Chapter 4

Secure Ad hoc On-Demand Distance Vector

4.1 Introduction

Some aspects of ad hoc networks have interesting security problems [1, 52, 46]. 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 [40] and [43].

4.2 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 [40] nor the survey by Royer and Toh [43] 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 [37], 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 [24] attempts to secure the Border Gateway Protocol by using PKI (Public Key Infrastructure) and IPsec.

In their paper on securing ad hoc networks [52], Zhou and Haas primarily discuss key management. 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., [44, 18]). And, lately, some work has been done to secure ad hoc networks by using misbehavior detection schemes (e.g., [30]). 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. [6] 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 here 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 [34] proposed a protocol (SRP) that can be applied to several existing routing protocols (in particular DSR [23] and IERP [17]). 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 [18], [5] and [39] they use them in order to provide delayed key disclosure. While, in [50], 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 [20] (by Hu, Johnson and Perrig) hash chains are also used in combination with DSDV-SQ [3] (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 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 used with any suitable authentication and key distribution scheme. But finding such a scheme is not straightforward.

Ariadne [21], by the same authors, is based on DSR [23] and TESLA [38] (on which it is based its authentication mechanism). It also requires clock synchronization, which is, arguably, 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, it could be 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).

4.3 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 [18]. Zhou and Haas [52] 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.

The problem of compromised nodes is not addressed here since it is, arguably, not critical in non military scenarios. Availability is considered to be outside of scope. 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 research work the following requirements were consid-

---

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

ered:

- **Import authorization**: It is important to note that in here it is not referring to the traditional meaning of authorization. What means is that the ultimate authority about routing messages regarding a certain destination node is that node itself. Therefore, route information will only be authorized in a 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**: Nodes need to be able to verify that the node is the one it claims to be.
- **Integrity**: In addition, nodes 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.

4.4 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 two types of information could be distinguished: 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 being used 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).

4.5 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 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 [36], 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.

4.6 Securing AODV

Let us 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

Value	Hash function
0	Reserved
1	MD5HMAC96 [28]
2	SHA1HMAC96 [29]
3-127	Reserved
128-255	Implementation dependent

Table 4.1: Possible values of the Hash Function field

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.

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 that will be refereed as Signature Extension.

4.6.1 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 4.1.

$$Hash\_Function = h$$

- 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 later, all of them but the Hash field are signed to protect its integrity.

4.6.2 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, SAODV 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.

4.6.3 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.

4.6.4 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.

4.6.5 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 modifying it. 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, it is nota 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 [44]. Another possibility would be to use forward-secure signature schemes [25]. 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, 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 [34] 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.

4.7 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.

Let us look now roughly, just as an example, to the Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR) [23], trying to see how it could be secured.

SRP [34] and Ariadne [21] 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 can be considered 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 here, 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 taking into account 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 [21]).

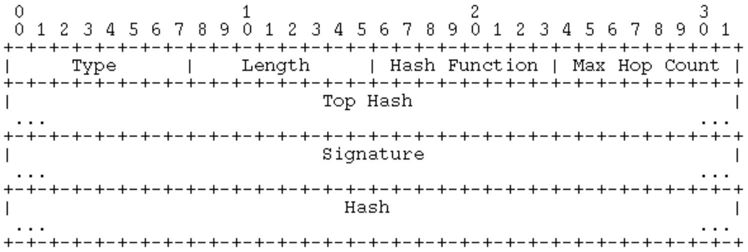
Another solution would be that intermediate nodes would sign the routing message, but that a node would only verify the signature of an intermediate node in the case it needs to send a packet to this route. But it still requires all intermediate nodes to sign the message (which is not good when the message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

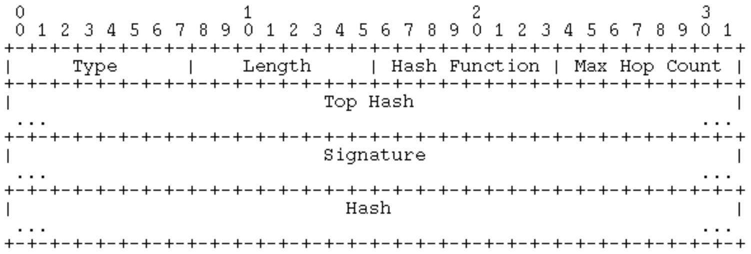
4.8 Secure AODV Extensions

The figures show the format of the SAODV Signature Extensions.



```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Length     | Hash Function | Max Hop Count |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Top Hash                           |
 ...                                                          ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Signature                          |
 ...                                                          ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              Hash                             |
 ...                                                          ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4.1: RREQ (Single) Signature Extension



```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Length     | Hash Function | Max Hop Count |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Top Hash                            |
 ...                                                         ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Signature                           |
 ...                                                         ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                             Hash                              |
 ...                                                         ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4.2: 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 4.2: RREQ and RREP Signature Extension Fields

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Length     | Hash Function | Max Hop Count |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|R|A|                    Reserved                   |Prefix Sz|
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Top Hash                           |
 ...                                                         ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Signature                           |
 ...                                                         ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                       Signature for RREP                      |
 ...                                                         ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              Hash                             |
 ...                                                         ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4.3: RREQ Double Signature Extension

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Length     | Hash Function | Max Hop Count |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                            Top Hash                           |
 ...                                                         ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Signature                           |
 ...                                                         ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                          Old Lifetime                         |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                Signature of the new Lifetime                  |
 ...                                                         ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                              Hash                             |
 ...                                                         ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4.4: RREP Double Signature Extension

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|     Type      |    Length     |            Reserved           |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Signature                           |
 ...                                                         ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4.5: RERR Signature Extension

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
                                +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
                            ... |     Type      |     Length    |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                           Signature                           |
 ...                                                         ...
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

Figure 4.6: RREP-ACK 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 4.3: RREQ Double Signature Extension Fields

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 4.4: RREP Double Signature Extension Fields

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 4.5: RERR and RREP-ACK Signature Extension Fields