FHC-NCTSR: Node Centric Trust Based secure Hybrid Routing Protocol for Ad Hoc Networks

Prasuna V G
Associate Professor. Basaveswara Institute of Information Technology, Barkathpura, Hyderabad, INDIA,

Dr. S Madhusudahan Verma
Professor , . Dept of OR&SQC, Rayalaseema University, Kurnool, AP, INDIA

***Abstract*— To effectively support communication in such a dynamic networking environment as the ad hoc networks, the routing mechanisms should adapt to secure and trusted route discovery and service quality in data transmission. In this context, the paper proposed a routing protocol called Node Centric Trust based Secure Hybrid Routing Protocol [FHC-NCTSR] that opted to fixed hash chaining for data transmission and node centric trust strategy for secure route discovery. The route discovery is reactive in nature, in contrast to this, data transmission is proactive, hence the protocol FHC-NCTSR termed as hybrid routing protocol. The performance results obtained from simulation environment concluding that due to the fixed hash chaining technique opted by FHC-NCTSR, it is more than one order of magnitude faster than other hash chain based routing protocols such as SEAD in packet delivery. Due to the node centric strategy of route discovery that opted by FHC-NCTSR, it elevated as trusted one against to Rushing, Routing table modification and Tunneling attacks, in contrast other protocols failed to provide security for one or more attacks listed, example is ARIADNE that fails to protect from tunneling attack.**

Keywords- Ad hoc network, Dynamic source routing; Hash cains; Manet; Mobile ad hoc networks; NCTS-DSR; Routing Potocol; SEAD; Secure routing .

I. INTRODUCTION

As ad-hoc networks do not rely on existing infrastructure and are self-organizing, they can be rapidly deployed to provide robust communication in a variety of hostile environments. This makes ad hoc networks very appropriate for a broad spectrum of applications ranging from providing tactical communication for the military and emergency response efforts to civilian forums such as convention centers and construction sites. With such diverse applicability, it is not difficult to envision ad hoc networks operating over a wide range of coverage areas, node densities, mobility patterns and traffic behaviors. This potentially wide range of ad hoc network operating configurations poses a challenge for developing efficient routing protocols. On one hand, the effectiveness of a routing protocol increases as network topological information becomes more detailed and up-to-date. On the other hand, in an ad hoc network, mobility may cause frequent changes in the set of communication links of a node [1], requiring large and regular exchanges of control information among the network nodes. And if this topological information is used infrequently, the investment by the network may not pay off. Moreover, this is in contradiction with the fact that all updates in the wireless communication environment travel over the air and are, thus, costly in transmission resources. Routing protocols for ad hoc networks can be classified either as proactive, reactive or hybrid. Proactive or table driven protocols continuously evaluate the routes within the network, so that when a packet needs to be forwarded, the route is already known and can be immediately used. Examples of proactive protocols include DSDV [2], TBRPF [3], and WRP [4]. In contrast, reactive or on-demand protocols invoke a route determination procedure on an on-demand basis by flooding the network with the route query. Examples of reactive protocols include AODV [5], DSR [6], and TORA [7]. The on-demand discovery of routes can result in much less traffic than the proactive schemes, especially when innovative route maintenance schemes are employed. However, the reliance on flooding of the reactive schemes may still lead to a considerable volume of control traffic in the highly versatile ad hoc networking environment. Moreover, because this control traffic is concentrated during the periods of route discovery, the route acquisition delay can be significant. In Section II, we explore the third class of routing protocols the hybrid protocols.

II. PROTOCOL HYBRIDIZATION

The diverse applications of ad hoc network pose a challenge for designing a single protocol that operates efficiently across a wide range of operational conditions and network configurations. Each of the purely proactive or purely reactive protocols described above performs well in a limited region of this range. For example, reactive routing protocols are well suited for networks where the "call to mobility" ratio is relatively low. Proactive routing protocols, on the other hand, are well suited for networks where this ratio is relatively high. The performance of both of the protocol classes degrades when they are applied to regions of ad hoc network space between the two extremes. Given multiple protocols, each suited for a different region of the ad hoc network design space, it makes sense to capitalize on each protocol's strengths by combining them into a single strategy (i.e. hybridization). In the most basic hybrid routing strategy, one of the protocols would be selected based on its suitability for the specific network's characteristics. Although not an elegant solution, such a routing strategy has the potential to perform as well as the best suited protocol for any scenario, and may outperform either protocol over the entire ad hoc network design space.

However, by not using both protocols together, this approach fails to capitalize on the potential synergy that would make the routing strategy perform as well as or better than either protocol alone for any given scenario. A more promising approach for protocol hybridization is to have the base protocols operate simultaneously, but with different "scopes" (i.e., hybridization through multi scoping). For the case of a two-protocol routing strategy, protocol A would operate locally, while the operation of protocol B would be global. The key to this routing strategy is that the local information acquired by protocol A is used by protocol B to operate more efficiently. Thus the two protocols reinforce each other's operation. This routing strategy can be tuned to network behavior simply by adjusting the size of the protocol A's scope. In one extreme configuration, the scope of protocol A is reduced to nothing, leaving protocol B to run by itself. As the scope of protocol A is increased, more information becomes available to protocol B, thereby reducing the overhead produced by protocol B. At the other extreme, protocol A is made global, eliminating the load of protocol B altogether. So, at either extreme, the routing strategy defaults to the operation of an individual protocol. In the wide range of intermediate configurations, the routing strategy performs better than either protocol on its own.

The rest of the paper, section II explores the related work, section III discuss the route discovery strategy of FHC-NCTSR and section IV describes the data transmission approach that fallowed by section V, which explores simulations and results discussion. Section VI is conclusion and section VII explores the bibliography.

III. Related Work

There are known techniques for minimizing 'Byzantine' failures caused by nodes that through malice or malfunction exhibit arbitrary behavior such as corrupting, forging, and delaying routing messages. A routing protocol is said to be Byzantine robust when it delivers any packet from a source node to a destination as long as there is at least one valid route [8]. However, the complexity of that protocol makes it unsuitable for ad hoc networks. Hauser et al[9] avoid that defect by using hash chains to reveal the status of specific links in a link-state algorithm. Their method also requires synchronization of the nodes. Hu[10] introduced another technique called SEAD that uses a node-unique hash chain that is divided into segments. The segments are used to authenticate hop counts. However, DSDV distributes routing information only periodically. The protocols[8, 9, 10] failed to perform when networks with hops in large scale due to their computational complexity in hash chain measurement. In many applications, reactive or on demand routing protocols are preferred. With on demand routing, source nodes request routes only as needed. On demand routing protocols performs better with significantly lower overhead than periodic routing protocols in many situations [11]. The authentication mechanism of Ariadne[11] is based on TESLA[12]. They use only efficient symmetric-key cryptographic primitives. The main drawback of that approach is the requirement of clock synchronization, which is very hard for wireless ad hoc networks. And the protocols [8, 9, 10, 11, 12] failed to protect networks from one or more attacks such as tunneling attack. The protocol FHC-NCTSR proposed in this paper having much scope to perform better in large networks, since its less computational complexity. The node centric and two hop level authentication strategies that opted by FHC-NCTSR helps to deal with various attacks that includes tunneling attack.

IV. Route Discovery Process

Objective of the NCTS-DSR route establishment process is preventing unauthorized hops to join in root during route request process

A. Privileges assumed at each node that exists in the network

- The node that belongs to the network contains the capabilities fallowing
- Able to generate hash method based id for broadcasting packets
- Ability to issue digital certificate
- Ability to maintain the id of hop from which egress data received and id of hop to which ingress data
- Elliptic Curve based cryptography functionality will be used to protect data transmission

B. Hop node registration process

Hop nodes exchange their digital certificates recursively with a time interval ς . The delay between two iterations represented by an interval referred as certificate exchange interval ς . Each node submits its certificate to one and two hop level nodes.

$$\varsigma_h = \frac{\iota}{dt_\iota}$$

ς_h is time interval for node h to submit its digital certificate to neighbor hop nodes
't' is interval threshold
' dt_t ' is distance that can travel by a node h in interval threshold 't'

C. Description of the notations used in route detection process

n_s	source node
n_d	destination node
n_r	relay node
n_e	node from which egress data received by ' n_r
$n_{e'}$	node from which egress data received by '', and two level hop to n_r
n_i	node to which ingress data send by ' n_r '
cer_h	digital certificate of hop *h*

add_h	address of hop node *h*
$lt(cer_h)=cTs_{n_r}-iTs_{n_e}$	$lt(cer_h)$ is certificate life time of the hop node *h*
cTs_{n_r}	current time stamp at the relay node n_r
iTs_{n_e}	is timestamp at created(cer_{n_r} creation time)
p_{id}	Is RREQ unique ID that generated in secure random way
$S_{p_{id}}$	is set of packet ids those transmitted by a node

D. Root Request Process

1) Process of RREQ construction at relay hop node of the source node.

RREQ packet at source node n_s contains

$< add_{n_s}$, add_{n_d} , p_{id} , cer_{n_e} , ς_e , cer_{n_s} , ς_{n_s} , cer_{n_i} , $ECPK_{n_s} >$

Note: Here cer_{n_e} is null.

TABLE 1: ALGORITHM FOR RREQ PACKET EVALUATION AT HOP NODE OF THE SOURCE NODE

Step 1: a. If $p_{id} \in S_{p_{id}}$ then RREQ packet will discarded

b. else adds $p_{id} \in S_{p_{id}}$ to $S_{p_{id}}$ and continues step 2

Step 2: a. If $lt(cer_{n_i})$ is valid and $cer_{n_i} == cer_{n_r}$ then continues step 3,

b. else discards RREQ packet.

[Here cer_{n_i} is because the current node certificate is available at sender node as certificate of one hop node that acts as target for ingress transaction]

Step 3: a. If cer_{n_e} is null then assumes sender is source and continues step 4.

b. Else If cer_{n_e} is not null and $lt(cer_{n_e}) < \varsigma_{n_e}$ and $cer_{n_e} == cer_{n_e'}$ then cer_{n_e} is valid and continues step 4

else RREQ will be discarded.

Step 3:

a. If cer_{n_e} is null then assumes sender is source and continues step 4.

b. Else If cer_{n_e} is not null and $lt(cer_{n_e}) < \varsigma_{n_e}$ and $cer_{n_e} == cer_{n_e'}$ then cer_{n_e} is valid and continues step 4

else RREQ will be discarded.

$cer_{n_e'}$ is certificate of the node that exists as two hop level to current rely node.

[Here cer_{n_e} for senders node is $cer_{n_e'}$ for current rely node]

$lt(cer_{n_e}) = cTs_{n_r} - iTs_{n_e'}$

Here cTs_{n_r} is timestamp at current relay node

cer_{n_e} is certificate carried by RREQ packet

Step 4:

a. If $lt(cer_{n_s}) < \varsigma_{n_s}$ and $cer_{n_s} == cer_{n_e}$ then source node n_s is valid and continues step 5

b. If cer_{n_e} is valid then that RREQ packet will be considered and continues step 5 else that packet will be discarded.

Step 5:

a. If $add_{n_d} \neq add_{n_r}$ then Update the RREQ packet as $< add_{n_s}$, add_{n_d} , add_{n_r} , cer_{n_e} , cer_{n_r} , cer_{n_i} , ς_{n_r} , $ECPK_{n_s} >$ and transmits to n_i .

b. Else if $add_{n_d} \equiv add_{n_r}$ n_r identified as destination node and starts RREP process

2) Process of RREQ construction at relay hop node of the source node

Once packet received by next hop (in that packet referred as) then continues the above four steps in sequence with minor changes, described here:

TABLE 2: ALGORITHM FOR RREQ PACKET EVALUATION AT RELAY NODE THAT IS NOT HOP NODE TO SOURCE NODE

Step 1:

If $p_{id} \in S_{p_{id}}$ then RREQ packet will discarded else adds p_{id} to $S_{p_{id}}$ and continues step Step 2:

Step 2:

a. If $lt(cer_{n_i})$ is valid and $cer_{n_i} == cer_{n_r}$ then continues step 3, else discards RREQ packet.

[Here cer_{n_i} is cer_{n_r} because the current node certificate is available at sender node as certificate of one hop node that acts as target for ingress transaction]

.

Step 3:

a. If cer_{n_e} is not null and $lt(cer_{n_e}) < \varsigma_{n_e}$ and $cer_{n_e} == cer_{n_e'}$ then cer_{n_e} is valid and continues step 4,

else RREQ will be discarded.

$cer_{n_e'}$ is certificate of the node that exists as two hop levels to current rely node.

[Here cer_{n_e} for senders node is $cer_{n_e'}$ for current rely node]

$lt(cer_{n_e}) = cTs_{n_r} - iTs_{n_e'}$

cTs_{n_r} is timestamp at current relay node

cer_{n_e} is certificate carried by RREQ packet

Step 4:

a. If $lt(cer_{n_r}) < \varsigma_{n_e}$ and $cer_{n_r} == cer_{n_e}$ then node n_r is valid and continues else discards the RREQ packet

Step 5:

a. If $add_{n_d} \neq add_{n_r}$ then update the RREQ packet as $< add_{n_s}$, add_{n_d}, $\{add_{m_1}, add_{n_2}, ...add_{n_{r-2}}, add_{n_{r-1}}, add_{n_r}\}$, cer_{n_e}, cer_{n_r}, cer_{n_i}, ς_{n_r}, $ECPK_{n_s} >$ and transmits to n_i

b. Else if $add_{n_d} \equiv add_{n_r}$ n_r identified as destination node and starts RREP process

When compared algorithm in table 2 with algorithm in table 1, a change can be observable at step 3, we are not accepting certificate carried by RREQ as null, since representing in RREQ packet is not source node.

V. RREP PROCESS

Once n_d receives RREQ it performs verification as mentioned in table 2.

Upon successful validation, It performs fallowing functionality.

If RREQ that was received is valid then It collects $ECPK_S$ and calculates $ECPK_d$ (Elliptic curve cryptography approach explained in next section B).

Then it constructs RREP packet at n_d as follows:

$\{add_S, add_d, ECPK_d, lst_{n_r}, cer_{n_e}, cer_{n_i}, cer_{n_d}, \varsigma_{n_d}, p_{id}\}$

Since the RREP packet constructed at n_d, cer_{n_e} is null.

A. Process of RREP packet validation and construction at first hop node of the destination node

TABLE 3: ALGORITHM FOR RREP PACKET EVALUATION AT HOP NODE OF THE DESTINATION NODE

➔Here n_r is hop node of the n_d

Step 1: If $n_r \in lst_{n_r}$ then continues step 2 else discards RREP packet

Step2: If $p_{id} \in S_{p_{id}}$ then RREP packet will discarded else adds p_{id} to $S_{p_{id}}$ and continues step 3

Step 3:

If $lt(cer_{n_i})$ is valid and $cer_{n_i} == cer_{n_r}$ then continues step 4,
else discards RREP packet.

➔[Here cer_{n_i} is cer_{n_r} because the current node certificate is available at sender node as certificate of one hop node that acts as target for ingress transaction]

Step 4: If cer_{n_e} is null then assumes sender is source of the RREP and continues

Else If cer_{n_e} is not null and $lt(cer_{n_e}) < \varsigma_{n_e}$ and $cer_{n_e} == cer_{n_e'}$ then cer_{n_e} is valid and continues step 4,
else RREP will be discarded.

➔ $cer_{n_e'}$ is certificate of the node that exists as two hop level to current rely node.

➔[Here cer_{n_e} for sender's node is $cer_{n_e'}$ for current rely node]

$$lt(cer_{n_e}) = cTs_{n_r} - iTs_{n_e'}$$

➔Here cTs_{n_r} is timestamp at current relay node, cer_{n_e} is certificate carried by RREP packet

Step 5: If $lt(cer_{n_d}) < \varsigma_{n_d}$ and $cer_{n_d} == cer_{n_e}$ then source node n_d is valid and continues step 5
else that packet will be discarded.

Step6: If $add_{n_s} \neq add_{n_r}$ then Update the RREP packet as $< add_{n_s}, add_{n_d}, ECPK_d, lst_{n_r}, cer_{n_e}, cer_{n_r}, cer_{n_i}, \varsigma_{n_r}, p_{id} >$ and transmits to n_i.

Else if $add_{n_s} \equiv add_{n_r}$ n_r identified as source node and stops RREP process

B. Process of RREP construction at relay hop node of the source node

Once RREP packet received by next hop (in that packet referred as n_i) then verifies and continues the process as described in table 4:

TABLE 4: ALGORITHM FOR RREP PACKET EVALUATION AT RELAY NODE THAT IS NOT HOP NODE TO DESTINATION NODE

Step 4:

If cer_{n_e} is not null and $lt(cer_{n_e}) < \varsigma_{n_e}$ and

Step 1:

If $n_r \in lst_{n_r}$ then continues step 2 else discards RREP packet

Step 2:

If $p_{id} \in S_{p_{id}}$ then RREP packet will discarded
else adds p_{id} to $S_{p_{id}}$ and continues step 3

Step 3:

If $lt(cer_{n_i})$ is valid and $cer_{n_i} == cer_{n_r}$ then continues step 4,
else discards RREP packet.
[Here cer_{n_i} is cer_{n_r} because the current node certificate is available at sender node as certificate of one hop node that acts as target for ingress transaction]

$cer_{n_e} == cer_{n_{e'}}$ then cer_{n_e} is valid and continues step 5,
else RREP will be discarded.

➔ $cer_{n_{e'}}$ is certificate of the node that exists as two hop levels to current rely node.

➔[Here cer_{n_e} for senders node is $cer_{n_{e'}}$ for current rely node]

$lt(cer_{n_e}) = cTs_{n_r} - iTs_{n_{e'}}$

cTs_{n_r} is timestamp at current relay node,

cer_{n_e} is certificate carried by RREP packet

Step 5:

If $lt(cer_{n_r}) < \varsigma_{n_e}$ and $cer_{n_r} == cer_{n_e}$ then node n_r is valid and continues step 5 else discards the RREP packet

Step 6 :If $add_{n_s} \neq add_{n_r}$ then update the RREP packet as

$< add_{n_s}, add_{n_d}, ECPK_d, lst_{n_r}, cer_{n_e}, cer_{n_r}, cer_{n_i}, \varsigma_{n_r}, p_{id} >$

and transmits to n_i.

Else if $add_{n_s} \equiv add_{n_r}$ n_r identified as source node and stop RREP process, collects routing path information.

1) Elliptic Curve Cryptography for constrained environments

To form a cryptographic system using elliptic curves, we need to find a "hard problem" corresponding to factoring the product of two primes or taking the discrete algorithm.

Consider the equation $Q = kP$, where Q, P belongs to elliptic curve over $GF(2^n)$ and $k < 2^n$. It is relatively easy to calculate Q given k and P, but it is relatively hard to determine k given Q and P. This is called the discrete algorithm problem for elliptic curves.

KEY EXCHANGE

Key exchange can be done in the following manner. A large integer $q = 2^n$ is picked and elliptic curve parameters a and b. This defines an elliptic curve group of points. Now, pick a base point G = (x1,y1) in E (a,b) whose order is a very large value n. The elliptic curve E and G are the parameters known to all participants.

A key exchange between users A and B can be accomplished as follows:

a) A selects an integer n_A less than n. This is private key of user A. Then user A generates a public key $ECPK_A = n_A * G$; then the public key $ECPK_A$ is appoint on E.

b) User B similarly selects a private key n_B and computes a public key $ECPK_B$.

c) User A generates the secret key $k = n_A * ECPK_B$ and user B generates the secret key $k = n_B * ECPK_A$.

The calculations in step 3 produce the same result. $n_A * ECPK_B \equiv n_A * (n_B * G) \equiv n_B * (n_A * G) \equiv n_B * ECPK_A$

Strength of this key exchange process is to break this scheme, an attacker would need to be able to compute k given G and kG, which is assumed hard and almost not possible in constrained environments

C. Elliptic Curve Encryption/Decryption

The plaintext message m is taken as input in the form of bits of varying length. This message m is encoded and is sent in the cryptographic system as x-y point P_m. This point is encrypted as cipher text and subsequently decrypted. The SHA hash function algorithm can be used as Message digestion and Signature authentication and verification for the message.

As with the key exchange system, an encryption/decryption system requires a point G and an elliptic group E (a, b) as parameters. Each user A selects a private key n_A and generates a public key $ECPK_A = n_A * G$.

To encrypt and send a message p_m to user B, A chooses a random positive integer k and produces the cipher text c_m consisting of pair of points $c_m = \{kG, p_m + kECPK_B\}$

User A has used public key $ECPK_B$ of user B. To decrypt the cipher text, user B multiplies the first point in the pair by secret key of user B and subtracts the result from the second point:

$p_m + kECPK_B - n_B(kG) = p_m + k(n_B G) - n_B(kG) = p_m$

The implementation of elliptic curve algorithm is done over $GF(2^{163})$ for providing security of more than 128 bits.

VI. ROUTING THE DATA PACKETS:

Here in this section we describe the procedure of authentication data packets forwarded from the source node to the destination node, along the selected route, while checking for faulty links.

In DSR, the source route information is carried in each packet header.

A. FHC: Fixed Hash Chaining

An algorithmic description of the FHC for data packet transmission

1. A verification process takes place for egress of each node in routing path.
2. A counter sign pw_c will be used for data packet p_c that to be sent currently in sequence.
3. Packet p_c transmits data d_c.

4. An irreversible code t_c generated by applying a secure hash function $f_{(h)}$ on countersign pw_c.
5. Let pw_n as countersign for next packet p_n that is in sequence, which fallows the packet p_c
6. Packet p_n includes data d_n, and irreversible code t_n will be generated by hashing pw_n using $f_{(h)}$.
7. Hash Tag H_c will be generated by using secure hashing $f_{(h)}$ that uses d_n, t_n and pw_c as input.
8. p_c Is then transmitted that includes the H_c, d_c, t_c, password pw_p of packet p_p that sent before p_c in sequence to authenticate d_c.

Algorithm to authenticate sequence transmission of the packets:

Countersign $cs_{(p_c)}$ will be selected for p_c that includes d_c to be transmitted.

$$t_c = f_{(h)}(cs_{(p_c)})$$

Countersign $cs_{(p_n)}$ will be selected for p_n with data d_n to be transmitted in sequence,

$$t_n = f_{(h)}(cs_{(p_n)})$$

Apply $f_{(h)}$ to the d_n, t_n and $cs_{(p_c)}$ that creates authentication tag for p_n referred as $at_{(p_n)}$

$$at_{(p_n)} = f_{(h)}(< d_n, t_n, cs_{(p_c)} >);$$

Transmit p_c from a source node n_s to a destination node n_d through hops in path selected through optimal route selection strategy.

The currently transmit $cs_{(p_p)}$ of packet p_p that transmitted before p_c to authenticate d_c.

In the interest of route maintenance, every hop in rout contains a cache that maintains hop list describing the route selected using an optimal route selection model. We apply $f_{(h)}$ on cache of each hop of the route to verify the integrity of the hop list cached.

Architecture of the proposed protocol

Proposed model provides ting packet contains $at_{(p_n)}, d_c, t_c$ and a countersign

an authentication protocol for a wireless ad hoc network where packets are transmitted serially. By serially, we mean a current packet p_c is immediately preceded by a previous packet p_p, and followed immediately by a next packet p_n.

More particularly, during a route discovery phase, we provide secure route selection, i.e., a shortest intact route, that is, a route without any faulty links. During route maintenance phase, while packets are forwarded, we also detect faulty links based on a time out condition. Receiving an acknowledgement control packet signals successful delivery of a packet.

For packet authentication, we use $f_{(h)}$ described by Benjamin Arazi et al [21]. The hash function encodes a countersign to form a tag.

By $f_{(h)}$ we mean that the countersign cannot be decoded from the tag and the countersign is used only once, because part of its value lies in its publication after its use. We have adapted that protocol for use in an ad hoc network where multiple packets need to be sent sequentially. Therefore, if a number of packets are sent sequentially, the countersign needs to be refreshed each time. Thus, a single authentication is associated with a stream of future packets that is significant difference between proposed and existing hash chain techniques. The existing models require stream of future events. In addition, the countersign is used to authenticate p_c but not for future packets.

As an advantage over prior art asymmetric digital signature or secret countersigns do not need to be known ahead of time or distributed among the nodes after the system becomes operational. It should also be noted, that each countersign is used only one time, because the countersign is published to perform the authentication.

The $f_{(h)}$ as implemented by the proposal is ideal for serially communicating packets along a route in an ad hoc network, without requiring the nodes to establish shared secret countersigns beforehand.

The protocol includes the following steps. Select a random countersign cs_r. Form a tag t_r, $t_r = f_{(h)}(cs_r)$, Construct a message mac_r. Form a hash value $H_r = f_{(h)}(< mac_r, t_r, cs_r >)$, and make it public. Perform the act and reveal mac_r, t_r, cs_r to authenticate the act.

B. Data transmission and malicious hop detection

To send a packet m_i that is a part of data to be sent to destination node n_d, the source node n_s picks two counter signs cs_r, cs_{r+1} and fixes the time limit to receive either one of

packet delivery acknowledgement ack or a control packet mn_{ack} that acknowledges about malicious link in the route path. The source node sends message with the format $msg_i = \{m_i, f_{(h)}(cs_r), f_{(ds)}(m_i, f_{(h)}(cs_r)), f_{(h)}(m_{i+1}, f_{(h)}(cs_{r+1}), cs_r)\}$ to the n_h along the route.

Here $f_{(ds)}(m_i, f_{(h)}(cs_r))$ is a digital signature to verify $(m_i, f_{(h)}(cs_r)$ by intermediate hops of the route selected, so that every 'n_h' and 'n_d' can verify that $(m_i, f_{(h)}(cs_r))$ is valid and indeed generated by the claimed n_s.

Then each hop updates route table entry for source node S by recording $f_{(h)}(cs_r)$ as $hcs_r(n_s)$, $f_{(h)}(m_{i+1}, f_{(h)}(cs_{r+1}), cs_r)$ as $h_{e2}(n_s)$, which is used to authenticate an immediate fallowing message msg_{i+1} in sequence.

When sending the data packet m_{i+1}, the n_s selects another countersign cs_{r+2} and forwards the msg_{i+1} to the first hop of the selected path:

$$msg_{i+1} = \{m_{i+1}, f_{(h)}(cs_{r+1}), cs_r, f_{(h)}(m_{i+2}, f_{(h)}(cs_{r+2}), cs_{r+1})\}$$

Each node on the route calculates $f_{(h)}(cs_r)$ and compares with $hcs_r(n_s)$ that available in routing table, if results equal then cs_r will be authenticated as valid. The n_h then calculates $f_{(h)}(m_{i+1}, f_{(h)}(cs_{r+1}), cs_r)$, and compares with $h_{e2}(s_n)$ result is equivalent then claims the validity of $(m_{i+1}, f_{(h)}(cs_{r+1}))$. The node then updates its routing entry by recording $hrc_{r+1} = f_{(h)}(rc_{r+1})$ and $h_{(e2)}(n_s) = f_{(h)}(m_{(i+2)}, f_{(h)}(cs_{r+2}), r_{r+1})$, and forwards the data packet to the node along the route as specified in the header of the packet header.

During the packet sending process described earlier, if any of the checks fails, then the packet is dropped. If both checks succeed, then the node updates its routing entry associated with n_s. If the check at n_h, then either n_{h-1} or $f_{(h)}(m_{i+1}, f_{(h)}(cs_{r+1}), cs_r)$ in msg_i has been modified, or node n_{h-1} modified $f_{(h)}(m_{i+1}, f_{(h)}(cs_{r+1}), cs_r)$ in msg_{i+1}. In either case, the current hop node n_h drops the packet. Consequently, hop node n_{h-1} does not receive a valid ack after time out, and the node can report a malicious activity at (n_{h-1}, n_h) connection, or the hop node n_{h-2} reports about malicious activity between (n_{h-2}, n_{h-1}) to n_s. In either case, the fault link includes the malicious node n_{h-1}.

In our proposed model the authentication tag of each packet limited to two hashes and one countersign; while in the existing models required N authentication tags for a route with N hops. Therefore, our method has a lower communication and storage overhead.

The packet authentication process at n_d is identical to the authentication process at any intermediate hop n_h. If any of the checks fails, then the packet is dropped. If both checks succeed, the packet is delivered successfully, and schedules the 'ack' for transmission along the reverse of path of the route. The ack reflects the packet identification number i.

The destination node also appends an authentication tag to the ack message for the nodes on the reverse path. The authentication tag bears the same structure as the one generated by the source node. Specifically, when sending ack_i, for the packet 'm_i', the destination node randomly selects two countersigns cs_{re} and cs_{re+1}, and sends the following information:

$$ack_i, f_{(h)}(cs_{re}), f_{(ds)}(ack_i, f_{(h)}(ack_i)), f_{(h)}(ack_{i+1}, f_{(h)}(cs_{re+1}), cs_{re})$$

Similarly, $f_{(ds)}(ack_i, f_{(h)}(cs_{re}))$ is used to verify $(ack_i, f_{(h)}(cs_{re}))$ by each node along the reverse path of the route. When sending the acknowledgement for packet 'm_i', the destination selects a new countersign cs_{re+1} and forwards:

$$(ack_{i+1}, f_{(h)}(cs_{re+1}), cs_{re}, f_{(h)}(ack_{i+2} f_{(h)}(cs_{re+2}), cs_{re+1})).$$

.If the timeout at an intermediate node expires, then that node sends mn_{ack} with an identification number according to our hash function for authentication of the mn_{ack} by the upstream nodes. When a node receives the ack, the node verifies its authenticity and that a timeout is pending for the corresponding data packet. If the 'ack' is not authentic or a timeout is not pending, the node discards the ack. Otherwise; the node cancels the timeout and forwards the 'ack to the next node.

When a node receives mn_{ack}, it verifies its authenticity, and that a timeout is pending for the corresponding data packet, and that the link reported in the mn_{ack} is the first downstream to the node that generated mn_{ack}. If the mn_{ack} is not authentic, or a timeout is not pending, or the link is not the downstream to the node reporting 'mn_{ack}', then the node drops mn_{ack}. Otherwise, the node cancels the timeout and further forwards the mn_{ack} control packet. Upon receiving '

mn_{ack}', the source node deletes the link that connecting n_h referred in mn_{ack} and finds a new route. In this proposed model, the packets are always received as in the order they sent. This is because all packets are forwarded along the same route in DSR. In the case of congestion and buffering, the messages are stored in a first-in-first-out buffer according to the order that they are received.

The experiments were conducted using NS 2. We build a simulation network with hops under mobility and count of 50 to 200. The simulation parameters described in table 5. Authentication ensures that the buffer is properly allocated to valid packets. The simulation model aimed to compare ARIADNE [11] and FHC-NCTSR for route establishing phase, SEAD[10] and FHC-NCTSR for data transmission. The performance check of ARIADNE[11] and FHC-NCTS protocols carried out against to the threats listed below.

a) Rushing attack
b) Denial of service
c) Routing table modification
d) Tunneling

The protection against tunneling attack is the advantage of the NCTS-DSR over Ariadne.

TABLE5: SIMULATION PARAMETERS THAT WE CONSIDERED FOR EXPERIMENTS

Number of nodes Range	50 to 200
Dimensions of space	1500 m × 300 m
Nominal radio range	250 m
Source–destination pairs	20
Source data pattern (each)	4 packets/second
Application data payload size	512 bytes/packet
Total application data load range	128 to 512 kbps
Raw physical link bandwidth	2 Mbps
Initial ROUTE REQUEST timeout	2 seconds
Maximum ROUTE REQUEST timeout	40 seconds
Cache size	32 routes
Cache replacement policy	FIFO
Hash length	80 bits
certificate life time	2 sec

The metrics to verify the performance of the proposed protocol are

a) Data packet delivery ratio: It can be calculated as the ratio between the number of data packets that are sent by the source and the number of data packets that are received by the sink.

b) PACKET DELIVERY FRACTION: It is the ratio of data packets delivered to the destinations to those generated by the sources. The PDF tells about the performance of a protocol that how successfully the packets have been delivered. Higher the value gives the better results.

c) AVERAGE END TO END DELAY: Average end-to-end delay is an average end-to-end delay of data packets. Buffering during route discovery latency, queuing at interface queue, retransmission delays at the MAC and transfer times, may cause this delay. Once the time difference between packets sent and received was recorded, dividing the total time difference over the total number of CBR packets received gave the average end-to-end delay for the received packets. Lower the end to end delay better is the performance of the protocol.

d) Packet Loss: It is defined as the difference between the number of packets sent by the source and received by the sink. In our results we have calculated packet loss at network layer as well as MAC layer. The routing protocol forwards the packet to destination if a valid route is known, otherwise it is buffered until a route is available. There are two cases when a packet is dropped: the buffer is full when the packet needs to be buffered and the time exceeds the limit when packet has been buffered. Lower is the packet loss better is the performance of the protocol.

e) ROUTING OVERHEAD: Routing overhead has been calculated at the MAC layer which is defined as the ratio of total number of routing packets to data packets.

Figure 3(a) shows the Packet Delivery Ratio (PDR) for FHC-NCTSR, ARIADNE and SEAD. Based on these results it is evident that FHC-NCTSR recovers most of the PDR loss that observed in ARIADNE against to SEAD. The approximate PDR loss recovered by FHC-NCTSR over ARIADNE is 1.5%, which is an average of all pauses. The minimum individual recovery observed is 0.18% and maximum is 2.5%. Figure 3(b) indicates ARIADNE minimal advantage over FHC-NCTSR in Path optimality. FHC-NCTSR used average 0.019 hops longer than in ARIADNE because of the hop level certification validation process of the FHC-NCTSR that eliminates nodes with invalidate certificate. Here slight advantage of ARIADNE over FHC-NCTSR can be observable.

The packet delivery fraction (PDF) can be expressed as:

$$P' = \sum_{f=1}^{e} \frac{R_f}{N_f}$$

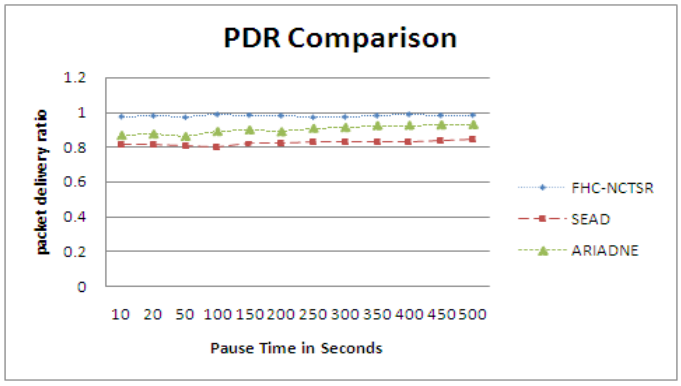
$$P = \frac{1}{c} * P'$$

- P is the fraction of successfully delivered packets,
- c is the total number of flow or connections,
- f is the unique flow id serving as index,
- R_f is the count of packets received from flow f
- N_f is the count of packets transmitted to flow f.

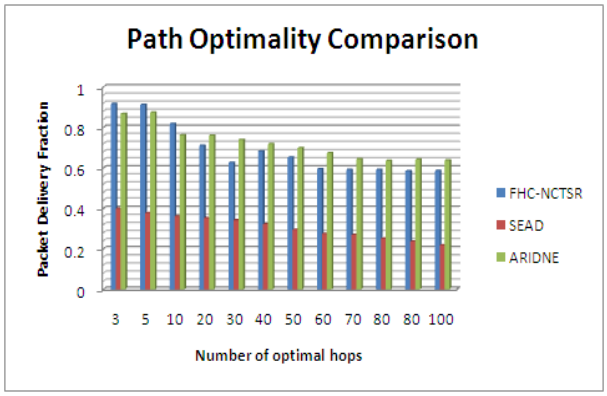
Figure 3(c) confirms that FHC-NCTSR is having fewer packets overhead when compared to ARIADNE. Due to stable paths with no compromised or victimized nodes determined by

FHC-NCTSR this advantage become possible. The Packet overhead observed in ARIADNE is average 5.29% more than packet overhead observed in FHC-NCTSR. The minimum and maximum packet overhead in ARIADNE over FHC-NCTSR observed is 3.61% and 7.29% respectively. It is quite evident from fig 3(c), that SEAD is not stable to handle the packet overhead, over a period of time the packet overhead is abnormal compared to other two protocols considered.

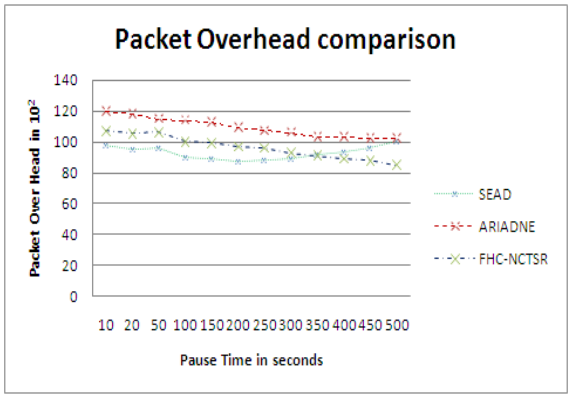
MAC load overhead is slightly more in FHC-NCTSR over ARIADNE. We can observe this in figure 3(d), which is because of additional control packet exchange in FHC-NCTSR for neighbor hop validation through certificate exchange. The average MAC load overhead in FHC-NCTSR over ARIADNE 1.64%. The minimum and maximum MAC load overhead observed is 0.81 and 3.24% respectively.



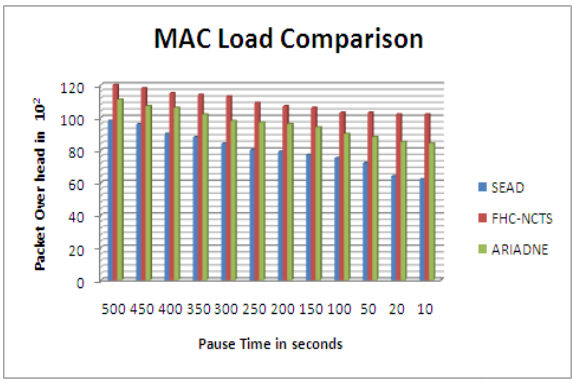
(a) Packet delivery ratio comparison using line chart



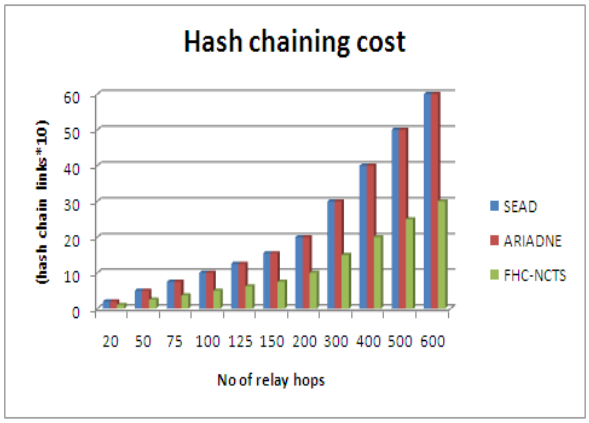
(b) Bar chart representation of Path optimality



(c) A line chart representation of Packet overhead comparison report



(d) Mac load comparison represented in bar chart format



(e) Hash chaining cost comparison report

In fig 3(e) we describe the performance of FHC-NCTSR over ARIADNE and SEAD in terms of Hash chain evaluation cost. Let λ be the cost threshold to evaluate each hash in hash chain.

We measure the Hash chain evaluation cost as

$$\frac{\sum_{i=1}^{z}\sum_{j=1}^{n}\lambda}{z}$$

, here z is number of nodes and n is number of hashes, as of the chaining concept of SEAD and ARIADNE z is equal to n but in FHC-NCTSR n always 2

VII. Conclusion

This paper was presented an evaluation of security protocols such as QoS-Guided Route Discovery [13], sQos[15], Ariadne [16] and CONFIDANT [17], which are based on reactive DSR approach, and describes their limitations and attacks against these protocols that can be subtle and difficult to discover by informal reasoning about the properties of the protocols.

The proposed a hybrid protocol FHC-NCTSR protocol applies digital signature exchange on the RREQ and RREP that they contribute the neighbors within 2 hops away from a node in computing them and a fixed hash chaining technique was used to achieve scalable data sending process. In route discovery phase, these digital signatures enable the protocol to

avoid malicious nodes from participating in routing and route discovery and also able to detect falsified routing messages and the responsible nodes.

And the fixed hash chaining in data transfer limits the computation cost and resource utilization.

VIII. REFERENCES

[1] P. Samar and S. B. Wicker, "On the behavior of communication links in a multi-hop mobile environment," in Frontiers in Distributed Sensor Networks, S. S. Iyengar and R. R. Brooks, Eds. Boca Raton, FL: CRC Press, 2004.

[2] C.E. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers," Proc. ACM SIGCOMM, vol. 24, no. 4, pp. 234–244, Oct. 1994.

[3] B. Bellur and R. G. Ogier, "A reliable, efficient topology broadcast protocol for dynamic networks," presented at the IEEE INFOCOM, Mar. 1999.

[4] S. Murthy and J. J. Garcia-Luna-Aceves, "An efficient routing protocol for wireless networks," MONET, vol. 1, no. 2, pp. 183–197, Oct. 1996.

[5] C. E. Perkins and E. M. Royer, "Ad hoc on-demand distance vector routing," presented at the IEEE WMCSA, New Orleans, LA, Feb. 1999.

[6] D. B. Johnson and D. A. Maltz, "Dynamic source routing in ad hoc wireless networking," in Mobile Computing, T. Imielinski and H. Korth, Eds. Boston, MA: Kluwer, 1996.

[7] V. D. Park and M. S. Corson, "Ahighly adaptive distributed routing algorithm for mobile wireless networks," presented at the IEEE INFOCOM, Kobe, Japan, Apr. 1997.

[8] Perlman: "Network Layer Protocols with Byzantine Robustness," Ph.D. thesis, MIT LCS TR-429, October 1998.

[9] Hauser: "Reducing the Cost of Security in Link State Routing," Symposium on Network and Distributed Systems Security, February 1997

[10] Hu: "SEAD: Secure efficient distance vector routing for mobile wireless ad hoc networks," Fourth IEEE Workshop on Mobile Computing Systems and Applications June 2002

[11] Hu: "Ariadne: A secure On-Demand Routing Protocol for Ad hoc Networks", MobiCom, September 2002

[12] Perrig: "Efficient and Secure Source Authentication for Multicast," Network and Distributed System Security Symposium, February 2001

[13] T. Clausen and P. Jacquet, "Optimized link state routing protocol (OLSR)," IETF, RFC 3626, Oct. 2003.

[14] M. Gerla, X. Hong, and G. Pei, "Landmark routing for large ad hoc wireless networks," presented at the IEEE GLOBECOM, San Francisco, CA, Nov. 2000.

[15]Z.J.Haas and M. R. Pearlman, "The performance of query control schemes for the zone routing protocol," IEEE/ACM Trans. Networking, vol. 9, pp. 427–438, Aug. 2001.

[16] Z. J. Haas, M. R. Pearlman, and P. Samar, "The bordercast resolution protocol (BRP) for ad hoc networks," IETF,MANET Internet Draft, July 2002.

[17] "The interzone routing protocol (IERP) for ad hoc networks," IETF, MANET Internet Draft, July 2002.

[18] "The intrazone routing protocol (IARP) for ad hoc networks," IETF, MANET Internet Draft, July 2002.

[19] "The zone routing protocol (ZRP) for ad hoc networks," IETF, MANET Internet Draft, July 2002.

[20] A. Iwata, C.-C. Chiang, G. Pei, M. Gerla, and T.-W. Chen, "Scalable routing strategies for ad hoc wireless networks," IEEE J. Select. Areas Commun., vol. 17,Aug. 1999.

[21] B. Liang and Z. J. Haas, "Hybrid routing in ad hoc networks with a dynamic virtual backbone," IEEE Trans. Wireless Commun., to be published.

[22] A. B. McDonald and T. Znati, "Predicting node proximity in Ad-Hoc networks: A least overhead adaptive model for electing stable routes," presented at the MobiHoc 2000, Boston, MA, Aug. 2000. , Mar. 1997