Chapter 9

Appendix: Publications

Secure Ad hoc On-Demand Distance Vector Routing

Manel Guerrero Zapata
manel.guerrero-zapata@nokia.com
Mobile Networks Laboratory
Nokia Research Center
FIN-00045 NOKIA GROUP, Finland

This article gives an overview of different approaches to provide security features to routing protocols in mobile ad hoc networks (MANET). It also looks to Secure AODV (an extension to AODV that provides security features) giving a summary of its operation and talking about future enhancements to the protocol.

I. Introduction

Mobile ad hoc networks (MANET) protocols are being designed without having security in mind. In most of their specifications it is assumed that all the nodes in the network are friendly. The security issue has been postponed and there used to be the common feeling that it would be possible to make those routing protocols secure by retrofitting pre-existing cryptosystems.

Nevertheless, securing network transmissions without securing the routing protocols is not sufficient. Moreover, by retrofitting cryptosystems (like IPSec [KA98]) security is not necessarily achieved.

Therefore, in manet networks with security needs, there must be two security systems: one to protect the data transmission and one to make the routing protocol secure. There are already well studied point to point security systems that can be used for protecting network transmissions. But there is no much work about how make manet routing protocols discover routes in a secure manner [ZH99, JC99].

II. Symmetric vs. Asymmetric Cryptography

If in a MANET network all routing messages are encrypted with a symmetric cryptosystem, it means that everybody that we want to be able to participate in the network has to know the key. That is not a big problem if we are a "team" of persons that meet to let every member of the team to know the "team-key" and then we go to play on the ground creating a MANET network. A member of the team trust the other members of the team, so they assume that a member of the team will not do anything nasty to the other members. They trust and authorize the other members to change their routing tables.

Maybe this is the best thing to do for military scenarios (besides the problem of the compromised nodes and some others).

But now, let's thing that we want to create a MANET network where everybody can participate. Maybe in a convention, in a meeting room, in a campus, or in our neighborhood. Then we have a problem, we do not trust the others. We are not a team. So what do we do now? How do we force everybody to be honest? Maybe what we can do is to only believe a routing information if the originator of such information is the destination of the route (in such a way that if you lie (since you can only lie about yourself) the only benefit you get is that people is not able to communicate with you.

With this scenario in mind, the best option would be to use an asymmetric cryptosystem (with public an private key pairs) so that the originator of the route messages signs the message. It would not be needed to encrypt the routing messages because they are not secret. The only requirement is that the nodes will be able to detect forged routing messages.

III. Misbehaving Detection Schemes

Some work has been done to secure ad hoc networks by using misbehavior detection schemes (e.g., [MGLB00]). This kind of approach has two main problems:

- It is quite likely that it will be not feasible to detect several kind of misbehaving (specially because it is very hard to distinguish misbehaving from transmission failures and other kind of failures).
- It has no real means to guarantee the integrity and authentication of the routing messages.

Therefore, it is quite obvious that this approach is just not feasible. Any malicious node can generate forged misbehaving reports, making believe everybody that the rest of the nodes are even more evil that itself.

IV. Obscurity and Tamper Resistant Devices

Since there has not been, so far, a clear way to secure ad hoc networks, some people have decided to dust off the tamper resistant approaches. We will just refer to [AK96, AK97, BS97] where it is discussed why "trusting tamper resistance is problematic".

Obscurity is not the way to obtain security. There is not such a thing as a tampering resistant device. Therefore, trying to combine symmetric cryptography solutions with tamper resistant devices to create the same result provided by alternatives that use asymmetric cryptography does not make sense.

V. Secure AODV

The Secure Ad hoc On-Demand Distance Vector (SAODV) [Gue01] addresses the problem of securing a MANET network. SAODV is an extension of the AODV [PRD02] routing protocol that can be used to protect the route discovery mechanism providing security features like integrity, authentication and non-repudiation.

SAODV assumes that each ad hoc node has a signature key pair from a suitable asymmetric cryptosystem. Further, each ad hoc node is capable of securely verifying the association between the address of a given ad hoc node and the public key of that node. How this is achieved is the concern of the key management scheme.

Two mechanisms are used to secure the AODV messages: digital signatures to authentify the non-mutable fields of the messages, and hash chains to secure the hop count information (the only mutable information in the messages). This is because for the non-mutable information, authentication can be performed in a point-to-point manner, but the same kind of techniques cannot be applied to the mutable information.

Route error messages are protected in a different manner because they have a big amount of mutable information. In addition, it is not relevant which node started the route error and which nodes are just forwarding it. The only relevant information is that a neighbor node is informing to another node that it is not going to be able to route messages to certain destinations anymore.

Therefore, every node (generating or forwarding a route error message) uses digital signatures to sign the whole message and that any neighbor that receives verifies the signature.

VI. Future Work

Nowadays, I am working in a new version of SAODV. In the new version there will be some minor modifications to avoid certain possible attacks that could be performed against SAODV. In addition, some other modifications will address the need to reduce the processing power requirements of SAODV due to the use of asymmetric cryptography. This is going to be achieved by allowing nodes to forward routing messages before verifying it. In the case of a route discovery, the node will only need to verify the route request message after receiving and forwarding the corresponding route reply. This will avoid that all the nodes that will be not in the selected path will have to verify route request messages (with all the computation overhead that this requires).

Another thing I am planning to do is to add SAODV extension to the NRC-AODV (the Nokia Research Center AODV implementation for Linux created by me). NRC-AODV has most of the AODV features, and was tested in the first AODV interoperability test.

VII. Acknowledgments

N. Asokan (working in the Communication Systems Laboratory at Nokia Research Center) has contributed to SAODV with several improvements and corrections. Among other contributions, he came up with the way to authenticate the hop count in the routing messages by using hash chains.

Elizabeth M. Belding-Royer has kindly posted the SAODV draft under its AODV web page:
`http://www.cs.ucsb.edu/~eroyer/txt/saodv.txt`

References

[AK96] R. Anderson and M. Kuhn. Tamper resistance - a cautionary note. Proceedings of the Second Usenix Workshop on Electronic Commerce, pp. 1–11, November 1996., November 1996.

[AK97] R. Anderson and M. Kuhn. Low cost attacks on tamper resistant devices. In *IWSP: International Workshop on Security Protocols, LNCS*, 1997.

[BS97] Eli Biham and Adi Shamir. Differential fault analysis of secret key cryptosystems. In *CRYPTO*, pages 513–525, 1997.

[Gue01] Manel Guerrero. Secure ad hoc on-demand distance vector (SAODV) routing, August 2001. INTERNET-DRAFT draft-guerrero-manet-saodv-00.txt.

[JC99] S. Jacobs and M. S. Corson. Manet authentication architecture, February 1999. INTERNET-DRAFT draft-jacobs-imep-auth-arch-01.txt.

[KA98] S. Kent and R. Atkinson. Security architecture for the internet protocol. IETF Request for Comments, November 1998. `RFC 2401`.

[MGLB00] Sergio Marti, T. J. Giuli, Kevin Lai, and Mary Baker. Mitigating routing misbehavior in mobile ad hoc networks. In *Mobile Computing and Networking*, pages 255–265, 2000.

[PRD02] Charles E. Perkins, Elizabeth M. Royer, and Samir R. Das. Ad hoc on-demand distance vector (AODV) routing. IETF INTERNET DRAFT, MANET working group, January 2002. `draft-ietf-manet-aodv-10.txt`.

[ZH99] Lidong Zhou and Zygmunt J. Haas. Securing ad hoc networks. *IEEE Network Magazine*, 13(6), November/December 1999.

Securing Ad hoc Routing Protocols

Manel Guerrero Zapata
Mobile Networks Laboratory
Nokia Research Center
FIN-00045 NOKIA GROUP, Finland
manel.guerrero-zapata@nokia.com

N. Asokan
Communication Systems Laboratory
Nokia Research Center
FIN-00045 NOKIA GROUP, Finland
n.asokan@nokia.com

ABSTRACT

We consider the problem of incorporating security mechanisms into routing protocols for ad hoc networks. Canned security solutions like IPSec are not applicable. We look at AODV [20] in detail and develop a security mechanism to protect its routing information. We also briefly discuss whether our techniques would also be applicable to other similar routing protocols and about how a key management scheme could be used in conjunction with the solution that we provide.

Categories and Subject Descriptors

C.2.2 [**Computer-Communication Networks**]: Network Protocols—*Routing protocols*

General Terms

Security, Algorithms

Keywords

Security, Ad hoc Wireless Networks, Routing Protocols, Hash Chains, SAODV, Secure AODV

1. INTRODUCTION

An *ad hoc network* is often defined as an "infrastructureless" network, meaning a network without the usual routing infrastructure like fixed routers and routing backbones. Typically, the ad hoc nodes are mobile and the underlying communication medium is wireless. Each ad hoc node may be capable of acting as a router. Such ad hoc networks may arise in personal area networking, meeting rooms and conferences, disaster relief and rescue operations, battlefield operations, etc.

Some aspects of ad hoc networks have interesting security problems [1, 30, 27]. Routing is one such aspect. Several routing protocols for ad hoc networks have been developed, particularly in the *MANET* working group of the Internet Engineering Task Force (IETF). Surveys of routing protocols for ad hoc wireless networks are presented in [24] and [25].

In this paper, we consider the security of routing protocols for ad hoc networks. Section 2 takes a look at related work. Section 3 analyzes the security requirements in ad hoc networks. Section 4 discusses how ad hoc protocols should be secured. Section 5 focuses on AODV, its security flaws and describes our proposed security mechanism to protect AODV's routing information: Secure AODV (SAODV) [6]. Section 6 studies how the approach shown in this paper could be ported to other ad hoc routing protocols. Section 7 considers different ways to achieve the required key management for SAODV. Section 8 summarizes and shows what is the current status of this work.

2. RELATED WORK

There is very little published prior work on the security issues in ad hoc network routing protocols. Neither the survey by Ramanathan and Steenstrup [24] nor the survey by Royer and Toh [25] mention security. None of the draft proposals in the IETF *MANET* working group have a non-trivial "security considerations" section. Actually, most of them assume that all the nodes in the network are friendly, and a few declare the problem out-of-scope by assuming some canned solution like IPSec may be applicable.

There are some works on securing routing protocols for fixed networks that also deserved to be mentioned here. Perlman, in her thesis [21], proposed a link state routing protocol that achieves Byzantine Robustness. Although her protocol is highly robust, it requires a very high overhead associated with public key encryption. Secure BGP [12] attempts to secure the Border Gateway Protocol by using PKI (Public Key Infrastructure) and IPsec.

In their paper on securing ad hoc networks [30], Zhou and Haas primarily discuss key management (we discuss key management in Section 7). They devote a section to secure routing, but essentially conclude that "nodes can protect routing information in the same way they protect data traffic". They also observe that denial-of-service attacks against routing will be treated as damage and routed around.

Security issues with routing in general have been addressed by several researchers (e.g., [26, 8]). And, lately, some work has been done to secure ad hoc networks by using misbehavior detection schemes (e.g., [16]). This approach has two main problems: first, it is quite likely that it will be not feasible to detect several kinds of misbehaving (especially because it is very hard to distinguish misbehaving from transmission failures and other kind of failures); and second, it

WiSe'02, September 28, 2002, Atlanta, Georgia, USA.

has no real means to guarantee the integrity and authentication of the routing messages.

Dahill et al. [5] proposed ARAN, a routing protocol for ad hoc networks that uses authentication and requires the use of a trusted certificate server. In ARAN, every node that forwards a route discovery or a route reply message must also sign it, (which is very computing power consuming and causes the size of the routing messages to increase at each hop), whereas the proposal presented in this paper only require originators to sign the message. In addition, it is prone to reply attacks using error messages unless the nodes have time synchronization.

Papadimitratos and Haas [19] proposed a protocol (SRP) that can be applied to several existing routing protocols (in particular DSR [11] and IERP [7]). SRP requires that, for every route discovery, source and destination must have a security association between them. Furthermore, the paper does not even mention route error messages. Therefore, they are not protected, and any malicious node can just forge error messages with other nodes as source.

Hash chains have being used as an efficient way to obtain authentication in several approaches that tried to secure routing protocols. In [8], [4] and [23] they use them in order to provide delayed key disclosure. While, in [29], hash chains are used to create one-time signatures that can be verified immediately. The main drawback of all the above approaches is that all of them require clock synchronization.

In SEAD [9] (by Hu, Johnson and Perrig) hash chains are also used in combination with DSDV-SQ [2] (this time to authenticate hop counts and sequence numbers). At every given time each node has its own has chain. The hash chain is divided into segments, elements in a segment are used to secure hop counts in a similar way as we do in SAODV. The size of the hash chain is determined when it is generated. After using all the elements of the hash chain a new one must be computed.

SEAD can be used with any suitable authentication and key distribution scheme. But finding such a scheme is not straightforward. In Section 7 we suggest some non-standard approaches that can be used to achieve key distribution.

Ariadne [10], by the same authors, is based on DSR [11] and TESLA [22] (on which it is based its authentication mechanism). It also requires clock synchronization, which we consider to be an unrealistic requirement for ad hoc networks.

It is quite likely that, for a small team of nodes that trust each other and that want to create an ad hoc network where the messages are only routed by members of the team, the simplest way to keep secret their communications is to encrypt all messages (routing and data) with a "team key". Every member of the team would know the key and, therefore, it would be able to encrypt and decrypt every single packet. Nevertheless, this does not scale well and the members of the team have to trust each other. So it can be only used for a very small subset of the possible scenarios.

Looking at the work that had been done in this area previously, we felt that the security needs for ad hoc networks had not been yet satisfied (at least for those scenarios where everybody can freely participate in the network). In the next section, we specify what are those needs in the format of a list of security requirements.

3. SECURITY REQUIREMENTS

In most domains, the primary security service is **authorization**. Routing is no exception. Typically, a router needs to make two types of authorization decisions. First, when a routing update is received from the outside, the router needs to decide whether to modify its local routing information base accordingly. This is *import authorization*. Second, a router may carry out *export authorization* whenever it receives a request for routing information. Import authorization is the critical service.

In traditional routing systems, authorization is a matter of policy. For example, *gated*, a commonly used routing program[1], allows the administrator of a router to set policies about whether and how much to trust routing updates from other routers: e.g., statements like "trust router X about routes to networks A and B". In mobile ad hoc networks, such static policies are not sufficient (and unlikely to be relevant anyway).

Authorization may require other security services such as **authentication** and **integrity**. Techniques like digital signatures and message authentication codes are used to provide these services.

In the context of routing, confidentiality and non-repudiation are not necessarily critical services [8]. Zhou and Haas [30] argue that non-repudiation is useful in an ad hoc network for isolating misbehaving routers: a router A which received an "erroneous message" from another router B may use this message to convince other routers that B is misbehaving. This would indeed be useful if there is a reliable way of detecting erroneous messages. This does not appear to be an easy task.

We do not address the problem of compromised nodes since we believe that it is not critical in non military scenarios. Availability is also outside of the scope of this paper. Although of course it would be desirable, it does not seem to be feasible to prevent denial-of-service attacks in a network that uses wireless technology (where an attacker can focus on the physical layer without bothering to study the routing protocol).

Therefore, in this paper we consider the following requirements:

- **Import authorization**: It is important to note that in here we are not referring to the traditional meaning of authorization. What we mean is that the ultimate authority about routing messages regarding a certain destination node is that node itself. Therefore, we will only authorize route information in our routing table if that route information concerns the node that is sending the information. In this way, if a malicious node lies about it, the only thing it will cause is that others will not be able to route packets to the malicious node.
- **Source authentication**: We need to be able to verify that the node is the one it claims to be.
- **Integrity**: In addition, we need to be able to verify that the routing information that it is being sent to us has arrived unaltered.
- The two last security services combined build **data authentication**, and they are requirements derived from our import authorization requirement.

[1] `http://www.gated.org`

4. SECURING AD HOC PROTOCOLS

In an ad hoc network, from the point of view of a routing protocol, there are two kinds of messages: the routing messages and the data messages. Both have a different nature and different security needs. Data messages are point-to-point and can be protected with any point-to-point security system (like IPSec). On the other hand, routing messages are sent to immediate neighbors, processed, possibly modified, and resent. Moreover, as a result of the processing of the routing message, a node might modify its routing table. This creates the need for the intermediate nodes to be able to authenticate the information contained in the routing messages (a need that does not exist in point-to-point communications) to be able to apply their import authorization policy.

Another consequence of the nature of the transmission of routing messages is that, in many cases, there will be some parts of those messages that will change during their propagation. This is very common in Distance-Vector routing protocols, where the routing messages usually contain a hop count of the route they are requesting or providing. Therefore, in a routing message we could distinguish between two types of information: mutable an non-mutable. It is desired that the mutable information in a routing message is secured in such a way that no trust in intermediate nodes is needed. Otherwise, securing the mutable information will be much more expensive in computation, plus the overall security of the system will greatly decrease.

If the security system we are using to secure the network transmissions in a MANET network is IPSec, it is necessary that the IPSec implementation can use as a selector the TCP and UDP port numbers. This is because it is necessary that the IPSec policy will be able to apply certain security mechanisms to the data packets and just bypass the routing packets (that typically can be identified because they use a reserved transport layer port number).

5. AODV

5.1 Overview

Ad Hoc On-Demand Vector Routing (AODV) protocol [20] is a reactive routing protocol for ad hoc and mobile networks that maintains routes only between nodes which need to communicate. The routing messages do not contain information about the whole route path, but only about the source and the destination. Therefore, routing messages do not have an increasing size. It uses destination sequence numbers to specify how fresh a route is (in relation to another), which is used to grant loop freedom.

Whenever a node needs to send a packet to a destination for which it has no 'fresh enough' route (i.e., a valid route entry for the destination whose associated sequence number is at least as great as the ones contained in any RREQ that the node has received for that destination) it broadcasts a route request (RREQ) message to its neighbors. Each node that receives the broadcast sets up a reverse route towards the originator of the RREQ (unless it has a 'fresher' one). When the intended destination (or an intermediate node that has a 'fresh enough' route to the destination) receives the RREQ, it replies by sending a Route Reply (RREP). It is important to note that the only mutable information in a RREQ and in a RREP is the hop count (which is being monotonically increased at each hop). The RREP travels back to the originator of the RREQ (this time as a unicast). At each intermediate node, a route to the destination is set (again, unless the node has a 'fresher' route than the one specified in the RREP). In the case that the RREQ is replied to by an intermediate node (and if the RREQ had set this option), the intermediate node also sends a RREP to the destination. In this way, it can be granted that the route path is being set up bidirectionally. In the case that a node receives a new route (by a RREQ or by a RREP) and the node already has a route 'as fresh' as the received one, the shortest one will be updated.

If there is a subnet (a collection of nodes that are identified by a common network prefix) that does not use AODV as its routing protocol and wants to be able to exchange information with an AODV network, one of the nodes of the subnet can be selected as their 'network leader'. The network leader is the only node of the subnet that sends, forwards and processes AODV routing messages. In every RREP that the leader issues, it sets the prefix size of the subnet.

Optionally, a Route Reply Acknowledgment (RREP-ACK) message may be sent by the originator of the RREQ to acknowledge the receipt of the RREP. RREP-ACK message has no mutable information.

In addition to these routing messages, Route Error (RERR) message are used to notify the other nodes that certain nodes are not anymore reachable due to a link breakage. When a node rebroadcasts a RERR, it only adds the unreachable destinations to which the node might forward messages. Therefore, the mutable information in a RERR are the list of unreachable destinations and the counter of unreachable destinations included in the message. Anyway, it is predictable that, at each hop, the unreachable destination list may not change or become a subset of the original one.

5.2 Security flaws of AODV

Since AODV has no security mechanisms, malicious nodes can perform many attacks just by not behaving according to the AODV rules. A malicious node M can carry out the following attacks (among many others) against AODV:

1. Impersonate a node S by forging a RREQ with its address as the originator address.
2. When forwarding a RREQ generated by S to discover a route to D, reduce the hop count field to increase the chances of being in the route path between S and D so it can analyze the communication between them. A variant of this is to increment the destination sequence number to make the other nodes believe that this is a 'fresher' route.
3. Impersonate a node D by forging a RREP with its address as a destination address.
4. Impersonate a node by forging a RREP that claims that the node is the destination and, to increase the impact of the attack, claims to be a network leader of the subnet SN with a big sequence number and send it to its neighbors. In this way it will became (at least locally) a blackhole for the whole subnet SN.
5. Selectively, not forward certain RREQs and RREPs, not reply to certain RREPs and not forward certain data messages. This kind of attack is especially hard

Value	Hash function
0	Reserved
1	MD5HMAC96 [14]
2	SHA1HMAC96 [15]
3-127	Reserved
128-255	Implementation dependent

Table 1: Possible values of the Hash Function field

to even detect because transmission errors have the same effect.

6. Forge a RERR message pretending it is the node S and send it to its neighbor D. The RERR message has a very high destination sequence number dsn for one of the unreachable destinations (U). This might cause D to update the destination sequence number corresponding to U with the value dsn and, therefore, future route discoveries performed by D to obtain a route to U will fail (because U's destination sequence number will be much smaller than the one stored in D's routing table).

7. According to the current AODV draft [20], the originator of a RREQ can put a much bigger destination sequence number than the real one. In addition, sequence numbers wraparound when they reach the maximum value allowed by the field size. This allows a very easy attack in where an attacker is able to set the sequence number of a node to any desired value by just sending two RREQ messages to the node.

5.3 Securing AODV

We assume that there is a key management sub-system that makes it possible for each ad hoc node to obtain public keys from the other nodes of the network. Further, each ad hoc node is capable of securely verifying the association between the identity of a given ad hoc node and the public key of that node. How this is achieved depends on the key management scheme. We discuss key management in Section 7.

Two mechanisms are used to secure the AODV messages: digital signatures to authenticate the non-mutable fields of the messages, and hash chains to secure the hop count information (the only mutable information in the messages). For the non-mutable information, authentication is perform in an end-to-end manner, but the same kind of techniques cannot be applied to the mutable information. The figures in Appendix A show the structure of the AODV messages and indicate what are the mutable fields of the messages.

The information relative to the hash chains and the signatures is transmitted with the AODV message as an extension message that we will refer to as Signature Extension. The format of the SAODV Signature Extensions is shown in Appendix B.

5.3.1 SAODV hash chains

SAODV uses hash chains to authenticate the hop count of RREQ and RREP messages in such a way that allows every node that receives the message (either an intermediate node or the final destination) to verify that the hop count has not been decremented by an attacker. This prevents an attack of type 2. A hash chain is formed by applying a one-way hash function repeatedly to a seed.

Every time a node originates a RREQ or a RREP message, it performs the following operations:

- Generates a random number ($seed$).
- Sets the Max_Hop_Count field to the TimeToLive value (from the IP header).

$$Max\_Hop\_Count = TimeToLive$$

- Sets the Hash field to the $seed$ value.

$$Hash = seed$$

- Sets the Hash_Function field to the identifier of the hash function that it is going to use. The possible values are shown in Table 1.

$$Hash\_Function = h$$

- Calculates Top_Hash by hashing $seed$ Max_Hop_Count times.

$$Top\_Hash = h^{Max\_Hop\_Count}(seed)$$

Where:

  - h is a hash function.
  - $h^i(x)$ is the result of applying the function h to x i times.

In addition, every time a node receives a RREQ or a RREP message, it performs the following operations in order to verify the hop count:

- Applies the hash function h Maximum_Hop_Count minus Hop_Count times to the value in the $Hash$ field, and verifies that the resultant value is equal to the value contained in the Top_Hash field.

$$Top\_Hash == h^{Max\_Hop\_Count - Hop\_Count}(Hash)$$

Where:

  - $a == b$ reads: to verify that a and b are equal.

- Before rebroadcasting a RREQ or forwarding a RREP, a node applies the hash function to the Hash value in the Signature Extension to account for the new hop.

$$Hash = h(Hash)$$

The Hash_Function field indicates which hash function has to be used to compute the hash. Trying to use a different hash function will just create a wrong hash without giving any advantage to a malicious node. Hash_Function, Max_Hop_Count, Top_Hash, and Hash fields are transmitted with the AODV message, in the Signature Extension. And, as it will be explained in the next subsection, all of them but the Hash field are signed to protect its integrity.

5.3.2 SAODV digital signatures

Digital signatures are used to protect the integrity of the non-mutable data in RREQ and RREP messages. That means that they sign everything but the Hop_Count of the AODV message and the Hash from the SAODV extension.

The main problem in applying digital signatures is that AODV allows intermediate nodes to reply RREQ messages if they have a 'fresh enough' route to the destination. While

this makes the protocol more efficient it also makes it more complicated to secure. The problem is that a RREP message generated by an intermediate node should be able to sign it on behalf of the final destination. And, in addition, it is possible that the route stored in the intermediate node would be created as a reverse route after receiving a RREQ message (which means that it does not have the signature for the RREP).

To solve this problem, this paper offers two alternatives. The first one (and also the obvious one) is that, if an intermediate node cannot reply to a RREQ message because it cannot properly sign its RREP message, it just behaves as if it didn't have the route and forwards the RREQ message. The second is that, every time a node generates a RREQ message, it also includes the RREP flags, the prefix size and the signature that can be used (by any intermediate node that creates a reverse route to the originator of the RREQ) to reply a RREQ that asks for the node that originated the first RREQ. Moreover, when an intermediate node generates a RREP message, the lifetime of the route has changed from the original one. Therefore, the intermediate node should include both lifetimes (the old one is needed to verify the signature of the route destination) and sign the new lifetime. In this way, the original information of the route is signed by the final destination and the lifetime is signed by the intermediate node.

To distinguish the different SAODV extension messages, the ones that have two signatures are called RREQ and RREP Double Signature Extension.

When a node receives a RREQ, it first verifies the signature before creating or updating a reverse route to that host. Only if the signature is verified, will it store the route. If the RREQ was received with a Double Signature Extension, then the node will also store the signature for the RREP and the lifetime (which is the 'reverse route lifetime' value) in the route entry. An intermediate node will reply to a RREQ with a RREP only if it fulfills the AODV's requirements to do so and the node has the corresponding signature and old lifetime to put into the Signature and Old Lifetime fields of the RREP Double Signature Extension. Otherwise, it will rebroadcast the RREQ.

When a RREQ is received by the destination itself, it will reply with a RREP only if it fulfills the AODV's requirements to do so. This RREP will be sent with a RREP Single Signature Extension.

When a node receives a RREP, it first verifies the signature before creating or updating a route to that host. Only if the signature is verified, will it store the route with the signature of the RREP and the lifetime.

Using digital signatures prevents attack scenarios 1 and 3.

5.3.3 SAODV error messages

Concerning RERR messages, someone could think that the right approach to secure them should be similar to the way the other AODV messages are (signing the non-mutable information and finding out a way to secure the mutable information). Nevertheless, RERR messages have a big amount of mutable information. In addition, it is not relevant which node started the RERR and which nodes are just forwarding it. The only relevant information is that a neighbor node is informing another node that it is not going to be able to route messages to certain destinations anymore.

Our proposal is that every node (generating or forwarding a RERR message) will use digital signatures to sign the whole message and that any neighbor that receives it will verify the signature. In this way it can verify that the sender of the RERR message is really the one that it claims to be. And, since destination sequence numbers are not signed by the corresponding node, a node should never update any destination sequence number of its routing table based on a RERR message (this prevents a malicious node from performing attack type 6). Implementing a mechanism that will allow the destination sequence numbers of a RERR message to be signed by their corresponding nodes would add too much overhead compared with the advantage of the use of that information.

Although nodes will not trust destination sequence numbers in a RERR message, they will use them to decide whether they should invalidate a route or not. This does not give any extra advantage to a malicious node.

5.3.4 When a node reboots

The attack type 7 was based on the fact that the originator of the RREQ can set the sequence number of the destination. This should have not been specified in AODV because it is not needed. In the case everybody behaves according to the protocol the situation in which the originator of a RREQ will put a destination sequence number bigger than the real one will never happen. Not even in the case that the destination of the RREQ has rebooted. After rebooting, the node does not remember its sequence number anymore, but it waits for a period long enough before being active, so that when it wakes up nobody has stored its old sequence number anymore.

To avoid this attack, in the case that the destination sequence number in the RREQ is bigger than the destination sequence number of the destination node, the destination node will not take into account the value in the RREQ. Instead, it will realize that the originator of the RREQ is misbehaving and will send the RREP with the right sequence number.

In addition, if one of the nodes has a way to store its sequence number every time it modifies it, it might do so. Therefore, when it reboots it will not need to wait long enough so that everybody deletes routes towards it.

5.3.5 Analysis

The digital signature $Digital\_signature_X(routing\_message)$ can be created only by X. Thus, it serves as proof of validity of the information contained in the routing message. This prevents attack scenarios 1, 3, 4, and 6.

The hop authenticator reduces the ability of a malicious intermediate hop for mounting the attack type 2 by arbitrarily modifying the hop count without detection. A node that is n hops away from T will know the n^{th} element in the hash chain $(h^n(x))$, but it will not know any element that comes before this because of the one-way property of $h()$. However, the malicious node could still pass on the received authenticator and hop count without changing them as specified in the previous section. Thus, the effectiveness of this approach is limited.

In addition, there is another type of attack that cannot be detected by SAODV: tunneling attacks. In that type of attack, two malicious nodes simulate that they have a link between them (that is, they can send and receive messages directly to each other). They achieve this by tunneling

AODV messages between them (probably in an encrypted way). In this way they could achieve having certain traffic through them.

In our opinion, no security scheme has been able, so far, to detect this. Misbehaving detection schemes could, in principle, detect the so-called tunnel attacks. If the monitor sees a routing message with $Hop\_Count = X + 1$ being sent by a node but did not see a routing message with $Hop\_Count = X$ being sent to the same node, then the node is either fabricating the routing message or there is a tunnel. In either case it is cause for raising the alarm. Nevertheless, this kind of scheme has as main problems that there is no way for any node to validate the authenticity of the misbehavior reports and the there is the possibility of falsely detecting misbehavior nodes. Therefore, we don't consider it as a feasible solution so far.

The way the hop count is authenticated could be changed to a more secure one. For instance, intermediate nodes forwarding the routing messages could include the address of the next hop to which the message it is forwarded and sign it [26]. Another possibility would be to use forward-secure signature schemes [13]. A forward-secure signature scheme is like a hash chain, except that to prove that you are n hops away from the target you should sign the routing message with the key corresponding to the n^{th} link. Unlike in the hash chain case, the same signing key is not given to the next hop. Only the next signing key is given. This prevents the attack based on the possibility that a malicious node does not increase the hop count when it forwards a routing message. With this scheme, at any time the routing message has only one signature. The problem is, of course, efficiency. There are schemes where the message sizes are reasonably small, but signing and verification are quite expensive. Then there are other schemes where RSA signing could be used, but the public key needed to verify the signatures is size $O(m)$, where m is the diameter of the network. All those approaches would be very expensive (probably not even feasible) and, still, it would not prevent tunneling attacks at all. Therefore, we consider that the use of hash chains might be, so far, the option that deals best with the tradeoff between security and performance.

The use of sequence numbers should prevent most of the possible reply attacks. A node will discard a replied message if it has received a original message because the replied message won't be "fresh enough". In order to make the prevention of reply attacks stronger, a node could consider to increase its sequence number in more situations than what AODV mandates (or even periodicaly).

Papadimitratos and Haas suggest in [19] that it is possible to mount an attack by maliciously modifying the IP header of the SAODV messages. This is not true because SAODV does not trust the contents of the IP header, and all the information that needs to operate is inside the AODV message and the SAODV extension.

6. OTHER ROUTING PROTOCOLS

In principle, the same approach that SAODV takes to protect AODV could be used to create a "secure version" of other routing protocols: Signing the non-mutable routing information by the node to which the route will be processed, and securing the hop count by hash chains. In the case there are some other mutable fields, it should be studied how to protect each of them.

Nevertheless, if the routing protocol has some other mutable information than the hop count (and it does not mutate in a predictable way), protecting this information might end up being quite complex. It will probably require that the intermediate nodes that mutate part of the message also have to sign it. This will, typically, imply a reduction of performance (due to all the additional cryptographic computations) and also a possible decrease of the overall security.

We look now roughly, just as an example, to the Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR) [11], trying to see how it could be secured.

SRP [19] and Ariadne [10] (both refered in Section 2) also attempt to secure DSR. Nevertheless, SRP requires that, for every route discovery, source and destination must have a security association between them and does not protect error messages. And, Ariadne requires clock synchronization, which we consider to be an unrealistic requirement for ad hoc networks.

When trying to secure DSR, the main difference with respect to AODV is that DSR includes in its routing messages the IP addresses of all the intermediate nodes that have forwarded the packet.

A first approach to secure DSR, with the scheme proposed in this paper, would be to make each of the intermediate nodes sign the routing message after adding its own IP address, and also to verify all the signatures in the routing message. But this would greatly decrease the performance of the routing discovery. And it is not really worthwhile if we think that the routes to the intermediate nodes are going to be used very seldom. Anyway, hash chains should be used to avoid that a malicious node would eliminate intermediate nodes and their signatures from the routing message (a very similar technique is also used in [10]).

Another solution would be that intermediate nodes would sign the routing message, but that a node would only verify the signature of an intermediate node in the case it needs to send a packet to this route. But it still requires all intermediate nodes to sign the message (which is not good when the message is a route request).

Therefore, maybe a better solution would be that intermediate nodes do not sign the message. And if later a node wants to use a route to one of the intermediate nodes it should ask with a unicast message for a signature that certifies that it is the one who it claims to be.

Obviously, a much more detailed analysis should be made to study the different attacks that can be performed against DSR and against this "secure DSR" to see if there are new attacks as a consequence of differences between AODV and DSR.

7. KEY MANAGEMENT

In Section 5.3, we assumed that each ad hoc node has a trustworthy means of checking the association between the address of some other node and the signature verification key of that node. Let us now consider how such key management could be achieved.

Bindings between public keys and other attributes is typically achieved by using public key certificates. One approach could be for a certification authority (CA) to issue such certificates. This is reasonable if ad hoc nodes could have permanent addresses. However, addressing in ad hoc networks is likely to follow recent trends towards dynamic address allocation and autoconfiguration [28, 3]. In these schemes,

typically a node picks a tentative address and checks if it is already in use by broadcasting a query. If no conflict is found, the node is allowed to use that address. If a conflict is found, the node is required to pick another tentative address and repeat the process.

One solution that has been proposed [17, 18] would be to pick a key pair, and map the public key to a tentative address in some deterministic way; if there is a collision, pick a new key pair and try again. This is relatively secure, although potentially expensive.

The approach of distributing the Certification Authority functionality among ad hoc nodes (by dividing the private keys into shares) discussed in [30] implies a huge overhead, and it may be ineffective in a network were partitions occur or where there is high mobility. In addition, it won't work at all in trivial scenarios like when a network partition is composed of only two nodes.

Besides how key distribution is achieved, when distributing a public key, this should be binded to the identity of the node (of course) and also to its netmask (in the case the node is a network leader). Another alternative is to assume that there are no network leaders in scenarios were it is not needed to have connectivity outside the AODV network. Either of both alternatives prevents the type attack 4 in which a malicious node becomes a black hole for a whole subnet by claiming that it is their network leader.

8. STATUS

For more detailed information about the format of the Signature Extensions and the Secure AODV operation we recommend that the reader take a look at the Secure Ad hoc On-Demand Distance Vector (SAODV) Routing draft [6].

SAODV is still a work in progress. We are currently trying to reduce the processing power requirements of SAODV due to the use of asymmetric cryptography. There has been some concern (e.g., [19], [9], [10]) that SAODV's signatures might require a processing power that might be excessive for certain kinds of ad hoc scenarios.

One of the authors, Manel Guerrero, has created an AODV implementation called NRC-AODV (NRC standing for Nokia Research Center). NRC-AODV, which already has all the basic AODV features, was tested in the first AODV interoperability test. SAODV is planned to be added to the NRC-AODV implementation in the near future.

9. ACKNOWLEDGMENTS

The authors want to thank all the people that have been discussing SAODV (in the MANET mailing list and also privately), the anonymous reviewers and John Marcow (who corrected spelling and grammar).

10. REFERENCES

[1] N. Asokan and P. Ginzboorg. Key agreement in ad-hoc networks. *Computer Communication Review*, 23(17):1627–1637, Nov. 2000.

[2] J. Broch, D. A. Maltz, D. B. Johnson, Y. C. Hu, and J. Jetcheva. A performance comparison of multi-hop wireless ad hoc network routing protocols. In *Proceedings of the Fourth Annual International Conference on Mobile Computing and Networking*, pages 85–97, 1998.

[3] S. Cheshire and B. Aboba. Dynamic configuration of ipv4 link-local addresses. IETF INTERNET DRAFT, zeroconf working group, June 2001. `draft-ietf-zeroconf-ipv4-linklocal-03.txt`.

[4] S. Cheung. An efficient message authentication scheme for link state routing. In *13th Annual Computer Security Applications Conference*, pages 90–98, 1997.

[5] B. Dahill, B. N. Levine, E. Royer, and C. Shields. A secure routing protocol for ad hoc networks. Technical Report UM-CS-2001-037, University of Massachusetts, Departament of Computer Science, Aug. 2001.

[6] M. Guerrero. Secure ad hoc on-demand distance vector (SAODV) routing. IETF MANET Mailing List, Message-ID 3BC17B40.BBF52E09@nokia.com, http://www.cs.ucsb.edu/ eroyer/txt/saodv.txt, Oct. 2001.

[7] Z. J. Haas, M. R. Pearlman, and P. Samar. The interzone routing protocol (IERP) for ad hoc networks. INTERNET DRAFT, MANET working group, July 2002. draft-ietf-manet-zone-ierp-02.txt.

[8] R. Hauser, A. Przygienda, and G. Tsudik. Reducing the cost of security in link state routing. In *Symposium on Network and Distributed Systems Security (NDSS '97)*, pages 93–99, San Diego, California, Feb. 1997. Internet Society.

[9] Y. C. Hu, D. Johnson, and A. Perrig. SEAD: Secure efficient distance vector routing for mobile wireless ad hoc networks. In *Fourth IEEE Workshop on Mobile Computing Systems and Applications (WMCSA '02), June 2002*, pages 3–13, June 2002.

[10] Y. C. Hu, A. Perrig, and D. Johnson. Ariadne: A secure on-demand routing protocol for ad hoc networks. Technical Report TR01-383, Rice University, Dec. 2001.

[11] D. B. Johnson et al. The dynamic source routing protocol for mobile ad hoc networks (DSR). INTERNET DRAFT, MANET working group, Feb. 2002. draft-ietf-manet-dsr-07.txt.

[12] S. Kent, C. Lynn, J. Mikkelson, and K. Seo. Secure border gateway protocol (S-BGP) — real world performance and deployment issues, 2000.

[13] H. Krawczyk. Simple forward-secure signatures from any signature scheme. In *ACM Conference on Computer and Communications Security*, pages 108–115, 2000.

[14] C. Madson and R. Glenn. The use of HMAC-MD5-96 within ESP and AH. Internet Request for Comment RFC 2403, Nov. 1998.

[15] C. Madson and R. Glenn. The use of HMAC-SHA-1-96 within ESP and AH. Internet Request for Comment RFC 2404, Nov. 1998.

[16] S. Marti, T. J. Giuli, K. Lai, and M. Baker. Mitigating routing misbehavior in mobile ad hoc networks. In *Proceedings of the Sixth Annual International Conference on Mobile Computing and Networking*, pages 255–265, 2000.

[17] G. Montenegro and C. Castelluccia. Statistically unique and cryptographically verifiable (SUCV) identifiers and addresses. Network and Distributed System Security Symposium (NDSS '02), Feb. 2002.

[18] G. O'Shea and M. Roe. Child-proof authentication for mipv6 (CAM). ACM Computer Communication

Review, Apr. 2001.

[19] P. Papadimitratos and Z. J. Haas. Secure routing for mobile ad hoc networks. SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS 2002), Jan 2002.

[20] C. E. Perkins, E. M. Royer, and S. R. Das. Ad hoc on-demand distance vector (AODV) routing. IETF INTERNET DRAFT, MANET working group, Jan. 2002. `draft-ietf-manet-aodv-10.txt`.

[21] R. Perlman. Fault-tolerant broadcast of routing information. In *Computer Networks, n. 7*, pages 395–405, 1983.

[22] A. Perrig, R. Canetti, D. Song, and D. Tygar. Efficient and secure source authentication for multicast. In *Network and Distributed System Security Symposium (NDSS'01)*, Feb. 2001.

[23] A. Perrig, R. Szewczyk, V. Wen, D. E. Culler, and J. D. Tygar. SPINS: security protocols for sensor netowrks. In *Proceedings of the Seventh Annual International Conference on Mobile Computing and Networking*, pages 189–199, 2001.

[24] S. Ramanathan and M. Steenstrup. A survey of routing techniques for mobile communications networks. *Mobile Networks and Applications*, 1(2):89–104, 1996.

[25] E. M. Royer and C.-K. Toh. A review of current routing protocols for ad hoc mobile wireless networks. *IEEE Personal Communications*, pages 46–55, Apr. 1999.

[26] B. R. Smith, S. Murthy, and J. J. Garcia-Luna-Aceves. Securing distance-vector routing protocols. In *Symposium on Network and Distributed Systems Security (NDSS '97)*, pages 85–92, San Diego, California, Feb. 1997. Internet Society.

[27] F. Stajano and R. Anderson. The resurrecting duckling: Security issues for ad-hoc wireless networks. In *Proceedings of the 7th International Workshop on Security Protocols*, number 1796 in Lecture Notes in Computer Science, pages 172–194. Springer-Verlag, Berlin Germany, Apr. 1999.

[28] S. Thomson and T. Narten. Ipv6 stateless address autoconfiguration. IETF Request for Comments, Dec. 1998. `RFC 2462`.

[29] K. Zhang. Efficient protocols for signing routing messages. In *Proceedings of the Symposium on Network and Distributed Systems Security (NDSS'98)*, July 2001.

[30] L. Zhou and Z. J. Haas. Securing ad hoc networks. *IEEE Network Magazine*, 13(6):24–30, November/December 1999.

APPENDIX

A. AODV MESSAGE FORMATS

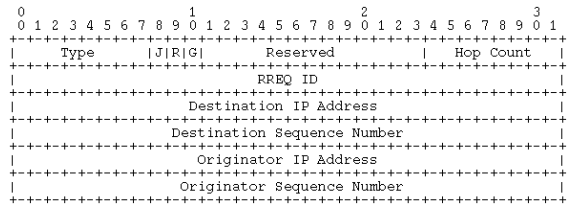


Figure 1: Route Request (RREQ) Message Format
Mutable fields: Hop Count

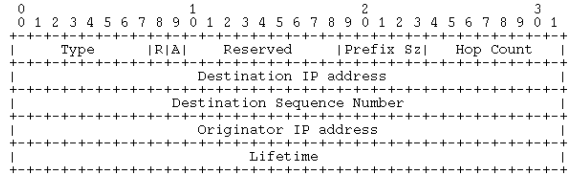


Figure 2: Route Reply (RREP) Message Format
Mutable fields: Hop Count

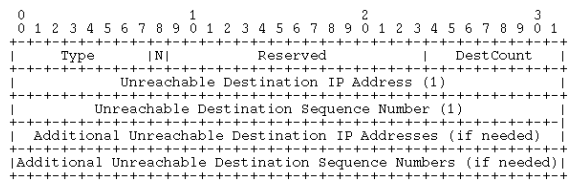


Figure 3: Route Error (RERR) Message Format
Mutable fields: None

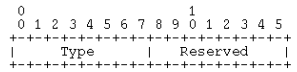


Figure 4: Route Reply Acknowledgment (RREP-ACK) Message Format
Mutable fields: None

B. SECURE AODV EXTENSIONS

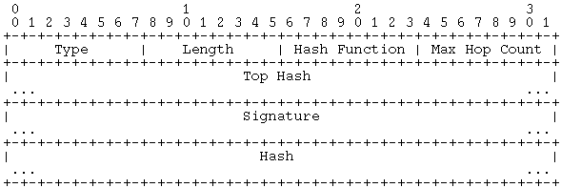


Figure 5: RREQ (Single) Signature Extension

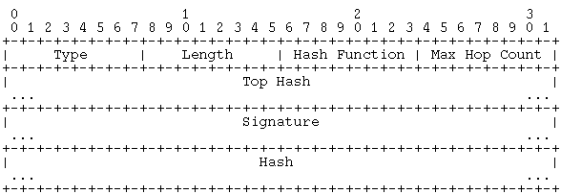


Figure 6: RREP (Single) Signature Extension

Field	Value
Type	64 in RREQ-SSE and 65 in RREP-SSE
Length	The length of the type-specific data, not including the Type and Length fields of the extension.
Hash Function	The hash function used to compute the Hash and Top Hash fields.
Max Hop Count	The Maximum Hop Count supported by the hop count authentication.
Top Hash	The top hash for the hop count authentication. This field has variable length, but it must be 32-bits aligned.
Signature	The signature of the all the fields in the AODV packet that are before this field but the Hop Count field. This field has variable length, but it must be 32-bits aligned.
Hash	The hash corresponding to the actual hop count. This field has variable length, but it must be 32-bits aligned.

Table 2: RREQ and RREP Signature Extension Fields

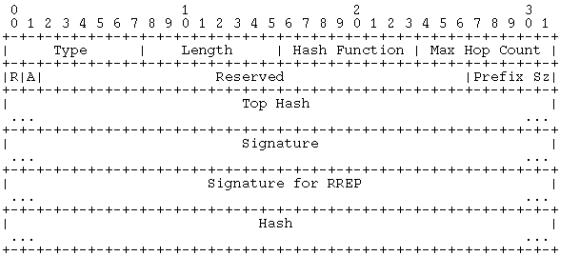


Figure 7: RREQ Double Signature Extension

Field	Value
Type	66
Length	The length of the type-specific data, not including the Type and Length fields of the extension.
Hash Function	The hash function used to compute the Hash and Top Hash fields.
Max Hop Count	The Maximum Hop Count supported by the hop count authentication.
R	Repair flag for the RREP.
A	Acknowledgment required flag for the RREP.
Reserved	Sent as 0; ignored on reception.
Prefix Size	The prefix size field for the RREP.
Top Hash	The top hash for the hop count authentication. This field has variable length, but it must be 32-bits aligned.
Signature	The signature of the all the fields in the AODV packet that are before this field but the Hop Count field. This field has variable length, but it must be 32-bits aligned.
Signature for the RREP	The signature that should be put into the Signature field of the RREP Double Signature Extension when an intermediate node (that has previously received this RREQ and created a reverse route) wants to generate a RREP for a route to the source of this RREQ. This field has variable length, but it must be 32-bits aligned. Both signatures are generated by the requesting node.
Hash	The hash corresponding to the actual hop count. This field has variable length, but it must be 32-bits aligned.

Table 3: RREQ Double Signature Extension Fields

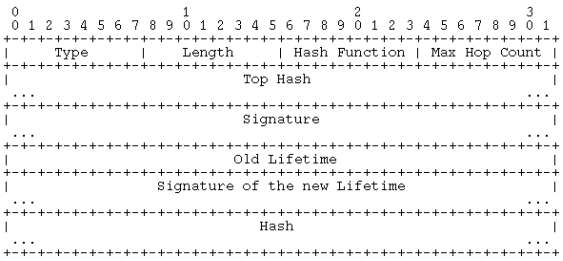


Figure 8: RREP Double Signature Extension

Field	Value
Type	67
Length	The length of the type-specific data, not including the Type and Length fields of the extension.
Hash Function	The hash function used to compute the Hash and Top Hash fields.
Max Hop Count	The Maximum Hop Count supported by the hop count authentication.
Top Hash	The top hash for the hop count authentication. This field has variable length, but it must be 32-bits aligned.
Signature	The signature of all the fields of the AODV packet that are before this field but the Hop Count field, and with the Old Lifetime value instead of the Lifetime. This signature is the one that was generated by the final destination. This field has variable length, but it must be 32-bits aligned.
Old Lifetime	The lifetime that was in the RREP generated by the final destination.
Signature of the new Lifetime	The signature of the RREP with the actual lifetime (the lifetime of the route in the intermediate node). This signature is generated by the intermediate node. This field has variable length, but it must be 32-bits aligned.
Hash	The hash corresponding to the actual hop count. This field has variable length, but it must be 32-bits aligned.

Table 4: RREP Double Signature Extension Fields

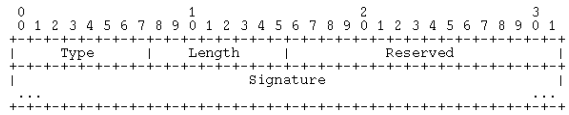


Figure 9: RERR Signature Extension

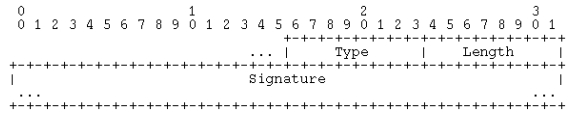


Figure 10: RREP-ACK Signature Extension

Field	Value
Type	68 in RERR-SE and 69 in RREP-ACK-SE
Length	The length of the type-specific data, not including the Type and Length fields of the extension.
Reserved	(Only in RERR-SE). Sent as 0; ignored on reception.
Signature	The signature of the all the fields in the AODV packet that are before this field. This field has variable length, but it must be 32-bits aligned.

Table 5: RERR and RREP-ACK Signature Extension Fields

Key Management and Delayed Verification for Ad hoc Networks

Manel Guerrero Zapata
Technology Department
Universitat Pompeu Fabra
Passeig de Circumval·lació 8, 08003 Barcelona
Email: manel.guerrero@upf.edu

***Abstract*— MANET (mobile and ad hoc networks) are networks in which nodes are mobile and link connectivity might change all the time. In this kind of networks key management is an important and complex problem.**

This paper studies how to design key management schemes for such networks that will allow to identify nodes without the need of any kind of certification authority. In addition, it presents a method to reduce the delays in route establishment in cases where routing messages are signed and need to be verified. Finally, it applies all these to SAODV (an extension of the AODV MANET routing protocol that protects the route discovery mechanism providing security features like integrity and authentication), and presents results from simulations that show how this method provides the same security with minimum impact in the network performance. Therefore, providing a more complete solution to the problem of security in MANET networks.

I. Introduction

In an ad hoc network, from the point of view of a routing protocol, there are two kinds of messages: the routing messages and the data messages. Both have a different nature and different security needs. Data messages are point-to-point and can be protected with any point-to-point security system (like IPSec). On the other hand, routing messages are sent to immediate neighbors, processed, possibly modified, and resent. Another consequence of the nature of the transmission of routing messages is that, in many cases, there will be some parts of those messages that will change during their propagation. This is very common in Distance-Vector routing protocols, where the routing messages usually contain a hop count of the route they are requesting or providing. Therefore, in a routing message one could distinguish between two types of information: mutable an non-mutable. It is desired that the mutable information in a routing message is secured in such a way that no trust in intermediate nodes is needed. Otherwise, securing the mutable information will be much more expensive in computation, plus the overall security of the system will greatly decrease.

Moreover, as a result of the processing of the routing message, a node might modify its routing table. This creates the need for the intermediate nodes to be able to authenticate the information contained in the routing messages (a need that does not exist in point-to-point communications).

SAODV [1] uses digital signatures to authenticate the non-mutable fields of the messages, and hash chains to secure the hop count information (the only mutable information in the messages). The use of digital signatures (asymmetric cryptography) has generated some concern (e.g., [2], [3], [4]) that SAODV's signatures might require a processing power that might be excessive for certain kinds of ad hoc scenarios and that not providing a key management scheme that explains how nodes get the public keys they require it does not solve the whole problem.

This paper studies both problems and provides a general solution and a specific method for SAODV. Section II takes a look at related work. Section III considers different ways to achieve the key management in MANET networks. Section IV provides a method that reduces the required processing power due to the use of asymmetric cryptography. Section V gives an overview of AODV. Section VI describes the security mechanism to protect AODV's routing information: Secure AODV (SAODV) [1]. Section VII focuses on how the key management methods explained in this paper can be used in conjunction with SAODV. Finally, section VIII presents simulation results of using SAODV with delayed verification.

II. Related Work

There is very little published prior work on the security issues in ad hoc network routing protocols. Neither the survey by Ramanathan and Steenstrup [5] nor the survey by Royer and Toh [6] mention security. None of the proposals in the IETF *MANET* working group have a non-trivial "security considerations" section. Actually, most of them assume that all the nodes in the network are friendly, and a few declare the problem out-of-scope by assuming some canned solution like IPSec may be applicable.

In their paper on securing ad hoc networks [7], Zhou and Haas primarily discuss key management (key management is discussed in Section III). They devote a section to secure routing, but essentially conclude that "nodes can protect routing information in the same way they protect data traffic". They also observe that denial-of-service attacks against routing will be treated as damage and routed around.

Security issues with routing in general have been addressed by several researchers (e.g., [8], [9]). And, lately, some work has been done to secure ad hoc networks by using misbehavior detection schemes (e.g., [10]). This approach has two main problems: first, it is quite likely that it will be not feasible

to detect several kinds of misbehaving (especially because it is very hard to distinguish misbehaving from transmission failures and other kind of failures); and second, it has no real means to guarantee the integrity and authentication of the routing messages.

Dahill et al. [11] proposed ARAN, a routing protocol for ad hoc networks that uses authentication and requires the use of a trusted certificate server. In ARAN, every node that forwards a route discovery or a route reply message must also sign it, (which is very computing power consuming and causes the size of the routing messages to increase at each hop), whereas the proposal presented in this paper only require originators to sign the message. In addition, it is prone to reply attacks using error messages unless the nodes have time synchronization.

Papadimitratos and Haas [2] proposed a protocol (SRP) that can be applied to several existing routing protocols (in particular DSR [12] and IERP [13]). SRP requires that, for every route discovery, source and destination must have a security association between them. Furthermore, the paper does not even mention route error messages. Therefore, they are not protected, and any malicious node can just forge error messages with other nodes as source.

Hash chains have being used as an efficient way to obtain authentication in several approaches that tried to secure routing protocols. In [9], [14] and [15] they use them in order to provide delayed key disclosure. While, in [16], hash chains are used to create one-time signatures that can be verified immediately. The main drawback of all the above approaches is that all of them require clock synchronization.

We suggested the use of hash chains to authenticate hop counts [17], [1]. This technique is used in SAODV. In SEAD [3] (by Hu, Johnson and Perrig) hash chains are also used in combination with DSDV-SQ [18] in a very similar way (this time to authenticate both hop counts and sequence numbers). At every given time each node has its own hash chain. The hash chain is divided into segments, elements in a segment are used to secure hop counts in a similar way as it is done in SAODV. The size of the hash chain is determined when it is generated. After using all the elements of the hash chain a new one must be computed.

SEAD can be, in theory, used with any suitable authentication and key distribution scheme. But finding such a scheme is not straightforward.

Ariadne [4], by the same authors, is based on DSR [12]. The authentication mechanism of Ariadne is based on TESLA [19]. It also requires clock synchronization. Clock synchronization introduces a big overhead in the network due to the messages needed to be exchanged to achieve it. Therefore, it is arguably not appropriate for MANET protocols.

It is quite likely that, for a small team of nodes that trust each other and that want to create an ad hoc network where the messages are only routed by members of the team, the simplest way to keep secret their communications is to encrypt all messages (routing and data) with a "team key". Every member of the team would know the key and, therefore, it would be able to encrypt and decrypt every single packet. Nevertheless, this does not scale well and the members of the team have to trust each other. So it can be only used for a subset of the possible scenarios.

This is why SAODV uses asymmetric cryptography. But then, the challenge is to design a key management scheme that works in a mobile and ad hoc network where you cannot assume network connectivity with any kind of server.

Solving this challenge is the one of the aims of this paper.

III. Key Management in MANET Networks

One of the most important consequences of the nature of the MANET networks is that one cannot assume that a node that is part of a network will be always reachable by all the other nodes. This implies that there cannot be servers in the conventional meaning of the fixed networks. Therefore, the use of Certification Authorities (CAs) in MANET networks is not feasible.

The approach of distributing the Certification Authority functionality among ad hoc nodes (by dividing the private keys into shares) discussed in [7] implies a huge overhead, and it may be ineffective in a network were partitions occur or where there is high mobility. In addition, it will not work at all in trivial scenarios like when a network partition is composed of only two nodes.

Another characteristic of servers in fixed networks, besides its continuous availability, is the fact that clients have to know the server's IP address (or to know its human address and have the IP address of a DNS server). The same thing happens in MANET networks for any node you want to make a request or initiate an exchange of data.

However, current trends about addressing in ad hoc networks are driving towards dynamic address allocation and autoconfiguration [20], [21]. In these schemes, typically a node picks a tentative address and checks if it is already in use by broadcasting a query. If no conflict is found, the node is allowed to use that address. If a conflict is found, the node is required to pick another tentative address and repeat the process.

But then, If IP addresses do not identify a node (because they are dynamically allocated), how does a node know the IP address of the node to which it wants to sent data. In fixed networks, if a node wants to send data to another one, it needs to know its address (it cannot send anything to a node that has a dynamic address, because it does not know its IP address).

The Binding between public keys and other attributes is typically achieved by using public key certificates. In some limited scenarios, a possible approach could be for a certification authority (that would live in a fixed network) to issue such certificates that the nodes could collect before going to the MANET "playground". However, this is not feasible for a big group of the targeted scenarios. An added problem is that the IP address should be one of the attributes binded to the public keys, because it is binded to your identity.

To sum up, what is required is a system that achieves that: IP addresses will be assigned dynamically, nodes will be identifiable by their IP addresses, there should be a binding

between the public key and the IP address of a node, and all this without any kind of certification authorities. Which is quite a challenge.

A couple of papers [22], [23] have proposed a solution to solve the “address ownership” problem in the context of Mobile IP. It consists in to pick a key pair, and map the public key to a tentative address in some deterministic way. Our earlier paper [1] already proposed that this approach of ”cryptographically generated addresses” could be used in the key management for SAODV. In this paper, we describe the details of CGA-based key management.

If a node ’A’ receives a routing message that is signed by a node ’B’ that has the same IP address than one of the nodes for which ’A’ has a route entry (node ’C’), it will not process normally that routing message. Instead, it will inform ’B’ that it is using a duplicated IP and it will prove it by adding the public key of ’C’ (so ’B’ can verify the truthfulness of the claim).

When the node ’B’ receives a routing message that indicates that somebody else has the same IP address than itself (or it realizes about it by itself), it will have to generate a new pair of public/private keys. After that, it will derive its IP address from its public key and it might inform all the other nodes (through a broadcast) of which is its new IP address with an special message that contains: the two IP addresses (the old and the new ones) and the two public signatures (old and new) signed with the old private key and, all this, signed with the new private key. Nevertheless, it is much better if, that message, is unicast (instead of broadcast) to all the nodes it considers that should receive this information (in the case they are just a few). This unicast will be answered with an acknowledge message by the receiver if it verifies that everything is in order.

After this, the node will generate a route error message for his old IP address. Its propagation will delete the route entries for the old IP address and, therefore, eliminate the duplicated addresses. This route error message may have a message extension that tells which is the new address. In this way, the nodes that receive the routing message can already create the route to the new IP address.

This solution allows two nodes to coexist in the same network with the same IP address until one of them realizes about it. However, in the author’s opinion, it gives a good trade-off between the impact of changing address (and having a coexisting period of two nodes with the same IP address) and the extremely low probability of having address collision.

Intermediate nodes could decide to store the IP addresses and public keys of all the nodes they would meet (or of the last ’N’ nodes, depending on their capabilities). That would allow an earlier detection of duplicated IP addresses in the network.

An alternative to this solution could be that, when a node detects that another node is using the same IP address, it would keep its public/private key pair and change the used IP address by applying a salt to the algorithm that derives the IP address from the public key. Salt variations of hash algorithms have been used in order to avoid dictionary attacks of passwords [24]. The “salt” is a random string that is added to the password before being hashed. This idea can be adapted with a very different purpose. If the statistically unique IP address is the derived from the public key and a salt (instead of only from the public key), the node that detects or is informed that its IP address is also used by another node can change its IP address without change its public key by just changing the salt.

Nevertheless, that would imply that the salt used by a node should be included in all the routing messages and stored in all the entries of the routing tables. And, still, the node has to inform the others of its change of IP address. Therefore, it will not be used for the purpose of this paper.

In conclusion, the approach described in this section handles properly the very unlikely situation of two nodes with the same IP address, without adding any complexity to the typical situation. Next section, explains how to reduce the number of verification of signatures which reduces importantly the computer power required by a node to run SAODV.

IV. Delayed Verification of Signatures

As stated in the introduction, there has been some concern (e.g., [2], [3], [4]) that SAODV’s signatures might require a processing power that might be excessive for certain kinds of ad hoc scenarios. This section addresses this problem by revising one of SAODV’s security requirements from the list that was stated in [1].

A. Security Requirements

The security requirements that will be provided are source authentication and integrity (that combined provide data authentication) and delayed import authorization.

Import authorization was defined in [1] as:

- **Import authorization**: The ultimate authority about routing messages regarding a certain destination node is that node itself. Therefore, a node will only authorize route information in its routing table if that route information concerns the node that is sending the information. In this way, if a malicious node lies about it, the only thing it will cause is that others will not be able to route packets to the malicious node.

Delayed import authorization allows to have route entries and route entry deletions in the routing table that are pending of verification. They will be verified whenever the node has spared processor time or before these entries should be used to forward data packages.

The security requirements will not include confidentiality and non-repudiation because they are not necessarily critical services in the context of routing [9]. They will not include either availability (since an attacker can focus on the physical layer without bothering to study the routing protocol) and they will not address the problem of compromised nodes (since it is arguably not critical in non military scenarios).

B. How does it work?

In reactive ad hoc routing protocols, most of the routing messages that circulate in the network are (by far) route requests. This is due to the fact that route requests are broadcast. Route replies are unicast back through the selected path. And, route error messages are unicast down through the tree of nodes that had a route to the now unreachable node that is advertised by the route error message.

When a node receives a routing message, it creates a new entry in its routing table (the so called "reverse route"). Therefore, after the broadcast of the route request, all the nodes in the network (or in the broadcast ring) have created reverse routes to the originator of the route request. From all these reverse routes, most of them will expire soon (typically all but the ones that are in the selected path through which the route reply will travel).

Then, the question is: why should all this route requests be verified (with the consequent delay in the propagation of the broadcast), when most of them are going to be soon discarded. The answer is: there is no need to verify them until the corresponding route reply comes back and the node knows that it is in the selected path. The other reverse routes will expire without being verified.

Actually, the two signatures (the ones from the route request and route reply) will be verified after the node has forwarded the route reply. In this way transmissions of the route requests and replies occur without any kind of delay due to the verification of the signatures.

Following the same idea, the signature of route error messages (and in general, any routing message that has to be forwarded) can also be verified after forwarding them.

Routes pending of verification will not be used to forward any packet. If a packet arrives for a node for which there is a route pending of verification. The node will have to verify it before using that route. If the verification fails, it will delete the route and request a new one.

V. AODV

This section gives an introduction to AODV, necessary to understand how it is secured and how the key management technique is applied to it.

Ad Hoc On-Demand Vector Routing (AODV) protocol [25] is a reactive routing protocol for ad hoc and mobile networks that maintains routes only between nodes which need to communicate. The routing messages do not contain information about the whole route path, but only about the source and the destination. Therefore, routing messages do not have an increasing size. It uses destination sequence numbers to specify how fresh a route is (in relation to another), which is used to grant loop freedom.

Whenever a node needs to send a packet to a destination for which it has no 'fresh enough' route (i.e., a valid route entry for the destination whose associated sequence number is at least as great as the ones contained in any RREQ that the node has received for that destination) it broadcasts a route request (RREQ) message to its neighbors. Each node that receives the broadcast sets up a reverse route towards the originator of the RREQ (unless it has a 'fresher' one). When the intended destination (or an intermediate node that has a 'fresh enough' route to the destination) receives the RREQ, it replies by sending a Route Reply (RREP). It is important to note that the only mutable information in a RREQ and in a RREP is the hop count (which is being monotonically increased at each hop). The RREP travels back to the originator of the RREQ (this time as a unicast). At each intermediate node, a route to the destination is set (again, unless the node has a 'fresher' route than the one specified in the RREP). In the case that the RREQ is replied to by an intermediate node (and if the RREQ had set this option), the intermediate node also sends a RREP to the destination. In this way, it can be granted that the route path is being set up bidirectionally. In the case that a node receives a new route (by a RREQ or by a RREP) and the node already has a route 'as fresh' as the received one, the shortest one will be updated.

If there is a subnet (a collection of nodes that are identified by a common network prefix) that does not use AODV as its routing protocol and wants to be able to exchange information with an AODV network, one of the nodes of the subnet can be selected as their 'network leader'. The network leader is the only node of the subnet that sends, forwards and processes AODV routing messages. In every RREP that the leader issues, it sets the prefix size of the subnet.

In addition to these routing messages, Route Error (RERR) messages are used to notify the other nodes that certain nodes are not anymore reachable due to a link breakage.

VI. SAODV

SAODV assumes that there is a key management sub-system that makes it possible for each ad hoc node to obtain public keys from the other nodes of the network. Further, each ad hoc node is capable of securely verifying the association between the identity of a given ad hoc node and the public key of that node. This paper provides a possible solution of how this can be achieved. This section provides an overview to SAODV that will be need it to understand how this solution is applied to SAODV. Please, refer to [1] for a detailed analysis of SAODV.

Two mechanisms are used to secure the AODV messages: digital signatures to authenticate the non-mutable fields of the messages, and hash chains to secure the hop count information (the only mutable information in the messages). For the non-mutable information, authentication is perform in an end-to-end manner, but the same kind of techniques cannot be applied to the mutable information.

The information relative to the hash chains and the signatures is transmitted with the AODV message as an extension message (let us refer to it as Signature Extension). To see the exact format of the SAODV Signature Extensions, please, refer to the version 0 of the SAODV draft [26].

A. SAODV hash chains

SAODV uses hash chains to authenticate the hop count of RREQ and RREP messages in such a way that allows every

TABLE I
POSSIBLE VALUES OF THE SIGNATURE METHOD FIELD

Value	Signature method
0	Reserved
1	RSA [27]
2	Elliptic curve [28]
3-127	Reserved
128-255	Implementation dependent

node that receives the message (either an intermediate node or the final destination) to verify that the hop count has not been decremented by an attacker.

The delayed verification could also be applied to the hash chains. But, since the time that it requires to verify a hash chain is practically negligible, there is no need for that.

B. SAODV digital signatures

Digital signatures are used to protect the integrity of the non-mutable data in RREQ and RREP messages. That means that they sign everything but the Hop_Count of the AODV message and the Hash from the SAODV extension.

When a RREQ is received by the destination itself, it will reply with a RREP only if it fulfills the AODV's requirements to do so. This RREP will be sent with a RREP Signature Extension.

When a node receives a RREP, it first verifies the signature before creating or updating a route to that host. Only if the signature is verified, will it store the route with the signature of the RREP and the lifetime.

VII. SAODV WITH DELAYED VERIFICATION AND KEY MANAGEMENT

This section shows how SAODV could be modify to implement the different techniques developed in this paper.

A. New fields

The public key should be included in the routing messages that are signed, so that the nodes can verify the signature. Since, obviously, that public key should be signed by the signature, it is placed before the signature field.

The identifier of the algorithm that is used to sign the message is specified in the Signature_Method field. The possible values are shown in Table I (being mandatory to support RSA). Since SAODV could allow more than one possible signature method, it might happen that a node has to verify a signature with a method it does not know. If this happens the node will consider that the verification of the signature has failed.

This implies that all the nodes that form part of a MANET network should know all the methods used by all the other nodes to sign their messages. This is not a problem since, typically, all nodes of a MANET network will use the same method (or two different methods the most). The fact that there is more than one possible signature methods is because different networks may have tighter security requirements than some others and, therefore, use different signature methods.

B. Network Leaders

The original SAODV design established that besides how key distribution is achieved, when distributing a public key, this should be binded to the identity of the node (of course) and also to its netmask (in the case the node is a network leader). This was to prevent the type attack in which a malicious node becomes a black hole for a whole subnet by claiming that it is their network leader.

In the new approach presented in this paper, ad hoc nodes will typically never be network leaders. Network leaders will be only fixed nodes that typically give access to the fixed network and the nodes in the MANET network should know their IP addresses, prefix size and public keys.

Network leaders will not change its IP address in case that there is a MANET node that happen to generate the same IP address. A node generating its IP address will check if the resulting IP address corresponds to the network leader or to the subnet corresponding to its prefix size. A node detecting another node using the network leader IP address or any of the ones corresponding to the leader subnet will inform to the MANET node, and not to the network leader.

C. Generation of the IP address

SAODV can generate the IP addresses is very similar to the generation of SUCV (Statistically Unique and Cryptographically Verifiable) addresses [22]. SUCV addresses where designed to protect Binding Updates in Mobile IPv6. The main difference between SUCV and the method proposed in this paper is that SUCV addresses are generated by hashing an "imprint" in addition to the public key. That imprint (that can be a random value) is used to limit certain attacks related to Mobile IP.

In SAODV, the address can be a network prefix of 64 bits with a 64 bit SAODV_HID (Half IDentifier) or a 128 bit SAODV_FID (Identifier). These two identifiers are generated almost in the same way than the sucvHID and the sucvID in SUCV (with the difference that they do not include an imprint):

$$SAODV\_HID = SHA1HMAC\_64(PublicKey)$$

$$SAODV\_FID = SHA1HMAC\_128(PublicKey)$$

There will be a flag in the SAODV routing message extensions (the 'H' flag) that will be set to '1' if the IP address is a HID and to '0' if it is a FID.

Finally, if it has to be a real IPv6 address, there is a couple of things that should be done [29].

If HID is used, then the HID behaves as an interface identifier and, therefore, its sixth bit (the universal/local bit) should be set to zero (0) to indicate local scope (because the IP address is not guaranteed to be globally unique).

And, if FID is used, then a format prefix corresponding to the MANET network should be overwritten to the FID. Format prefixes '010' through '110' are unassigned and would take only three bits of the FID. Format prefixes '1110' through '1111 1110 0' are also unassigned and they would take

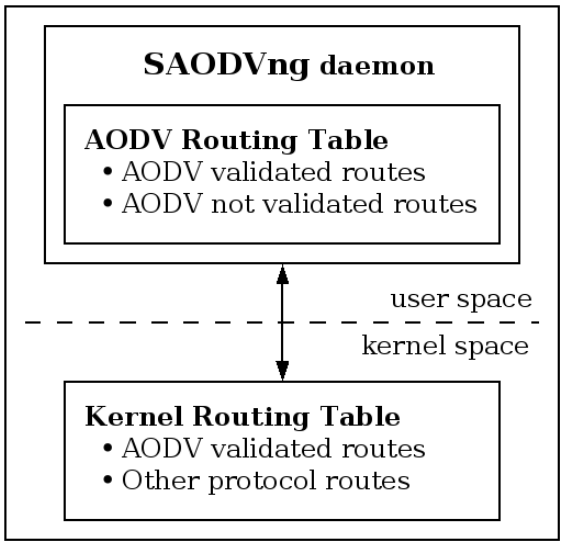


Fig. 1. SAODV daemon

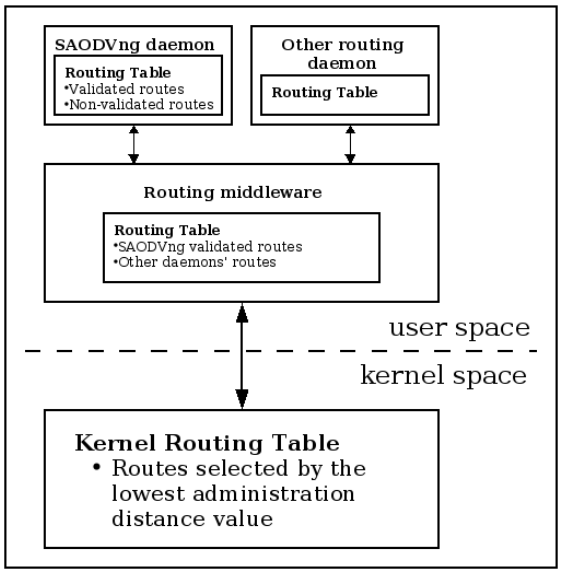


Fig. 2. SAODV daemon with a routing middleware

between 4 and 9 bits of the FID. All of these format prefixes required to have to have 64-bit interface identifiers in EUI-64 format, so universal/local bit should be set to zero (0).

This paper does not propose a scheme for IPv4 since the author considers the length of an IPv4 address to be too short to provide the statistical uniqueness that this scheme requires.

D. Duplicated IP Address Detection

SAODV can deal with the duplicated IP address problem as described in section III. Duplicate Address (DADD) Detected message is send to notify to a node that its address is already being used by another node. New Address (NADD) Notification Message is used to inform that the node has change key pair and IP address. Finally, New Address Acknowledgment (NADD-ACK) Message is used to confirm the reception of the NADD. In SAODV, NADD is always unicast (never broadcast).

E. Implementation Considerations

When a node needs to send or to forward a packet to a destination for which it does not have an active route, first it will check if it has a route pending of validation. If it does, it will try to validate it and, if it was successfully validated, it will mark it as active and use it. If after all this there is not an active route the node will start a route discovery process.

As shown in figure 1, only once the validation is done successfully, the route is incorporated in the routing table of the node. That avoids doing dirty hacks into the routing table of the operating system of the node: The packets can be routed normally, and only when there is a route lookup that the routing table cannot resolve, the petition is captured by the SAODV routing daemon.

Figure 2 shows that in the case where there is a routing middleware (like zebra[1] or quagga[2]), the middleware routing table will contain the validated routes from the SAODV daemon combined with the ones from the other routing daemons and the routing table in the kernel the ones with lowest "administrative distance" (in case there is a route to the same destination provided by two different routing daemons).

Talking about administrative distances, none of the MANET routing protocols that are being designed or standardized have specified which would be the appropriate administrative distance for them. Let us look to the "standard de facto" (Cisco, Zebra, etc.) default administrative distance values. Probably a good default distance value would be between 160 (Cisco's On-Demand Routing) and 170 (external routes in EIGRP). Therefore, this paper recommends a default distance value of 165 for SAODV (and also for AODV in general).

VIII. Simulation Results

The simulations were done with 30 nodes moving at a maximum speed of 10 meters per second in a square of 1000x1000 meters. They established 10 connections that started between second 0 and second 25 (according to an uniform distribution). The simulation time was of 100 seconds, and the connections where constant bit rate (a packet of 512 each 0.25 seconds).

The simulations have used as routing protocols: plain AODV, SAODV with RSA, SAODV with ECC (Elliptic Curve Cryptography), and SAODV with delayed verification (SAODV2 in the figure) with ECC. There is no point to use delayed verification with RSA since its verification time is completely negligible. RSA and ECC have used key lengths with equivalent security (1368 bit RSA and 160 bit ECC).

Table II shows the times for signing/verifying in a Compaq iPAQ 3670 (206Mhz, 16M ROM, 64M RAM) according to

[1]www.zebra.org
[2]www.quagga.net

TABLE II
TIMES FOR A COMPAQ IPAQ 3670

	RSA	DSA	ECC
Key length	1368	1368	160
Sign	210	90	42
Verify	6	110	160

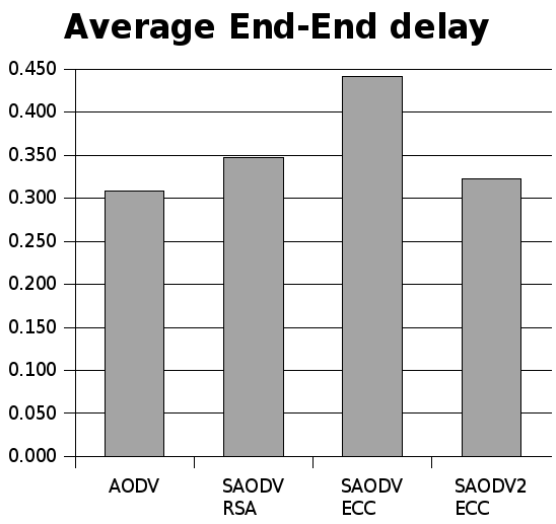


Fig. 3. Simulation Results
The delay is measured in milliseconds

[30]. DSA is not used in the simulations as it presents the worst of RSA and ECC (slow signature and verification, and fast increase of computational overhead as the key length needs to be bigger).

Figure 3 shows the averaged result of the simulations. There were practically no differences among the routing protocols in packet delivery fraction (that was around 90 percent) and in normalized routing load (that was around 1).

One could expect quite different results with some other simulation scenarios, but almost always having SAODV with delayed verification and ECC as the best of the SAODV options and with a performance very close to plain AODV.

In the future, when longer keys are needed, ECC results will look even better than with the key lengths used in these simulations. This is due to the fact that, as they key size increases the computational overhead of ECC increases much more slowlier.

IX. CONCLUDING REMARKS

Although it is true that there is no way to preclude a node of inventing many identities, that cannot be used to create an attack against the secure routing algorithm.

Delayed verification makes possible that a malicious node creates invalid route requests that could flood the MANET network. But, the same malicious node can flood the network with perfectly valid route requests. And there would be no easy way to know if it is trying to flood the network or if it is just trying to see if any of its friend nodes are present in the network (for instance).

As explained in the paper an attacker cannot forge a public/private key pair from an IP address so the identity token becomes the IP address itself.

With the current technology, SAODV with delayed verification and ECC provides security features to AODV with an almost negligible performance penalty.

In the future, when longer keys are required, the gain of using delayed verification in conjunction to ECC compared to other SAODV options will be even bigger that it is nowadays.

ACKNOWLEDGMENT

This work was supported in part by CYCIT TIC2003-09279-C02-01 and by the I2CAT Foundation.

The author wants to thank all the people from the Nokia Research Center in Helsinki (where he worked for five years) that helped to make SAODV a reality. Special mention deserve his colleague N. Asokan and his bosses Jari Juopperi and Asko Vilavaara.

He also wants to thank Ana Escudero, from the Department of Technology at the Universitat Pompeu Fabra, for her help with the simulations.

REFERENCES

[1] M. Guerrero Zapata and N. Asokan, "Securing Ad hoc Routing Protocols," in *Proceedings of the 2002 ACM Workshop on Wireless Security (WiSe 2002)*, September 2002, pp. 1–10.

[2] P. Papadimitratos and Z. J. Haas, "Secure routing for mobile ad hoc networks," SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS 2002), Jan 2002.

[3] Y. C. Hu, D. Johnson, and A. Perrig, "SEAD: Secure efficient distance vector routing for mobile wireless ad hoc networks," in *4th IEEE Workshop on Mobile Computing Systems and Applications (WMCSA '02), June 2002*, June 2002, pp. 3–13. [Online]. Available: citeseer.nj.nec.com/hu02sead.html

[4] Y. C. Hu, A. Perrig, and D. Johnson, "Ariadne: A secure on-demand routing protocol for ad hoc networks," Rice University, Tech. Rep. TR01-383, Dec. 2001. [Online]. Available: citeseer.nj.nec.com/531013.html

[5] S. Ramanathan and M. Steenstrup, "A survey of routing techniques for mobile communications networks," *Mobile Networks and Applications*, vol. 1, no. 2, pp. 89–104, 1996. [Online]. Available: citeseer.nj.nec.com/ramanathan96survey.html

[6] E. M. Royer and C.-K. Toh, "A review of current routing protocols for ad hoc mobile wireless networks," *IEEE Personal Communications*, pp. 46–55, Apr. 1999.

[7] L. Zhou and Z. J. Haas, "Securing ad hoc networks," *IEEE Network Magazine*, vol. 13, no. 6, pp. 24–30, November/December 1999. [Online]. Available: citeseer.nj.nec.com/zhou99securing.html

[8] B. R. Smith, S. Murthy, and J. J. Garcia-Luna-Aceves, "Securing distance-vector routing protocols," in *Symposium on Network and Distributed Systems Security (NDSS '97)*. San Diego, California: Internet Society, Feb. 1997, pp. 85–92.

[9] R. Hauser, A. Przygienda, and G. Tsudik, "Reducing the cost of security in link state routing," in *Symposium on Network and Distributed Systems Security (NDSS '97)*. San Diego, California: Internet Society, Feb. 1997, pp. 93–99.

[10] S. Marti, T. J. Giuli, K. Lai, and M. Baker, "Mitigating routing misbehavior in mobile ad hoc networks," in *Proceedings of the 6th Annual International Conference on Mobile Computing and Networking*, 2000, pp. 255–265. [Online]. Available: citeseer.nj.nec.com/marti00mitigating.html

[11] B. Dahill, B. N. Levine, E. Royer, and C. Shields, "A secure routing protocol for ad hoc networks," University of Massachusetts, Departament of Computer Science, Tech. Rep. UM-CS-2001-037, Aug. 2001.

[12] D. B. Johnson *et al.*, "The dynamic source routing protocol for mobile ad hoc networks (DSR)," INTERNET DRAFT, MANET working group, Feb. 2002, draft-ietf-manet-dsr-07.txt.

[13] Z. J. Haas, M. R. Pearlman, and P. Samar, "The interzone routing protocol (IERP) for ad hoc networks," INTERNET DRAFT, MANET working group, July 2002, draft-ietf-manet-zone-ierp-02.txt.

[14] S. Cheung, "An efficient message authentication scheme for link state routing," in *13th Annual Computer Security Applications Conference*, 1997, pp. 90–98.

[15] A. Perrig, R. Szewczyk, V. Wen, D. E. Culler, and J. D. Tygar, "SPINS: security protocols for sensor netowrks," in *Proceedings of the 7th Annual International Conference on Mobile Computing and Networking*, 2001, pp. 189–199. [Online]. Available: citeseer.nj.nec.com/article/perrig01spins.html

[16] K. Zhang, "Efficient protocols for signing routing messages," in *Proceedings of the Symposium on Network and Distributed Systems Security (NDSS'98)*, July 2001.

[17] N. Asokan, "Presentation at an informal workshop on mobile and ad hoc networking security, EPFL, Lausanne, December 2001," Dec. 2001.

[18] J. Broch, D. A. Maltz, D. B. Johnson, Y. C. Hu, and J. Jetcheva, "A performance comparison of multi-hop wireless ad hoc network routing protocols," in *Proceedings of the 4th Annual International Conference on Mobile Computing and Networking*, 1998, pp. 85–97. [Online]. Available: citeseer.nj.nec.com/broch98performance.html

[19] A. Perrig, R. Canetti, D. Song, and D. Tygar, "Efficient and secure source authentication for multicast," in *Network and Distributed System Security Symposium (NDSS'01)*, Feb. 2001. [Online]. Available: citeseer.nj.nec.com/perrig01efficient.html

[20] S. Thomson and T. Narten, "Ipv6 stateless address autoconfiguration," IETF Request for Comments, Dec. 1998, RFC 2462.

[21] S. Cheshire and B. Aboba, "Dynamic configuration of ipv4 link-local addresses," IETF INTERNET DRAFT, zeroconf working group, June 2001, draft-ietf-zeroconf-ipv4-linklocal-03.txt.

[22] G. Montenegro and C. Castelluccia, "Statistically unique and cryptographically verifiable (SUCV) identifiers and addresses," Network and Distributed System Security Symposium (NDSS '02), Feb. 2002. [Online]. Available: citeseer.nj.nec.com/502628.html

[23] G. O'Shea and M. Roe, "Child-proof authentication for mipv6 (CAM)," ACM Computer Communication Review, Apr. 2001. [Online]. Available: http://doi.acm.org/10.1145/505666.505668

[24] A. J. Menezes, P. C. van Oorschot, and S. A. Vanstone, *The Handbook of Applied Cryptography*. CRC Press, 1996, iSBN 0-8493-8523-7. [Online]. Available: http://perlmonks.thepen.com/113573.html

[25] C. E. Perkins, E. M. Belding-Royer, and S. R. Das, "Ad hoc on-demand distance vector (AODV) routing," Internet Request for Comments RFC 3561, Nov. 2003.

[26] M. Guerrero Zapata, "Secure ad hoc on-demand distance vector (saodv) routing," first published in the IETF MANET Mailing List (October 8th 2001), Aug. 2002, iNTERNET-DRAFT draft-guerrero-manet-saodv-00.txt.

[27] R. Rivest, A. Shamir, and L. Adleman, "A Method for Obtaining Digital Signatures and Public-Key Cryptosystems," *Communications of the ACM*, vol. 21, no. 2, February 1978.

[28] RSA Laboratories, "Elliptic Curve Cryptography Standard," Public-Key Cryptography Standards (PKCS) 13, 1998.

[29] R. Hinden and S. Deering, "Ip version 6 addressing architecture," Internet Request for Comments RFC 2373, 1998.

[30] J. Walter, J. Oleksy, and J. Kong, "The role of ecdsa in wireless communications," Master Thesis. Computer Science Department. University of California, 2002.

Shortcut Detection and Route Repair in Ad hoc Networks

Manel Guerrero Zapata
Technology Department
Universitat Pompeu Fabra
Passeig de Circumval·lació 8, 08003 Barcelona
Email: manel.guerrero@upf.edu

Abstract

When a routing protocol for manet networks (mobile and ad hoc networks) does a route discovery, it does not discover the shortest route but the route through which the route request flood traveled faster. In addition, since nodes are moving, a route that was the shortest one at discovery time might stop being so in quite a short period of time. This causes, not only a much bigger end-to-end delay, but also more collisions and a faster power consumption.

In order to avoid all the performance loss due to these problems, this paper develops a technique to periodically discover shortcuts to the active routes that can be used with any destination vector routing protocol. It also shows how the same mechanism can be used as a bidirectional route recovery mechanism.

1 Introduction

One of the main consequences, if we are using IEEE 802.11b wireless cards, of the so-called "gray zones" defined and studied in [4], is that the shortest path is not always a good path. "Gray zones" are zones in which a node can receive short broadcast messages but not reasonably large data messages from a certain node.

This means that finding a method to discover the shortest path (or a shorter path) is not necessarily a very good idea. Since there is nothing that tells us that this newly discovered route will be better than the original one.

Nevertheless, if we use one of the proposed solutions mentioned in [4] we can be quite sure that the routes will not contain any "gray link". And then, the shortest path will be, with high probability one of the best path (if not the best).

Section 2 describes the mechanism used to perform shortcut detection. Section 3 explains how the same techniques can be applied to perform route repair. Section 4 gives a short overview of AODV needed to understand the following section. Section 5 studies how to use the mechanisms described in this paper in conjunction with AODV as base routing protocol. Finally, section 6 takes a look at related work.

2 Shortcut Detection

In order to detect possible shortcuts in active routes, source and destination of any active route will periodically (with jitterized periods) start a shortcut discovery by issuing a "Shortcut Request" (SREQ). The SREQ will be a routing message that will be broadcast with TimeToLive equal to zero.

Therefore, all the neighbors will receive it (without the need to be in promiscuous mode) but will not re-broadcast it. If any of the nodes that receive the request knows a shortcut it will reply with a "Shortcut Reply" (SREP) that will contain information about that shortcut. The node (or nodes) that will receive a SREQ, are part of the current route and have not generated SREQ yet, will also generate another SREQ.

This will be done until the other end of the bidirectional route receives an SREQ and also originates one. This last SREQ is sent because shortcuts that use a different link to that end of the route would not be discovered otherwise.

One could think that generating SREQs with TimeToLive equal to 1 would make more shortcuts to be discovered. Nevertheless, this would generate much more network traffic in the zone.

2.1 Shortcut Routes

Upon receival of SREQs, nodes will create "shortcut routes" with a shorter live time than the other routes. And, in a very similar manner to how reverse routes work, they will be deleted quite soon if the node does not receive a SREP confirming that there is a shortcut. The difference is that shortcut routes cannot be used until they are not confirmed by the corresponding SREP, since we do not know

if, in effect, there is a shortcut nor the resulting distance in hops of using that shortcut.

2.2 Shortcut Requests

Shortcut Requests will contain the both ends of the bidirectional current route with the corresponding distance in hops to them. In the case the routing protocol uses sequence numbers, it will also include the corresponding sequence numbers of the routes to both ends.

2.3 Shortcut Replies

Shortcut Replies might just be normal route reply messages, maybe with a flag that indicates that they are a shortcut reply.

2.4 One-Hop Shortcuts

One-hop shortcuts occur when two non-neighbor nodes that were part of a route discover that they are now neighbors. They are quite easy to discover: When a node receives a SREQ, it can check if the node can be part of a one-hop shortcut by doing the following:

- Verify that the sequence numbers of the route to each of the end points of the bidirectional route are the same in the received SREQ and in its routing table.
- Verify that the hop count of the bidirectional route (sum of the hop counts to each of the end points of the received SREQ or of its own routing tables) is bigger than the hop count of the possible shortcut route (sum of the smallest hop count of the received SREQ with the hop count of the route to the other end point registered in the routing table plus one).

Figure 1 shows an example. There is a bidirectional route from $'S'$ to $'D'$. $'A'$ and $'C'$ move towards each other until they become neighbors. If $'S'$ starts a shortcut discovery, $'A'$ will send a SREQ that $'C'$ will receive. Then, it will see that the current bidirectional route ($1+3$ or $3+1$) is longer than the possible shortcut ($1+1+1$) and it will send SREPs to both end points through $'A'$ and $'C'$.

2.5 Two-Hop Shortcuts

Two-hop shortcuts occur when a node is neighbor of two nodes that are part of a route in which they are three or more hops away from each other.

In order to be able to detect two-hop shortcuts, each node needs to keep track of the SREQs that it has received recently (just in the same way that it needs to keep track of the rebroadcast route requests to avoid rebroadcasting the same route request more than once).

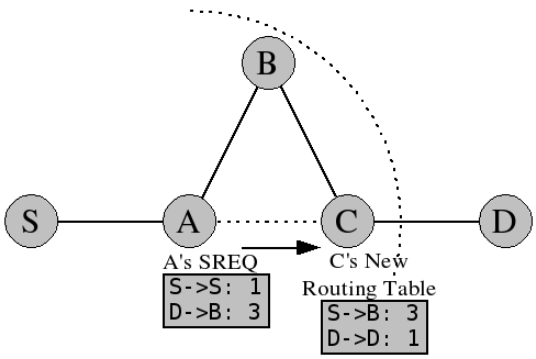


Figure 1. One-hop Shortcut Detection

If a node receives a SREQ for a route which has already received another SREQ, it will check if it is part of a two-hop shortcut route by doing the following:

- Verify that the sequence numbers of the route to each of the end points of the bidirectional route are the same in both SREQs.
- Verify that the hop count of the bidirectional route (sum of the hop counts to each of the end points of any of the two SREQs) is bigger than the hop count of the possible shortcut route (sum of the smallest hop count of one SREQ with the smallest hop count of the other SREQ plus two).

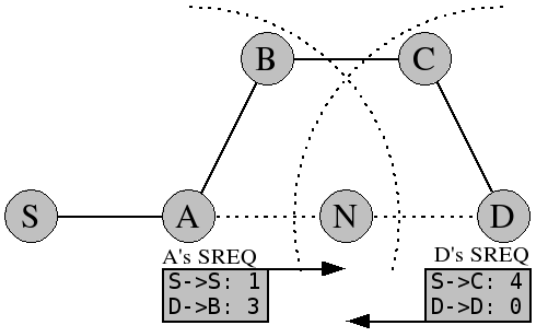


Figure 2. Two-hop Shortcut Detection

Figure 2 shows an example. There is a bidirectional route that goes from $'S'$ to $'D'$. Nodes move, and later on $'N'$ is a new neighbor of $'A'$ and $'D'$. Either $'S'$ or $'D'$ decide that it is time for a new shortcut discovery. This shortcut discovery will be propagated through the route. Therefore, $'N'$ will receive SREQs from both $'A'$ and $'D'$. It will

check their contents and it will see that the hop count of the current bidirectional route ($1+3$ or $4+0$) is bigger than the hop count of the shortcut route ($1+0+2$). And it will send SREPs to both end points of the bidirectional route through $'A'$ and $'D'$.

In the case $'N'$ would be neighbor of only $'A'$ and $'C'$, it will see that the current route ($1+3$ or $3+1$) is not longer than the possible shortcut ($1+1+2$).

2.6 Other Kind of Shortcuts

Figure 3 shows an example of a three-hop shortcut that cannot be shortcut in any way by the method presented in this paper. This kind of situation will happen seldomly in networks with certain density of nodes.

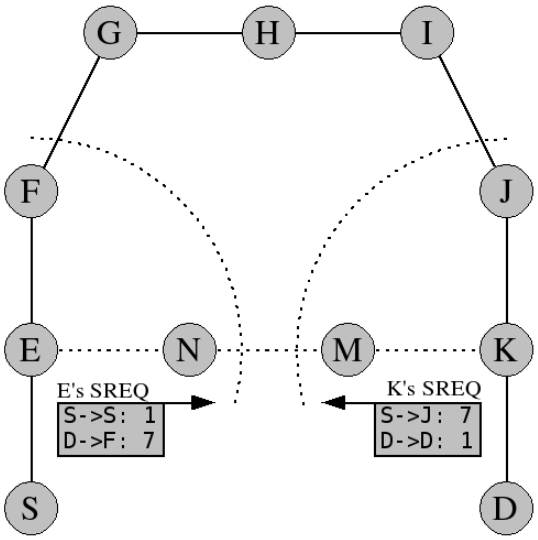


Figure 3. Three-hop Shortcut

However, in a network with certain density of nodes it is highly probable that routes that are not the shortest ones can be shortcut by one-hop and two-hop shortcuts.

A method could be designed to also detect three-hop shortcuts, but it would imply too much traffic. Basically, it could consist in that receivers of the shortcut requests from the nodes that are in the current route would also send the request to its neighbors.

3 Route Repair

Some routing protocols in manet networks have a mechanism to try to repair a broken route (due to a link breakage) that does not imply a complete route discovery. An example would be the "local repair" in AODV [6] in which when a link used to send data packets breaks, the node upstream of the link that got broken may (if it was close to the destination) do a route discovery of the destination broadcasting the route request with a TimeToLive that is assumed to be enough to reach the destination.

This method has the problem that it only repairs the route in one direction. Chances are, that the route is used in both directions. Therefore, if it only repairs the route in one direction, another route discovery will be needed to repair the route in the other direction.

A possible solution, would be to use the shortcut discovery method described in this paper to do the route repair. To do so, when a link breakage occurs, the two nodes that were connected through that link will initiate a "repaired route discovery". This repaired route discovery will consist of sending a SREQ to the end of the route to which they are still connected. The differences with a normal SREQ message will be:

- The message will be flagged as repair route SREQ.
- The hop count to the endpoint that is not available anymore will be set to infinity (typically indicated by the value 255).
- Optionally, the original SREQ (the one originated by one of the two nodes that were connected through that link) might be also forwarded by all their immediate neighbors that were not part of the original route. Of course, if they forward it, the forwarded SREQ should have increased the hop count that is not set to infinity in the SREQ (to account for the new hop that has been done).

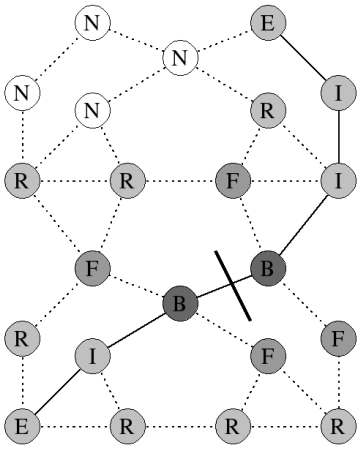


Figure 4. Propagation of SREQs

Figure 4 shows how SREQs are propagated. End points of the previous route are marked as $'E'$. The two nodes that where connected through the link that has just broken are

mark as $'B'$ and intermediate nodes that where part of the route as $'I'$. The neighbors of the $'B'$ nodes that are not part of the route but will forward the SREQ are marked as $'F'$ nodes. The rest of the nodes that will receive a SREQ are marked as $'R'$ nodes. Finally, the other nodes are marked as $'N'$.

Due to the fact that the neighbors of the $'B'$ nodes (the $'F'$ nodes) forward the SREQs, there will be a broader diffusion of the SREQs in the zone nearby the link breakage.

Figure 5 shows how routes get repaired. When $'N'$ receives both SREQs and updates its routing table, it will send both SREPs to $'S'$ and $'D'$ and the route will be repaired. The routing table of $'N'$ is updated in the following manner: Since it receives two SREQs for the route between $'S'$ and $'D'$, one of them with a finite hop count to $'S'$ and the other with a finite hop count to $'D'$, it can deduce that it can do a shortcut route between $'S'$ and $'D'$ incrementing the finite hopcounts that were in the SREQs by one (to account for the last hop the messages did to arrive to $'N'$).

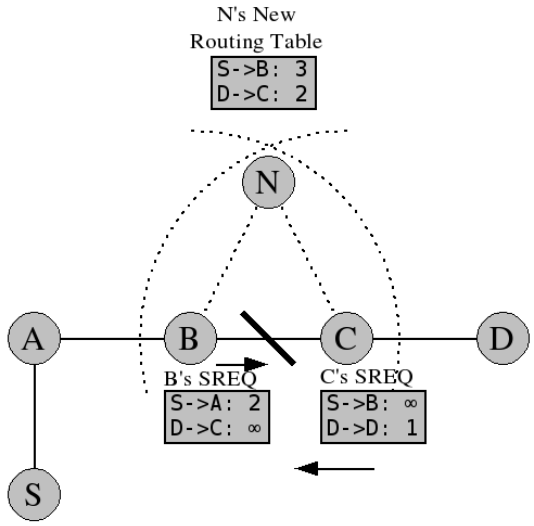


Figure 5. Simple Repaired Route Discovery

Figure 6 shows a more complicated example, in which the neighbors of the nodes that were connected through the link that broke will also forward the SREQ. This increases the chances of getting the route repaired, but implies a little bit more traffic.

4 AODV

This section gives an introduction to AODV, necessary to understand how the techniques explained in this paper can be applied to it.

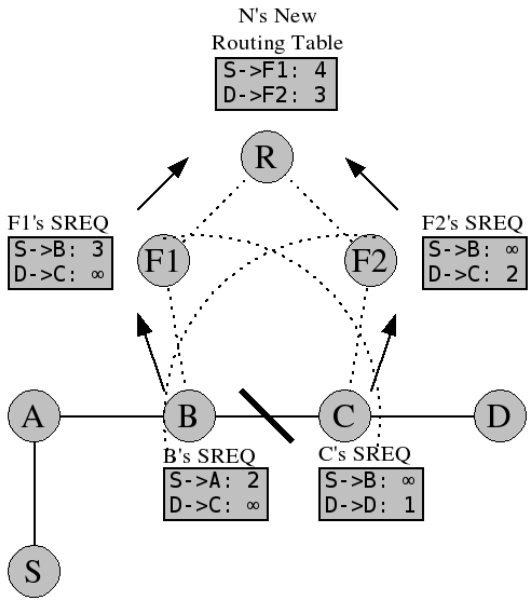


Figure 6. Repaired Route Discovery with Additional Forwarding by neighbors

Ad Hoc On-Demand Vector Routing (AODV) protocol [6] is a reactive routing protocol for ad hoc and mobile networks that maintains routes only between nodes which need to communicate. The routing messages do not contain information about the whole route path. Instead, they only contain information about the source and destination nodes. Therefore, routing messages do not have an increasing size. It uses destination sequence numbers to specify how fresh a route is (in relation to another), which is used to grant loop freedom.

Whenever a node needs to send a packet to a destination for which it has no 'fresh enough' route (i.e., a valid route entry for the destination whose associated sequence number is at least as great as the ones contained in any RREQ that the node has received for that destination) it broadcasts a route request (RREQ) message to its neighbors. Each node that receives the broadcast sets up a reverse route towards the originator of the RREQ (unless it has a 'fresher' one). When the intended destination (or an intermediate node that has a 'fresh enough' route to the destination) receives the RREQ, it replies by sending a Route Reply (RREP). It is important to note that the only mutable information in a RREQ and in a RREP is the hop count (which is being monotonically increased at each hop). The RREP travels back to the originator of the RREQ (this time as a unicast). At each intermediate node, a route to the destination is set (again, unless the node has a 'fresher' route than the one specified

in the RREP). In the case that the RREQ is replied to by an intermediate node (and if the RREQ had set this option), the intermediate node also sends a RREP to the destination. In this way, it can be granted that the route path is being set up bidirectionally. In the case that a node receives a new route (by a RREQ or by a RREP) and the node already has a route 'as fresh' as the received one, the shortest one will be updated.

In addition to these routing messages, Route Error (RERR) messages are used to notify the other nodes that certain nodes are not anymore reachable due to a link breakage.

5 AODV-SDR

AODV-SDR (AODV with shortcut discovery and route repair) incorporates two new types of messages: Shortcut REQuest (SREQ) and Shortcut REPly (SREP). The format of these messages is shown in figures 7 and 8, and its fields are specified in the tables1 and 2.

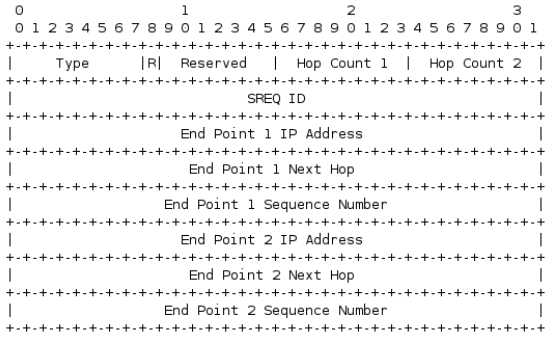


Figure 7. Shortcut Request (SREQ) Message Format

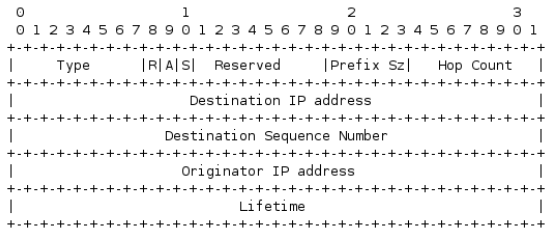


Figure 8. Shortcut Reply (RREP) Message Format

SREQs have a "R flag" that is set if the SREQ is used to do a route repair. They also contain a "SREQ ID", that is a sequence number that identifies uniquely the SREQ with

Table 1. Shortcut Request Message Fields

Field	Value
Type	64
R flag	The flag that indicates that this is a repair route SREQ.
Reserved	Sent as 0; ignored on reception.
Hop Count 1	The hop count to the end point 1.
Hop Count 2	The hop count to the end point 2.
SREQ ID	A sequence number uniquely identifying the SREQ with then end point 1.
End Point 1 IP Address	The IP address of the end point 1.
End Point 1 Next Hop	The next hop in the route to the end point 1.
End Point 1 Sequence Number	The sequence number of the route to the end point 1.
End Point 2 IP Address	The IP address of the end point 2.
End Point 2 Next Hop	The next hop in the route to the end point 2.
End Point 2 Sequence Number	The sequence number of the route to the end point 2.

Table 2. Shortcut Reply Message (RREP with the 'S' flag) Fields

Field	Value
Type	2
R flag	Repair flag. It is used for multicast.
A flag	Acknowledgment required flag.
S flag	The flag that indicates that is a shortcut reply.
Reserved	Sent as 0; ignored on reception.
Prefix Size	Specifies the prefix size of the address.
Hop Count	The hop count from the originator to the destination.
Destination IP Address	The IP address of the destination.
Destination Sequence Number	The destination sequence number associated to this route.
Originator IP Address	The IP address of the node for which the route is supplied.
Lifetime	The lifetime in milliseconds of the route.

the end point that originated the SREQ. In case this SREQ was originated due to a route repair both nodes that where connected through the link that broke will generate SREQs that will probably have different sequence numbers.

SREQs also contain the following information about both end points of the route: IP address , the next hop of the route that goes to the end point, the sequence number of that route, and the hop count to the end point.

SREPs are basically AODV's "Route Reply" (RREP) messages with a flag set to indicate that they are SREPs. Once the shortcut is discovered they propagate back the shortcut route. Therefore they contain all the information about that route: hop count, IP address, lifetime, etc.

6 Related Work

In discussions in the manet mailing list was argued that distance vector routing protocols where not discovering the shortest route, but the one through which the route requests where broadcasted faster.

"Coping with Communication Gray Zones in IEEE 802.11b based Ad hoc Networks" by Lundgren et al. [4] (published in 2002) was a paper that came out as a result of physical experimentation with real ad hoc networks. In the paper they find out different solutions to cope with the "gray zones" (zones through which short broadcast packets are received but not data packets).

Therefore, it was clear that, on one hand, distance vector routing protocols did not discover the shortest path and that, on the other hand, the shortest path was not always the best path (not if one of the links is a "gray link").

In 2003, some publications tried to address this problem. Like SHORT [2], that also tries to find shortcuts, but in order to do that all data packets must carry certain information (like a "hop count"). Which is a very strong requirement that would, arguably, render it unfeasible for a lot of scenarios. It is always tempting to add control information in data packets. It simplifies the problem you are trying to solve, but it complicates extremely the problems in the rest of the system.

In addition, SHORT fails to find shortcuts in a very simple scenarios (when the shortcut involves the source node, when the short cut involves the destination node, ...). Finally, it also fails to look at the routes as bidirectional routes.

A paper by De Couto et al. [1] defines a metric to measure the throughput of a multi-hop route to be used with DSDV [5] and DSR [3]. It expects that nodes will calculate the throughput using dedicated link probe packets. But, it is not clear how a node collects information of the links that are far away from him to do all the statistics.

There are other papers, that try to address route efficiency by finding minimum energy disjoint paths, like this one by Srinivas and Modiano [7].

7 Conclusion

This paper provides a technique that can be used with any destination vector routing protocol in a manet network to periodically discover shortcuts to the active routes. Therefore, making the network communications much more optimal. In addition, it also shows how the same mechanism can be used as a bidirectional route recovery mechanism. Finally, it specifies how to implement this techniques on top of the AODV routing protocol.

Acknowledgment

This work was supported in part by CYCIT TIC2003-09279-C02-01 and by the I2CAT Foundation.

The author wants to thank his new colleagues at the Universitat Pompeu Fabra for their support, friendship and help (specially with all those little things that might end up burying you into a red tape mountain).

References

[1] Douglas S. J. De Couto, D. Aguayo, J. Bicket, and R. Morris. A high-throughput path metric for multi-hop wireless routing. In *Proceedings of the 9th annual international conference on Mobile computing and networking*, pages 134 – 146, September 2003.

[2] C. Gui and P. Mohapatra. SHORT: self-healing and optimizing routing techniques for mobile ad hoc networks. In *Proceedings of the 4th ACM international symposium on Mobile ad hoc networking and computing*, pages 279 – 290, June 2003.

[3] D. B. Johnson, D. A. Maltz, and Y.-C. Hu. The dynamic source routing protocol for mobile ad hoc networks (DSR). INTERNET DRAFT, MANET working group, Apr. 2003. draft-ietf-manet-dsr-09.txt.

[4] H. Lundgren, E. Nordstrm, and C. Tschudin. Coping with communication gray zones in IEEE 802.11b based ad hoc networks. In *Proceedings of the 5th ACM international workshop on Wireless mobile multimedia*, pages 49 – 55, September 2002.

[5] C. Perkins and P. Bhagwat. Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers. In *ACM SIGCOMM'94 Conference on Communications Architectures, Protocols and Applications*, pages 234–244, 1994.

[6] C. E. Perkins, E. M. Belding-Royer, and S. R. Das. Ad hoc on-demand distance vector (AODV) routing. Internet Request for Comments RFC 3561, Nov. 2003.

[7] A. Srinivas and E. Modiano. Minimum energy disjoint path routing in wireless ad-hoc networks. In *Proceedings of the 9th annual international conference on Mobile computing and networking*, pages 122 – 133, September 2003.

Mobile Ad Hoc Networking Working Group
INTERNET DRAFT
2 February 2005

Manel Guerrero Zapata
Technical University
of Catalonia (UPC)

Secure Ad hoc On-Demand Distance Vector (SAODV) Routing
draft-guerrero-manet-saodv-05.txt

Intellectual Property Rights Statement

Status of this Memo

Copyright

Abstract

The Secure Ad hoc On-Demand Distance Vector (SAODV) is an extension of the AODV routing protocol that can be used to protect the route discovery mechanism providing security features like integrity and authentication.

Table of Contents

1. Introduction

In an ad hoc network, from the point of view of a routing protocol, there are two kinds of messages: the routing messages and the data messages. Both have a different nature and different security needs. Data messages are point-to-point and can be protected with any point-to-point security system (like IPSec). On the other hand, routing messages are sent to immediate neighbors, processed, possibly modified, and resent.

Another consequence of the nature of the transmission of routing messages is that, in many cases, there will be some parts of those messages that will change during their propagation. This is very common in Distance-Vector routing protocols, where the routing messages usually contain a hop count of the route they are requesting or providing. Therefore, in a routing message one could distinguish between two types of information: mutable an non-mutable. It is desired that the mutable information in a routing message is secured in such a way that no trust in intermediate nodes is needed. Otherwise, securing the mutable information will be much more expensive in computation, plus the overall security of the system will greatly decrease.

Moreover, as a result of the processing of the routing message, a node might modify its routing table. This creates the need for the intermediate nodes to be able to authenticate the information contained in the routing messages (a need that does not exist in point-to-point communications).

SAODV is an extension of the AODV[1] routing protocol that protects the route discovery mechanism providing security features like integrity and authentication. It uses digital signatures to authenticate the non-mutable fields of the messages, and hash chains to secure the hop count information (the only mutable information in the messages).

SAODV can use the Simple Ad hoc Key Management (SAKM)[2] as a key management system.

2. Preliminary notes

It is important to have in mind that this paper is describing how to protect the routing messages, not the data messages. This section contains some preliminary notes about which security features SAODV provides, and about IPSec interacting with SAODV.

2.1. Security Features

Before designing a protocol extension that provides security to AODV it is required to think what are the security needs and what issues just cannot be solved. The main thing that cannot be avoid is that there might be malicious nodes that do not respect protocols (they will forge AODV packets, listen to the others, reply packets in their own interests, report errors where there are none, etc).

It is needed to have integrity, authentication. But what about confidentiality? Well, maybe it is needed for scenarios with a very high security needs, but it does not make sense if the scenario is a public ad hoc network that everybody can joint at any moment. Therefore, it is not taken into account in the proposed protocol extension.

2.2. Interaction with IPSec

When trying to use IPSec to secure network transmissions in a MANET network, it is needed that the IPSec implementation can use as a selector the TCP or UDP port number. Sadly, there are quite many implementations that cannot do that. The importance of that is because it is needed that the IPSec policy will be able to apply certain security mechanisms to the data packets and just bypass the routing packets.

3. Overview

The solution presented in this paper is an extension of the AODV protocol mainly by using new extension messages. In these extension messages there is a signature of the AODV packet with the private key of the original sender of the Routing message (not of the intermediate nodes that just forward it).

Concerning to RREQ and RREP messages there are two alternatives: The first one in which only final destinations are allowed to reply a RREQ, and the second in which there is no such limitation.

In the first one, when a RREQ is sent, the sender signs the message. Intermediate nodes verify the signature before creating or updating a reverse route to that host. And only if the signature is fine they store the reverse route. The final destination node signs the RREP with its private key. Intermediate and final nodes, again verify the signature before creating or updating a route to that host, also storing the signature with the route entry.

In the second one, when a RREQ is sent, the sender signs the message. Intermediate nodes verify the signature before creating or updating a

reverse route to that host. And, again, only if the signature is fine they store the reverse route. But the difference is that the RREQ message has also a second signature that is always stored with the reverse route. This second signature is needed to be added in the gratuitous RREPs of that RREQ and in regular RREPs to future RREQs that the node might reply as an intermediate nodes. An intermediate node that wants to reply a RREQ needs not only the correct route, but also the signature corresponding to that route to add it in the RREP and the 'Lifetime' and the 'Originator IP address' fields that work with that signature. If it has them, it generates the RREP, (adding the stored signature, lifetime and the originator IP address) signs the actual lifetime and the actual originator IP address and sends it. All the nodes that receive the RREP and that update the route store the signature the lifetime and the originator IP address with that route.

If a node wants to be able to reply as an intermediate node for a route to a node that has been added due to a RREQ or to a RREP, it has to store the 'RREQ Destination' or 'RREP Originator' IP address, the lifetime and the signature. And use them as the 'Signature', 'Old Lifetime', and 'Old Originator IP address' fields in the RREP-DSE message.

Hello messages are RREP messages, so they are signed in the same way. Hello Interval extensions are not signed. There is no attack from changing hello interval extension. Actually, if the hello interval extension would be added in the signature, the nodes that received a hello message from a node 'D' would not be able to reply as intermediate node when a node 'S' would issue a RREQ for 'D', because they wouldn't have a valid signature for the RREP without the hello interval extension.

Extension messages that include a second signature also include the RREP fields (right now only the prefix size) that are not derivable from the RREQ but not zeroed when computing the signature.

RREP-ACK messages may be authentified by using a digital signature, that might be verified by any one that receives them.

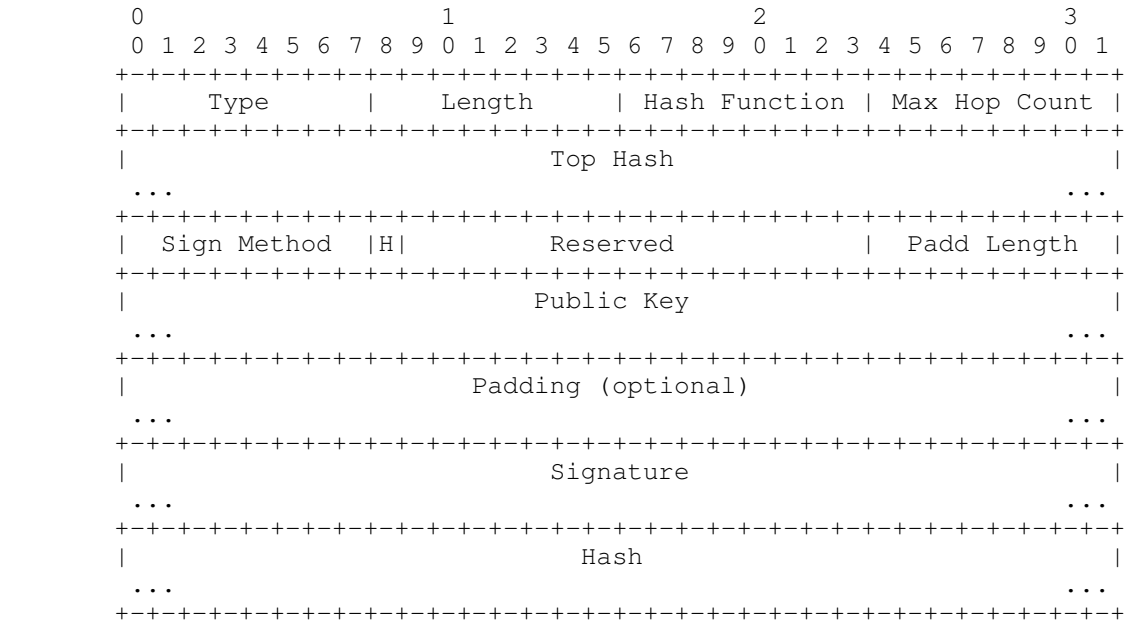
Every node, generating or forwarding a RERR message, uses digital signatures to sign the whole message and any neighbor that receives verifies the signature.

The hop count of all these messages is authentified by using a hash chain.

4. Terminology

This memo uses the conventional meanings [3] for the capitalized words MUST, SHOULD and MAY. It also uses terminology taken from the specifications of AODV and IPSec [4].

5. RREQ (Single) Signature Extension



Type 64

Length The length of the type-specific data, not including the Type and Length fields of the extension in bytes.

Hash Function
The hash function used to compute the Hash and Top Hash fields.

Max Hop Count
The Maximum Hop Count supported by the hop count authentication.

Top Hash The top hash for the hop count authentication. This field has variable length, but it must be 32-bits aligned.

Signature Method
: The signature method used to compute the signatures.

H
: Half Identifier flag. If it is set to '1' indicates the use of HID and if it is set to '0' the use of FID.

Reserved
: Sent as 0; ignored on reception.

Padding Length
: Specifies the length of the padding field in 32-bit units. If the padding length field is set to zero, there will be no padding.

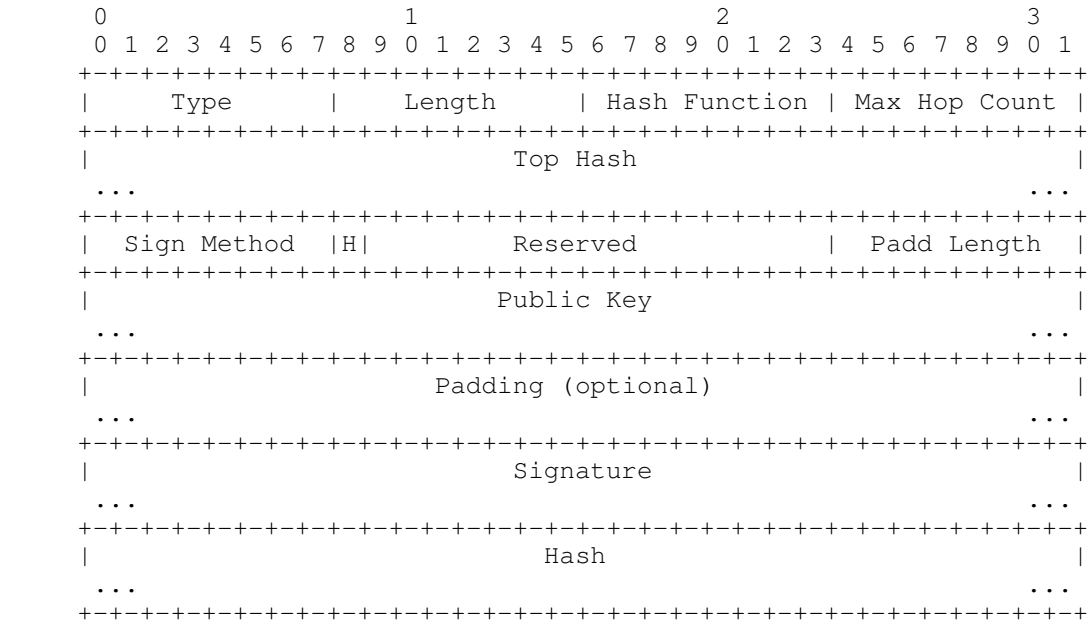
Public Key
: The public key of the originator of the message. This field has variable length, but it must be 32-bits aligned.

Padding
: Random padding. The size of this field is set in the Padding Length field.

Signature
: The signature of the all the fields in the AODV packet that are before this field but the Hop Count field. This field has variable length, but it must be 32-bits aligned.

Hash
: The hash corresponding to the actual hop count. This field has variable length, but it must be 32-bits aligned.

6. RREP (Single) Signature Extension



Type 65

Length The length of the type-specific data, not including the Type and Length fields of the extension in bytes.

Hash Function
The hash function used to compute the Hash and Top Hash fields.

Max Hop Count
The Maximum Hop Count supported by the hop count authentication.

Top Hash The top hash for the hop count authentication. This field has variable length, but it must be 32-bits aligned.

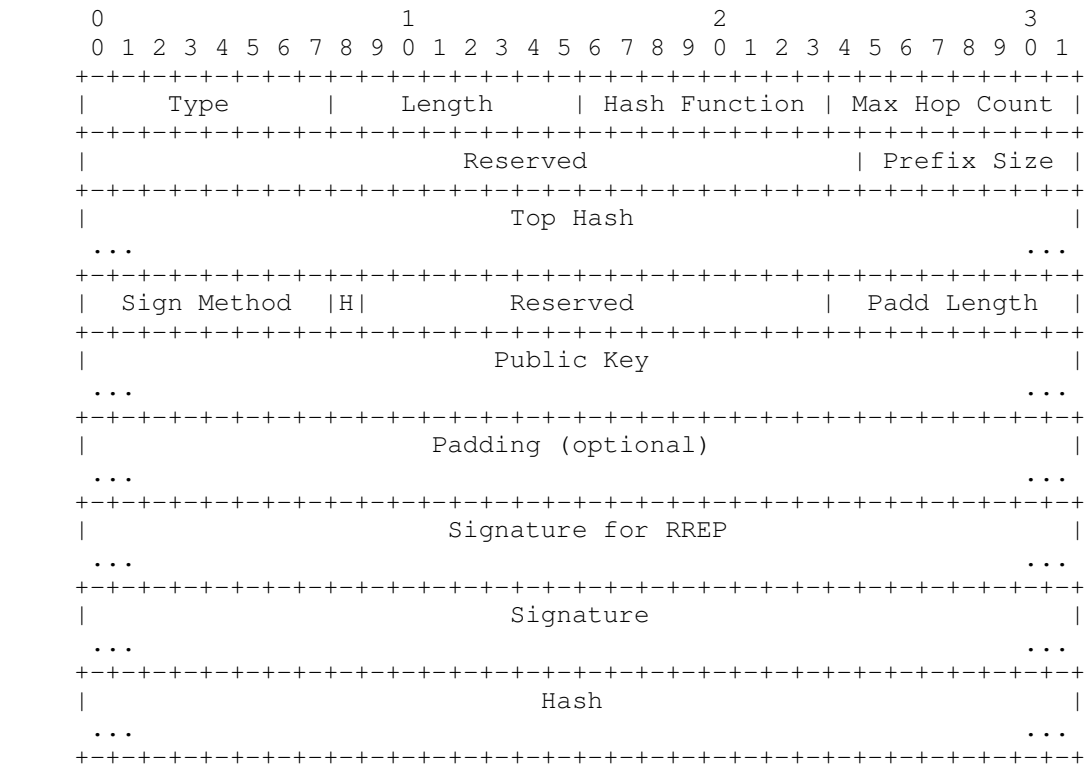
Signature Method ... Padding
The same than in RREQ (Single) Signature Extension.

Signature The signature of the all the fields in the AODV packet that are before this field but the Hop Count field. This field has variable length, but it must be 32-bits

aligned.

Hash The hash corresponding to the actual hop count. This field has variable length, but it must be 32-bits aligned.

7. RREQ Double Signature Extension



Type 66

Length The length of the type-specific data, not including the Type and Length fields of the extension in bytes.

Hash Function
The hash function used to compute the Hash and Top Hash fields.

Max Hop Count
The Maximum Hop Count supported by the hop count authentication.

Reserved Sent as 0; ignored on reception.

Prefix Size The prefix size field for the RREP (it is 7 bit long to allow IPv6 prefixes).

Top Hash The top hash for the hop count authentication. This field has variable length, but it must be 32-bits aligned.

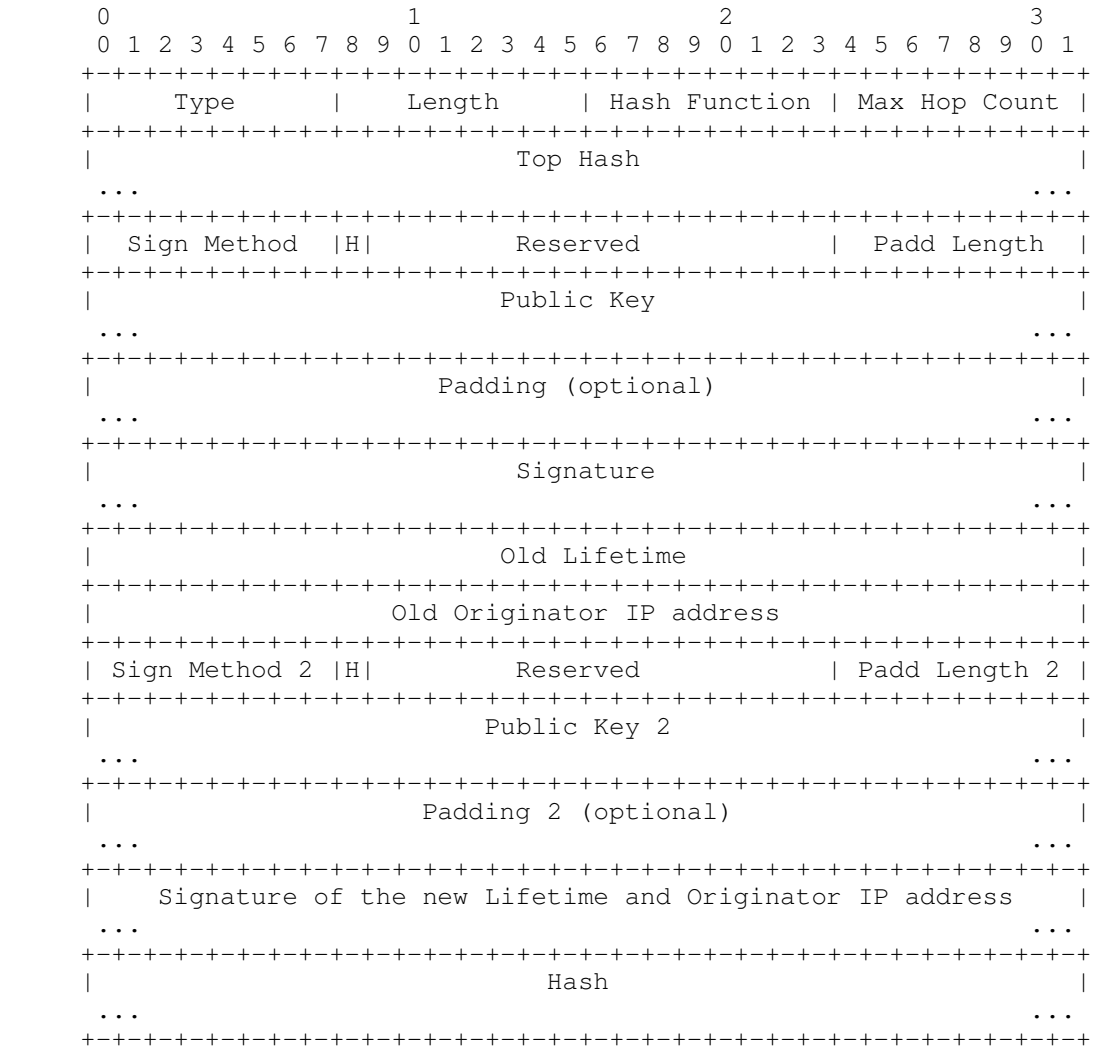
Signature Method ... Padding
The same than in RREQ (Single) Signature Extension.

Signature for RREP
The signature that should be put into the Signature field of the RREP Double Signature Extension when an intermediate node (that has previously received this RREQ and created a reverse route) wants to generate a RREP for a route to the source of this RREQ. This field has variable length, but it must be 32-bits aligned.

Signature The signature of the all the fields in the AODV packet that are before this field but the Hop Count field. This field has variable length, but it must be 32-bits aligned. Both signatures are generated by the requesting node.

Hash The hash corresponding to the actual hop count. This field has variable length, but it must be 32-bits aligned.

8. RREP Double Signature Extension



Type 67

Length The length of the type-specific data, not including the Type and Length fields of the extension in bytes.

Hash Function
The hash function used to compute the Hash and Top Hash fields.

Max Hop Count
: The Maximum Hop Count supported by the hop count authentication.

Top Hash
: The top hash for the hop count authentication. This field has variable length, but it must be 32-bits aligned.

Signature Method ... Padding
: The same than in RREQ (Single) Signature Extension.

Signature
: The signature of all the fields of the AODV packet that are before this field but the Hop Count field, and with the Old Lifetime value instead of the Lifetime. This signature is the one that was generated by the originator of the RREQ-DSE). This field has variable length, but it must be 32-bits aligned.

Old Lifetime
: The lifetime that was in the RREP generated by the originator of the RREQ-DSE).

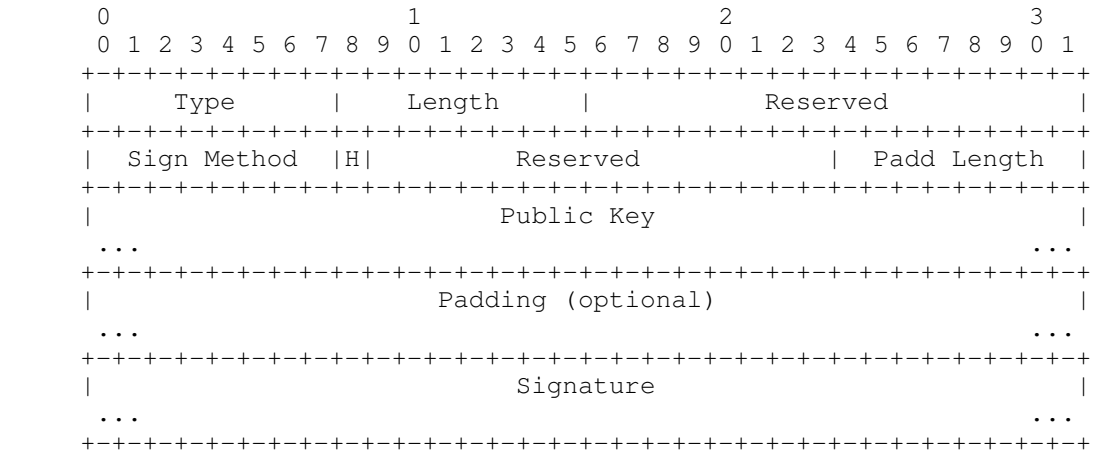
Old Originator IP address
: The Originator IP address that was in the RREP generated by the originator of the RREQ-DSE).

Signature Method 2 ... Padding 2
: The whole block of fields is repeated. This time for the 'Signature of the New Lifetime and Originator IP address' signature.

Signature of the new Lifetime and Originator IP address
: The signature of the RREP with the actual lifetime (the lifetime of the route in the intermediate node) and with the actual Originator IP address. This signature is generated by the intermediate node. This field has variable length, but it must be 32-bits aligned.

Hash
: The hash corresponding to the actual hop count. This field has variable length, but it must be 32-bits aligned.

9. RERR Signature Extension



Type 68

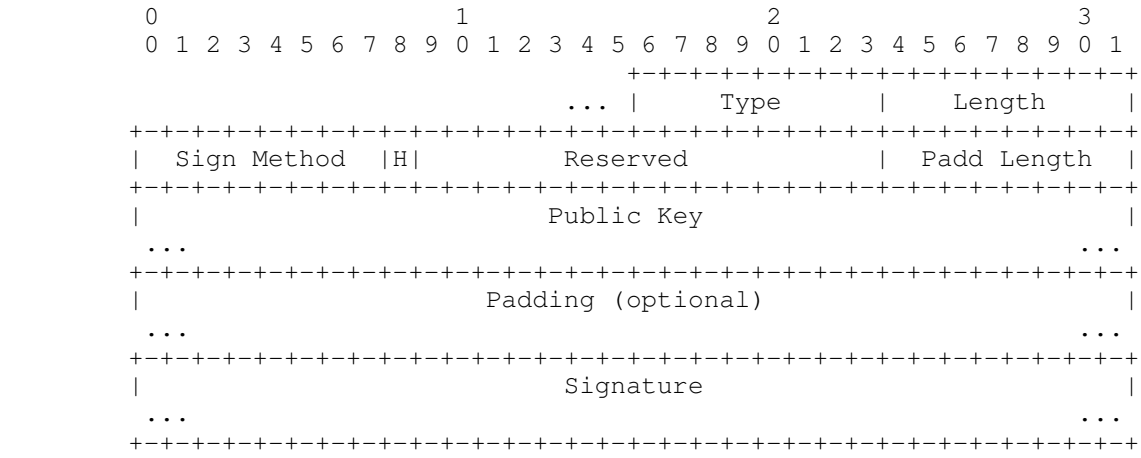
Length The length of the type-specific data, not including the Type and Length fields of the extension in bytes.

Reserved Sent as 0; ignored on reception.

Signature Method ... Padding
The same than in RREQ (Single) Signature Extension.

Signature The signature of the all the fields in the AODV packet that are before this field. This field has variable length, but it must be 32-bits aligned.

10. RREP-ACK Signature Extension



Type 69

Length The length of the type-specific data, not including the Type and Length fields of the extension in bytes.

Signature Method ... Padding
The same than in RREQ (Single) Signature Extension.

Signature The signature of the all the fields in the AODV packet that are before this field. This field has variable length, but it must be 32-bits aligned.

11. SAODV Operation

This section describes how SAODV allows to authenticate the AODV routing data. Two mechanisms are used to achieve this: hash chains and signatures.

11.1. SAODV Signatures

When calculating signatures, Hop Count field is always zeroed, because it is a mutable field. In the case of the Signature for RREP field of the RREQ Double Signature Extension, what is signed is the future RREP message that nodes might send back in response to the RREQ. To construct this message it uses the values of the RREQ and the Prefix Size (the RREP field that is not derivable from the RREQ but not zeroed when computing the signature.

In the case of RREPs, R and A flags are also zeroed. SAODV is not

designed taking into account AODV multicast ('R' flag is used in multicast) and 'A' flag is mutable and, if an attacker alters it, it can only lead to some sort of denial of service.

Every time a node generates a RREQ it decides if it should be signed with a Single Signature Extension or with a Double Signature Extension. All implementations MUST support RREQ Single Signature Extension, and SHOULD support RREQ Double Signature Extension. A node that generates a RREQ with the gratuitous RREP flag set SHOULD sign the RREQ with a Double Signature Extension. A node SHOULD never generate a RREQ without adding a Signature Extension.

When a node receives a RREQ, first verify the signature before creating or updating a reverse route to that host. Only if the signature is verified, it will store the route. If the RREQ was received with a Double Signature Extension, then the node will also store the signature, the lifetime and the Destination IP address for the RREP in the route entry.

If a node receives a RREQ without a Signature Extension it SHOULD drop it.

An intermediate node will reply a RREQ with a RREP only if fulfills the AODV requirements to do so, and the node has the corresponding signature and the old lifetime and old originator IP address to put into the 'Signature', 'Old Lifetime' and 'Old Originator IP address' fields of the RREP Double Signature Extension. Otherwise, it will rebroadcast the RREQ.

When a RREQ is received by the destination itself, it will reply with a RREP only if fulfills the AODV requirements to do so. This RREP will be sent with a RREP Single Signature Extension.

All implementations MUST support RREP Single Signature Extension, and SHOULD support RREP Double Signature Extension. A node SHOULD never generate a RREP without adding a Signature Extension. This also applies to gratuitous RREPs.

When a node receives a RREP, first verifies the signature before creating or updating a route to that host. Only if the signature is verified, it will store the route with the signature and the lifetime and the originator IP address of the RREP. If a node receives a RREP without a Signature Extension it SHOULD drop it.

Every node, generating or forwarding a RERR message, uses digital signatures to sign the whole message and any neighbor that receives verifies the signature. In this way it can verify that the sender of the RERR message is really the one that claims to be. And, since

destination sequence numbers are not singed by the corresponding node, a node SHOULD never update any destination sequence number of its routing table based on a RRER message.

Although nodes will not trust destination sequence numbers in a RERR message, they will use them to decide whether they should invalidate a route or not.

RREP-ACK messages MAY be authentified by using the RREP-ACK Signature Extension.

The block 'Signature Method ... Padding' is included before the 'Signature' field in all the extension messages, and before the 'Signature of the new Lifetime and Originator IP address' field in the RREQ-DSE message.

SAKM specifies the list of possible values of the Signature Method field and how public keys and signatures are encoded en the extension messages.

11.2. SAODV Hash Chains

Hash chains are used in SAODV to authenticate the hop count of the AODV routing messages (not only by the end points, but by any node that receives one of those messages).

Every time a node wants to send a RREQ or a RREP it generates a random number (seed). Selects a Maximum Hop Count. Maximum Hop Count SHOULD be set to the TTL value in the IP header, and it SHOULD never exceed its configuration parameter NET_DIAMETER. The Hash field in the Signature Extension is set to the seed. The Top Hash field is set to the seed hashed Max Hop Count times.

Every time a node receives a RREQ or a RREP it verifies the hop count by hashing Max Hop Count - Hop Count times the Hash field, and checking that the resultant value is the same than the Top Hash. If the check fails, the node SHOULD drop the packet.

Before rebroadcasting a RREQ or forwarding a RREP, a node hashes one time the Hash field in the Signature Extension.

The function used to compute the hash is set in the Hash Function field. Since this field is signed, a forwarding node will only be able to use the same hash function that the originator of the routing message has selected. If an node cannot verify or forward a routing message because it does not support the hash function that has been used, then it drops the packet.

The list of possible values of the Hash Function field are the same as the one for the hash functions used for the signature ('Hash F Sign') that are specified in SAKM.

12. Adaptations to AODV that are needed

According to the AODV RFC, the originator of a RREQ can put (on purpose) a much more bigger destination sequence number than the real one. This allows a very easy attack that consists in setting the destination sequence number to 0xFFFFFFFF (the maximum value that fits in the 32-bits field). Then, the originator of the RREP and all the intermediate nodes will have that as sequence number for the route. The next time the node increments the sequence number, its sequence number counter will overflow. This might cause completely unexpected results, none of them good.

The fact that the originator of the RREQ can set the sequence number of the destination is because it is going to be needed if the destination node has rebooted (see section 6.13. 'Actions After Reboot' in the AODV RFC). After rebooting, a node does not remember its sequence number anymore and trusts anybody that sends to it a RREQ with the number. But this just cannot be allowed.

Therefore, all the AODV-enabled nodes SHOULD have a way to keep their destination sequence number even after rebooting. In addition, in the case that the destination sequence number in the RREQ is bigger than the destination sequence number of the destination node, the destination node SHOULD NOT take into account the value in the RREQ. Instead, it will realize that the originator of the RREQ is misbehaving and will send the RREP with the right sequence number.

Finally, and concerning to the AODV port (the UDP port used to send AODV messages), AODV nodes SHOULD never accept AODV messages sent from a different port than the standard one.

13. Security Considerations

The goal of the protocol extension described here, is to achieve that a node that plans to build an attack by not behaving according to the AODV routing protocol, will be only able to selectively don't reply to certain routing messages and to lie about information about itself. Nevertheless, It does not do much to avoid denial-of-service attacks.

If a malicious node receives a packet and resends it after a while, it will not alter the network topology because of the sequence number system.

It might seem that lifetime is not very strongly authenticated in the case that intermediate nodes are allowed to reply RREQs, because they could lie about the lifetime. Anyway, the goal of the protocol extension is achieved, because the node would be only lying about itself.

What about the originator IP address (also in the case that intermediate nodes are allowed to reply RREQs)? If an intermediate node lies about it, the RREP will travel to the fake originator IP address but the routes that will be generated by the nodes that will propagate the routing message will be correct. So the attack is practically equivalent to the one in which the intermediate node ignores the RREQ.

Using hash chains for authentifying hop counter has a problem: A malicious node forwarding a route might not increment the hop counter by using the same hash value. If it does so, the subsequent nodes will think that this route is one hop shorter (having more chances to be chosen as the route to use). This is not really a big threat, because to launch an attack, a group of malicious nodes should be close to the shortest path (each of the malicious nodes forwarding the routing messages would not increment the hop counter), and the less malicious nodes are, the more close they have to be to the shortest path. A path that is changing with the time.

14. Modifications of the draft

Version 5

- The intro has been changed.

- RERR cannot use delayed verification.

- The key management part has been moved to draft-guerrero-manet-sakm-00.txt. And now is called 'Simple Ad hoc Key Management (SAKM)'.

Version 4

- 'A' flag is not signed (as proposed by Francesco Dolcini). Neither is 'R' flag.

- Section 14.4. SAODV Key Management: IPv4 addresses can now be generated in a similar fashion than IPv6 ones.

- Section 7. RREQ Double Signature Extension: Prefix Size is now 7 bit long to be able to hold IPv6 Prefix Sizes.

Version 3

- Clarification: Now, in section '3. Overview', it explicitly says that Hello Interval extension is not signed.

- Adds sections: '14.1.1. Encoding of Public Key and Signature', '14.1.2. Signature Method #1 (RSA)', '14.1.3. Signature Method #2 (DSA)' and '14.1.4. Signature Method #3 (ElGamal)'.

- Clarification: Now all lengths specify if we are talking about bytes or 32-bit words.

- In section '14.4. SAODV Key Management', adds the list of what is used as PublicKey depending on which Signature Method is use.

- In section '14.2. SAODV Hash Chains', the list of hash functions has changed, and now includes more hash functions. Note that the hash functions that already existed in the previous version now have a different value.

Version 2

- Correction: In section '14.1. SAODV Signatures' instead of "and the lifetime (that is REV_ROUTE_LIFE) and the Originator IP address for the RREP in the route entry" now it says "the lifetime and the Destination IP address for the RREP in the route entry.". Thanks to Moritz Killat.

- Adds a bit more of explanation of what a node has to do if it wants to be able to reply as an intermediate node for a route that has been added due to a RREQ or to a RREP in the section '3. Overview'.

- Correction: When an intermediate node generates a RREP, the 'Originator IP Address' of the AODV message with a RREP-DSE might be different than the one that was in the RREQ with a RREQ-DSE (so we have to add a field in the RREP-DSE for the old Originator IP Address just in the same way as we do with the lifetime). Thanks to Moritz Killat for noticing it.

- Correction: In RREQ-DSE 'Signature' should also sign the 'Signature for RREP' and, to make things clear the 'Signature for RREP' field goes before the 'Signature' field. I noticed this when discussing the DSE mechanism with Moritz Killat.

- Correction: Hash functions must be MD5 and SHA1 (not HMACs). Thanks to Varaporn Pangboonyanon for noticing it.

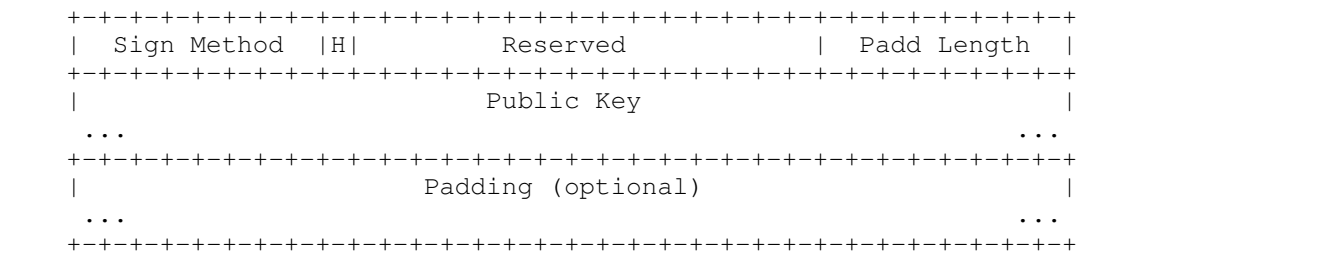
- Correction: In the HMACs used to get the SAODV_HID and the SAODV_FID, the data to which the HMACs are going to be applied was missing (now it is PublicKey). So it is an HMAC of the public key

with the public key as a key.

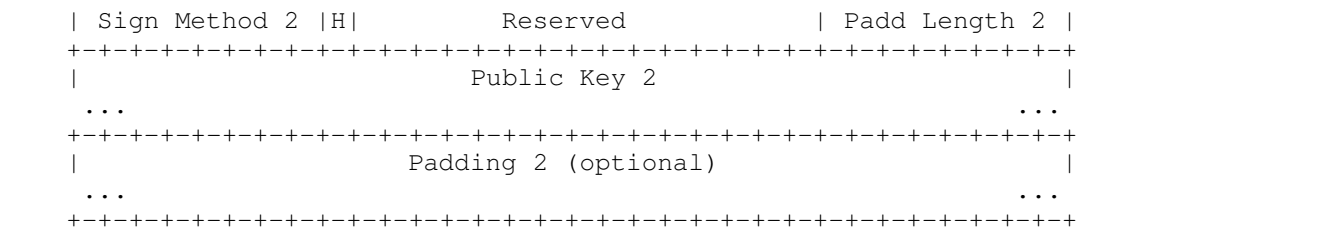
Version 1

- Adds this section. ;)

- Adds the following fields just before the 'Signature' field in all the extension messages:



- And adds these other fields just before the 'Signature of the new Lifetime' field in the RREQ-DSE extension message:



- Adds the section "11. Duplicated Address (DADD) Detected Message".

- Adds the section "12. New Address (NADD) Notification Message".

- Adds the section "13. New Address Acknowledgment (NADD-ACK) Message".

- Adds some text at the end of the section "14.1. SAODV Signatures" to explain the new fields of the extension messages.

- Adds the section "14.3. SAODV Delayed Verification of Signatures".

- Adds the section "14.4. SAODV Key Management".

- Removes the section "2.3. Key distribution".

- Other stuff I might be forgetting.

15. Acknowledgments

I want to thank all the people from the Nokia Research Center in Helsinki (where I worked for five years) that helped to make SAODV a reality. Special mention deserve my colleague N. Asokan and my bosses Jari Juopperi and Asko Vilavaara.

N. Asokan (from Nokia Research Center) has contributed to this draft with several improvements and corrections. He suggested the use of hash chains for authenticate the hop count and that intermediate nodes should sign the lifetime of the RREPs.

I also want to thank the following persons for their help and improvements to the draft: Sampo Sovio (from Nokia Research Center), Toni Barrera Arboix (while he was working for Nokia Research Center), Varaporn Pangboonyanon, Moritz Killat (from NEC Europe Ltd.) and Francesco Dolcini.

References

[1] Charles E. Perkins, Elizabeth M. Belding Royer, Samir R. Das: Ad hoc On-Demand Distance Vector (AODV) Routing. RFC 3561, November 2003.

[2] M. Guerrero Zapata: Simple Ad hoc Key Management (SAKM). draft-guerrero-manet-sakm-00.txt, February 2006.

[3] S. Bradner: Key words for use in RFCs to Indicate Requirement Levels. RFC 2119, March 1997.

[4] S. Kent, R. Atkinson: Security Architecture for the Internet Protocol. RFC 2402, November 1998.

Author's Address:

Questions about this memo can be directed to the author:

Manel Guerrero Zapata
Computer Architecture Department (DAC)
Technical University of Catalonia (UPC)
UPC-AC C6-123 Campus Nord
C. Jordi Girona 1-3
08034 Barcelona SPAIN
(+34) 93 4054044
guerrero@ac.upc.edu

Appendix A. Full Copyright Statement

Mobile Ad Hoc Networking Working Group
INTERNET DRAFT
2 February 2005

Manel Guerrero Zapata
Technical University
of Catalonia (UPC)

Simple Ad hoc Key Management (SAKM)
draft-guerrero-manet-sakm-00.txt

Intellectual Property Rights Statement

Status of this Memo

Copyright

Abstract

The Simple Ad hoc Key Management (SAKM) is a key management system that allows to the nodes of an ad hoc network to use asymmetric cryptography with zero configuration. It is intended to be applied to MANET routing protocols that provide security features that require the use of asymmetric cryptography (like SAODV and SDYMO).

Table of Contents

1. Introduction

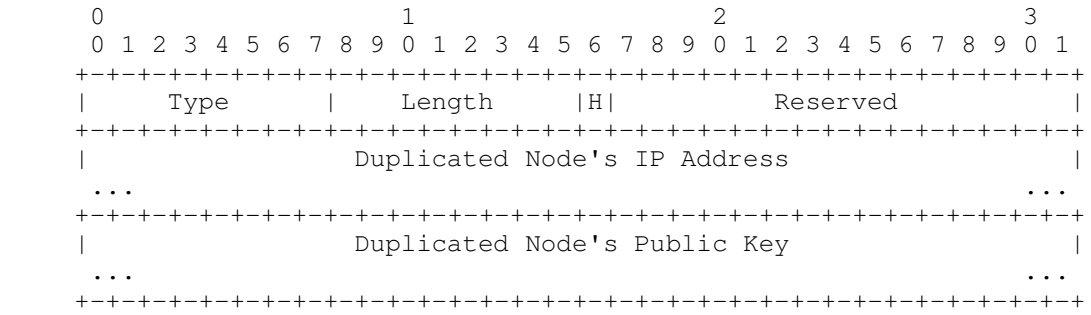
The Simple Ad hoc Key Management (SAKM) is a key management system that allows to the nodes of an ad hoc network to use asymmetric cryptography with zero configuration. It is intended to be applied to MANET routing protocols that provide security features that require the use of asymmetric cryptography (like SAODV[1] and SDYMO[2]). SAKM messages will be sent through the same port as the routing protocol (be it SAODV, SDYMO, or some other).

SAKM protects the non-mutable fields of the routing messages. It is assumed that mutable fields (like hop count) are protected by some other means.

2. Terminology

This memo uses the conventional meanings [3] for the capitalized words MUST, SHOULD and MAY. It also uses terminology taken from the specification of IPSec [4].

3. Duplicated Address (DADD) Detected Message



Type 64

Length The length of the type-specific data, not including the Type and Length fields of the message in bytes.

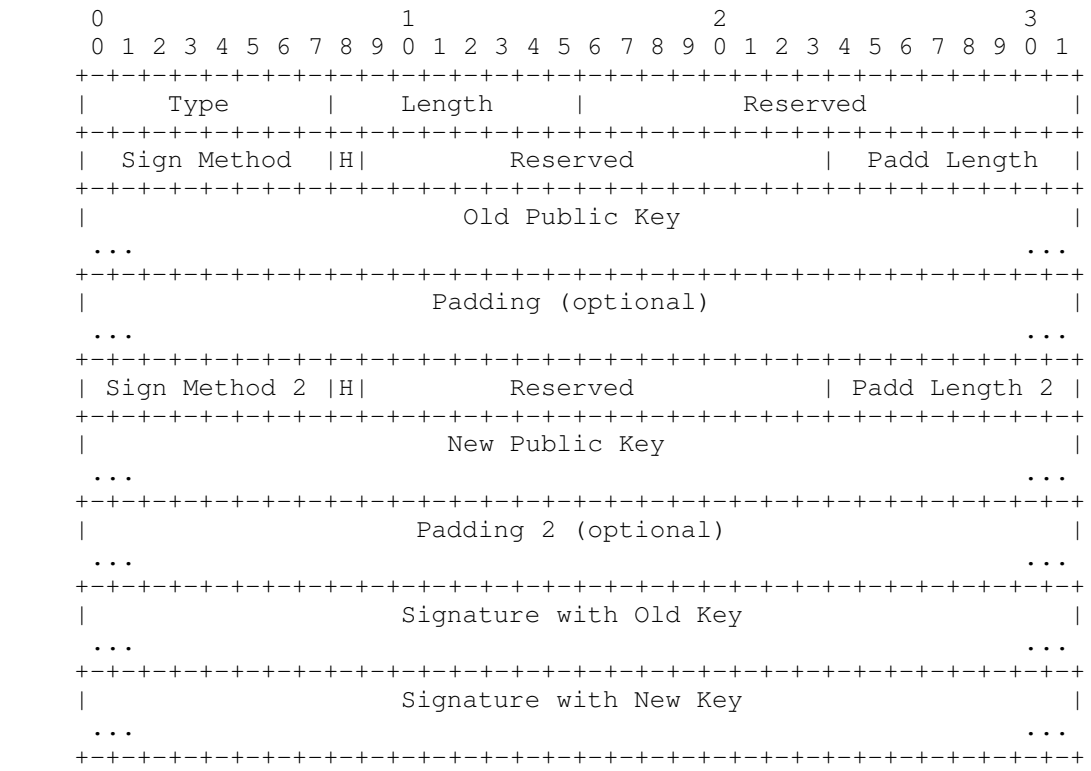
H Half Identifier flag. If it is set to '1' indicates the use of HID and if it is set to '0' the use of FID.

Reserved Sent as 0; ignored on reception.

Duplicated Node's IP Address
The IP Address of the node that uses a Duplicated IP Address.

Duplicated Node's Public Key
The Public Key of the node that uses a Duplicated IP Address.

4. New Address (NADD) Notification Message



Type 65

Length The length of the type-specific data, not including the Type and Length fields of the message in bytes.

Reserved Sent as 0; ignored on reception.

Signature Method ... Padding
The same than in RREQ (Single) Signature Extension. Corresponds to the 'Signature with Old Public Key' signature.

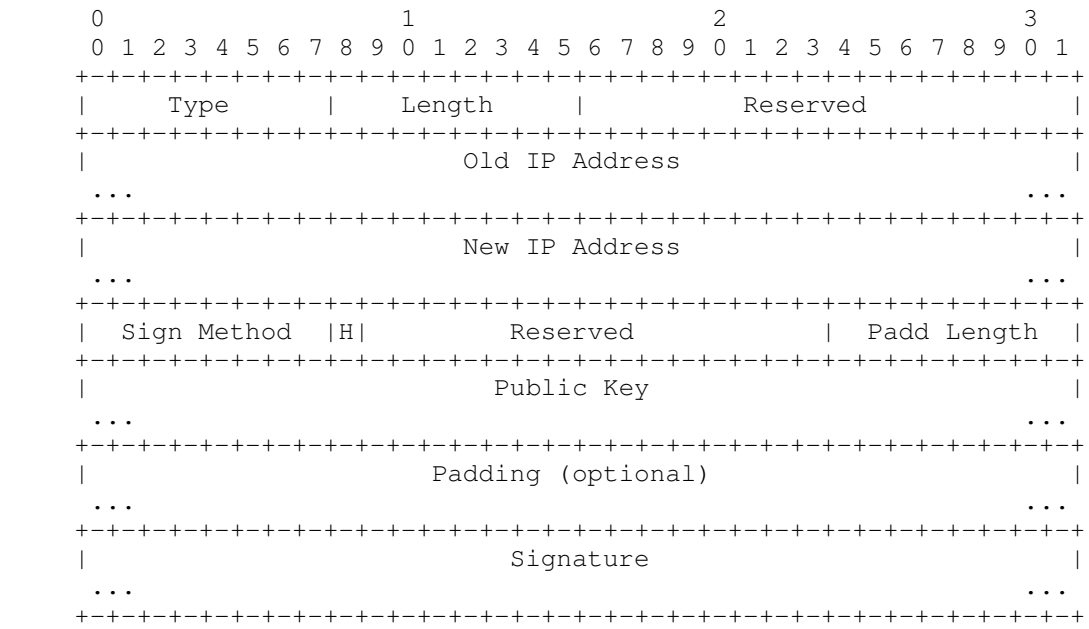
Signature Method 2 ... Padding 2
The whole block of fields is repeated. Corresponds to

the 'Signature of the New Public Key' signature.

Signature with Old Key
The signature (with the old key) of the all the fields in the routing message that are before this field. This field has variable length, but it must be 32-bits aligned.

Signature with New Key
The signature (with the new key) of the all the fields in the routing message that are before this field. This field has variable length, but it must be 32-bits aligned.

5. New Address Acknowledgment (NADD-ACK) Message



Type 66

Length The length of the type-specific data, not including the Type and Length fields of the message in bytes.

Reserved Sent as 0; ignored on reception.

Old IP Address
The old IP address.

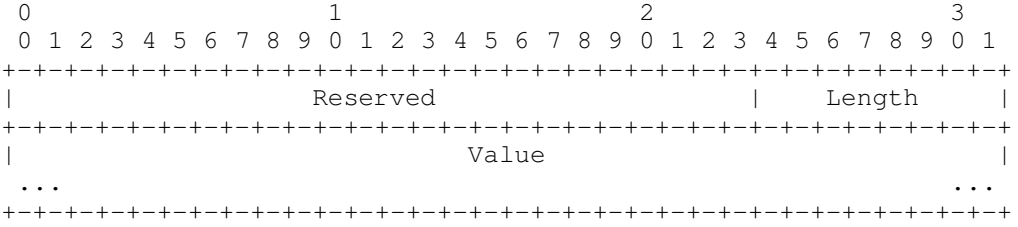
New IP Address
The new IP address.

Signature Method ... Padding
The same than in RREQ (Single) Signature Extension.

Signature The signature of the all the fields in the routing message that are before this field. This field has variable length, but it must be 32-bits aligned.

6. Encoding of Public Key and Signature

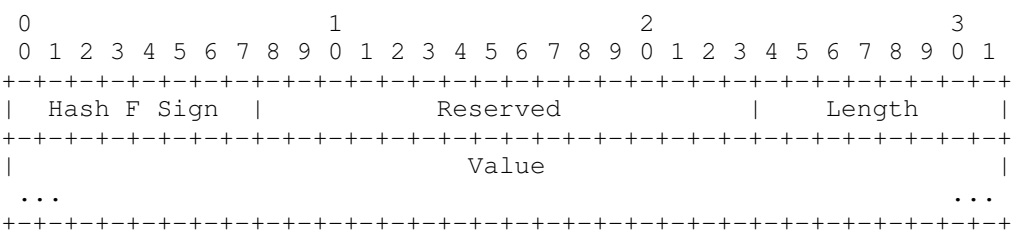
Encoding of each of the components of Public Key will be done in the following manner unless stated otherwise:



Reserved Sent as 0; ignored on reception.

Length The length of the Value field, (not including the Length and Reserved fields) in 32-bit units.

Encoding of the Signature will be done in the following manner unless stated otherwise:



Hash F Sign The hash function used to compute the hash that will be signed. Because, typically you don't want to sign the whole message, you sign a hash of the message.

The other fields work just like the ones of the encoding of the

components of Public Key.

This is the list of possible values of the 'Hash F Sign' field:

Hash F Sign	Hash length	Value
RESERVED	-	0
MD2	(128 bit)	1
MD5	(128 bit)	2
SHA1	(160 bit)	3
SHA256	(256 bit)	4
SHA384	(384 bit)	5
SHA512	(512 bit)	6
Reserved	-	7-127
Implementation dependent	-	128-255

All the implementations MUST support the SHA1 option.

MD2 is a relatively slow hash function, but I decided to include it anyway. About SHA512 and SHA384, somebody might argue that nowadays they generate a much longer hash that what it is needed. But I believe they will be needed in the future.

7. Signature Methods

This is the list of possible values of the Signature Method field that MAY be included in the routing message (otherwise it is assumed to be RSA):

RESERVED	0
RSA	1
DSA	2
ElGamal	3
Reserved	4-127
Implementation dependent	128-255

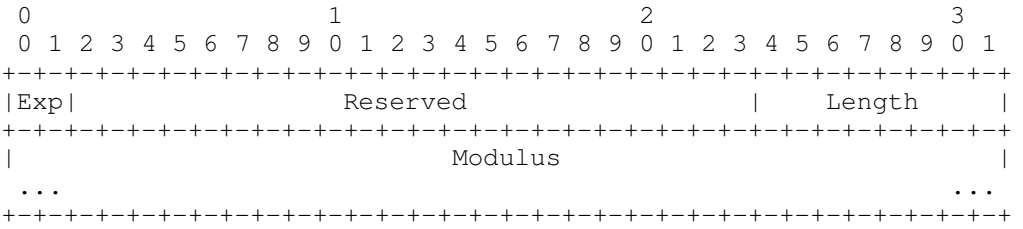
All the implementations MUST support the RSA option.

7.1. Signature Method #1 (RSA)

Public Key is composed of:
- Modulus (n)
- Exponent (e)

Signature is composed of:
- Signature

Where all these components may be encoded in the standard way or in the following way:



Reserved Sent as 0; ignored on reception.

Length The length of the Modulus field, (not including the Length and Reserved fields) in 32-bit units.

Exp The Exponent (e):

00	The components are encoded in the standard way. The Exponent (e) will be specified after the Modulus (n).
01	Specifies that Exponent (e) is 65537 (2^16+1).
10	Specifies that Exponent (e) is 17 (2^4+1).
11	Specifies that Exponent (e) is 3.

A message that uses any of these 'smartly chosen' exponents MUST include random padding (in the Padding field). There is no security problem with everybody using the same exponent.

7.2. Signature Method #2 (DSA)

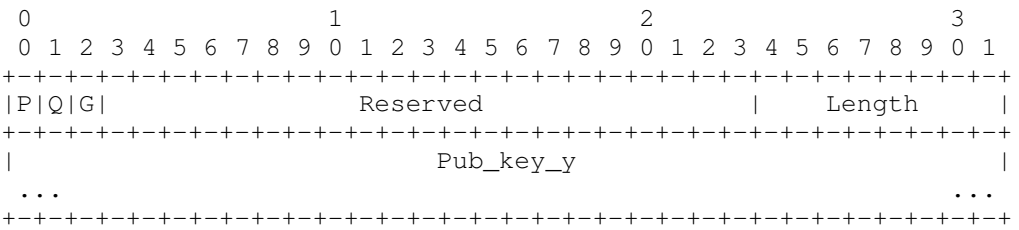
Public Key is composed of:

- Pub_key_y (y = g^x mod p)
- Prime (p)
- Group_order (q)
- Group_generator (g)

Signature is composed of:

- Signature

Where all these components may be encoded in the standard way or in the following way:



Reserved Sent as 0; ignored on reception.

Length The length of the Modulus field, (not including the Length and Reserved fields) in 32-bit units.

P Shared Prime (p) flag. If it is set to '1' indicates that Prime (p) is shared among the nodes of the network.

Q Shared Group_order (q) flag.

G Shared Group_generator (g) flag.

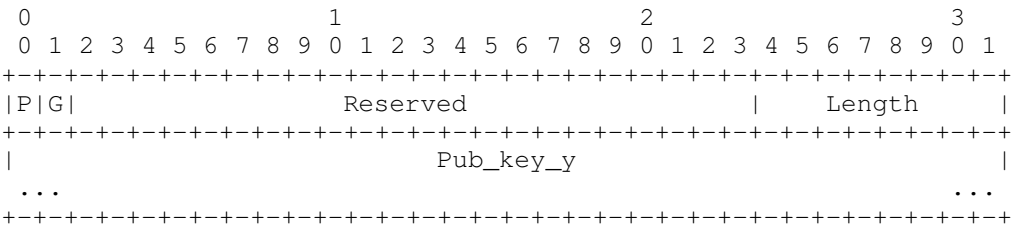
After this block, the non shared values will be included in the usual order.

7.3. Signature Method #3 (ElGamal)

Public Key is composed of:
- Pub_key_y (y = g^x mod p)
- Prime (p)
- Group_generator (g)

Signature is composed of:
- Signature

Where all these components may be encoded in the standard way or in the following way:



Reserved Sent as 0; ignored on reception.

Length The length of the Modulus field, (not including the Length and Reserved fields) in 32-bit units.

P Shared Prime (p) flag. If it is set to '1' indicates that Prime (p) is shared among the nodes of the network.

G Shared Group_generator (g) flag.

After this block, the non shared values will be included in the usual order.

8. Delayed Verification of Signatures

The signatures in route requests and route replies will be verified after the node has forwarded the route reply. In this way transmissions of the route requests and replies occur without any kind of delay due to the verification of the signatures.

Routes pending of verification will not be used to forward any packet. If a packet arrives for a node for which there is a route pending of verification. The node will have to verify it before using that route. If the verification fails, it will delete the route and request a new one.

9. IP address generation

The first part of this section describes the key management scheme to be used with IPv6.

SAKM generates the IP addresses is very similar to the generation of SUCV (Statistically Unique and Cryptographically Verifiable) addresses [5]. SUCV addresses where designed to protect Binding Updates in Mobile IPv6. The main difference between SUCV and the method proposed in here is that SUCV addresses are generated by hashing an "imprint" in addition to the public key. That imprint (that can be a random value) is used to limit certain attacks related to Mobile IP.

In SAKM, the address can be a network prefix of 64 bits with a 64 bit SAKM_HID (Half IDentifier) or a 128 bit SAKM_FID (Identifier). These two identifiers are generated almost in the same way than the sucvHID and the sucvID in SUCV (with the difference that they hash the public key instead of an imprint):

SAKM_HID = SHA1HMAC_64(PublicKey, PublicKey)

SAKM_FID = SHA1HMAC_128(PublicKey, PublicKey)

This is the list of what is used as PublicKey depending on which Signature Method is used:

Signature Method	PublicKey
1 (RSA)	Modulus (n)
2 (DSA)	Pub_key_y (y = g^x mod p)
3 (ElGamal)	Pub_key_y (y = g^x mod p)

There MAY be a flag in the routing message extensions (the 'H' flag) that will be set to '1' if the IP address is a HID and to '0' if it is a FID. Otherwise it the underlying protocol MUST specify which of them uses.

Finally, if it has to be a real IPv6 address, there is a couple of things that should be done [6].

If HID is used, then the HID behaves as an interface identifier and, therefore, its sixth bit (the universal/local bit) should be set to zero (0) to indicate local scope (because the IP address is not guaranteed to be globally unique).

And, if FID is used, then a format prefix corresponding to the MANET network should be overwritten to the FID. Format prefixes '010' through '110' are unassigned and would take only three bits of the FID. Format prefixes '1110' through '1111 1110 0' are also unassigned and they would take between 4 and 9 bits of the FID. All of these format prefixes required to have to have 64-bit interface identifiers in EUI-64 format, so universal/local bit should be set to zero (0).

The length of an IPv4 address is probably too short to provide the statistically uniqueness that this scheme requires when the number of nodes is very big. Nevertheless, if the number of nodes is assumed to be low, (let's say, under 100 nodes) it is not very unrealistic to expect that the statistically uniqueness property will hold.

The SAKM IPv4 address will have a network prefix of 8 bits and a SAKM_4ID (IPv4 Identifier). The network prefix can be any number between 1 and 126 (both included) with the exception of 14, 24 and 39 (see RFC3330). The network prefix 10 can only be used if it is granted that it will not be connected to any other network (RFC1918).

The SAKM_4ID will be the first bits of the SAKM_HID and the 'H' flag will be set.

9.1. Duplicated IP Address Detection

If a node 'A' receives a routing message that is signed by a node 'B' that has the same IP address than one of the nodes for which 'A' has a route entry (node 'C'), it will not process normally that routing message. Instead, it will inform 'B' (sending to it a Duplicated Address (DADD) Detected message) that it is using a duplicated IP and it will prove it by adding the public key of 'C' (so 'B' can verify the truthfulness of the claim).

When the node 'B' receives a DADD message that indicates that somebody else has the same IP address than itself (or it realizes about it by itself), it will have to generate a new pair of public/private keys. After that, it will derive its IP address from its public key and it MIGHT inform to all the nodes it finds relevant (through a broadcast) of which is its new IP address with an special message (New Address (NADD) Notification message) that contains: the two IP addresses (the old and the new ones) and the two public signatures (old and new) signed with the old private key and, all this, signed with the new private key. This unicast MIGHT be answered with the New Address Acknowledgment (NADD-ACK) Message by the receiver if it verifies that everything is in order.

After this, the node will generate a route error message for his old IP address. Its propagation will delete the route entries for the old IP address and, therefore, eliminate the duplicated addresses. This route error message may have a message extension that tells which is the new address. In this way, the nodes that receive the routing message can already create the route to the new IP address.

10. Security Considerations

Although it is true that there is no way to preclude a node of inventing many identities, that cannot be used to create an attack against the routing algorithm.

Delayed verification makes possible that a malicious node creates invalid route requests that could flood the network. But, the same malicious node can flood the network with perfectly valid route requests. And there would be no easy way to know if it is trying to flood the network or if it is just trying to see if any of its friend nodes are present in the network (for instance).

An attacker cannot forge a public/private key pair from an IP address so the identity token becomes the IP address itself.

11. Modifications of the draft

Version 0

- This draft describes the key management system that was contained in the SAODV draft till its version 04.

12. Acknowledgments

I want to thank everybody who contributed to SAODV, since SAKM was originally part of it.

References

[1] M. Guerrero Zapata: Secure Ad hoc On-Demand Distance Vector (SAODV) Routing. draft-guerrero-manet-saodv-05.txt, February 2006.

[2] M. Guerrero Zapata: Secure Dynamic MANET On-Demand (SDYMO) Routing Protocol. draft-guerrero-manet-sdymo-00.txt, February 2006.

[3] S. Bradner: Key words for use in RFCs to Indicate Requirement Levels. RFC 2119, March 1997.

[4] S. Kent, R. Atkinson: Security Architecture for the Internet Protocol. RFC 2402, November 1998.

[5] Gabriel Montenegro, Claude Castelluccia: Statistically Unique and Cryptographically Verifiable (SUCV) Identifiers and Addresses. Network and Distributed System Security Symposium (NDSS '02). February 2002,

[6] R. Hinden and S. Deering: IP Version 6 Addressing Architecture. RFC 2373, July 1998.

Author's Address:

Questions about this memo can be directed to the author:

Manel Guerrero Zapata
Computer Architecture Department (DAC)
Technical University of Catalonia (UPC)
UPC-AC C6-123 Campus Nord
C. Jordi Girona 1-3
08034 Barcelona SPAIN
(+34) 93 4054044
guerrero@ac.upc.edu

Appendix A. Full Copyright Statement

Mobile Ad Hoc Networking Working Group
INTERNET DRAFT
2 February 2005

Manel Guerrero Zapata
Technical University
of Catalonia (UPC)

Secure Dynamic MANET On-Demand (SDYMO) Routing Protocol
draft-guerrero-manet-sdymo-00.txt

Intellectual Property Rights Statement

Status of this Memo

Copyright

Abstract

The Secure Dynamic MANET On-Demand (SDYMO) Routing Protocol is an extension of the DYMO routing protocol that can be used to protect the route discovery mechanism providing security features like integrity and authentication.

Table of Contents

1. Introduction

SDYMO is an extension of the DYMO[1] routing protocol that protects the route discovery mechanism providing security features like integrity and authentication. It uses digital signatures to authenticate the non-mutable fields of the messages, and hash chains to secure the hop count information contained in the Routing Block Hop Count (RBHopCnt).

The way SDYMO secures DYMO is very similar compared to the way SAODV[2] secures AODV[3]. The reader might find useful to read the existing drafts and papers about SAODV.

SDYMO can use the Simple Ad hoc Key Management (SAKM)[4] as a key management system.

2. Overview

The solution presented in this paper is an extension of the DYMO protocol mainly by using new extension messages. In these extension messages there is a signature of the DYMO packet with the private key of the original sender of the Routing message (not of the intermediate nodes that just forward it).

When RREQ is sent, the sender signs the message. Intermediate nodes verify the signature before creating or updating a reverse route to that host. And only if the signature is fine they store the reverse route. The final destination node signs the RREP with its private key. Intermediate and final nodes, again verify the signature before creating or updating a route to that host, also storing the signature with the route entry.

Every node, generating or forwarding a RERR message, uses digital signatures to sign the whole message and any neighbor that receives verifies the signature.

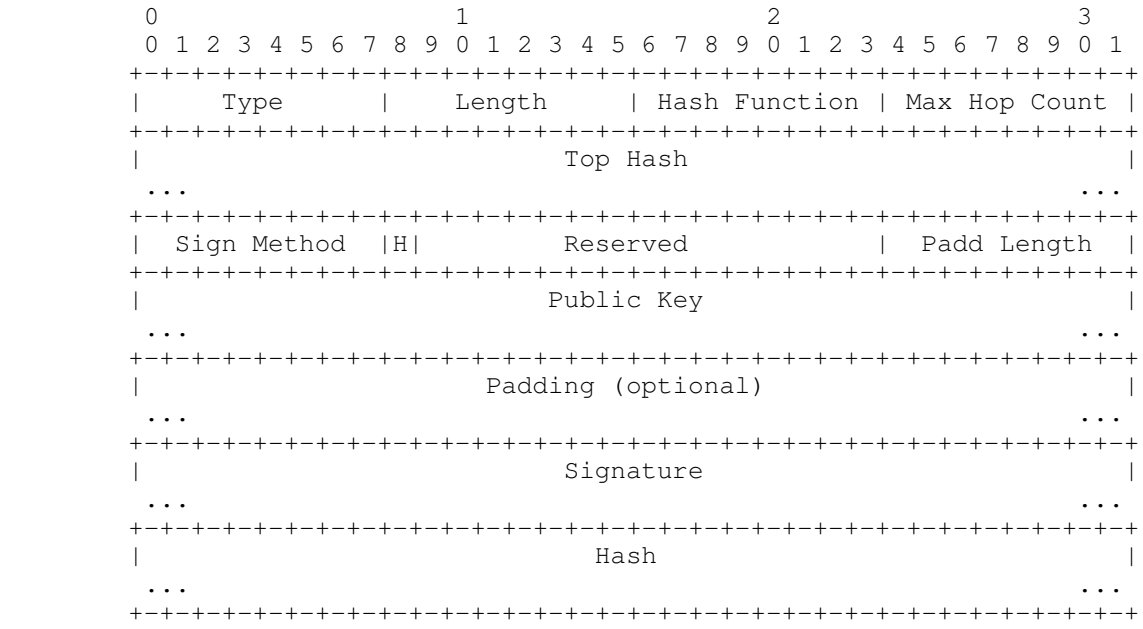
The hop counts are authentified by using a hash chain.

TTLs and 'I' flags are not signed.

3. Terminology

This memo uses the conventional meanings [5] for the capitalized words MUST, SHOULD and MAY. It also uses terminology taken from the DYMO specifications.

4. Routing Element (RE) Signature Extension



Type 64

Length The length of the type-specific data, not including the Type and Length fields of the extension in bytes.

Hash Function
The hash function used to compute the Hash and Top Hash fields.

Max Hop Count
The Maximum Hop Count supported by the hop count authentication.

Top Hash The top hash for the hop count authentication. This field has variable length, but it must be 32-bits aligned.

Signature Method
The signature method used to compute the signatures.

H Half Identifier flag. If it is set to '1' indicates the use of HID and if it is set to '0' the use of FID.

Reserved Sent as 0; ignored on reception.

Padding Length
Specifies the length of the padding field in 32-bit units. If the padding length field is set to zero, there will be no padding.

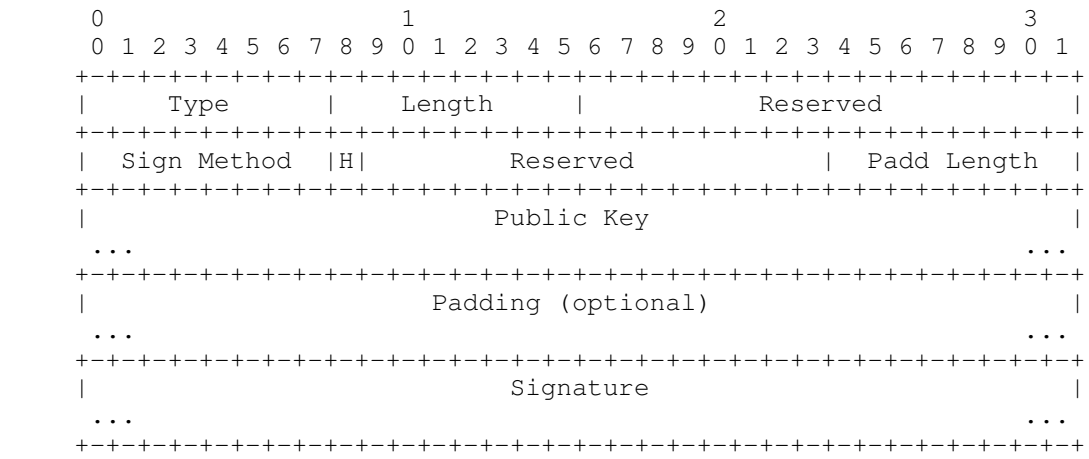
Public Key The public key of the originator of the message. This field has variable length, but it must be 32-bits aligned.

Padding Random padding. The size of this field is set in the Padding Length field.

Signature The signature of the all the fields in the DYMO message that are before this field but the Hop Count field. This field has variable length, but it must be 32-bits aligned.

Hash The hash corresponding to the actual hop count. This field has variable length, but it must be 32-bits aligned.

5. RERR Signature Extension



Type 65

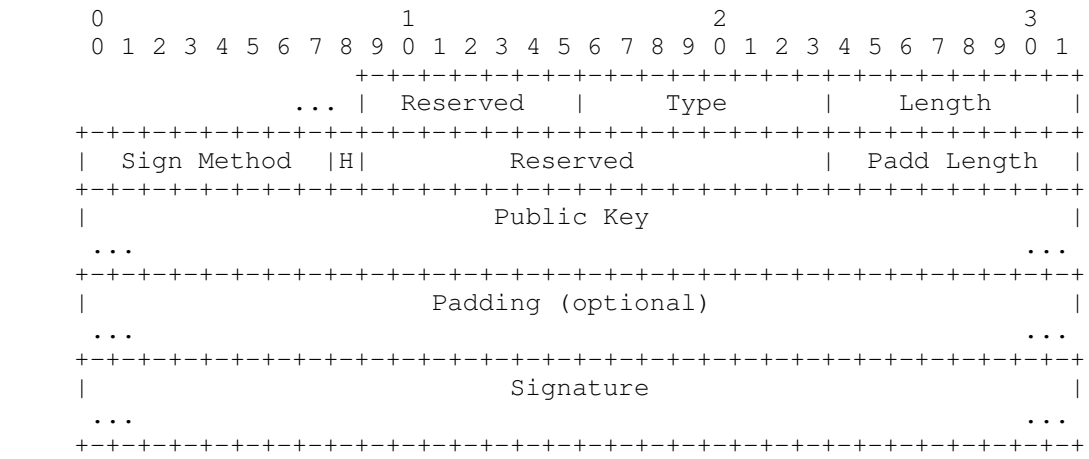
Length The length of the type-specific data, not including the Type and Length fields of the extension in bytes.

Reserved Sent as 0; ignored on reception.

Signature Method ... Padding
The same than in RBlock Signature Extension.

Signature The signature of the all the fields in the DYMO message that are before this field. This field has variable length, but it must be 32-bits aligned.

6. UERR Signature Extension



Type 66

Length The length of the type-specific data, not including the Type and Length fields of the extension in bytes.

Signature Method ... Padding
The same than in RBlock Signature Extension.

Signature The signature of the all the fields in the DYMO message that are before this field. This field has variable length, but it must be 32-bits aligned.

7. SDYMO Operation

This section describes how SDYMO allows to authenticate the DYMO routing data. Two mechanisms are used to achieve this: hash chains and signatures.

7.1. SDYMO Signatures

When calculating signatures, Hop Count field is always zeroed, because it is a mutable field.

When a node receives a RE, first verify the signature. Only if the signature is verified, it process the message.If a node receives a RE without a Signature Extension it SHOULD drop it.

Every node, generating or forwarding a RERR message, uses digital signatures to sign the whole message and any neighbor that receives verifies the signature. In this way it can verify that the sender of the RERR message is really the one that claims to be. And, since destination sequence numbers are not singed by the corresponding node, a node SHOULD never update any destination sequence number of its routing table based on a RRER message.

Although nodes will not trust destination sequence numbers in a RERR message, they will use them to decide whether they should invalidate a route or not.

UERR messages SHOULD be authentified by using the UERR Signature Extension.

SAKM specifies the list of possible values of the Signature Method field and how public keys and signatures are encoded en the extension messages.

7.2. SDYMO Hash Chains

Hash chains are used in SDYMO to authenticate the hop count of the RBlocks (not only by the end points, but by any node that receives one of those messages).

Every time a node wants to send a RREQ or a RREP it generates a random number (seed). Selects a Maximum Hop Count. Maximum Hop Count SHOULD be set to the TTL value in the IP header, and it SHOULD never exceed its configuration parameter NET_DIAMETER. The Hash field in the Signature Extension is set to the seed. The Top Hash field is set to the seed hashed Max Hop Count times.

Every time a node receives a RE it verifies the hop count by hashing Max Hop Count - Hop Count times the Hash field, and checking that the resultant value is the same than the Top Hash. If the check fails, the node SHOULD drop the packet.

Before forwarding a RE, a node hashes one time the Hash field in the Signature Extension.

The function used to compute the hash is set in the Hash Function field. Since this field is signed, a forwarding node will only be able to use the same hash function that the originator of the routing message has selected. If an node cannot verify or forward a routing

message because it does not support the hash function that has been used, then it drops the packet.

The list of possible values of the Hash Function field are the same as the one for the hash functions used for the signature ('Hash F Sign') that are specified in SAKM.

8. Adaptations to DYMO that are needed

Routing Elements (REs) MUST have only one Routing Block (RB).

DYMO does not let intermediate node to originate a RREP, which makes things easier for SDYMO.

9. Modifications of the draft

Version 1

Not yet.

10. Acknowledgments

I want to thank to thank everybody who contributed to SAODV, since SDYMO is based on it.

References

[1] I. Chakeres, E. Belding-Royer, C. Perkins: Dynamic MANET On-demand (DYMO) Routing. draft-ietf-manet-dymo-03, October 2005.

[2] M. Guerrero Zapata: Secure Ad hoc On-Demand Distance Vector (SAODV) Routing. draft-guerrero-manet-saodv-05.txt, February 2006.

[3] Charles E. Perkins, Elizabeth M. Belding Royer, Samir R. Das: Ad hoc On-Demand Distance Vector (AODV) Routing. RFC 3561, November 2003.

[4] M. Guerrero Zapata: Simple Ad hoc Key Management (SAKM). draft-guerrero-manet-sakm-00.txt, February 2006.

[5] S. Bradner: Key words for use in RFCs to Indicate Requirement Levels. RFC 2119, March 1997.

Author's Address:

Questions about this memo can be directed to the author:

Manel Guerrero Zapata
Computer Architecture Department (DAC)
Technical University of Catalonia (UPC)
UPC-AC C6-123 Campus Nord
C. Jordi Girona 1-3
08034 Barcelona SPAIN
(+34) 93 4054044
guerrero@ac.upc.edu

Appendix A. Full Copyright Statement