service models respectively.

7.1.1 Design Approaches to Meet the Requirements

That said, the development of this thesis has been guided by the inclusion in the network architecture of functions that meet the requirements we stated at the beginning of this document, in Section 1.2. In Table 7.1 we give a brief description of the approach we have adopted to meet each requirement. We discuss them below.

First, the *flexibility* requirement states that different communication instances need different behaviors of the same network and such networks should be able to accept those instances without obliging them to model their interactions to the network offerings. This is the case of content delivery. The current network provides suboptimal content delivery and forces the creation of external mechanisms to improve it. Future underlying networks should be provisioned *as raw as possible*, allowing upper layer mechanisms to model them to their needs. This is the approach followed in the development of this thesis. We propose

Table 7.1: Summary of requirements and how they are met.

Requirement	Brief Description of how the requirement is met
Flexibility	Abstraction from the underlying network and integration with SDN
Adaptability	Separation of control plane and relying on overlay networks
Granularity	Abstracting hosts to <i>network objects</i> represented by their identities
Dynamicity	Dropping static systems in favor of overlay discovery and communication
Heterogeneity	Using overlays to embed communications onto underlying networks
Integrated Security	Policies to control the access to identity information
Privacy	Dynamic communication identifiers to prevent traceability

to abstract the network communications and leave the *embedding* to the last moment, especially supported by SDN in most scenarios.

The *adaptability* requirement states that the network architecture should be able to exploit the evolution of underlying networks without being intrusive to typical network operations and without adding complexity to upper layers and communication endpoints. In this thesis we enforce the total separation of control and data planes, implementing the control plane in a totally different infrastructure. Therefore, as presented in [104], with our architecture we propose to exploit the possibilities offered by overlay networks to achieve high level of abstraction for network operations that are unaffected by specific underlying network details. This allows our architecture to *consume* new network capabilities without modifying endpoints.

Regarding the *granularity* requirement, the current Internet is designed to conceive endpoints as hosts but it is no longer the case. New trends in networked elements include devices of many kinds and sizes, software, services, persons, etc. We have generalized the term to call them *network objects* and we provide support for such granular objects by considering them as they are and representing them by their *identities*. This way, network endpoints are placed in the identities instead of the hosts, freeing those network objects from the strict link they had with the hosts.

One of the requirements that has the biggest impact in current networks is the *dynamicity*. It reflects the current trends of an always changing network environment and network objects with very dynamic properties (e.g. context, surroundings, location, and even mood). In this thesis we approached this requirement from the beginning by considering everything to be dynamic: from the attributes of identities used to represent network objects, to the identifiers used in specific communication instances. Everything can change and our architecture provides a way for objects and networks to be aware of those changes.

7. Conclusions and Future Work

It is clear that the network of the future will not be unique. Several network proposals are proliferating from different organizations and research communities. Current underlying networks are based on different technologies, they have different properties, and shapes. Even the coexistence of IPv4 and IPv6 is increasing the *heterogeneity* of underlying networks. To this extent, a proper network architecture for the future that goes beyond the *all-IP* enforcement has to be able to interact with different underlying networks at the same time. We meet this requirement by making extensive use of overlay networks, which are a perfect mechanism to embed control and data planes of communications into the network, also exploiting their properties as much as possible.

Security has been a central topic when discussing about architecture designs and network models. Moreover, *privacy* has gained much attention when considering the current societal behavior and its reflection into the Internet. It is a common believe now that future network workloads and operations require some sort of *integrated security* that ensures communication reliability and considering *privacy* as a key point in network interactions. We have approached these two requirements from the same point of view: providing a set of basic security and privacy measures and enforcing policies to allow users to determine what can and cannot be disclosed. Both properties are highly related to the raise of identities as network endpoints. Controlling the access to identity information, including static and dynamic information such as the identifiers used in specific sessions, has been crucial in the design of our architecture, which has been reflected in the provision of the features offered by the DTEi and ODIN functional blocks.

7.1.2 Features and Functional Blocks

In order to provide the architectonic functions mentioned above, and as discussed throughout this document, we determined to differentiate different components or functional blocks from the whole architecture. Here we describe the main features of the architecture from the point of view of the functions introduced in Section 1.1 and relate them to the functional blocks that collaborate to provide them.

As shown in Table 7.2 the final architecture we have achieved in this thesis has some outstanding features that complement the standard communication and session management as well as the establishment of a data plane. These features are more related to the network model and the enhancement of the data plane in comparison to other architectures and proposals, as discussed in Chapter 2. That said, as the major contribution of this thesis we have the singular design of a new network model that places identities (attribute sets) as communication endpoints in order to provide unambiguous

7.1 Main Contributions

Table 7.2: Summary of features and functional blocks of the final architecture.

Feature	Mechanism or function	Involved functional blocks
Identification Type	Identities	DTEi, IBNP, IBCP
High Granularity	Extended Endpoints	DTEi, IBNP
Mobility Support	Dynamic Multihoming	IBNP, IBCP
Multi-homing Support	Dynamic Multihoming	IBNP, IBCP
Discovery	Based on partial identities	DTEi, ODIN
Flexible Naming	Identities	DTEi, ODIN
Integrated Security	Policies on Identities	DTEi

and unequivocal identification. The functional blocks that bring such benefits to network objects are the DTEi, IBNP, and IBCP. These blocks, together with ODIN, enclose the most remarked differences with current network architectures but still being integrated with other architectures, including current ones.

Regarding the functions that extend the endpoint concept, as presented in [137], our architecture approached them by enlarging the granularity of network objects, breaking the *host* metaphor into several pieces and providing more entity to each one, incarnated as network object. This function is mainly reflected in the DTEi and IBNP, as presented in [138] and [106] respectively. It supposes the abstraction and generalization of network endpoints, so communications can be established between any pair of network objects (identities), independently of their nature, size, power, etc. This also includes groups of network objects, so many-to-many communications are also supported with this model.

Both mobility and multi-homing are supported by the *dynamic multihoming* concept. It states that a network object can be attached to more than one underlying network at the same time. It also states that the points of attachment of a network object to the broad network can be increased or reduced at any time, so a network object can be attached to a different access network with or without detaching from the previous access networks to which it is attached. In summary, as presented in [128], this concept considers that a network object has N points of attachment to the Internet, where each points has its own properties and they are connected to the same or different access network. Such N points of attachment can vary on time, including new and discarding old, and the architecture proposed in this thesis is totally able to manage such operations.

All communications begin with the discovery of network objects. However, current networks consider that the initiator entity already knows the identifier (name, address) of the target entity, which is not the typical case and users have to rely on search engines to find out the specific identifier to which they have to communicate. To resolve this issue, our

7. Conclusions and Future Work

architecture includes the ODIN functional block, which we presented in [139]. It enables network objects to find each other by providing a set of attributes instead of a complete identifier. This set is not fixed, so network objects provide what they know about the entity they want to find, which we call partial identity, and the system will retrieve the necessary information to establish a communication session with it.

Flexible naming is another key objective in current research towards the Future Internet, and it is tightly related to the discovery functions. Our architecture interprets this feature as extreme, and identities provide enormous flexibility to the naming. Since they are sets of attributes, the final name of a network object will provide an higher degree of meaning to the entity that is processing it. Also, as new attributes can be added, it provides enough flexibility to evolve, grow, be organized (or not) in different hierarchies, etc. This function is provided by the DTEi when working together with ODIN. They support this flexibility for both *naming* and *identifying* network objects prior to establishing communication sessions.

Finally, as also discussed in previous subsection, the proposed architecture provides integrated security and privacy by means of policies applied to identities. Instead of enclosing and hiding identity attributes, our architecture controls the access to them and allows the authorized entities to retrieve the attributes to which they have been given rights but not others, including specific session identifiers. That said, as presented in [105], privacy is also enhanced by allowing different mechanisms to avoid traceability, so external network entities are not able to know *to whom* pertains certain flow of information just by seeing the identifiers used in the network data packets.

7.1.3 Integration with Other Architecture Proposals

The key point to asses the success of the research work carried out throughout the development of this thesis relayed in the integration of its newly designed functions with other network architectures and proposals. Although we achieved some outstanding level of integration within a disruptive virtualization architecture for the Future Internet, as published in [140], our main concerns were targeting other mainstream architectures, such as MOFI [16] and HIMALIS [14].

The rationale behind the integration with MOFI and HIMALIS has been that they are still in development, their current functions are in good shape, as shown in Chapter 2, they embrace current architectures, so they do not propose to jettison them in order to be deployed, and the are the most outstanding representative architecture proposals in South Korea and Japan respectively. Moreover, we could establish good relations and strong collaboration for the present and the future.

Regarding the integration with HIMALIS, being a major achievement of this thesis, we performed the integration of our architecture by connecting HIMALIS elements with the DTEi, adapting IBNP, and using IBCP to extend its control plane. We have presented the results of this work in [120, 141, 142]. It enhances multi-domain and mobility operations in HIMALIS by means of using identities for address them. As, unlike pure Information Centric Networking (ICN) architectures, HIMALIS supports direct communications, we could be able to embed and translate communication instances (mainly sessions) into the network without requiring any adaptation layer. We also demonstrated the feasibility of the system, both individually working and working in collaboration with HIMALIS, so this step of our assessment was successful and the contribution showed itself to be strong. Moreover, the HIMALIS architecture was enlightened with the future use of ODIN for the discovery of fine-grain network objects like those found in the Internet of Things, as we describe in the following section.

In parallel, the integration with MOFI is also an important achievement of this thesis. We have presented the main aspects of such integration in [143]. It consists in the connection of MOFI elements with our architecture by means of integrating the DTEi together with IBNP as an additional component of MOFI. In this case we did not integrate IBCP as MOFI design is more abstract and has no clear point of integration with it. However, we achieved our goal to asses our design against an almost complete architecture by integrating our identity-related functions to ensure that MOFI operations are secure and trustworthy. This also demonstrates the timeliness and usefulness of the approaches we designed in this thesis. Furthermore, the next evolution of MOFI, which is more close to be a clean-slate approach than an evolutionary approach, has just started to being researched and we have an open topic for the continuation of the work started with this thesis.

7.2 Future Work

As mentioned throughout this document, during our research towards the new network model and architecture for the Future Internet we have identified many aspects, topics, and specific functions that have been left for future research work, as they were out of the scope of this thesis. This way we could concentrate in the main functions and most important aspects of the architecture, so our design and experimentation results are as clear, simple, and easy to understand as possible, so the essence of the architecture can be appreciated at a glance.

7. Conclusions and Future Work

In this section we address the possible research lines and topics for the future. The most outstanding aspects that have to be researched beyond the conclusion of this thesis are centered around the integration of the main functions we defined, specially enclosed in the identity-based control plane, within other architectures and their corresponding evolutions. We still believe that such functions are worth for future iterations of the network model of the Internet, and it has been demonstrated by the collaborative work with the researchers behind their design.

Moreover, being a huge trending research topic, the enormous advantages that Software Defined Networking (SDN) provides to mold the behavior of running networks has influenced the final stages of our research. In this sense, the present thesis has intentional and unintentional parallelisms with SDN because our research objectives met theirs, but from a different point of view. Furthermore, the Internet of Things (IoT) has placed huge challenges to network architectures. We tried to meet some of them with this thesis but others are still open so we can continue targeting them in the future.

7.2.1 Continuous Integration with New Network Architectures

As we have stated throughout this thesis, the integration with other architectures of the network functions or whole functional blocks defined in this thesis is one of our main objectives. The main rationale behind the integrations is twofold. First, we can use it to validate our results and get feedback about specific mechanisms from the perspective of other research approaches. Second, we encourage and push the inclusion of valuable functions into such architectures as well as the reuse of existing techniques, which end up improving their success.

That said, the end of the present work will not mark the end of our research. There are new architecture proposals that would benefit from the functions we defined here. The MOFI architecture [16] is a good representation of it. Although we have presented the initial integration results, MOFI has advanced behind the scenes and it has exposed new areas to which we can improve by integrating our approach, specially with the ODIN component and the identity-based control plane.

On the other hand, being a continuation of MOFI, IDnet ¹ tries to revolutionize the concept and network model for the Future Internet. Although it is essentially a clean-slate architecture, it can benefit from our work to facilitate the inclusion of some or all of its benefits in current networks by offering its functions as evolutions of the current network through the control plane, specifically the identity-based control plane. Moreover, our

¹<http://idnet.re.kr>

discovery functions are highly compatible with the routing scheme defined by IDnet, so we can also benefit from it by incorporating some aspects of such routing scheme into our approach.

Regarding the deepening into the security and privacy aspects of our work, as mentioned in relation to the management of identities, it is worth to research the actual integration of our architecture with other identity management infrastructures, such as Shibboleth [81] and general SAML solutions [79]. However, there are other approaches worth to consider, such as OpenID [82] and OAuth [144], which are gaining importance in the last years in application and service access, with some interest in lower level network access. The result of this work would be the improvement of security and privacy by enhancing the policy mechanisms used to regulate the access to identity attributes.

The progress and advancement of the HIMALIS architecture [14] has also been parallel to the development of the work presented in this thesis. This has led us to work with an outdated snapshot of HIMALIS and leave for future work the integration with newer versions and derivatives, such as NIN [145]. However, we have tried to be up-to-date in such advancements, so our latest research efforts, especially the design of the identity-based control plane described in Section 4.6, have considered the new approaches and network model introduced in new iterations of HIMALIS. During this aspect of our work we have verified that our approach still have benefits to give to HIMALIS, so continuing this integration in the future is encouraged.

7.2.2 Tighten Relations with Software Defined Networking

It is no secret and has been stated throughout this document that the research work carried out in this thesis has a lot of parallelism with the Software Defined Networking (SDN) movement. Even though our work and SDN did not have anything in common at the beginning, the fact that most challenges regarding network architecture were important to resolve inspired both this work and the support for SDN. However, the main work of this thesis has been concentrated in abstracting the network model to deliver specific qualities to both users and services. In contrast, SDN has been concentrated in the point of view of network governance and administration.

Despite it targets a different purpose, the mechanisms offered by SDN are highly valuable for the design of any network architecture. We have stated it in Section 6.2, while discussing the integration of HIMALIS. This is because SDN has been proven a very good mechanism to mold underlying network substrate to the needs of the middle network architectures, specifically HIMALIS and the approach we presented in this thesis. Having

7. Conclusions and Future Work

this in mind, and watching carefully the results we obtained during our research, it is worth to continue the research towards the incorporation of SDN mechanisms in our architecture.

On the other hand, general network softwarization ² is an open research area, which is obviously influenced and related to SDN, but applies to any networking mechanism. The softwarization of the functions defined by our architecture and its functional blocks is the most obvious direction for our future research and design to improve their capacity of success, especially working on top of SDN infrastructures. Moreover, it will continue the work already started with the definition of the identity-based control plane [142].

Finally, it is worth to mention that SDN retains our main design principle, which states that changes in the network are introduced in increments, evolving current infrastructures with additions of new functional blocks connected by standardized APIs to the control plane of the network. We have followed this approach during the development of this thesis, so it would be easy to adapt our functional blocks to SDN and connect our identity-based control plane with the general control plane defined by SDN.

7.2.3 Evolution Towards the Internet of Everything

It is widely known that the Internet is a continuously changing environment. Accepting changes, in any aspect of both network and application services, is the invariant of any discussion about the future of the network, although most changes are avoided because of the limitations of the current network architecture to accept them, as mentioned throughout this document. That is the case of the Internet of Things (IoT) [76]. It imposes very particular requirements to the Internet that are difficult to meet. However, a fully connected world is becoming a reality more than a foresight or an illusion.

There is a huge hype around IoT as the next (or even current) major challenge and interest for both business and network researchers, as we stated in Section 1.1. The huge benefits that such fully connected world provides to the society also support the highly demanding environment it exposes. However, the network model shared by the current Internet and access networks is not valid to support the workload required by IoT. There is a clear need for new integral models that not just include the specificities of *things* into their objectives, but also into their premises and invariants.

Raising the role of identities in IoT has been claimed to be mandatory to meet those challenges [90], since they provide the necessary semantics to unequivocally identify *things*, which are also network objects in the terms used in this thesis. Well-established semantics around such concept is a key function for achieving the consistence and coherence of IoT

²<http://sites.ieee.org/netsoft>

networks (see Section 4.4 and Appendix C). This way, the research results presented in this thesis can be directly applied to IoT, as demonstrated in [146,147], so we will continue our research work in discovery and identification processes towards the provision of functions to meet the requirements of future IoT environments.

On the other hand, Machine-to-Machine (M2M) communications concentrate most part of the network complexity of IoT scenarios [100] because involving only machines as communication endpoints has been less common in the past and, therefore, it has not been considered in the design of the core networking infrastructures. This means that M2M scenarios and workloads need to be properly researched in order to know the necessary functions that network architectures must provide in order to fully support them without negatively affecting other scenarios and enabling the proliferation of new application and network services around IoT and M2M technologies.

According to the latest trends in network architectures for the Future Internet, it looks like new approaches for both SDN and M2M are converging to the same direction. Once again, the improvements in SDN reflect the requirements of the operators and administrators of the underlying networks while the improvements in M2M/IoT reflect the requirements from users, applications, and service. The necessary mechanisms to instantiate (embed) M2M/IoT workloads into underlying networks are placed just in the middle and here is where the continuation of the work presented in this thesis enters into the equation. Our research on discovery and scalability will be useful for allowing all elements of the network to accept large number of network objects without impacting the performance of the system. It is an essential requirement of the most prominent scenarios found in IoT. This adequacy will also affect to M2M communications so we agree on that modelling network object interactions, abstracting them, and reflecting their intrinsic qualities is mandatory to support current and future M2M interactions without limiting their scope. And it has been supported by newer evolutions of the other network architectures with which we are collaborating, especially HIMALIS.

7. Conclusions and Future Work

Bibliography

- [1] Raj Jain. Internet 3.0: Ten problems with current internet architecture and solutions for the next generation. In *Proceedings of Military Communications Conference*, pages 1–9, Los Alamitos, CA, USA, 2006. IEEE Computer Society.
- [2] T. Li. Design Goals for Scalable Internet Routing, 2011. <http://www.ietf.org/rfc/rfc6227.txt>.
- [3] Gabriel López, Oscar Cánovas, Antonio F. Gómez-Skarmeta, and Joao Girao. A swift take on identity management. *IEEE Computer*, 42(5):58–65, 2009.
- [4] B. Carpenter. Architectural Principles of the Internet, 1996. <http://www.ietf.org/rfc/rfc1958.txt>.
- [5] Dimitri Papadimitriou, Theodore Zahariadis, Pedro Martinez-Julia, Ioanna Papafili, Vito Morreale, Francesco Torelli, Bernard Sales, and Piet Demeester. Design principles for the future internet architecture. In *The Future Internet*, pages 55–67. Springer, Berlin, Heidelberg, Germany, 2012.
- [6] J. Schonwalder, M. Fouquet, G. Rodosek, and I. Hochstatter. Future internet = content + services + management. *IEEE Communications Magazine*, 47(7):27–33, 2009.
- [7] Alex Galis et al. Position Paper on Management and Service-aware Networking Architectures (MANA) for Future Internet. http://www.future-internet.eu/fileadmin/documents/prague_documents/MANA_PositionPaper-Final.pdf, 2010.
- [8] M.A. Rappa. The utility business model and the future of computing services. *IBM Systems Journal*, 43(1):32–42, 2004.

BIBLIOGRAPHY

- [9] X. Liu, Y. Hui, W. Sun, and H. Liang. Towards service composition based on mashup. In *2007 IEEE Congress on Services*, pages 332–339, 2007.
- [10] Ved P. Kafle, Hideki Otsuki, and Masugi Inoue. An id/locator split architecture for future networks. *IEEE Communications Magazine*, 48(2):138–144, 2010.
- [11] Matthew Caesar, Tyson Condie, Jayanthkumar Kannan, Karthik Lakshminarayanan, and Ion Stoica. Roff: Routing on flat labels. *SIGCOMM Computer Communication Review*, 36(4):363–374, 2006.
- [12] T. Li. Design goals for scalable internet routing. Internet-draft, Internet Research Task Force, 2007.
- [13] ITU-T. Series X: Data Networks, Open system communications and security. Cyberspace security - Identity management. Baseline capabilities for enhancing global identity management and interoperability. Recommendation ITU-T X.1250, 2009.
- [14] Ved P. Kafle and Masugi Inoue. HIMALIS: Heterogeneity inclusion and mobility adaptation through locator id separation in new generation network. *IEICE Transactions on Communications*, E93-B(3):478–489, 2010.
- [15] Van Jacobson, Diana K. Smetters, James D. Thornton, Michael F. Plass, Nicholas H. Briggs, and Rebecca L. Braynard. Networking named content. In *Proceedings of the 5th International Conference on Emerging Networking Experiments and Technologies (CoNEXT '09)*, pages 1–12, New York, NY, USA, 2009. ACM.
- [16] Heeyoung Jung, Moneeb Gohar, Ji-In Kim, and Seok Joo Koh. Distributed mobility control in proxy mobile ipv6 networks. *IEICE Transactions on Communications*, E94-B(8):2216–2224, 2011.
- [17] P. Srisuresh and M. Holdrege. IP Network Address Translator (NAT) Terminology and Considerations. Rfc 2663, IETF, 1999.
- [18] European Future Internet Portal. Future Internet Assembly. <http://www.future-internet.eu/home/future-internet-assembly.html>, 2010.
- [19] FIRE - Future Internet Research and Experimentation, 2011. <http://cordis.europa.eu/fp7/ict/fire>.

- [20] NSF NeTS FIND (Future Internet Design) Initiative, 2011. <http://www.nets-find.net>.
- [21] Global Environment for Network Innovation (GENI), 2011. <http://www.geni.net>.
- [22] Future Internet Forum, 2011. <http://fif.kr>.
- [23] National Institute of Information and Communications Technology. “AKARI” Architecture Design Project for New Generation Network, 2010. <http://akari-project.nict.go.jp>.
- [24] D. R. Cheriton and M. Gritter. TRIAD: A Scalable Deployable NAT-based Internet Architecture. Technical report, Technical Report, Stanford University, 2000.
- [25] M. Gritter and D. R. Cheriton. Name-based Routing Protocol Specification. Technical report, Technical Report, Stanford University, 1999.
- [26] R. Braden, T. Faber, and M. Handley. From protocol stack to protocol heap: role-based architecture. *ACM SIGCOMM Computer Communication Review*, 33(1):17–22, 2002.
- [27] R. Moskowitz and P. Nikander. Host Identity Protocol (HIP) Architecture, 2006. <http://www.ietf.org/rfc/rfc4423.txt>.
- [28] David Meyer. The locator identifier separation protocol (lisp). *The Internet Protocol Journal*, 11(1):23–36, 2008.
- [29] D. Farinacci, V. Fuller, D. Meyer, and D Lewis. Locator/id separation protocol (LISP). Internet-draft, IETF, 2011.
- [30] Jukka Ylitalo and Pekka Nikander. BLIND: A complete identity protection framework for end-points. *Lecture Notes in Computer Science*, 3957:163–176, 2006.
- [31] X. Xu. Routing Architecture for the Next Generation Internet (RANGI). Internet-draft, IETF, 2009.
- [32] B. Ahlgren, J. Arkko, L. Eggert, and J. Rajahalme. A node identity internetworking architecture. In *Proceedings of the 25th IEEE International Conference on Computer Communications (INFOCOM 2006)*, pages 1–6, Washington, DC, USA, 2006. IEEE.

BIBLIOGRAPHY

- [33] Heeyoung Jung and Seok Joo Koh. MOFI: Future internet architecture with address-free hosts for mobile environments. *Telecommunications Review*, 21(2):343–358, 2011.
- [34] Randall Atkinson, Saleem Bhatti, and Stephen Hailes. ILNP: mobility, multihoming, localised addressing and security through naming. *Telecommunication Systems*, 42(3):273–291, 2009.
- [35] Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, and Hari Balakrishnan. Chord: A scalable peer-to-peer lookup service for internet applications. In *Proceedings of the 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communications*, pages 149–160, New York, NY, USA, 2001. ACM.
- [36] Hui Zhang, Ashish Goel, and Ramesh Govindan. Incrementally improving lookup latency in distributed hash table systems. In *Proceedings of the 2003 ACM SIGMETRICS International Conference on Measurement and Modeling of Computer Systems*, pages 114–125, New York, NY, USA, 2003. ACM.
- [37] Matthew Chapman Caesar. *Identity-based Routing*. PhD thesis, EECS Department, University of California, Berkeley, 2007.
- [38] Matthew Caesar, Miguel Castro, Edmund B. Nightingale, Greg O’Shea, and Antony Rowstron. Virtual Ring Routing: Network Routing Inspired by DHTs. *SIGCOMM Computer Communication Review*, 36(4):351–362, 2006.
- [39] Prasanna Ganesan, Krishna Gummadi, and H. Garcia-Molina. Canon in g major: designing dhts with hierarchical structure. In *Proceedings of the 24th International Conference on Distributed Computing Systems, 2004*, pages 263–272, Washington, DC, USA, 2004. IEEE.
- [40] Gregorio Martinez Perez, Felix J. Garcia Clemente, and Antonio F. Gomez Skarmeta. Building and managing policy-based secure overlay networks. In *Proceedings of the 16th Euromicro Conference on Parallel, Distributed and Network-Based Processing*, pages 597–603, Washington, DC, USA, 2008. IEEE Computer Society.
- [41] Petar Maymounkov and David Mazières. Kademlia: A peer-to-peer information system based on the xor metric. In *Proceedings of the First International Workshop on Peer-to-Peer Systems*, pages 53–65, London, UK, 2002. Springer-Verlag.

- [42] Lawrence Cheng, Alex Galis, Bertrand Mathieu, Kerry Jean, Roel Ocampo, Lefteris Mamatras, Javier Rubio-Loyola, Joan Serrat, Andreas Berl, Hermann Meer, Steven Davy, Zeinab Movahedi, and Laurent Lefevre. Self-organising management overlays for future internet services. In *Proceedings of the 3rd IEEE International Workshop on Modelling Autonomic Communications Environments*, pages 74–89, Berlin, Heidelberg, Germany, 2008. Springer-Verlag.
- [43] Jeroen Famaey, Jef Donders, Tim Wauters, Frederic Iterbeke, Niels Sluijs, Bart De Vleeschauwer, Filip De Turck, Piet Demeester, and Rudy Stoop. Comparative study of peer-to-peer architectures for scalable resource discovery. In *Proceedings of the First International Conference on Advances in P2P Systems*, pages 27–33, Los Alamitos, CA, USA, 2009. IEEE Computer Society.
- [44] Laurent Mathy and Luigi Iannone. Lisp-dht: Towards a dht to map identifiers onto locators. In *Proceedings of the ACM CoNEXT Conference*, pages 1–6, New York, NY, USA, 2008. ACM.
- [45] Marc Sanchez Artigas, Pedro Garcia Lopez, and Antonio F. Gomez Skarmeta. A comparative study of hierarchical dht systems. In *Proceedings of the 32th Conference on Local Computer Networks*, pages 325–333, Washington, DC, USA, 2007. IEEE Computer Society.
- [46] Jianli Pan, Subharthi Paul, Raj Jain, and Mic Bowman. Milsa: A mobility and multihoming supporting identifier locator split architecture for naming in the next generation internet. In *Proceedings of the Global Communications Conference*, pages 2264–2269, Washington, DC, USA, 2008. IEEE.
- [47] Jianli Pan, Raj Jain, Subharthi Paul, Mic Bowman, Xiaohu Xu, and Shanzhi Chen. Enhanced milsa architecture for naming, addressing, routing and security issues in the next generation internet. In *Proceedings of the International Conference on Communications*, pages 14–18, Washington, DC, USA, 2009. IEEE.
- [48] J. Rosenberg et al. SIP: Session Initiation Protocol, 2002. <http://www.ietf.org/rfc/rfc3261.txt>.
- [49] Xiaohu Xu and Dayong Guo. Hierarchical routing architecture (hra). In *Next Generation Internet Networks, 2008. NGI 2008*, pages 92–99, 2008.

BIBLIOGRAPHY

- [50] Louise Burness, Philip Eardley, and Robert Hancock. The Trilogy Architecture for the Future Internet. In *Towards the Future Internet - A European Research Perspective*, pages 79–90. IOS Press, Amsterdam, 2009.
- [51] C. Tschudin and C. Jelger. An "Autonomic Network Architecture" Research Project. *PIK Magazine*, 30(1):26–31, 2007.
- [52] Marcus Brunner, Henrik Abramowicz, Norbert Niebert, and Luis M. Correia. 4WARD: A european perspective towards the future internet. *IEICE Transactions on Communications*, E93-B(3):442–445, 2010.
- [53] Thomas Edwall et al. Scalable and Adaptive Internet Solutions (SAIL), 2011. <http://www.sail-project.eu>.
- [54] Bengt Ahlgren, Matteo D’Ambrosio, Christian Dannewitz, et al. Netinf evaluation. Technical Report FP7-ICT-2007-1-216041-4WARD/D-6.3, 2010. Deliverable D-6.3, 4WARD EU FP7 Project, <http://www.4ward-project.eu>.
- [55] Vladimir Dimitrov and Ventsislav Koptchev. PSIRP project – publish-subscribe internet routing paradigm: New ideas for future internet. In *Proceedings of the 11th International Conference on Computer Systems and Technologies and Workshop for PhD Students in Computing on International Conference on Computer Systems and Technologies*, pages 167–171, New York, NY, USA, 2010. ACM.
- [56] Dirk Trossen et al. Pursuing a Pub/Sub Internet (PURSUIT), 2011. <http://www.fp7-pursuit.eu>.
- [57] Teemu Koponen, Mohit Chawla, Byung-Gon Chun, Andrey Ermolinskiy, Kye Hyun Kim, Scott Shenker, and Ion Stoica. A data-oriented (and beyond) network architecture. *SIGCOMM Computer Communication Review*, 37(4):181–192, 2007.
- [58] P. Mockapetris et al. Domain Names - Implementation and Specification, 1987. <http://www.ietf.org/rfc/rfc1035.txt>.
- [59] Zeilenga K. et al. Lightweight Directory Access Protocol (LDAP): Technical Specification Road Map, 2006. <http://www.ietf.org/rfc/rfc4510.txt>.
- [60] Graham Klyne and Jeremy J. Carroll. Resource Description Framework (RDF): Concepts and Abstract Syntax, 2004. <http://www.w3.org/TR/rdf-concepts/>.

- [61] Min Cai, Martin Frank, Baoshi Yan, and Robert MacGregor. A subscribable peer-to-peer rdf repository for distributed metadata management. *Web Semantics: Science, Services and Agents on the World Wide Web*, 2(2):109–130, 2004.
- [62] Andy Seaborne. RDQL - A Query Language for RDF, 2004. <http://www.w3.org/Submission/RDQL/>.
- [63] Eric Prud'hommeaux and Andy Seaborne. SPARQL Query Language for RDF, 2008. <http://www.w3.org/TR/rdf-sparql-query/>.
- [64] Bluetooth SIG. Bluetooth Service Discovery Protocol, 2012. <http://www.bluetooth.com>.
- [65] UPnP Forum. Universal Plug and Play, 2012. <http://upnp.org>.
- [66] The Salutation Consortium. Salutation, 2012. <http://salutation.org>.
- [67] E. Guttman, C. Perkins, J. Veizades, and M. Day. Service Location Protocol, Version 2, 1999. <http://www.ietf.org/rfc/rfc2608.txt>.
- [68] The Apache Foundation. Apache River, 2012. <http://river.apache.org>.
- [69] S. Cheshire and M. Krochmal. Multicast DNS, 2011. <http://tools.ietf.org/html/draft-cheshire-dnsext-multicastdns-15>.
- [70] S. Cheshire and M. Krochmal. DNS-Based Service Discovery, 2011. <http://tools.ietf.org/html/draft-cheshire-dnsext-dns-sd-11>.
- [71] E. Meshkova, J. Riihijärvi, M. Petrova, and P. Mähönen. A survey on resource discovery mechanisms, peer-to-peer and service discovery frameworks. *Computer Networks*, 52(11):2097–2128, 2008.
- [72] Daniel Elenius and Magnus Ingmarsson. Ontology-based service discovery in p2p networks. In *Proceedings of the MobiQuitous'04 Workshop on Peer-to-Peer Knowledge Management (P2PKM 2004)*, pages 1–9, Washington, DC, USA, 2004. IEEE.
- [73] Knarig Arabshian and Henning Schulzrinne. An ontology-based hierarchical peer-to-peer global service discovery system. *Journal of Ubiquitous Computing and Intelligence*, 1(2):133–144, 2007.

BIBLIOGRAPHY

- [74] S. Ratnasamy, P. Francis, M. Handley, R. Karp, and S. Shenker. A scalable content-addressable network. *Computer Communication Review*, 31(4):161–172, 2001.
- [75] Christian Bizer, Tom Heath, and Tim Berners-Lee. Linked data - the story so far. *International Journal on Semantic Web and Information Systems (IJSWIS)*, 5(3):1–22, 2009.
- [76] D. Pfisterer, K. Romer, D. Bimschas, O. Kleine, R. Mietz, C. Truong, H. Hasemann, M. Pagel, M. Hauswirth, M. Karnstedt, et al. Spitfire: Toward a semantic web of things. *IEEE Communications Magazine*, 49(11):40–48, 2011.
- [77] J. Saltzer et al. On the Naming and Binding of Network Destinations, 1993. <http://www.ietf.org/rfc/rfc1498.txt>.
- [78] ITU-T. NGN identity management framework. Recommendation Y.2720, 2009.
- [79] Security assertion markup language (saml). <http://saml.xml.org>.
- [80] Liberty Alliance Project. Liberty ID-WSF Web Services Framework Overview, Version: 2.0. http://www.projectliberty.org/specifications__1.
- [81] Shibboleth. <http://shibboleth.internet2.edu>.
- [82] David Recordon and Drummond Reed. Openid 2.0: A platform for user-centric identity management. In *Proceedings of the Second ACM Workshop on Digital Identity Management*, pages 11–16, New York, NY, USA, 2006. ACM.
- [83] D. Eastlake and P. Jones. US Secure Hash Algorithm 1 (SHA1), 2001. <http://www.ietf.org/rfc/rfc3174.txt>.
- [84] IETF. Hypertext Transfer Protocol – HTTP/1.1, 1997. <http://www.ietf.org/ids.by.wg/http.html>.
- [85] R. Moskowitz and P. Nikander. Host identity protocol (HIP) architecture. Rfc 4423, IETF, 2006.
- [86] Gennaro Cordasco, Francesca Della Corte, Alberto Negro, Alessandra Sala, and Vittorio Scarano. Relaxed-2-chord: Efficiency, flexibility and provable stretch. In *Proceedings of the International Parallel and Distributed Processing Symposium*, pages 1–8, Los Alamitos, CA, USA, 2009. IEEE Computer Society.

- [87] Marc Sánchez Artigas, Pedro García López, Jordi Pujol Ahulló, and Antonio F. Gómez Skarmeta. Cyclone: A novel design schema for hierarchical dhds. In *Proceedings of the IEEE International Conference on Peer-to-Peer Computing*, pages 49–56, Los Alamitos, CA, USA, 2005. IEEE Computer Society.
- [88] N. Fotiou, G.C. Polyzos, and D. Trossen. Illustrating a publish-subscribe internet architecture. In *2nd Euro-NF Workshop on Future Internet Architectures and New Trends in Network and Services Architectures*, 2009.
- [89] Tom Leighton. Improving performance on the internet. *Communications of the ACM*, 52(2):44–51, 2009.
- [90] Amardeo C. Sarma and Joao Girao. Identities in the future internet of things. *Wireless Personal Communications*, 49(3):353–363, 2009.
- [91] Drummond Reed, Les Chasen, and William Tan. Openid identity discovery with XRI and XRDS. In *Proceedings of the 7th Symposium on Identity and Trust on the Internet (IDtrust '08)*, pages 19–25, New York, NY, USA, 2008. ACM.
- [92] Yves Lafon et al. Simple Object Access Protocol, 2007. <http://www.w3.org/TR/soap>.
- [93] Petros Daras and John Domingue. The question of discovery and search in the future internet. In *Future Internet Assembly*, November 23-24 2009.
- [94] Donald E. Knuth. *The Art of Computer Programming*. Addison-Wesley, 2006.
- [95] David Beckett, Tim Berners-Lee, and Eric Prud'hommeaux. Turtle, Terse RDF Triple Language, 2011. <http://www.w3.org/TR/turtle/>.
- [96] Dan Brickley and R. V. Guha. RDF Vocabulary Description Language 1.0: RDF Schema, 2004. <http://www.w3.org/TR/rdf-schema/>.
- [97] Aldo Garmeni. DOLCE+DnS Ultralite, 2012. http://ontologydesignpatterns.org/wiki/Ontology:DOLCE+DnS_Ultralite.
- [98] Dan Brickley and Libby Miller. FOAF Vocabulary Specification, 2010. <http://xmlns.com/foaf/spec/>.

BIBLIOGRAPHY

- [99] Boris Motik, Peter F. Patel-Schneider, and Bijan Persia. OWL 2 - Web Ontology Language - Structural Specification and Functional-Style Syntax, 2009. <http://www.w3.org/TR/owl2-syntax/>.
- [100] Inge Gronbek and Pratik K. Biswas. Ontology-based abstractions for m2m virtual nodes and topologies. In *International Conference on Ultra Modern Telecommunications & Workshops, 2009 (ICUMT'09)*, pages 1–8, Washington, DC, USA, 2009. IEEE.
- [101] A. Daouadji, KK Nguyen, M. Lemay, and M. Cheriet. Ontology-based resource description and discovery framework for low carbon grid networks. In *First IEEE International Conference on Smart Grid Communications (SmartGridComm), 2010*, pages 477–482, Washington, DC, USA, 2010. IEEE.
- [102] Mikael Nilsson, Andy Powell, Pete Johnston, and Ambjörn Naeve. Expressing Dublin Core metadata using the Resource Description Framework (RDF), 2008. <http://dublincore.org/documents/dc-rdf/>.
- [103] The Apache Software Foundation. Apache Avro 1.7.0, 2012. <http://avro.apache.org/docs/1.7.0/>.
- [104] Pedro Martinez-Julia, Antonio F. Gomez-Skarmeta, Joao Girao, and Amardeo Sarma. Protecting digital identities in future networks. In *Proceedings of the Future Network and Mobile Summit 2011*, pages 1–8. International Information Management Corporation, 2011.
- [105] Pedro Martinez-Julia and Antonio F. Gomez-Skarmeta. Using identities to achieve enhanced privacy in future content delivery networks. *Computers and Electrical Engineering*, 38(2):346–355, 2012.
- [106] Pedro Martinez-Julia and Antonio F. Gomez-Skarmeta. A novel identity-based network architecture for next generation internet. *Journal of Universal Computer Science*, 18(12):1643–1661, 2012.
- [107] Dan Boneh and Matthew Franklin. Identity-based encryption from the weil pairing. *SIAM Journal on Computing*, 32(3):586–615, 2003.
- [108] S. Gundavelli et al. Proxy Mobile IPv6, 2008. <http://www.ietf.org/rfc/rfc5213.txt>.

- [109] H. Soliman et al. Hierarchical Mobile IPv6 (HMIPv6) Mobility Management, 2008. <http://www.ietf.org/rfc/rfc5380.txt>.
- [110] Luca Viganò. Automated security protocol analysis with the AVISPA tool. *Electronic Notes in Theoretical Computer Science*, 155:61–86, 2006.
- [111] Pedro Martinez-Julia, Antonio J. Jara, and Antonio F. Skarmeta. Gaia extended research infrastructure: Sensing, connecting, and processing the real world. In *Proceedings of the TridentCom 2012*, pages 3–4. Springer, 2012.
- [112] Janez Brank, Marko Grobelnik, and Dunja Mladenić. A survey of ontology evaluation techniques. In *Proceedings of the Conference on Data Mining and Data Warehouses (SiKDD 2005)*, pages 1–4, Ljubljana, Slovenia, 2005.
- [113] Marta Sabou, Jorge Gracia, Sofia Angeletou, Mathieu d’Aquin, and Enrico Motta. Evaluating the semantic web: A task-based approach. In *Proceedings of the 6th International Conference on the Semantic Web and 2nd Asian Conference on Asian Semantic Web*, pages 423–437, London, UK, 2007. Springer-Verlag.
- [114] Nicholas J. Humfrey. RedStore, 2012. <http://www.aelius.com/njh/redstore/>.
- [115] The Trustees of Princeton University. PlanetLab, 2007. <http://www.planet-lab.org>.
- [116] A. Quereilhac, M. Lacage, C. Freire, T. Turletti, and W. Dabbous. NEPI: An integration framework for network experimentation. In *Proceedings of the 19th International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*, pages 1–5, Washington, DC, USA, 2011. IEEE.
- [117] P Nikander et al. End-Host Mobility and Multihoming with the Host Identity Protocol, 2008. <http://www.ietf.org/rfc/rfc5206.txt>.
- [118] J. Laganier et al. Host Identity Protocol (HIP) Rendezvous Extension, 2008. <http://www.ietf.org/rfc/rfc5204.txt>.
- [119] C. Perkins, D. Johnson, and J. Arkko. Mobility Support in IPv6, 2011. <http://www.ietf.org/rfc/rfc6275.txt>.

BIBLIOGRAPHY

- [120] Pedro Martinez-Julia, Antonio F. Gomez-Skarmeta, Ved P. Kafle, and Masugi Inoue. Secure and robust framework for id/locator mapping system. *IEICE Transactions on Information and Systems*, E95-D(1):108–116, 2012.
- [121] C. Kaufman et al. Internet Key Exchange (IKEv2) Protocol, 2005. <http://www.ietf.org/rfc/rfc4306.txt>.
- [122] Antonio F. Gomez-Skarmeta, Pedro Martinez-Julia, Joao Girao, and Amardeo Sarma. Identity based architecture for secure communication in future internet. In *Proceedings of the 6th ACM Workshop on Digital Identity Management*, pages 45–48, New York, NY, USA, 2010. ACM.
- [123] Jarret Falkner, Michael Piatek, John P. John, Arvind Krishnamurthy, and Thomas Anderson. Profiling a million user dht. In *Proceedings of the 7th ACM SIGCOMM Conference on Internet Measurement*, pages 129–134, New York, NY, USA, 2007. ACM.
- [124] Raul Jimenez, Flutra Osmani, and Björn Knutsson. Sub-second lookups on a large-scale kademlia-based overlay. In *Proceedings of the 2011 IEEE International Conference on Peer-to-Peer Computing (P2P)*, pages 82–91, Washington, DC, USA, 2011. IEEE.
- [125] Maxim Krasnyansky. Virtual Tunnels over TCP/IP networks (VTun), 2011. <http://vtun.sourceforge.net>.
- [126] P. Jokela et al. Host Identity Protocol, 2008. <http://www.ietf.org/rfc/rfc5201.txt>.
- [127] V.P. Kafle, Ruidong Li, D. Inoue, and H. Harai. An integrated security scheme for id/locator split architecture of future network. In *Proceedings of the IEEE International Conference on Communications (ICC) 2012*, pages 5866–5871, Washington, DC, USA, 2012. IEEE.
- [128] Pedro Martinez-Julia and Antonio F. Gomez-Skarmeta. Empowering security and mobility in future networks with an identity-based control plane. *IEICE Transactions on Communications*, E97-B(12):2571–2582, 2014.
- [129] Norm Matloff. Introduction to discrete-event simulation and the simpy language. Technical report, 2008.

- [130] E. Weingartner, H. vom Lehn, and K. Wehrle. A performance comparison of recent network simulators. In *Proceedings of the IEEE International Conference on Communications (ICC'09)*, pages 1–5, Washington, DC, USA, 2009. IEEE.
- [131] Morgan Stanley. Report on Internet Trends, 2010. http://www.morganstanley.com/institutional/techresearch/pdfs/Internet_Trends_041210.pdf.
- [132] C. Perkins et al. IP Mobility Support for IPv4, 2002. <http://www.ietf.org/rfc/rfc3344.txt>.
- [133] Mobile Oriented Future Internet (MOFI), 2011. <http://www.mofi.re.kr>.
- [134] Heeyoung JUNG and Seok Joo KOH. Hinlo: An id/loc split scheme for mobile oriented future internet. In *Proceedings of the Future Network and Mobile Summit 2011*, pages 1–8. International Information Management Corporation, 2011.
- [135] H. Chan et al. Problem statement for distributed and dynamic mobility management. Internet-draft, IETF, 2011.
- [136] OpenLab: extending FIRE testbeds and tools, 2011. <http://www.ict-openlab.eu>.
- [137] Pedro Martinez-Julia and Antonio F. Skarmeta. Beyond the separation of identifier and locator: Building an identity-based overlay network architecture for the future internet. *Computer Networks*, 57(10):2280–2300, 2013.
- [138] Pedro Martinez-Julia and Antonio F. Gomez-Skarmeta. Secure identity-to-identity communications over content-centric networking. In *Proceedings of the 4th IEEE International Conference on Internet Multimedia Systems Architecture and Applications*, pages 1–6, Washington, DC, USA, 2010. IEEE.
- [139] Pedro Martinez-Julia and Antonio F. Skarmeta. An overlay approach to find objects in the future internet. In *Proceedings of the Future Network and Mobile Summit 2013*, pages 1–10. International Information Management Corporation, 2013.
- [140] Pedro Martinez-Julia, Antonio F. Skarmeta, and Alex Galis. Towards a secure network virtualization architecture for the future internet. In Alex Galis and Anastasius Gavras, editors, *The Future Internet*, volume 7858 of *Lecture Notes in Computer Science*, pages 141–152. Springer Berlin Heidelberg, 2013.

BIBLIOGRAPHY

- [141] Pedro Martinez-Julia, Antonio F. Skarmeta, and Ved P. Kaffe. Research and experimentation with the himalis network architecture for future internet. In *Proceedings of the Future Network and Mobile Summit 2012*, pages 1–8. International Information Management Corporation, 2012.
- [142] Pedro Martinez-Julia, Ved P. Kaffe, and Antonio F. Skarmeta. Integrating an identity-based control plane with the himalis network architecture. In *Proceedings of the IEEE Conference on Network Softwarization (NetSoft) 2015*, pages 1–5, Washington, DC, USA, 2015. IEEE.
- [143] Pedro Martinez-Julia, Antonio F. Skarmeta, Hee Young Jung, and Seok Joo Koh. Evaluating secure identification in the mobile oriented future internet (mofi) architecture. In *Proceedings of the Future Network and Mobile Summit 2012*, pages 1–8. International Information Management Corporation, 2012.
- [144] Oauth. <http://oauth.net>.
- [145] Ruidong Li, Masugi Inoue, and Hiroaki Harai. Scalable named information network. In *World Telecommunications Congress (WTC)*, May 4-7 2012.
- [146] Pedro Martinez-Julia and Antonio F. Skarmeta. A lightweight and identity-based network architecture for the internet of things. In *Proceedings of the Sixth International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS) 2012*, pages 711–716, Washington, DC, USA, 2012. IEEE.
- [147] Pedro Martinez-Julia, Elena Torroglosa García, Jordi Ortiz Murillo, and Antonio F. Skarmeta. Evaluating video streaming in network architectures for the internet of things. In *Proceedings of the Seventh International Conference on Innovative Mobile and Internet Services in Ubiquitous Computing (IMIS) 2013*, pages 411–415, Washington, DC, USA, 2013. IEEE.
- [148] RIPE NCC. YouTube Hijacking: A RIPE NCC RIS case study, 2008. <http://www.ripe.net/internet-coordination/news/industry-developments/youtube-hijacking-a-ripe-ncc-ris-case-study>.
- [149] BGPmon. Chinese ISP hijacked 10% of the Internet, 2010. <http://bgpmon.net/blog/?p=282>.

- [150] Ion Stoica, Daniel Adkins, Shelley Zhuang, Scott Shenker, and Sonesh Surana. Internet indirection infrastructure. *IEEE/ACM Transactions on Networking*, 12(2):205–218, 2004.
- [151] V. Fuller, D. Fabrinacci, D. Meyer, and D. Lewis. LISP Alternative Topology (LISP+ALT), 2011. <http://www.ietf.org/id/draft-ietf-lisp-alt-10.txt>.
- [152] S. Brim, N. Chiappa, D. Fabrinacci, V. Fuller, D. Lewis, and D. Meyer. LISP-CONS: A Content distribution Overlay Network Service for LISP, 2008. <http://www.ietf.org/id/draft-meyer-lisp-cons-04.txt>.
- [153] Ehab Al-Shaer, Albert Greenberg, Charles Kalmanek, David A. Maltz, T. S. Eugene Ng, and Geoffrey G. Xie. New frontiers in internet network management. *ACM SIGCOMM Computer Communication Review*, 39(5):37–39, 2009.
- [154] A. Pras, J. Schonwalder, M. Burgess, O. Festor, G.M. Perez, R. Stadler, and B. Stiller. Key research challenges in network management. *IEEE Communications Magazine*, 45(10):104–110, 2007.
- [155] Jeffrey O. Kephart and David M. Chess. The vision of autonomic computing. *IEEE Computer*, 36(1):41–50, 2003.
- [156] Hajer Derbel, Nazim Agoulmine, and Mikaël Salaün. Anema: Autonomic network management architecture to support self-configuration and self-optimization in ip networks. *Computer Networks*, 53(3):418–430, 2009.
- [157] Danilo Ardagna, Marco Comuzzi, Enrico Mussi, Barbara Pernici, and Pierluigi Plebani. Paws: A framework for executing adaptive web-service processes. *IEEE Software*, 24(6):39–46, 2007.
- [158] Massimiliano Rak, Umberto Villano, and Emilio P. Mancini. Autonomic composite-service architecture with mawes. In *Proceedings of the International Conference on Complex, Intelligent and Software Intensive Systems*, pages 1050–1056, Washington, DC, USA, 2010. IEEE Computer Society.
- [159] Yu Cheng, A. Leon-Garcia, and I. Foster. Toward an autonomic service management framework: A holistic vision of soa, aon, and autonomic computing. *IEEE Communications Magazine*, 46(5):138–146, 2008.

BIBLIOGRAPHY

- [160] Pedro Martinez-Julia, Diego R. Lopez, and Antonio F. Gomez-Skarmeta. The gembus framework and its autonomic computing services. In *Proceedings of the International Symposium on Applications and the Internet Workshops*, pages 285–288, Washington, DC, USA, 2010. IEEE Computer Society.

Appendix A

Side Extension: Gateway Overlay Network

In this chapter we discuss a side approach to build a secure and scalable network interconnection mechanism based on a gateway overlay network for address-less, identifier-based networks. The purpose of the overlay network is twofold. First, it provides a mapping between end-node identifiers and gateway identifiers. Second, it is the mechanism used to deliver messages from one network to another. As the underlying network infrastructure, our proposal supports many network infrastructures, such as the current and future Internet.

A.1 Introduction

The Internet has advanced from a network for a small community to the support communication infrastructure that it is today. The Internet Protocol (IP) has facilitated this growth but its limitations to evolve and accomplish many requirements of new and future network and service architectures has caused the commonly called ossification, which has infected the current interconnection mechanisms.

In this way, as specified by RFC 6227 [2], the requirements and challenges presented by the Internet evolution have imposed the necessity of new design goals to define a new routing architecture. Among them we highlight the request for an improved routing scalability, the decoupling of location and identification, the scalable support for mobility, the simplified renumbering, and the routing security.

Specifically concerning security issues, the major events discussed in [148] and [149] demonstrate that it is not impossible to hijack IP addresses in the Border Gateway Protocol

A. Side Extension: Gateway Overlay Network

(BGP), the currently widespread global interconnection routing mechanism in the Internet. Thus, the security is a delicate aspect to be treated by any interconnection mechanism.

Moreover, observing the proliferation of new architectural models for the Future Internet (FI), with very different objectives, we determine that the heterogeneity design principle [4] is outdated and must be enhanced to include not just the hardware and software capabilities but also the network architecture itself. This way, many architectures will coexist in the FI. From the Information-Centric Networking (ICN) side we highlight PURSUIT [56] and CCN [15], and from the *clean-slate* or evolutionary (but not revolutionary) side we highlight HIMALIS [14], MOFI [16], MILSA [46], and the identity-based network architecture we described in [104]. Each of them has its own advantages and disadvantages but, as already commented, it is very probable and desirable to find many of them coexisting in the FI.

Taking into account the introduced challenges introduced in RFC 6227 for new routing architecture together with the expected network heterogeneity, we determine that an interconnection mechanism must meet with the following requirements:

- Routing tables residing in each routing element should be kept to a manageable and agile size even when the address space grows dramatically, which is not unreasonable watching the huge address space of IPv6 and the new identifier-based approaches.
- Interconnection networks must support a simplified and dynamic renumbering together with scalable support for mobility, but this affects to the previous problem.
- Border gateways must prevent the intentional and unintentional communication hijacking or impersonation as an essential security mechanism.
- Addresses should not be needed for the interconnection mechanism to route the messages/packets to accomplish with the enhanced heterogeneity design principle we introduced here and thus allow the development of the emerging network architectures not based on addresses.

To meet with these requirements we design a general purpose interconnection architecture based on the overlay network concept and leveraging its distributed capabilities in a scalable, dynamic, secure, and general purpose network interconnection solution for the Future Internet.

The remainder of this chapter is organized as follows. In Section A.2 we comment the related work and how it meets or not the mentioned problems. Then, in Section A.3 we discuss the proposed interconnection architecture for identifier-based networks and how it

resolves each requirement. In Section A.5 we show a theoretical scalability analysis of the architecture and in Section A.6 we conclude the chapter.

A.2 Related Work

Being one of the most related and similar previous work, the Internet Indirection Infrastructure (I3) [150] proposes a to facilitate the multicast, anycast, and mobility network services by building an overlay network to offer rendezvous-based communication abstractions. Its overlay network is based on the ring topology and routing mechanisms also found in Chord [35], the same approach we propose to use in this chapter to build the overlay network, but the scope is very different. While I3 proposes to include every node in the overlay network, our approach is focused in the interconnection, so its overlay network is built only by the gateways.

Apart from I3, we can find other proposals approaching similar objectives. For instance, LISP-DHT [44] proposes to use a Distributed Hash Table (DHT) as mapping resolution mechanism for LISP [28]. This approach does not cover the general interconnection process, just provides the necessary adjustments in the gateways to permit LISP traffic crossing the global transit network (Internet), but it is useless with general traffic or identifier-based traffic. Nevertheless, our proposal can be seen as a generalization of the concepts found in this approach to be applied to network architectures based on identifiers. This can be clearly seen in Section A.4

There are other variations and alternatives to LISP-DHT that are also based in the overlay concept, mainly LISP-ALT [151] and LISP-CONS [152]. In addition to the same criticism applied LISP-DHT, these approaches do not provide a structured and distributed overlay network.

A.3 Proposed Approach

In this section we discuss the proposed interconnection architecture, which is based on the distributed capabilities of an overlay network instead of the classical routing tables of current interconnection mechanisms. The overlay network is built with the gateways of each edge network and is then used to build a Distributed Hash Table (DHT) to store the mapping between each end-node identifier (ID) and its current gateway identifier (GWID), as well as to deliver messages among edge networks. In contrast with classical routing

A. Side Extension: Gateway Overlay Network

tables, which stores network prefixes, the ID-to-GWID mapping stores individual entries for end-nodes.

We follow the identifier definition given by ITU-T X.1250 [13]: a fixed-size sequence of bytes used to unequivocally identify a node in the whole network. Here, end-node identifiers can be of any nature, like IP addresses or flat labels, with the only restriction that an identifier must point to one and only one node. To support direct multicast delivery, we can use a different profile to break this rule and thus the identifiers can point to more than one node at the same time, but it is out of the scope of the current work.

The ID-to-GWID entries are secured by authorizing mapping updates only to the entities that first established the mapping for the same ID. We call *entity* to any network element like end-nodes or gateways. Moreover, to prevent malicious entities reserve all the identifier space, which in turn is very difficult, each mapping entry has an expiration time, so the responsible entity needs refresh it from time to time.

To perform the message delivery among gateways, the overlay network uses the global transit network, which is a (current or future) network that has all edge networks connected to its borders. Today, the role of the global transit network is played by the worldwide IPv4 and IPv6 infrastructures. To build the overlay network we can use any overlay routing method, like Chord [35] or Kademia [41]. In our proposal we do not bind the overlay network to any concrete algorithm but we start using Chord because of its simplicity, easy extensibility, and the (many) improvements already proposed for it.

In order to build the overlay network and map it to the underlying network we use the Chord approach. On it, each node of the overlay network is assigned a different identifier. Then, the nodes are placed sorted in a virtual ring by applying a circular distance function to their identifiers. Each node keeps a pointer to its previous and next nodes in the ring, as well as a set of *finger* pointers to other nodes that are at 2^n units of the distance metric for n in $0 \dots N$, where N is the length (bits) of the identifiers. Thus, each entry of the DHT is stored in the nearest node to the key of the entry, by using the same circular distance function.

Figure A.1 shows an overview of the architecture. The end-node “A” sends to its gateway a message for “B”. Then, the GW sends the message through the overlay network to “GW.1”, which has the mapping for “B”. “GW.1” resolves the mapping and sends it the message to “GW.5”, also through the overlay network. Finally, “GW.5” sends the message to the end-node “B”. It also shows a communication where a node sends its messages to its current gateway. Then, the messages are sent through the overlay network to the gateway that has the mapping of the destination identifier which, in turn, resolves the mapping to know the gateway of the network where the destination node is connected and then sends

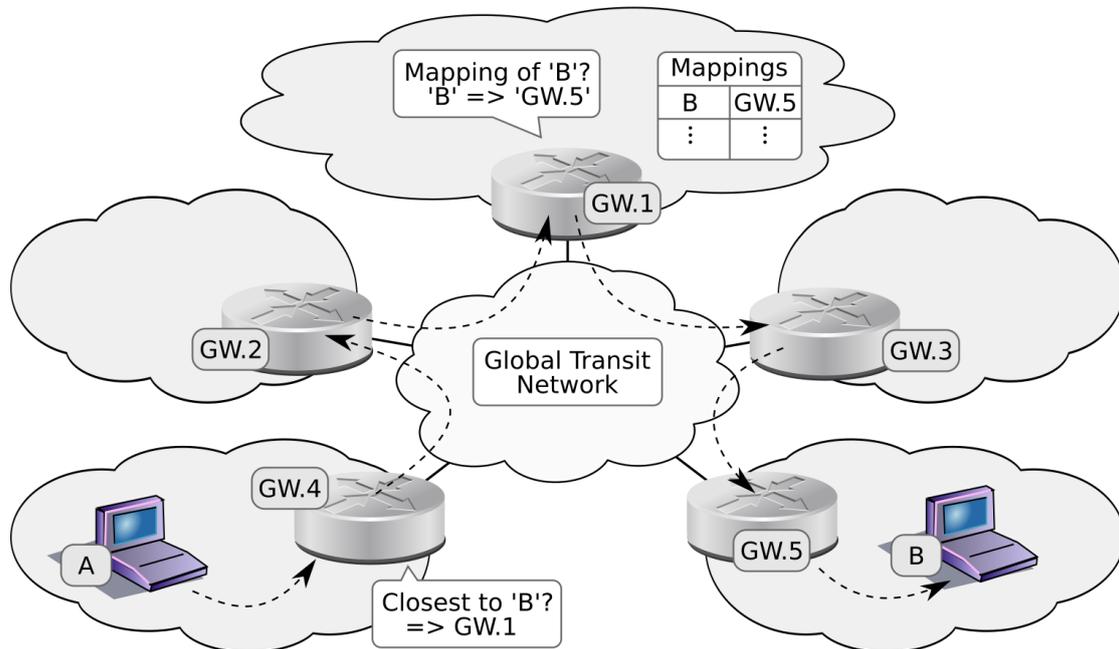


Figure A.1: Overview of the general networking operation.

the messages to it. Finally, the gateway of the destination network receives the information and sends it to the destination node.

In the following subsections we describe how this simple architecture is able to overcome the requirements described in the beginning of this chapter and also give other side benefits over the current interconnection mechanisms.

A.3.1 Security

The security of a network interconnection mechanism is mandatory because of the huge impact of its flaws. Existing approaches do not cover this issue and do not let edge networks to control their own security. Since the network of the future is presupposed to be mobile, the most important network aspect to manage is the identifier-to-locator mapping, which here is seen as ID-to-GWID mapping. We propose to control the overall security in a distributed manner where each gateway is in full control of all security aspects of the end-nodes connected to its network domain, including the ID-to-GWID mapping, but letting end-nodes manage the security aspects so they can give rights over their ID-to-GWID mapping to any gateway.

A. Side Extension: Gateway Overlay Network

When a node is first connected to a gateway it receives a new identifier together with a security token, which is used by the gateway to set the ID-to-GWID mapping in the DHT. If the end-node connects to other network, either by moving or not (multihoming), it prepares its new ID-to-GWID mapping and signs it with the token, so the new gateway can successfully update the existing entry of the DHT with the new mapping information of the end-node.

As a side benefit, using the gateway overlay network to deliver internetwork messages improves the privacy in the sense that do not reveals the current location of a node to any other entity apart from the current gateway to which the node is connected.

A.3.2 Scalability, Performance, and Efficiency

In the current interconnection solutions the border gateways need routing tables of $O(M)$ in size, where M is the number of network prefixes used in the whole network, so it has scalability problems when M grows out of control. Instead, our solution requires internal routing tables of fixed size to reach the nodes of the overlay network and global mapping tables of $O(N/G)$ in size, where N is the number of mapped identifiers and G is the number of gateways connected to the overlay network, and thus to the DHT. Therefore, our solution scales well if the number of gateways grows with the number of end-nodes.

The only important performance problem with overlay networks, and hence DHTs, is that, in principle, they require many hops of the underlying network for each hop of the overlay network. This is not a big problem because, as demonstrated in [122], overlay network efficiency can be easily improved. For instance, the improvement proposed in LPRS [36] can vastly reduce the number of hops both in the overlay and underlying network, raising the performance to a level comparable to hierarchical or graph-based networks.

A.3.3 Mobility by Default and Multihoming

The routing tables used by current interconnection mechanisms only store address prefixes. Even through it is is very efficient for routing traffic of an aggregated address-space, it is unable and would be inefficient for routing unaggregated (independent) addresses to networks where they are not supposed to be. In contrast, the DHT built with the overlay network stores ID-to-GWID mappings for single end-node identifiers so it does not aggregates the end-node IDs in any way. This assumption is essential when a network is

built primarily with mobile nodes, so our approach supports almost transparent mobility (by default) without needing any mobility-specific mechanism.

To support multihoming, the ID-to-GWID mapping is set with many GWIDs and a selection method, such as round-robin, is used to get the final GWID used. The delivery mechanism do not need to be changed.

A.3.4 Generic Interconnection Mechanism

Since the main requirement of some future network architectures is to deliver information based on identifiers that are not only addresses, as introduced above, the architecture discussed here does not presupposes any type of identifier. Initially, the only requirement is that the identifiers used by end-nodes must follow the same scheme and, as in every identification scheme, the identifiers can not be duplicated, so end-nodes can be unequivocally identified in the whole network. This attribute permits our architecture to be used by many different network technologies to interconnect separated networks using the same technology.

For instance, when an ICN provider delivers to its clients or consumers the information items (content) they request, whose delivery is totally based on identifiers, it may need to traverse the global transit network that does not support the delivery of address-less, identifier-based traffic and can benefit from the architecture proposed here to interconnect those networks.

A.4 Instantiation Example

Being one of the main objectives of this work, the interconnection mechanism discussed in previous section is able to interconnect networks of different types. In this section we show how to use it as interconnection mechanism for an identifier-based network architecture in which end-nodes are reached by means of identifiers (as defined by ITU-T X.1250 [13]) of many types, such as plain identifiers (also called flat-labels), hierarchical identifiers, URI-like identifiers, etc.

In a previous work [104], we defined an identity-based network architecture to enhance security and privacy in the Future Internet that is instantiated on top of an identifier-based network. Here we show how to use the approach proposed in this chapter to interconnect networks built with this architecture, but the same principles can be interpolated to any other identifier-based architecture, like the ICN and *clean-slate* approaches mentioned in the introduction of the chapter.

A. Side Extension: Gateway Overlay Network

The identity-to-identity network model we propose in [104] defines a global infrastructure, the Domain Trusted Entity (DTE) infrastructure, which is responsible of managing the security of all network nodes and intermediate in their network operations, primarily the resolution from identities (as also defined by ITU-T X.1250 [13]) to identifiers and the session settings. Once communication parties (entities) have their corresponding identifiers they use an identifier-based overlay network to reach each other, so they finally communicate in an identifier-to-identifier approach. This architecture does not define any underlying network mechanism, it can be instantiated on top of different network infrastructures as we show in [138]. Thus, it is very easy to be integrated with the interconnection mechanism we propose in this chapter.

The first step in the interconnection of such network architectures is the definition of the gateway element, which acts both as traffic gateway and as node of the overlay network. This element is introduced into the network architecture commented above to let it interoperate with the interconnection mechanism. However, the DTE element of each edge network is the responsible to set the ID-to-GWID mapping entries in the overlay DHT, so it will communicate those entries to its corresponding gateway. To maintain the security aspects of the interconnection mechanism, a gateway only accepts mapping updates from the DTE of its domain. Therefore, the clients do not need to manage the mappings, which simplifies client operation as well as security and privacy management.

Once we have integrated the gateway and set the DTE elements to report them the ID-to-GWID mappings, we approach the final communications. In the identity-base network architecture we are integrating, the clients communicate through an overlay network. Thus, the gateway needs to be connected to that overlay network in order to get client packets that go to another network and thus needing to cross the interconnection network. Then, the DTE, which is the entity that gives the communication identifiers to the communication parties, will set the clients to send the gateway their messages that go to outside their edge network.

This can be done in a totally transparent way just returning to the clients new identifiers that represent the gateway when they want to communicate to a distant node, because in the identity-based architecture, the identifiers are not statically bound to clients. That said, Figure A.2 shows an overview of the proposed protocol. The identifiers “AAA” and “BBB” correspond to the two half-sessions, the former is used when sending messages from the Target to the Initiator and the latter is used when sending messages from the Initiator to the Target. In concordance to the protocol, the global operation of a communication is as follows:

A.4 Instantiation Example

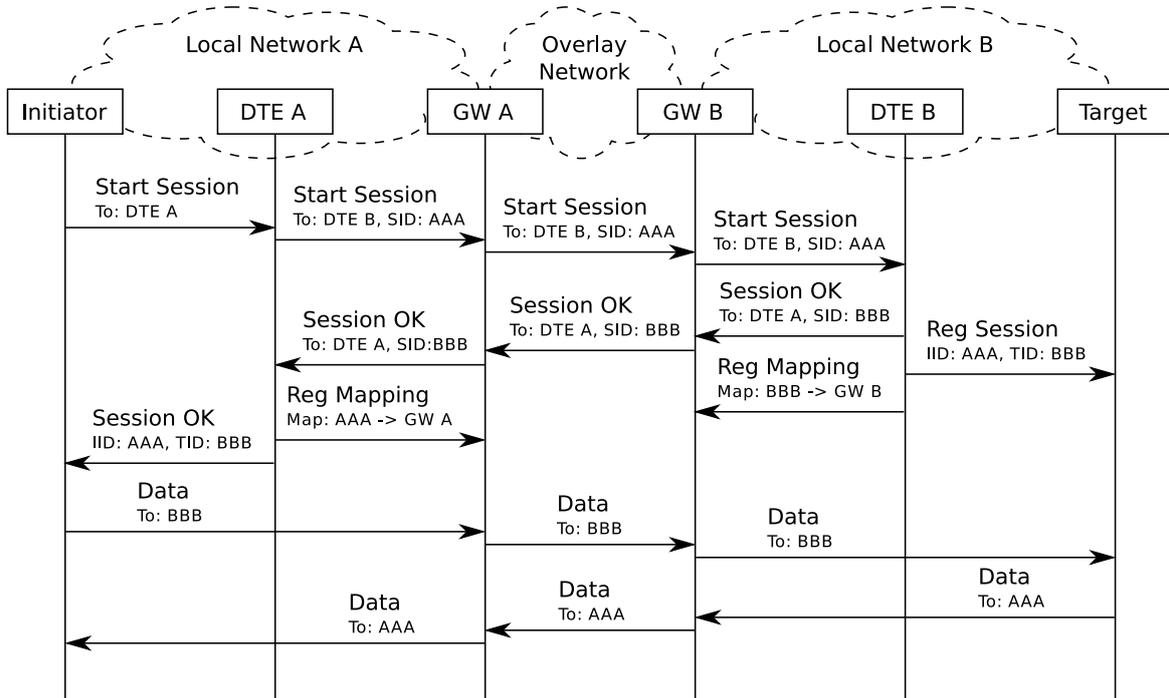


Figure A.2: Protocol overview: Initiator communicates with Target.

1. A initiator entity asks to its DTE to begin a session with an entity of a distant network by means of its identity.
2. The DTE communicates with the destination DTE to send the session start request, which includes the identifier used by the initiator.
3. Assuming that the session is accepted, the DTE of the target sends back the session start response to the DTE of the initiator, which includes the identifier used by the target.
4. Both DTEs of the initiator and destination entities set their corresponding ID-to-GWID mapping in the DHT of the interconnection network.
5. The DTE of the initiator entity contacts with its gateway to assign a new identifier associated to the target identifier, and communicates this new identifier to the initiator entity.
6. The initiator entity starts to send messages/packets to the destination identifier returned by its DTE, which corresponds to the target entity but the messages are actually received by the gateway.

A. Side Extension: Gateway Overlay Network

7. The gateway receives the messages and follows the same process described in the example and in the previous section to deliver the messages to the destination gateway.
8. The destination gateway uses the overlay network built on top of its edge network to deliver the messages to the target entity.
9. The target entity follows this same process to send its responses (messages) back to the initiator entity.

This process shows the main steps followed by the identity-based architecture together with the necessary interactions to integrate it with the interconnection overlay network proposed in this chapter. The global security is achieved as described in the previous section but here is managed by the DTE infrastructure, which is the responsible entity for this task in the identity-based architecture.

A.5 Scalability Analysis

In this section we analyze the scalability of the proposed interconnection mechanism and compare it with the classical interconnection approach based on IP routing.

For each approach we built an equation based on common parameters, so both approaches are totally comparable. Each equation gives the time (in milliseconds) spent in the whole communication operation between two distant end-nodes. Thus, we used the following parameters:

- α : Cost, in milliseconds, to process an entry of a routing or hash table.
- N_p : Total number of address prefixes.
- N_a : Average number of addresses per each prefix.
- N_{GW} : Number of gateways in the network.
- $T_{A \rightarrow B}$: Cost, in milliseconds, to communicate from “A” to “B”, independently of the intermediate hops.

We need to remark that the analysis contemplates a variable number of addresses. Therefore, N_p and N_a do not depend on each other. Thus, the total number of addressed nodes is $N_p \times N_a$.

In the following subsections we describe the equations for each analyzed approach and then discuss the results obtained with them.

A.5.1 Classical Interconnection Approach

As commented above, we are comparing the architecture discussed in this chapter with the widely used (classical) interconnection approach in the Internet. We aim to demonstrate the advantages and disadvantages of the proposed approach, expecting to give enough arguments to determine its feasibility. Thus, we first model an equation to represent the classical interconnection approach.

This approach starts when an end-node sends a message to its gateway [$T_{CLI \rightarrow GW}$]. Then, the gateway looks for the destination gateway in its routing table [αN_p] and sends the message to it [$T_{GW \rightarrow GW}$]. The destination gateway looks its routing table to know which link use to send the message [αN_p] and, finally, sends the message to the destination [$T_{CLI \rightarrow GW}$]. The final equation [C_{old}] is as follows:

$$\begin{aligned}
C_{old} &= T_{CLI \rightarrow GW} \\
&+ \alpha N_p + T_{GW \rightarrow GW} + \alpha N_p \\
&+ T_{CLI \rightarrow GW}
\end{aligned} \tag{A.1}$$

It is worth to notice that this equation does not uses the parameter N_a , because the behavior and scalability of the classical interconnection approach does not depend on the number of addresses per each prefix. Also, it does not use the parameter N_{GW} because the number of gateways in the network does not affect to the message delivery.

A.5.2 Proposed Interconnection Approach

In contrast with the approach described above, this approach can not be represented with only one equation. Instead, we describe it with the upper limit and lower limit of the time spent to deliver a message. Before, we denote that a single hop in the overlay network is translated to as much as $\log_2(N_{GW})$ hops in the underlying network and that each gateway has as much as $\frac{N_p N_a}{N_{GW}}$ entries in the mapping table.

The upper limit equation, as above, starts when an end-node sends a message to its gateway [$T_{CLI \rightarrow GW}$]. Then, this gateway finds the next hop in the overlay network [$\alpha \log_2(N_p N_a)$] towards the gateway that hash the mapping and sends the message to it [$\log_2(N_{GW}) \times T_{GW \rightarrow GW}$]. This part multiplies the maximum number of hops from a node pair in the overlay [$\log_2(N_{GW})$] by the cost of each hop [$T_{GW \rightarrow GW}$]. Now, this intermediate gateway finds the destination gateway in the mapping using the destination identifier [$\alpha \times \frac{N_p N_a}{N_{GW}}$]. After that, the message is again sent through the overlay network to the

A. Side Extension: Gateway Overlay Network

destination gateway $[\log_2(N_{GW}) \times T_{GW \rightarrow GW} + \alpha \log_2(N_p N_a)]$. Finally, the gateway of the destination network delivers the message to the destination node $[T_{CLI \rightarrow GW}]$. The final equation for the upper limit of this approach $[C_{\max}]$ is as follows:

$$\begin{aligned}
 C_{\max} &= T_{CLI \rightarrow GW} \\
 &+ \alpha \log_2(N_p N_a) + \log_2(N_{GW}) \times T_{GW \rightarrow GW} \\
 &+ \alpha \times \frac{N_p N_a}{N_{GW}} \\
 &+ \log_2(N_{GW}) \times T_{GW \rightarrow GW} + \alpha \log_2(N_p N_a) \\
 &+ T_{CLI \rightarrow GW}
 \end{aligned} \tag{A.2}$$

In contrast with the equation for the current approach, this equation uses all parameters because the message delivery through the overlay network depends on the number of total addresses $[N_p \times N_a]$ and the number of gateways forming part of the interconnection network and thus the DHT $[N_{GW}]$.

The lower limit in the operation of the architecture is very similar to the upper limit operation. The main difference is that the gateways of the overlay network can be reached in just one hop in the underlying network, which represent some specific circumstances (e.g. when the gateways are neighbors) or certain performance improvements.

Thus, the lower limit equation is equal to the upper limit equation but with changes in the communications between overlay network nodes. Here, the overlay network only takes $T_{GW \rightarrow GW}$, but all other elements of the equation are the same as in the previous equation. The final equation for the lower limit of this approach $[C_{\min}]$ is as follows:

$$\begin{aligned}
 C_{\min} &= T_{CLI \rightarrow GW} \\
 &+ \alpha \log_2(N_p N_a) + T_{GW \rightarrow GW} \\
 &+ \alpha \times \frac{N_p N_a}{N_{GW}} \\
 &+ T_{GW \rightarrow GW} + \alpha \log_2(N_p N_a) \\
 &+ T_{CLI \rightarrow GW}
 \end{aligned} \tag{A.3}$$

Once defined the equations, in the following subsection, we show and discuss the results of the analysis.

Table A.1: Parameter values and ranges.

Parameter	Default	Min	Max
α	0.1	0.01	0.2
N_p	1000	100	2000
N_a	256	100	1000
N_{GW}	250	100	1000
$T_{CLI \rightarrow GW}$	10	-	-
$T_{GW \rightarrow GW}$	5	-	-

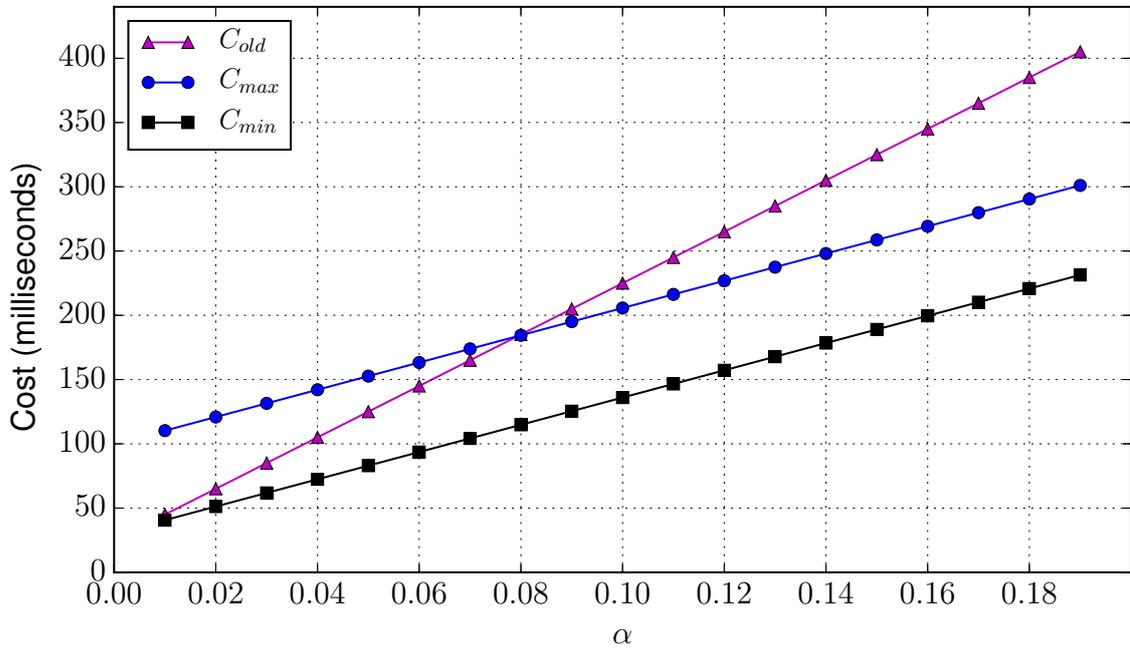


Figure A.3: Analysis results, variation of α .

A.5.3 Results

Using the equations defined above, we now get numerical results using the parameters described above with the values and ranges shown in Table A.1, which are defined to show the crossing points of the approaches in the resulting plots. Each plot analyzes the variation of a parameter within the defined range and keeps the others to their default value.

Figure A.3 shows the results for the variation of α in which we see that for low values (under 0.08) the old approach is better than the upper limit of the new approach and even than the lower limit of the new approach when the α values are very low.

A. Side Extension: Gateway Overlay Network

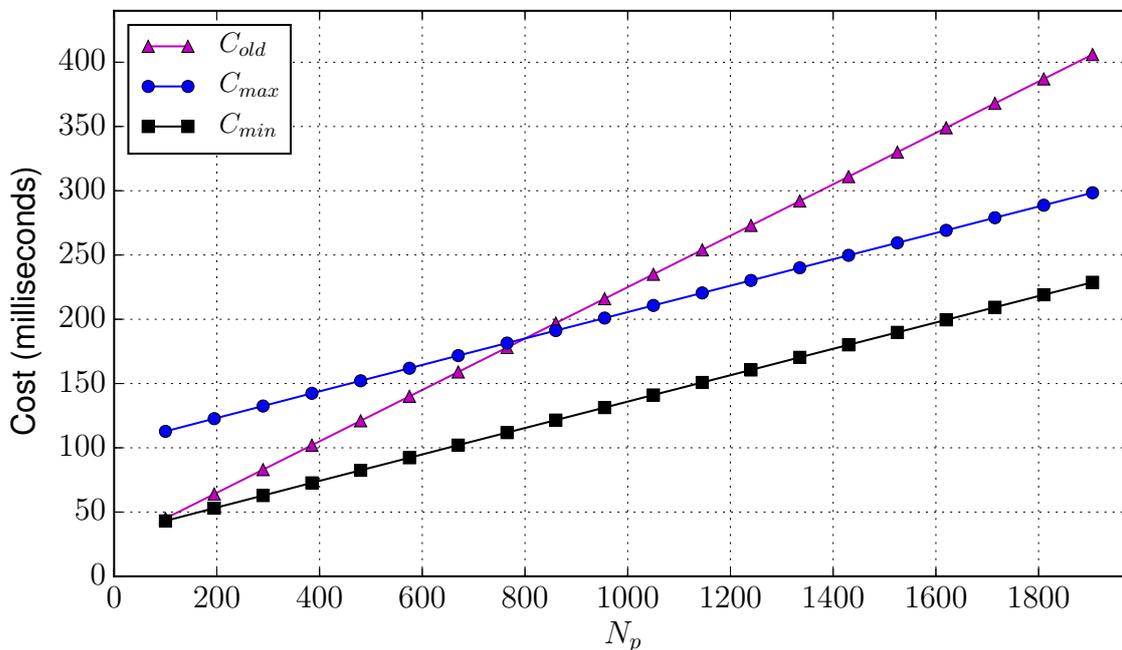


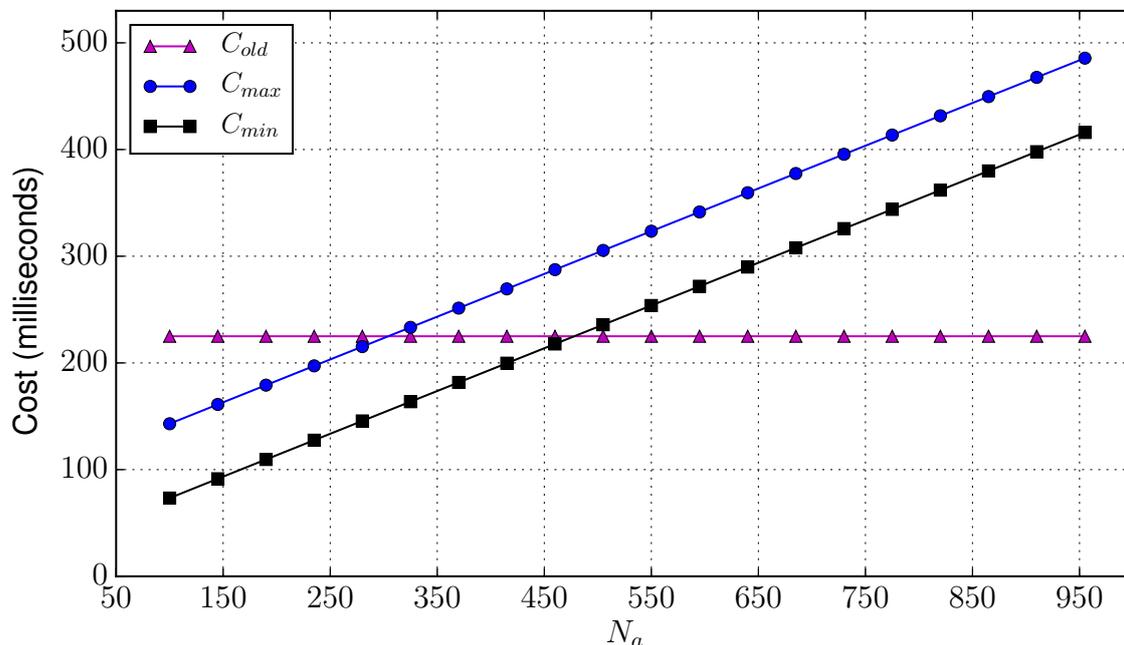
Figure A.4: Analysis results, variation of N_p .

Figure A.4 shows the results for the variation of N_p . As in the equations this variable is typically associated with α , the resulting plot is almost the same, so the discussion about the results is the same given above. Here, we highlight that for values under 1000 prefixes, the total delivery cost is kept under 149 ms for the lower limit of our proposal, so it can overcome the old approach up to 40%.

Figure A.5 shows the results for the variation of N_a . As the old approach aggregates addresses in prefixes, its plot is totally flat. Instead, our approach is linearly dependant to the number of addresses per prefix. The strong point to highlight in our approach is that for low number of addresses per prefix it improves the classical approach.

Finally, Figure A.6 shows the results for the variation of N_{GW} . Again, the old approach plot is flat because it is independent of the number of gateways in the network. Here, the strong point of our approach to highlight is that the results of the proposed approach decrease with the number of gateways in the network, improving the old approach between a 50% and a 76%. This behavior is due to the search in the mapping tables, which are huge for low number of gateways but small for high number of gateways.

From the results discussed above we can determine that our proposal overcomes the classical approach from 1000 prefixes and 250 gateways onwards, but loses when the

Figure A.5: Analysis results, variation of N_a .

number of addresses per prefix grows without growing the number of gateways. The weak point of our architecture is clearly the processing time. Even though when the α value increases, our proposal clearly beats the classical approach, in the future, this parameter is not willing to increase but decrease, so we need to investigate how to improve our proposal in this respect.

Finally, we need to notice that the remaining parameters are set to huge values in comparison to actual values obtained from the network. We used 5 ms for gateway-to-gateway traffic and 10 ms for client-to-gateway traffic to enlarge the differences between the analyzed approaches but we also run the analysis using other values to ensure that for more realistic parameter values the differences in the results are proportional, so the comparison and results discussed here are still valid when those parameters change while retaining their relations.

A.6 Conclusions

Throughout the chapter we discussed how to use a gateway overlay network as a secure and scalable interconnection architecture for the Future Internet, specifically for identifier-based

A. Side Extension: Gateway Overlay Network

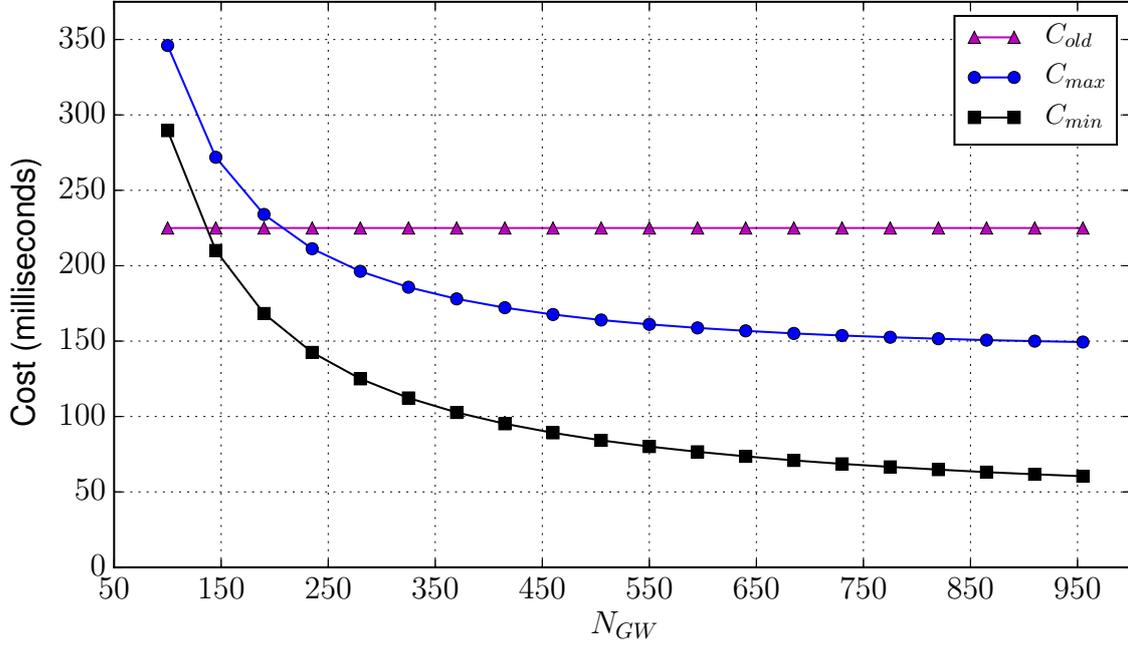


Figure A.6: Analysis results, variation of N_{GW} .

network architectures. Thus, the requirements set in the first section are, in brief, covered as follows:

- The scalability is provided by using the DHT built by the gateways to store the ID-to-GWID mappings.
- Since the mappings are set for individual IDs, the architecture supports fine-grain routing and helps end-node mobility.
- The security against communication hijacking or unauthorized impersonation is ensured by signing the entries of the DHT that resolves IDs to GWIDs. Also, the autonomic nature of the overlay network prevents a malfunctioning gateway to affect the network.
- The necessary genericity to achieve architecture heterogeneity is achieved by not fixing the identifier type used in the delivery.

After that, we demonstrated that the scalability of the proposed approach overcomes the classical approach for some realistic parameters.

Appendix B

The GAIA Experimentation Infrastructure

The GAIA Extended Research Infrastructure is located at the southeast of Spain. It targets the research of Future Internet architectures and comprises several facilities from the University of Murcia and the Spanish government. It offers a vertical infrastructure, composed of a backend with high capacity of data storage, communication, and processing, together with a frontend with an extended set of multidisciplinary testbeds, deployments, and living labs for the ubiquitous monitoring, sensing, and processing. That said, it offers a highly flexible framework for experimentation with architectures and protocols for the Future Internet. In fact, it has been used in many research projects to evaluate their outputs from the communications and telematics point of view.

B.1 Infrastructure Description

The GAIA Extended Research Infrastructure, as illustrated in Figure B.1, is composed of several deployments, living labs, and multidisciplinary scenarios based on mobile communications, ubiquitous computing and Internet of Things (IoT). All of them are focused on the Future Internet and supported by our backend infrastructure.

The backend, as depicted in the inner sub-figure of Figure 1, is built by a computer cluster of 22 nodes with high processing capacity, connected to a storage area network (SAN) to provide high capacity/speed remote storage based on the Fiber Channel technology. The SAN is provisioned with a total capacity of 2 TiB.

In addition, the GAIA infrastructure offers many experimentation machines, interconnected by a Gigabit Ethernet (GE) network. Moreover, the experimentation

B. The GAIA Experimentation Infrastructure

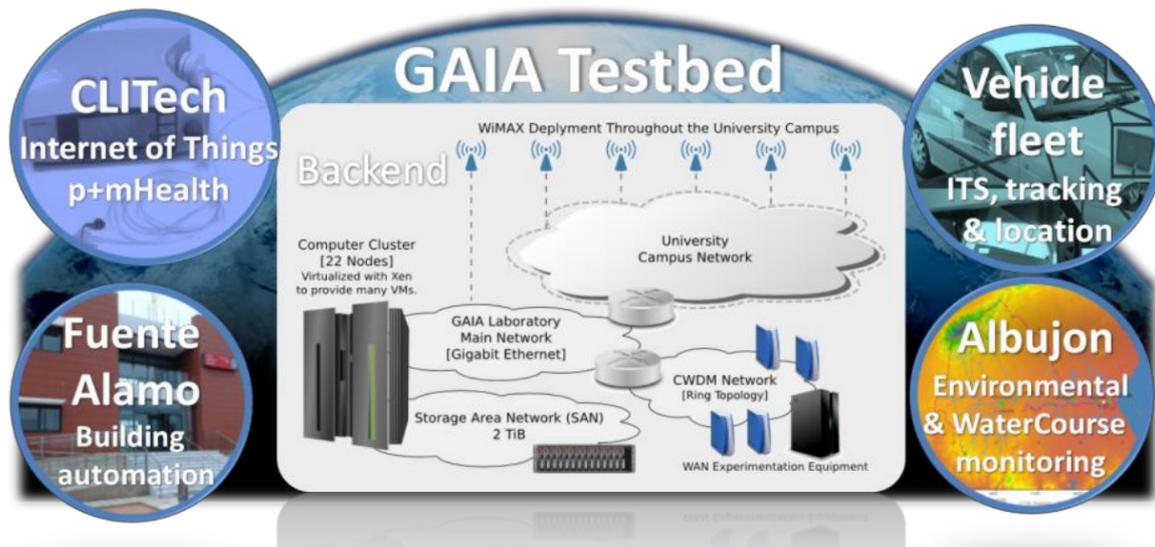


Figure B.1: GAIA extended infrastructure with the living labs and deployments.

infrastructure has a dedicated CWDM network to evaluate elements destined to backbone networks. Furthermore, the infrastructure also has a WiMAX network deployed throughout the university campus and connected to the main network by dedicated VLANs.

Apart from the central GAIA facilities, the research infrastructure is extended with a set of multidisciplinary deployments focused on sensor networks and monitoring platforms. They target Intelligent Transport Systems, environmental monitoring, mobile health, and buildings automation. Specifically, these deployments are:

- **Building Automation:** Many facilities, including a complete building from the Fuenté Alamo Technology Park (FATP), are managed by over 50 multiprotocol cards developed by our research lab. These platforms are focused on energy sustainability to reach positive-net building with the deployed solar power plants.
- **Internet of Things (IoT):** Our building at FATP also presents an IoT network with a weather station, 20 parking spots, 20 air-quality sensors located at the streetlights, 50 environmental sensors (temperature, humidity, pressure), and 50 activity sensors.
- **Environmental Monitoring:** We have a real-time monitoring system for the main drainage basin of the watercourse (in Albujon), covering an area of $550km^2$. These platforms are mainly focused on watercourse improvement and flood forecast.

B.1 Infrastructure Description

- **Clinical Technology:** The clinical research lab is also located at FATP, in two dedicated rooms: an Ambient Assisted Living room with 15 personal and wearable clinical devices, and a hospital room with 5 patient monitors.
- **Vehicle fleet:** The vehicle fleet of the University of Murcia, composed of 48 cars destined to personnel mobility and various internal services of the university, has integrated a platform for location and tracking developed and interconnected with our research lab. For instance, this permits us to track the vehicles when experimenting with WiMAX mobility scenarios.

Finally, it is worth to mention that this infrastructure has been used for many EU projects, like SWIFT and DAIDALOS, and it is currently used for experimentation within the IoT6 project and prospectively within the OpenLab project.

B. The GAIA Experimentation Infrastructure

Appendix C

Schemas for the Network Object Discovery Functions

Here we show the schemas for the ontology used in the discovery functions and the protocol defined in Avro, as well as examples for both. The ontology and object instances are serialized in Turtle while the Avro and interaction (request/response) messages are serialized in JSON. Figure C.1 shows the serialization of the ontology, Figure C.2 shows the description of a simple object, a color printer, including the security constraints it has. Figure C.3 shows the description of another simple object, but in this case it is the information of a person and makes use of extensions to the ontology (e.g. DUL). Figure C.4 shows the description of a black and white printer as noted in the capabilities it exposes, which are different from the color printer described before. Figure C.5 shows a different simple object, a color scanner, as clearly described by the capabilities it exposes.

Regarding queries and protocols, Figure C.6 shows a simple query to retrieve the information object described in Figure C.3, as stated by the conditions included to *express* the concept *identifier* with the value: *"pedromj"*. On the other hand, Figure C.7 shows the necessary query to retrieve the description of the color printer and it is instructed to retrieve the identifier, location, interface, and necessary parameters. With this information, the retriever will be able to communicate with such printer. Figures C.8 and C.9 show the responses of the queries. They are in JSON, as it is better for the communication of such information without the schema, which is supposed to be known by both communication parties. Finally, Figure C.10 shows the schema used in the discovery protocol when instanced with Avro.

C. Schemas for the Network Object Discovery Functions

```
1 @prefix : <http://odp-project.org/ontology#> .
2 @prefix rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
3 @prefix rdfs: <http://www.w3.org/2000/01/rdf-schema#> .
4 @prefix owl: <http://www.w3.org/2002/07/owl#> .
5 @prefix dul: <http://www.lov-cnr.it/ontologies/DUL.owl#> .
6 @prefix dc: <http://purl.org/dc/elements/1.1/> .
7 @prefix dcterms: <http://purl.org/dc/terms/> .
8 @prefix foaf: <http://xmlns.com/foaf/0.1/> .
9 @prefix wsp: <http://www.w3.org/2004/08/20-ws-pol-pos/ns> .
10 @prefix spitfire: <http://spitfire-project.eu/ontology/ns> .
11 @base <http://odp-project.org/ontology> .
12
13 <http://odp-project.org/ontology>
14   a owl:Ontology ;
15   dc:creator foaf:me ;
16   dc:creator "Pedro Martinez-Julia (pedromj@um.es)" ;
17   dc:rights "Copyright 2012 - Pedro Martinez-Julia (pedromj@um.es)" ;
18   dc:title "ODIN" ;
19   dc:identifier "http://odp-project.org/ontology" ;
20   dcterms:license "MIT/X11" .
21
22 # Classes
23 #
24
25 :NetworkObject
26   a owl:Class ;
27   rdfs:subClassOf dul:Object ;
28   rdfs:isDefinedBy : .
29
30 :Network
31   a owl:Class ;
32   rdfs:subClassOf :NetworkObject ;
33   owl:equivalentClass spitfire:SensorNetwork ;
34   rdfs:isDefinedBy : .
35
36 :Interface
37   a owl:Class ;
38   rdfs:isDefinedBy : .
39
40 :Capability
41   a owl:Class ;
42   rdfs:isDefinedBy : .
43
44 :Retrievable
45   a owl:Class ;
46   rdfs:subClassOf :Interface ;
47   rdfs:subClassOf dul:InformationObject ;
48   rdfs:isDefinedBy : .
49
50 :Subscribable
51   a owl:Class ;
52   rdfs:subClassOf :Interface ;
53   rdfs:isDefinedBy : .
54
55 :Communicable
56   a owl:Class ;
57   rdfs:subClassOf :Interface ;
58   rdfs:isDefinedBy : .
59
60 :Operation
61   a owl:Class ;
62   rdfs:isDefinedBy : .
63
64 #
65 # Relations
66 #
67
68 :belongsTo
69   a owl:ObjectProperty ;
70   rdfs:subPropertyOf spitfire:belongsToNetwork ;
71   rdfs:domain :NetworkObject ;
72   rdfs:range :Network ;
73   rdfs:isDefinedBy : .
74
75 :exposes
76   a owl:ObjectProperty ;
77   rdfs:domain :NetworkObject ;
78   rdfs:range :Interface ;
79   rdfs:isDefinedBy : .
80
81 :offers
82   a owl:ObjectProperty ;
83   rdfs:domain :NetworkObject ;
84   rdfs:range :Capability ;
85   rdfs:isDefinedBy : .
86
87 :controlledBy
88   a owl:ObjectProperty ;
89   rdfs:domain :NetworkObject ;
90   rdfs:range wsp:Policy ;
91   rdfs:isDefinedBy : .
92
93 :hasMany
94   a owl:ObjectProperty ;
95   rdfs:domain :Interface ;
96   rdfs:range :Operation ;
97   rdfs:isDefinedBy : .
98
99 :provides
100   a owl:ObjectProperty ;
101   rdfs:domain :Operation ;
102   rdfs:range :Capability ;
103   rdfs:isDefinedBy : .
104
105 :associatedTo
106   a owl:ObjectProperty ;
107   rdfs:domain :Capability ;
108   rdfs:range dul>Action ;
109   rdfs:isDefinedBy : .
110
111 # Literal Attributes
112 #
113
114 :identifier
115   a owl:ObjectProperty ;
116   rdfs:domain :NetworkObject ;
117   rdfs:range rdfs:Literal ;
118   rdfs:isDefinedBy : .
119
120 :location
121   a owl:ObjectProperty ;
122   rdfs:domain :NetworkObject ;
123   rdfs:range rdfs:Literal ;
124   rdfs:isDefinedBy : .
125
126 :name
127   a owl:ObjectProperty ;
128   rdfs:domain :Capability ;
129   rdfs:range rdfs:Literal ;
130   rdfs:isDefinedBy : .
131
132 :signature
133   a owl:ObjectProperty ;
134   rdfs:domain :Operation ;
135   rdfs:range rdfs:Literal ;
136   rdfs:isDefinedBy : .
```

Figure C.1: Ontology schema in Turtle.

```

1 @prefix odin: <http://odin-project.org/ontology#> .
2 @prefix wsp: <http://www.w3.org/2004/08/20-ws-pol-pos/ns> .
3 @prefix wsse: <http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd> .
4
5 <urn:object:printer1>
6   a odin:NetworkObject ;
7   odin:identifier "urn:networkobject:printer1" ;
8   odin:location "urn:location:printer1" ;
9   odin:belongsTo <urn:network:a> ;
10  odin:offers <urn:capability:print> ;
11  odin:offers <urn:capability:color> ;
12  odin:exposes <urn:interface:print> ;
13  odin:controlledBy [
14    a wsp:Policy ;
15    wsp:require [
16      a wsse:SecurityToken ;
17      wsp:Usage wsp:Required ;
18      wsse:tokenType wsse:Kerberosv5TGT
19    ]
20 ] .

```

Figure C.2: Description of network object 1, color printer.

```

1 @prefix odin: <http://odin-project.org/ontology#> .
2 @prefix wsp: <http://www.w3.org/2004/08/20-ws-pol-pos/ns> .
3 @prefix wsse: <http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd> .
4 @prefix dul: <http://www.loa-cnr.it/ontologies/DUL.owl#> .
5 @prefix dcterms: <http://purl.org/dc/terms/> .
6
7 <urn:object:information1>
8   a odin:NetworkObject ;
9   odin:identifier "urn:content:information1" ;
10  odin:location "http://webs.um.es/pedromj/index.html" ;
11  odin:belongsTo <urn:network:a> ;
12  odin:offers <urn:capability:information> ;
13  odin:exposes <urn:interface:content> ;
14  odin:controlledBy [
15    a wsp:Policy ;
16    wsp:require [
17      a wsse:SecurityToken ;
18      wsp:Usage wsp:Required ;
19      wsse:tokenType wsse:Kerberosv5TGT
20    ]
21 ] ;
22 dul:expressesConcept [ dcterms:identifier "Pedro Martinez-Julia" ] ;
23 dul:expressesConcept [ dcterms:identifier "pedromj" ] ;
24 dul:expressesConcept [ dcterms:identifier "webpage" ] .

```

Figure C.3: Description of network object 2, information item.

```

1 @prefix odin: <http://odin-project.org/ontology#> .
2 @prefix wsp: <http://www.w3.org/2004/08/20-ws-pol-pos/ns> .
3 @prefix wsse: <http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd> .
4
5 <urn:object:printer2>
6   a odin:NetworkObject ;
7   odin:identifier "urn:networkobject:printer2" ;
8   odin:location "urn:location:printer2" ;
9   odin:belongsTo <urn:network:a> ;
10  odin:offers <urn:capability:print> ;
11  odin:offers <urn:capability:duplex> ;
12  odin:exposes <urn:interface:print> ;
13  odin:controlledBy [
14    a wsp:Policy ;
15    wsp:require [
16      a wsse:SecurityToken ;
17      wsp:Usage wsp:Required ;
18      wsse:tokenType wsse:Kerberosv5TGT
19    ]
20 ] .

```

Figure C.4: Description of network object 3, black and white printer.

C. Schemas for the Network Object Discovery Functions

```
1 @prefix odin: <http://odin-project.org/ontology#> .
2 @prefix wsp: <http://www.w3.org/2004/08/20-ws-pol-pos/ns> .
3 @prefix wsse: <http://docs.oasis-open.org/wss/2004/01/oasis-200401-wss-wssecurity-secext-1.0.xsd> .
4
5 <urn:object:scanner1>
6   a odin:NetworkObject ;
7   odin:identifier "urn:networkobject:scanner1" ;
8   odin:location "urn:location:scanner1" ;
9   odin:belongsTo <urn:network:a> ;
10  odin:offers <urn:capability:scan> ;
11  odin:offers <urn:capability:color> ;
12  odin:exposes <urn:interface:scan> ;
13  odin:controlledBy [
14    a wsp:Policy ;
15    wsp:require [
16      a wsse:SecurityToken ;
17      wsp:Usage wsp:Required ;
18      wsse:tokenType wsse:Kerberosv5TGT
19    ]
20  ] .
```

Figure C.5: Description of network object 4, color scanner.

```
1 PREFIX odin: <http://odin-project.org/ontology#>
2 PREFIX dul: <http://www.loa-cnr.it/ontologies/DUL.owl#>
3 PREFIX dcterms: <http://purl.org/dc/terms/>
4 SELECT ?identifier ?location ?interface
5 WHERE {
6   ?x a odin:NetworkObject ;
7   odin:identifier ?identifier ;
8   odin:location ?location ;
9   odin:offers [ odin:name "information" ] ;
10  odin:exposes [
11    odin:operatedThrough [
12      odin:name ?interface
13    ]
14  ] ;
15  dul:expressesConcept [ dcterms:identifier "pedromj" ] .
16 }
```

Figure C.6: Query (SPARQL) to retrieve the information object description.

```
1 PREFIX odin: <http://odin-project.org/ontology#>
2 PREFIX dul: <http://www.loa-cnr.it/ontologies/DUL.owl#>
3 PREFIX dcterms: <http://purl.org/dc/terms/>
4 SELECT ?identifier ?location ?interface ?parameter
5 WHERE {
6   ?x a odin:NetworkObject ;
7   odin:identifier ?identifier ;
8   odin:location ?location ;
9   odin:offers [ odin:name "color_support" ] ;
10  odin:offers [ odin:name "print" ] ;
11  odin:exposes [
12    odin:operatedThrough [
13      odin:name ?interface ;
14      odin:provides [
15        odin:associatedTo [
16          dul:executesTask [
17            dul:isTaskDefinedIn [
18              dul:usesConcept [
19                dul:hasParameter [ dcterms:identifier ?parameter ]
20            ]
21          ]
22        ]
23      ]
24    ]
25  ] .
26 } .
27 }
```

Figure C.7: Query (SPARQL) to retrieve the printer object description.

```

1 {
2   'interface': 'get',
3   'identifier': 'urn:content:information1',
4   'location': 'http://webs.um.es/pedromj/index.html'
5 }

```

Figure C.8: Query response in JSON of the information object.

```

1 {
2   'interface': 'printFile',
3   'identifier': 'urn:networkobject:printer1',
4   'parameter': ['filename', 'printing_parameters', 'subject'],
5   'location': 'urn:location:printer1'
6 }

```

Figure C.9: Query response in JSON of the printer object.

```

1 [
2   {
3     "name": "MessageType",
4     "type": "enum",
5     "symbols": [
6       "RegisterRequest",
7       "RegisterResponse",
8       "FindObjectRequest",
9       "FindObjectResponse"
10    ]
11  },
12  {
13    "name": "Message",
14    "type": "record",
15    "fields": [
16      { "name": "mtype", "type": "MessageType" },
17      { "name": "body", "type": "bytes" }
18    ]
19  },
20  {
21    "name": "RegisterRequest",
22    "type": "record",
23    "fields": [
24      { "name": "identifier", "type": "string" },
25      { "name": "description", "type": "string" }
26    ]
27  },
28  {
29    "name": "RegisterResponse",
30    "type": "record",
31    "fields": [
32      { "name": "identifier", "type": "string" },
33      { "name": "response", "type": "string" }
34    ]
35  },
36  {
37    "name": "FindObjectRequest",
38    "type": "record",
39    "fields": [
40      { "name": "identifier", "type": "string" },
41      { "name": "query", "type": "string" }
42    ]
43  },
44  {
45    "name": "FindObjectResponse",
46    "type": "record",
47    "fields": [
48      { "name": "identifier", "type": "string" },
49      { "name": "object", "type": "string" }
50    ]
51  }
52 ]

```

Figure C.10: Protocol schema for Avro in JSON.

C. Schemas for the Network Object Discovery Functions

Appendix D

Protocols Represented in HPSL for AVISPA

First we show the HPSL source code for the mobility protocol, differentiating the blocks for the *roles*, the *goal*, and the *environment* execution. There is one role for each entity involved in the protocol, as described in Chapter 4.

```
1  % -----
2
3  role alice (
4      A, DTE3, DTE1 : agent,
5      TokenADTE1, AsesID, Aloc : text,
6      KeyA, KeyDTE1 : public_key,
7      SendDTE3, RecvDTE3 : channel(dy),
8      Hash : hash_func
9  )
10 played_by A
11 def=
12     local
13         State : nat,
14         OK : text
15     const
16         auth_alice_dtel : protocol_id
17     init
18         State := 0
19     transition
20         1. State = 0 /\ RecvDTE3(start) =>
21            State' := 1 /\ SendDTE3({{TokenADTE1.AsesID.Aloc}_inv(KeyA)}_KeyDTE1)
22         2. State = 1 /\ RecvDTE3({{Hash(TokenADTE1).OK'}_inv(KeyDTE1)}_KeyA) =>
23            State' := 2 /\ secret(TokenADTE1, secret_alice_dtel, {A, DTE1})
24            /\ wrequest(A, DTE1, auth_alice_dtel, TokenADTE1)
25 end role
26
27 % -----
28
29 role dte3 (
30     DTE3, GW3, A, DTE1 : agent,
31     TokenDTE13 : text,
32     KeyDTE3, KeyGW3, KeyA, KeyDTE1 : public_key,
33     SendA, RecvA, SendGW3, RecvGW3, SendDTE1, RecvDTE1 : channel(dy),
34     Hash : hash_func
35 )
36 played_by DTE3
37 def=
38     local
39         State : nat,
```

D. Protocols Represented in HPSL for AVISPA

```

40     TokenADTE1, AsesID, Alloc, BsesID, Bloc, OK : text
41     const
42     auth_dte1_dte3, secret_dte1_dte3 : protocol_id
43     init
44     State := 0
45     transition
46     1. State = 0 /\ RecvA({{TokenADTE1'.AsesID'.Alloc'}_inv(KeyA)}_KeyDTE1) =|>
47     State' := 1 /\ SendDTE1({{TokenDTE13.{{TokenADTE1'.AsesID'.Alloc'}_inv(KeyA)}_KeyDTE1}_inv(KeyDTE3)}_KeyDTE1)
48     2. State = 1 /\ RecvDTE1({{Hash(TokenDTE13).AsesID'.Alloc'.BsesID'.Bloc'}_inv(KeyDTE1)}_KeyDTE3) =|>
49     State' := 2 /\ SendGW3({{AsesID'.Alloc'.BsesID'.Bloc'}_inv(KeyDTE3)}_KeyGW3)
50     3. State = 2 /\ RecvDTE1({{Hash(TokenDTE13).{{Hash(TokenADTE1).OK'}_inv(KeyDTE1)}_KeyA}_inv(KeyDTE1)}_KeyDTE3) =|>
51     State' := 3 /\ SendA({{Hash(TokenADTE1).OK'}_inv(KeyDTE1)}_KeyA)
52     /\ secret(TokenDTE13, secret_dte1_dte3, {DTE3, DTE1})
53     /\ wrequest(DTE3, DTE1, auth_dte1_dte3, TokenDTE13)
54 end role
55
56 % -----
57
58 role dte1 (
59     DTE1, GW1, DTE3, DTE2, A : agent,
60     TokenDTE13, TokenDTE12, TokenADTE1, AsesID, BsesID, Bloc : text,
61     KeyDTE1, KeyA, KeyGW1, KeyDTE3, KeyDTE2 : public_key,
62     SendDTE3, RecvDTE3, SendDTE2, RecvDTE2, SendGW1, RecvGW1 : channel(dy),
63     Hash : hash_func
64 )
65 played_by DTE1
66 def=
67     local
68     State : nat,
69     Alloc, OK : text
70     init
71     State := 0
72     transition
73     1. State = 0 /\ RecvDTE3({{TokenDTE13.{{TokenADTE1.AsesID'.Alloc'}_inv(KeyA)}_KeyDTE1}_inv(KeyDTE3)}_KeyDTE1) =|>
74     State' := 1 /\ SendDTE2({{TokenDTE12.AsesID'.Alloc'}_inv(KeyDTE1)}_KeyDTE2)
75     /\ SendGW1({{AsesID'.Alloc'}_inv(KeyDTE1)}_KeyGW1)
76     /\ SendDTE3({{Hash(TokenDTE13).AsesID'.Alloc'.BsesID'.Bloc'}_inv(KeyDTE1)}_KeyDTE3)
77     /\ witness(DTE1, DTE3, auth_dte1_dte3, TokenDTE13)
78     2. State = 1 /\ RecvDTE2({{Hash(TokenDTE12).OK'}_inv(KeyDTE2)}_KeyDTE1) =|>
79     State' := 2 /\ SendDTE3({{Hash(TokenDTE13).{{Hash(TokenADTE1).OK'}_inv(KeyDTE1)}_KeyA}_inv(KeyDTE1)}_KeyDTE3)
80     /\ witness(DTE1, A, auth_alice_dte1, TokenADTE1)
81     /\ secret(TokenDTE12, secret_dte1_dte2, {DTE1, DTE2})
82     /\ wrequest(DTE1, DTE2, auth_dte1_dte2, TokenDTE12)
83 end role
84
85 % -----
86
87 role dte2 (
88     DTE2, GW2, DTE1 : agent,
89     TokenDTE12 : text,
90     KeyDTE2, KeyGW2, KeyDTE1 : public_key,
91     SendGW2, RecvGW2, SendDTE1, RecvDTE1 : channel(dy),
92     Hash : hash_func
93 )
94 played_by DTE2
95 def=
96     local
97     State : nat,
98     AsesID, Alloc, OK : text
99     init
100    State := 0
101    transition
102    1. State = 0 /\ RecvDTE1({{TokenDTE12.AsesID'.Alloc'}_inv(KeyDTE1)}_KeyDTE2) =|>
103    State' := 1 /\ SendGW2({{AsesID'.Alloc'}_inv(KeyDTE2)}_KeyGW2)
104    /\ OK' := new() /\ SendDTE1({{Hash(TokenDTE12).OK'}_inv(KeyDTE2)}_KeyDTE1)
105    /\ witness(DTE2, DTE1, auth_dte1_dte2, TokenDTE12)
106 end role
107
108 % -----
109
110 role session (
111     A, DTE3, GW3, DTE1, GW1, DTE2, GW2 : agent,
112     TokenDTE13, TokenDTE12, TokenADTE1, AsesID, Alloc, BsesID, Bloc : text,

```

```

113     KeyA, KeyDTE3, KeyGW3, KeyDTE1, KeyGW1, KeyDTE2, KeyGW2 : public_key,
114     Hash : hash_func
115 )
116 def=
117     local
118         ChADTE3, ChDTE3A, ChDTE3DTE1, ChDTE1DTE3, ChDTE3GW3, ChGW3DTE3, ChDTE1DTE2, ChDTE2DTE1, ChDTE1GW1, ChGW1DTE1, \
119         ChDTE2GW2, ChGW2DTE2 : channel(dy)
120     composition
121         alice(A, DTE3, DTE1, TokenADTE1, AsesID, Aloc, KeyA, KeyDTE1, ChADTE3, ChDTE3A, Hash)
122         /\ dte3(DTE3, GW3, A, DTE1, TokenDTE13, KeyDTE3, KeyGW3, KeyA, KeyDTE1, ChDTE3A, ChADTE3, ChDTE3GW3, ChGW3DTE3, \
123            ChDTE3DTE1, ChDTE1DTE3, Hash)
124         /\ dte1(DTE1, GW1, DTE3, DTE2, A, TokenDTE13, TokenDTE12, TokenADTE1, AsesID, BsesID, Bloc, KeyDTE1, KeyA, KeyGW1, \
125            KeyDTE3, KeyDTE2, ChDTE1DTE3, ChDTE3DTE1, ChDTE1DTE2, ChDTE2DTE1, ChDTE1GW1, ChGW1DTE1, Hash)
126         /\ dte2(DTE2, GW2, DTE1, TokenDTE12, KeyDTE2, KeyGW2, KeyDTE1, ChDTE2GW2, ChGW2DTE2, ChDTE2DTE1, ChDTE1DTE2, Hash)
127     end role
128
129 % -----
130
131 role environment ()
132 def=
133     const
134         a, dte3, gw3, dte1, gw1, dte2, gw2 : agent,
135         tokendte31, tokendte12, tokenadte1, asesid, aloc, bsesid, bloc : text,
136         keya, keydte3, keygw3, keydte1, keygw1, keydte2, keygw2 : public_key,
137         secret_alice_dte1, auth_alice_dte1, secret_dte1_dte3, auth_dte1_dte3, dte1_dte2_secret, dte1_dte2_auth : protocol_id,
138         h : hash_func
139     intruder_knowledge = {a, keya, dte3, keydte3, gw3, keygw3, dte1, keydte1, gw1, keygw1, dte2, keydte2, gw2, keygw2}
140     composition
141         session(a, dte3, gw3, dte1, gw1, dte2, gw2, tokendte31, tokendte12, tokenadte1, asesid, \
142            aloc, bsesid, bloc, keya, keydte3, keygw3, keydte1, keygw1, keydte2, keygw2, h)
143         /\ session(a, dte3, gw3, dte1, gw1, dte2, gw2, tokendte31, tokendte12, tokenadte1, asesid, \
144            aloc, bsesid, bloc, keya, keydte3, keygw3, keydte1, keygw1, keydte2, keygw2, h)
145     end role
146
147 % -----
148
149 goal
150     secrecy_of secret_alice_dte1, secret_dte1_dte3, secret_dte1_dte2
151     authentication_on auth_alice_dte1
152     authentication_on auth_dte1_dte3
153     authentication_on auth_dte1_dte2
154 end goal
155
156 % -----
157
158 environment()
159
160 % -----

```

Second, we show the HLPSSL source code for starting a communication session, also differentiating the blocks for the *roles*, the *goal*, and the *environment* execution. There is one role for each entity involved in the protocol, as well as an additional role for the session, which has interactions in the security verification, as described in Chapter 4.

```

1 % -----
2
3 role alice (
4     A, DTE, B : agent,
5     AfacetID, BfacetID : text,
6     KeyA, KeyDTE1 : public_key,
7     SendDTE, RecvDTE, SendB, RecvB : channel(dy)
8 )
9 played_by A
10 def=
11     local
12         State : nat,
13         AsesID, BsesID : text
14     init
15         State := 0
16     transition

```

D. Protocols Represented in HPSL for AVISPA

```

17     1. State = 0 /\ RecvDTE(start) =|>
18     State' := 1 /\ AsesID' := new() /\ SendDTE(A.DTE.{AfacetID.AsesID'.BfacetID}_inv(KeyA))_KeyDTE1)
19     /\ witness(A, B, alice_session_id, AsesID')
20     2. State = 1 /\ RecvDTE(DTE.A.{BsesID'}_inv(KeyDTE1))_KeyA) =|>
21     State' := 2 /\ request(A, B, bob_session_id, BsesID') /\ SendB(A.B.BsesID')
22     3. State = 2 /\ RecvB(B.A.AsesID) =|>
23     State' := 3
24     /\ secret(AfacetID, alice_facet_id, {A, DTE, B})
25 end role
26
27 % -----
28
29 role dtel (
30     DTE1, A, DTE2 : agent,
31     AfacetID : text,
32     KeyDTE1, KeyA, KeyDTE2 : public_key,
33     SendA, RecvA, SendDTE2, RecvDTE2 : channel(dy)
34 )
35 played_by DTE1
36 def=
37     local
38         State : nat,
39         AsesID, BfacetID, BsesID : text
40     init
41         State := 0
42     transition
43     1. State = 0 /\ RecvA(A.DTE1.{AfacetID.AsesID'.BfacetID'}_inv(KeyA))_KeyDTE1) =|>
44     State' := 1 /\ SendDTE2(DTE1.DTE2.{AfacetID.AsesID'.BfacetID'}_inv(KeyDTE1))_KeyDTE2)
45     2. State = 1 /\ RecvDTE2(DTE2.DTE1.{BsesID'}_inv(KeyDTE2))_KeyDTE1) =|>
46     State' := 2 /\ SendA(DTE1.A.{BsesID'}_inv(KeyDTE1))_KeyA)
47 end role
48
49 % -----
50
51 role dte2 (
52     DTE2, B, DTE1 : agent,
53     BfacetID : text,
54     KeyDTE2, KeyB, KeyDTE1 : public_key,
55     SendB, RecvB, SendDTE1, RecvDTE1 : channel(dy)
56 )
57 played_by DTE2
58 def=
59     local
60         State : nat,
61         AfacetID, AsesID, BsesID : text
62     init
63         State := 0
64     transition
65     1. State = 0 /\ RecvDTE1(DTE1.DTE2.{AfacetID'.AsesID'.BfacetID}_inv(KeyDTE1))_KeyDTE2) =|>
66     State' := 1 /\ SendB(DTE2.B.{AfacetID'.AsesID'.BfacetID}_inv(KeyDTE2))_KeyB)
67     2. State = 1 /\ RecvB(B.DTE2.{BsesID'}_inv(KeyB))_KeyDTE2) =|>
68     State' := 2 /\ SendDTE1(DTE2.DTE1.{BsesID'}_inv(KeyDTE2))_KeyDTE1)
69 end role
70
71 % -----
72
73 role bob (
74     B, DTE, A : agent,
75     BfacetID, AfacetID : text,
76     KeyB, KeyDTE : public_key,
77     SendDTE, RecvDTE, SendA, RecvA : channel(dy)
78 )
79 played_by B
80 def=
81     local
82         State : nat,
83         BsesID, AsesID : text
84     init
85         State := 0
86     transition
87     1. State = 0 /\ RecvDTE(DTE.B.{AfacetID.AsesID'.BfacetID}_inv(KeyDTE))_KeyB) =|>
88     State' := 1 /\ BsesID' := new() /\ SendDTE(B.DTE.{BsesID'}_inv(KeyB))_KeyDTE)
89     /\ request(B, A, alice_session_id, AsesID') /\ witness(B, A, bob_session_id, BsesID')

```

```

90         2. State = 1 /\ RecvA(A.B.BsesID) =|>
91         State' := 2 /\ SendA(B.A.AsesID)
92         /\ secret(BfacetID, bob_facet_id, {B, DTE, A})
93     end role
94
95
96     % -----
97
98     role session (
99         A, DTE1, B, DTE2 : agent,
100        AfacetID, BfacetID : text,
101        KeyA, KeyDTE1, KeyB, KeyDTE2 : public_key
102    )
103    def=
104        local
105            ChADTE1, ChDTE1A, ChDTE1DTE2, ChDTE2DTE1, ChBDTE2, ChDTE2B, ChAB, ChBA : channel(dy)
106        composition
107            alice(A, DTE1, B, AfacetID, BfacetID, KeyA, KeyDTE1, ChADTE1, ChDTE1A, ChAB, ChBA)
108            /\ dtel(DTE1, A, DTE2, AfacetID, KeyDTE1, KeyA, KeyDTE2, ChDTE1A, ChADTE1, ChDTE1DTE2, ChDTE2DTE1)
109            /\ dte2(DTE2, B, DTE1, BfacetID, KeyDTE2, KeyB, KeyDTE1, ChDTE2B, ChBDTE2, ChDTE2DTE1, ChDTE1DTE2)
110            /\ bob(B, DTE2, A, BfacetID, AfacetID, KeyB, KeyDTE2, ChBDTE2, ChDTE2B, ChBA, ChAB)
111    end role
112
113    % -----
114
115    role environment ()
116    def=
117        const
118            a, dtel, b, dte2 : agent,
119            afacetid, bfacetid : text,
120            keya, keydte1, keyb, keydte2 : public_key,
121            bob_session_id : protocol_id,
122            alice_facet_id : protocol_id,
123            bob_facet_id : protocol_id
124        intruder_knowledge = {a, keya, dtel, keydte1, b, keyb, dte2, keydte2}
125        composition
126            session(a, dtel, b, dte2, afacetid, bfacetid, keya, keydte1, keyb, keydte2)
127    end role
128
129    % -----
130
131    goal
132        secrecy_of alice_facet_id, bob_facet_id
133        authentication_on alice_session_id
134        authentication_on bob_session_id
135    end goal
136
137    % -----
138
139    environment()
140
141    % -----

```

D. Protocols Represented in HPSL for AVISPA

Appendix E

Source Code for the Formal Verification of HIMALIS with Alloy

In this appendix we include the source code used for the formal verification of the HIMALIS architecture using the Alloy tool.

```
1  --
2  -- HIMALIS Model
3  --
4
5  open util/ordering[Time]
6
7  sig Time {}
8
9  sig Domain {}
10
11 sig Hostname { domain : Domain }
12
13 sig NodeID {}
14 sig HostID extends NodeID {}
15 sig GatewayID extends NodeID {}
16
17 abstract sig Locator {}
18 sig LLoc extends Locator {}
19 sig GLoc extends Locator {}
20
21 abstract sig Node {
22   t : Time,
23   id : NodeID,
24   domain : Domain
25 }
26
27 fact {
28   all n : Node | n.id != none
29 }
30
31 sig Host extends Node {
32   hostname : Hostname,
33   gateways : set Gateway,
34   gwid : set GatewayID,
35   lloc : set LLoc,
36   gloc : set GLoc
37 }
38
39 fact {
40   all h : Host | h.id in HostID && h.domain = h.hostname.domain
41   all disj h, h' : Host | h.hostname = h'.hostname => h.id = h'.id else h.id != h'.id
```

E. Source Code for the Formal Verification of HIMALIS with Alloy

```
42  all h : Host, gid : h.gwid | some gw : h.gateways | gid = gw.id
43  }
44
45  sig Gateway extends Node {
46    authenticated_hosts : set Host,
47    gw_gloc : set GLoc,
48    gw_lloc : set LLoc,
49    mapping_local : HostID -> LLoc,
50    mapping_global : HostID -> GLoc,
51    bindings : HostID -> HostID
52  }
53
54  fact {
55    all gw : Gateway | gw.id in GatewayID
56    all gw : Gateway, hid : HostID | hid -> hid not in gw.bindings
57  }
58
59  sig HNR extends Node {
60    authenticated_hosts : set Host,
61    mapping_hostid : Hostname -> HostID,
62    mapping_locator : Hostname -> GLoc
63  }
64
65  sig DNR extends Node {
66    mapping : Domain -> HNR
67  }
68
69  fact {
70    all dnr : DNR | dnr.domain = none
71  }
72
73  sig IDR extends Node {}
74
75  fact {
76    all n : Node - Gateway - Host | n.id not in GatewayID + HostID
77  }
78
79  pred can_reach_direct [n, n' : Node] {
80    n in Host => n' in n.gateways || n' in {n'' : Host | some gw : n''.gateways | gw in n.gateways}
81    n in Gateway => n' in Node - Host || n' in {n'' : Host | n in n''.gateways}
82    n in Node - Host - Gateway => n' in Node - Host
83  }
84
85  pred can_reach [n, n' : Node] {
86    can_reach_direct[n, n'] ||
87    some n'' : Node | can_reach[n, n''] && can_reach[n'', n']
88  }
89
90
91  --
92  -- Found/Standard Constraints
93  --
94
95  fact llocs_owned_by_gateways {
96    all loc : LLoc, disj gw, gw' : Gateway | loc in gw.gw_lloc && loc in gw'.gw_lloc => gw.id = gw'.id
97  }
98
99  fact glocs_owned_by_gateways {
100    all loc : GLoc, disj gw, gw' : Gateway | loc in gw.gw_gloc && loc in gw'.gw_gloc => gw.id = gw'.id
101  }
102
103  fact glocs_are_from_authenticated_gateways {
104    all gw : Gateway, loc : gw.gw_gloc, h : Host | loc in h.gloc => is_authenticated[h, gw, h.domain]
105  }
106
107  fact unique_node_id {
108    no disj n, n' : Node | n.t = n'.t && n.id = n'.id
109    no disj n, n' : Node | n.t = n'.t && n.domain = n'.domain
110  }
111
112  fact all_ids_in_nodes {
113    all id' : NodeID | some n : Node | n.id = id'
114  }
```

```

115
116 fact ids_and_domains {
117   all disj n, n' : Node | n.id = n'.id => n.domain = n'.domain
118 }
119
120 fact host_domain_symmetry {
121   all h : Host, gw : Gateway | h in gw.authenticated_hosts <=> gw in h.gateways
122 }
123
124
125 --
126 -- General Model Predicates
127 --
128
129 pred is_authenticated [h : Host] {
130   is_authenticated[h, h.domain]
131 }
132
133 pred is_authenticated [h : Host, dom : Domain] {
134   h.domain = dom
135   some hnr : HNR | h.domain = hnr.domain && h in hnr.authenticated_hosts
136 }
137
138 pred is_authenticated [h : Host, gw : Gateway, dom : Domain] {
139   h.domain = dom
140   h in gw.authenticated_hosts
141   some loc : h.lloc | loc in gw.gw_lloc && h.id -> loc in gw.mapping_local
142   some gl : h.gloc | gl in gw.gw_gloc && h.id -> gl in gw.mapping_global
143   some hnr : HNR | {
144     h.domain = hnr.domain
145     h in hnr.authenticated_hosts
146     h.hostname -> h.id in hnr.mapping_hostid
147     all gl' : h.gloc | h.hostname -> gl' in hnr.mapping_locator
148   }
149 }
150
151 pred has_updated_hnr_record [h : Host] {
152   is_authenticated[h] &&
153   let hnr = {hnr' : HNR | hnr'.domain = h.domain} | { --find_hnr_by_domain[h.domain] | {
154     h.hostname -> h.id in hnr.mapping_hostid
155     all loc : h.gloc | h.hostname -> loc in hnr.mapping_locator
156   }
157 }
158
159 pred can_resolve_hostname [h : Host, hname : Hostname] {
160   is_authenticated[h] &&
161   let tg_hnr = {hnr : HNR | hnr.domain = hname.domain} | --find_hnr_by_domain[hname.domain] |
162   let tg_h = {h' : Host | h'.hostname = hname} | { --find_host_by_hostname[hname] | {
163     can_reach[h, tg_hnr]
164     (some hid : HostID | hname -> hid in tg_hnr.mapping_hostid)
165     (some loc : GLoc | hname -> loc in tg_hnr.mapping_locator)
166     tg_h.hostname = hname
167     can_reach[tg_hnr, tg_h]
168   }
169 }
170
171 pred can_indicate_handover [h : Host, gw, gw' : Gateway] {
172   is_authenticated[h]
173   has_updated_hnr_record[h]
174   can_reach[h, gw]
175   is_authenticated[h, gw, h.domain]
176   can_reach[h, gw']
177   is_authenticated[h, gw', h.domain]
178 }
179
180 pred has_transferred_idloc_map [hid : HostID, old_gw, new_gw : Gateway] {
181   let chids = {hid' : HostID | hid -> hid' in old_gw.bindings} |
182   all chid : chids | {
183     hid -> chid in new_gw.bindings
184     let clocs = {loc : GLoc | chid -> loc in old_gw.mapping_global} |
185     all cloc : clocs | {
186       chid -> cloc in new_gw.mapping_global
187     }
188   }

```

E. Source Code for the Formal Verification of HIMALIS with Alloy

```
188 }
189 }
190
191 pred has_updated_idloc_map [hid : HostID, gw, cgw : Gateway] {
192   let glocs = {loc : GLoc | hid -> loc in gw.mapping_global} |
193   all loc : glocs | {
194     hid -> loc in cgw.mapping_global
195   }
196 }
197
198 pred has_completed_handover [h : Host, gw, gw' : Gateway] {
199   can_indicate_handover[h, gw, gw']
200   has_transferred_idloc_map[h.id, gw, gw']
201   let chids = {hid : HostID | h.id -> hid in gw.bindings} |
202   let cglocs = {loc : GLoc | some chid : chids | chid -> loc in gw.mapping_global} |
203   let cgws = {gw : Gateway | some loc : cglocs | loc in gw.gw_gloc} |
204   all cgw : cgws | {
205     has_updated_idloc_map[h.id, gw, cgw]
206     has_updated_idloc_map[h.id, gw', cgw]
207   }
208 }
209
210 pred has_deleted_idloc_map [h : Host, gw : Gateway] {
211   no hid : HostID | h.id -> hid in gw.bindings
212   no loc : GLoc | h.id -> loc in gw.mapping_global
213   no loc : LLoc | h.id -> loc in gw.mapping_local
214   h not in gw.authenticated_hosts
215 }
216
217 pred can_communicate [h, h' : Host] {
218   h != h' && h.id != h'.id
219   is_authenticated[h]
220   can_resolve_hostname[h, h'.hostname]
221   is_authenticated[h']
222   can_reach[h, h']
223 }
224
225
226 --
227 -- General Model Assertions and Checks
228 --
229
230 assert connectivity {
231   no disj h, h' : Host, disj gw, gw', ngw : Gateway | {
232     h.domain != h'.domain
233     h.domain = gw.domain
234     h'.domain = gw'.domain
235     h.domain != ngw.domain
236     h'.domain != ngw.domain
237     is_authenticated[h, gw, h.domain]
238     is_authenticated[h', gw', h'.domain]
239     !can_communicate[h, h']
240   }
241 }
242
243 check connectivity for 7
244
245 assert mobility {
246   no disj h, h' : Host, disj gw, gw', ngw : Gateway | {
247     h.domain != h'.domain
248     h.domain = gw.domain
249     h'.domain = gw'.domain
250     h.domain != ngw.domain
251     h'.domain != ngw.domain
252     is_authenticated[h, gw, h.domain]
253     is_authenticated[h', gw', h'.domain]
254     has_completed_handover[h, gw, ngw]
255     !can_communicate[h, h']
256   }
257 }
258
259 check mobility for 7
260
```

```

261 assert multihoming_ability {
262   all h : Host, disj gw, gw' : Gateway | {
263     (gw.domain != gw'.domain && gw.id != gw'.id && is_authenticated[h, gw, h.domain] &&
264      is_authenticated[h, gw', h.domain]) =>
265     (some disj gl, gl' : h.gloc | gl in gw.gw_gloc && gl' in gw'.gw_gloc)
266   }
267 }
268
269 check multihoming_ability for 7
270
271 assert idloc_deletion {
272   no h : Host, gw : Gateway | {
273     has_deleted_idloc_map[h, gw]
274     some loc : gw.gw_gloc | loc in h.gloc
275   }
276 }
277
278 check idloc_deletion for 7
279
280
281 --
282 -- General Model Instances
283 --
284
285 pred mobility_instance {
286   some disj h, h', nh : Host, disj gw, gw', ngw : Gateway | {
287     h.domain != h'.domain
288     h.domain = gw.domain
289     h'.domain = gw'.domain
290     h.domain != ngw.domain
291     h'.domain != ngw.domain
292     is_authenticated[h, gw, h.domain]
293     is_authenticated[h', gw', h'.domain]
294     can_communicate[h, h']
295     h.id = nh.id
296     has_completed_handover[nh, gw, ngw]
297     can_communicate[nh, h']
298   }
299 }
300
301 run mobility_instance for 10
302
303 pred multihoming_instance {
304   some h : Host, disj gw, gw' : Gateway | {
305     gw.domain != gw'.domain
306     gw.id != gw'.id
307     is_authenticated[h, gw, h.domain]
308     is_authenticated[h, gw', h.domain]
309   }
310 }
311
312 run multihoming_instance for 7
313
314 pred idloc_deletion_instance {
315   some disj h, h' : Host, disj gw, gw' : Gateway | {
316     h.id = h'.id && gw.id = gw'.id
317     is_authenticated[h, gw, h.domain]
318     has_deleted_idloc_map[h', gw']
319     !is_authenticated[h', gw', h'.domain]
320   }
321 }
322
323 run idloc_deletion_instance for 7
324
325
326 --
327 -- New Theorems
328 --
329
330 pred roaming_ability {
331   some h : Host, gw : Gateway | {
332     h.domain != gw.domain
333     is_authenticated[h, gw, h.domain]

```

E. Source Code for the Formal Verification of HIMALIS with Alloy

```
334     (all gw' : Gateway - gw | !is_authenticated[h, gw', h.domain])
335   }
336 }
337
338 run roaming_ability for 7
339
340 pred traceability {
341   some disj ti, ti' : Time, disj h, h' : Host, hid : HostID | {
342     h.t = ti && h'.t = ti'
343     hid = h.id && hid = h'.id
344   }
345 }
346
347 run traceability for 7
```

Appendix F

Autonomic Network Management

The advent of Software Defined Networking (SDN) has opened the door to new network functions that were difficult or even impossible to have. This has been the case of typically complex network management operations, which now can be layered on top of SDN controllers in order to adapt network behavior to achieve some objectives or quickly react to network events so network consistence is unaltered by them. However, users and services have little to say in current SDN architectures. In this appendix we discuss how to use an *Identity-Based Control Plane* to carry user and service identities and requirements to network controllers, which would use them to implement the necessary measures to meet their requirements. Such operations fall both into the control and management domains, so the proposed approach is a link between them. In this sense, the solution follows the model proposed by Autonomic Computing (AC), which has been set by the research community as the best paradigm for future network management, together with the model proposed by Service Oriented Architecture (SOA). To illustrate the operation of such solution we show how it can be used to interact with existing network services to create paths between users and services that accomplish with their requirements.

F.1 Introduction

In current networks, most network management operations are performed using certain mechanisms and protocols for monitoring and configuring network elements, from virtual to physical elements and from hosts to network equipment. Even though those mechanisms are normally used by specific applications that permits administrators to manage multiple elements from a central place, all tasks use to need human intervention. This behavior is also spread along the current Internet but, as the number of network elements grows, this

F. Autonomic Network Management

task is becoming more and more complicated to accomplish. Therefore, a new management paradigm is emerging from the Autonomic Computing (AC) initiative. It will overcome future requirements on self-management of systems and services, which are key challenges for the Future Internet (FI) [6, 153, 154].

Apart from resolving the issue with the rapid growth of systems to manage, the AC also address the added problem found in the also rapidly growing computer systems complexity, dynamism, and heterogeneity. Thus, AC systems are defined as “computing systems that can manage themselves given high-level objectives from administrators” [155]. This definition encompasses the key principle behind AC: Administrators (humans) set the rules (policies) by which systems should be guided and those systems are responsible of enforcing them. This way, AC presents a good solution to add self-management capabilities to modern networks and services, and so is supported by the Future Internet Assembly (FIA) [18] on its MANA position paper [7] on which autonomic network management plays a fundamental role, incorporating to the FI service model the main activities found in AC.

On the other hand, the Software Defined Networking (SDN) model is experiencing a huge growth, providing the necessary underlying mechanisms to implement control and management operations with little or no impact to underlying elements. Such model clearly separates control and data planes, which is an important feature, but also provides the necessary interfaces to build network services and applications on top of network controllers, which means the breaking of network ossification and the provisioning of huge flexibility to the network.

This has led us to define a mechanism that connects an *Identity-Based Control Plane*, as described in Chapter 4, to the SDN controller and the necessary connection points for them to properly address management operations and reflect their results into the network. Intermediate network elements will not have to be changed because the connection is performed through the SDN control plane. This way, network entities will be able to declare their network requirements and the network will be able to respond to such declaration by taking the appropriate determinations to meet those requirements as best as possible. All of this will be done without human intervention but some humans, particularly the network administrators, have to define the policies to which management and control operations will be enforced to accomplish.

The remainder of this chapter is organized as follows. First, we introduce the motivation of the present work and the challenges it exposes in Section F.2. Then, in Section F.3 we briefly analyze the most outstanding proposals for autonomic network management. In Section F.4 we describe the proposed solution, composed of network management modules

and the interconnection to the *Identity-Based Control Plane* and the SDN control plane. In Section F.5 we present an experimental instance of the proposed solution and discuss the experimentation results we have obtained with it. Finally, in Section F.6 we conclude the chapter.

F.2 Future Identity-Based Network Management

The search towards future networks has exposed many challenges [1], among which we highlight the separation of identity and location, and the person-to-person communications. As of today, there are many proposals that try to meet them with new and disruptive network architectures that deprecate the current OSI model. Our vision is that, in the future, many of these network architectures will coexist. It is desirable to combine and couple them in some or another way to provide the best service to future network users. On the other hand, the advances in technology are encouraging all of us to integrate our digital lives with our real lives. This means to us that user identity (and context) must be taken into account and be placed in a central point of the network architecture, which means to consider identities not only at application level but also at network level, allowing the establishment of zones of privacy (as in real life), as well as controlled identity linkability and information disclosure.

With the motivation described above, we are working in a new identity-based network architecture [104–106] that builds an *Identity-Based Control Plane* to facilitate the establishment of secure communications on top of heterogeneous infrastructures. The new plane is introduced to gather user intentions and their willingness to use multiple devices in a communication, which requires more mechanisms than just network mobility. As user identity is a delicate item, every decision is taken with a maximal constraint: privacy and data protection. It leads us to avoid the disclosure of the relation of data and identities, the identification via IP or MAC address, and the ability to keep the privacy across layers, cross-layer security. Thus, it is mandatory to use identities to address communication parties, instead of using identifiers or locators.

To achieve these goals, the architecture we propose is heavily based on the overlay network concept and the possibilities offered by SDN. On the one hand, an overlay network is used to address entities by their identities, so entities themselves are the communication endpoints and they can reach each other without dealing with network location and keeping their privacy and overall security. On the other hand, the mechanisms provided by SDN are used to instantiate final communications into the underlying networks, ensuing they

F. Autonomic Network Management

commit the necessary security and the requirements specified by communication parties. This way, the network is divided into different *identity domains*, managed by the Domain Trusted Entity Infrastructure (DTEi). This infrastructure has an instance on each identity domain that is responsible of the identities pertaining to such domain. However, any instance may be used to relay identity operations to the corresponding instance through the overlay network. The DTEi is also used to instantiate communications without disclosing any information from the entities they involve.

Since this new architecture is not bound to any specific underlying network and since they can be combined during communication, the control mechanisms provided by SDN are used to establish, configure, and release communication paths among communicating entities. Such paths are defined by their communication properties or parameters that, among others, are:

- source and destination endpoints, represented by the rules accepted by the SDN, including the addressing scheme of the specific underlying networks;
- service type, or the operation that is requested to the network: send a file, ask for a file, web browsing, voice call, etc.;
- traffic behavior (variable/constant bit rate, rate + strength, etc.);
- maximum delay and throughput; and
- levels of priority, security, and privacy.

At the end, each underlying network will use these parameters when reserving the necessary resources to create the requested low-level communication paths. Moreover, these paths will not be static but dynamic, so they support mobility and other environment changes. However, path management must not cause a significant increase in complexity or disrupt the normal network operation. Finally, due to dynamic interactions, the complexity of the operations, and the number of elements that can be involved, the operations must not need the constant supervision of network administrators, just under enforcement of the policies they set.

These requirements have led us to opt for AC principles [155] to design the management blocks of the solution. AC can provide the required features by being stimulated by network control events and with very little disruption to end or intermediate network entities. As described in Section F.3, current proposals for autonomic network management can not meet these requirements while being integrated with the *Identity-Based Control Plane* and

the SDN control. Therefore, as described in the following section, we have designed a simple management approach and included it into the integrated solution.

F.3 Current Autonomic Management Proposals

In this section we discuss some interesting proposals for autonomic network management that were the starting point to design our solution. An important feature is the service orientation because SOAs are better suited for the specific requirements of our solution. Thus, we briefly comment their strengths and weaknesses, with special attention on the capabilities we require but they lack.

The main features that a proper autonomic management solution should provide in order to fit with the integrated solution can be summarized as follows:

- Transparent support to build distributed systems where management involves many elements from different administrative domains (cross-domain management operations).
- The ability to easily integrate the architecture with existing systems, services, and applications, being or not service oriented and supporting different platforms.
- Exploit the control-loop concept of AC while offering fine granularity in management tasks.

To find these features we have analyzed some outstanding solutions, choosing them because, although they lack in some aspects, they are the best suited for our proposal.

First we have ANEMA [156]. It is an autonomic network management architecture that is driven by many types of policies and target goals. It follows AC principles and incorporates many network level elements and functions, but lacks the definition of specific mechanisms to involve multiple domains in the management process. Also, its mechanisms are not clearly provided as generic and reusable components, so it is difficult to customize it and make it interact with other external systems.

From the web service point of view, we have PAWS [157], that is a framework to build self-managed applications based on adaptive web services and following both AC and SOA principles. Its main purpose is to enhance business process execution language (BPEL) service composition adding self-configuration and self-healing capabilities. Although this architecture is generic and flexible enough to cover many management requirements, it

F. Autonomic Network Management

does not offer a complete AC control loop nor the necessary mechanisms to extend the self-management capabilities out of the service scope.

MAWeS [158] is a new architecture for building service oriented systems that follows SOA principles to provide self-tuning capabilities using automatically generated performance predictions. Although it provides many interesting capabilities, it lacks the specific definition of AC tasks and misses the AC control loop. Also, this architecture does not define how to make services from different domains collaborate to reach distributed objectives.

Finally we have ASMF [159], a framework that follows SOA and AC principles to provide dynamic service composition and enforcement of SLA contracts compliance. Although this architecture is very interesting, it is very tied to service level management and lacks certain interesting features such as the component generalization to permit the customization of the self-management operations, the definition of a policy decision point (PDP) and policy administration point (PAP), and the definition of elements to achieve cross-domain management.

F.4 The Proposed Solution

As introduced above, the *Identity-Based Control Plane* is governed by the DTEi and connected to communicating entities in order to allow them to operate securely. This way, the DTEi is privately aware of both identities and objectives of the entities that communicate so it is in the precise position to determine the requirements of communication instances and ask the network to build the necessary paths. However, this task is out of the scope of the DTEi itself so an external component is introduced to perform it. This component is the Autonomic Network Manager (ANM), whose specific design is discussed at the end of this section.

The ANM is responsible of monitoring network operations by receiving events from the DTEi, determine which actions should be taken on response to such events, and communicate such actions to the SDN controller so it can enforce them to the network. Typical network events would be to establish new communication sessions or the movement of an entity from one network to another. In general, it would be required to change the network parameters very frequently in response of the dynamism of requirements specified by communicating entities but also in response of changes of the network. This also implies that the SDN controller will report to ANM the events related to the network paths it manages.

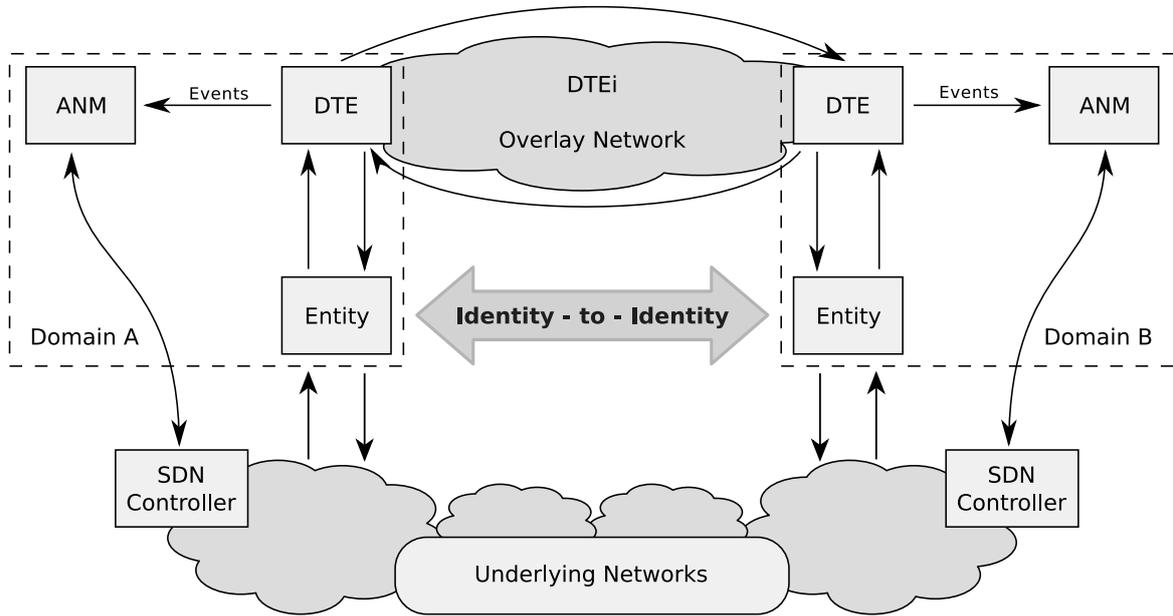


Figure F.1: Overview of the integrated architecture.

Instead of explicitly indicating the necessities to the ANM, it will be able to infer them by knowing what happens in the environment, like knowing when a network session is starting between two entities. To guide the whole management, the ANM will do an extensive use of policies, both to guess the reactions to certain events and to check if an operation is allowed or not. Those policies are set by network administrators, together with the necessary statements to match complex events from many simple events.

From now on we discuss how to integrate the ANM with the *Identity-Based Control Plane* and thus how this solution meets the requirements commented in the previous section. Figure F.1 illustrates how we propose to integrate them. It shows how network elements from different domains interact to achieve the global autonomic management objectives. The elements and their functions are described as follows:

- The DTEi, formed by the union of all its instances via an overlay network, is the main element of the *Identity-Based Control Plane*. The overlay network is used to decentralize its operation across all identity/network domains. Its main function, as described above, is to mediate in communication negotiations for the entities (communication parties) of each domain. Thus, the DTEi manages, for each communication, the session establishment, the security aspects, etc. Thus, the DTEi is also in place to send the necessary events for the ANM.

F. Autonomic Network Management

- Entity represents a communication party. As commented above, entities can be persons, software, machines, things, etc. Each entity relays its network operations to its corresponding DTE_i instance but the final data exchanges are performed through the data plane of the underlying network.
- ANM is the element that watches its environment by receiving events from DTE_i instances and the SDN controller. There is a different ANM deployed in each domain, connected to the DTE_i instance of such domain. It receives the events, analyzes the environment, checks the policies, and decides what to do in response. Then, it will contact the SDN controller to communicate such decisions so the underlying network meets the necessary requirements.
- SDN Controller represents the controller of the current domain of the underlying network. It will report to the ANM the changes in the network regarding the communications it is managing. Also, it will receive requirements from the ANM to be enforced into the network. Those requirements are mainly represented by communication parameters, such as bandwidth, latency, security level, etc.

That said, the ANM is only coupled with the *Identity-Based Control Plane* by means of the messages (events) sent by the DTE_i instance of its domain, so it respects the high decoupling design principle, which is widely recommended in network architecture design. Thus, this point is the main and only point of interaction between the two architectures. As it is totally asynchronous, the network operations are not delayed or disrupted by the management operations. Finally, the inter-domain nature of management and control architectures allows the integration and interaction of different domains. Below we discuss the internals of the ANM.

F.4.1 Autonomic Network Manager

As we will analyze in Section F.3, existing proposals for autonomic network management are centered in the interior part of the network so they do not consider entities into such task. Also, they do not use proper identification mechanisms and they are difficult to connect to current SDN control plane in a lightweight, non-intrusive manner. Therefore, in the integrated solution we have included our own approach to autonomic management.

The management solution is designed as a service-oriented architecture. Thus, it has a different service for each activity defined in Autonomic Computing (AC), together with the necessary components to integrate them and help them to be integrated with other

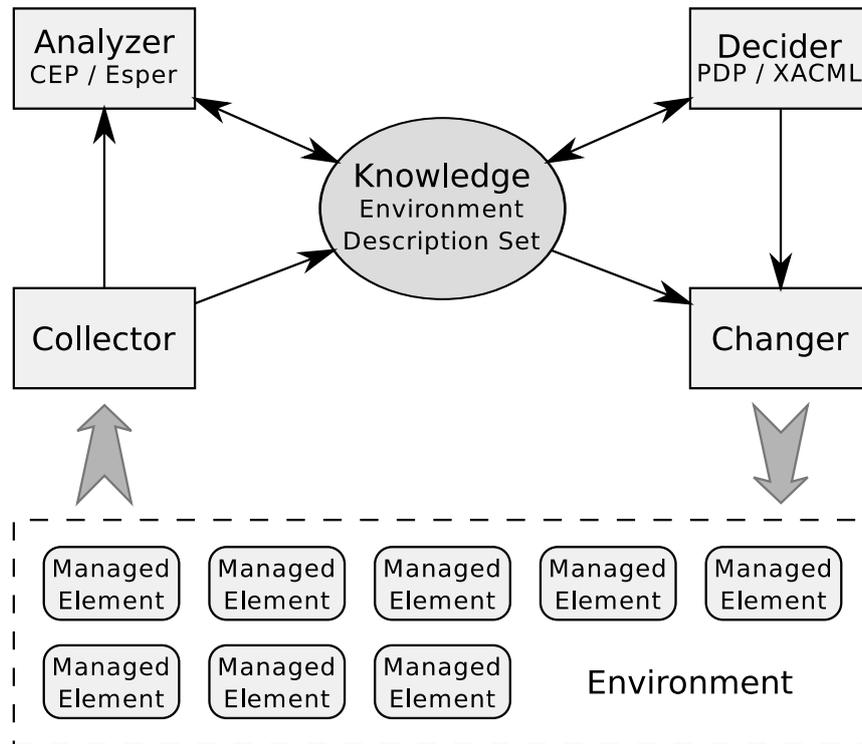


Figure F.2: Architecture of the management component, the ANM.

services or applications. These services are focused on genericity and flexibility. Instead of concentrated in the construction of certain solution, the services can be combined in many ways and with other services.

As depicted in Figure F.2, we have defined a different service for each AC task (collector, analyzer, decider, and changer). Also, we have defined a *knowledge* element that represents the knowledge that has the manager of its environment and that is built from the events received, the results of the analysis, the application of policies, etc.

Once we have defined the services we deploy them in top of GEMBus [160], a framework developed inside the GÉANT project to provide a new environment to enable users to create, integrate, and request service facilities on demand by means of the expansion of the current service model to produce the basic framework for a Multi-Domain ESB (MDESB). The inclusion of AC services into GEMBus framework may provide valuable capabilities to applications and services already deployed on GEMBus framework. Moreover, when the autonomic management solution is built in top of GEMBus it can interact with other

F. Autonomic Network Management

management solutions, such as AutoBAHN¹ and PerfSONAR² as we show in the following section.

Below we show a brief description of each necessary component to assembly the whole solution but before we want to illustrate how it works:

1. The collector receives messages that can be sent from different sources containing one or more registers to describe (part of) the current environment state. These messages are used to build the environment description set (knowledge) and sent to the analyzer. If there is no message received from the outside, a special element built with the Quartz Service Engine sends periodic messages to keep the knowledge alive.
2. The analyzer is built with a Complex Event Processor (CEP) based on Esper³ so it is capable to detect situations comprising several knowledge items. Thus, it analyzes the environment information and extracts new or updated items that are then sent back to the collector to complete the knowledge set.
3. For each received message, either from the outside, the *keepalive* service, or from the analyzer, the collector composes a new knowledge message and sends it to the decider.
4. With the knowledge it has received, the decider checks policies to know the environment correctness and with the result composes a new message to be sent to the changer. The policies are checked against a policy manager provided by an external service that implements a policy administration and decision point (PAP, PDP). It is based on XACML and here we decided to incorporate the implementation offered in XACML-Light⁴. This service is configured by administrators using its own interface to manage XACML policies.
5. The changer receives orders from the decider and communicates the actions (obligations) determined by the policies to the elements that should perform them.

This process shows that the key points of the architecture are the analyzer and the decider. They must be configured by administrators to determine the behavior of the manager but then it are designed to run by itself without other human intervention.

¹Bandwidth on Demand with AutoBAHN, <http://www.geant2.net/server/show/ConWebDoc.2544>

²perfSONAR: PERformance Service Oriented Network monitoring ARchitecture, <http://www.perfsonar.net>

³Esper - Complex Event Processing, <http://esper.codehaus.org>

⁴XACML Light, <http://xacmlight.sourceforge.net>

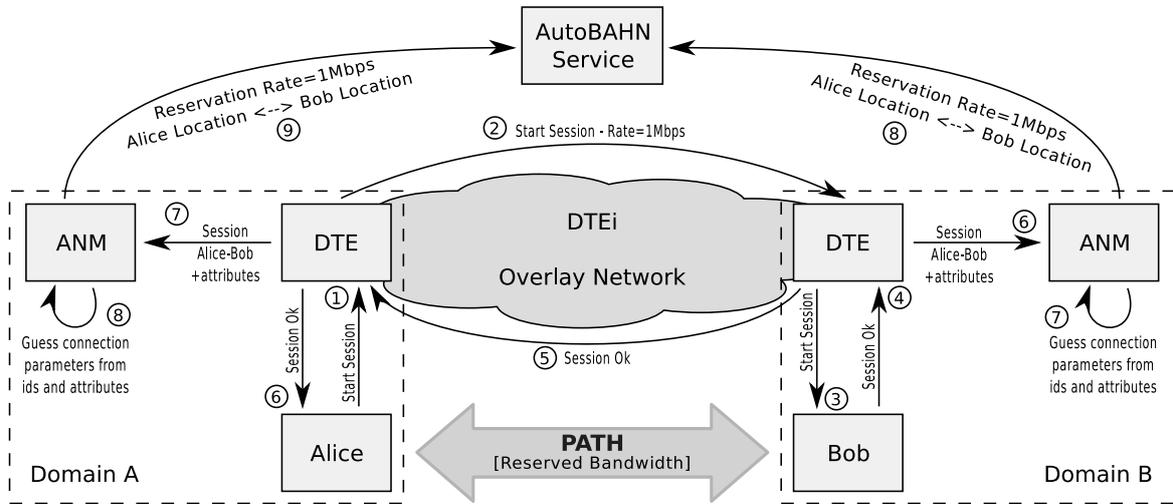


Figure F.3: Example scenario: Interaction with the AutoBAHN service.

Although it is not depicted in the architecture overview, the services are connected through an Enterprise Service Bus (ESB) which is used as service container and communication bus. It hosts the message router that is used to deliver messages among components. Here we use FUSE ESB because it is a standards based, free open-source software and is actively supported. Delving in the ESB concept, it incorporates JBI and OSGi. JBI is a specification for an approach to implementing a SOA. It is built on a message centered model, thus a key component in JBI is the Normalized Message Router (NMR) which, in turn, delivers messages among services. JBI defines two types of services, Binding Component (BC) and Service Engine (SE). The former is intended for connecting external services and adapt their protocol while the latter is intended to hold business models and other tools. The OSGi standard defines a complete component model and life-cycle management tool, also acting as component and service container, maintaining a registry of the services provided by installed components.

F.5 Evaluation and Results

To show the behavior of the integrated architecture we first define an example scenario that illustrates how the management solution knows the bandwidth requirement of a session and how it contacts with the network management service responsible of performing the bandwidth reservation. Then, we describe the experimental management solution we used to get an approximation of the performance and behavior of the proposed architecture.

F. Autonomic Network Management

Table F.1: Performance results (milliseconds).

Concurrent sessions	1	2	4	8	16	32
Average time	18.700	24.600	40.100	78.531	147.156	312.625
Total time, average	18.700	48.700	79.100	176.250	297.500	701.000
Total time, median	18.000	41.000	86.000	175.500	297.500	701.000
Normali. total time	18.700	24.350	19.775	22.031	18.594	21.906
Ses. estab. overhead	16.775	16.674	8.339	4.170	3.127	1.042

First of all, Figure F.3 shows the example scenario. On it, two entities initiate a communication and the ANM sets the communication path, calculating the parameters from the description of the session, which includes the attributes of the identities of both communication parties and the objective of the communication, and the policies that has been set by network administrators.

In the example, steps from 1 to 5 are necessary to establish the communication through the *Identity-Based Control Plane*. On these steps, Alice requests to start a session with Bob, specifying both identities and the aim of this session. Then, the DTEi instances talk to each other in order to negotiate the session. During this negotiation, Bob is actually asked to start the session and it accepts, so its DTEi instance accepts the negotiation and Alice's DTEi instance communicates it to Alice.

Once the communication has been accepted, the DTEi instances report to their ANMs that such session has been started, including the identities involved with their relevant attributes and the aim of the session. Then, the ANM checks the policies that apply to such identities and communication objective to determine the action to take. In the example, the action told and accepted by by the policies implicated in such operation is to contact the AutoBAHN service in order to reserve a bandwidth of 1 Mbps between Alice and Bob. The AutoBAHN service here plays the role of the SDN controller because bandwidth reservation is performed on top of the SDN controller, as a network service or application running on it.

This also demonstrates the great benefit of deploying the management architecture in top of GEMBus, because it wins easy access to many network services, like AutoBAHN. We should notice that since the *Identity-Based Control Plane* does not know the locators of the entities that start the communication until it has been accepted, the reservation request can not be fired up before receiving the *Session OK*. However, other scenarios may benefit from such action, so DTEi may be configured to trigger more events.

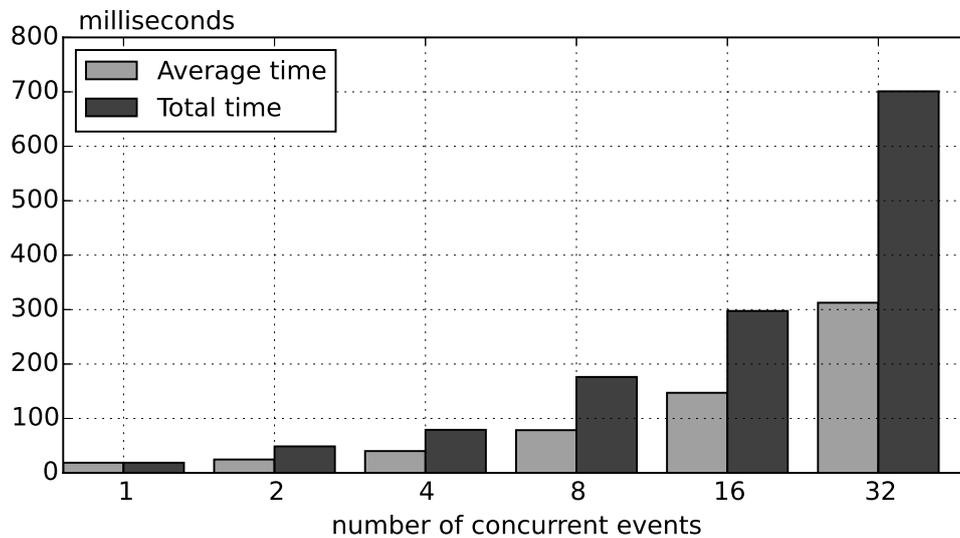


Figure F.4: Performance overview: Average and total processing times.

Once the scenario has been defined and the experimental implementation has been built, we have evaluated the solution by the generation of arbitrary events at different rates and the measurement of the time spent from the reception of a message (event) to the emission of a response (decision). We get the measures directly from the host where resides the management solution to avoid any latency that could be introduced by the network. The results are shown by Table F.1 and we discuss them below.

Figure F.4 shows the overall performance of the system, including the average and total times needed to process each event. The *average time* is calculated as the average of the time measured to process each individual event. The *total time* is the time spent to process all the events. The total time does not match the product of the average time by the number of concurrent events because different events are processed in parallel. With these results we analyze the behavior of the solution, comparing the average and total times spent to process each event while increasing the number of concurrent events. The average time is taken from the reception of the first message to the emission of the final decision, but the total time is the time spent in processing all concurrent events. Although the average time increases exponentially with the number of concurrent events, this does not mean poor scalability because the events are processed in series, thus there are many events processed at the same time but in different stage of the process. This parallelism is proved watching the total time spent in processing all messages. The total time does not match with the sum of the time spent in each individual event (or the multiplication of the average time spent processing an event by the total number of events being processed). On

F. Autonomic Network Management

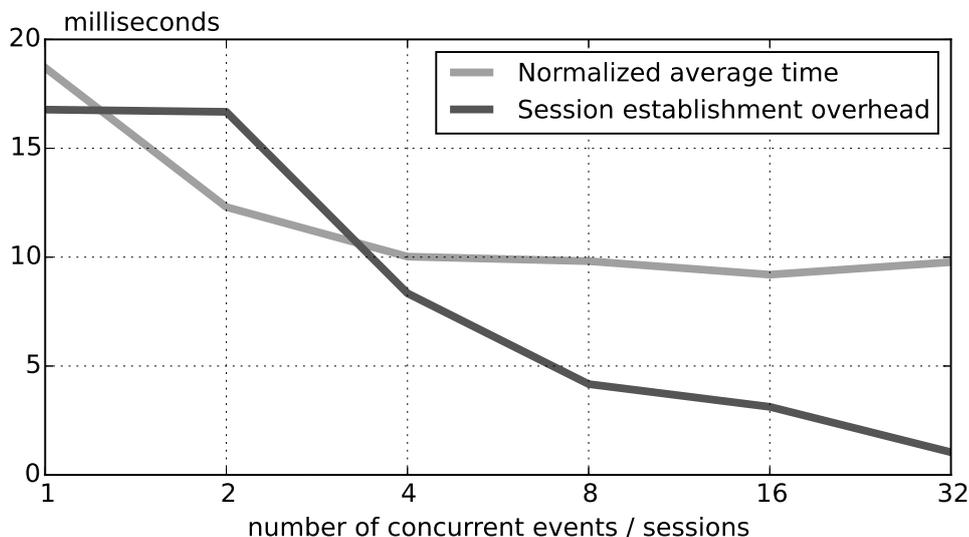


Figure F.5: Scalability overview.

the contrary, we can see that the total time is around the double of the average time, what demonstrate the high level of parallelism and concurrent behavior of the architecture.

Finally, Figure F.5 shows the scalability of the solution. The normalized average time is calculated by dividing the average time spent in event processing by the number of concurrent events. The session establishment overhead is calculated by dividing the difference between the total time and average time by the number of concurrent events. With these results we demonstrate that there is a correlation of the manager load, represented in number of threads, and the average time spent in event processing, from the reception of the first message to the emission of the final decision. We obtain this correlated time by dividing the average time spent in event processing by the number of concurrent events being processed at the same time. While concurrency level increases, the correlated time converges to 10 ms. This means that a manager processing 32 events at the same time is going to spend an average of 320 ms to each event, that is the product of 10 ms by 32 concurrent events. Thus, we can infer that if a manager average load reaches 64 concurrent events, each event takes an average of 640 ms to be processed, if it reaches 128 concurrent events, each event takes an average of 1280 ms, and so on. Moreover, the figure shows the evolution of the overhead induced to session establishment by the identity-based architecture per each concurrent session, which states that it spends less than 17 ms to establish single sessions and around 33 ms in all parallel sessions, regardless of the number of concurrent sessions, thus demonstrating the scalability of the architecture proposed here.

F.6 Conclusions

In this chapter we have discussed an approach to bring self-management features to the network by benefiting from a newly defined *Identity-Based Control Plane* and the enormous benefits provided by Autonomic Computing (AC), all applied to a Software Defined Networking (SDN) approach. Even though current architecture proposals follow AC principles in one level or another, they lack important features to build distributed systems in general, and cross-domain, federated systems in particular. Thus, we proposed to use our simple but complete solution to overcome the challenge but following the same AC principles as the other architectures. Moreover, we have defined how to perform the integration, the connection points between the management functional block and the SDN controller, and the connection points between the *Identity-Based Control Plane* and the management solution. In addition, we have demonstrated how the integrated architecture can be used and its good behavior by running an experimental implementation over an example scenario.

