A New Secure Route Discovery Protocol for MANETs to Prevent Hidden Channel Attacks

Kavitha Ammayappan[1,2], Vinjamuri Narsimha Sastry[1], and Atul Negi[2]
(Corresponding author: Kavitha Ammayappan)

Institute for Development and Research in Banking Technology[1]
Castle Hills, Road No. 1, Masab Tank, Hyderabad 500 057, India. (E-mail: kavithaammayappan@gmail.com)
Department of Computer and Information Sciences, University of Hyderabad[2]
Hyderabad 500 046, India.
(Received Jan. 1, 2011; revised and accepted Apr. 6, 2011)

Abstract

In this paper, we propose a new secure route discovery protocol for MANETs that overcomes the vulnerabilities of Ariadne and EndairA, due to hidden channel attacks. It uses 'authentic neighborhood' for route discovery process which potentially protects hidden channels of routing control packets, besides ensuring authenticity and integrity of routing control messages at hop-by-hop level. This authentic neighborhood is augmented by a process of traceability which uses promiscuous mode of a node to detect, diagnose and isolate the adversarial nodes, that disrupt the route discovery process. We observe, from the comparative analysis of the proposed protocol with Ariadne and EndiarA, that the proposed protocol has a balance between security and computational overhead. Security analysis and formal verification, through extended BAN logic and AVISPA toolkit, shows that the proposed protocol is safe against the vulnerabilities identified in Ariadne and EndairA, due to unprotected hidden channels.

Keywords: Hidden channel attacks, plausible route, promiscuous mode, secure route discovery, tunnelling attack

1 Introduction

Several route discovery protocols, found in the literature [17, 20, 22, 23, 24], focus on efficiency and adaptability issues. Some routing protocols [12, 13, 14, 21, 26, 29] address security issues. Hu et al. [15] have proposed Aridane as a secure on demand routing protocol, using various cryptographic techniques. However, it has been proved insecure to hidden channel and replay attacks by Mike Burmester et al. [7]. Buttyan Vajda and Acs et al. [1, 2, 3, 8] have specifically addressed the security aspects of route discovery protocols in MANETs and suggested EndairA [3] protocol to avoid the vulnerabilities of Ariadne [15]. Hidden channel attack is a kind of worm hole attack which uses the fields carrying random values, in routing control packets. On the other hand, worm hole attack uses out-of-band resources, such as wire line, long range wireless transmission, or optical link to embed and convey signature information among adversarial nodes [19]. The outcome of hidden channel and worm-hole attacks result in the establishment of non-plausible routes, which shortens the actual route by removing the non adversarial intermediate nodes. In this manner, the vulnerability of Ariadne and EndairA, to hidden channel attacks, has been proved in [7]. In the proposals cited above, hidden channels of the routing control packets are uncovered and exploited to achieve attacks. Mike Burmester et al. [7] have also suggested the inclusion of traceability, for finding out plausible routes, during MANET route discovery process. We have defined traceability as a monitoring and identification mechanism in this paper which helps to find out plausible routes and avoid hidden channel attacks. In this paper, we propose a new secure route discovery protocol which overcomes the hidden channel attacks of Ariadne and EndairA, by employing authentic neighborhood and traceability mechanisms. In our view, the proposed protocol is novel and unique one to protect hidden channels and to prevent worm hole and hidden channel attacks, thereby ensuring plausible routes, during route discovery process in MANETs. Our proposed protocol, when compared with Ariadne and EndairA, has the following advantages:

1) it makes use of authentic neighborhood for route discovery process and thus, protects hidden channels;

2) it secures route request contents and path information in a hop-by-hop manner in an uncompromised way, by preserving hidden channels; and

3) it integrates authentication and key agreement among source and target nodes along route discovery process, by using Elliptic Curve Digital Signa-

ture Algorithm (ECDSA) [16, 28] and Elliptic Curve Diffie-Hellman (ECDH) Algorithm [11].

Our traceability technique helps to achieve plausible route discovery, besides identifying and avoiding adversarial nodes. Thus, it is suited to any wireless network in general and MANET in particular, which specifies privacy as a Quality of Service requirement.

The rest of the paper is organized as follows: In Section 2, we present the notation, assumptions, adversarial model, framework and algorithm of the proposed protocol. Section 3 presents the description of the proposed protocol. Section 4 presents security analysis of the proposed protocol under various attack scenarios. Formal verification proof using extended BAN and AVISPA toolkit are given in Section 5. Section 6 presents comparative analysis on computational overhead. Trade-off between security and computational overhead of proposed protocol with Ariadne and EndairA, are discussed in Section 7 and Section 8 concludes the paper.

2 Proposed Protocol

The objective of the proposed protocol is to secure a route discovery process, by implementing security mechanisms to protect hidden channels and prevent hidden channel attacks. Possible hidden channel attacks, in the route discovery process of Ariadne and EndairA, are of the following types: Active 1-2, Active 2-2 and an attack on the signature version of Ariadne, and tunneling attack [2, 3, 8, 10]. We suggest a mechanism to prevent these possible attacks, identify and avoid malicious nodes during route discovery process. And the mechanism consists of

- maintaining a monitor table as shown in Figure 4 by each of the nodes and it is filled with specific fields from the received RREQ and RREP packets;
- correctness of the route reply (RREP) packet propagation is monitored by verifying the routing information sent by the predecessor node with the stored information in its monitor table;
- verification is done with respect to the contents of the monitor table;
- information on the identified malicious node is propagated by broadcasting an alarm message;
- alarm acceptance message must be send by upstream nodes of the discovered path, excluding the alarm sender;
- based on the successful verification of both alarm and alarm acceptance messages, malicious node is removed from the authentic neighborhood of its neighbors;
- in case, RREP packet reaches the source node before the arrival of alarm and alarm acceptance messages, source immediately holds back or terminates the path based on the validity of the received alarm and alarm acceptance messages; and
- if no such alarm messages are reported, then it indicates that the path from source to target is plausible and secure with respect to various types of wormhole and hidden channel attacks.

In this section, we present notation, assumptions, adversarial model, framework and pseudo codes of the proposed protocol.

2.1 Notation

Table 1 is the notation used in our protocol.

Table 1: Notation used in our protocol

Notation	Description
RREP()	Route Request Packet
RREP()	Route Reply Packet
ECDH	Elliptic Curve Diffie-Hellman
ECDSA	Elliptic Curve Digital Signature Algorithm
ECDLP	Elliptic Curve Discrete Logarithm Problem
$\|$	Concatenation operator
α	Number of intermediate nodes except source and target
(S,T)	(Source, Target) pair of route discovery process
H( )	Secure oneway hash function
HMAC	Keyed message authentication code function
Msg	Represents the corresponding message flows between nodes
$Cert_X$	Certified Token of Node X
T_{RREQ}	Timestamp, RREQ() generation time
σ, σ'	Signature
Pub_X, Pri_X	Public and Private key of Node X
K_{XY}	Symmetric secret key generated between Node X and Y using ECDH
Sign()	Signature Generation Algorithm
Verify()	Signature Verification Algorithm
MAC	Message Authentication Code
2-hop ID	Predecessor of a current node which is in 2-hop distance through which route request packet has arrived
Nodelist	List of nodes via RREQ() packet has traversed

2.2 Assumptions

We assume that the base protocol is an on-demand route discovery protocol and all wireless links in the network are bidirectional. It is also assumed that all the nodes have enough computational power to perform ECC based functions. Each node maintains a list of authentic neighbors. And their relative positions are in reachable range. RREQ() packets are propagated via authentic neighbor nodes through unicast manner. Authentic neighbor nodes in the MANET share pairwise symmetric key with each other. Newly joining node can establish a symmetric key

with others, using ECDH by authenticating each others certified token. An authentic neighborhood (Figure 1) of a particular node in MANET is a collection of nodes and each node must be authenticated and established a symmetric key with that particular MANET node, using ECDSA and ECDH [11]. The neighborhood table of each such MANET node consists certified token and symmetric key of its authentic neighbors. RREQ packet propagation via authentic neighborhood ensures data origin authenticity and data integrity, of the received RREQ messages in a hop-by-hop level. This is highly required for MANET environment. Each node keeps a monitor table. This is filled during route discovery process in its format, as shown in Table 2. We assume that there is a secure way to update such details. Monitor table helps to identify maliciously behaving nodes, during route discovery phase.

Table 2: Structure of a proposed monitor table

(Source, Target)
Timestamp
2-hopID
Nodelist

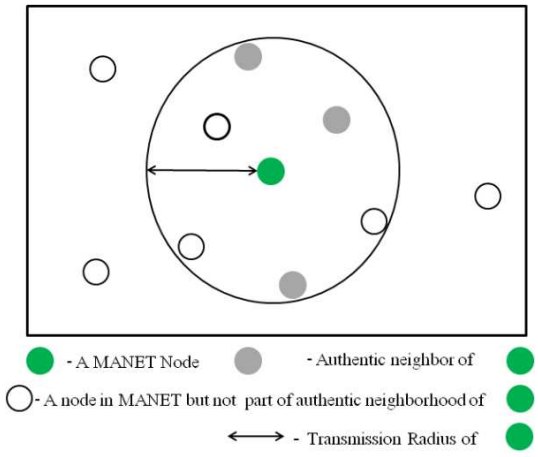

Figure 1: Authentic neighborhood

2.3 Adversarial Model

Adversarial model is essential to evaluate the design of the protocol, by modeling variety of known attacks. Evaluation of the proposed protocol, under such attacks, ensures its design robustness. We use the adversarial model shown, in Figure 2, to emulate attacks found in Ariadne and EndairA [2, 3, 10] and to evaluate our protocol design.

Analysis, focusing on Araidne [2, 8] and EndairA [3], reveals that the aim of attackers is to manipulate hidden channels such as nodelist, route request identifier and other fields which carry random values on routing control packets during route discovery process to establish non-plausible route. Therefore, attacks are exploited successfully by suppressing original RREQ() and RREP() packets, with fake ones by embedding appropriate MACs and digital signatures in hidden channels. Though the attacker succeeds in generating signature, using non-plausible *Nodelist*, yet it requires the plausible path to forward the RREP() and make the source node to accept the discovered non-plausible path, as a plausible one. Taking our protocol design into consideration, we confirm a node as malicious one when

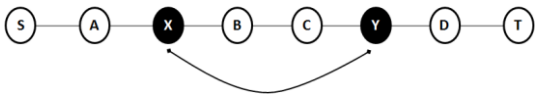

Figure 2: Adversarial model

- it violates the secure routing protocol specification in general and in specific
- it changes RREP() contents before forwarding the same to its downstream neighbor.

Therefore, in our adversarial model as in [18], we consider that there is, at least, one honest intermediate node between adversarial Nodes X and Y of a given route, such that it can be useful to emulate hidden channel attacks during route discovery process.

2.4 Framework of the Proposed Protocol

Proposed protocol framework is shown in Figure 3. Generally, threats have been exploited in MANET routing protocols both at route discovery and data forwarding phases. Our solution framework has 3 components as follows.

1) authentic neighborhood, to avoid impersonation and man-in-the-middle attacks;
2) authentication and key agreement(AKA) between source and target nodes, to protect data forwarding phase; and
3) tracing technique, to avoid and to identify malicious nodes which perform routing misbehaviour.

2.5 Algorithm of the Proposed Protocol

The proposed protocol operates in two phases namely, route request and route reply phases. During route request phase, source initiates the route discovery process. It unicasts RREQ() to its authentic one-hop neighbors as per the format given in Table 3. Any authentic one-hop neighbor node that receives the RREQ(), verifies the same for its source authenticity and data integrity, and, in turn, it forwards the RREQ() by including the newly computed keyed hash over the received route request contents to its

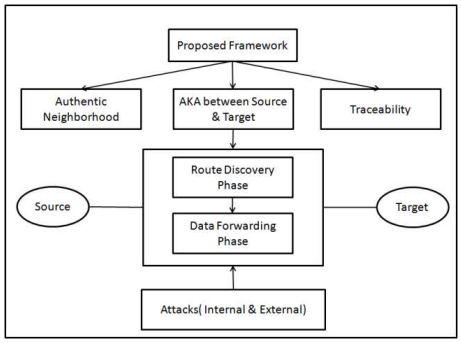

Figure 3: Framework of the proposed protocol

authentic one-hop neighbors. It also updates its monitor table entries for backward tracking, which will be used to monitor and identify maliciously behaving nodes, during route reply propagation. Likewise, RREQ() reaches the target node of the route discovery process. Then, it selects the secure and shortest route among the multiple ones.

Table 3: Format of the proposed RREQ packet

(Source, Target)
Timestamp
$Cert_{Source}$
(HMAC, HMAC)
Nodelist

Table 4: Format of the proposed RREP packet

(Source, Target)
Timestamp
$Cert_{Target}$
Nodelist
$HMAC_{Nodelist}$
$HMAC_{RREP}$

And during the route reply phase, the complete route is carried in the RREP() and its format is given in Table 4. Upon receiving the RREP() packet, receiving node verifies the keyed hash, computed on *Nodelist*, to ensure one-hop source authenticity and data integrity. Based on the validity of the received RREP(), it unicasts the updated RREP() to its downstream node. After unicasting the RREP(), it monitors the RREP(), forwarded by its downstream neighbor, under promiscuous mode and makes use of its monitor table to identify the maliciously behaving forwarding node.The description of the proposed protocol is given in Section 3.

Algorithms from 1 to 6 outline the pseudo code of the proposed protocol which is based on Figure 4. Algorithm 6 describe various cases of monitoring and identification process.

Table 5: Proposed alarm message format

(Source,Target)
Timestamp
Malicious Node ID
Modified Nodelist
σ
$Cert_{AlarmSender}$

Table 6: Proposed alarm acceptance message format

(Source,Target)
Timestamp
σ'
$Cert_{AlarmAcceptanceSender}$

Algorithm 1 Pseudo code: RREQ initialization by source Node S

Begin
Initiates Route Discovery through authentic neighborhood();
End

Algorithm 2 Pseudo code: RREQ() processing at intermediate node

Begin
Processes RREQ()
Propagates RREQ() through its authentic neighborhood();
End

Algorithm 3 Pseudo code: RREQ() processing at target node

Begin
Processes RREQ();
Generates and Unicasts RREP();
Switches to Monitoring and identification Routine();
End

Algorithm 4 Pseudo code: RREP() processing at intermediate node

Begin
Processes RREP();
Modifies and Unicasts RREP();
Switches to Monitoring and identification Routine();
End

Algorithm 5 Pseudo code: RREP() processing at source node

Begin
Processes RREP();
End

Algorithm 6 Pseudo code: Monitoring and identification routine

Begin

Case 1: Node which already sent RREP() to its downstream node and is listening in promiscuous mode.

If (it receives RREP() from its downstream node through promiscuous listening within a given time threshold)

Compares Nodelist and 2-hop ID obtained through promiscuous listening with its Monitor table content.

If (there is no semantic change) switches off its promiscuous mode.

Else, broadcasts alarm message as per format given in Table 5.

Else, broadcasts alarm message as per format given in Table 5.

Case 2: Upstream node which receives the alarm message

If (it is not a Target node)

Checks whether it has already processed RREP() corresponding to this alarm message

If not, it just broadcasts the alarm message

Else it, verifies alarm message for nodelist validation with its monitor table content.

If alarm message is valid, it broadcasts alarm acceptance message as per format given in Table 6

Else Discards alarm message

It broadcasts alarm acceptance message after successful verification of alarm message

Case 3: Downstream node which receives the alarm message

If (malicious node is in its authentic neighborhood)

It checks whether it has processed RREQ() and RREP() corresponding to this alarm message or not.

If (it has processed only RREQ()), it verifies alarm message and waits for an alarm acceptance message and holds back the RREP() packet without processing.

If (it has processed RREQ() and RREP()), it verifies alarm message and waits for an alarm acceptance message to confirm.

Else, broadcasts the alarm message.

Case 4: Downstream node which already processes alarm message and waits for an alarm acceptance message

If (verification of alarm acceptance message is successful), removes malicious node from authentic neighborhood and discards RREP() packet.

Else, it unicasts the RREP() and discards alarm and alarm acceptance messages.

End

3 Description of the Proposed Protocol

1) **Route Request Phase:**
In this section, proposed route discovery protocol is explained based on the path given in Figure 4. Source S initiates route discovery towards target T. Nodes (S,A,X,B,C,Y,D,T) are authentic neighbors to each other. We illustrate, diagrammatically, the route request packet propagation of the proposed algorithm in a step by step process through $Msg1$ to $Msg7$.

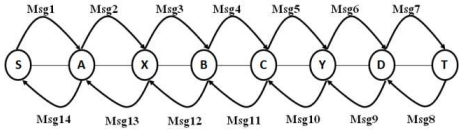

Figure 4: Sample path

- **Step 1. At Node S**
Node S sends *Msg1* to Node A and fills its monitor table as in Table 7.

Table 7: Monitor table of Node S

Source, Target	(S,T)
Timestamp	T_{RREQ}
2-hop ID	-
Nodelist	

$S \xrightarrow{Msg1} A$: $((S,T), T_{RREQ}, Cert_S, (HMAC_A), ())$ where $HMAC_A = H((S \parallel T \parallel T_{RREQ} \parallel Cert_A \parallel A), K_{SA})$ to A.

Node A checks whether Node S is in its authentic neighborhood or not. If it is not, it rejects the RREQ() message. It verifies $HMAC_A$ of the received message, by computing HMAC, using $(S, T, T_{RREQ}, Cert_S)$ parameters from Msg1, its own identity and symmetric key shared with Node S, from which it has received RREQ(). Thus, Node A compares $H((S \parallel T \parallel T_{RREQ} \parallel Cert_A \parallel A), K_{AS})$ with $HMAC_A$ of Msg1 for validity. If it is identical, Node A fills its monitor table as given in Table 8. *Nodelist* entry of the monitor table will be filled after receiving the respective RREP().

Table 8: Monitor table of Node A

Source, Target	(S,T)
Timestamp	T_{RREQ}
2-hop ID	S
Nodelist	

Further, Node A compares it target identity

of the route request packet with its own identity to generate RREP() if it is intended for itself. Node A finds $A \neq T$, therefore, Node A computes $HMAC_X = H((HMAC_A \parallel S \parallel X), K_{AX})$ and enters identity of Node S in *Nodelist* field of RREQ() and sends *Msg2* to Node X for propagation.

$$A \xrightarrow{Msg2} X : ((S,T), T_{RREQ}, Cert_S, (HMAC_A, HMAC_X), (S)).$$

- **Step 2. At Node X**
  On receiving *Msg2*, Node X checks whether Node A is in its authentic neighborhood or not. If it is not in its authentic neighborhood, it rejects *Msg2*. Else, it computes $H((HMAC_A \parallel S \parallel X), K_{XA})$, using received $HMAC_A$, latest entry of the *Nodelist* (i.e. Node S), its own identity and symmetric key shared with the Node A, from which it receives RREQ(). The latest entry of the *Nodelist* (i.e., Node S) indicates the 2-hop ID of Node X which is ensured by its predecessor node (i.e., Node A). During route reply propagation, Node X monitors whether Node A is forwarding to the same 2-hop Node S or not. In this way, each intermediate node, including target node of the route discovery process, records its 2-hop downstream neighbor node through which RREQ() propagates. Likewise, each intermediate node, including target node, monitors the correctness of its immediate downstream neighbor node's forwarding behavior.
  Node X compares $H((HMAC_A \parallel S \parallel X), K_{XA})$ with $HMAC_X$ of *Msg2* for correctness. If integrity check holds, Node X fills its monitor table as in Table 9. Else, Node X rejects *Msg2*.

Table 9: Monitor table of Node X

Source, Target	(S,T)
Timestamp	T_{RREQ}
2-hop ID	S
Nodelist	

Further, it finds $T \neq X$. Therefore, it computes keyed hash $HMAC_B = H((HMAC_X \parallel A \parallel B), K_{XB})$ and enters the identity of Node A, in *Nodelist* field of RREQ() and sends *Msg3* to Node B to propagate RREQ() packets.

$$X \xrightarrow{Msg3} B : (S, T, T_{RREQ}, Cert_S, (HMAC_X, HMAC_B), (S, A)).$$

- **Step 3. At Node B**
  On receiving *Msg3*, Node B checks whether Node X is in its authentic neighborhood or not. If it is not, it rejects *Msg3*. Else, it computes $H((HMAC_X \parallel A \parallel B), K_{BX})$ as mentioned in Step 2 and compares with $HMAC_B$ of *Msg3* for validity. If integrity check fails, Node B rejects *Msg3*. If the integrity check holds, Node B fills its monitor table as in Table 10.

Table 10: Monitor table of Node B

Source, Target	(S,T)
Timestamp	T_{RREQ}
2-hop ID	A
Nodelist	

Node B also compares the target identity of the RREQ() packet with its own identity, to format the RREP(), if it matches. Node B finds $T \neq B$ and Node B computes keyed hash $HMAC_C = H((HMAC_B \parallel X \parallel C), K_{BC})$, and enters the identity of Node X, in *Nodelist* field of RREQ(), and sends *Msg4* to Node C for RREQ() propagation.

$$B \xrightarrow{Msg4} C : ((S,T), T_{RREQ}, Cert_S, (HMAC_B, HMAC_C), (S, A, X)).$$

- **Step 4. At Node C**
  On receiving *Msg4*, Node C checks whether Node B is in its authentic neighborhood or not. If it is not, it rejects *Msg4*. Else, it computes $H((HMAC_B \parallel X \parallel C), K_{CB})$ as mentioned in Step 2 and compares with $HMAC_C$ of *Msg4* for validity. If integrity check fails, Node C rejects *Msg4*. If integrity check holds, Node C fills its monitor table as given in Table 11.

Table 11: Monitor table of Node C

Source, Target	(S,T)
Timestamp	T_{RREQ}
2-hop ID	X
Nodelist	

Further, Node C compares the target identity of the RREQ() with its own identity to format the RREP(), if it matches. It finds $C \neq T$, therefore, it computes $HMAC_Y = H((HMAC_C \parallel B \parallel Y), K_{CY})$ and enters the identity of Node B, in *Nodelist* field of RREQ(), and sends *Msg5* to Node Y for propagation.

$$C \xrightarrow{Msg5} Y : ((S,T), T_{RREQ}, Cert_S, (HMAC_C, HMAC_Y), (S, A, X, B)).$$

- **Step 5. At Node Y**
  On receiving *Msg5*, Node Y checks whether Node C is in its authentic neighborhood or not.

If it is not, it rejects *Msg5*. Else, it computes keyed hash $H((HMAC_C \parallel B \parallel Y), K_{YC})$ as mentioned in Step 2 and compares with $HMAC_Y$ of *Msg5* for validity. If integrity check fails, Node Y rejects *Msg5*. If integrity check holds, Node Y fills its monitor table as in Table 12.

Table 12: Monitor table of Node Y

Source, Target	(S,T)
Timestamp	T_{RREQ}
2-hop ID	B
Nodelist	

Further, Node Y finds $Y \neq T$, therefore, it computes keyed hash $HMAC_D = H((HMAC_Y \parallel C \parallel D), K_{YD})$ and enters the identity of Node C in *Nodelist* field of RREQ(), and sends *Msg6* to Node D for RREQ() propagation.

$$Y \xrightarrow{Msg6} D : ((S,T), T_{RREQ}, Cert_S, (HMAC_Y, HMAC_D), (S, A, X, B, C)).$$

- **Step 6. At Node D**
  On receiving *Msg6*, Node D checks whether Node Y is in its authentic neighborhood or not. If it is not, it rejects *Msg6*. Else, it computes $H((HMAC_Y \parallel C \parallel D), K_{DY})$ as mentioned in Step 2 and compares with $HMAC_D$ of *Msg6* for validity. If integrity check fails, Node D rejects *Msg6*. If integrity check holds, Node D fills its monitor Table 13 as shown below.

Table 13: Monitor table of Node D

Source, Target	(S,T)
Timestamp	T_{RREQ}
2-hop ID	C
Nodelist	

Further, Node D finds $D \neq T$ and computes keyed hash $HMAC_T = H((HMAC_D \parallel Y \parallel T), K_{DT})$, and enters the identity of Node Y, from which it receives the valid RREQ() in *Nodelist* field of RREQ(), and sends *Msg7* to Node T for RREQ() propagation.

$$D \xrightarrow{Msg7} T : ((S,T), T_{RREQ}, Cert_S, (HMAC_D, HMAC_T), (S, A, X, B, C, Y)).$$

2) **Route Reply Phase**
We describe here route reply propagation of the proposed protocol through *Msg8* to *Msg14*.

- **Step 7. At Node T**
  On receiving *Msg7*, Node T checks whether Node D is in its authentic neighborhood or not. If it is not, it rejects *Msg7*. Else, it computes it $H((HMAC_D \parallel Y \parallel T), K_{TD})$ as mentioned in Step 2 and compares with $HMAC_T$ of *Msg7* for validity. If integrity check fails, Node T rejects Msg7. If integrity check holds, Node T includes the identity of both Node D and its own identity in *Nodelist* of the received RREQ(), and fills *Nodelist* field of its monitor table as given in Table 14.

Table 14: Monitor table of Node T

Source, Target	(S,T)
Timestamp	T_{RREQ}
2-hop ID	Y
Nodelist	(S, A, X, B, C, Y, D, T)

Moreover, Node T finds its identity match with the target node of the route request packet. It verifies $Cert_S$ to validate the authenticity of source S, using the public key of trusted third party Pub_{TTP}, and computes secret key $K_{TS}(K_{ST}) = Pri_T \cdot Pub_S$ with respect to source S.

It also computes $HMAC_{Nodelist} = H(Nodelist, K_{TD})$ and $HMAC_{RREP} = H((S \parallel T \parallel T_{RREP} \parallel Nodelist), K_{TS})$. Then, Node T formats RREP() as per the format shown in Table 4 and sends *Msg8* to Node D.

$$T \xrightarrow{Msg8} D : ((S,T), T_{RREQ}, Cert_T, (S, A, X, B, C, Y, D, T), HMAC_{Nodelist}, HMAC_{RREP}).$$

Immediately after unicasting the RREP(), Node T enters into promiscuous mode to monitor Node D's activity.

- **Step 8. At Node D**
  On receiving *Msg8*, Node D verifies the validity of *Msg8* by checking the presence of RREP() parameters $((S,T), T_{RREQ})$, in its monitor table. If they are not present, it simply rejects the received route reply message. Else, it confirms the presence of its own identity in the *Nodelist* and checks its upstream node (T) and downstream (Y) neighbor node identities from *Nodelist* of *Msg8*. If there is a change in the upstream and downstream neighbour nodes identities, it simply rejects the RREP(). Else, it computes $H((S \parallel A \parallel X \parallel B \parallel C \parallel Y \parallel D \parallel T), K_{DT})$ and compares it with $HMAC_{Nodelist}$ of *Msg8*. If integrity check fails, it rejects RREP(). Else, Node D computes $H_{Nodelist} = H((Nodelist), K_{DY})$ and sends

Msg9 to Node Y.

$$D \xrightarrow{Msg9} Y : ((S,T), T_{RREQ}, Cert_T, (S,A,X,B,C,Y,D,T), HMAC_{Nodelist}, HMAC_{RREP}).$$

In the meantime, Node T listens Node D's forwarding behavior through promiscuous mode and checks its monitor table content (Table 14) against the received message *Msg9*. As per our protocol design, if Node T does not receive the forwarded RREP message from Node D, within a time threshold (t+$delta$), it flags Node D to be malicious, by broadcasting an alarm message.

If Node T receives RREP() within a given time threshold, at first, it checks 2-hop ID of Table 14 with destination ID of *Msg9* (i.e., Node Y) and both must be same to ensure the correct forwarding operation of Node D. Secondly, it computes hash over *Nodelist* of Table 14 and *Nodelist* of *Msg9* separately and compares them for validity. Both should be same to ensure *Nodelist* immutability and ensure Node D's correct forwarding behavior. Apart from these validations, Node T compares $((S,T), T_{RREQ})$ of *Msg9* with contents of same entries of Table 14 and should be identical. If any one of these validation fails, Node T flags Node D as malicious and broadcasts alarm message as per the format shown in Table 5.

In brief, as per our adversarial model, as an honest node, Node T judges Node D as non-malicious, due to the following: $((S,T), T_{RREQ})$ of *Msg9* matches with contents of Table 14. And the identity of downstream neighbor of Node D (Node Y) to which Node D sends the RREP() is identical to the identity of Node T's 2-hop ID (Node Y). And the hash over *Nodelist* of *Msg9* is also identical to hash over *Nodelist* stored in Node T's monitor table (Table 14). Therefore, Node T confirms that Node D forwards *Msg9* without any modification.

- **Step 9. At Node Y**
  Likewise, on receiving *Msg9* as a RREP(), Node Y evaluates it, as explained in Step 8 and sends *Msg10*, to its immediate downstream neighbor Node C.

$$Y \xrightarrow{Msg10} C : ((S,T), T_{RREQ}, Cert_T, (S,A,X,B,C,Y,D,T), HMAC_{Nodelist}, HMAC_{RREP}).$$

  where $HMAC_{Nodelist} = H(Nodelist, K_{YC})$.

- **Step 10. At Node C**
  On receiving *Msg10*, Node C processes it, as explained in Step 8 and sends *Msg11* as RREP() to Node B.

$$C \xrightarrow{Msg11} B : ((S,T), T_{RREQ}, Cert_T, (S,A,X,B,C,Y,D,T), HMAC_{Nodelist}, HMAC_{RREP}).$$

  where $HMAC_{Nodelist} = H(Nodelist, K_{CB})$.

- **Step 11. At Node B**
  After receiving *Msg11*, Node B evaluates it, as detailed in Step 8 and sends *Msg12* as RREP to Node X.

$$B \xrightarrow{Msg12} X : ((S,T), T_{RREQ}, Cert_T, (S,A,X,B,C,Y,D,T), HMAC_{Nodelist}, HMAC_{RREP}).$$

  where $HMAC_{Nodelist} = H(Nodelist, K_{BX})$.

- **Step 12. At Node X**
  Node X processes *Msg12*, as elaborated in Step 8 and sends *Msg13* as RREP() to Node A.

$$X \xrightarrow{Msg13} A : ((S,T), T_{RREQ}, Cert_T, (S,A,X,B,C,Y,D,T), HMAC_{Nodelist}, HMAC_{RREP}).$$

  where $HMAC_{Nodelist} = H(Nodelist, K_{XA})$.

- **Step 13. At Node A**
  On receiving *Msg13*, Node A processes it, as explained in Step 8 and sends *Msg14* to Node S.

$$A \xrightarrow{Msg14} S : ((S,T), T_{RREQ}, Cert_T, (S,A,X,B,C,Y,D,T), HMAC_{Nodelist}, HMAC_{RREP}).$$

  where $HMAC_{Nodelist} = H(Nodelist, K_{AS})$.

- **Step 14. At Node S**
  On receiving *Msg14* as RREP(), Node S checks the presence of $((S,T), T_{RREQ})$ of *Msg14* with its monitor Table 7, computes $H((S \parallel A \parallel X \parallel B \parallel C \parallel Y \parallel D \parallel T), K_{SA})$ and compares it with $HMAC_{Nodelist}$ of *Msg14*. If integrity check fails, it rejects RREP(). Else, it verifies $Cert_T$, using Pub_{TTP}. If verification fails, it rejects RREP(). Else, it computes symmetric secret key $K_{ST} = Pri_S \cdot Pub_T$ with respect to target Node T using ECDH. Then, it computes $H((S \parallel T \parallel T_{RREQ} \parallel Nodelist), K_{ST})$ and compares with $HMAC_{RREP}$ of *Msg14*. If integrity check fails, Node S rejects RREP(). Else, it confirms the received RREP() as valid and contains plausible route. Further data transmission

between Nodes S and T will be encrypted, using symmetric key $K_{ST}(K_{TS})$ as both computed the same at its end.

3) **Traceability - Monitoring and Identification Process**
This process is part of route request and reply propagation of our proposed design. In this process, target node generates the route reply packet and sends it via reverse of discovered path towards source node. After sending the RREP(), sender node of the RREP() monitors the forwarding nature of its downstream neighbor node through promiscuous mode. It identifies the malicious nature of its downstream neighbor node, examining the 2-hop ID and *Nodelist* field of the forwarded RREP(), as explained in Section 3. Based on the inference, the sender node decides the malicious behavior and broadcasts an alarm message. In order to avoid intentional rumors of maliciousness, we have added here alarm acceptance concept. Using alarm acceptance message, upstream neighbor nodes that already process RREP() endorse the validity of the alarm message. In this way, intentional rumors are avoided.

4 Attack Scenarios and Security Analysis of the Proposed Protocol with respect to Ariadne and EndairA

4.1 Analysis on the possibilities of exploiting hidden channels of Routing Control Packets used in the Proposed Protocol

In this section, we analyze the various possibilities of using hidden channels of routing control packets of the proposed protocol for worm-hole generation, during route discovery process. Route request identifier, *Nodelist* and fields which contain random values are generally exploited as hidden channels of routing control packets, for achieving hidden channel attacks. We present, here, our protocol's resilience design in preventing hidden channels of routing control packets from embedding signatures in detail.

- **Route Request Identifier:** In our protocol, the purpose of the route request identifier is replaced with the combination of (source, destination) pair and timestamp. Because they are unique enough to differentiate various route requests. Hence, they cannot be used to embed random signature information for achieving hidden channel attacks.

- **Nodelist:** To achieve hidden channel attacks, adversary embeds signature information on *Nodelist* field of the RREP(). As per our protocol design, integrity of *Nodelist* field of RREP() is verified by each intermediate node. Forwarded RREP() is also monitored by its upstream neighbor node, during route reply propagation. Therefore, modification, in this field, triggers upstream neighbor node to send an alarm message, as per the adversarial model shown in Figure 2. Hence, it cannot be used as a hidden channel.

- **HMAC:** As per our protocol, HMAC fields cannot be exploited as hidden channels, since they are verifiable by the receiving node, during RREQ() and RREP() propagation, which is described in Section 3.

- **Embedding Signature Information through RREQ():** We examine the scenario as per Figure 2, wherein adversarial Node Y may generate a RREQ() towards adversarial Node X, so as to embed and transmit signature information to achieve various types of worm-hole attacks.

  There are two possibilities in this scenario:

  1) In the first possibility, we assume that adversarial Node Y generates fake RREQ() towards adversarial Node X to embed signatures through hidden channels of the fake RREQ(). Adversary Y manipulates *Nodelist* and makes the target node to sign on non-plausible route during RREQ() propagation. RREP(), which contains non-plausible route, cannot be routed via nodes C-B as per the proposed design. Therefore, adversarial Node Y generates fake RREQ() towards adversary X to embed the signature generated on non-plausible route. The generated RREQ() looks as shown below.

$$Y \rightarrow C : ((Y, X), T_{RREQ}, Cert_Y, (HMAC_C, ()), ()). \quad (1)$$

  In the above message, hidden channels are 4th and 5th fields. 5th field of Equation (1) is empty, as RREQ() is generated by Node Y and nodes Y and C are in one-hop distance. Therefore, Node Y cannot embed any information in it. And due to the same reason, 4th field of Equation (1) has only one HMAC entry and it is verifiable at Node C as explained in Section 3. If Node Y embeds signature, to achieve worm-hole attack in 4th field, HMAC verification will fail at Node C and it discards RREQ() generated by Node Y as per our protocol.

  2) In the second possibility, we assume that Node Y generates a RREQ(), to embed fake signatures in its HMAC and *Nodelist* fields and claims that it has originated from remote node. In this case, the generated RREQ() looks as shown below.

$$Y \rightarrow C : (((remotenode, X), T_{RREQ}, Cert_{remotenode}, (HMAC_Y, HMAC_C), (signatures, \cdots))). \quad (2)$$

Let us consider the detailed description of Section 3, with respect to Figure 4, and the adversarial nature of Figure 5 for analyzing this particular attack possibility. Adversarial Nodes X and X(Y) of Figure 5 refer two compromised nodes but they use same identity and the details of this attack may be seen from G [3].

In the *Active 1-2 attack*, both adversarial Nodes X and X(Y) jointly forced the target node to sign on *Nodelsit* (S,A,X,X(Y),D,T) as a valid path, instead of *Nodelsit* (S,A,X,B,C,X(Y),D,T). Target Node T generates RREP() and it reaches Node D without rejection. In order to achieve the worm-hole attack, Node X(Y) needs to include Nodes B-C in the *Nodelist* so as to enable Nodes B and C to forward RREP() towards S. As per our protocol, *Nodelist* modification to be done by Node X(Y) in RREP() will be identified by Node D, as a routing misbehavior through its promiscuous listening.

In the above situation, in order to achieve the attack, Node X(Y) embeds signature information in *Nodelist* field of newly generated RREQ() as shown in Equation (2) by claiming that it has originated from remote node. Therefore, Node X(Y) holds the original RREP() without forwarding it to its downstream neighbor C. As per our adversarial model, this behavior is modeled as malicious and observed by Node D through promiscuous mode.

Subsequently, Node D broadcasts Node X(Y)'s malicious behavior by broadcasting alarm message. Thus from the analysis based on our proposed protocol, it is ensured that fake RREQ() cannot be exploited as hidden channels for achieving worm-hole attacks which are elaborated further in Subsection 4.2.

As an alternative way to achieve the worm-hole attacks, Node X(Y) encapsulates signature information through data packets and sends to other end adversary X. After sending information through data packets, Node X(Y) sends the original RREP() to Node C to be considered as non-malicious. In the meantime, fake route information reaches adversary X via dataplane and it subsequently, unicasts non-plausible route to source S. On the other side, with respect to the attacker model shown in Figure 2, node X receives the original RREP() and drops the same without forwarding further, in order to achieve the attack. As per our protocol, as an upstream neighbor node, Node B monitors the non-forwarding nature of adversary X and announces the same as misbehavior, by broadcasting an alarm message. Therefore, as per our protocol, hidden channels of the routing control packets and dataplane transmission cannot be exploited, for achieving attacks through concurrent route request messages.

4.2 Security Analysis

This section presents possible attack scenarios in *Ariadne* and *EndairA*, during route discovery process and presents results of security analysis of our protocol.

4.2.1 Active 1-2 Attack

This attack has been realized in a network configuration as shown in Figure 5 and it is assumed that shaded nodes are malicious. Detailed description of this attack is found in [3]. Here, we explain the resilience design of our protocol, which prevents the use of hidden channels of routing control packets in achieving this attack.

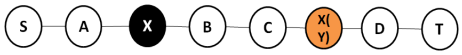

Figure 5: Active 1-2 attack model

This attack can be realized in two ways, by embedding signatures in hidden channels of routing control packets as follows:

- Embedding signatures in the original routing control packets
  Let us consider the message flows from *Msg1* to *Msg5*, described in Section 3, for analysis. To achieve this attack, Node X(Y) (shown in red) removes Nodes B-C from the actual *Nodelist* entries of *Msg6* and sends $Msg6'$ to Node D, instead of Msg6 with the appropriately computed HMACs.

$$X \xrightarrow{Msg6'} D : ((S,T), T_{RREQ}, Cert_S, (HMAC_Y, HMAC_D), (S,A,X)).$$

  After receiving *Msg6'* from Node X, Node D processes the same, as illustrated in Step 2 of Section 3 and caches the required information in its monitor table as shown in Table 15 except *Nodelist* field.

Table 15: Node D's monitor table

Source, Target	(S,T)
Timestamp	T_{RREQ}
2-hop ID	X
Nodelist	(S,A,X,X(Y),D,T)

Then, Node D sends *Msg7'* to Node T.

$$D \xrightarrow{Msg7'} T : ((S,T), T_{RREQ}, Cert_S, (HMAC_D, HMAC_T), (S, A, X, X(Y)).$$

As a target node, Node T processes *Msg7'* and caches relevant information in its monitor table, as shown in Table 16. Node T forms *Msg8'* and sends the same, as a RREP() to Node D.

$$T \xrightarrow{Msg8'} D : ((S,T), T_{RREQ}, Cert_T, (S, A, X, X(Y), D, T), HMAC_{Nodelist}, HMAC_{RREP}).$$

After validating $Msg8'$, Node D fills *Nodelist* column of Table 15 with *Nodelist* value of *Msg8'*. Later Node D sends *Msg9'* to Node X(Y).

$$D \xrightarrow{Msg9'} X(Y) : ((S,T), T_{RREQ}, Cert_T, (S, A, X, X(Y), D, T), HMAC_{Nodelist}, HMAC_{RREP}).$$

Node X(Y) (shown in red) being a malicious one, alters *Nodelist* of *Msg9'* by adding B-C with *Nodelsit* $(S, A, X, X(Y), D, T)$ so as to route the RREP() packet via Nodes B-C link, to achieve active 1-2 attack. Therefore, Node X transmits *Msg10'* to Node C.

$$X(Y) \xrightarrow{Msg10'} C : ((S,T), T_{RREQ}, Cert_T, (S, A, X, B, C, X(Y), D, T), HMAC_{Nodelist}, HMAC_{RREP}).$$

As per our protocol design and adversarial model, Node D overhears *Msg10'* and compares the same with the contents of Table 15. Specifically, it compares the value of 2-hop ID stored in Table 15 against destination ID of *Msg10'* (i.e., node C) and finds a mismatch. Since, Node X(Y) (shown in red) sends to Node C instead of sending Node X (shown in black). Node D, again, compares the value of *Nodelist* captured from *Msg10'*, against the *Nodelist* stored in Table 15, by computing H(Nodelist) and, in this way, the check fails due to change of *Nodelist*.

Based on the promiscuous listening, Node D ascertains the differences in *Nodelist* and 2-hop ID at the time of Node X(Y) (shown in red) transmits *Msg10'*. Therefore, Node D ensures that Node X(Y) (shown in red) transmits the modified RREP() packet to Node C, instead of Node X (shown in black) via backtracking with the help of Table 15. After finding this discrepancy, Node D computes signature $\sigma = Sign(H(S, A, X, B, C, X(Y), D, T), Pri_D)$ and formats alarm message, as given in Table 17 and broadcasts the same to flood Node X(Y)'s (shown in red) malicious activity in the network.

Table 16: Node T's monitor table

Source, Target	(S,T)
Timestamp	T_{RREQ}
2-hop ID	X
Nodelist	(S,A,X,X(Y),D,T)

Table 17: Alarm message sent by Node D

(Source,Target)	(S,T)
Timestamp	T_{RREQ}
Malicious Node ID	X
Modified Nodelist	H(S,A,X,B,C,X(Y),D,T)
Signature	σ
$Cert_{AlarmSender}$	$Cert_D$

Nodes C, B, X, A, S and T including Node X(Y) (shown in red) receive alarm message sent by Node D, since it is a broadcast message. Node T performs Verify $(H(Nodelist), Pub_D)$ using ECDSA by computing hash over *Nodelist*, which is present in Table 16. Pub_D of Node D is obtained from $Cert_D$. This verification fails due to *Nodelist* mismatch. Because *Nodelist* of $Table\ 16 \neq$ *Nodelist* of alarm message is shown in Table 17. Thus, Node T confirms Node X(Y)'s (shown in red) malicious activity, as it has already processed RREP() message. Subsequently, it computes $\sigma' = Sign(H(S, A, X, X(Y), D, T), Pri_T)$ by computing hash over *Modified Nodelist* stored in Table 17 and formats alarm acceptance message as shown in Table 18 and broadcasts the same.

Table 18: Alarm acceptance message sent by Node T

(Source,Target)	(S,T)
Timestamp	T_{RREQ}
Signature	σ'
$Cert_{AlarmAcceptanceSender}$	$Cert_T$

Node C compares $H(Nodelist)$ of *Msg10'* with $H(ModifiedNodelist)$ of alarm message and their integrity check fails. Also it verifies alarm message by performing $Verify(\sigma, H(ModifiedNodelist), Pub_D)$ and the verification holds true. On receiving alarm acceptance message from node T, node C performs $Verify(\sigma', H(ModifiedNodelist), Pub_T)$. It uses the *Modified Nodelist* of alarm message and it succeeds. Therefore, node C confirms the malicious activity of node X(Y) and removes it from its authentic neighbor table and discards the RREP() packet. Thus non-plausible route discovery is prevented in the proposed protocol by protecting hidden channels.

- Embedding Signature in remote RREQ()
  Let us assume that Node X(Y) (shown in red) holds back the original RREP()(*Msg9'*) received

from Node D and embeds signature ($HMAC_{RREP} = H(S \parallel T \parallel Cert_T \parallel T_{RREQ} \parallel Nodelist, K_{TS})$ in the *Nodelist* field of new RREQ() packet generated at Node X(Y) (shown in red) and destined to Node X (show in black). But Node X(Y) (shown in red) claims that RREQ() has been originated at a remote node so that it embeds signature in RREQ() *Nodelist* field.

Once, Node X (shown in black) gets the signature, generated on non-plausible route $(S, A, X, X(Y), D, T)$, it will be conveyed to source S as valid path and hence this attack is possible. As per our protocol described in Section 4.1, adversary cannot use hidden channels of RREQ() either originated at adversarial Node X(Y) or claimed as originated from remote node to embed fake signatures. Thus, proposed protocol design prevents the use of hidden channels of routing control packets to achieve this attack.

4.2.2 Active 2-2 Attack

This attack has been realized in a network configuration as shown in Figure 6. Detailed description of this attack can be found in [3]. Adversarial Nodes X and Y of Figure 6 jointly involved to achieve this attack. Here, we examine this attack possibility scenario of our proposed protocol. We consider specific message flows described in Section 3 for analysis.

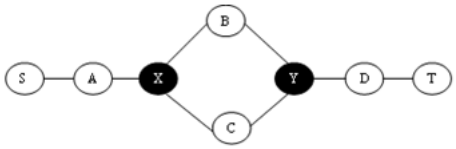

Figure 6: Active 2-2 attack model

The RREQ() propagation for the network configuration, shown in Figure 6, is identical to message flows described in Section 3 except few changes. In order to achieve this attack, in our proposed protocol, Node Y alters *Msg6* into *Msg6'* with appropriately computed HMACs and sends the same to Node D.

$$Y \xrightarrow{Msg6'} D : ((S,T), T_{RREQ}, Cert_S, (HMAC_Y, HMAC_D), (S, A, X)).$$

where $HMAC_D = H((HMAC_Y \parallel X \parallel D), K_{YD})$.

It verifies *Msg6'* and stores the relevant information in its monitor table which is shown in Table 19.

Based on the received *Msg6'*, Node D generates *Msg7'* and sends to Node T as RREQ().

$$D \xrightarrow{Msg7'} T : ((S,T), T_{RREQ}, Cert_S, (HMAC_D, HMAC_T), (S, A, X, Y)).$$

where $HMAC_T = H((HMAC_D \parallel Y \parallel T), K_{DT})$.

Table 19: Node D's monitor table in Active 2-2 attack

Source, Target	(S,T)
Timestamp	T_{RREQ}
2-hop ID	X
Nodelist	(S,A,X)

Node T verifies *Msg7'* and stores the relevant information in its monitor Table 20 and forms the RREP() by signing (S,A,X,Y,D,T) as a *Nodelist*.

Table 20: Node T's monitor table in Active 2-2 attack

Source, Target	(S,T)
Timestamp	T_{RREQ}
2-hop ID	Y
Nodelist	(S,A,X,Y,D,T)

Therefore, Node T generates *Msg8'* and sends the same as RREP() to Node D, where $HMAC_{Nodelist} = H((S||A||X||Y||D||T), K_{TD})$.

$$T \xrightarrow{Msg8'} D : ((S,T), T_{RREQ}, Cert_T, (S, A, X, Y, D, T), HMAC_{Nodelist}, HMAC_{RREP}).$$

Node D validates *Msg8'* and replaces $HMAC_{Nodelist}$ of *Msg8'* with $HMAC_{Nodelist} = H((S \parallel A \parallel X \parallel Y \parallel D \parallel T), K_{DY})$ sends it to Node Y as *Msg9'*.

$$D \xrightarrow{Msg9'} Y : ((S,T), T_{RREQ}, Cert_T, (S, A, X, Y, D, T), HMAC_{Nodelist}, HMAC_{RREP}).$$

Node Y does not send *Msg9'* as it is either to Node B or to Node C. Hence, adversarial Node Y modifies the *Nodelist* as either (S,A,X,B,Y,D,T) or (S,A,X,C,Y,D,T) to enable either Node B or Node C to forward RREP() through them in order to achieve active 2-2 attack. Subsequently, Node Y sends either *Msg10'* as a RREP() to Node C, where $HMAC_{Nodelist} = H((S||A||X||C||Y||D||T), K_{YC})$.

$$Y \xrightarrow{Msg10'} C : ((S,T), T_{RREQ}, Cert_T, (S, A, X, C, Y, D, T), HMAC_{Nodelist}, HMAC_{RREP}).$$

or *Msg11'* as a RREP() to Node B, where $HMAC_{Nodelist} = H((S||A||X||C||Y||D||T), K_{YB})$.

$$Y \xrightarrow{Msg11'} B : ((S,T), T_{RREQ}, Cert_T, (S, A, X, B, Y, D, T), HMAC_{Nodelist}, HMAC_{RREP}).$$

As per our protocol design, Node D listens the messages, forwarded by Node Y via promiscuous mode. Thus, Node D has either *Msg10'* or *Msg11'* transmitted by

Table 21: RREQ propagation as per Figure 7

$S \xrightarrow{RREQ1} V$:$((S,D), T_{RREQ}, Cert_S, HMAC_V, ())$
$V \xrightarrow{RREQ2} A$:$((S,D), T_{RREQ}, Cert_S, HMAC_v, HMAC_A, \{S\})$
$A \xrightarrow{RREQ3} D$:$((S,D), T_{RREQ}, Cert_S, HMAC_A, HMAC_D, \{S,V\})$
$V \xrightarrow{RREQ4} W$:$((S,D), T_{RREQ}, Cert_S, HMAC_V, HMAC_W, \{S\})$
$W \xrightarrow{RREQ5} X$:$((S,D), T_{RREQ}, Cert_S, HMAC_W, HMAC_X, \{S,V\})$
$X \xrightarrow{RREQ6} A$:$((S,D), T_{RREQ}, Cert_S, HMAC_X, HMAC_A, \{S,V,W\})$
$A \xrightarrow{RREQ7} D$:$((S,D), T_{RREQ}, Cert_S, HMAC_A, HMAC_D, \{S,V,W,X\})$

Table 22: RREP propagation as per Figure 7

$D \xrightarrow{RREP1} A$:$((S,D), T_{RREQ}, Cert_D, (HMAC_{Nodelist}, HMAC_{REPP}), (S,V,W,X,A,D))$
$A \xrightarrow{RREP2} X$:$((S,D), T_{RREQ}, Cert_D, (HMAC_{Nodelist}, HMAC_{RREP}), (S,V,W,X,A,D))$
$X \xrightarrow{RREP3} W$:$((S,D), T_{RREQ}, Cert_D, (HMAC_{Nodelist}, HMAC_{RREP}), (S,V,W,X,A,D))$
$W \xrightarrow{RREP4} V$:$((S,D), T_{RREQ}, Cert_D, (HMAC_{Nodelist}, HMAC_{RREP}), (S,V,W,X,A,D))$
$W \xrightarrow{RREP5} X$:$((S,D), T_{RREQ}, Cert_D, (HMAC_{Nodelist}, HMAC_{RREP}), (S,V,W,X,A,D))$

Node Y. It compares $H(Nodelist)$ and 2-hop ID values of either *Msg10'* or *Msg11'* with the contents of Table 19. The comparison fails due to discrepancy in *Nodelist* and 2-hop ID values.

As a result of this, Node D broadcasts an alarm message as per the format shown in Table 5 where the value of signature is either $\sigma = Sign(H(S \parallel A \parallel X \parallel C \parallel Y \parallel D \parallel T), Pri_D)$ or $\sigma = Sign(H(S \parallel A \parallel X \parallel B \parallel Y \parallel D \parallel T), Pri_D)$. This depends on *Msg10'* or *Msg11'*. Nodes T, C, B, X, A and Y receives the alarm message due to its broadcast nature. Among these nodes, Node T is an upstream node and sends alarm acceptance message, after validating the alarm message as explained in Section 4.2.1. On receiving alarm message, Node C or B do not process the RREP() message sent, by Node Y, until it receives alarm acceptance message from Node T. In this manner, either Node C or B which receives the alarm message confirms Node Y's malicious activity after validating alarm acceptance message. Rest of the nodes, which receive alarm and alarm acceptance messages, process them, according to the algorithm given in Section 2.5.

If the sender of an alarm message is the target node of a route discovery process, then nodes which receive alarm message do not expect alarm acceptance message, as there are no upstream neighbors to send alarm acceptance message.

If there is no upstream neighbor found after Node D from the *Nodelist*, then Node C discards the RREP(), based on the reception of alarm message and its validity. Nodes, that have Node Y as their authentic neighbor, remove Node Y from their authentic neighborhood table, based on the successful validation of alarm and/or alarm acceptance messages. As per our security analysis, Node Y cannot make use of either RREQ() originated at its end or RREQ() claimed as originated by remote node, to embed fake signatures in their hidden channels to achieve non-plausible route discovery. Therefore, as per our protocol design and the adversarial model under consideration, hidden channels of routing control packets cannot be used to achieve Active 2-2 attack.

4.2.3 Attack in the Signature Version of Ariadne

This attack has been realized in a network configuration as shown in Figure 7. Detailed description of this attack can be found in [8]. We analyze, here, the chances of occurring this particular attack in our proposed protocol. As per the given network topology as in Figure 7, actual RREQ() and RREP() message flows based on our proposed protocol, are given in Tables 21 and 22.

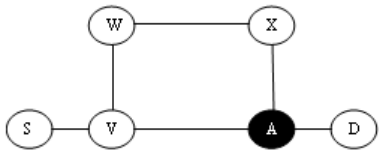

Figure 7: Ariadne's signature version attack model

In order to achieve this attack, adversary A of Figure 7 sends $RREQ7'$ to Node D, instead of RREQ7 as shown in Table 21.

$$A \xrightarrow{RREQ7'} D : (S, D, T_{RREQ}, Cert_S, (HMAC_A, HMAC_D), (S, V, W)).$$

Node D verifies *RREQ7'* and stores the relevant credentials in monitor Table 23.

Based on the received *RREQ7'* Node D generates *RREP1'* which contains non-plausible path (S,V,W,A,D).

Table 23: Monitor table of Node D in the attack of signature version of Ariadne

Source, Target	(S,T)
Timestamp	T_{RREQ}
2-hop ID	W
Nodelist	(S,V,W,A,D)

Node D sends *RREP1'* to Node A.

$$D \xrightarrow{RREP1'} A : ((S,D), T_{RREQ}, Cert_T, (S,V,W,A,D), HMAC_{Nodelist}, HMAC_{RREP}).$$

In brief, the success of this attack lies in convincing Node S to accept the non-plausible path (S-V-W-A-D) as a plausible one. In order to achieve this attack, adversary A needs to send $RREP1'$ to node V in the name of node W. To perform this, adversary A, as per our protocol design, needs to compute $HMAC_{Nodelist} = (H(S,V,W,A,D), K_{WV})$. But it cannot compute $HMAC_{Nodelist}$. Because it is not aware of symmetric key K_{WV} and its security is based on ECDLP. Computation of symmetric key requires the knowledge of either Pri_W or Pri_V. But obtaining these private keys from their respective public keys is equivalent to solving the ECDLP.

Thus, adversary A cannot succeed in achieving this attack, as per our protocol design. On the other hand, we assume that adversary succeeds in computing $HMAC_{Nodelist}$ and sends *RREP1'* to node V in the name of node W, and simultaneously this malfunction is identified by upstream Node D via proposed monitoring and identification process. Thus, our protocol effectively prevents this attack.

4.2.4 Hidden Channel Attack in EndairA

As per [7], hidden channel attack is discovered in *EndairA*, by exploiting RREQ identifier, *Nodelist* and other fields which carry random values. Tunneling attack [10] is exploited in *EndairA*, with the help of two or more adversarial nodes. The nature of this attack is similar to worm-hole attack. Based on the analysis described in Sections 4.1 and 4.2, hidden channels of routing control packets cannot be exploited in our protocol for achieving hidden channel and tunneling attacks, due to flexibility of HMAC and *Nodelist* verification, by the receiving node, during RREQ() and RREP() propagation. Thus, the proposed protocol is resilient to hidden channel and tunneling attacks. Apart from this, our proposed protocol is resilient against replay attack, due to the usage of *timestamp* and impersonation, using certificate.

5 Formal Verification of the Proposed Protocol Using Extended BAN Logic and AVISPA

5.1 Proof Using Extended BAN Logic

BAN logic is an important formal tool to analyze the correctness of an authentication protocol [6]. Extended BAN logic incorporates additional logic and constructs, for public key cryptography [27]. Hence, we apply extended BAN logic to formally verify our protocol, as it integrates end-to-end authentication and key agreement with secure route discovery process.

To avoid subtle design flaws, correctness of the proposed protocol is verified in terms of

- achieving mutual authentication and key agreement between source and target nodes, during the route discovery process;
- validating the integrity of the received route request and route reply messages among one-hop neighbors; and
- ensuring the data origin authenticity and integrity of the route reply packet at source.

BAN logic constructs and inference rules, used to prove our protocol, is given in appendix A. We formally verify the following goals of the proposed protocol.

1) achieving mutual authentication between nodes S and T; [Derived in Equations (3) and (5)]
2) believing the symmetric key shared with target Node T by Node S as an initiator of the route discovery process; [Shown in Equation (4)]
3) achieving data origin authenticity and integrity of the received route request message among one-hop neighbors; and [Shown in Equations (6) and (7)]
4) achieving data origin authenticity and integrity of the RREP packet at source S. [Shown in Equation (8)]

We consider, here, the relevant message flows of Section 3 with respect to Figure 4. These are shown in Table 24 and are used to deduce the above mentioned goals using extended BAN logic. HMACs' of Table 24 are defined in Table 25.

Idealized forms of messages that appear in Table 24 are given in Table 26.

5.1.1 Initial State Assumptions

1) $S \mid\equiv\mid \xrightarrow{Pub_S} S$. It's natural for Node S to believe the public key of itself.
2) $S \mid\equiv\mid \xrightarrow{Pub_{TTP}} TTP$; As Node S is registered with TTP, it has the public key of TTP and believes the same.

Table 24: Relevant message flows

$Msg1 = S \rightarrow A$:{(S,T), T_{RREQ}, $Cert_S$, { $HMAC_A$, () }, {}}
$Msg2 = A \rightarrow X$:{(S,T), T_{RREQ}, $Cert_S$, {$HMAC_A, HMAC_X$}, {S}}
$Msg7 = D \rightarrow T$:{(S,T), T_{RREQ}, $Cert_S$, {$HMAC_D, HMAC_T$}, {S,A,X,B,C,Y}}
$Msg8 = T \rightarrow D$:{(S,T), T_{RREQ}, $Cert_T$, $HMAC_{Nodelist}$, $HMAC_{RREP}$, {S,A,X,B,C,Y,D,T}}
$Msg14 = A \rightarrow S$:{(S,T), T_{RREQ}, $Cert_T$, $HMAC_{Nodelist}$, $HMAC_{RREP}$, {S,A,X,B,C,Y,D,T}}

Table 25: Expansions of HMACs used in Table 24

$HMAC_S$ of Msg1 is defined as $H\{S \parallel T \parallel T_{RREQ} \parallel Cert_S \parallel A\}K_{SA}$
$HMAC_X$ of Msg2 is defined as $H\{HMAC_A \parallel S \parallel X\}K_{AX}$
$HMAC_{Nodelist}$ of Msg8 is defined as $H\{T_{RREQ} \parallel S \parallel A \parallel X \parallel B \parallel C \parallel Y \parallel D \parallel T, K_{TD}\}$

3) $TTP \mid\equiv \prod(Pri_{TTP})$. TTP believes that it has a good private key Pri_{TTP}.

4) $S \mid\equiv T \Longrightarrow T \overset{K_{ST}}{\longleftrightarrow} S$.

5) $T \mid\equiv S \Longrightarrow S \overset{K_{TS}}{\longleftrightarrow} T$. Based on the specification of the protocol, symmetric key is generated between source and target of the route discovery process, using ECDH. So both nodes S and T control symmetric key K_{ST} (K_{TS}).

6) $S \mid\equiv T \overset{K_{ST}}{\longleftrightarrow} S$. Node S believes the symmetric key K_{ST} which is produced by itself.

7) $T \mid\equiv \overset{Pub_T}{\longrightarrow} T$. It's natural for Node T to believe the public key of itself.

8) $T \mid\equiv \mid \overset{Pub_{TTP}}{\longrightarrow} TTP$. As Node T is registered with TTP, it has the public key of TTP and believes the same.

9) $T \mid\equiv S \overset{K_{TS}}{\longleftrightarrow} T$. Node T believes the symmetric key K_{TS} which is produced by itself.

10) $T \lhd$ {(S,T), T_{RREQ}, $Cert_S$, {$HMAC, HMAC$}, {$Nodelist$}}. $Cert_S$ is like a digital certificate issued by TTP which contains certificate statement Pub_S. So that,

11) $T \lhd \sigma(Pub_S, Pri_{TTP})$. From 2, 3 and by message meaning rule (2) for public key mentioned in Appendix A, it can be deduced that

12) $T \mid\equiv TTP \mid\sim Pub_S$ hence, it implies

$$T \mid\equiv \overset{\text{Pubs}}{\mapsto} S. \tag{3}$$

From 6 it can be deduced that

13) $S \mid\equiv T \mid\sim \{S \overset{K_{ST}}{\longleftrightarrow} T, (S,T), T_{RREQ}, Cert_T, HMAC_{RREP}, Nodelist\}$ except $HMAC_{Nodelist}$ as it is generated by its one-hop neighbor Node A.

14) $S \lhd$ {(S,T), T_{RREQ}, $Cert_T$, $HMAC_{Nodelist}$, $HMAC_{RREP}$, $Nodelist$}.

15) $S \lhd \{S \overset{K_{TS}}{\longleftrightarrow} T, (S,T), T_{RREQ}, Cert_T, HMAC_{Nodelist}, \{HMAC_{RREP}\}K_{TS}, Nodelist\}$. From 6, 15 and message meaning rule of 1 mentioned in Appendix A, it can be deduced that

16) $S \mid\equiv T \mid\sim \{S \overset{K_{ST}}{\longleftrightarrow} T, (S,T), T_{RREQ}, Cert_T, HMAC_{RREP}, \{Nodelist\}\}$. According to the freshness rule mentioned in Appendix A, it can be deduced that

17) $S \mid\equiv \#\{S \overset{K_{ST}}{\longleftrightarrow} T, (S,T), T_{RREQ}, Cert_T, HMAC_{RREP}, \{Nodelist\}\}$. According to nonce verification rule mentioned in Appendix A, it can be deduced from 16 and 17 that

18) $S \mid\equiv T \mid\equiv \{S \overset{K_{TS}}{\longleftrightarrow} T, (S,T), T_{RREQ}, Cert_T, HMAC_{Nodelist}, \{HMAC_{RREP}\}K_{TS}, \{Nodelist\}\}$. According to the jurisdiction rule mentioned in Appendix A, it can be deduced from 18 and assumption 4 that

$$S \mid\equiv S \overset{K_{TS}}{\longleftrightarrow} T. \tag{4}$$

From the idealized message format V, it is sure that S can see the message sent by Node T.

19) $S \lhd \{S \overset{K_{TS}}{\longleftrightarrow} T, (S,T), T_{RREQ}, Cert_T, H\{Nodelist\}K_{AS}, H\{S \parallel T \parallel T_{RREQ} \parallel Cert_T \parallel Nodelist\}K_{TS}, Nodelist\}$. $Cert_T$ is like a digital certificate issued by TTP which contains certificate statement Pub_T. So that,

20) $S \lhd \sigma(Pub_T, Pri_{TTP})$. From 2, 3 and by message meaning rule of 2 for public key mentioned in Appendix A, it can be deduced that

21) $S \mid\equiv TTP \mid\sim Pub_T$ hence, it implies

$$S \mid\equiv \overset{\text{Pubt}}{\mapsto} T. \tag{5}$$

Mutual authentication is performed by validating public keys of nodes S and T and is derived at Equations (3) and (5). Moreover, as an initiator of the route discovery process, node S believes that it has

Table 26: Idealized messages

$I : S \rightarrow A : \{(S,T), T_{RREQ}, Cert_S, (H\{S \parallel T \parallel T_{RREQ} \parallel Cert_S \parallel A\}K_{SA}), \{\}, \{\}\}$
$II : A \rightarrow X : \{(S,T), T_{RREQ}, Cert_S, (H\{S \parallel T \parallel T_{RREQ} \parallel Cert_S \parallel A\}K_{SA}), (H\{HMAC_A \parallel S \parallel X \parallel\}K_{AX}), \{S\}\}$
$III : D \rightarrow T : \{(S,T), T_{RREQ}, Cert_S, (H\{H\{H\{H\{H\{\}K_{SA}\}K_{AX}\}K_{XB}\}K_{BC}\}K_{CY}\}K_{YD} = AccumulatedHash),$
$(H\{AccumulatedHash \parallel Y \parallel T\}K_{DT}), \{S, A, X, B, C, Y\}\}$
$IV : T \rightarrow D : \{(S,T), T_{RREQ}, Cert_T, (H\{Nodelist\}K_{TD}), (H\{S \parallel T \parallel T_{RREP} \parallel Cert_T \parallel Nodelist\}K_{TS}), \{S, A, X, B, C, Y, D, T\}\}$
$V : A \rightarrow S : \{(S,T), T_{RREQ}, Cert_T, (H\{Nodelist\}K_{AS}), (H\{S \parallel T \parallel T_{RREP} \parallel Cert_T \parallel Nodelist\}K_{TS}), \{S, A, X, B, C, Y, D, T\}\}$

shared a symmetric key with target node T and is derived at 20. From the idealized message format I of Table 26, it is sure that node A can see the message sent by node S.

22) $A \lhd \{S \xleftrightarrow{K_{SA}} A, (S,T), T_{RREQ}, Cert_S, \{HMAC\}K_{SA}, (), \{Nodelist\}\}$

23) $A \lhd \{S \xleftrightarrow{K_{SA}} A, (S,T), T_{RREQ}, Cert_S, (H\{S||T||T_{RREQ}||Certs||A\}K_{SA})\}$. It can be deduced from 22, 24 and by message meaning rule of 1,

$$A \mid\equiv S \mid\sim \{S \xleftrightarrow{K_{SA}} A, (S,T), T_{RREQ}, Cert_S, (H\{S||T||T_{RREQ}||Certs||A\}K_{SA}), (), \{Nodelist\}\}. \quad (6)$$

Integrity assurance of the route request packet between authentic one-hop neighbor nodes (A and S) is derived at Equation (6). From the idealized message format II of Table 26, it is ensured that Node X can see the message sent by Node A. As per the assumptions, in subsection B of Section 3, authentic neighbors share a symmetric key among themselves. So that,

24) $A \mid\equiv X \xleftrightarrow{K_{XA}} A$.

25) $X \lhd \{A \xleftrightarrow{K_{AX}} X\ (S,T), T_{RREQ}, Cert_S, H\{S \parallel T \parallel T_{RREQ} \parallel Cert_S \parallel A\}K_{SA}, H\{HMAC_S \parallel S \parallel X\}K_{AX}, Nodelist\}$. By applying message meaning rule of 1 mentioned in Appendix A on 24 and 25, it can be deduced that

$$X \mid\equiv A \mid\sim \{H\{S \parallel T \parallel T_{RREQ} \parallel Cert_S \parallel A\}K_{SA}, (S)\}. \quad (7)$$

Integrity of received route request message sent by Node A is achieved at Node X and is derived at 7. By applying message meaning rule of 1, mentioned in Appendix A on 6 and 19 it can be deduced that

$$S \mid\equiv T \mid\sim \{(S,T), T_{RREQ}, Cert_T, Nodelist\}. \quad (8)$$

Integrity of the route reply packet sent by Node T is achieved at Node S and is derived at Equation (8).

5.2 AVISPA Model Checking Through HLPSL

Chun Chen et al. [9] have strongly argued the use of AVISPA [4], in testing the security protocols by mentioning that"AVISPA is a powerful tool with industrial strength technology for the analysis of different Internet security protocols and applications. The security protocols, standardized by the Internet engineering task force (IETF), have been analyzed by the AVISPA community, and, indeed, some of the protocols have been found to be flawed". Moreover, several proposals [5, 9], on security, are validated using AVISPA.

AVISPA consists of independently developed but interconnected modules. A protocol designer specifies the protocol with the security properties, which the protocol is supposed to achieve. It is a role-based formal language that allows specifying data structures, intruder models, complex security properties, different cryptographic primitives and their algebraic properties.

AVISPA translates the specified HLPSL model of the security protocol into an intermediate format IF. This is the input of several backends that are integrated into the AVISPA framework: SATMC,OFMC,Cl-Atse and TA4SP [4]. This is the base on which we have preferred to use AVISPA model checker for the evaluation of the proposed protocol. Therefore, we have modeled the route reply phase of the proposed protocol, using HLPSL because adversarial nodes manipulate particularly *Nodelist* and 2-hop ID fields of the RREP() to achieve routing misbehavior. In order to verify the authentication and security properties of the proposed protocol, we have modeled three roles such as source node, next hop (intermediate node) and target node of the route discovery process. The security properties, mainly data integrity and authenticity of the RREQ and RREP messages between source, and, next hop nodes, and also between source and target nodes, are modeled using HLPSL. And it is verified using AVISPA's OFMC and CL-AtSe back ends. And the outputs are shown in Figure 8.

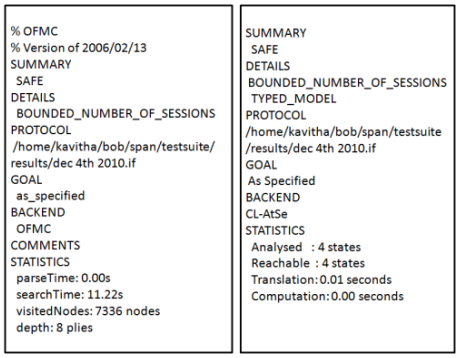

Figure 8: Results reported by OFMC and CL-Atse backends of AVISPA

6 Performance Analysis

Computational cost of the proposed protocol is measured, separately, at source, intermediate and target nodes,based on the number of energy intensive operations involved during the route discovery process. Here, based on the result of [25], we consider signature generation, verification and scalar multiplication operations which are energy intensive. Based on our performance analysis, we have compared the computational overhead at source, intermediate and target nodes for the proposed protocol, with respect to Ariadne [15] and EndairA [3] and the results are shown in Tables 27, 28 and 29.

Table 27: Computational overhead at source level

	Proposed	**Ariadne** [15]	**EndairA** [3]
No. of SG	-	-	-
No. of SV	1	$\alpha+1$	$\alpha+1$
No. of SM	1	-	-
No. of KH	3	1	-

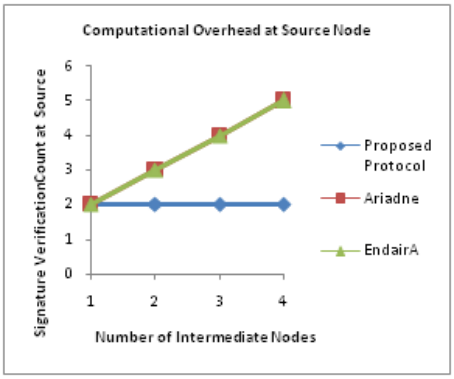

Figure 9: Computational overhead at source node

In Ariadne [15], each intermediate node generates signature while forwarding RREQ(). Target node generates digital signature over the previous signatures, which have been generated by intermediate nodes. Source needs to verify all the signatures to ensure the correctness of the discovered route. There is no monitoring mechanism employed in Ariadne for detecting routing misbehaviour.

Table 28: Computational overhead at i^{th} intermediate node level

	Proposed	**Ariadne** [15]	**EndairA** [3]
No. of SG	-	1	1
No. of SV	-	-	i
No. of SM	-	-	-
No. of KH	4	-	-

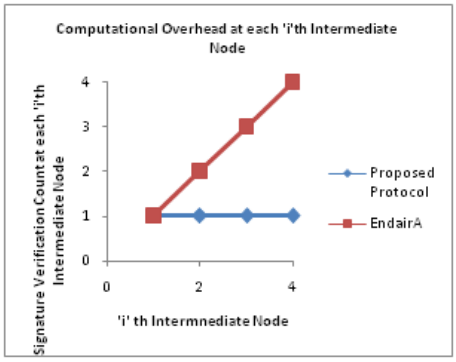

Figure 10: Computational overhead at intermediate node

In contrast to Ariadne, EndairA [3] employs signature generation at each intermediate node, during RREP() propagation. Each intermediate node verifies all the signatures, generated by its predecessors during RREP() propagation. Finally, source node needs to verify all those signatures, to assure the validity of the discovered path. And there is no design proposal to monitor the routing misbehavior.

6.1 Total Computational Cost

We have calculated the total computational overhead to establish a route of N nodes based on the proposed protocol and compared it with Ariadne and EndairA.

1) Proposed protocol
$(2 * Scalar\_MultiplicationCost + 2 * Sign\_VerifyCost + (N-2)*4 \cdot Keyed\_HashCost + 6 * Keyed\_HashCost)$.

2) Ariadne
$((2 * (N-2) + 1) * Sign\_VerifyCost + (N-1) * Sign\_GenCost)$.

3) EndairA
$((N-1) * Sign\_GenCost + \sum_{i=1}^{N-1} i * Sign\_VerifyCost)$.

Table 30 summarizes the evaluation outcome of the proposed protocol against Ariadne and EndairA [3, 15], with respect to computational simplicity, scalability and resilience to various attack scenarios. In the proposed

Table 29: Computational overhead at target node level

	Proposed	Ariadne [15]	EndairA[3]
No. of SG	-	1	1
No. of SV	1	α	-
No. of SM	1	-	-
No. of KH	3	1	-

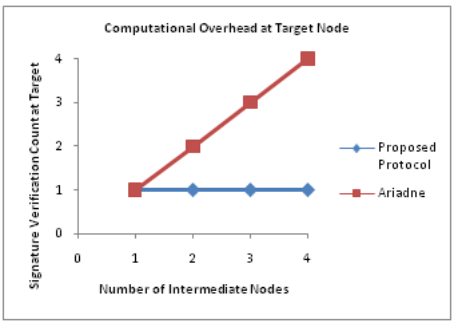

Figure 11: Computational overhead at target node

work, computational simplicity is measured in terms of less number of energy intensive cryptographic operations employed, during route discovery process. Scalability allows the flexibility for a new node to join in the network and to establish unique symmetric keys with other existing nodes and traceability refers the capability of the proposed protocol in identifying and precluding the aberrant nodes, during the route discovery process.

7 Discussion

Based on the analysis of the proposed protocol design, with respect to well known attacks and computational overhead, here, we brief several interesting observations:

Our unicast protocol design ensures source authenticity and integrity for route request, and route reply messages in a hop-by-hop level. This prevents worm hole, hidden channel and tunneling attacks during route discovery whereas *Ariadne* does not protect hidden channels in a hop-by-hop level. *EndairA* does not provide hop-by-hop level source authenticity and integrity for route request messages. This enables adversaries to exploit routing control packets as hidden channels.

Though our unicast approach incurs communication overhead, it avoids flooding and can be included with location based routing concept for effective utilization and this will be considered for future research work.

Our protocol design protects hidden channels of RREQ() and RREP() packets and proposed monitoring design identifies, and avoids maliciously behaving nodes during route reply propagation, whereas *Ariadne* and *EndairA* do not have such design.

Security analysis of the proposed protocol design has a profound impact on the resilience of various identified attacks as explained in **Section 5**. Formal verification, through extended BAN logic and AVISPA, ensures its correctness.

From the total computation cost presented in **Section 6**, it is observed that, in proposed protocol, the keyed hash computation depends on N (number of nodes on the established route) whereas in *Ariadne* and *EndairA*, signature generation and verification operations depend on N, and they are energy intensive, compared to keyed hash [25]. Figures 9, 10 and 11 illustrate the same.

The proposed protocol design has unfolded several areas for future work. Identification of false positive nodes in the monitoring and identification design is one such area of research which is beyond the scope of this research attempt.

8 Conclusion

In this paper, we have proposed a secure route discovery protocol to prevent hidden channel and worm hole attacks. In our protocol, authentic neighborhood concept is introduced to secure routing control packets. Traceability concept, defined in our protocol, uses promiscuous mode to detect and analyze routing misbehavior and isolate adversarial nodes. Authentication and key agreement between source and target are integrated in the route discovery process and hence, end-to-end mutual authenticity and data confidentiality are achieved. The most important factor, here, is that the proposed protocol not only prevents hidden channel attacks but also finds a plausible route between source and target. Therefore it can be applied in any wireless network in general and MANET in particular, which specifies privacy as a Quality of Service requirement. We have analyzed the security efficiency of the proposed protocol through formal verification methods such as extended BAN logic and AVISPA toolkit.

Acknowledgments

The authors are grateful for the constructive suggestions and comments, of the anonymous reviewers, which have improved the content and the presentation of this paper. Author Kavitha Ammayappan expresses her gratitude to IDRBT for its financial support to carry out this research work.

References

[1] G. Acs, L. Buttyan, and I. Vajda, “Provably Secure On-Demand Source Routing in Mobile Ad Hoc Net-

Table 30: Evaluation of the proposed protocol with related protocols

	Computational Simplicity	Scalability	Traceability of Malicious Nodes	Resilience to Attacks			
				A1	A2	A3	HC
Ours	Yes	Yes	Yes	Yes	Yes	Yes	No
Ariadne[15]	No	Yes	No	No	No	No	No
EndairA[3]	No	Yes	No	Yes	Yes	Yes	No

Note: A1: Active 1-2 Attack; A2: Active 2-2 Attack. A3: Attack on the Signature Version of Ariadne. HC: Hidden Channel Attack.

works," *Technical Report 159, International Association for Cryptologic Research*, 2004.

[2] G. Acs, L. Buttyan, and I. Vajda, "Provable security of on-demand distance vector routing in wireless ad hoc networks," *Proceedings of European Workshop on Security and Privacy in Ad Hoc and SensorNetworks (ESAS'05)*, pp. 113-127, 2005.

[3] G. Acs, L. Buttyan, and I. Vajda, "Provably secure on-demand source routing in mobile ad hoc networks," *IEEE Transactions on Mobile Computing*, vol. 5, no. 11, pp. 1533-1546, Nov. 2006.

[4] "AVISPA v1.1 User Manual," 2006.

[5] M. S. Bouassida, "Authentication vs. privacy within vehicular ad hoc networks," *International Journal of Network Security*, vol. 12, no. 3, pp. 256-272, 2011.

[6] M. Burrows, M. Adabi and R. Needham, "A logic of authentication," *Proceedings of Royal Society*, pp. 426, 1989.

[7] M. Burmester and B. de Medeiros, "On security of route discovery in MANETs," *IEEE Transactions on Mobile Computing*, vol. 8, no. 9, pp. 1180-1187, 2009.

[8] L. Buttyan and I. Vajda, "Towards provable security for ad hoc routing protocols, "*Proceedings of ACM Workshop on Ad Hoc and SensorNetworks (SASN'04)*, pp. 94-105, 2004.

[9] C. Chen, D. He, S. Chan, J. Bu, Y. Gao and R. Fan, "Lightweight and provably secure user authentication with anonymity for the global mobility network," *International Journal of Communication Systems*, vol. 24, no. 3, pp. 347-362, 2011.

[10] M. Fanaei, A. Fanian and M. Berenjkoub, "Prevention of tunnelling attack in Endaira," *Proceedings of Advanced Communication Technology*, pp. 1461-1466, 2008.

[11] N. Gura, A. Patel, A. Wander, H. Eberle, S. C. Shantz, "Comparing elliptic curve cryptography and RSA on 8-bit CPUs," *Proceedings of Cryptographic Hardware and Embedded Systems (CHES)*, pp. 119-132, 2004.

[12] Y. C. Hu, D. B. Johnson, and A. Perrig, "SEAD: Secure efficient distance vector routing for mobile wireless ad hoc networks," *Ad Hoc Networks*, vol. 1, no. 1, pp. 175-192, 2003.

[13] Y. C. Hu, A. Perrig, and D. B. Johnson, "Packet leashes: A defense against wormhole attacks in wireless networks," *Proceedings of IEEE INFOCOM*, pp. 1976-1986, 2003.

[14] Y. C. Hu and A. Perrig, "A survey of secure wireless ad hoc routing," *IEEE Security and Privacy*, vol. 2, no. 3, pp. 28-39, Mar. 2004.

[15] Y. C. Hu, A. Perrig and D. Johnson, "Ariadne: A secure on-demand routing protocol for ad hoc networks," *Wireless Networks Journal*, vol. 11, Issue 1, pp. 21-38, 2005.

[16] M. S. Hwang, C. C. Lee, J. Z. Lee, and C. C. Yang, "A secure protocol for bluetooth piconets using elliptic curve cryptography," *Telecommunication Systems*, vol. 29, no. 3, pp. 165-180, 2005.

[17] D. Johnson and D. Maltz, "Dynamic source routing in ad hoc wireless networks, "*Mobile Computing*, Thomasz Imielinski and Hank Korth (Editors), vol. 353, Chapter 5, pp.153-181, Kluwer Academic Publishers, 1996.

[18] J. Kim and G. Tsudik, "SRDP: Secure route discovery for dynamic source routing in MANETs," *Ad Hoc Networks*, vol. 7, Issue. 6, pp. 1097-1109, 2009.

[19] L. Lazos, R. Poovendran, C. Meadows, P. Syverson, L. W. Chang, "Preventing wormhole attacks on wireless ad hoc networks: A graph theoretic approach," *Proceedings of Wireless Communications and Networking Conference*, pp. 1193-1199, 2005.

[20] P. Papadimitratos and Z. Haas, "Secure routing for mobile ad hoc networks," *Proceedings of SCS Communication Networks and Distributed Systems Modeling and Simulation Conference (CNDS'02)*, pp. 193-204, 2002.

[21] P. Papadimitratos and Z. Haas, "Securing Mobile Ad Hoc Networks," *Handbook of Ad Hoc Wireless Networks*, M. Ilyas, ed., CRC Press, 2002.

[22] C. Perkins, "Ad-hoc on-demand distance vector routing," *Proceedings of Military Communication Conference (MILCOM'97)*, panel on ad hoc networks, 1997.

[23] C. E. Perkins and E. M. Belding-Royer, "Ad-hoc on-demand distance vector routing," *Proceedings of Second Workshop Mobile Computing Systems and Applications (WMCSA'99)*, pp. 90-100, 1999.

[24] C. E. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers," *Proceedings of ACM SIGCOMM*, pp. 234-244, 1994.

[25] N. R. Potlapally, S. Ravi, A. Raghunathan and N. K. Jha, "A study of the energy consumption characteristics of cryptographic algorithms and security

protocols," *IEEE Transactions on Mobile Computing*, vol. 5, pp. 128-148, 2006.
[26] K. Sanzgiri, B. Dahill, B. N. Levine, C. Shields, and E. M. Belding-Royer, "A secure routing protocol for ad hoc networks," *Proceedings of IEEE International Conference on Network Protocols (ICNP'02)*, pp. 78-89, 2002.
[27] Sufatrio and R. H. C. Yap, "Extending the Logic for Reasoning with Modern PKI based protocols,"*Proceedings of International Conference on NPC*, pp. 190-197, 2008.
[28] S. F. Tzeng, M. S. Hwang, "Digital signature with message recovery and its variants based on elliptic curve discrete logarithm problem," *Computer Standards & Interfaces*, vol. 26, no. 2, pp. 61-71, 2004.
[29] M. G. Zapata, "Secure ad hoc on-demand distance vector routing," *Mobile Computing and Communication Reviews*, vol. 6, no. 3, pp. 106-107, 2002.

Appendix A:

BAN and Extended BAN Logic Constructs

- $S \mid\equiv X$: S believes X;
- $S \lhd X$: S sees X;
- $S \mid\sim X$: S once said X;
- $S \Longrightarrow X$: S has jurisdiction over X;
- $\#(X)$: X is fresh;
- X_K: X encrypted with key K;
- $S \overset{K_{ST}}{\longleftrightarrow} T$: S and T may use a secret key K_{ST};
- $\Pi(Pri_{TTP})$: TTP has a good private key Pri_{TTP};
- $\mu(X, K_{ST})$: Keyed MAC (HMAC) calculated using key K_{ST};
- $\sigma(X, Pri_{TTP})$: X signed with TTP's private key Pri_{TTP};
- $\overset{\mathrm{PK_T}}{\mapsto} T$: PK_T is a public key for T;
- $\wp\kappa(TTP, Pub_{TTP})$: TTP has associated a good public key Pub_{TTP};

BAN Logic Inference Rules

- **Message Meaning Rule 1:** S believes $S \overset{K_{ST}}{\longleftrightarrow} T$ $\wedge$ S received $X\{K_{ST}\} \longrightarrow S$ believes T said X
- **Message Meaning Rule 2:** S believes $\wp\kappa(TTP, Pub_{TTP})$ $\wedge$ S believes $\Pi(Pri_{TTP})$ $\wedge$ S sees $\sigma(X, Pri_{TTP}) \longrightarrow S$ believes TTP said X
- **Nonce verification Rule 1:** S believes fresh (X) $\wedge$ S believes T said $X \longrightarrow S$ believes T believes X
- **Jurisdiction Rule 1:** S believes T controls X $\wedge$ S believes T believes $X \longrightarrow S$ believes X
- **Belief Conjuncatenation Rule 1:** S believes X $\wedge$ S believes $Y \longrightarrow S$ believes (X, Y)
- **Belief Conjuncatenation Rule 2:** S believes T believes $(X, Y) \longrightarrow S$ believes T believes X
- **Belief Conjuncatenation Rule 3:** S believes T said $(X, Y) \longrightarrow S$ believes T said X
- **Freshness Conjuncatenation 1:** S believes fresh $(X) \longrightarrow S$ believes fresh (X, Y)
- **Receiving Rules: Seeing is receiving 1:** S believes $S \overset{K}{\longleftrightarrow} T$ $\wedge$ S received $\{X\}_K \longrightarrow S$ received X
- **Receiving Rules: Seeing is receiving 2:** S believes $S \overset{K_{ST}}{\longleftrightarrow} T$ $\wedge$ S sees $\mu(X, K_{ST}) \longrightarrow S$ believes T said X

Kavitha Ammayappan received her BE in Computer Science and Engineering from the College of Engineering, Guindy, Anna University, India, in 2001 and her ME in Wireless Technologies from Thiagarajar College of Engineering, Madurai, Anna University, India in 2005. She is currently a PhD Scholar with the Department of Computer and Information Sciences, University of Hyderabad, India. She works as a research fellow at Institute for Development and Research in Banking Technology, a research and development wing of Reserve Bank of India, Hyderabad. Her current research interests include security in mobile ad hoc and wireless sensor networks, key management protocols, secure routing protocols and formal verification methods.

V. N. Sastry is currently working as an Associate Professor at the Institute for Development and Research in Banking Technology (IDRBT), Hyderabad, India since 1999. Prior to this he served at the National Institute of Technology, Tiruchirappalli, Tamilnadu, India for seven years as a faculty member. Dr. Sastry obtained his PhD Degree from the Indian Institute of Technology, Kharagpur in 1994. His areas of research interest are Routing Algorithms, Mobile Adhoc Networks, Access Control Models, Multi-objective optimization, Fuzzy Control and Risk Modelling.

Atul Negi is currently working as an Associate Professor in the Department of Computer and Information Sciences, AI Lab, University of Hyderabad, India. He has research interests in Document Analysis: Handwriting Segmentation and Recognition, Optical Character Recognition of machine printed Telugu Script. Pattern Recognition and its applications: to Systems Security, and Systems Research: Linux system architecture and applications, Mobile Adhoc networks. Dr. Negi is a Senior Member of IEEE, a Co-Founder Member and Moderator of Linux

User group of Hyderabad, Life Member of Indian Unit of International Association for Pattern Recognition. He has been associated as an investigator with funded projects from the Ministry of Home Affairs, Ministry of Communications and Information Technology and with Indian Space Research Organization.