A Network Defense System for Detecting and Preventing Hacking

Suchithra C, Vaishnavi R, Ashwini Gharde

[1]Assistant Professor ,Department of Computer,K.C College of Engineering & Management Studies & Research
[2]Assistant Professor,Department of Information Technology,SNS College of Engineering
[3]Assistant Professor ,Department of Computer,K.C College of Engineering & Management Studies & Research

---***---

Abstract - Many computing systems have recently suffered significantly from hacking and preventing hacking is important in protecting business, sensitive information and every day network communications. Many efforts have been made to provide security assurance by proposing security solutions. However, the gap between hacking incidents and current security solutions is significant. Fortunately, hacking attempts can be addressed if the pre-hacking step, called scanning, is properly investigated and good counter measures are in place. The importance of scanning appears is in providing sophisticated hackers with the necessary information about nominated victims systems which eventually forms their hacking strategies. Therefore, this article proposes a security solution that aims to make scanning difficult by addressing its properties, which makes developing hacking strategies against protected computer networks unpractical. The proposed security solution protects computer networks by dynamically generating a unique protocol to replace the expected standard protocols and changing network paths periodically in order to confuse scanning attempts, as well as to prevent unauthorized scanning and hacking traffic.

Key Words: hacking,sensitive information, unauthorized scanning, hacking traffic, network communications

1. INTRODUCTION

In today's digital landscape, the prevalence of hacking incidents poses a significant threat to businesses, sensitive information, and everyday network communications. As cyber criminals become increasingly sophisticated, the need for robust security measures has never been more critical. Despite numerous efforts to enhance security assurance through various solutions, a notable gap remains between the frequency of hacking attempts and the effectiveness of current protective measures. One crucial aspect of this cyber security challenge is the pre-hacking phase known as scanning, where attackers gather vital information about their targets to formulate their strategies. This article explores the importance of addressing scanning vulnerabilities and proposes an innovative security solution designed to complicate the scanning process. By dynamically generating unique protocols and periodically altering network paths, this approach aims to thwart unauthorized scanning attempts and enhance the overall security of computer networks.

1.1 Mobile ad hoc network

Mobile Ad-hoc network is a set of wireless devices called wireless nodes, which dynamically connect and transfer information. Wireless nodes can be personal computers (desktops/laptops) with wireless LAN cards, Personal Digital Assistants (PDA), or other types of wireless or mobile communication devices. Figure 1.1 illustrates what MANET is. In general, a wireless node can be any computing equipment that employs the air as the transmission medium. As shown, the wireless node may be physically attached to a person, a vehicle, or an airplane,to enable wireless communication among them

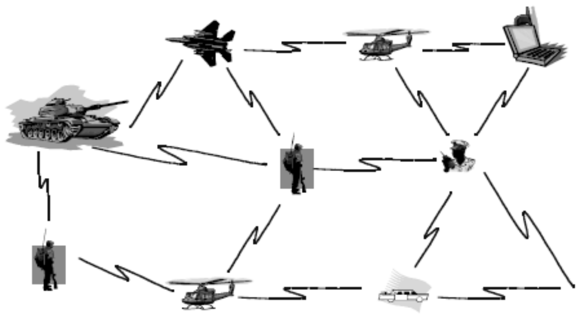


Figure 1.1 Overview of Mobile Ad-hoc Network

In MANET, a wireless node can be the source, the destination, or an intermediate node of data transmission. When a wireless node plays the role of intermediate node, it serves as a router that can receive and forward data packets to its neighbor closer to the destination node. Due to the nature of an ad-hoc network, wireless nodes tend to keep moving rather than stay still.

Therefore the network topology changes from time to time.

Wireless ad-hoc network have many advantages:

Low cost of deployment: Ad hoc networks can be deployed on the fly; hence no expensive infrastructure such as copper wires or data cables is required.

Fast deployment: Ad hoc networks are very convenient and easy to deploy since there are no cables involved. Deployment time is shortened.

Dynamic Configuration: Ad hoc network configuration can change dynamically over time. When compared to configurability of LANs, it is very easy to change the network topology of a wireless network.

MANET has various potential applications. Some typical examples include emergency search-rescue operations, meeting events, conferences, and battlefield communication between moving vehicles and/or soldiers. With the abilities to meet the new demand of mobile computation, the MANET has a very bright future.

1.2 Current challenges

In a mobile ad hoc network, all the nodes cooperate with each other to forward the packets in the network, and hence each node is effectively a router. Thus one of the most important issues is routing. This thesis focuses mainly on routing issues in ad hoc networks. In this section, some of the other issues in ad hoc networks are described:

Distributed network:A MANET is a distributed wireless network without any fixed infrastructure. That means no centralized server is required to maintain the state of the clients.

Dynamic topology: The nodes are mobile and hence the network is self-organizing. Because of this, the topology of the network keeps changing over time. Consequently, the routing protocols designed for such networks must also be adaptive to the topology changes.

Power awareness: Since the nodes in an ad hoc network typically run on batteries and are deployed in hostile terrains, they have stringent power requirements. This implies that the underlying protocols must be designed to conserve battery life.

Addressing scheme: The network topology keeps changing dynamically and hence the addressing scheme used is quite significant. A dynamic network topology requires a ubiquitous addressing scheme, which avoids any duplicate addresses. In wireless WAN environments, Mobile IP [10] is being used. Because the static home agents and foreign agents are needed, hence, this solution is not suitable for ad hoc network.

Network size: The ability to enable commercial applications such as voice transmission in conference halls, meetings, etc., is an attractive feature of ad hoc networks. However, the delay involved in the underlying protocols places a strict upper bound on the size of the network.

Security: Security in an ad hoc network is extremely important in scenarios such as a battlefield. The five goals of security – availability, confidentiality, integrity authenticity and non-repudiation - are difficult to achieve in MANET, mainly because every node in the network participates equally in routing packets.

1.3 Thesis target

The mobile ad hoc network is a new model of wireless communication and has gained increasing attention from industry. As in a general networking environment, mobile ad-hoc networks have to deal with various security threats. Due to its nature of dynamic network topology, routing in mobile ad-hoc network plays a vital role for the performance of the networks. It is understandable that most security threats target routing protocols – the weakest point of the mobile ad-hoc network. There are various studies and many researches in this field in an attempt to propose more secure protocols [1][2][6]. However, there is not a complete routing protocol that can secure the operation of an entire network in every situation. Typically a “secure” protocol is only good at protecting the network against one specific type of attacks. Many researchers have been done to evaluate the performance of secure routing protocols in comparison with normal routing protocols [1][4][6]. One of the objectives of this research is to examine the additional cost of adding a security feature into non-secure routing protocols in various scenarios. The additional cost includes delay in packet transmission, the low rate of data packets over the total packets sent, etc. It is well known that the real-world network does not operate in an ideal working environment, meaning that there are always threats and malicious actions affecting the performance of the network. Thus, studying the performance of secure routing protocols in malicious environments is needed in order to effectively evaluate the performance of those routing protocols. In the thesis, I have implemented two secure routing protocols: a secure version of the dynamic source routing - DSR (ARIADNE) [1] and Secure Ad hoc On-demand Distance Vector routing protocol (SAODV)[2] in the OPNET simulation environments [23]. I will also create malicious scenarios by implementing several attacks in the simulation environments. By implementing secure routing protocols and running these two routing protocols in malicious environments, I have evaluated those secure routing protocols, and have proposed solutions to remove the weaknesses and/or to improve the performance of these secure routing protocols

Classification of basic routing protocols:

Routing protocols in ad hoc mobile wireless network can generally be divided into three groups [2] (Figure 1.2)

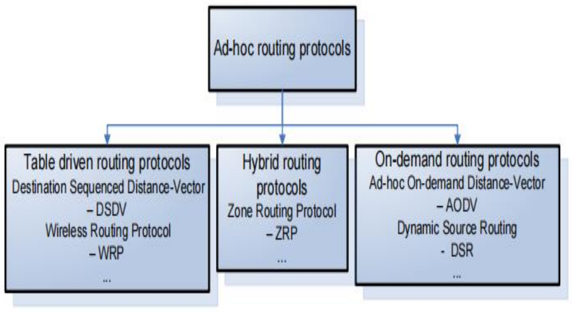

Ad-hoc routing protocols
Table driven routing protocols
Destination Sequenced Distance-Vector – DSDV
Wireless Routing Protocol – WRP
...
Hybrid routing protocols
Zone Routing Protocol – ZRP
...
On-demand routing protocols
Ad-hoc On-demand Distance-Vector – AODV
Dynamic Source Routing - DSR
...

Figure 1.2 Hierarchy of ad-hoc routing protocols

Table driven: Every node in the network maintains complete routing information about the network by periodically updating the routing table. Thus, when a node needs to send data packets, there is no delay for discovering the route throughout the network. This kind of routing protocols roughly works the same way as that of routing protocols for wired networks.

Source initiated (or demand driven): In this type of routing, a node simply maintains routes to active destination that it needs to send data. The routes to active destinations will expire after some time of inactivity, during which the network is not being used.

Hybrid: This type of routing protocols combines features of the above two categories. Nodes belonging to a particular geographical region or within a certain distance from a concerned node are said to be in the routing zone and use table driven routing protocol. Communication between nodes in different zones will rely on the on-demand or source-initiated protocols. In the rest of this chapter, I will give an overview of two of the most common routing protocols used in mobile ad hoc network: Dynamic Source Routing protocol (DSR) [8] and Ad hoc On-demand Distance Vector routing protocol (AODV) [9].

1.4 Dynamic Source Routing protocol (DSR)

The Dynamic Source Routing Protocol [8] is one of the on-demand routing protocols, and is based on the concept of source routing. In source routing, a sender node has in the packet header the complete list of the path that the packet must travel to the destination node. That is, every node in the path just forwards the packet to its next hop specified in the header without having to check its routing table as in table-driven routing protocols. Besides, the nodes don’t have to periodically broadcast their routing tables to the neighboring nodes. This saves a lot of network bandwidth. The two phases of the DSR operation are described below:

1.4.1 Route Discovery phase

In this phase, the source node searches a route by broadcasting route request (RREQ) packets to its neighbors. Each of the neighbor nodes that has received the RREQ broadcast then checks the packet to determine which of the following conditions apply: (a) Was this RREQ received before? (b) Is the TTL (Time To Live) counter greater than zero? (c) Is it itself the destination of the RREQ? (d) Should it broadcast the RREQ to its neighbors? The request ids are used to determine if a particular route request has been previously received by the node. Each node maintains a table of RREQs recently received. Each entry in the table is a <initiator, request id> pair. If two RREQs with the same <initiator, request id> are received by a node, it broadcasts only the one received first and discards the other. This mechanism also prevents formation of routing loops within the network. When the RREQ packet reaches the destination node, the destination node sends a reply packet (RREP) on the reverse path back to the sender. This RREP contains the recorded route to that destination. Figure 1.3 shows an example of the route discovery phase. When node A wants to communicate with node G, it initiates a route discovery mechanism and broadcasts a request packet (RREQ) to its neighboring nodes B, C and D as shown in the figure. However, node C also receives the same broadcast packets from nodes B and D. It then drops both of them and broadcasts the previously received RREQ packet to its neighbors. The other nodes follow the same procedure. When the packet reaches node G, it inserts its own address and reverses the route in the record and unicasts it back on the reversed path to the destination which is the originator of the RREQ. The destination node unicasts the best route (the one received first) and caches the other routes for future use. A route cache is maintained at every node so that, whenever a node receives a route request and finds a route for the destination node in its own cache, it sends a RREP packet itself instead of broadcasting it further.

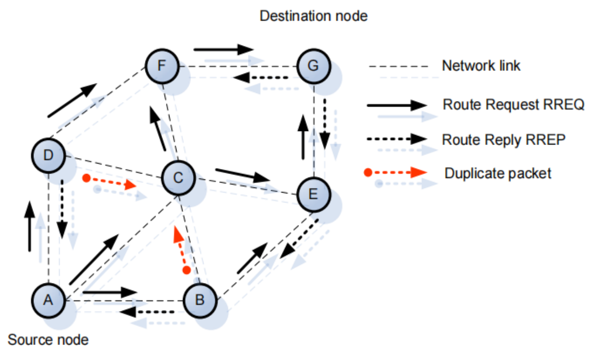


Figure 1.3 Route Discovery in DSR

1.4.2 Route Maintenance

The route maintenance phase is carried out whenever there is a broken link between two nodes. A broken link can be detected by a node by either passively monitoring in promiscuous mode or actively monitoring the link. As shown in Figure 1.4, when a link break (F-G) happens, a route error packet (RERR) is sent by the intermediate node back to the originating node. The source node re-initiates the route discovery procedure to find a new route to the destination. It also removes any route entries it may have in its cache to that destination node.

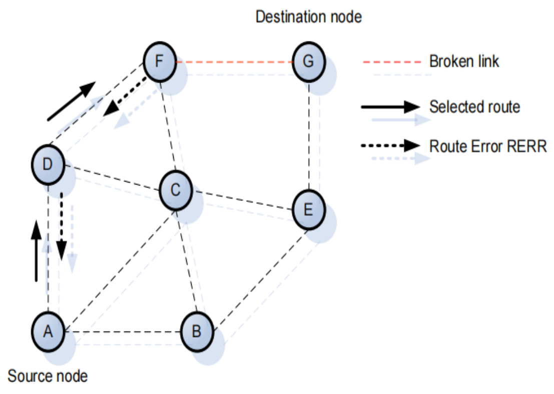


Figure 1.4 Route Maintenance in DSR

DSR benefits from source routing since the intermediate nodes do not need to maintain up-to-date routing information in order to route the packets that they receive. There is also no need for any periodic routing advertisement messages.However, as size of the network increases, the routing overhead increases since each packet has to carry the entire route to the destination along with it. The use of route caches is a good mechanism to reduce the propagation delay but overuse of the cache may result in poor performance [7]. Another issue of DSR is that whenever there is a link break, the RERR packet propagates to the original source, which in turn initiates a new route discovery process. The link is not repaired locally. Several optimizations to DSR have been proposed, such as non-propagating route requests (when sending RREQ, nodes set the hop limit to one preventing them from re-broadcasting), gratuitous route replies (when a node overhears a packet with its own address listed in the header, it sends a RREP to the originating node bypassing the preceding hops), etc. A detailed explanation of DSR optimizations can be found in [8]. 10

1.5Ad-hoc On-demand Distance Vector (AODV) routing protocol

To find routes, the AODV routing protocol [9] uses a reactive approach and to identify the most recent path it uses a proactive approach. That is, it uses the route discovery process similar to DSR to find routes and to compute fresh routes it uses destination sequence numbers. The two phases of the AODV routing protocol are described below.

1.5.1 Route Discovery

In this phase, RREQ packets are transmitted by the source node in a way similar to DSR. The components of the RREQ packet include fields such as the source identifier (SId), the destination identifier (DId), the source sequence number (SSeq), the destination sequence number (DSeq), the broadcast identifier (BId), and TTL. When a RREQ packet is received by an intermediate node, it could either forward the RREQ packet or prepare a Route Reply (RREP) packet if there is an available valid route to the destination in its cache. To verify if a particular RREQ has already been received to avoid duplicates, the (SId, BId) pair is used. While transmitting a RREQ packet, every intermediate node enters the previous node's address and its BId. A timer associated with every entry is also maintained by the node in an attempt to delete a RREQ packet in case the reply has not been received before it expires.When a node receives a RREP packet, the information of the previous node is also stored in it in order to forward the packet to it as the next hop of the destination. This plays a role of a "forward pointer" to the destination node. By doing it, each node contains only the next hop information; whereas in the source routing, all the intermediate nodes on the route towards the destination are stored.Figure 1.5 depicts an example of route discovery mechanism in AODV.Suppose that node A wishes to forward a data packet to node G but it has not an available route in its cache. It then initiates a route discovery process by broadcasting a RREQ packet to all its neighboring nodes (B, C and D).

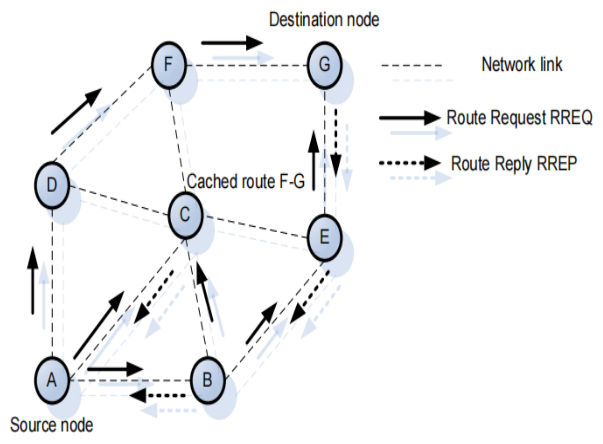


Figure 1.5 Route discovery in AODV

All the SId, DId, SSeq, DSeq, BId, and TTL fields are inserted in the RREQ packet. When RREQ packet reaches to nodes B, C and D, these nodes immediately search their respective route caches for an existing route. In the case where no route is available, they forward the RREQ to their neighbors; otherwise a comparison is made between the destination sequence number (DSeq) in the RREQ packet and the DSeq in its corresponding entry in the route cache. It replies to the source node with a RREP packet consisting of the route to the destination in the case the DSeq in the RREQ packet is greater. In Figure 2.4, node C gets a route to G in its cache and its DSeq is greater when compared with that in the RREQ packet. Consequently, it sends a RREP back to the source node A. By doing this, node A has already stored the path A-C-F-G. A RREP is also sent back by the destination node to the source. One possible route is A-B-E-G. The intermediate nodes on the path from source to destination make an update on their routing tables with the latest DSeq in the RREP packet.

1.5.2 Route Maintenance

The way that the route maintenance mechanism works is described below.Whenever a node finds out a link break (via link layer acknowledgments or Hello messages [9]), it broadcasts an RERR packet (in a way similar to DSR) to notify the source and the end nodes. This process is illustrated in Figure 2.5. If the link between nodes C and F breaks on the path A-C-F-G, RERR packets will be sent by both F and C to notify the source and the destination nodes. The main advantage of AODV is the avoidance of source routing to reduce the routing overload in a large network. Another good feature of AODV is its application of expanding-ring-search to control the flood of RREQ packets and search for routes to unknown destinations [10]. In addition, it also supplies destination sequence numbers, allowing the nodes to have more up-to-date routes.However, some notes have to be taken into consideration when using AODV.Firstly, it requires bidirectional links and periodic link layer acknowledgments to detect broken links. Secondly, unlike DSR, it needs to maintain routing tables for route maintenance unlike DSR.

2. SYSTEM ANALYSIS

2.1 EXISTING SYSTEM

Hacking attempts can be addressed if the pre-hacking step, called scanning, is properly investigated and good counter measures are in place. The importance of scanning appears is in providing sophisticated hackers with the necessary information about nominated victims' systems which eventually forms their hacking strategies. Therefore, this article proposes a security solution that aims to make scanning difficult by addressing its properties, which makes developing appropriate hacking strategies against protected computer networks unpractical. The proposed security solution protects computer networks by dynamically generating a unique protocol to replace the expected standard protocols and changing network paths periodically in order to confuse scanning attempts, as well as prevent unauthorized scanning and hacking traffic.

DISADVANTAGES

- This negatively affects energy-efficiency, reliability, and the operational lifetime of the network altogether.
- Discovered routes by these algorithms may neither be energy-efficient nor be reliable.
- No Provide any Data Secure Communication.

2.2 PROPOSED SYSTEM

Novel energy-aware routing algorithms to be proposed for wireless sensor networks, called reliable minimum Hybrid Dynamic Energy Routing Protocol. HDERP addresses three important requirements of ad hoc networks: energy-efficiency, reliability, and prolonging network lifetime. It considers the energy consumption and the remaining battery energy of nodes as well as quality of links to find energy-efficient and reliable routes that increase the operational lifetime of the network. HDERP, on the other hand, is an energy-efficient routing algorithm which finds routes minimizing the total energy required for end-to-end packet traversal. HDERP are proposed for networks in which either hop-by-hop or end-to-end re-transmissions ensure reliability. Hybrid Cryptography provides the hybrid cryptography method. Main objective of paper is exploring way of encryption done. Improve some aspects of the algorithm which is already existed and create way for the excellent security.

Implementation of encryption of the information is done in such a way that it will be impossible for the attackers to read the resources sent on the web. Advanced Encryption Standard (AES) and Elliptic Curve Cryptography (ECC) are the methods used for the encryption.

ADVANTAGES:-

- Combining two security features then improve security enhancements.
- Enhanced Attackers detection and prevention.
- Throughput and packet delivery ratio can be improved significantly, with slightly
- Reduced average end-to-end delay and Routing overhead of messages.

2.3 LIST OF MODULES

2.3.1 MANET Network Deployment

Each device in a MANET is free to move independently in any direction, and will therefore change its links to other devices frequently. Each must forward traffic unrelated to its own use, and therefore be a router. The primary challenge in building a MANET is equipping each device to continuously maintain the information required to properly route traffic

2.3.2 Data Communication

This Module is developed to MANET networks data communication and aggregation process. The radio and IEEE 802.11 MAC layer models were used. The network based data processing or most expensive and data communication level on their performance on the network. Multiple sources create and end sending packets; each data has a steady size of 512 bytes.

2.3.3 HDERP

HDERP is used since the network requires high reliability, higher life period and effective utilization of energy. Effective utilization of battery power in sensors nodes improves the life spam of the network. Reliability is ensured by the HDERP which determines the path of transmission of data in hop-by-hop. Hence by using HDERP reliability is ensured.

2.3.4 Hybrid Cryptography

Hybrid Cryptography provides the hybrid cryptography method. Main objective of paper is exploring way of encryption done. Improve some aspects of the algorithm which is already existed and create way for the excellent security.

2.3.5 Performance Analysis

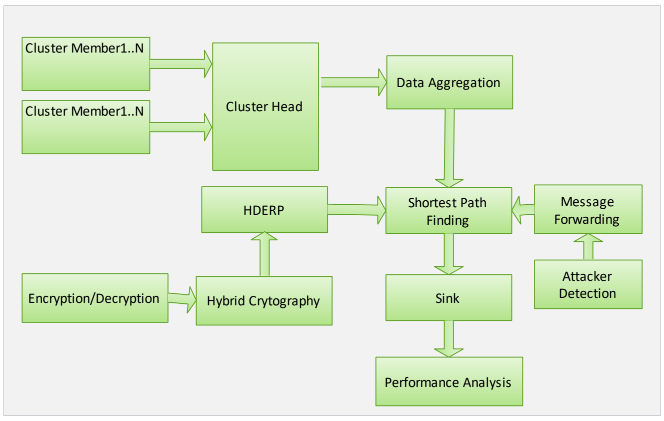
This module is developed to improve Wireless network performance, Reduce Average end –to-end delay.

2.4 DATA FLOW DIAGRAM

- The DFD is also called as bubble chart. It is a simple graphical formalism that can be used to represent a system in terms of input data to the system, various processing carried out on this data, and the output data is generated by this system.
- The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system.
- DFD shows how the information moves through the system and how it is modified by a series of

transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.

- DFD is also known as bubble chart. A DFD may be used to represent a system at any level of abstraction. DFD may be partitioned into levels that represent increasing information flow and functional detail.



2.1 System architecture

3.SYSTEM REQUIREMENTS

3.1 HARDWARE CONFIGURATION

PROCESSOR : INTEL PENTIUM IV

HARD DISK SPACE : 40GB

MONITOR: 14''

PRINTER: HP 1020

KEYBOARD : 104 KEYS

INTERNAL MEMORY CAPACITY : 256 MB

MOUSE: OPTICAL MOUSE

3.2 SOFTWARE CONFIGURATION

SOFFTWARE:NS-2

LANGUAGE:OBJECT ORIENTED TOOL COMMAND LANGUAGE(OTCL)

NS2 STRUCTURE INTRODUCTION

NS2 is an object oriented simulator, written in C++, with a Tcl interpreter as a front-end. The simulator supports a class hierarchy in C++ (also called the compiled hierarchy), and a similar class hierarchy within the Tcl interpreter (also called the interpreted hierarchy)

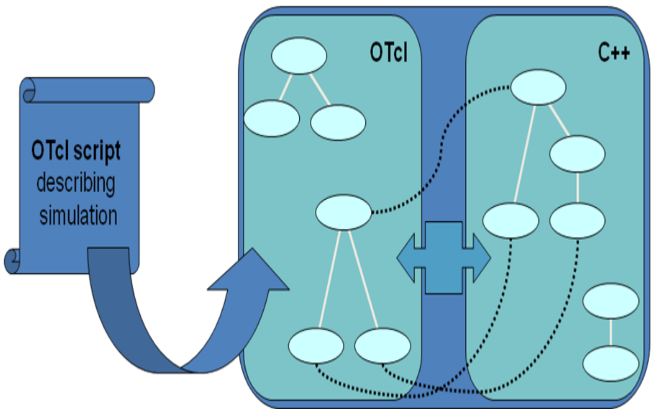


Figure 3.1: NS2 internal schematic diagram

The two hierarchies are closely related to each other; from the user's perspective, there is a one-to-one correspondence between a class in the interpreted hierarchy and one in the compiled hierarchy.

NS2 uses two languages because it has two different kinds of things it needs to do: Detailed simulations of protocols require a systems programming language which can efficiently manipulate bytes, packet headers, and implement algorithms that run over large data sets. For these tasks run-time is important and turn-around time (run simulation, find bug, fix bug, recompile, re-run) is less important. C++ is fast to run but slower to change, making it suitable for detailed protocol implementation. A large part of network research involves slightly varying parameters or configurations, or quickly exploring a number of scenarios. In these cases, iteration time (change the model and re-run) is more important. Since configuration runs once (at the beginning of the simulation), run-time of this part of the task is less important. Tcl runs slower than C++ but can be changed very quickly (and interactively), making it ideal for simulation configuration. Users create new simulator objects through the Tcl interpreter. These objects are instantiated within the interpreter, and are closely mirrored by a corresponding object in the compiled hierarchy. Class TclObject is the base class for most of the other classes in the interpreted and compiled hierarchies. Every object in the class TclObject is created by the user from within the interpreter. An equivalent shadow object is created in the compiled hierarchy. The two objects are closely associated with each other. The interpreted class hierarchy is automatically established through methods defined in the class TclClass. User instantiated objects are mirrored through methods defined in the class TclObject. Tcl /

C++ variable binding: Class InstVar defines the methods and mechanisms to bind a C++ member variable in the compiled shadow object to a specified Tcl instance variable in the equivalent interpreted object. The binding is set up such that the value of the variable can be set or accessed either from within the interpreter, or from within the compiled code at all times. Whenever the variable is read through the interpreter, the trap routine is invoked just prior to the occurrence of the read. The routine invokes the appropriate get function that returns the current value of the variable. This value is then used to set the value of the interpreted variable that is then read by the interpreter. Likewise, whenever the variable is set through the interpreter, the trap routine is invoked just after to the write is completed. The routine gets the current value set by the interpreter, and invokes the appropriate set function that sets the value of the compiled member to the current value set within the interpreter. The basic primitive for creating a node is: set ns [new Simulator] $ns node The instance procedure node constructs a node out of simpler classifier objects (to be discussed later). The Node itself is a standalone class in Tcl. However, most of the components of the node are themselves TclObjects.

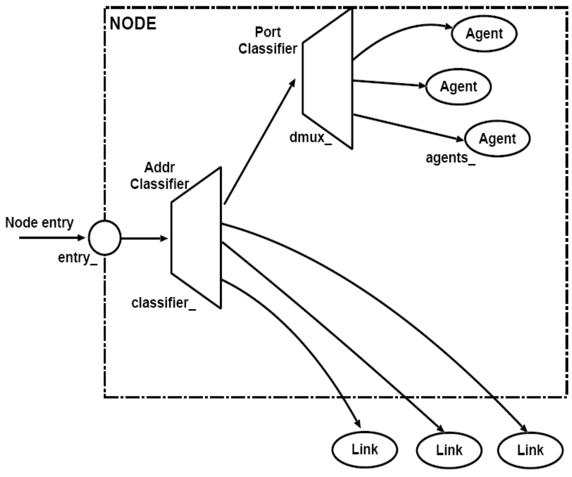


Figure 3.2: Node structure

This simple structure consists of two TclObjects: an address classifier (classifier_) and a port classifier (dmux_). The function of these classifiers is to distribute incoming packets to the correct agent or to correct outgoing link.

TRACE AND MONITORING SUPPORT

There are a number of ways of collecting output or trace data on a simulation. Generally, trace data is either displayed directly during execution of the simulation, or (more commonly) stored in a file to be post-processed and analyzed. There are two primary but distinct types of monitoring capabilities currently supported by the simulator. The first, called traces, record each individual packet as it arrives, departs, or is dropped at a link or queue. Trace objects are configured into a simulation as nodes in the network topology, usually with a Tcl "Channel" object hooked to them, representing the destination of collected data (typically a trace file in the current directory). The other types of objects, called *monitors*, record counts of various interesting quantities such as packet and byte arrivals, departures, etc.

SIMULATOR

The simulator is an event-driven simulator. The scheduler runs by selecting the next earliest event, executing it to completion, and returning to execute the next event. Unit of time used by scheduler is seconds. Presently, the simulator is singlethreaded and only one event in execution at any given time. If more than one event is scheduled to execute at the same time, their execution is performed on the FIFO manner (first scheduled – first dispatched). No partial execution of events or pre-emption is supported. An event generally comprises an event time, event id and a handler function.Two types of objects are derived from the base class Event - packets events and "at-events".Packets events will be discussed later in detail. An "at-event" is a Tcl procedure execution scheduled to occur at a particular time. This is frequently used in simulation scripts. A simple example of how it is used is as follows:

```
set ns [new Simulator]

$ns use-scheduler Heap

$ns at 300.5 "finish"
```

This Tcl code first creates a simulation object, then changes the default scheduler implementation to be heap-based, and finally schedules the function "finish" to be executed at time 300.5 (in seconds). In communication and computer network research, network simulation is a technique where a program models the behavior of a network either by calculating the interaction between the different network entities (hosts/routers, data links, packets, etc) using mathematical formulas, or actually capturing and playing back observations from a production network. The behavior of the network and the various applications and services it supports can then be observed in a test lab;various attributes of the environment can also be modified in a controlled manner to assess how the network would behave under different conditions. When a simulation program is used in conjunction with live applications and services in order to observe end-to-end performance to the user desktop, this technique is also referred to as network emulation.

Network simulator

A network simulator is a software program that imitates the working of a computer network. In simulators, the computer network is typically modelled with devices, traffic etc and the performance is analysed. Typically, users can then customize the simulator to fulfill their specific analysis needs. Simulators typically come with support for the most popular protocols in use today, such as WLAN, Wi-Max, UDP, and TCP.

Why Network simulation?

• Protocol validation

• controlled experimental conditions

• Low cost in $, time, collaboration, complexity.

NS Provides:

• Protocols: TCP, UDP, HTTP, etc.

• Traffic Models: Web Traffic, CBR,

• Topology Generation tools

• Visualization tools

• Large validation package (people believe it works)

NS Structure:

• C++ event scheduler protocols (most)

• TCL scripts protocols (mostly extensions to C++ core)

• TCL objects expose an interface to C++ objects (shadow objects) system

configuration (defaults, etc.)

Advantage

- Flexible and state of the art tool
- contains wide classes of internet protocols including Multicasting, SRM, RTP, ATM and wireless networks
- Widely used => respectful results + easy to compare

Disadvantages

- "alpha" quality
- Minimal docs
- Incomplete API27

Simulations

Most of the commercial simulators are GUI driven, while some network simulators require input scripts or commands (network parameters). The network parameters describe the state of the network (node placement, existing links) and the events (data transmissions, link failures, etc). Important outputs of simulations are the trace files.

Trace files can document every event that occurred in the simulation and are used for analysis. Certain simulators have added functionality of capturing this type of data directly from a functioning production environment, at various times of the day, week, or month, in order to reflect average, worst-case, and best-case conditions.

Network simulators can also provide other tools to facilitate visual analysis of trends and potential trouble spots.

Most network simulators use discrete event simulation, in which a list of pending "events" is stored, and those events are processed in order, with some events triggering future events -- such as the event of the arrival of a packet at one node triggering the event of the arrival of that packet at a downstream node.

Some network simulation problems, notably those relying on queuing theory, are well suited to Markov chain simulations, in which no list of future events is maintained and the simulation consists of transiting between different system "states" in a memory less fashion. Markov chain simulation is typically faster but less accurate and flexible than detailed discrete event simulation. Some simulations are cyclic based simulations and these are faster as compared to event based simulations.

Simulation of networks can be a difficult task. For example, if congestion is high, then estimation of the average occupancy is challenging because of high variance. To estimate the likelihood of a buffer overflow in a network, the time required for an accurate answer can be extremely large. Specialized techniques such as "control variates " and "importance sampling" have been developed to speed simulation.

4.SYSTEM DESIGN

4.1 INPUT DESIGN

Input Screen must be design in such a way to give an easy navigation throughout the screen without the violation of the input validation. Input design is the process of converting the user-originated data into a computer-based format. Inaccurate input data are the most common cause of error in data processing. The goal of an input data are collected and organized into a group and error free. Input data are collected and organized into a group of similar data. Once identified, appropriated input media are selected for processing. The design was done with six major objectives in mind

- Effectiveness
- Accuracy
- Ease o Use
- Consistency
- Simplicity
- Attractiveness

4.2 OUTPUT DESIGN

Designing computer output should proceed in an organized, well throughout manner; the right output must be developed while ensuring that each output element is designed so that candidates will find the system easy to use effectively. The term output refers to any effect produced by a system whether displayed or executed. When we design an output we must identify the specific output that is needed to meet the system. The usefulness of the new system is evaluated on the basis of their output. The output from the computer systems is required primarily to communicate the results of processing to users. An output generally refers to the result that is generated by the system. An application is successful only when it can produce efficient and effective reports. The reports generated must be useful for the management and for the future reference.

4.3 SYSTEM TESTING

TESTING PRINCIPLES

Before applying method to design effective test cases, a software engineer must understand the basic principles that guide software testing. Davis (DAV95) suggests a set of testing principles which have been adapted for use in this book. All tests should be traceable to customer requirements. Test should be planned long before testing begins. Test pare to principle applets to software testing. Testing should begin "in the small" and progress towards testing "in the page" Exhaustive testing is not possible.

Unit testing:

Unit testing focuses on verification errors on the smallest unit of software design-the module. Using the procedural design description as a guide, important control paths are tested to uncover errors within the boundary of the module. The module interface is tested to ensure that the information properly flows into and out of the program unit under test. Boundary conditions are tested to ensure that the module operates properly at the boundaries established to limit of restrict processing.

Integration testing:

Integration testing is a systematic technique for constructing the program structure while conducting test to uncover errors associated with interfacing. The objective is to take unit tested modules and build a program structure that has been dictated by design.

White box testing:

White box testing is some time is called glass box testing, is a test case design that uses a control structure of the procedural design to drive the test cases. Using white-box testing methods, the software engineer can drive test cases that Guarantee that logical decisions are on the true and false sides Exercise all logical decisions are on the true and false sides Execute all loops at their boundaries and within their operational bounds Exercise internal data structure to assure the validity

Acceptance testing:

Finally when the software is completely built, a series of acceptance tests are conducted to enable the client to validate all requirements. The user conducts these tests rather than the system developer, which can range from informal test drive to a planned and systematical executed series of tests. These acceptance tests are conducted over a period of weeks or months, there by uncovering cumulative errors that might degrade the system order time. In this process alpha testing and beta testing are used to uncover the errors that only the end user seems able to find.

Alpha testing:

The customer conducts the alpha test at the developer's site. The client notes the errors and usage problems and gives report to the developer. Alpha tests are conducted in a control environment.

Beta testing:

The beta testing is conducted at one or more customer's sites by the end users of the software. Unlike the alpha testing, the developer is not present. Therefore a beta test is a "live" application of the software in the environment that cannot be developed by the developer. The customer records all the problems encountered during the beta testing and reports these to the developers at regular intervals.

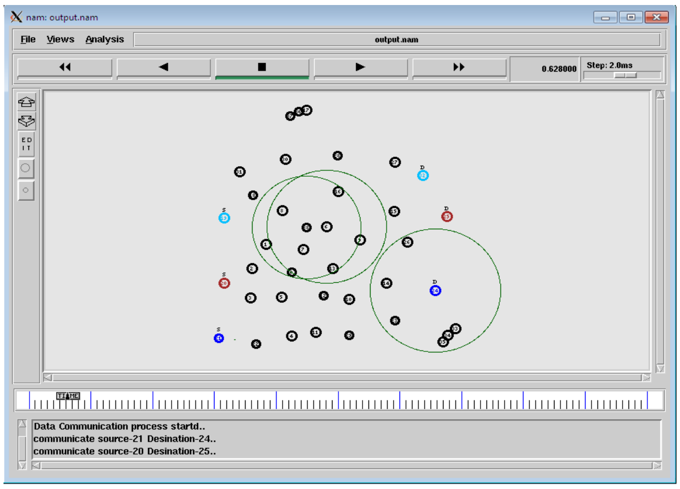
Black box testing:

Black box testing focuses on the functional requirements of the software. That is black box testing enables the software engineer to drive a set of input conditions that will fully exercise the requirements for a program. Black box testing is not an alternative for white box testing techniques. Rather, it is a complementary approach that is likely to uncover different class of errors.

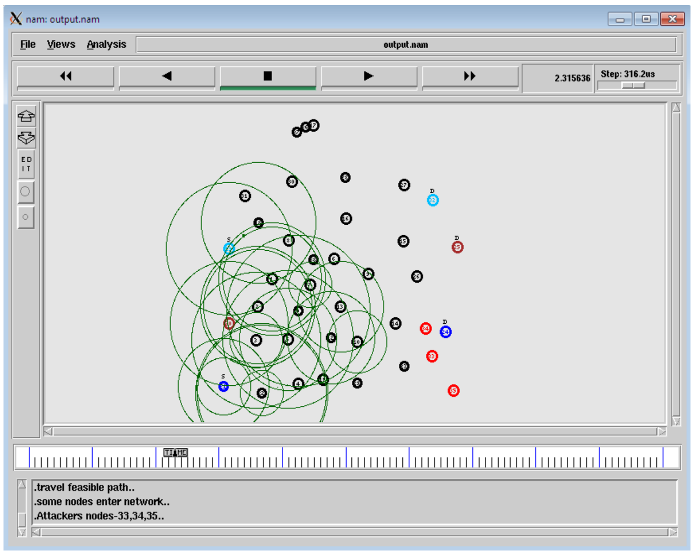
Black box testing attempts to find errors in the following categories:

- Interface errors.
- Performances in data structures or external database access.
- Performance errors.
- Initialization and termination errors.
- Incorrect or missing functions.
- All the above-mentioned errors were checked in the process of black.

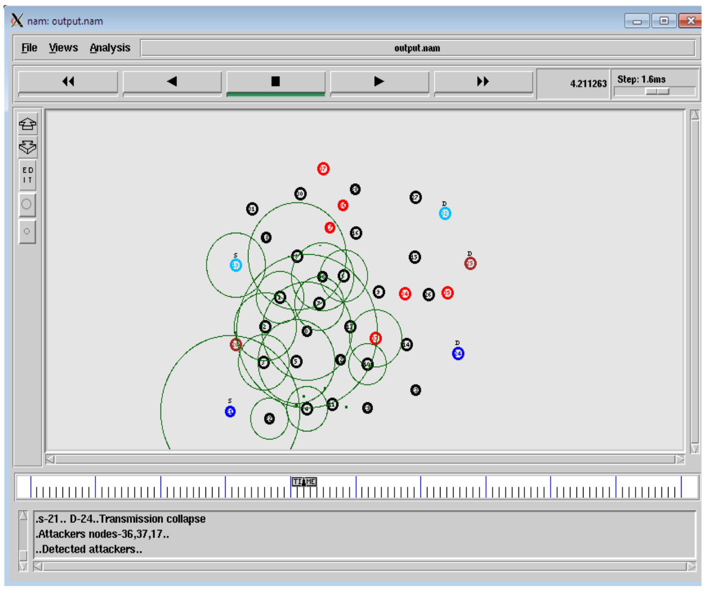
5. IMPLEMENTATION SNAPSHOT



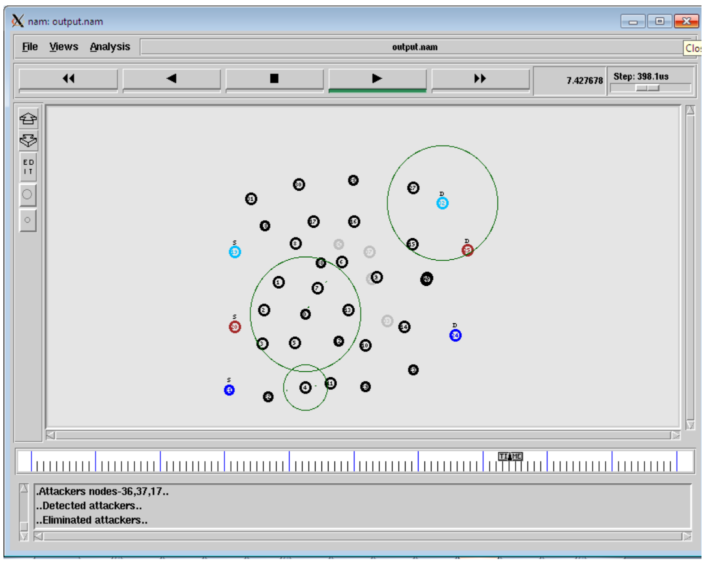
Data Communication process startd..
communicate source-21 Desination-24..
communicate source-20 Desination-25..



.travel feasible path..
.some nodes enter network..
.Attackers nodes-33,34,35..



```
.s-21.. D-24..Transmission collapse
.Attackers nodes-36,37,17..
..Detected attackers..
```



```
.Attackers nodes-36,37,17..
..Detected attackers..
..Eliminated attackers..
```

We proposed a unified trust management scheme that enhances the security of MANETs. Using recent advances in uncertain reasoning, Bayesian inference and Dempster-Shafer theory, we evaluate the trust values of observed nodes in MANETs. Misbehavior's such as dropping or modifying packets can be detected in our scheme through trust values by direct and indirect observation. Nodes with low trust values will be excluded by the routing algorithm. Therefore, secure routing path can be established in malicious environments. Based on the proposed scheme, more accurate trust can be obtained by considering different types of packets, indirect observation from one-hop neighbors and other important factors such as buffers of queues and states of wireless connections, which may cause dropping packets in friendly nodes. The results of MANET routing scenario positively support the effectiveness and performance of our scheme, which improves throughput and packet delivery ratio considerably, with slightly increased average end-to-end delay and overhead of messages. In our future work, we will extend the proposed scheme to MANETs with cognitive radios.

REFERENCES

1. R. Badonnel, R. State, and O. Festor.Self-configurable fault monitoring in ad-hoc networks. Ad Hoc Networks, 6(3):458–473, May 2008.

2. P. Bahl and V. N. Padmanabhan. RADAR: An in-building RF-based user location and tracking system. In Proc. of IEEE INFOCOM, 2000.

3. Y. Bar-Shalom, T. Kirubarajan, and X.-R. Li. Estimation with Applications to Tracking and Navigation. John Wiley & Sons, Inc., 2002.

4. D. Ben Khedher, R. Glitho, and R. Dssouli. A Novel Overlay-Based Failure Detection Architecture for MANET Applications. In IEEE International Conference on Networks, pages 130–135, 2007.

5. C. Bettstetter. Smooth is Better than Sharp: A Random Mobility Model for Simulation of Wireless Networks. In Proc. of ACM International Workshop on Modeling, Analysis and Simulation of Wireless and Mobile Systems, pages 19–27, New York, NY, USA, 2001. ACM.

6. C. Bettstetter. Topology Properties of Ad Hoc Networks with Random Waypoint Mobility. ACM SIGMOBILE Mobile Computing and Communications Review, 7(3):50–52, 2003.

7. J. Broch, D. A. Maltz, D. B. Johnson, Y.-C.Hu, and J. Jetcheva. A Performance Comparison of Multi-Hop Wireless Ad hoc Network Routing Protocols. In Proc. of MobiCom, pages 85–97, New York, NY, USA, 1998. ACM.

8. T. D. Chandra and S. Toueg. Unreliable Failure Detectors for Reliable Distributed Systems. Journal of the ACM, 43:225–267, 1996.

9. I. Constandache, R. R. Choudhury, and I. Rhee. Towards Mobile Phone Localization without War-Driving.In Proc. of IEEE INFOCOM, March 2010.

10. K. Dantu, M. H. Rahimi, H. Shah, S. Babel, A. Dhariwal, and G. S. Sukhatme.Robomote: enabling mobility in sensor networks. In Proc. of IEEE/ACM IPSN, 2005.

11. M. Elhadef and A. Boukerche.A Failure Detection Service for Large-Scale Dependable Wireless Ad-Hoc and Sensor Networks. In International Conference on Availability, Reliability and Security, pages 182–189, 2007.

12. K. Fall. A delay-tolerant network architecture for challenged internets. In Proc. of ACM SIGCOMM, pages 27–34. ACM, 2003.