IOSR Journal of Computer Engineering (IOSR-JCE)
e-ISSN: 2278-0661, p- ISSN: 2278-8727Volume 16, Issue 1, Ver. IX (Feb. 2014), PP 21-33
www.iosrjournals.org

Security Measures in Aodv for Ad Hoc Network

Nidhi Gour[1], Monika Agarwal[2], Heena Singh[3], Ajay Kumar[4]
[1]*Department of CSE, JECRC University, India*
[2]*Department of CSE, JECRC University, India*
[3]*Department of CSE, JECRC University, India*
[4]*Assistant Professor, CSE department JECRC University, India*

Abstract: *Mobile ad hoc network (MANET) is an autonomous system of mobile nodes connected by wireless links. Each node operates not only as an end system, but also as a router to forward packets. The nodes are free to move about and organize themselves into a network. In either case, the proliferation of MANET-based applications depends on a multitude of factors, with trustworthiness being one of the primary challenges to be met. Despite the existence of well-known security mechanisms, additional vulnerabilities and features pertinent to this new networking paradigm might render such traditional solutions inapplicable. In particular, the absence of a central authorization facility in an open and distributed communication environment is a major challenge, especially due to the need for cooperative network operation. In this paper*
We detail solution against security threat as impersonation specifically in AODV routing protocol that are widely used in MANET. Through this proposal we can identify such kind of threat in decentralized network.
Keyword: *MANET, AODV, SEAD, ARAN, ARIADNE etc.*

I. Introduction

Mobile Ad-hoc Network is self configured structure collection of mobile nodes that communicate to each other by forwarding packets within its certain radio range whereas others nodes need the help of intermediate nodes to route their packets. Mobile phone,Laptop, personal computer, personal Digital Assistance are included in nodes. Mobile ad hoc network offers quick and horizontal network deployment in conditions where it is not possible otherwise. Ad-hoc is a Latin word, which means "for this or for this only."
The routers, the participating nodes act as router, are free to move randomly and manage themselves arbitrarily thus, the network's wireless topology may change rapidly and unpredictably. Such a network may operate in a standalone fashion, or may be connected to the larger Internet. To establish a route between nodes routing protocols are engaged.

Routing protocols are divided into three categories in MANET first is Proactive or table driven in which route are fixed and quickly connected to nodes for collaborate, routes are updated in each connection. Second is
Reactive or On demand protocol in which connection establish when they are needed if route is not available to destination node it initiate route discovery process until route made him available. Third one is hybrid protocol that are combination of both proactive and reactive protocols.

Optimized Link State routing (OLSR) and Destination Sequenced Distance Vector Routing (DSDV) protocols are proactive routing protocols. Ad-hoc On demand Distance Vector (AODV) and Dynamic Source Routing (DSR) protocols are reactive protocols. AODV is comparatively secure routing protocol that are carrying widely in wireless communication system.

In this paper we summarize the solution that provide security aspect to AODV against malicious threat specifically impersonation, man-in-middle, Denial of Service.

II. AODV

Ad-hoc On Demand Distance Vector Routing is well known routing protocol in wireless media. As In [2],It is a reactive protocol: nodes in the network exchange routing information only when a communication must take place and keep this information up-to-date only as long as the communication lasts. When a node must send a packet to another node, it starts a route discovery process in order to establish a route toward the destination node. Therefore, it sends its neighbours a route request message (RREQ). Neighbouring nodes receive the request, increment the hop count, and forward the message to their neighbours, so that RREQs are actually broadcasted using a flooding approach. The goal of the RREQ message is to find the destination node, but it also has the side effect of making other nodes learn a route towards the source node (the reverse route): a node that has received a RREQ message, with source address S from its neighbour A, knows that it can reach S through A and records this information in its routing table along with the hop count. The RREQ message will eventually reach the destination node, which will react with a route reply message (RREP). The RREP is sent as a unicast, using the path towards the source node established by the RREQ. Similarly to what happens with

RREQs, the RREP message allows intermediate nodes to learn a route toward the destination node (i.e., the originator of the RREP). Therefore, at the end of the route discovery process, packets can be delivered from the source to the destination node and vice versa. A third kind of routing message, called route error (RERR), allows nodes to notify errors, for example, because a previous neighbour has moved and is no longer reachable. If the route is not active (i.e., there is no data traffic flowing through it), all routing information expires after a timeout and is removed from the routing table.

Source address	Broadcast Id	Source Sequence number	Destination address	Destination Sequence number	Hop count

III. Security Threat In Aodv

A node is malicious if it is an attacker that cannot identify itself as a legitimate node due to the lack of valid cryptographic information. A node is compromised if it is an inside attacker who is behaving maliciously but can be identified by the network as a legitimate node and is trusted by other nodes. A node is called selfish when it tends to deny its own resources for the benefits of other nodes in order to save its own resources. Since AODV has no security mechanisms several attacks can be launched against the AODV routing protocol [2].

Four types of attacks addressed by authors of [1] are:

a) Distributed false route request
b) Denial of service
c) Destination is compromised
d) Impersonation

a) Distributed false route request:

A route request is generated whenever a node has to send data to the particular destination. A malicious node might generate frequent, unnecessary route requests then it will be difficult to identify the malicious node. Route request messages are broadcast messages. When the node in the network receive a number of route requests that is greater than a threshold count by a specific source for a destination in a particular time interval, the node is declared as malicious and the information is propagated in the network.

b) Denial of service:

A malicious node launches the denial of service attack by transmitting false control packets and using the entire network resources. Thereby other nodes are underprivileged of the resources. Denial of service can be launched by transmitting fake routing packets or data packets. It can be identified if a node is generating the control packets that are more than the threshold count in a particular time interval t frequency.

c) Destination is compromised:

A destination might not be able to reply, if it is (i) not in the network (ii)overloaded (iii) it did not receive route request or if it is (iv) malicious. This attack is identified when the source does not receive the reply from the destination in a particular time interval *t* wait.Besides the neighbours generate probe/ hello packets to determine connectivity. If the node is in the network and does not respond to route requests destined for it, it is identified as malicious.

d) Impersonation:

It can be avoided if sender encrypts the packet with its private key and other nodes decrypts with the public key of the sender. If the receiver is not able to decrypt the packet, the sender might be not the real source and hence packet will be dropped.

Three types of attacks addressed by authors of [2] are:

a) Message Tampering Attack
b) Message Dropping Attack
c) Message Reply Attack

a) Message Tampering Attack:

An attacker can alter the content of routing messages and forward them with fallacious information. For instance, when forwarding a RREQ generated by a source node to discover a route to the destination node, an attacker can reduce the hop count field to increase the chances of being in the route path between source and destination so it can analyze the statement 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.

b) Message Dropping Attack:

Both attackers and selfish nodes can intentionally drop some or all routing and data messages. while all the mobile nodes within a MANET function as both end hosts and router, this attack can paralyze the network completely as the number of message dropping increase.

c) Message Reply Attack:

Attackers can retransmit eavesdropped messages again later in a different place. One type of reply attacks is the wormhole attack. A wormhole attacker can tunnel an RREQ directly to a destination node. Since a wormhole attacker may not increase the hop-count value, it prevents any other routes from being discovered. The worm hole attack can be combined with the message dropping attack to prevent the destination node from receiving packets.

The security requirements for AODV routing protocol include [1]:
a) Source authentication:
The receiver should be able to confirm that the identity of the source is indeed who or what it claims to be.

b) Neighbour authentication:
The receiver should be able to confirm that the identity of the sender is to be sure who or what it claims to be.

c) Message integrity:
The receiver should be able to verify that the content of a message has not been transformed either maliciously or accidentally in transit.

d) Access control:
It is necessary to ensure that mobile nodes looking for to gain access to the network have the appropriate access rights.

We described the safety measure against all of above attack in AODV routing protocol for Ad hoc network that have been proposed by the authors in [1],[2] that are:
a) Secure Ad hoc On Demand Distance Vector (SAODV)
b) Authenticated Routing for Ad hoc Network (ARAN)
c) TESLA
d) ARIADNE
e) SEAD
f) SAR
g) SRP
h) CONFIDENT
g) NASRP

The major vulnerabilities present in the AODV protocol that have been discussed by author in [1] are:
1) ***Deceptive incrementing of Sequence Numbers:***
Destination Sequence numbers determine the freshness of a route. The destination sequence numbers maintained by different nodes are only update when a newer control packet is received with a higher sequence number. On the other hand, a malicious node can increase this number in order to advertise fresher route to a particular destination.

2) ***Deceptive decrementing of Hop Count:***

AODV prefers route freshness over route length. A node would prefer a control packet with a larger destination sequence number and hop count over a control packet with a smaller destination sequence number and hop count. On the other hand, in case where the destination sequence numbers are same for two control packets, the route with the smaller hop count is chosen. A malicious node can easily exploit this mechanism by decrementing the Hop Count to generate fallacious smaller routes to destination.

IV. Security Measures

To overcome at these vulnerabilities Secure Ad hoc On-demand Distance Vector Routing protocol have been discussed by authors in [2].

1. Secure Ad-hoc On-demand Distance Vector (SAODV)

Securing the AODV protocol can be divided into the following three broad categories:
1) Key Exchange

2) Secure Routing
3) Data Protection

1) Key Exchange:
Most of the current key exchange protocols are reliant upon a central trust authority for initial authentication. All nodes, before entering a network, procure a one-time public and private key pair from the CA along with the CA's public key. After this, the nodes can negotiate session keys among each other, without any reliance on the CA, using any suitable key exchange protocol for ad-hoc networks. These session keys are used for securing the routing process and subsequently the data flow. To avoid multiple peer-to-peer encryptions during broadcast or multicast operations, a group session key may be established between immediate neighbours using a suitable Group Keying Protocol. This mechanism absolves the ad-hoc network of superfluous requirements and provides necessary elements to secure both routing and data in presence of malicious nodes by providing security services like authentication, confidentiality and integrity.

2) Secure Routing:
Ad-hoc On- Demand distance Vector routing protocol operates at the third layer of the TCP/IP protocol suite using UDP port 654. The source node that requires a route to a destination broadcasts a ROUTE REQUEST packet, each intermediate recipient node retransmits the packet, if not a duplicate, and the final destination unicast a ROUTE REPLY packet back to the original sender. For route maintenance it uses ROUTE ERROR packets that inform active users of route failures. The ROUTE REQUEST and ROUTE REPLY packets are usually customized by the intermediate nodes so as to add necessary routing information to these packets.
To restrict modification of routing packets by intermediate nodes, we recommend peer-to-peer symmetric encryption of all routing information. All routing control packets between nodes are first encrypted and then transmitted.

3) Data Protection:
Once protected routes have been established, secure data transfer is relatively straightforward. To ensure connection confidentiality a source node adopts the following steps:
1) Any Node 'x' desiring to establish an end-to-end secure data channel, first establishes a session key Kxy with the intended Node 'y' using the key exchange protocol.
2) It then symmetrically encrypts the data packet using the session key Kxy and transmits it over the secure route.
3) The intermediate nodes simply forward the packet in the intended direction.
4) When the encrypted data packet reaches the destination it is decrypted using the session key Kxy.
5) Steps 2 to 4 are followed for all further data communication.

There are two options available for obtaining the session key for secure data exchange. Following symbols will be used in the proposed options
Source- S
Destination –D
Session key –KS
Encrypted session key -KE
Public key of x-KAX
Private key of x-KBX where X is either source or destination.
EK encryption using key K, DK decryption using key K.

Option 1
A source generates RREQ, attaches its certificate and sends it for route discovery of destination. Besides source also requests for a session key from the destination node. The intermediate nodes rebroadcast the RREQ packet in accordance with the operation of AODV protocol. On receipt of RREQ, the destination node verifies the certificate of source and on confirmation generates a session key. The destination also encrypts the session key with the public key of the source (KE = EKAS (KS)). The destination finally sends RREP including encrypted session key to the source. On receipt source decrypts the encrypted session key by its private key and obtain the session key (KS = DKBS (KE)). The obtained session key will finally be used for secure data exchange

Option 2
Source generates RREQ, attaches its certificate and sends it for route discovery of destination. Source also attaches a request for a session key from the destination node. The intermediate nodes rebroadcast the RREQ

packet in accordance with the operation of AODV protocol. On receipt of RREQ, the destination node verifies the certificate of source and on confirmation generates a session key. Destination encrypts the session key with its private key as (KE1= EKBD (KS)) and further encrypts KE1 with the public key of the source as (KE= EKAS (KE1)).Destination respond with RREP attach its certificate and encrypted session key KE.
On receipt, source confirms the authenticity of destination from its certificate, decrypts the session key first through its private key and then through public key of destination as (KE1= DKBS (KE)) and (KS= DKAD (KE1) respectively. Finally session key is obtained that will subsequently be used for secure data exchange.

This approach provides authentication through the key exchange and all other services like confidentiality, integrity and non-repudiation rely on the accuracy of the authentication service. Following are the seven requirements that this approach satisfies:
1) Authorized nodes perform route computation and discovery.
2) Minimal exposure to network topology.
3) Detection of spoofed routing messages.
4) Detection of fabricated routing messages.
5) Detection of altered routing messages.
6) Avoiding formation of routing loops.
7) Prevent redirection of routes from shortest paths.

2. Authenticated Routing for Ad-hoc Network (ARAN)
Kimaya, et al. have proposed a design to protect against malicious activity in Ad hoc network that was ARAN. Authenticated Routing for Ad hoc Networks (ARAN) detects and protects against malicious actions by third parties and peers in one particular ad hoc environment. ARAN introduces *authentication*, *message integrity* and *non-repudiation* to an ad hoc environment as a part of a minimal security policy[3].

ARAN consists of a preliminary certification process followed by a route instantiation process that guarantees end-to-end authentication.

a) Certification
ARAN requires the use of a trusted certificate server T whose public key is known to all valid nodes. Keys are a priori generated and exchanged through an existing, perhaps out of band, relationship between T and each node. Before entering the ad hoc network, each node must request a certificate from T. Each node receives exactly one certificate after securely authenticating their identity to T.
A node 'A' receives a certificate from T as follows:

$$T\text{->}A: cert_A = [IP_A, K_{A+}, t, e]\ K_{T-} \qquad (1)$$

The certificate contains the IP address of A the public key of A, a timestamp t of when the certificate was created, and a time e at which the certificate expires. All nodes must maintain fresh certificates with the trusted server. Nodes use these certificates to authenticate themselves to other nodes during the exchange of routing messages.

b) Authenticated Route Discovery
The goal of end-to-end authentication is for the source to verify that the intended destination was reached. In this process, the source trusts the destination to chose the return path. Source node, A begins route instantiation to destination X by broadcasting to its neighbours a *route discovery packet* (RDP):

$$A\text{->}\ brdcast: [RDP, IP_X, cert_A, N_A, t]\ K_{A-} \qquad (2)$$

The RDP includes a packet type identifier ("RDP"), the IP address of the destination (IP_X), A's certificate ($cert_A$), a nonce N_A and the current time t, all signed with A's private key. Each time A performs route discovery, it monotonically increases the nonce. The nonce and timestamp are used in concurrence with each other to allow for ease of nonce recycling. The nonce is made large enough that it will not need to be recycled within the probable clock skew between receivers. Other nodes then store the nonce they have last seen for a particular node along with its timestamp. If a nonce later re-appears in a valid packet that has a later timestamp, the nonce is assumed to have wrapped around, and is therefore accepted. Note that a hop count is not included with the message.

When a node receives an RDP message, it sets up a reverse path back to the source by recording the neighbour from which it received the RDP. This is in anticipation of eventually receiving a reply message that it will need to forward back to the source. The receiving node uses A's public key, which it extracts from A's

certificate, to validate the signature and verify that A's certificate has not expired. The receiving node also checks the (N_A, IP_A) tuple to verify that it has not already processed this RDP. Nodes do not forward messages for which they have already seen the tuple; otherwise, the node signs the contents of the message, appends its own certificate, and forward broadcasts the message to each of its neighbours. The signature prevents spoofing attacks that may alter the route or form loops.

Let B be a neighbour that has received from A, the RDP broadcast, which it subsequently rebroadcasts.

$$B\text{-> brdcast: } [[RDP, IP_X, cert_A, N_A, t]\ K_{A-}]\ K_{B-}\ ,\ cert_B \qquad (3)$$

Upon receiving the RDP, B's neighbour C validates the signature with the given certificate. C then removes B's certificate and signature, records B as its predecessor, signs the contents of the message originally broadcast by A, appends its own certificate, and forward broadcasts the message. C then rebroadcasts the RDP.

$$C\text{-> brdcast: } [[RDP, IP_X, cert_A, N_A, t]\ K_{A-}]\ K_{C-},\ cert_C \qquad (4)$$

Each node along the path repeats these steps of validating the previous node's signature, removing the previous node's certificate and signature, recording the previous node's IP address, signing the original contents of the message, appending its own certificate and forward broadcasting the message.

c) Authenticated Route Setup

Ultimately, the message is received by the destination, X, who replies to the first RDP that it receives for a source and a given nonce. There is no guarantee that the first RDP received travelled along the shortest path from the source. An RDP that travels along the shortest path may be prevented from reaching the destination first if it encounters congestion or network delay, either legitimately or maliciously manifested. In this case,

Conversely, a non-congested, non-shortest path is likely to be preferred to a congested shortest path because of the reduction in delay. Because RDPs do not contain a hop count or specific recorded source route, and because messages are signed at each hop, malicious nodes have no opportunity to redirect traffic with the exploits

After receiving the RDP, the destination unicasts a Reply (REP) packet back along the reverse path to the source.

Let the first node that receives the REP sent by X be node D.

$$X\text{->D: } [REP, IP_A, cert_X, N_A, t]\ K_{X-} \qquad (5)$$

The REP includes a packet type identifier ("REP"), the IP address of A(IP_A), the certificate belonging to X ($cert_X$), the nonce and associated timestamp t sent by A. Nodes that receive the REP forward the packet back to the predecessor from which they received the original RDP. Each node along the reverse path back to the source signs the REP and appends its own certificate before forwarding the REP to the next hop.

Each node checks the nonce and signature of the previous hop as the REP is returned to the source. This avoids attacks where malicious nodes instantiate routes by impersonation and replay of X's message. When the source receives the REP, it verifies the destination's signature and the nonce returned by the destination

d) Key Revocation

In the event that a certificate needs to be revoked, the trusted certificate server,T, sends a broadcast message to the ad hoc group that announces the revocation. Calling the revoked certificate $cert_r$ the transmission appears as:

$$T\text{-> brdcast: } [revoke, cert_r]\ K_{T-} \qquad (6)$$

ARAN provides a solution for securing routing in the managed-open environment. ARAN provides authentication and non-repudiation services using pre-determined cryptographic certificates that guarantees end-to-end authentication.

3. TESLA

Yih Chun hu et al. have reviewed TESLA that is efficient and adds only a single message authentication code (MAC) to a message for broadcast authentication. Adding a MAC (computed with a shared key) to a message can provide secure authentication in point-to-point communication [4].

To use TESLA for authentication, each sender chooses a random initial key K_N and generates a *one-way key chain* by repeatedly computing a one-way hash function H on this starting value: $K_{N-1} = H[K_N]$, $K_{N-2} = H[K_{N-1}]$,

. . . . In general, $K_i = H[K_{i+1}] = H_{N-i}[K_N]$. To compute any previous key K_j from a key K_i , $j < i$, a node uses the equation $K_j = H_{i-j}[K_i]$. To authenticate any received value on the one-way chain, a node applies this equation to the received value to determine if the computed value matches a previous known authentic key on the chain. Coppersmith and Jakobsson present efficient mechanisms for storing and generating values of hash chains [10]

Each sender pre-determines a schedule of the time at which it publishes each key of its one-way key chain, in the reverse order from generation; that is, a sender publishes its keys in the order $K_0, K_1, \ldots, K_N$. A simple key exposé schedule, for example, would be to publish key K_i at time $T_{0+i} \cdot I$, where T_0 is the time at which K_0 is published, and *I* is the key publication interval.

TESLA relies on a receiver's ability to determine which keys a sender may have already published, based on loose time synchronization between nodes.

TESLA relies on a receiver's ability to determine which keys a sender may have already published, based on loose time synchronization between nodes. Let Δ be the maximum time synchronization error between any two nodes, the value Δ must be known by all nodes. To send a packet, the sender uses a pessimistic upper bound τ on the end-to-end network delay between nodes and picks a key K_i from its one way key chain which, at the time any receiver is expected to receive the packet. For example, the sender could choose a key K_i that it will not publish until a time at least $\tau + 2\Delta$ in the future. The value $2\Delta$is used here because the receiver's clock may be ahead of the sender's clock by Δ, so at time t_s at the sender, it is $t_s + \Delta$ at the receiver. In sending the packet, the sender adds a message authentication code (MAC), computed using key K_i , to the packet. When the packet reaches the receiver, it will be $t_s + \tau + \Delta$, and the receiver will discard the packet if the key might have been published already. Since the receiver knows the sender's clock may be faster by Δ, the receiver will reject the packet unless it is received at least Δ before the scheduled key release time, so the receiver must be able to verify that the key is released at time $t_s + \tau + 2\Delta$ or later.

When a receiver receives a packet authenticated with TESLA, it first verifies the TESLA *security condition* that the key K_i used to authenticate the packet cannot yet have been published.

4. ARIADNE

Yhi Chun Hu et. al have proposed a protocol was ARIADNE to overcome the vulnerabilities of TESLA in [4]. We first overview the features of the protocol in three stages:

- A mechanism that enables the target of a Route Discovery to verify the authenticity of the ROUTE REQUEST
- Three alternative mechanisms for authenticating data in ROUTE REQUESTs and ROUTE REPLYs
- An efficient per-hop hashing technique to verify that no node is missing from the node list in the REQUEST.

We assume that some initiator node *S* performs a Route Discovery for a target node D, and that they share the Secret keys K_{SD} and K_{DS}, respectively, for message authentication in each direction.

- Target authenticates ROUTE REQUESTs.

To convince the target of the legitimacy of each field in a ROUTE REQUEST, the initiator simply includes in the REQUEST a MAC computed with key K_{SD} over unique data, for example a timestamp.
The target can easily verify the authenticity and freshness of the ROUTE REQUEST using the shared key K_{SD}

- Three techniques for route data authentication.

In a Route Discovery, the initiator wants to authenticate each individual node in the node list of the ROUTE REPLY. A secondary requirement is that the target can authenticate each node in the node list of the ROUTE REQUEST, so that it will return a ROUTE REPLY only along paths that contain only legitimate nodes. In this section, we present three alternative techniques to achieve node list authentication: the TESLA protocol, digital Signatures, and standard MACs.

- ❖ When Ariadne Route Discovery is used with TESLA, each hop authenticates the new information in the REQUEST. The target buffers and does not send the REPLY until intermediate nodes can release the corresponding TESLA keys. The TESLA security condition is verified at the target, and the target includes a MAC in the REPLY to certify that the security condition was met.
- ❖ Ariadne Route Discovery is used with digital signatures, the MAC list in the ROUTE REQUEST becomes a signature list, where the data used to compute the MAC is instead used to compute a signature. More willingly than computing the target MAC using a Message Authentication Code, a signature is used. Finally, no key list is required in the REPLY.
- ❖ Ariadne Route Discovery using MACs is the most efficient of the three alternative authentication mechanisms, but it requires pair wise shared keys between all nodes. When Ariadne is used in this manner,

the MAC list in the ROUTE REQUEST is computed using a key shared between the target and the current node, before using the TESLA key of the current node. The MACs are verified at the target and are not returned in the ROUTE REPLY. Consequently, the target MAC is not computed over the MAC list in the REQUEST. No key list is required in the REPLY.

- Per-hop hashing.

Authentication of data in routing messages is not sufficient, as an attacker could remove a node from the node list in a ROUTE REQUEST. We use one-way hash functions to verify that no hop was omitted, and we call this approach *per-hop hashing*. To change or remove a previous hop, an attacker must either hear a ROUTE REQUEST without that node listed, or it must be able to invert the one way hash function.

- Ariadne Route Discovery with TESLA

We now describe in detail the version of Ariadne Route Discovery using TESLA broadcast authentication. We assume that every end-to-end communicating source–destination pair of nodes A and B share the MAC keys K_{AB} and K_{BA}. We also assume that every node has a TESLA one-way key chain, and that all nodes know an authentic key of the TESLA one-way key chain of each other node. Route Discovery has two stages:

- The initiator floods the network with a ROUTE REQUEST
- The target returns a ROUTE REPLY

To secure the ROUTE REQUEST packet, Ariadne provides the following properties:

- The target node can authenticate the initiator (using a MAC with a key shared between the initiator and the target)
- The initiator can authenticate each entry of the path in the ROUTE REPLY (each intermediate node appends a MAC with its TESLA key)
- No intermediate node can remove a previous node in the node list in the REQUEST or REPLY (a one-way function prevents a compromised node from removing a node from the node list).

A ROUTE REQUEST packet in Ariadne contains eight fields:

1) ROUTE REQUEST
2) Initiator
3) Target
4) Id
5) Time interval
6) Hash chain,
7) Node list
8) MAC list

The initiator and target are set to the address of the initiator and target nodes, respectively. The initiator sets the id to an identifier that it has not recently used in initiating a Route Discovery. The time interval is the TESLA time interval at the pessimistic expected arrival time of the REQUEST at the target, accounting for clock skew specifically, given τ , a pessimistic transit time, the time interval could be set to any time interval for which the key is not released within the next $\tau + 2\Delta$time.

The initiator of the REQUEST then initializes the hash chain to MAC K_{SD} (initiator, target, id, time interval) and the node list and MAC list to empty lists. When any node A receives a ROUTE REQUEST for which it is not the target, the node checks its local table of (initiator, id) values from recent REQUESTs it has received, to determine if it has already seen a REQUEST from this same Route Discovery. If it has, the node discards the packet.

The node also checks whether the time interval in the REQUEST is valid If the time interval is not valid, the node discards the packet. Otherwise, the node modifies the REQUEST by appending its own address, A, to the node list in the REQUEST, replacing the hash chain field with H [A, hash chain], and appending a MAC of the entire REQUEST to the MAC list.

The node uses the TESLA key K_{Ai} to compute the MAC, where i is the index for the time interval specified in the REQUEST. Finally, the node rebroadcasts the modified REQUEST.

When the target node receives the ROUTE REQUEST, it checks the validity of the REQUEST by determining that the keys from the time interval specified have not been disclosed yet, and that the *hash chain* field is equal to $H[\eta n, H[\eta n-1, H[\ldots, H[\eta 1, \text{MAC } K_{SD} \text{ (initiator, target, id, time interval)}] \ldots]]]$

S: h0 = MACK_{SD} (REQUEST, S, D, id, ti)
S → *: {REQUEST, S, D, id, ti, h0, (), ()}

A: h1 = H [A, h0]
MA = MAC K_{Ati} (REQUEST, S,D, id, ti, h1, (A), ())
A → *: {REQUEST, S, D, id, ti, **h1**, (**A**), (**MA**)}

B: h2 = H[B, h1]
MB = MAC K_{Bti} (REQUEST, S, D, id, ti, h2, (A,B), (MA))
B → *: {REQUEST, S, D, id, ti, **h2**, (A,**B**), (MA,**MB**)}

C: h3 = H[C, h2]
MC = MAC K_{Cti} (REQUEST, S, D, id, ti, h3, (A,B,C), (MA,MB))
{REQUEST, S, D, id, ti, **h3**, (A,B,**C**), (MA,MB,**MC**)}

D: MD = MACKDS (REPLY, D, S, ti, (A,B,C), (MA,MB,MC))
D →C: {REPLY, D, S, ti, (A,B,C), (MA,MB,MC),**MD**, ()}
C → B: {REPLY, D, S, ti, (A,B,C), (MA,MB,MC),MD, (**K_{Cti}**)}
B → A: {REPLY, D, S, ti, (A,B,C), (MA,MB,MC),MD, (KCti , **K_{Bti}**)}
A →S: {REPLY, D, S, ti, (A,B,C), (MA,MB,MC),MD, (KCti, KBti ,**K_{Ati}**)}

Figure 1.Route Discovery example in Ariadne. The initiator node *S* is attempting to discover a route to the target node *D*. The bold underlined font indicates changed message fields, relative to the previous message of that type.

Where ηi is the node address at position i of the node list in the REQUEST, and where n is the number of nodes in the node list. If the target node determines that the REQUEST is valid, it returns a ROUTE REPLY to the initiator, containing eight fields:

1) ROUTE REPLY
2) Target
3) Initiator
4) Time interval
5) Node list
6) MAC list
7) Target MAC
8) Key list

The target, initiator, time interval, node list, and MAC list fields are set to the corresponding values from the ROUTE REQUEST, the target MAC is set to a MAC computed on the preceding fields in the REPLY with the key K_{DS}, and the key list is initialized to the empty list. The ROUTE REPLY is then returned to the initiator of the REQUEST along the source route obtained by reversing the sequence of hops in the node list of the REQUEST.

A node forwarding a ROUTE REPLY waits until it is able to disclose its key from the time interval specified it then appends its key from that time interval to the key list field in the REPLY and forwards the packet according to the source route indicated in the packet.

When the initiator receives a ROUTE REPLY, it verifies that each key in the key list is valid, that the target MAC is valid, and that each MAC in the MAC list is valid. If all of these tests succeed, the node accepts the ROUTE REPLY; otherwise, it discards it.

- Optimizations for Ariadne

Caching improvements

When Ariadne is used with broadcast authentication such as TESLA, additional route caching is possible. In the basic Route Discovery mechanism only the initiator of the Discovery can use the route in the ROUTE REPLY, since the target MAC field of the REPLY can only be verified by the initiator. However, if the appropriate data is also broadcast authenticated, any node along a path returned in a REPLY can use that route to reach the target.

Reduced overhead

When Ariadne is used with symmetric authentication such as TESLA or pair wise shared keys, the MAC list in both the ROUTE REQUEST and ROUTE REPLY can be eliminated, and hi can be computed using MAC K_{Ai} {REQUEST, S,D, id, ti, h_{i-1}, (A1, . . . ,Ai)}

The verifier (initiator with delayed broadcast authentication, and target with pair wise shared keys) can then recomputed each h_i given the disclosed symmetric keys.

5. SEAD

Secure Efficient Ad hoc Distance Vector (SEAD) is a proactive routing protocol, based on the design of DSDV [5]. As well the fields common with DSDV, such as destination, metric, next hop and sequence number, SEAD routing tables maintain a hash value for each entry.

The key feature of the proposed security protocol is the use one-way hash chains, using an one way hash function H. Each node computes a list of hash values h_0, h_1 ,..... h_n, where $h_i = H(h_{i-1})$ and $0 < i \leq n$, based on an initial random value h_0. The paper assumes the existence of a mechanism for distributing h_n to all intended receivers. If a node knows H and a trusted value h_n, then it can authenticate any other value h_i, $0 < i \leq n$ by successively applying the hash function H and then comparing the result with h_n.

To authenticate a route update, a node adds a hash value to each routing table entry. For a metric j and a sequence number i, the hash value h_{n-mi+j} is used to authenticate the routing update entry for that sequence number, where m – 1 is the maximum network diameter. Since an attacker cannot compute a hash value with a smaller index than the advertised value, he is not able to advertise a route to the same destination with a greater sequence number, or with a better metric.

SEAD provides a robust protocol against attackers trying to create incorrect routing state in other node by modifying the sequence number or the routing metric. SEAD does not provide a way to prevent an attacker from tampering next hop or destination field in a routing update. Also, it cannot prevent an attacker to use the same metric and sequence number learned from some recent update message, for sending a new routing update to a different destination.

6. Security Aware Routing (SAR)

Security Aware Routing (SAR) [6] is an on demand routing protocol based on AODV. It integrates the trust level of a node and the security attributes of a route to provide an integrated security metric for the requested route. SAR introduces the notion of a trust hierarchy, where nodes of the ad hoc wireless network are divided into different trust levels such that an initiator can inflict a minimum trust level for all the nodes participating in the source-destination communication.

Note that a path with the required trust level might not exist even if the network is connected. .

The initiator of the route in SAR includes a security metric in the route request. This security metric is the minimum trust level of the nodes that can participate in the route discovery. Subsequently, only those nodes that have this minimum security level can participate in the route discovery. All other nodes that are below that trust level will drop the request packets. If an end-to-end path with the required security is found, the intermediate node or destination sends a suitably modified Route Reply.

In the case of multiple paths satisfying the required security attributes, SAR selects the shortest such route. If route discovery fails, then a message can be sent to the initiator so that it can lower the trust level.

In the case of a successful path search, SAR always finds a route with quantifiable guarantee of security. This can be done by having nodes of a trust level share a key. Thus, a node that does not have a particular trust level will not acquire the key for that level, and as a result it will not be able to decrypt the packets using the key of that level. Therefore, it will not have any other option but to drop the packet.

SAR uses sequence numbers and timestamps to stop replay attacks. Threats like interception and subversion can be prevented by trust level key authentication. Modification and fabrication attacks can be stopped by verifying the digital signatures of the transmitted packets.

One of the main drawbacks of using SAR is the excessive encrypting and decrypting required at each hop during the path discovery. A route discovered by SAR may not be the shortest route in terms of hop-count, but it is secure. Such a path ensures that only the nodes having the required trust level will read and re-route the packets, but at the same time malicious node can steal the required key, a case in which the protocol is still open for all kinds of attacks.

7. Secure Routing Protocol (SRP)

Secure Routing Protocol (SRP) [7], is another protocol extension that can be applied to many of the on demand routing protocols used today. SRP defends against attacks that disrupt the route discovery process and guarantees to identify the correct topological information.

The basic idea of SRP is to set up a security association (SA) between a source and a destination node without the need of cryptographic validation of the communication data by the intermediate nodes. SRP assumes that this SA can be achieved through a shared key KST between the source S and target T. Such a security association should exist prior to the route initiation phase. The source S initiates the route discovery by sending a route request packet to the destination T. The SRP uses an additional header called SRP header to the underlying routing protocol packet. SRP header contains the following fields:

- Query sequence number QSEC
- Query identifier number QID
- 96 bit MAC field

Intermediate nodes discard a route request message if SRP header is missing. Otherwise, they forward the request towards destination after extracting QID, source, and destination address. Highest priority is given to nodes that generate requests at the lowest rates and vice versa.

When the target T receives this request packet, it verifies if the packet has originated from the node with which it has SA. If QSEC is greater or equal to QMAX, the request is dropped as it is considered to be replayed. Otherwise it calculates the keyed hash of the request fields and if the output matches SRP MAC then authenticity of the sender and integrity of the request are verified.

On the reception of a route reply, S checks the source address, destination addresses, QID, and QSEC. It discards the route reply if it does not match the currently pending query. In case of a match, it compares reply IP source route with the exact reverse of the route carried in reply packet. If the two routes match then S calculates the MAC by using the replied route, the SRP header fields, and the secure key between source and destination.

If the two MAC match then the validation is successful and it confirms that the reply did came from the destination T. SRP suffers from the lack of validation mechanism for route maintenance messages as it does not stop a malicious node from harming routes to which that node already belongs to. SRP is immune to IP spoofing because it secures the binding of the MAC and IP address of the nodes but it is prone to wormhole attacks and invisible node attacks.

8. Cooperation Of Nodes Fairness In Dynamic Ad-Hoc Networks (Confidant)

Cooperation Of Nodes Fairness In Dynamic Ad-hoc Networks (CONFIDANT) [8] protocol is designed as an extension to reactive source-routing protocol.

It is a collection of components which interact with each other for monitoring, reporting, and establishing routes by avoiding misbehaving nodes. CONFIDANT components in each node include a network monitor, reputation system, trust manager, and a path manager.

Each node in this protocol monitors their neighbours and updates the reputation accordingly. If they detect any misbehaving or malicious node, they can inform other friend nodes by sending an ALARM message. When a node receives such an ALARM either directly from another node or by listening to the ad hoc network, it calculates how honourable the ALARM is based on the source of the ALARM and the total number of ALARM messages about the misbehaving node.

Trust manager sends alarm messages to other nodes to warn them of malicious nodes. Incoming alarms are checked for credibility. Trust manager contains an alarm table, trust level table and a friend list of all trust worthy nodes to which a node will send alarms. Local rating lists and black lists are maintained in the reputation system. These lists are exchanged with friend nodes and timeouts are used to avoid old lists. A node gives more importance to its own experience than to those events which are observed and reported by others.

Whenever the threshold for certain behaviour is crossed, path manager does the re-ranking by deleting the paths containing malicious nodes and ignoring any request from misbehaving nodes. At the same time, it sends an alert to the source of the path so that it can discover some other route.

9. Novel Approach for Secure Routing Protocol (NASRP)

Imran Hossain Faruk have proposed Novel Approach for Secure Routing Protocol in Mobile Ad hoc Network in [9]. NASRP to guarantee the integrity,non-repudiation and confidentiality of routing packets without the presence of central authority. In this approach, it has three steps:

- First step, each node performs a key exchange operation with its one and two hop distance neighbours.
- Second step, secure route establishment
- Third step, secure data communication

Key exchange operation is done in two steps; in the first step, source node (S) exchanges public key (e) with its one hop distance nodes and establish a secret key (SK), and in the second step, source node exchanges public key with its two hop distance nodes and establish a secret key. On establishing the key exchange process node can participate in routing process. In route establishment process, secure route will be established between the

sender and receiver. In the third step, sender and receiver will exchange their public key securely and establish a secret key for communication and then data communication is performed.

a) Key agreement between one hop distance neighbours

In the key agreement between one hop neighbours process, each node sends its public key(e_s) and a sign of hash of public key ($hash(e_s)^{ds}$) to its one hop distance neighbours. Neighbour node receive the request and verify it. After verifying the packet, it generates a reply message which contains the public key (e_n) and a sign MDC (Modification Detection Code) of public key (hash $(e_n)^{dn}$) of itself. After completing the negotiation of public key, initiator node generates a secret key (S_K) and send it by encrypting the receiver's public key (encrypt $e_n(S_K)$). The steps are shown bellow, where S represents source node and N_1 represents one hop distance node and "->" represents direction of communication.

1. S-> N_1 : <Key_Agreement_Req, Request_Id, Sender _Addr, e_S, $hash(e_S)^{d}{}_S$>
2. N_1->S : <Key_Agreement_Rep, Request_Id, Sender_Addr, Neighbour_Addr, e_{N1} , $hash(e_{N1})dN_1>^{e}{}_S$
3. Sender Node (S) generate a secret key (S_K)
4. S->N_1 : < Key_Offer_Req, Request_Id, S_K, $hash(S_K)$ $>^{e}{}_{N1}$
5. N_1->S : < Key_Offer_Rep, Request_Id, hash S_K(Request_Id) $>^{e}{}_s$

b) Key agreement between two hop distance neighbours

In the key agreement process of two hop distance nodes, each node gather information about the two hope neighbours and sends its public key (e_S) and a sign of MDC of public key ($hash(e_S)^{d}{}_S$) to its two hop distance neighbours. After receiving the request neighbour node verify it and send acknowledgement, which contains Request Id, Sender and Neighbour address, Public Key(e_{N2}) of itself, a sign of MDC of public key ($hash(e_{N2})d_{N2}$) of the neighbour and the sing of MDC of public key ($hash(e_S)d_S$) of the sender.

c) Route Request

For finding the route, source node, say S generate the route request(RREQ) packet and broadcasts it. RREQ message is propagated by the intermediates nodes until it reaches the destination node (D). After receiving RREQ message, intermediate node (I) checks whether the message needs to be re-broadcast or not. If it is needed to be re-broadcast it sends a message authentication request (unicast) to the super sender of the RREQ message. On receiving the message authentication request, super sender create a MAC (Message Authentication Code) of RREQ message ($hashS_K(RREQ)$) by using the secret key (S_K) and encrypting it using the intermediates public key (e_I) and then send the entire message ($hashS_K(RREQ)e_I$) to the intermediate node. This process continues until the RREQ reaches the destination node.

Lets A, B and C are three consecutive nodes, where A is source and B and C are the intermediate node through which packets are relaid. On receiving the route request, B doesn't check it's integrity because its directly coming from the source node but C will check it by doing following steps:

1. C-> A : <RREQ_Authen_Req, Broadcast_Id, Sequence_Number, Sender_Addr$>^{e}{}_{A.}$
2. A-> C : <RREQ_Authen _Rep, Broadcast_Id, Sender _Addr, Super_Sender_Addr, $hashS_K(RREQ)>^{e}{}_C$

d) Route Reply

On receiving the route request, destination node (D), generates route reply (RREP) message and send it (unicast) through the reverse path of the arrival path. During the propagation of the RREP packet, intermediate nodes check the authenticity and integrity of the route reply message in the similar way of authentication of RREQ message.

Let X, Y and D are three nodes where D is destination node, which sending route reply packet through Y and X path. X is the one hop distance node so the there is no need of checking the integrity of the packet. Y is two hop distance node so it will check the integrity of the message by sending the authentication request.

Steps are shown below:

1. Y->D : <RREP_Authen_Req, Broadcast_Id, Sequence _Number, Sender_Addr$>^{e}{}_D$
2. D->Y : <RREP_Authen_Rep, broadcast_ Id, Sender _Addr, Super_Sender_Addr, $hashS_K(RREP)>^{e}{}_Y$

e) Route Maintenance

In route maintenance process, during route finding if destination node is un reachable then a error message (RERR) is generated and propagated to the source node. During the RERR message propagation, it follows the message authentication process. Authentication steps are shown bellow.

Let P,Q and R three nodes and R is the error message (RERR) generator, and it will propagate through p and Q nodes, steps are as follows:

1. Q -> R : <RERR_Authen _Req, Host_Unreachable_Id, Sender_Addr>$^{e}_{R}$
2.R->Q : <RERR_Authen_Rep, Host_Unreachable_Id, Sender _Addr, Super_Sender_Addr, hashS_K(REER)>$^{e}_{Q}$

f) Data Communication Between Source and Destination

• Public Key Exchange:

Before start data communication source (S) and destination node (D) must know the public key of each other. To exchange the public key we considered the similar to RREQ message authentication process during the propagation of key exchange message

1. S ->D : <Destin_Addr, (e_S), hash(e_S)>
2. D->S : <Destin_Addr,(e_D), hash(e_D)>$^{e}_{S}$

• Data Packet Exchange:

On receiving the public key, source node (S) generate a share key (sh_K) and encrypt ((sh_K)$^{e}_{d}$) it by destinations public key (e_d) and sends it followed by the data packet. On receiving the key packet, destination node decrypt it and get the shared key. Destination node decrypt all rest of the packets by using the shared key. The detail steps are shown below.

1. S -> D:
   - For secret key: < SK >$^{e}_{D}$
   - For data packet: < data >$^{S}_{K}$

NASRP prevent confidentiality, integrity and non-repudiation in routing protocol against security threat.

V. Conclusion

The lack of central authority and dynamic topology makes MANET network more vulnerable. Our proposed security measures provides security in closed environment where high security is needed and all nodes deployed in the area are from single authority and all these provides mainly authenticity, integrity, non-repudiation andconfidentiality to the communication of the MANET networks. But these have a limitation, more than two malicious node in the routing path in the network, the secure protocol may violates. Secure routing protocol with prevention of all attacks still a open challenge problem, We can add trust evaluation feature so that each node will calculate the trust of its neighbour nodes and based on the trust value, it will be decided that the node will be part of the network of not

References

[1] Tahira Farid_ and Anitha Prahladachar *Secure Routing with AODV Protocol for Mobile Ad Hoc Networks* University of Windsor.

[2] Preeti Bathla, Bhawna Gupta "Security Enhancements in AODV Routing Protocol" *International Journal of Computer Sci ence and Technology* Vol. 2,page no 295-298,june 2011

[3] Kimaya Sanzgiri, Bridget Dahill, Brian Neil Levine, Clay Shields, Elizabeth M. Belding-Royer "A Secure Routing Protocol for Ad Hoc Networks" http:/www.aran.icnp02.pdf

[4] Yih-Chun Hu, Adrian Perrig, David B. Johnson "A Secure On-Demand Routing Protocol for Ad Hoc Networks" Springer,11,page no 21-38,2005

[5] Rajendra Prasad Mahapatra "Taxonomy of Routing Security for Ad-Hoc Network" International Journal of Computer Theory and Engineering, Vol. 2, No. 2 April, 2010

[6] R. Kravets, S. Yi, and P. Naldurg, A Security-Aware Routing Protocol for Wireless Ad Hoc Networks, In ACM Symp. on Mobile= Ad Hoc Networking and Computing, 2001.

[7] P. Papadimitratos and Z. J. Haas, Secure Routing for Mobile Ad hoc Networks, In Proc. of the SCS Communication Networks andDistributed Systems Modeling and Simulation Conference (CNDS 2002), Jan. 2002.

[8] S. Buchegger and J. L. Boudec, Performance Analysis of the CONFIDANT Protocol Cooperation Of Nodes Fairness In Dynamic Ad-hoc NeTworks, In Proc. of IEEE/ACM Symposium on Mobile Ad Hoc Networking and Computing (MobiHOC), Jun. 2002.

[9] Imran Hossain Faruk "A Novel Approach of Secure Routing Protocol for Mobile Ad Hoc Network" www.nitrkl.ac.in

[10] D. Coppersmith and M. Jakobsson, Almost optimal hash sequence traversal, in: *Proceedings of the 4th Conference on Financial Cryptography (FC'02)*, Lecture Notes in Computer Science (2002) pp. 102–119.