Performance Evaluation and Testing of EAACK-Secure IDS for MANETs: Brief Study

Harisha Datla[#1], Mallikarjun Reddy D[#2], Santosh Naidu P[#3]

[1, 2, 3] *Computer Science and Engineering, MVGR College of Engineering, India*

***Abstract*— Among all the up to date wireless networks, Mobile circumstantial Network (MANET) is one amongst the foremost necessary and classifiable applications. On the adverse to ancient spec, Manet doesn't need a set network infrastructure; each single node works as each a sender and a receiver and they trust their neighbours to relay messages. Unfortunately, the open medium and remote distribution of Manet create it at risk of numerous kinds of attacks. So, it is essential to develop efficient intrusion-detection mechanisms to protect MANET from attacks. In this paper, we define solid privacy requirements regarding malicious attackers in MANET. Then we propose and implement a new intrusion-detection system named Enhanced Adaptive Acknowledgment (EAACK) specially designed for MANETs. Compared to contemporary approaches, EAACK demonstrates higher malicious-behaviour-detection rates in certain circumstances while does not greatly affect the network performances.**

***Keywords*— EAACK, MANET, AODV, RERR,**

I. General Terms

Network Security, Cryptography

II. Introduction

The migration to wireless network from wired network has been a global trend in the past few decades. The mobility and scalability brought by wireless network made it possible in many applications. Among all the coeval wireless networks, Mobile Ad hoc Network (MANET) is one of the most important and unique applications. On the contrary to traditional Network architecture, MANET does not require a fixed network infrastructure; every single node works as both a transmitter and a receiver. Nodes communicate directly with each other when they are both within the same communication range. Otherwise, they rely on their neighbours to relay messages. The self-configuring ability of nodes in MANET made it popular among critical mission applications like military use or emergency recovery. However, the open medium and wide distribution of nodes make MANET vulnerable to malicious attackers.

In this case, it is crucial to develop efficient intrusion-detection mechanisms to protect MANET from attacks an intrusion detection system (IDS) is a device or software application that monitors network or system activities for malicious activities or policy violations and produces reports to a Management Station. Some systems may attempt to stop an intrusion attempt but this is neither required nor expected of a monitoring system. Intrusion detection and prevention systems (IDPS) are primarily focused on identifying possible incidents, with the improvements of the technology and cut in hardware costs, we are witnessing a current trend of expanding MANETs into industrial applications. To adjust to such Trend, we strongly believe that it is vital to address its potential security issues.

Existing System:

A number of secure routing schemes have been brought forward for intrusion-detection in MANETs.

1. WATCHDOG: It is responsible for detecting malicious node misbehaviours in the network. Watchdog detects malicious misbehaviours by promiscuously listening to its next hop's transmission. It will improve the throughput of network with the presence of malicious nodes.

2. TWOACK: In order to overcome the drawbacks in watchdog, a new scheme is proposed that is TWOACK, to resolve the receiver collision and limited transmission power problems of Watchdog, TWOACK detects misbehaving links by acknowledging every data packet transmitted over every three consecutive nodes along the path from the source to the destination.

3. AACK: It is similar to TWOACK, AACK is an acknowledgment-based network layer scheme which can be considered as a combination of a scheme called TACK (identical to TWOACK) and an end-to-end acknowledgment scheme called Acknowledge (ACK). It can significantly reduce overhead when compared with TWOACK.

Disadvantages: Existing schemes are largely depend on the acknowledgment packets. Hence, it is crucial to guarantee that the acknowledgment packets are valid and authentic but they suffer from the problem that they fail to detect malicious nodes with the presence of false misbehaviour report and forged acknowledgment packets.
Another drawback of most previous schemes is the significant amount of unwanted network overhead. Due to the limited battery power nature of MANETs, such overhead can easily degrade the life span of the entire network.

Proposed System:

In this project, we propose and implement a new and efficient intrusion-detection system named Enhanced Adaptive Acknowledgment (EAACK) specially designed for MANETs. Compared to contemporary approaches, EAACK demonstrates higher malicious-behaviour-detection rates in

certain circumstances while does not greatly affect the network performances.
In the proposed method we incorporated digital signature in our proposed scheme. In order to ensure the integrity of the IDS, EAACK requires all acknowledgment packets to be digitally signed before they are sent out and verified until they are accepted.

Advantages: Our proposed approach EAACK completely overcomes the weaknesses like false misbehaviour, limited transmission power, and receiver collision.
All acknowledgment packets in EAACK are authentic and untainted. Our proposed method can significantly improve the packet delivery ratio.

III. RELATED WORK

- In 1999, [1] Lidong Zhou and Zygmunt J. Haas. studed the threats an ad hoc network faces and the security goals to be achieved and identified the new challenges and opportunities posed by this new networking environment and explore new approaches to secure its communication. In particular, they take advantage of the inherent redundancy in ad-hoc networks - multiple routes between nodes - to defend routing against denial-of-service attacks. And they also used replication and new cryptographic schemes, such as threshold cryptography, to build a highly secure and high1 available key management service.

  Pros and cons:
  Focused on how to secure routing and how to establish a secure key management service in an ad hoc networking environment. These two issues are essential to achieving our security goals. Besides the standard security mechanisms, we take advantage of the redundancies in ad hoc network topology and use diversity coding on multiple routes to tolerate both benign and Byzantine failures. But, more work needs to be done to deploy these security mechanisms in an ad hoc network and to investigate the impact of these security mechanisms on network performance.
- In 2002, [2] Yih-Chun Hu, David B. Johnson and Adrian Perrig designed and evaluate the Secure Efficient Ad hoc Distance vector routing protocol (SEAD), a secure ad hoc network routing protocol based on the design of the Destination-Sequenced Distance-Vector routing protocol (DSDV).

  Pros and cons:
  SEAD is efficient and can be used in networks of computation- and bandwidth-constrained nodes. SEAD actually outperforms DSDV-SQ in terms of packet delivery ratio. But the self-advertising routes of the nodes are not included and DSDV is not behaving like a path vector routing protocol.
- In 2004, [3] James Parker, Jeffrey Undercoffer, John Pinkston, and Anupam Joshi presented network intrusion detection (ID) mechanisms that rely upon packet snooping to detect aberrant behavior in mobile ad hoc networks. Their extensions, which are applicable to several mobile ad hoc routing protocol, offer two response mechanisms, passive - to singularly determine if a node is intrusive and act to protect itself from attacks, or active - to collaboratively determine if a node is intrusive and act to protect all of the nodes of an ad-hoc cluster.

  Pros and cons:
  A dropping of the packet can easily be recognized and logged. The implementation of both the Passive and Active ID algorithms in GloMoSim led to a number of parameters that can be adjusted. But the performance is not greatly enhanced and the node density is not determined.

- In 2007, [4] Kejun Liu, Jing Deng, Pramod K. Varshney and Kashyap Balakrishnan proposed the 2ACK scheme that serves as an add-on technique for routing schemes to detect routing misbehavior and to mitigate their diverse effect. The main idea of the 2ACK scheme is to send two-hop acknowledgment packets in the opposite direction of the routing path.

  Pros and cons:
  Compared with other approaches to combat the problem, such as the overhearing technique, the 2ACK scheme overcomes several problems including ambiguous collisions, receiver collisions, and limited transmission powers. The 2ACK scheme can be used as an add-on technique to routing protocols such as DSR in MANETs. But, the knowledge of topology of the 2-hop neighbourhood may be used. In addition, the 2ACK scheme can only work in managed MANETs (as compared to open MANETs).

- In 2007, [5] Nidal Nasser and Yunfeng Chen overcome the weakness of Watchdog and introduce our intrusion detection system called ExWatchdog. The main feature of the proposed system is its ability to discover malicious nodes which can partition the network by falsely reporting other nodes as misbehaving and then proceeds to protect the network.

  Pros and cons:
  ExWatchdog solves a fatal problem of Watchdog, i.e., a malicious node can partition the network by falsely reporting other nodes as misbehaving. Our solution decrease the overhead greatly, though it does not increase the throughput obviously. It is not reliable, because the solution is based on assumption only.

- In 2008, [6] Yunseop (James) Kim, Robert G. Evans, and William M. Iversen described details of the design and instrumentation of variable rate Irrigation, a wireless sensor network, and software for real-time in-field sensing and control of a site-specific precision linear-move irrigation system.

  Pros and cons:
  Proved a concept of a promising low-cost wireless solution for an in-field WSN and remote control of precision irrigation. Bluetooth wireless technology used in here offered a plug-and-play communication module. But it does not provide features for real time field monitoring, automated irrigation control, and remote operation of field machinery.

- In 2008, [7] Jin-Shyan Lee proposed a command filtering framework to accept or reject the human-issued commands so that undesirable executions are never performed. In the present approach, Petri nets are used to model the operated behaviours.

  Pros and cons:
  In the proposed filtering framework, human operators could request unrestricted commands, and the command filters would make real-time decisions based on an event-trigger and on-demand P2P networking. Since, it does not satisfy the security, integrated attempt is needed.

- In 2008, [8] Denis Dondi, Alessandro Bertacchini, Davide Brunelli, Luca Larcher and Luca Benini proposed a methodology for optimizing a solar harvester with maximum power point tracking for self-powered wireless sensor network (WSN) nodes. And focussed on maximizing the harvester's efficiency in transferring energy from the solar panel to the energy storing device.

  Pros and cons:
  This method significantly improves the whole harvester efficiency, which is crucial in WSN applications. Noticeably, the harvester implemented using commercial components features a high efficiency (85%), which to our knowledge is the highest ever reported in the literature. But a trade-off between harvester Performances and physical implementation to keep the silicon costs below a bearable threshold is not considered. This leads to increased cost.

- In 2004, [9] Ye Zhu, Xinwen Fu, Bryan Graham, Riccardo Bettati and Wei Zhao focused on a particular class of traffic analysis attack, flow correlation attacks, by which an adversary attempts to analyse the network traffic and correlate the traffic of a flow over an input link at a mix with that over an output link of the same mix.

  Pros and cons:
  Analysing of mix networks was done in terms of their effectiveness in providing anonymity and quality-of-service and it shows that it can achieve a guaranteed low detection rate while maintaining high throughput for normal payload traffic but unlink ability alone is not enough in hostile environments like battlefields as important information like packet type is still available to attackers. Then a passive attacker can mount traffic analysis based on packet type.

- In 2004, [10] Yanchao Zhang, Wei Liu and Wenjing Lou proposed a novel anonymous on-demand routing protocol, termed MASK, to enable anonymous communications thereby thwarting possible traffic analysis attacks. Based on a new cryptographic concept called pairing, he first propose an anonymous neighbourhood authentication protocol which allows neighbouring nodes to authenticate each other without revealing their identities.

  Pros and cons:
  A pairing-based anonymous on-demand routing protocol MASK is which provides strong sender and receiver anonymity, the relationship anonymity between senders and receivers, the unlocatability of mobile nodes, and the intractability of packet flows under a rather strong adversarial model but the routing information is not authenticated in the current design of MASK.

IV. MODULE DESCRIPTION

1. Basic routing module

a. If the source has no route to the destination, then source v initiates the route discovery in an on-demand fashion.
b. After generating RREQ, node looks up its own neighbour table to find if it has any closer neighbour node toward the destination node.
c. If a closer neighbour node is available, the RREQ packet is forwarded to that node.
d. If no closer neighbour node is the RREQ packet is flooded to all neighbour nodes.

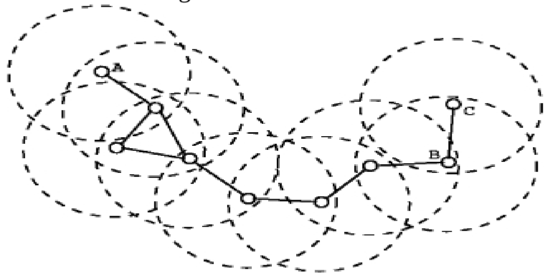


Figure 1 Basic Routing Module between nodes

2. Include hacking in basic routing module

In this module Attack issues will arise in to the network. Providing security to the attacks will be considered.

Black hole Attack:

MANETs face different securities threats i.e. attack that are carried out against them to disrupt the normal performance of the networks. In these attacks, black hole attack is that kind of attack which occurs in Mobile Ad-Hoc networks (MANET).

In black hole attack, a malicious node uses its routing protocol in order to advertise itself for having the shortest path to the destination node or to the packet it wants to intercept. This hostile node advertises its availability of fresh routes irrespective of checking its routing table. In this way attacker node will always have the availability in replying to the route request and thus intercept the data packet and retain it.

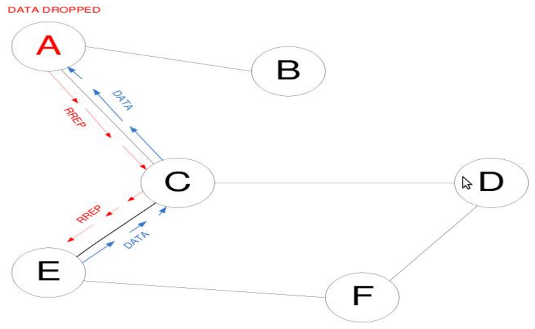


Figure 2 Data Packet Transfer between nodes during Black hole Attack

3. Secure Acknowledgement

a. In this module, we are implementing secure acknowledgement to detect misbehaving nodes in the routing environment. [11]

b. In this module we are ensuring that acknowledgement is authentic and untainted by Digital Signature.

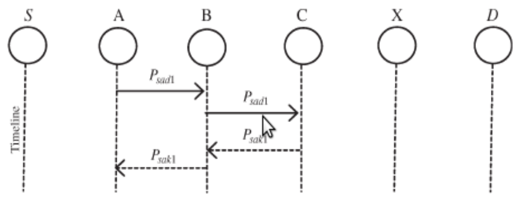


Figure 3 Secure Acknowledge between nodes

In the above figure S is the source node whenever it doesn't receives the acknowledgement it will start secure acknowledgement process within three-three nodes. Here A, B, C is the 1st group which node A sending one packet to node B, it will forward to node C after that both nodes B and C have to send acknowledgement to node A within time. If acknowledgement not received means it will report those nodes as misbehaving nodes to source node. But in this process there is a chance of false reports to avoid this we are implementing MRA.

4. MRA

a. In this module we are avoiding false reports generated by the Misbehaving nodes.

b. The main aim of MRA scheme is to authenticate whether the destination node has received the reported missing packet through a different route.

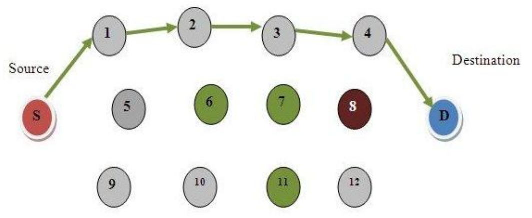


Figure 4 MRA Scheme authentications

In the above figure we can observe that between source and destination there are multiple paths available in MANET. So, to avoid false reports in secure ACK scheme we will find another path between source and destination and source will check the reports which it gotten from intermediate nodes if any false report found means it will treat the node which sent that report as a misbehaving node.

V. ALGORITHM

```
#routing packets from source to destination#
Create a list N (all); #A set contains all the information about nodes
Initiate Route discovery using RREQ and RREP;
Transmit the packets (Sdata to Ddata)

#checking node activity#

If {Dack == receive} {
Ddata;
} else {
Initiate Sack
}
If(Receiveddata == Sack){
Misbehavior report(a);
If(Misbehavior report(a)==0) {
Send Dack;
} else {
Initiate MRA;
}
If(Receiveddata == MRA){
Find another path to Destination;
If(Destination node doesn't has packet) {
```

```
**Trust the report**
} else {
Mark reporter as malicious;
}
Create a list H(i);# storing information about malicious nodes;
}
```

VI. DESIGN

The purpose of the design phase is to plan a solution of the problem specified by the requirement document. This phase is the first step in moving from problem domain to the solution domain. The design of a system is perhaps the most critical factor affecting the quality of the software, and has a major impact on the later phases, particularly testing and maintenance. The output of this phase is the design document. This document is similar to a blue print or plan for the solution, and is used later during implementation, testing and maintenance.

The design activity is often divided into two separate phase-system design and detailed design. System design, which is sometimes also called top-level design, aims to identify the modules that should be in the system, the specifications of these modules, and how they interact with each other to produce the desired results. At the end of system design all the major data structures, file formats, output formats, as well as the major modules in the system and their specifications are decided.

During detailed design the internal logic of each of the modules specified in system design is decided. During this phase further details of the data structures and algorithmic design of each of the modules is specified. The logic of a module is usually specified in a high-level design description language, which is independent of the target language in which the software will eventually be implemented. In system design the focus is on identifying the modules, whereas during detailed design the focus is on designing the logic for each of the modules. In other words, in system design the attention is on what components are needed, while in detailed design how the components can be implemented in software is the issue.
During the design phase, often two separate documents are produced, one for the system design and one for the detailed design.

Together, these documents completely specify the design of the system. That is they specify the different modules in the system and internal logic of each of the modules.

A design methodology is a systematic approach to creating a design by application of set of techniques and guidelines. Most methodologies focus on system design. The two basic principles used in any design methodology are problem partitioning and abstraction.

A large system cannot be handled as a whole, and so for design it is partitioned into smaller systems. Abstraction is a concept related to problem partitioning. When partitioning is used during design, the design activity focuses on one part of the system at a time. Since the part being designed interacts with other parts of the system, a clear understanding of the interaction is essential for properly designing the part. For this, abstraction is used. An abstraction of a system or a part defines the overall behaviour of the system at an abstract level without giving the internal details.

While working with the part of a system, a designer needs to understand only the abstractions of the other parts with which the part being designed interacts. The use of abstraction allows the designer to practice the "divide and conquer" technique effectively by focusing one part at a time, without worrying about the details of other parts.

Like every other phase, the design phase ends with verification of the design. If the design is not specified in some executable language, the verification has to be done by evaluating the design documents. One way of doing this is thorough reviews. Typically, at least two design reviews are held-one for the system design and one for the detailed and one for the detailed design.

Design Objectives:

a. Design models help us to visualize a system as it is or as we want it to be.
a. Design models permits us to specify the structure or behaviour of the system.
b. Design models give us a template that guides us in constructing a system.
c. Design models document the decision we have made.

VII. REQUIREMENTS

Hardware Requirements:

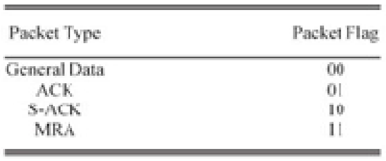
1. Single PC
2. 20 Gb Hard disc space
3. 1Gb RAM

Software Requirements:

1. Linux OS (Ubuntu 10.04)
2. NS2.34

VIII. SCHEME DESCRIPTION

In this section, [10] we describe our proposed EAACK scheme in detail. The approach described in this research paper is based on our previous work , where the backbone of EAACK was proposed and evaluated through implementation. In this paper, we extend it with the introduction of digital signature to prevent the attacker from forging acknowledgment packets. EAACK is consisted of three major parts, namely, ACK, secure ACK (S-ACK), and misbehaviour report authentication (MRA). In order to distinguish different packet types in different schemes, we included a 2-b packet header in EAACK. According to the Internet draft of DSR, there is 6-b reserved in the DSR header. In EAACK, we use 2-b of the 6-b to flag different types of packets. Details are listed in Table 1.



Packet Type	Packet Flag
General Data	00
ACK	01
S-ACK	10
MRA	11

Table 1 Packet type indications

Figure 5 presents a flowchart describing the EAACK scheme. Please note that, in our proposed scheme, we assume that the link between each node in the network is bidirectional. Furthermore, for each communication process, both the source node and the destination node are not malicious. Unless specified, all acknowledgment packets described in this research are required to be digitally signed by its sender and verified by its receiver.

A. **ACK:** As discussed before, ACK is basically an end-to-end acknowledgment scheme. It acts as a part of the hybrid scheme in EAACK, aiming to reduce network overhead when no network misbehaviour is detected. In Fig. 8, in ACK mode, node S first sends out an ACK data packet Pad1 to the destination node D. If all the intermediate nodes along the route between nodes S and D are cooperative and node D successfully receives Pad1, node D is required to send back an ACK acknowledgment packet Pak1 along the same route but in a reverse order. Within a predefined time period, if node S receives Pak1, then the packet transmission from node S to node D is successful. Otherwise, node S will switch to S-ACK mode by sending out an S-ACK data packet to detect the misbehaving nodes in the route.

B. S-ACK: The S-ACK scheme is an improved version of the TWOACK scheme proposed by Liu et al. [16]. The principle is to let every three consecutive nodes work in a group to detect misbehaving nodes. For every three consecutive nodes in the route, the third node is required to send an S-ACK acknowledgment packet to the first node. The intention of introducing S-ACK mode is to detect misbehaving nodes in the presence of receiver collision or limited transmission power. As shown in Fig. 9, in S-ACK mode, the three consecutive nodes (i.e., F1, F2, and F3) work in a group to detect misbehaving nodes in the network. Node F1 first sends out S-ACK data packet Psad1 to node F2. Then, node F2 forwards this packet to node F3. When node F3 receives Psad1, as it is the third node in this three-node group, node F3 is required to send back an S-ACK acknowledgment packet Psak1 to node F2. Node F2 forwards Psak1 back to node F1. If node F1 does not receive this acknowledgment packet within a predefined time period, both nodes F2 and F3 are reported as malicious. Moreover, a misbehaviour report will be generated by node F1 and sent to the source node S. Nevertheless, unlike the TWOACK scheme, where the source node immediately trusts the misbehaviour report, EAACK requires the source node to switch to MRA mode and confirm this misbehaviour report. This is a vital step to detect false misbehaviour report in our proposed scheme.

C. **MRA**: The MRA scheme is designed to resolve the weakness of Watchdog when it fails to detect misbehaving nodes with the presence of false misbehaviour report. The false misbehaviour report can be generated by malicious attackers to falsely report innocent nodes as malicious. This attack can be lethal to the entire network when the attackers break down sufficient nodes and thus cause a network division. The core of MRA scheme is to authenticate whether the destination node has received the reported missing packet through a different route. To initiate the MRA mode, the source node first searches its local knowledge base and seeks for an alternative route to the destination node. If there is no other that exists, the source node starts a DSR routing request to find another route. Due to the nature of MANETs, it is common to find out multiple routes between two nodes. By adopting an alternative route to the destination node, we circumvent the misbehaviour reporter node. When the destination node receives an MRA packet, it searches its local knowledge base and compares if the reported packet was received. If it is already received, then it is safe to conclude that this is a false misbehaviour report and whoever generated this report is marked as malicious. Otherwise, the misbehaviour report is trusted and accepted. By the adoption of MRA scheme, EAACK is capable of detecting malicious nodes despite the existence of false misbehaviour report.

D. Digital Signature

As discussed before, EAACK is an acknowledgment-based IDS. All three parts of EAACK, namely, ACK, S-ACK, and MRA, are acknowledgment-based detection schemes. They all rely on acknowledgment packets to detect misbehaviours in the network. Thus, it is extremely important to ensure that all acknowledgment packets in EAACK are authentic and untainted. Otherwise, if the attackers are smart enough to forge acknowledgment packets, all of the three schemes will be vulnerable. With regard to this urgent concern, we incorporated digital signature in our proposed scheme. In order to ensure the integrity of the IDS, EAACK requires all acknowledgment packets to be digitally signed before they are sent out and verified until they are accepted. However, we fully understand the extra resources that are required with the introduction of digital signature in MANETs. To address this concern, we implemented both DSA and RSA digital signature schemes in our proposed approach. The goal is to find the most optimal solution for using digital signature in MANETs.

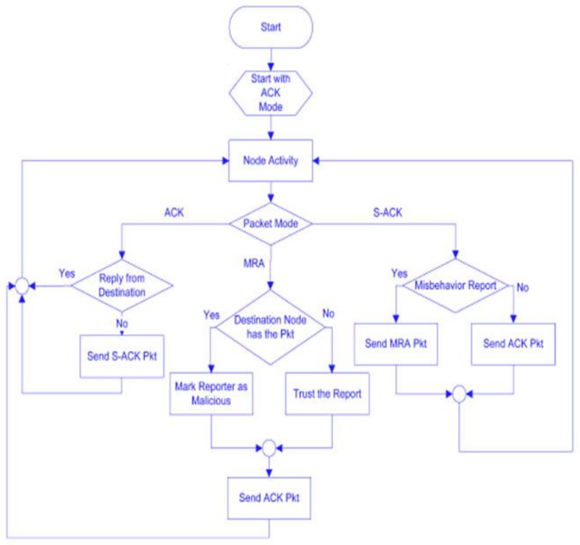


Figure 5 S-ACK scheme: Node C is required to send back an acknowledgment packet to node A.

IX. PERFORMANCE EVALUATION

To better investigate the performance of EAACK under different type of attacks, we propose three scenario settings to simulate different type of misbehaviours or attacks [11]

1) **Scenario 1:** In this scenario, we simulated a basic packet dropping attack. Malicious nodes simply drop all the packets they receive. The purpose of this scenario is to test the performance of IDSs against two weaknesses of Watchdog; namely, receiver collision and limited transmission power.

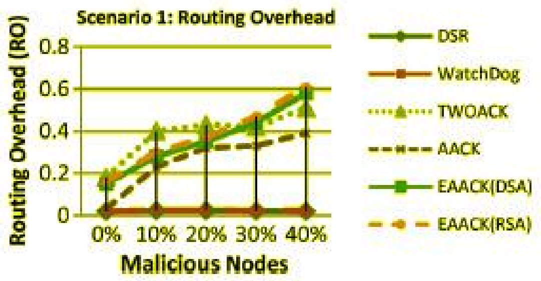


Figure 6 Simulation Results for Scenario 1

2) **Scenario 2:** This scenario is designed to test IDSs' performances against false misbehaviour report. In this case, malicious nodes always drop the packets they receive and send back a false misbehaviour report whenever it is possible.

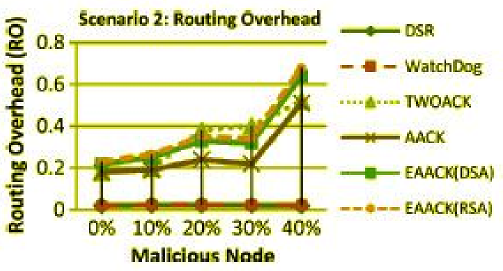


Figure 7 Simulation Results for Scenario 2

3) **Scenario 3:** This scenario is used to test IDSs' performances when the attackers are smart enough to forge acknowledgement packets and claiming positive result while in fact it is negative. As Watchdog is not an acknowledgement based scheme, it is not eligible for this scenario setting**.**

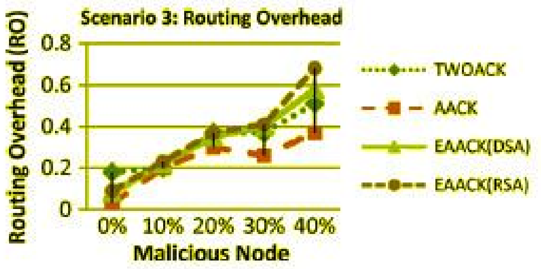


Figure 8 Simulation Results for Scenario 3

X. TESTING AND DEBUGGING

Testing and debugging a program is one of the most tedious parts of computer programming. The testing and debugging phase of a project can easily take more time than it took to write the application. Testing includes both checking that the code runs at all, that it runs correctly under all circumstances, and that it runs the same way it did before you made changes. Tcl's error diagnostics make it easy to track down coding errors; the modular nature of Tcl code makes it easy to do unit testing of functions, and the tcl test package makes it easy to write integrated regression test suites.

Debugging Code:

The first step to debugging a Tcl script is to examine the Tcl error output closely. Tcl provides verbose error information that leads you to the exact line where a coding error occurs. Tcl error messages consist of a set of lines. The first line will describe the immediate cause of the error (Incorrect number of arguments, invalid argument, undefined variable, etc). The rest of the message describes more details about where the error occurs. For example, this procedure has a fairly common error - the closing brace and bracket are in the wrong order:

```
proc hasError {a} {
return [expr {$a+2]}
}
```

The error message is:

```
missing close-bracket
```

```
while executing
"return [expr {$a+2]}
"
(procedure "hasError" line 2)
invoked from within
"hasError 1"
```

The first line describes the error (missing curly bracket), and the rest of the lines show the exact line in the program (return [expr {$a+2]}), and where that line occurs (second line in the hasError procedure) Here are a few of the common error messages and a description of the code that generates them.
wrong # args: should be ...set result "a b" button .b -text set -command "set x $result" This example looks reasonable, however, the command to be evaluated when the button is clicked is created by substituting the the $result and concatenating the values into a string resembling this: set x a b. The recommended way of creating a button like this is to use the list command to maintain grouping:

```
set result "a b"
button .b -text set -command [list set x $result]
missing "
missing close-brace
missing close-bracket
```

Your code has an unterminated string that starts with a double-quote. The usual causes of this are typos (not hitting shift fast enough and having a double quote at one end of a string and a single quote at the other), having a space after a back-slash line continuation character, or mismatching the open/close pairs of a set of nested quotes, braces and brackets. Here are some examples of lines that would generate one of these errors.

```
set rtn "there is a space after the backslash \
causing this to generate an error"
puts "Missing close quote
puts "mismatched quote and brace}
puts "should be double quote to close'
can't read "...": no such variable
```

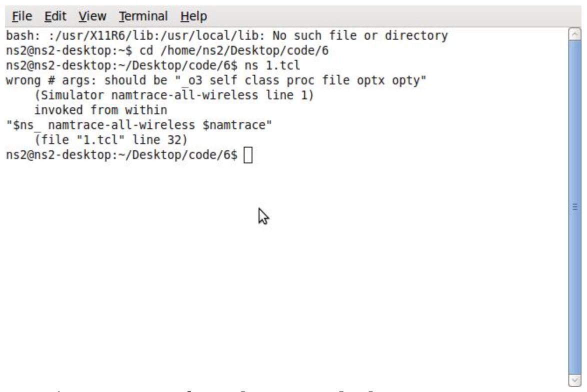
There is a $varName in your code for which varName has never been set. This can happen:

1. Because you were reworking code and forgot to initialize a variable.
2. Because a global scope variable is referenced in a procedure without declaring it global.
3. Because a procedure invoked from a widget did not declare the widget's –text variable to be in global scope.
4. Because code is being evaluated outside the scope in which it was created. Code connected with a button, after event, etc is evaluated in the global scope (unless a namespace or class scope is assigned) even if the widget is created within a procedure or method. Procedure scope local variables will no longer exist after the GUI has taken control of the application.

```
set globalVar 1
# Fails because of missing "global globalVar"
proc hasError {} {
puts "$globalVar"
}
# The entry widget is OK, but the button will cause an error.
entry .e -textvariable globalVar
button .b -text "generateError" -command hasError
# This button references a local variable.
proc makeBadButton {} {
set xx "local variable"
button .b -text "throw error" -command "puts $xx"
grid .b
}
```

invalid command name "..."
The first word on a command line is not a valid command or procedure name. This is most often caused by mistyping a command name, or forgetting to source or package require the Tcl code that defines a command or procedure.
Syntax error in expression "...": variable references require preceding $
invalid bareword "..."
This error is generated by the expr command. It's caused when you try to do arithmetic on a string. The usual causes are that you forgot to put a dollar-sign on a variable name or that a variable that should hold a number was assigned a string value.

Black Box Testing:
Black box testing also called behavioural testing focuses on the functional requirements of the software. That is black box testing enables the software engineer to derive sets of input conditions that will fully exercise all functional requirements for a program. Black box testing attempts to find errors in the following categories. Incorrect or missing functions. Interface errors. Errors in data structures or external data base access Behaviour or performance errors. Initialization and termination errors.

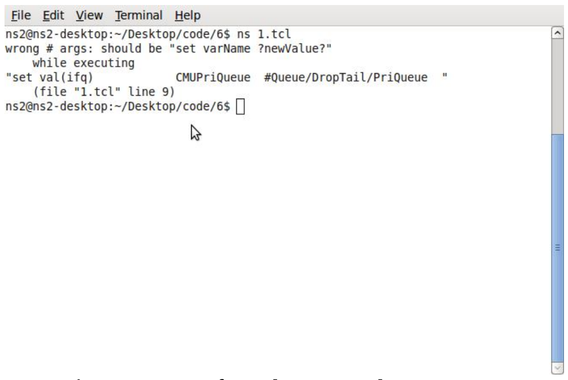


```
bash: :/usr/X11R6/lib:/usr/local/lib: No such file or directory
ns2@ns2-desktop:~$ cd /home/ns2/Desktop/code/6
ns2@ns2-desktop:~/Desktop/code/6$ ns 1.tcl
wrong # args: should be "_o3 self class proc file optx opty"
    (Simulator namtrace-all-wireless line 1)
    invoked from within
"$ns_ namtrace-all-wireless $namtrace"
    (file "1.tcl" line 32)
ns2@ns2-desktop:~/Desktop/code/6$
```

Figure 9 Interface showing Black Box Testing

White Box Testing:

White box testing sometimes called glass box testing is a test case design method that uses the control structure of the procedural design to derive test cases. Using white box testing methods, the software engineer can derive test cases that guarantee that all independent paths within a module have been exercised at least once. Exercise all logical decisions on their true and false sides. Execute all loops at their boundaries and within their operational bounds. Exercise internal data structures to ensure their validity.

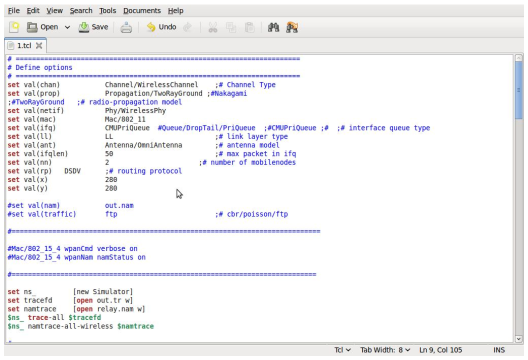


```
ns2@ns2-desktop:~/Desktop/code/6$ ns 1.tcl
wrong # args: should be "set varName ?newValue?"
    while executing
"set val(ifq)           CMUPriQueue  #Queue/DropTail/PriQueue  "
    (file "1.tcl" line 9)
ns2@ns2-desktop:~/Desktop/code/6$ 
```

Figure 10 Interface showing White Box Testing

Unit Testing:

The most 'micro' scale of testing to test particular functions or code modules. Typically, it is done by the programmer and not by tester, as it requires detailed knowledge of the internal program design and code. Not always easily done unless the application has a well-designed architecture with tight code; may require developing test modules or test harnesses.



```
# ======================================================================
# Define options
# ======================================================================
set val(chan)           Channel/WirelessChannel    ;# Channel Type
set val(prop)           Propagation/TwoRayGround ;#Nakagami
;#TwoRayGround   ;# radio-propagation model
set val(netif)          Phy/WirelessPhy
set val(mac)            Mac/802_11
set val(ifq)            CMUPriQueue  #Queue/DropTail/PriQueue  ;#CMUPriQueue ;#  ;# interface queue type
set val(ll)             LL                         ;# link layer type
set val(ant)            Antenna/OmniAntenna        ;# antenna model
set val(ifqlen)         50                         ;# max packet in ifq
set val(nn)             2                      ;# number of mobilenodes
set val(rp)   DSDV      ;# routing protocol
set val(x)              280
set val(y)              280

#set val(nam)           out.nam
#set val(traffic)       ftp                        ;# cbr/poisson/ftp

#==========================================================================

#Mac/802_15_4 wpanCmd verbose on
#Mac/802_15_4 wpanNam namStatus on

#==========================================================================

set ns_         [new Simulator]
set tracefd     [open out.tr w]
set namtrace    [open relay.nam w]
$ns_ trace-all $tracefd
$ns_ namtrace-all-wireless $namtrace
```

Figure 11 Interface showing Unit Testing

XI. CONCLUSION AND FUTURE WORK

EAACK makes MANETs more secure. The major threats like false misbehaviour report and forge acknowledgement can be detected by using this scheme. Possibilities of adopting hybrid cryptography techniques to further reduce the network overhead caused by digital signature. Examine the possibilities of adopting a key exchange mechanism to eliminate the requirement of pre-distributed keys and also test the performance of EAACK in real time

XII. ACKNOWLEDGEMENTS

I would like to thank Department of Computer Science and Engineering staff and management of MVGR College of Engineering for their support in doing this work.

REFERNECES

[1] Lidong Zhou and Zygmunt J. Haas, *IEEE*, no. Network Security, pp. 1-12, November 1999.

[2] David B. Johnson b, Adrian Perrig Yih-Chun Hu a, "SEAD: secure efficient distance vector routing for mobile wireless ad hoc networks," *Elsevier*, pp. 175-192, 2003.

[3] Undercoffer, J. Parker J, J. Pinkston, and A. Joshi, "On intrusion detection and response for mobile ad hoc networks," in *Performance, Computing, and Communications, 2004 IEEE International Conference* , USA, 2004, pp. 747-752.

[4] Jing Deng, Pramod K. Varshney, and Kayshap Balakrishnan Kejun Liu, "An Acknowledgment-Based Approach for the Detection of Routing Misbehavior in MANETs," *IEEE*, vol. 6, pp. 536-550, May 2007.

[5] N. Nasser, Guelph Univ. of Guelph, and Yunfeng Chen, "Enhanced Intrusion Detection System for Discovering Malicious Nodes in Mobile Ad Hoc Networks," in *Communications, 2007. ICC '07. IEEE International Conference*, Glasgow, 2007, pp. 1154 - 1159.

[6] Evans, R.G. Yunseop Kim and W.M. Iversen, "Remote Sensing and Control of an Irrigation System Using a Distributed Wireless Sensor Network," *IEEE*, vol. 57, no. 7, pp. 1379 - 1387, May 2008.

[7] Jin-Shyan Lee and Hsinchu Ind. Technol. Res. Inst., "A Petri Net Design of Command Filters for Semiautonomous Mobile Sensor Networks," *IEEE*, vol. 55, no. 4, pp. 1835 - 1841, April 2008.

[8] Alessandro Bertacchini, Davide Brunelli, Luca Larcher, Luca Benini Denis Dondi, "Modeling and Optimization of a Solar Energy Harvester System for Self-Powered Wireless Sensor Networks," *IEEE*, vol. 55, no. 7, pp. 2759-2766, July 2008.

[9] Xinwen Fu , Bryan Graham , Riccardo Bettati , Wei Zhao Ye Zhu, "On flow correlation attacks and countermeasures in mix networks," in *PET'04 Proceedings of the 4th international conference on Privacy Enhancing Technologies*, Heidelberg, 2005, pp. 207-225.

[10] Wei Liu Yanchao Zhang, Wenjing Lou, and Yuguang Fang, "Anonymous handshakes in mobile ad hoc networks," in *Military Communications Conference*, 2004, pp. 1193 - 1199.

[11] Nan Kang, and Tarek R. Sheltami Elhadi M. Shakshuki, "EAACK—A Secure Intrusion-Detection System for MANETs," *IEEE*, vol. 60, no. 3, pp. 1089-1098, March 2013.

[12] ABIRAMI.S.P., KALYANI.R., SUBHASHRI.M., SUCHITHRA.R MS. M.PONNRAJAKUMARI, "A SECURE INTRUSION DETECTION USING IMPROVED EAACK IN MANETS," in *5th IRF International Conference*, Chennai, 2014, pp. 41-47.

[13] Vairachilai.S Arockia Rubi.S, "A Survey on Intrusion Detection System in Mobile Adhoc Networks," *International Journal of Computer Science and Mobile Computing*, vol. 2, no. 12, pp. 389 – 393, December 2013.