ISSN:0975-9646

An Efficient Trusted Computing Base for Routing in MANET's

G.Krishna Kishore [1] , K.V.Sambasiva Rao [2]

[1] *Department of Computer Science & Engineering, V.R.Siddhartha Engineering College, Vijayawada, India*

[2] *Principal, MVR College of Engineering & technology, Paritala, Vijayawada, India*

Abstract— **We propose an approach to secure MANETs by employing low-complexity low-cost trustworthy MANET mod-ules (TMM), which perform some trivial hard-wired functions involving simple logical operations and block-cipher operations. We describe the functionality of such TMMs to offer a sound trusted computing base (TCB) for securing MANETs. We compare the performance of simulated MANET subnets which include non-cooperative nodes, employing the ad hoc on demand distance vector (AODV) protocol, to evaluate the efficacy of the proposed approach.**

Keywords: **MANET, AODE, TCB, TMM, TCG**

I. INTRODUCTION

A Mobile ad hoc network (MANET) is constituted by battery operated mobile computers with limited wireless transmission range. The MANET routing protocol is a set of rules which dictate the actions to be performed by every node to enable any two nodes in a subnet to establish multi-hop paths and relay data packets.

Securing a MANET involves providing some tangible assurances that nodes will abide by rules. In practice some nodes may be under the control of an untrusted or malicious user who may modify the software that encodes the MANET protocol; some nodes may misbehave accidentally due to hardware failure, or bugs in the MANET software, or even in the operating system of the mobile computer.

A. Trusted Computing Base

Realizing assurances towards securing any system is achieved by amplifying the trust in a trusted computing base (TCB) [3]. Most secure MANET protocols employ crypto-graphic authentication of routing data to limit an attackers ability to disseminate inconsistent routing information. The TCB for facilitating cryptographic authentication includes a set of cryptographic algorithms (which are assumed to be unbreakable), and a trusted authority (TA) who distributes cryptographic material to all nodes of a MANET network. Secure MANET protocols that leverage this limited TCB (trust in the authority and some cryptographic algorithms) to realize useful assurances typically demand substantial over-head for resource limited battery operated mobile devices.

B. Contributions

In this paper we propose a simple and efficient TCB for MANET nodes which can be leveraged to improve the performance of MANETs i) by providing assurances that reduce the scope of attacks that can be launched by attackers, and ii) by reducing the overhead required for leveraging the TCB. In the proposed approach, some TCB functions are executed inside trustworthy MANET modules (TMM) housed in every MANET node. We assume that only the TMMs are trusted: the rest of the node - all other hardware and software - are untrusted.

An important prerequisite for a trustworthy module to warrant trust is that the TCB functions executed inside the module are simple, and consequently, easily verifiable; simple TCB functions can also be implemented as hardwired logic, thus thwarting a wide range of attacks launched by modifying the functionality. It is also desirable that the modules consume as little power as possible, and consequently disseminate negligible heat, and thus can be physically well shielded from deliberate and accidental intrusions.

With these self-imposed limitations on TCB functions aimed at i) improving the reliability of TMMs and ii) lowering their cost, the proposed approach seeks a set of simple TCB functions to secure MANETs. To analyze the performance of our approach, in this paper we restrict ourselves to one popular MANET routing protocol - the ad hoc on demand distance vector (AODV) protocol [1]. We model ad hoc subnets which include some misbehaving nodes and compare performance of the proposed approach, AODV-TMM, with a more traditional approach, secure AODV (SAODV) [2], which employs only cryptographic authentication to dissuade some attacks.

The rest of this paper is organized as follows: Section II provides an overview of AODV, SAODV and trusted computing bases. Section III investigates the TCB for MANET nodes and details the functionality and interfaces offered by TMMs. Simulation results are presented in Section IV; conclusions are offered in Section V.

II.BACKGROUND

MANET routing protocols can be broadly classified into proactive and reactive protocols. In the former, nodes strive to maintain a consistent view of the subnet topology, whereas in the later, topological information is acquired on demand.

A. AODV

AODV is an ad hoc on-demand routing protocol where the route to a destination is stored in a node's routing table as a routing record (RR), indexed using the destination identity (ID). Other fields in the RR include a sequence number, hop-count, next-hop and validity time.

When a node desires to communicate with a destination, and finds that it does not have a fresh route to the destination, a route request (RREQ) is flooded, requesting a path to the destination. The RREQ contains sequence numbers of the initiator and destination, and a hop-count field which is initially set to zero. Every node that receives the RREQ updates the sequence number of the source, and adds an RR (for the source) in its table.

A receiver that does not have a path to the requested destination forwards the RREQ after incrementing the hop count field by one. When the RREQ arrives at a node that has a fresh enough path to the destination, or the destination itself, it responds by unicasting a route response (RREP) towards the source along the reverse path (AODV assumes that all links are bidirectional). In the case of RREP by an intermediate node the hop count in RREP is set to the stored value, and in the case of RREP by the destination, initialized to zero. Every node receiving the RREP adds an RR for the destination, increments the hop count, and unicasts the packet towards the source node.

Every entry in the routing table has a validity time after which it cannot be used. However, due to the mobility, the information in the RRs can become invalid even before the expiry period. AODV handles such premature expiry using route error (RERR) messages.

1) SAODV: SAODV [2] is a secure extension of AODV where every node has a public-private key pair with a certified public key. Digital signatures are used to authenticate im-mutable fields in RREQ, RREP and RERR messages, which includes a commitment to a hash chain. The hash chain is employed to prevent a malicious nodes from decreasing the received hop-count.

In order to secure route responses by intermediate nodes SAODV employs double signature extensions for RREQ and RREP packets. The RREPs generated by an intermediate node includes the signature of the destination to validate the immutable fields, and also a signature of the intermediate node to authenticate the new validity time.

B. Trusted Computing Base

The trusted computing base (TCB) of a system is "a small amount of software and hardware we rely on, and that we distinguish from a much larger amount that can misbehave without affecting security" [3].

As an example consider a generic communication system, where an important assurance sought is the ability to verify that a message sent from one entity to an another cannot be modified in transit by intermediaries. To realize such an assurance we typically rely on a TCB which includes a certificate authority (CA), who (we assume) does due diligence before signing public key certificates, and ensures that it's private key is well protected. We also rely on the assumption that cryptographic algorithms like RSA, DSA, AES, SHA-1 etc., are unbreakable. When one receives a message authenticated using the secrets belonging to an entity A, it is assumed that the message is from A, as we implicitly assume that the secrets of A are privy only to A.

As a more concrete example, the widely used web-security protocol, SSL, leverages such a TCB to provide an assurance that data sent by clients will be privy only to SSL servers, However, when a client sends some sensitive information (like a credit-card number) to a server over an SSL connection, it only ensures that the information remains private till it reaches the server. There is no assurance that such information cannot be abused after it reaches the server, by entities who have unfettered access to the server.

Thus, in many practical scenarios, the limited TCB which caters only for cryptographic authentication, is not sufficient as a basis for realizing important assurances. The most common approach to expand the TCB is by employing trustworthy computing modules which provide some "specialized" TCB functions, performed inside trustworthy boundaries.

1)Trustworthy Computing Modules: In the trustworthy computing group (TCG) model [4] for realizing trusted platforms[1], a trustworthy platform module (TPM) performs several specialized functions to provide i) the ability for remote parties to verify that the platform equipped with the TPM is in an "acceptable state" - that only authorized software has been loaded and executed by the CPU (even though the TPM does not have direct control over the CPU); and ii) for entities to seal secrets inside the TPM to some platform states, such that they will be released by the TPM only when the platform is in that specific state.

Unlike the TCG model where execution of code is per-formed outside the trusted boundary, the IBM 4758 [5] trust-worthy computing module sports a general purpose processor inside a protected boundary, running a specialized operating system, and can execute application code unmolested inside the trusted boundary. The rich set of programmable functions that can be executed inside the boundary can provide a rich TCB that can be leveraged to realize assurances that may not be possible otherwise.

III. TCB FOR MANET NODES

Unlike inexpensive TPM chips (a few dollars) with fixed functionality, the programmable TCB offered by IBM module comes at a substantially higher cost (a few thousand dollars). In this paper we seek a set of fixed functionality suitable for securing MANETs. We deliberately impose some restrictions on such fixed functionality to ensure that TMMs which offer such functionality can be easily verified, will consume negligible power, and thus can be simultaneously trustworthy and inexpensive to realize.

TMMs that offer the TCB for securing MANETs will demand substantially lower complexity compared to even inexpensive TPM chips. Unlike TPMs which offer a set of over 200 fixed functions, TMMs will offer 2 to 3 such functions. Unlike TPM chips, TMMs do not require to perform asymmetric cryptographic computations.

[1]Unfortunately, to realize all assurances sought by the TCG model, the TCB has to include many components outside the TPM.

A. TMM Functionality

Every mobile computer capable of taking part in any MANET is equipped with a TMM. A specific MANET network is administered by a trusted off-line authority, who provides secrets to TMMs of nodes that belong to the network. Any two TMMs A and B (housed in nodes A and B respectively) that belong to the network, can use their respective secrets to independently compute a pair wise secret K_{AB}. Any subset of nodes belonging to the network can come together to create a temporary MANET subnet.

Several key distribution schemes for facilitating pair wise secrets between trustworthy modules have been proposed in the recent past. For scenarios involving trustworthy modules there are compelling reasons to reduce the computational overhead inside the module for the operations performed using protected secrets. For the scheme in [6] each module will be required to store a few tens of keys and perform a few tens of block cipher operations for computing any pair wise secret. For the scheme in [7] each module will need to store a single secret and perform a single block-cipher operation. While both schemes support asynchronous induction of nodes, the former [6] can support unlimited network sizes; the latter [7] imposes a soft limit on the maximum number of nodes (not much more than a few tens of millions).

While operating in a MANET subnet, a mobile computer uses the interfaces exposed by its TMM to submit some values to the TMM and receive some message authentication codes (MACs), which is also a function of the time at which the MACs were computed. These MACs are used to authenticate both the information sent by a node and the node itself. When a TMM A is provided some values along with a MAC computed using a secret K_{AB}, the TMM A assumes that B is a neighbor. Every TMM maintains a list of active neighbors. TMMs will only accept authenticated routing information, and in response, subject to some rules, prepare MACs for messages that are verifiable only by active neighbors.

1) High Level Architecture of TMMs: Internally, every TMM consists of i) a secure pseudo-random function (PRF): for example, AES block cipher or SHA-1 hash; ii) a protected battery-backed RAM (BBRAM) for storing one or a few symmetric secrets, ii) limited RAM for storing a "neighbor table"; iii) I/O registers; iv) a clock-tick counter; and v) some hardwired control logic.

Mobile nodes (which are untrusted) communicate with their TMM (which is trusted) by writing a few values into the input registers of the TMM, and reading the outputs from the TMMs output registers. Values written into the input registers are typically fields like identities of nodes, MACs, and a clock-tick value. The TMM computes pair wise secrets, and employs such values along with other values provided to the TMM, and some internal values (like its clock-tick), to verify and/or compute MACs. Typically the output of a TMM consists of a few MACs, and a clock-tick value used for verifying the generated MACs. Such operations performed by the TMM to map inputs to outputs are controlled by the control logic inside the TMM, which essentially makes repeated use of the PRF.

The clock-tick-counters of all TMMs are assumed to have some extent of time synchronization (for example, within a few tenths of a second) to enable them to agree on the expiry time of routing records.

2) Notations: In the rest of this paper we shall use the following notations to describe the functionality of a TMM assigned an identity A.

K_A: A TMM with identity A stores a symmetric secret K_A in the protected BBRAM.

t: the current clock-tick count of node A.

F(): An internal TMM function - a secure PRF o = F(i_1, i_2), where o; i_1 and i_2 are fixed size bit-strings.

F_{pk}(): An internal TMM function; the secret K_A is used along with the PRF F() to provide the functionality $K_{A, idj} = K_j = F_{pk}(id_j, p_j)$ required to compute a pair wise secret. The exact nature of how F_{pk}() is realized using F() depends on the key distribution scheme used. If the scheme in [7] is used, then

$$K_{A, idj} = K_j = F_{pk}(id_j, p_j) = F(id_j, K_A) \oplus p_j \quad (1)$$

where p_j is a non-secret value corresponding to id_j.

$mac_j = F_M$ (**bs,** id_j, p_j): An internal TMM function; **bs** is a bit-string (of any length); the output is a MAC computed using a secret K_j which can also be computed by the neighbor id_j. This function is also realized using F as the building block. First $K_j = F_{pk}(id_j, p_j)$ is computed. Then F is used in the CBC mode to encrypt each block of **bs** using K_j.

NT : The neighbor table NT has one row for each neighbor, with four columns per-row. The values in a row j, viz id_j, t_j, p_j, bd_j represent the identity of a neighbor, the clock-tick count at which the id_j was last heard, the public value p_j associated with id_j, and a flag bd_j which is set when a packet sent by A is acknowledged by a neighbor id_j.

A node id_j is an active neighbor only if $t - t_j < {}_n$ where ${}_n$ is the maximum duration for which a node can be silent to avoid being removed from the neighbor table in the TMMs of its neighbors.

N is the set of active neighbors with tested bidirectional paths. Or an active neighbor $id_j \in N$ only if $bd_j = 1$. On the other hand, if an active neighbor id_j has $bd_j = 0$, then $id_j \in N'$. Usually N' is a set of neighbors which were heard, but their bidirectional link status has not yet been verified.

Δ: the lifetime of routing records.

RemoveNeighbor(id_j): An internal TMM function; remove neighbor id_j from *NT* ;

UpdNT (id_p , t_c, p_p, ACK): Internal TMM function which updates the neighbor table *NT* . *ACK* is a flag; if set, it indicates that the received packet is an *ACK*. If $(id_p \notin N) AND (t - t_c < \delta_n)$ insert row id_p, t_c, p_p, bd = *ACK*. On the other hand, if $(id_p \in N)$ (say in row j of *NT*), then replace t_j with t_c if $t_c > t_j$, and set $bd_j = 1$ if ACK = 1 (if bd_j is already set and ACK = 0, bd_j is **not** reset).

B. External Interfaces of the TMM

The TMM exposes the following two interfaces which can be used by nodes to submit some values as inputs to the TMM and receive some MACs as the TMM outputs.

1) RelayRR (RR, id_o, p_o).
2) RelayData (RR, d_h, hc, mac_o, t_o)

TABLE I
Fields in RR

Routing information	
id	Creator of RR
t_x	Time of expiry of RR
v_{aux}	A one-way function of any number of protocol specific values.
n_p	Number of intermediate nodes (to reach id)
VAL	Flag (1 –valid)
ACK	Flag (l-ACK)
Provider's information	
id_p	identity of the neighbor authenticating the RR
p_p	public value for node id_p and A
mac_p	A MAC created by id_p
t_c	Clock tic value of $id_{p\ when}$ mac_p was computed
F	Flag. F=1 to force removal of $id_{p\ from\ NT}$

In the above interfaces, RR consists of the fields shown in Table I.

The interface RelayRR() is used to submit an authenticated RR from an active neighbor to the TMM, or to create a RR. The TMM outputs a MAC (verifiable by node id_o whose public value is p_o) after modifying the RR fields subject to some rules. If id_o = NULL, TMM computes one MAC for each active neighbor. The rules enforced by the TMM are as follows:

1) When an RR is created by A (for HELLO, RREQ or RREP [2]) the values n_p is set to zero, and the time of expiry is chosen by the TMM as t + . The RR is authenticated only to active neighbors.
2) For HELLO packets which may need to be sent to a node which is currently not an active neighbor (but is within range), the TMM ensures that the value v_{aux} is set to zero.
3) Apart from RREQ and HELLO packets, RR will be authenticated by the TMM to active [2]Intermediate nodes creating RREP is treated as forwarding RR of the destination, rather than creating a new RR.neighbors only if an authenticated RR is received from an active neighbor.
4) For valid RRs with V AL = 1 the TMM ensues that the hop count n_p is incremented by 1
5) RRs with V AL = 0 represent route error packets. They can be created only if a valid RR is provided, and the neighbor which provided the RR is currently not an active neighbor. The TMM sets n_p = INF .
6) An RR submitted with V AL = 0 (for relaying a route error) will be relayed without modifying any values.

The RelayRR() algorithm is depicted in Figure 1.

The interface RelayData() is used to submit the hash of a data packet (d_h), along with a currently valid RR. To relay a data packet to a destination id (indicated in the RR), received from a neighbor id_o, who requests a path length less than hc, along with a MAC mac_o created by id_o at time t_o, the following rules are enforced by the TMM:

1) the value d_h is authenticated only to the node id_p in the RR
2) the MAC for id_p will be created only if n_p < hc.
3) if A is the originator (specified by setting id_o = A) the field mac_o will not be verified.

The RelayData() algorithm is depicted in Figure 2.

When TMM enabled nodes are used in a MANET subnet employing AODV, the nodes adhere to the plain AODV protocol. The only difference is that along with plain AODV packets, every node sends some additional values - a clock-tick value and some MACs. RREQ and RRER packets are accompanied by one MAC for every neighbor. RREP and data packets are accompanied by one MAC for the next hop.

When a node (say A) first hears a transmission from a neighbor (say B) which is not listed in its NT , A would send a HELLO packet to B with the v_{aux} set to 0. Upon processing this HELLO B would add A to its N' and send back an ACK, which would make A to list B in its N. B would also send a HELLO to force A to send an ACK so that it can also list A in its N.

```
IF (F = 1)
RemoveNeighbor($id_p$)
IF ($id_p$! = A)
bs = $id \| t_x \| v_{aux} \| n_p \| VAL \| t_c \| ACK$
$mac'_p$ = $F_m$(bs, $id_p$, $p_p$)
IF ( $mac_p$ != $mac'_p$ ) RETURN;
IF (F = 0) AND ($id_p$! = A)
UpdNT ($id_p$, $t_c$, $p_p$, ACK)
IF (ACK = 1) RETURN;
$t_c$ = t;
IF ($id_p$ = A) AND (id = A)
$t_c$ = t;
IF ($id_p$ = A) AND (id = A)
$n_p$ = 0; V AL = 1; $t_x = t_c + \Delta$ ;
IF ($id_o$! = NULL)
$v_{aux}$ = 0;
IF ( $id_p \in N$ ) AND (id! = A) AND ($t_x > t_c$)
IF (V AL = 1) $n_p = n_p + 1$;
IF ( $id_p \notin N$ ) AND ($id_p$! = A)AND(id! = A) AND (V AL = 1)
V AL = 0; $n_p$ = INF ;
bs = id || $t_x$ || $v_{aux}$ || $n_p$ || V AL || $t_c$
IF ($id_o$ = NULL)
FOR ALL $id_j \in N$
IF ($id_j$ ! = $id_p$)
$mac_j$ = $F_m$(bs || 0, $id_j$ , $p_j$ ).
     ELSE
$mac_j$ = $F_m$(bs || 1, $id_j$ , $p_j$ ).
IF ( $id_p \in N'$ ) AND ( $id_p \notin N$ )
$mac_p$ = $F_m$(bs || 1, $id_p$, $p_p$).
 IF ($id_o$! = NULL) AND ($v_{aux}$ = 0)
$mac_o$ = $F_m$(bs || 0, $id_o$ , $p_o$)
RETURN $t_c$,  MACs and N
```

Fig. 1. Algorithm RelayRR (RR; id_o; p_o)

```
IF ( $id_p \notin N$ ) OR (V AL = 0) OR (ACK = 1) OR ($n_p \geq$ hc) OR ($t_x \leq t$)
 RETURN;
IF ($id_o \neq$ A) AND ( $id_o \notin N$ )
RETURN
IF ( $id_p \in N$ )
bs = id || $t_x$ || $v_{aux}$ || $n_p$ || V AL || $t_c$ || ACK.
$mac'_p$ = $F_m$(bs, $id_p$ , $p_p$)
IF ( $mac_p$ ! = $mac'_p$ ) RETURN;
IF ($id_o \neq$ A)
//Let $id_o$  be the $j^{th}$entry in N
bs = id || hc || $d_h$ || $t_o$
$mac'_o$ = $F_m$(bs, $id_j$ , $p_j$ )
IF ( $mac'_o \neq mac_o$ ) RETURN;
bs = id || $n_p$ || $d_h$ || $t_c$
$mac_o$ = $F_m$(bs, $id_p$, $p_p$)
RETURN $mac_o$, $t_c$
```

Fig. 2. Algorithm RelayData (RR, d_h, hc, mac_o , t_o).

IV. **SIMULATIONS**

Simulations were carried out to evaluate and compare the performance of MANET subnets employing SAODV and AODV with TMM (represented as AODV-TMM).

A. Attacker Model

We have N_m malicious nodes that attempt to disrupt a MANET subnet of N nodes. Based on the security mechanisms employed, we restrict the capabilities of these malicious nodes. SAODV lacks mechanisms to authenticate intermediate nodes; and hence an attacker can claim any random ad-dress/identity while forwarding packets. Moreover, due to the inadequacy of hash-chains, a bad node can forward RREQs without incrementing the hop-count, and thus stand a higher chance of being accepted in the path. Once in the path, the nodes can simply drop the received RREP.

In the case of AODV-TMM bad nodes can hide some RRs from the TMM, and drop some packets (dropping all packets will lead to removal of the node from the neighbor tables of neighbors). When a bad node has a valid (till time t_1) RR to a destination D, but receives a route error at time $t_2 < t_1$ indicating that the path to D does not exist anymore, the bad node can hide the fresh RR, and continue to submit the invalid RR till time t_1.

In our simulations we model malicious behavior of bad nodes in the SAODV subnet by making them claim random identities, forward the RREQ without incrementing the hop-count, and dropping RREP packets. In the subnet with TMM we model malicious behavior by making bad nodes forward invalidated RRs whenever it is possible for them to do so. Furthermore, once in a path, they drop 50% of all packets.

B. Simulation Parameters

We generate N = 150 randomly placed nodes in a square region with 500 meter edges. The radius of each node is 60 meters (every node had 5 neighbors on an average). The bit rate of the channel is assumed as 2Mb/s - the duration of a 256 byte packet is about 1 msec. The carrier sense delay was assumed to be T_{cs} = 1μ sec - or two nodes within the range of each other may not sense each other's transmission if they begin their transmissions within T_{cs} = 1μ sec of each other.

Nodes employ p-persistent CSMA with p ≈ 1/20 for broadcasts. If a node senses that the channel becomes available at a time t, it begins its transmission at a time $t + xT_{cs}$ where 1≤ x ≤ 20 is uniformly distributed), if no node within its range had started a transmission before t + (x - 1) T_{cs} .

1) Maintaining Neighborhood Information: In order to maintain dynamic neighborhood information every node is forced to break silence at least once in an interval of length T_s = 0.25 seconds. If a node A has not had the need to transmit a control or data packet in the last T_s seconds, A sends a HELLO packet to notify its presence to its neighbors. If A has not heard a transmission from a neighbor B for more than $2T_s$ seconds, the neighbor B is removed from A's neighbor table.

2) RREQ Initiation: From the N = 150 nodes we fixed a pool of 50 nodes as potential sources and destinations. RREQ initiations were modeled as poisson events with a mean of 3 events per second. In each event a random source-destination pair is chosen from the set of 50 nodes; after sending a RREQ the source expects an RREP within 0.5 seconds, failing which the RREQ is retransmitted. If route establishment fails during the second attempt too, the source gives up. If an RREP is received, the source sends n_d data packets (were n_d is uniformly distributed between 5 and 10). The source expects an ACK packet from the destination for every data packet within 0.5 seconds, failing which the data packet is retransmitted one more time. A RR is valid until Δ = 3 seconds from the time of its creation.

3)Mobility: Motion events in the subnet are modeled three types of poisson events. The first type is simple mobility, which occur at the rate 7.5 events per second; in each such event a node is relocated by a certain distance (uniformly distributed between ±15 meters) in X and Y directions. The second type of event is turning off of nodes, occurring about a rate of 3.75 events per second. Only the nodes that are not end-points in an RREQ or RREP are turned off. The third type occurs at a rate of 2 events per second, where a currently off node is turned on, and relocated at a random position.

In our simulations data and RREP packets are unicast. The sender (say A) will confirm retransmission by over-hearing the packet being forwarded by the receiver (say B). Upon identification of failure to retransmit (due to collisions or if the neighbor simply dropped the packet), A resends the packet to B. If two successive retransmissions fail, A will delete B from its neighbor-list[3].

4)Packet Sizes: SAODV employs digital signatures and hash-chains, while the AODV-TMM employs MACs. For SAODV we assume 40 byte elliptic curve signatures, 70 byte certificates and 10 byte hashes. For the AODV-TMM we assume 4 byte MACs. According to the draft AODV specification [1] the packet sizes for RREQ and RREP in plain AODV are 24 and 20. When the MACs are added AODV-TMM will require on an average (assuming five neighbors) 64 bytes for RREQ, 32 bytes for RREP. On the other hand the packet sizes for SAODV will be 224 for RREQ, 160 for single-signature RREP (and 228 bytes for double-signature extension when an intermediate node responds to an RREQ). For both cases data packets are assumed to be 256 bytes. The size of RERR messages are also close to that of RREQs.

C. Results

Each simulation run was for a network duration of 100 seconds. The simulations were repeated for varying number of malicious nodes N_m = (0, 10, 20). For fair comparison we ensure that both SAODV and AODV-TMM subnets have identical topology at all times, and have the same random triggers that influence node motion and RREQ generation. The parameters estimated were:

[3] This is the reason that in AODV-TMM bad nodes do not drop all packets (they drop only 50% of the packets), as if they do so they will be ejected from the neighborhoods of other nodes, and lose their ability to attack the subnet

TABLE II
COMPARING THE PERFORMANCES OF SAODV AND AODV-TMM

	SAODV			AODV – TMM		
	0	10	20	0	10	20
V_r	86%	66%	405	92%	89%	83%
b	4.1e5	5.1e5	5.3e5	1.7e5	1.8e5	1.8e5
V_d	78%	51%	30%	85%	80%	72%

a) V_r : percentage of successful path establishments, total RREQs initiated n_i; only $n_i^{'} < n_i$ pairs had a physical path; only n_r paths discovered; $V_r = n_r / n_i^{'}$.
b) *b*: the total number of bytes transmitted by each node (on an average) during the 100 second simulation interval.
c) V_d : the percentage of packets delivered to the destination; ideally, on an average $d_i = 7.5 \times n_i^{'}$ should be delivered; if d_r is delivered then $V_d = d_r / d_i$.

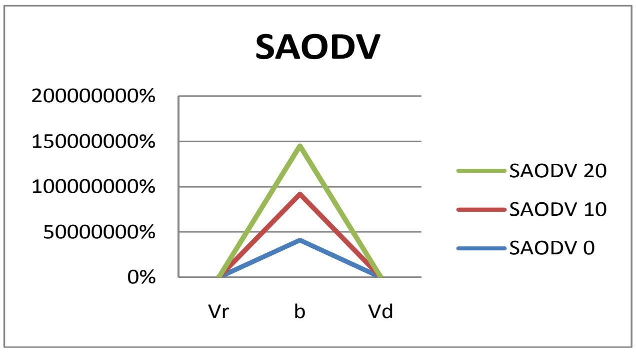

Fig 3. Performance of SAODV

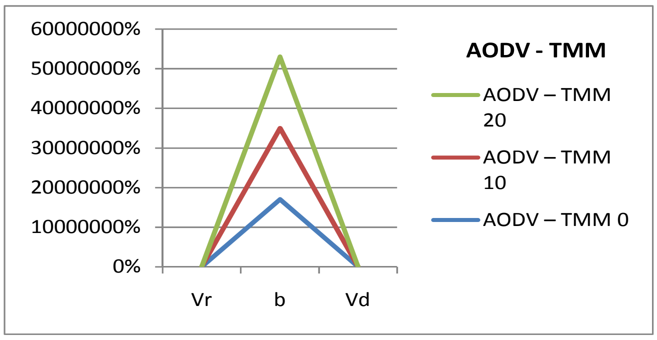

Fig 4. Performance of AODV - TMM

The substantially smaller success rate of path establishment for SAODV, with increasing number of bad nodes, is due to the fact that any path with a malicious node is bound to fail, and will necessitate a second RREQ (which is also highly likely to fail). On the other hand, malicious nodes cannot modify routing information in AODV-TMM - they can either use prematurely expired routing records to initiate RREPs that will fail, or drop some RREP / data packets.
The higher bandwidth overhead in SAODV can be attributed to three reasons - larger packet sizes, lower success of the path establishment process mandating RREQ reflooding, and a large number of RERR packets. As SAODV has no mechanism for authentication of neighboring nodes, bad nodes that impersonate random identities are seen as real neighbors for a short duration. The perceived loss of such neighbors then trigger RERR packets.
With respect to data traffic, for both subnets (SAODV and AODV-TMM) the number of RREQ events are identical, and even the total number of data packets created, d_i (about $7.5 n_i^{'}$) is also same. However as a smaller fraction of routes are established in SAODV, the total number of data packets that are actually transmitted, $d_t = 7.5n_r$, is substantially lower for SAODV. Out of the d_t data packets, only d_r are ultimately successful. For the case with 20 malicious nodes for SAODV d_i=1980, d_t=1193, and d_r=594; for AODV-TMM d_i=1980, d_t=1851, and d_r=1426. Overall, for the case of 20 malicious nodes, compared to SAODV, AODV-TMM results in higher data throughput by a factor 2.4 while demanding lower bandwidth overhead, by a factor 3. For the case of 10 malicious nodes AODV-TMM results in a 1.5 times increased throughput while requiring 2.8 times lower bandwidth.

V. RELEVANT WORK AND CONCLUSIONS

The need for trusted boundaries in which co-operative routing tasks are carried out has been addressed by some researchers. In [8] the authors include the wireless transceiver inside the trust boundary. In [9] the trusted computing module has complex features built into the wireless driver (executed within the confines of the trusted module) to verify the integrity of wireless transceiver. Arguably, bringing the wireless transceiver within the scope of the protected boundary as in [8] or including complex features in wireless software drivers (executed within the trusted boundary) as in [9] implies high cost for practical realization of such trust modules.
In [10] explicit consideration is given to the need for lowering the complexity of tasks to be performed inside the trusted boundary. The scheme employs “nuglets of currency” protected by smart-cards to promote faithful forwarding of packets. Nevertheless [10] still assumes that the trusted computing modules should be capable of performing asym-metric cryptographic computations, which raises the bar for the capabilities and the cost of such modules. More recently Gaines et al [11] have proposed a generic dual-agent approach to MANETs where some desired characteristics of a trustworthy network agent (like low computational and storage requirements) are enumerated.
One of the primary motivation for the proposed approach is to minimize the complexity of operations performed inside the TMM to the extent feasible, but yet effectively curtail the freedom of an attacker. Lower the complexity of the TMM, lower the cost, and higher is the extent of trust one can place on the immutability of the TCB. At the same time the specialized TCB functions can be used to substantially reduce the overhead for securing MANETs, as has been demonstrated by our simulations.
Our current research focus is to integrate effective monitoring strategies, in conjunction with the TMM approach, to realize further improvements in MANET performance. Even without such features, AODV-TMM has better performance than SAODV - both in terms of data throughput, and band-width overhead. We are also

investigating changes that may be required to the TMM interfaces to support other routing protocols like DSR and TORA.

ACKNOWLEDGEMENT

The authors would like to express their thanks to Management and Principal of V.R.Siddhartha Engineering College for their support and encouragement.

REFERENCES

[1] C. Perkins, E.Royer, S. Das "Ad hoc On-demand Distance Vector (AODV) Routing, Internet Draft, draft-ietf-manet-aodv-11.txt, Aug 2002. The 6th World Multi-Conference on Systemics, Cybernetics and Informatics (SCI 2002), 2002

[2] M.G.Zapata, N.Asokan, "Securing Ad hoc routing protocols," WISE-02, Atlanta, Georgia, 2002.

[3] B. Lampson, M. Abadi, M. Burrows, E. Wobber, "Authentication in Distributed Systems: Theory and Practice," ACM Transactions on Computer Systems, 1992.

[4] TCG Specification: Architecture Overview, Specification Revision 1.4, 2nd August 2007.

[5] S.W. Smith, S. Weingart, "Building a High-Performance Programmable Secure Coprocessor," IBM Technical Report RC21102, Feb 1998.

[6] M. Ramkumar, "The Subset Keys and Identity Tickets (SKIT) Key Distribution Scheme," IEEE Transactions on Information Forensics and Security, **5**(1), pp 39–51, March 2010.

[7] M. Ramkumar, "On the scalability of a "non-scalable" key distribution scheme," IEEE SPAWN, Newport Beach, CA, June 2008.

[8] J-H.Song,V.Wong,V.Leung,"Secure Routing with Tamper Resistant Module for Mobile Ad Hoc Networks," ACM SIGMOBILE Mobile Computing and Communications,vol.7,no.3,ACM Press,NY,Jul 2003.

[9] M. Jarrett, P. Ward,"Trusted Computing for Protecting Ad-hoc Routing," Proceedings of the 4th Annual Communication Networks and Services Research Conference, IEEE Computer Society, May 2006.

[10] J-P. Hubaux, L Buttyan, S. Capkun, "Quest for Security in Mobile Ad Hoc Networks," Proceedings of the ACM MOBIHOC 2001.

[11] B. Gaines, M. Ramkumar, "A Framework for Dual Agent Routing Protocols for MANETs," IEEE Globecom 2008, LA, Nov 2008.

.