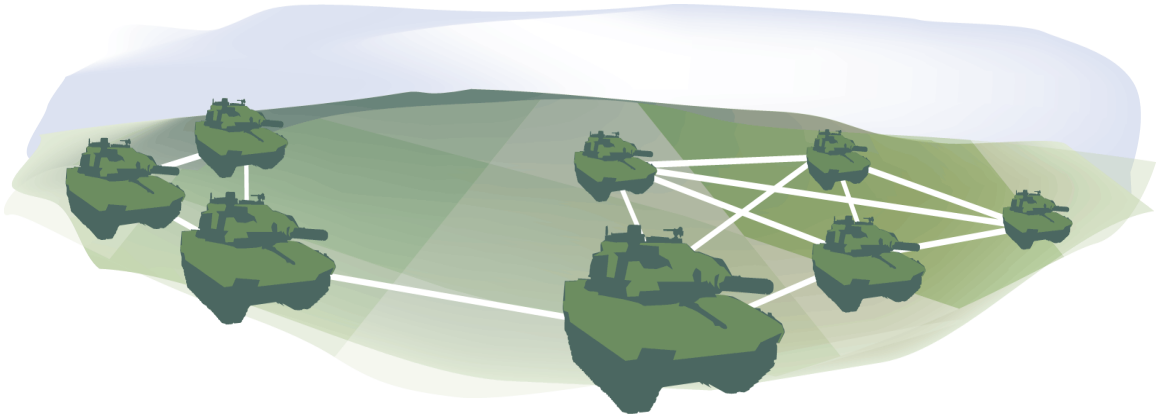
Specification-based intrusion detection combined with cryptography methods for mobile ad hoc networks

Elisabeth Hansson, Jimmi Grönkvist, Katarina Persson and Dan Nordquist



FOI-R--1867--SE
ISSN 1650-1942

Command and Control Systems
Technical report

December 2005

FOI is an assignment-based authority under the Ministry of Defence. The core activities are research, method and technology development, as well as studies for the use of defence and security. The organization employs around 1350 people of whom around 950 are researchers. This makes FOI the largest research institute in Sweden. FOI provides its customers with leading expertise in a large number of fields such as security-policy studies and analyses in defence and security, assessment of different types of threats, systems for control and management of crises, protection against and management of hazardous substances, IT-security and the potential of new sensors.

FOI
Defence Research Agency
Command and Control Systems
P.O. Box 1165
SE-581 11 Linköping

Tel: 013-378086
Fax:

www.foi.se

Specification-based intrusion detection combined with cryptography methods for mobile ad hoc networks

FOI-R--1867--SE
ISSN 1650-1942

Command and Control Systems
Tecnical report

December 2005

Issuing organization	Report number, ISRN	Report type
FOI – Swedish Defence Research Agency Command and Control Systems P.O. Box 1165 SE-581 11 Linköping	FOI-R--1867--SE	Technical report
	Research area code 4. C4ISTAR	
	Month year December 2005	**Project no.** E7075
	Sub area code 41 C4I	
	Sub area code 2	

Author/s (editor/s)	Project manager
Elisabeth Hansson Jimmi Grönkvist Katarina Persson Dan Nordqvist	Elisabeth Hansson
	Approved by Martin Rantzer
	Sponsoring agency Försvarsmakten
	Scientifically and technically responsible Jonas Hallberg

Report title

Specification-based intrusion detection combined with cryptography methods for mobile ad hoc networks

Abstract (not more than 200 words)

New challenges within the area of security have arisen due to a relatively new paradigm called mobile ad hoc networks. A mobile ad hoc network consists of wireless nodes that build a radio network without any pre-existing infrastructure or centralized servers. However, these networks have inherent vulnerabilities that make them susceptible to malicious attacks such as denial of service and propagation of incorrect routing information. Current security solutions for tactical radio networks, which mainly are based on cryptography, are not sufficient. We need to search for new solutions in order to obtain an acceptable level of security for tactical mobile ad hoc networks.

In this report, we examine the vulnerabilities of mobile ad hoc networks and argue that both cryptography solutions and intrusion detection must be included in mobile ad hoc networks. First, we present an architecture for intrusion detection that is applicable to mobile ad hoc networks. Second, we present an intrusion detection approach that detects attacks against mobile ad hoc networks. The key mechanism in this approach, specification-based detection, is evaluated through experiments. The experiments show that our specification-based model can detect attacks with high detection rates and few false alarms.

Keywords

Security, mobile ad hoc networks, intrusion detection and response, specification-based detection

Further bibliographic information	Language English
ISSN 1650-1942	**Pages** 27 p.
	Price acc. to pricelist

Utgivare	Rapportnummer, ISRN	Klassificering
FOI - Totalförsvarets forskningsinstitut Ledningssystem Box 1165 581 11 Linköping	FOI-R—1867--SE	Teknisk rapport
	Forskningsområde 4. Ledning, informationsteknik och sensorer	
	Månad, år December 2005	**Projektnummer** E7075
	Delområde 41 Ledning med samband och telekom och IT-system	
	Delområde 2	

Författare/redaktör	Projektledare
Elisabeth Hansson Jimmi Grönkvist Katarina Persson Dan Nordquist	Elisabeth Hansson
	Godkänd av Martin Rantzer
	Uppdragsgivare/kundbeteckning Försvarsmakten
	Tekniskt och/eller vetenskapligt ansvarig Jonas Hallberg

Rapportens titel (i översättning)

Policy-baserad intrångsdetektering för mobila ad hoc-nät

Sammanfattning (högst 200 ord)

Nya utmaningar inom IT-säkerhet har uppkommit på grund av en relativt ny systemtillämpning kallad mobila ad hoc-nät. Ett mobilt ad hoc-nät består av ett antal trådlösa noder, som bildar ett radionätverk utan fast infrastruktur och centraliserade funktioner. Dock är dessa nätverk sårbara för nya klasser av attacker t ex ”denial of service” och spridning av falsk routinginformation. Därmed är nuvarande säkerhetslösningar för taktiska radionätverk inte tillräckliga för mobila ad hoc-nät. Vi behöver ta fram nya säkerhetslösningar för att erhålla en acceptabel nivå på säkerhet i taktiska mobila ad hoc-nät.

I den här rapporten undersöker vi svagheter i mobila ad hoc-nät och argumenterar för att både kryptolösningar och intrångsdetekteringssystem behövs i mobila ad hoc-nät. Först presenteras en arkitektur som är applicerbar i mobila ad hoc-nät. Därefter presenteras en intrångsdetekteringsmetod som kan detektera attacker mot mobila ad hoc-nät. Huvudmekanismen i denna lösning, policy-baserad detektering, utvärderas med simuleringar. Experimenten visar att vår policy-baserade metod kan detektera attacker med hög noggrannhet och få falska larm.

Nyckelord

Säkerhet, mobila ad hoc-nät, intrångsdetektering med respons, policy-baserad detektering.

Övriga bibliografiska uppgifter	Språk Engelska
ISSN 1650-1942	**Antal sidor:** 27 s.
Distribution enligt missiv	**Pris: Enligt prislista**

Contents

1 Introduction

In recent years, with the rapid development and increased usage of wireless devices, security has become one of the major problems that wireless networks face today. A mobile ad hoc network is a wireless network that can be rapidly deployed as a multihop radio network without using any centralized functionality or fixed infrastructure such as base stations. Applications of mobile ad hoc networks include military operations and rescue work, as well as commercial applications like ad hoc-conferences.

1.1 Motivation

Securing mobile ad hoc networks is a challenge. A mobile ad hoc network has inherent vulnerabilities that make it susceptible to malicious attacks such as denial of service attacks, message replay, propagation of incorrect routing information, and physical compromise of nodes (see more on this in Chapter 2). Therefore, the traditional way to protect radio networks by cryptographic mechanisms, such as encryption and message authentication, is no longer sufficient. Cryptography can reduce the amount of successful intrusions, but cannot fully eliminate them. Encryption and authentication provide protection against attacks from external nodes and some internal attacks, but will not protect against many attacks from inside nodes, which already have the required keys [1].

Furthermore, it is difficult to design and implement software systems without introducing design and programming errors that an adversary can exploit. If an adversary has adequate resources and tries hard enough, there is a risk that the adversary succeeds in infiltrating the system. History has taught us that no matter how many security mechanisms (e.g. encryption, authentication and firewalls) are inserted in the network, there are always weak links that adversaries can exploit. For example, even though buffer overflow has been a known security problem for many years, there is still recently released software with buffer overflow security holes. If the buffer overflow security hole is exploited it may lead to an unauthorized root shell. In other words, someone can infiltrate the system.

Hence, to obtain an acceptable level of security in military contexts, traditional security solutions should be coupled with *intrusion detection systems* (IDS) that continuously monitor the network and determine whether the system (the network or any node of the network) is under attack. Once an intrusion is detected, e.g. in the early stage of a denial of service attack, a response can be put into place to minimize damage.

Intrusion detection can be classified into three broad categories [2]: anomaly detection, misuse (signature) detection, and specification-based detection. *Anomaly detection* recognizes deviations from normalcy by building models of normal behavior. Any deviation from normal is identified as an attack. *Misuse detection* use patterns of known attacks to recognize intrusions. *Specification-based detection* detects attacks with use of a set of constraints (rules) that define the correct operation of a program or a protocol.

Misuse detection has high detection accuracy and low false alarm rate for known attacks, but it is unable to detect novel attacks whose signatures are unknown. The ability to detect previously unknown attacks is essential in military contexts, since some military organizations have resources to develop specific targeted attacks that are unknown and not used in civilian contexts. Anomaly detection is able to detect unknown attacks. However, anomaly detection techniques also produce a high degree of false alarms, which is not acceptable for an intrusion detection system for a military network. Specification-based detection is similar to anomaly detection in

that it is able to detect unknown attacks. The main advantage of specification-based methods is that it provides the capability to detect previously unknown attacks, while providing a low false positive rate, i.e., few false alarms. However, the specifications are usually derived manually from RFCs or other descriptions of protocols. Thus, an obvious downside is that the development of specifications may be time-consuming and protocol specific.

1.2 Overview of solution

Given the complementary nature of the strengths and weaknesses of prevention approaches, such as authentication, and intrusion detection approaches, a natural approach is to combine the two approaches in such a way that we can realize the combination of their strength, while avoiding the weaknesses of either one. In this report, we combine authentication and specification-based detection in order to protect against known and unknown attacks.

To illustrate our approach, we combine authentication with novel a developed specification-based method that detects attacks against the standardized Ad hoc On-demand Distance Vector (AODV) routing protocol [3]. This is achieved by modeling the AODV protocol with the extended finite state machine method [4]. It is suitable to illustrate the approach for routing protocols, since these protocols implement typical characteristics of mobile ad hoc networks and are vulnerable to malicious attacks. We demonstrate the effectiveness of our approach with experiments. The results of the experiments show that it is possible to detect attacks with high detection rate and few false alarms.

1.3 Outline

The structure of the report is as follows. Related work on IDS for ad hoc networks is described in Chapter 2. Chapter 3 describes vulnerabilities in mobile ad hoc networks and attacks against the AODV routing protocol. An architecture for intrusion detection is described in Chapter 4. Chapter 5 describes the specification-based approach along with the prevention approach. Experiments and results are also described in Chapter 6. Conclusions and future work are given in Chapter 7.

2 Related Work

text Most protocols for ad hoc networks have been developed without any consideration for security, assuming that all nodes trust each other; specifically we can here mention the few ad hoc network protocols that have reached the standardization phase, i.e. AODV [3], DSR [15], OLSR [18], and TBRPF [22]. The few protocols for ad hoc networks developed with consideration for security can rather be seen as patches to the existing protocols, e.g. ARIADNE (for DSR) [10] and SAODV (for AODV) [28].

In this report we have chosen to focus on AODV, an attempt to secure this protocol have been done in SAODV, by letting each node sign the routing messages with a private key, this prevents nodes from sending most false messages. In addition, when a node replies to a route request it also adds a signature from the destination to prove that it has a route. Finally, hash chains are used in order to protect the part of a message that intermediate nodes will change on each hop (more on this in chapter 3.1.5).

However, even if SAODV can protect against several attacks it still has problems. One problem is a node that floods the network with large numbers of route requests but still correctly signs these messages. Another problem is connected to the key management, the present suggestion [28] is that each node picks an asymmetric key pair and from this deterministically generates an IP address. This does not only give huge overhead when the public keys need to be sent over the network to all nodes, but that transmission can also be attacked. There is also the additional problem with mapping an IP address to an identity. Which nodes should be allowed into the network?

Although, these protocols give a considerable improvement to security as compared to the original protocols their implementation is difficult and there are still unresolved security issues. Pure intrusion prevention methods will not be sufficient in mobile ad hoc network, therefore during the last few years the research community has put considerable attention to the area of intrusion detection.

One approach to solve this is to generalize intrusion prevention in order to make networks more resilient towards networks attacks. In [20] an architecture called Techniques for Intrusion-Resistant Ad Hoc Routing Algorithms (TIARA) is suggested, using firewalls, distributed traffic policing and other methods minimizing the effect of a malfunctioning or malicious node. However, this requires a major modification of existing algorithms, which makes it unlikely to be used in the near future.

One of the first papers on intrusion detection for ad hoc networks was published in the year 2000 [12]. In that paper, a basic architecture was described. The approach is based on IDS agents on every node that each performs local detection on local data. Global detection can be initiated when a node reports an anomaly. How the actual detection and verification should be handled was not addressed though.

In [19] this was used and expanded specifically for AODV. Cooperation between the nodes was assumed and a threshold for malicious nodes was defined, if the threshold is passed a node can initiate a response, and if two or more nodes respond to the same node that node can be purged from the network.

In [21] a so-called watchdog approach is described, in this the nodes study their neighbors to see if they are relaying packets correctly. For example, after a node sends a routing packet it will listen on the channel to see whether that neighbor retransmit the packet correctly without incorrect modifications within a specified time. If a node fails to do this within a certain time the node is marked as misbehaving, such a node is then avoided when a path is chosen.

The problem with this solution is that nodes do not necessarily receive their retransmitted packets. For example, this is the case when nodes are using power control and the next hop requires less power than the first hop. However, it may also have problems in fixed power transmissions since packets may be lost due to collisions and some medium access control protocols, e.g. STDMA, does not guarantee correct reception on any node but the designated receiver. (It may still be a compliment perhaps, since few other methods can determine whether another node randomly drops a packet. How responses should be taken must be considered carefully though since false and missed alarms will be common.)

Several suggestions for using mobile agents also exists [24, 25], in these protocols mobile agents move around from node to node and collect and study information that have been gathered locally in each node. This has the advantage that the mobile agent can take decisions based on the collected information from many nodes without having to send all information into a central node.

The problem with these methods is that how the mobile agent itself should be protected is unclear. If the agent is moved to a malicious node, that node cannot be allowed to change the agent, something that seems difficult in practice.

Traditionally the most used method for IDS in fixed networks have been misuse detection, however, for ad hoc networks most researchers have concentrated on anomaly detection. One motivation is that the data bases for misuse detection will be too expensive for a small ad hoc node, another motivation is that much financing to ad hoc networking research comes from the military sector, in which the ability to detect unknown attacks is essential. As previously mentioned though, anomaly detection gives a large number of false alarms and better methods would be preferable.

For specification-based IDS less work on ad hoc networks have been done. In [27] a finite state machine is used for correcting routing behavior for AODV, but their solution requires several assumptions that make it unsuited for our scenarios. The most problematic assumption is that they assume that MAC addresses cannot be forged (this is then used to control the origin of a message).

In [28], specification-based IDS is combined with anomaly-based IDS for AODV, but the anomaly part of the IDS incurs a large number of false alarms that makes it difficult to use in our scenarios.

3 Security vulnerabilities

Some of the critical requirements that mobile ad hoc networks are designed to meet, such as flexibility and robustness, naturally come at the cost of higher security challenges. In addition to the security threats common in fixed networks, some characteristics of mobile ad hoc networks impose further vulnerabilities. A tactical mobile ad hoc network is vulnerable to attacks because of its poor physical protection, wireless links, dynamic network topology, the property that each node is a router, the property of self-configuration, distributed algorithms, and lack of a clear line of defense. These vulnerabilities are described in Section 3.1. AODV routing protocol vulnerabilities are described in Section 3.2.

3.1 Vulnerabilities in mobile ad hoc networks

Medium access is simple in wireless mobile ad hoc networks. The *wireless link* allows passive eavesdropping but also simplifies active impersonation, message replay and message distortion [5]. Eavesdropping might give the adversary access to confidential information. Active attacks might results in, e.g., impersonation of a node or disruption of the communication in the network.

Mobile nodes in a hostile environment have *poor physical protection* [6]. A mobile node can be stolen or hijacked or an intruder can penetrate the security mechanisms and perform attacks from the node by injecting, e.g., a Trojan. The possibility of compromised nodes performing internal attacks is probably one of the most severe threats to a mobile ad hoc network. Thus, we must not only consider attacks from external nodes, but also take into account attacks from internal compromised nodes.

To achieve an autonomous ad hoc network many protocols developed for mobile ad hoc network are based on *distributed algorithms*. These distributed algorithms enable new types of attacks, since the algorithms are based on the cooperative participation of nodes. If one node is malicious, it can affect the entire network. For example, although there are many MAC protocols, their basic working principles are similar. In a contention-based MAC protocol, nodes must follow rules to reduce or avoid collisions. A malicious node not obeying the rules can create unfairness and congestion in the network. In a contention-free MAC protocol, each node must obtain an agreement from all other nodes to use the channel resource. In both cases, a malicious node not cooperating can create disruption of the network [7, 8].

IP address auto configuration introduces vulnerabilities. For example, the IPv6 stateless address auto configuration [11] and the similar auto configuration principle above IPv4 are vulnerable to false replies by malicious nodes. Both methods rely on the verification that a particular address is not already used by performing Duplicate Address Detection (DAD) [9]. A malicious node can pretend to use any of the address chosen by the incoming node, thus denying it the right to join the network.

Moreover, the fact that *each node is a router* also causes threats to all nodes. To reach the destination, the route from a node takes the packets across the network through various unknown nodes. Thus, an intermediate node can perform arbitrary man-in-the-middle attacks such as eavesdrop, modify and drop packets.

An additional threat, related to each node being a router, comes from attacks against the distributed routing protocols developed for mobile ad hoc networks. Routing attacks can be classified as routing disruption attacks or resource consumption attacks [10]. In routing

disruption attacks, packets are routed improperly whereas resource consumption aims at using up resources such as bandwidth, memory and computation capacity.

Finally, compared to a wired network, the mobile ad hoc network is vulnerable to several external attacks, since there is *no clear line of defense* such as firewalls. In wired networks, attacks against a node normally have to pass at least one centralized security mechanism, e.g. a firewall, whereas attacks against a mobile ad hoc network target the node directly. For example, the centralized firewall in wired networks can protect against several data link-layer attacks.

3.2 AODV vulnerabilities

AODV is a standardized routing protocol designed for mobile ad hoc networks [3]. The algorithm is on-demand. That is, routes between nodes are built when the source node needs them. AODV uses three routing packets to build a route to a destination: Route Request (RREQ), Route Reply (RREP) and Route Error (RERR). When a node does not have a route to a destination, which it want to communicate with, it broadcasts a *Route Request* in which it ask for a route to the destination.

When a node receives a Route Request, it sends a *Route Reply*, if the node is the destination node. An intermediate node can also respond to the request, if it has a fresh route to the specified destination in its route table. A route is considered fresh if the sequence number in the Route Request is lower than the corresponding value in the routing table or the sequence numbers are equal but the hop count is smaller.

The route is maintained as long as the route remains active. A route is considered active if data are sent from the source node to the destination node. If a link break occurs in an active route, a *Route Error* is propagated to the source node. Vulnerabilities of AODV are described in Table 1 below.

Type of attack	Attack description	Misuse goal
False Message Propagation of RREQs	1. An adversary impersonates a node and sends RREQs with the originator IP address of the other node in order to create route disruption. An adversary may also falsify other fields of the RREQ, e.g., false hop count in order to create false network picture.	Route disruption. For example, redirect traffic or disrupt communication.
False Message Propagation of RREPs	2. The node sends a forged RREP with an existing false originator IP address even though the node have not received any related RREQ.	Route disruption.
False route reply	3. These attacks are carried out by falsified reply to a valid RREQ. A malicious node advertises a route in the RREP to a node with a false destination sequence number (e.g. greater than the authentic value) or falsified value of the hop count.	Route disruption or suboptimal tour.
Rushing	4. To limit the overhead, each node typically forwards only one RREQ originating from any Route Discovery (identified by RREQ identity). In the rushing attack, an adversary sends RREQs with a false originator IP address and guessed value of the RREQ identity in order to suppress later transmitted legitimate RREQs from the impersonated node [15].	The misuse goal of the attack is to prevent a new route from being established.
Modification of routing messages	5. A malicious node modifies a field in a received RREP (or RREQ) and then forwards it to its neighbours.	Route disruption.
Resource depletion attack	6. Resource depletion refers to consuming the communication bandwidth in the network, computer capacity or storage space at individual nodes. This can be achieved by flooding the network with RREQs or RREPs.	Resource consumption.
Dropping of routing packets	7. The attacker simply drops (all or some) received routing messages.	No consequence, suboptimal tour or divided network.
Routing table overflow	8. A malicious node floods the network with non-existing routes by sending RREQs with non-existing originator addresses or RREPs with non-existing destination addresses.	Route disruption.
Modify routing table	9. An adversary modifies the routing table.	Route disruption.
Maintenance attack	10. A malicious node propagates false RERR messages.	Route disruption.

Table 1: AODV vulnerabilities

4 An architecture for intrusion detection

Intrusion detection in mobile ad hoc networks should be distributed to suit the architecture and features of a mobile ad hoc network [12]. We propose an architecture where every node is responsible for detecting local intrusions by listening on the receiving and transmitting interface of the node. Thus, all nodes contain an intrusion detection system (IDS) agent. The IDS agent will detect if neighbor nodes and other nodes perform attacks by using specification-based detection without interaction of other nodes.

4.1 The IDS agent

Even though the IDS agent is fairly complex it can be structured in five modules, see figure 1.

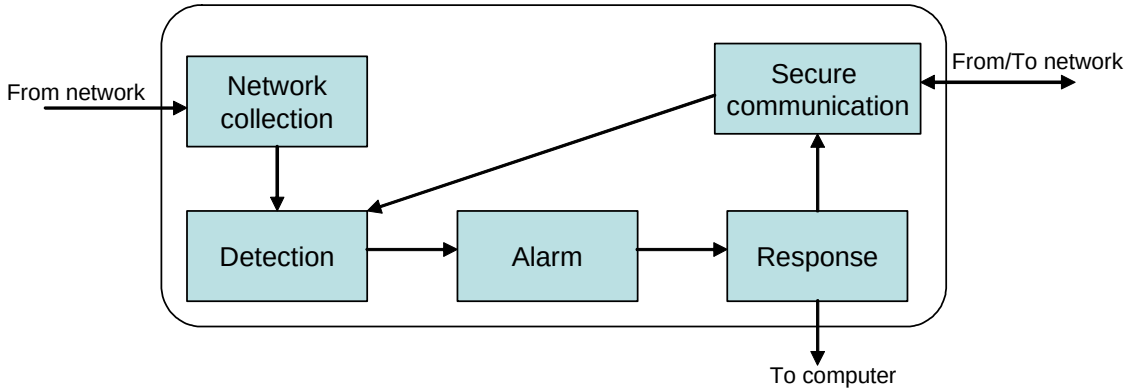


Figure 1: IDS agent

4.1.1 Network collection module

The network collection module gathers packets received from neighbor nodes and forwards the selected packets to the detection module. Likewise, packets transmitted from the node to other nodes are gathered and forwarded to the detection module.

4.1.2 Detection module

The ability to detect previously unknown attacks is essential in military contexts, since some military organizations have resources to develop attacks that are unknown and not used in civilian contexts. Thus, the aim is to provide a method that detects previously unknown attacks. Anomaly detection can detect previously unknown attacks, but also results in many false alarms. Experiments in wired networks show that specification-based detection may provide the capability to detect previously unknown attacks, while providing a low false positive rate, i.e., few false alarms [13, 14]. Thus, our detection module detects attacks with specification-based detection. In other words, the detection module processes local data received from the network collection module to detect intrusions. The approach is illustrated for AODV in Chapter 5.

4.1.3 Alerting module

The alerting module takes input from the internal detection module. The information of intrusions from the detection module is processed to categorize the attacks according to their consequence and minimize the number of false alarms.

4.1.4 Response module

We suggest that nodes perform responses without collaboration with other nodes, since it is still an open problem whether collaborative responses can be implemented without weakening the security in the ad hoc network. Responses can be local, only involving the detecting node itself, while global responses are directed to all nodes in the ad hoc network. Global responses are only allowed to be initiated from a centralized IDS server in the network. In some situations, the IDS server cannot be used, for example due to bad transmission conditions, low network capacity or radio silence mode. Then only local responses can be performed. The choice of response depends on the type of attack.

Responses can be classified as weak or strong, depending on their efficiency and their impact on the network, power or user resources. When an intrusion is detected, the node can either initiate weak local responses or inform a centralized IDS server about the attack if it is accessible.

Example of weak local responses
Local responses can be fast since less communication between nodes is needed. (The centralized node needs evidence from many nodes to make sure that it really is an attack.) Preferably, local responses are performed at a lower layer or at least within the same layer as the attack. Responses can be performed within the attacked protocol by configuring the protocol. For example, the AODV routing protocol can be configured to disallow route replies by intermediate nodes by setting the destination only flag in route requests. This could be an appropriate response if we suspect that an intermediate node has performed a routing attack. Other examples of weak local responses are to drop packets before they are processed by the MAC layer (in case of collision attacks or other MAC-layer attacks) or to warn the end user of the intrusion. If the node is not needed for a while, the user can then decide to restart the node in a safe mode for maintenance and system analysis. Whether on-line maintenance can be performed at the same time as the node is used must be studied further.

Example of strong global responses
From Zhang & Lee [12] and Mishra [5], we have three examples of strong responses:

- As a result of a forced re-keying at the wireless interface, the wireless network will be reinitialized and the malicious node cannot communicate with the other nodes any more.
- Identify the compromised nodes and force a reorganization of the routing paths in the network in order to avoid malicious traffic.
- All users in the network are requested to re-authenticate themselves.

4.1.5 Secure communication

The IDS agents may want to transmit information to other IDS agents. Our IDS agent transmits IDS messages to verify the information in a route reply (see more on this in Section 5.4, false route reply). The secure communication module provides a high-confidence communication channel among IDS agents.

In order to protect the information in an IDS message, we can divide it into two parts.

First, the protection of the information in a message that nodes on the path are not supposed to change, this includes for example source and destination addresses, and other information from the source. To protect this information from being changed by an intermediate node we assume the use of hash-based message authentication codes (HMAC), see e.g. [23]. In HMAC an

authentication tag for a message is generated by applying a hash function on the message and encrypting the result with a secret key, this so-called tag is then added to the message. Since only the destination must be able to authenticate the message the secret key can for example be a shared symmetric key between origin and destination. More advanced key structures may also be possible, e.g. using the private key in an asymmetric keys pair, but they are not necessary here. However, if asymmetric keys are used the added tag is usually denoted as a digital signature. We will discuss this more in the next chapter where also broadcast messages need to be authenticated.

The second part is information that needs to be changed in each intermediate hop. The only information that this is relevant for is the number of hops (from TTL) because this information is needed to determine if the route is as good as it should be. There are some properties regarding the number of hops field that makes it possible to protect this information.

First, each node on the path is supposed to decrease the TTL value by one.

Second, the main attack of malicious nodes will be to increase the value of TTL, thereby making the path look shorter and more likely to be used. Decreasing the value of TTL gives very little gain, since this can only lead to longer routes than necessary. (Without the malicious node though).

Therefore, we only need to make sure that the malicious node does not increase the TTL field. One way to do this is the use of hash chains. A hash function is in essence a one way function, i.e. we can easily calculate the output from applying the hash function to a value, but we cannot from the result of this reclaim the original value.

To clarify, if $y = h(x)$ where h is a hash function. We can easily determine y from x but not the opposite. Someone with y cannot determine the value of x. A hash chain can be described as applying the hash function on the same value multiple times, i.e. $x, h(x)$, $h(h(x))$, $h(h(h(x)))$, $h(h(h(h(x))))$... and so further on. We simplify the notation by writing this as $x, h(x)$, $h^2(x)$, $h^3(x)$, $h^4(x)$ …

When a user creates the IDS message, the TTL value in this is set to the maximum number of hops the message should be transmitted ($\max TTL$). This is either a fixed value or this information is also included in the unchanging part of the message.

The source then sends the hashed value of a seed x, that is $h(x)$, and $h^{\max TTL}(x)$. Then each node on the path changes this value by applying the hash function on the previous value. If it is z number of hops from source to destination the arrived value will be $d = h^z(x)$. The destination can check if the route has correct length by testing that $h^{(\max TTL - z)}(d)$ is equal to $h^{\max TTL}(x)$. The last value is unchanging and can be protected as described above.

A malicious node can possibly resend the value without changing it but it cannot find out the values before previous hops, which would be required in order to decrease the number of hops and thereby make the route seem better. (The malicious node could perhaps increase the length of the route just to make the IDS declare the route is incorrect, though, i.e. behave correctly in routing, then destroy all routes passing through it. The gain of this is not so great though, since only paths through the malicious node could be affected.)

5 A specification-Based Approach

Specification-based detection defines a set of rules that describe the correct operation of a protocol, and monitors the execution of the protocol with respect to the defined rules. In other words, the specification-based approach provides a model of the protocol in order to detect attacks based on the protocol specification. The specifications are usually derived manually from RFCs or other descriptions of protocols. Thus, a challenge in specification-based detection is how to define the set of rules that describe the correct operation of the protocols in order to detect attacks efficiently. In this report, this is achieved by modeling the protocol with the extended finite state machine method.

To illustrate our approach, we present an extended finite state machine model of the standardized Ad hoc On-demand Distance Vector (AODV) routing protocol. It is suitable to illustrate the approach for routing protocols, since these protocols implement typical characteristics of mobile ad hoc networks and are vulnerable to attacks. However, we believe that the principle behind the approach is general and applicable to other protocols as well, even though the protocols may differ in the format.

5.1 Assumptions

In order to narrow the scope of the problem, we make the following assumptions.

- Bidirectional communication is assumed.

  The method to detect attack 2 in table 1 (false RREP without any related RREQ) assumes that a node that receives a RREP also has transmitted the related RREQ. Thus, bidirectional communication is assumed.

- It is assumed that every sent RREP is coupled with the HMAC of the related RREQ.

  This makes it possible to identify if the RREP is an authentic gratuitous RREP, since it is possible to verify that the RREQ had the gratuitous flag set.

- Authentication and integrity protection of routing messages is assumed, see below.

- Assumption about key management is described, see below.

Authentication and integrity protection of route messages

For the routing messages, we assume the following:

Each message is protected using the same method as described for IDS messages in the previous chapter, i.e. the unchanging information is protected using HMAC (or digital signatures) and the hop count information is protected by using hash chains.

However, the difference here is that we no longer have point-to-point messages where only one destination needs to authenticate and verify the content. Route request messages are broadcasted, so every node needs to be able to authenticate them. For routing messages, all intermediate nodes should be able to authenticate the information as well. This will complicate the problem somewhat, since shared keys between all node pairs no longer is an option. At present, there are two solutions to this problem.

The first solution, use of asymmetric keys, assumes that all nodes have access to the public keys of all the other nodes (see key management below). This is the solution used in SAODV. A drawback is that asymmetric keys require much more processor overhead than symmetric keys, which may be a serious drawback in hand held devices and even more so in sensor networks. In vehicle-based networks, the incurred delay may be more of an issue.

The second solution is to generate symmetric key chains with help of hash chains as done in TESLA [32]. This solution is used in ADRIANE, which is a secure version of DSR. In TESLA, a key chain is generated by starting with one key and applying a hash function on this repeatedly. These keys are then used in reverse order, that is $y_i = h^{(N-i)}(x)$. The keys are "published", i.e. sent out to the network at certain times. Assuming that all sources have an authenticated key y_0 to start with, key chains can be used to authenticate and validate a message with multiple recipients.

Each sent message contains a HMAC using the next still "unpublished" key. When a message is received, a node will check if the key used is unpublished. If it is already published, the message will be discarded; if it is "unpublished", the message can be correctly authenticated when the next key is published since only the source could have generated this message.

A problem here is that the next key cannot be published until the packet has reached all destinations, otherwise the packet must be discarded. This means that intermediate nodes cannot authenticate (or discard) the message before retransmitting it. As a result, RREQ messages will always reach the full destination set, even if they all should discard the packet afterward, when the key will be published.

In addition, a new key needs to be released before a route reply can be sent, this key should only be known by the source of the route request and need to be broadcasted in the network. Nodes can authenticate the new key by applying the hash function and see that they get the same value as the old authenticated key.

This both increases delay and increases communication overhead, since two broadcast messages need to be sent, but it does have the advantage of using symmetric keys. We are also missing hop-by-hop authentication, which further increases the risk of denial of service attacks, but also digital signatures have that problem if the network is flooded with messages.

We will not decide here which solution that should be used since they both will give the same result for our evaluations. Depending on other limits as communication needs and power supply, they are preferable in different scenarios.

Key management

We also need to make some assumptions regarding the key management. There are several possible ways to ensure that the nodes are given the set of cryptographic keys that are necessary for secure communication in ad hoc networks, but they all have drawbacks. Most existing standards for key management, e.g. KERBEROS [29], assume an online centralized trusted server, something that is not always possible in ad hoc networks.

Key management for ad hoc networks are a research area itself and lots remain to be done. For simplicity, we here assume that keys are pre-created and pre-distributed with help of smartcards and smartcard readers in each node. On each card there should at least be a certificate signed by

a trusted CA for an asymmetric key structure or the required authenticated symmetric keys needed to start TESLA. This information should exist for all nodes that are allowed to be part of the ad hoc network.

The main problem with this manual approach is that revocation of compromised keys is difficult and takes a long time, one consequences of this is that it will be difficult to exclude a node from the network. On the other hand, for automatic responses this might not present such a problem, since responses that completely exclude nodes from the network probably will not be allowed anyway. Responses will probably be to make the protocol more resilient against nodes that function incorrectly.

Attacks on other layers are still a problem though and need to be further considered in future work. Manual initialization will probably be used also in the long term, but online updates will probably be a complement. However, we will not look into this in this report.

5.2 Extended finite state machine (EFSM)

A finite state machine is a model of computation consisting of a set of states, a start state, an input alphabet, and a transition function that maps input symbols and current states to a next state [16]. The behavior of the finite state machine can be described graphically in the form of a state transition diagram, as shown in figure 2a. That is, the state transition diagram specifies a set of transition functions for each state of the machine. These transition functions combined with input strings determine the next state of the finite state machine for a given state. If no transition rule is executable, the machine is said to be in an end-state.

However, the finite state machine computation model is insufficient to model the AODV protocol, since it lacks the ability to model the transfer of arbitrary values and manipulate variables conveniently. Therefore, AODV is modeled using the extended finite state machine, which is an extended version of the finite state machine with two major differences [17]. First, the extended finite state machine uses variables that have symbolic names and hold abstract objects, which in this case are integer values. Second, logical operators are used to manipulate the contents of the variables, as shown in figure 2b.

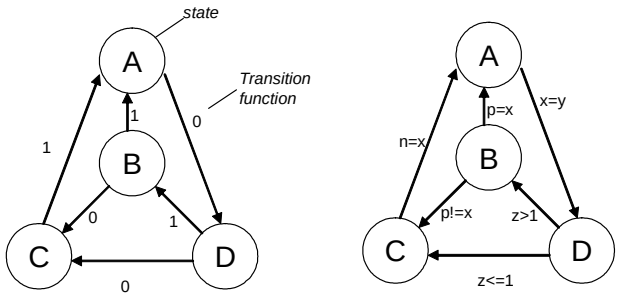


Figure 2: FSM State Transition Diagram (a) and EFSM State Transition Diagram (b)

5.3 AODV Specification

The proposed AODV specification is an abstract of the AODV protocol specified in RFC 3561 [3]. That is, only essential details of the protocol are modeled in order to detect if any node performs attacks against other nodes. First, the network collection module of the IDS agent gathers routing packets received from neighbor nodes and routing packets sent to other nodes. Next, the AODV state machine examines the received routing packets from the node to detect if

any node is performing routing attacks against other nodes. This is achieved by creating an instance of the AODV state machine for each received routing packet to a certain destination. If an instance of the AODV state machine already exists for the destination, the instance is updated with information from the received routing message.

AODV is modeled in two transition diagrams; Route Request and Route Reply. In the future, also RERR messages will be modeled. For each RREQ received from a unique destination, we create an instance of the AODV state machine that starts in the *RREQ receive* state, see figure 3. Similarly, we also create instances of the AODV state machine for every received RREP from a certain destination. Thus, there can be many instances of the state machine in runtime. It is important to find a way to limit the number of instances of the AODV state machine to save memory and computer capacity. Therefore, an instance of the AODV machine for a certain destination is deleted automatically when the state machine instance reach the final state (end-state). Thus, the maximal number of state machines is equal to the number of nodes in the network multiplied by two.

The RREQ transition diagram is depicted in figure 3 whereas the RREP transition diagram is depicted in Figure 4. It is described in Section 5.4 how these specifications can detect intrusions. State machines of protocols often have more than one transition rule per state, i.e. a non-deterministic choice. In our proposal, there are several transition rules for some states, but they are mutually exclusive. That is, at any time only one, or none, of the transitions is valid. Thus, a given input signal to a certain state always results in only one state.

The Route Request transition diagram
As shown in figure 3, the *RREQ receive* state has three transition functions with corresponding states;

1. *Drop RREQ*
   The RREQ is dropped if the node has received a RREQ with the same Originator IP address and RREQ identity within the last PATH_DISCOVERY_TIME.

2. *Invalid RREQ*
   A node broadcasting RREQ must follow certain rules. The state machine enters the invalid state if one of the following rules are true;
   a. A neighbor node originates more than RREQ_RATELIMIT RREQ messages per second. This rule detects certain types of resource depletion attacks (attack 6 in table 1).
   b. The waiting time between two RREQs with the same originator and destination IP address must be at least NET_TRAVERSAL_TIME. This rule detects certain types of resource depletion attacks (attack 6 in table 1).
   c. A node identified by its signature has not incremented the RREQ identity when broadcasted a new RREQ. This rule detects rushing attacks (attack 4 in table 1).
   d. Two nodes identified by their signatures originate a RREQ with the same originator IP address. This rule detects false message propagation attacks (attack 1 in table 1). Note that two nodes in a network never should use the same IP address. In case of dynamic IP addresses, a protocol such as DHCP (Dynamic Host Configuration Protocol) or DAD (Duplicate Address Detection) will make sure that two nodes do not use the same IP address. If static IP addresses are used, a certain signature should always correspond to a certain originator IP address.

3. *Valid RREQ*
   If neither of the above translation function is true, the AODV state machine enters the *valid RREQ* state.

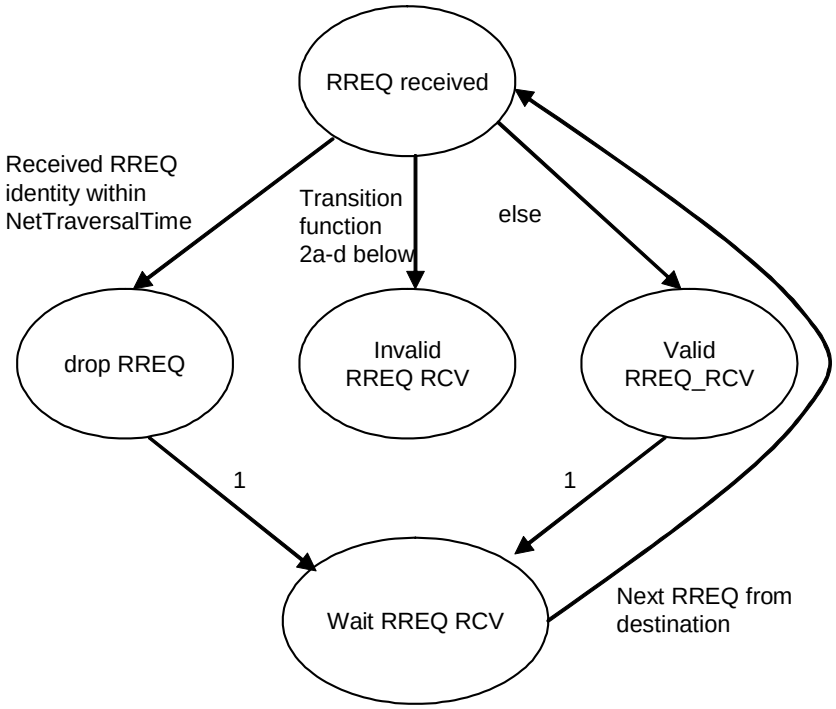


Figure 3: AODV *Route Request received* transition diagram

The Route Reply transition diagram
As shown in figure 4, the *RREP receive* state has two transition functions with corresponding states;

1. *Gratuitous* *RREP*
   There are two types of RREPs; normal RREP and gratuitous RREP. If an originated node wants to have bidirectional communications with the destination node, it sets the gratuitous RREP flag (G-flag) in the RREQ. In such cases, any generation of a RREP by an intermediate node to the originated node should be accompanied by a gratuitous RREP to the destination node.

   We assume that an authentic gratuitous RREP is sent together with the received HMAC RREQ. Thereby, it is possible for another node to verify that the RREQ really had the G-flag set. Thus, it is possible to decide if the RREP is a gratuitous RREP or not.
2. *Normal* *RREP*
   If the RREP is not a gratuitous RREP it is considered as a normal RREP. The node first checks whether it has sent a corresponding RREQ, otherwise the RREP is invalid. For a valid RREP there are two cases. If the D-flag is set only the destination may respond otherwise the RREP is invalid. However, if the D-flag is not set the message needs to be further studied, see details in Section 5.4.

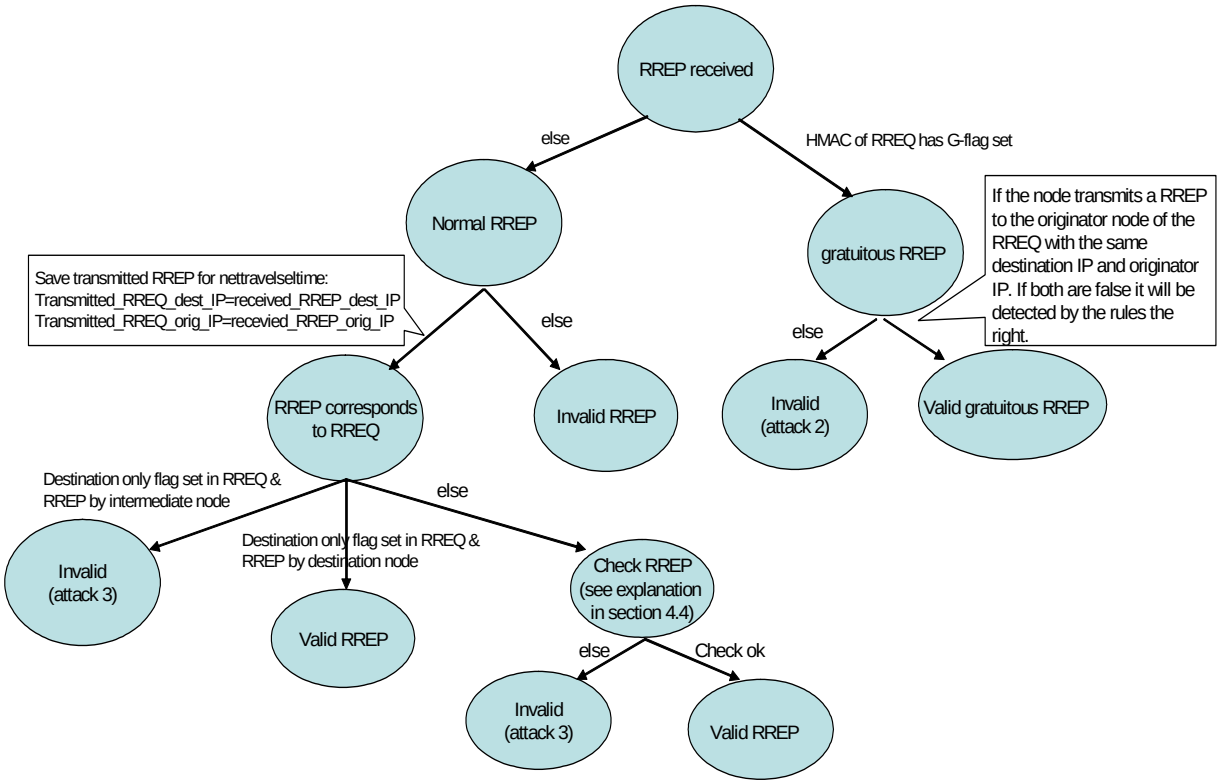


Figure 4: AODV RREP transition diagram.

5.4 Detection of attacks

In this section, we describe how our proposed intrusion detection system can detect attacks against the AODV protocol. These attacks are described in table 1 in Section 3.2.

*False message propagation of RREQs (attack 1):*In this attack, an adversary impersonate a node and sends RREQs with the originator IP address of another node in order to create route disruption. If two nodes identified by their signatures originate a RREQ with the same originator IP address, the intrusion detection system will regard this as an attack (rule 2d in Section 4.3).

If static IP addresses are used, the alerting module will categorize this as an attack with high accuracy. However, if dynamic IP addresses are used it may be possible that two nodes use the same IP address for a short period. Thus, the alerting module will log the event and regard it as an attack, if two nodes use the same IP address longer than a certain period.

False message propagation of RREPs (attack 2): In this attack, a node sends a forged RREP with an existing false originator IP address even though the node has not received any related RREQ. This attack is detected by verifying that a received RREP corresponds to an earlier transmitted RREQ, which was sent within NET_TRAVERSAL_TIME. Thus, all nodes need to save every sent RREQ for NET_TRAVERSAL_TIME.

False route reply (attack 3): It is a false RREP if the destination only flag (D-flag) is set and an intermediate node reply to the RREQ. Another example of a false RREP is when an adversary advertises a route in the RREP with a false value (e.g. false destination sequence number). A node that receives a RREP as a response to an originated RREQ verifies the RREP by sending an IDS message to the destination node, see state *Check RREP* in figure 4. The IDS message contains the destination sequence number and hop count value in the RREP. The destination

node sends an ACK to the originator node if the hop count corresponds to the TTL value in the IP header and the received destination sequence number is equal or lower than the current destination sequence number of the node. If no ACK is received within NET_TRAVERSAL_TIME the intrusion detection system will send a new RREQ with the destination only flag set.

Rushing (attack 4): In a rushing attack, a malicious node sends RREQ messages with a false originator IP address. This attack is detected by rule 2c in Section 5.3.

Modification of routing messages (attack 5): A modified routing message will be dropped, see explanation in Section 5.1 (authentication and integrity protection).

Resource depletion attack (attack 6): In this attack, an adversary is flooding the network with RREQs or RREPs. Flooding of RREQs are detected by rule 2a and 2b. Flooding of RREPs is false message propagation of RREPs. Thus, flooding of RREPs is detected in the same way as attack 2.

Dropping of routing packets (attack 7): The proposed intrusion detection system cannot detect this attack, but dropping of routing packets is normally not a serious attack. If one node on the nearest path between two nodes drops RREPs or RREQs, the consequence of the attack may be suboptimal tours. Otherwise, dropped routing packets will not cause any consequence at all. If there is only one path between two nodes, dropping of routing packets may result in that the network is divided in two networks. However, this can also be caused by the movement of the node.

Routing table overflow (attack 8): Routing table overflow attacks are performed by sending false RREQs with non-existing originator IP addresses or false RREPs with non-existing destination IP addresses. Routing table overflow with false RREQs are performed by sending many RREQs within a short period. Thus, this attack is detected by rule 2a in Section 5.3. Routing table overflow with false RREPs is false message propagation of RREPs. Thus, this attack is detected in the same way as attack 2.

Modify routing table (attack9). In this attack, an adversary modifies the routing tables. A node with a modified routing table may falsify replies to valid RREQs. The attack is detected in the same way as attack 3.

5.5 Simulation of attacks

The proposed methods for intrusion detection that are presented in this report have been tested on ad hoc networks consisting of eight nodes. The used simulation environment, Aquarius, is developed in C++ by the department of Communication Systems at FOI. The purpose of the environment is to support network simulations of different communication systems, mainly ad hoc networks. It does not contain a full implementation of the TCP/IP suite. However, it focuses on the lower layers and has mainly been used for analysis and evaluation of routing protocols and multiple access protocols. It is possible for the user to choose which protocols to use, and also to set the number of nodes in the network and to choose topology and which statistics to collect. The network topologies are based on the path loss between the nodes that have been calculated using the ground wave propagation library DetVag-90® [30], which takes the real terrain heights and terrain types into account.

To be able to evaluate the proposed method to detect attacks against the AODV routing protocol, we have simulated four of the attacks. AODV has been implemented in Aquarius according to RFC 3561 and the attacks have been implemented in one node by another routing protocol in that node. The network consists of eight nodes and one node is chosen to be malicious and is trying to attack the AODV routing protocol. The behavior of all other nodes in the network is as close to RFC 3561 as possibly.

The nodes are static and randomly distributed over a real terrain area. All nodes in the network are logging their outgoing and incoming routing packets and this data is used when detecting the attacks. Two different traffic situations are simulated. First, two nodes in the network are exchanging 60 bits/second. Second, all nodes in the network are randomly communicating at a rate of 100 bit/s. All data traffic is sent as unicast. The traffic load is chosen to be low so that the network never will get overloaded. A node pair will also have to start new route searches during the simulation when previously found routes time out. The simulation time is 1000 seconds.

Simulated attacks

Attack 1-4 in table 1 in Section 3.2 are implemented and tested in five randomly created networks. The traffic situation is also varying. To make detection of attacks harder, the malicious node is only attacking during certain periods.

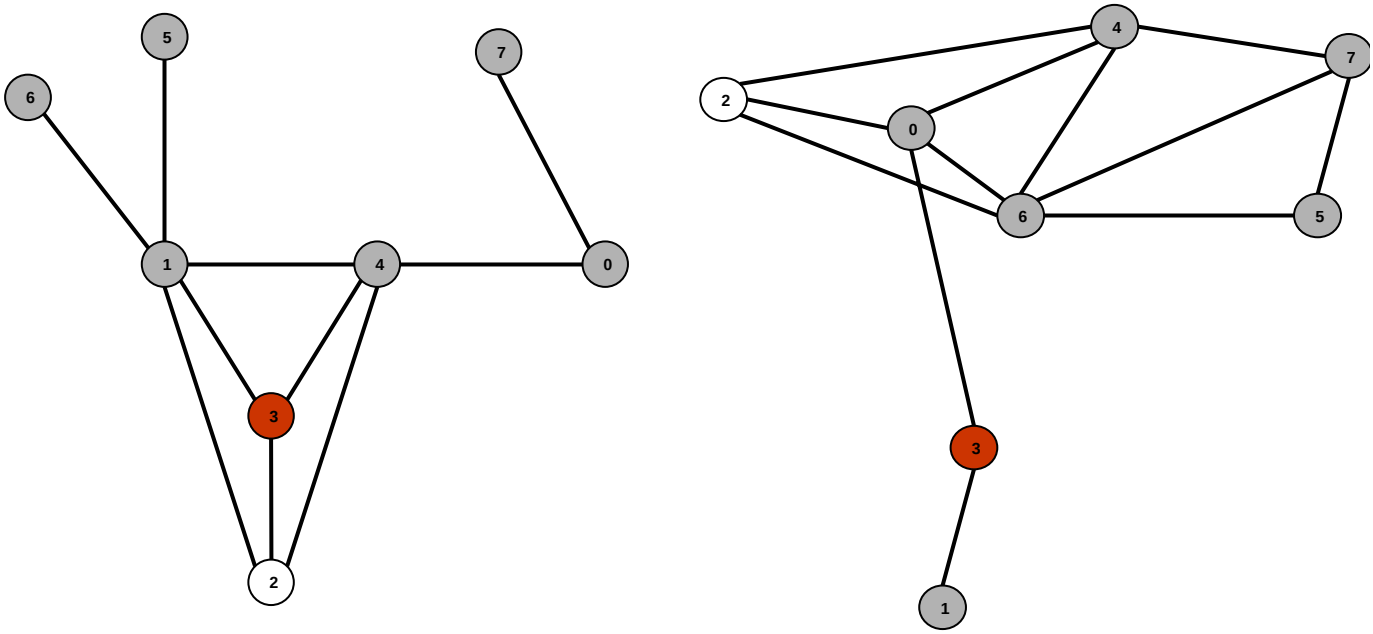


Figure 5. Two different examples of network topologies used in simulation. The nodes are static and the data traffic is randomly distributed as unicast.

The first implemented attack is the *rushing attack* where the malicious node generates false RREQs. The malicious node chooses for example to attack node 2, see the networks in Figure 5. The attack is initiated when the malicious node (in this example node 3) receives a RREQ from node 2. To suppress the node's own RREQ node 3 starts sending new false RREQs with the selected node as the *originator IP address* and with a false and incremented *RREQ_ID*. The result of this is that when node 2 later tries to send a true RREQ this message will be discarded by those nodes who have recently received a false RREQ. A node does not accept a RREQ with

the same RREQ_ID twice within a certain period. Another consequence is that all nodes that receive the RREQ will set up routes for node 2 through node 3. This attack can be successful but the effect is limited to the node that is chosen to be attacked. More than one node can of course be attacked but the effect of this attack is that the malicious node sends out quite large bursts of RREQs, and this could perhaps be easier to detect. An example of a response to this attack would be if the attacked node would listen to the false messages and increase its own RREQ_ID faster than the malicious node, or to randomly choose a RREQ_ID. This is however not described in RFC 3561.

False message propagation of RREQs is also implemented and has been a successful attack. The attacker is sending false RREQs, where the *originator IP address* is set to addresses of other nodes in the network. The destination IP address is here set to an address that does not exist in the network, just to make sure that the false RREQs do not reach the destination. The effect of this attack is that all routes in the network will be set through the malicious node, since nodes update their routing tables upon receiving new RREQs.

Another similar attack is *false message propagation of RREPs*. RREPs are here generated and sent without any incoming RREQ packet. In this attack, the malicious node selects a node to be attacked. RREPs with the selected node's address as destination are sent to all other nodes in the network. The result of this is that that all routes to the selected node will be set through the attacker. If we compare these two attacks, we can see that generating false RREQs may be more efficient since RREQs are sent as broadcast and fewer packets is needed to destroy a nodes possibility to route.

The fourth attack is to *send false RREP to a received RREQ*. The attacker reply to all received RREQs whether or not a route is known. Many routes will thereby be set through the malicious node. This attack can be hard to detect if it is done right, since intermediate nodes may be allowed to answer a RREQ if they have a valid route. In this case the malicious node set the hopcount in the false RREP to 0 and also increases the destination sequence number with two to ensure that this RREP will override any true one. This might be detected by the neighbours of the malicious node if AODV would have that functionality.

In the implementations of these attacks, the malicious node acts like a *black hole* and discards all received data packets and no RREPs or RREQs are forwarded. The success of the attacks is varying, mostly depending on the location of the malicious node. All the attacks can be successful and block the communication between nodes when the node is situated in the centre of the network, see for example the network to the left in Figure 5. If instead the malicious node is situated far from the communicating nodes, the attacks have less effect, see the network to the right in the same figure. The implementation of AODV also has a strong impact on the consequences of the attack. Since the algorithm is designed for a network without any malicious nodes there exists no description of actions taken, e.g., when receiving unexpected routing traffic. The consequences of some attacks could perhaps be diminished if the algorithm was better designed.

However, further investigation and more experiments are needed to be able to draw any conclusions about consequences of routing attacks.

5.6 Experiments

A specification-based detector was implemented in Java to test if the described rules were sufficient for detection. The detector was implemented according to the EFSM model described in Section 5.2. It had to decide states on all incoming packets. For this to work, a sufficient list of

the most recent incoming and outgoing packets had to be maintained in memory, to compare with new incoming packets. The analysis was not done in real-time, but rather post-mortem on traces of the data.

There are two types of attacks. The first does successfully harm the target, while the second fails in affecting it. The first type is crucial to detect and prevent, since it modifies the routing table with false information, and is flagged as an INVALID state by the detector. The second type is potentially dangerous, and might, with timing become successful and should be taken care of. This attack is however denoted as DROPPED by the detector, since the routing protocol will disregard it anyway. A large difference between the two types is that a DROPPED package might not be an attack. Anyway, requirements for mobile ad hoc networks dictate that computing power should be preserved [31], and it is unnecessary to waste it on investigating DROPPED packets.

The most complicated attack to discover was the RREP false reply (see attack 3 in Section 3.2). Since intermediate nodes, on the way between source and destination nodes, may answer according to the information in their routing tables, a route needs to be verified with the final destination. This was done by introducing two IDS messages: route verify request and route verify acknowledgment. In this implementation, the destination will not answer on faulty route verify request. For the requesting node, only two AODV packages are required to receive a route (RREQ and RREP); but to ensure it is correct two more packages are needed (request and acknowledgment). The detection process is performed in three asynchronous steps, where the second might not be executed, and the third works like a garbage collector:

- 1. When receiving an RREP, check that RREQ was sent that corresponds with it. Wait for an IDS route acknowledge to be received.
- 2. If received acknowledgment, check that it matches correct RREP and mark it as VALID state.
- 3. After a timeout, in not receiving an acknowledgement, check the status on the waiting RREPs. If they did affect the routing table, flag them as INVALID, or else flag them as DROPPED.

The generation of AODV and IDS packet was done in the simulator (see 5.5). The actual detection was done separately post-mortem on the logged data traffic. The detector could not intervene and protect the AODV routing table from attacks since it was not active during the simulation. Since the attacks were allowed to interfere with the nodes routing tables, the effect of the attacks were propagated among the nodes. Therefore, a node, not being an attacker, could indirectly perform an attack by sending false information based on its routing table. This was detected as an attack by other nodes.

5.7 Results

The detector was very efficient in detection rate, with few false alarms, see table 2. These good results could be achieved due to strong and exact rules, together with the presence of real signatures to verify the identities or originating nodes. The detection rates only consider INVALID attack packets, and not DROPPED packets (see 5.6).

The detection rate of the RREQ attacks has a perfect score, which was achieved by using the signatures. The attacker can try to lie about its identity; but when it does, it is easily detected.

To make it easier to verify if packets are bad or not, every packet in the simulator are marked as good or bad. The results from the detector can then be compared with the ones from the simulator. However, for attack 3 it was difficult to mark some packets correctly in the simulator. The simulator claims that some packets are valid even though they are attacks. However, the detector succeeds to detect these packets as attacks. In other words, the packets are marked as attacks by the detector, which is correct, but the statistics claim it is a false alarm. This is probably the case for all the false alarms of attack 3 in the table below. One way to avoid this is to integrate the detector into the simulator and use the results to protect the routing table, that is, to implement an intrusion prevention system (IPS).

The detection rate of attack 3 could probably have been perfect (100%) if the IDS packets had used a unique identity to ensure that a data packet would match only the correct acknowledgement packet. The attacker succeeded in confusing the detector since the attacking packet received an acknowledgement. The acknowledgement was in fact for another data packet, but due to perfect timing and equal hop distance the detector did not notice this.

Attack number	**Attack type**	**Packets**	**Bad packets (%)**	**Detection rate (%)**	**False alarms (%)**
1	RREQ false message propagation	92016	88	100	0
2	RREP false message propagation	26906	17.5	100	0
3	RREP false route reply	50190	6.6	99.98	1.6
4	RREQ rushing	139614	65	100	0

Table 2: Results from detection tests

6 Conclusions

In this report, we have described vulnerabilities of mobile ad hoc networks and attacks that an adversary could exploit. We have argued that intrusion prevention techniques (such as encryption, authentication) is not sufficient to protection against these attacks. Intrusion detection is proposed to complement intrusion prevention in order to secure mobile ad hoc networks. We have also described related research to show that new techniques for intrusion detection must be developed to make intrusion detection techniques more suitable for mobile ad hoc networks.

In our continuing investigation, we describe an architecture for intrusion detection that is suitable for mobile ad hoc networks. We propose to use a specification-based model for detection of attacks in mobile ad hoc networks. By examining violations of specifications, it is possible to detect attacks. The approach is illustrated for the AODV routing protocol. Attacks against the AODV protocol have been implemented and simulations have been conducted to evaluate our

model. We have simulated four types of attacks and each type of attack is implemented in five different ways. Performed experiments show that the specification-based method can detect attacks with high accuracy and few false alarms.

6.1 Future work

The following is a list of possible continuation of the project;

- Refine the developed method for AODV

In this work, we have proposed a method to detect attacks against the AODV routing protocol. The method is evaluated with experiments. In these experiments, many attacks are implemented and simulated. However, it is possible to perform more attacks against the AODV routing protocol. Thus, a possible continuation of the project is to implement more attacks and verify that the method also can detect these attacks. This also makes it possible to refine the method. Furthermore, the AODV model does not model RERRs. Thus, a possible continuation could be to extend the AODV model with RERRs and evaluate this.

- Show that the proposed method is applicable to other protocols

The evaluation of the proposed methods shows that it is possible to detect attacks with high accuracy and few false alarms. The approach is illustrated for the AODV routing protocol. We believe that the principle behind the approach also is applicable to other protocols as well even though the protocols may differ in the format. Thus, it would be interesting to show, with experiments, that the method also is applicable to other protocols. For example, it would be interesting to verify that the method is applicable to the Optimized Link State Routing protocol (OLSR) [18].

- Consequence of attacks against AODV

In this work, attacks against the AODV routing protocol is implemented in order to verify that it is possible to detect them. A related and relevant field is the consequences of these attacks. For example, can any node in the network disrupt the communication for another node in the network? How much of the communication in the network can be disrupted? In Section 5.5, consequences of attacks are briefly mentioned. However, it is necessary to run more traffic cases with many nodes, in order to make conclusions about the consequences of these attacks.

- Study and implement suitable intrusion response mechanisms

The detector implemented here almost reached 100% detection rate. One way to meet this goal could be to integrate the detector into the simulator. The role of the detector can be promoted to an automatic intrusion prevention system (IPS). Other forms of automatic response mechanisms discussed here can also be further investigated.

References

1 Y. Zhang, W. Lee, and Y. Huang. “Intrusion Detection Techniques for Mobile Wireless Networks”. In Report on a Working Session on Security in Wireless Ad Hoc Networks”. *Mobile Computing and Communications Review*. Vol. 7, No. 1. Pages: 74-94. ACM. January, 2003.

2 S. Axelsson, “Intrusion Detection Systems: A Taxonomy and Survey, ”Tech. Report no. 99-15, Dept. of Comp. Eng., Chalmers Univ. of Technology, Sweden, Mar. 20, 2003

3 C. Perkins et al., “Ad hoc On-Demand Distance Vector (AODV) Routing”, RFC 3561, July 2003.

4 Finite state machines, http://spinroot.com/spin/Doc/Book91_PDF/ch8.pdf (accesses 050206)

5 A. Mishra, K. Nadkarni, A. Patcha. ”Intrusion Detection in Wireless Ad Hoc Networks”. *Wireless Communications*. Pages: 48-60. IEEE. February, 2004.

6 L. Zhou, Z. Haas, “Securing Ad Hoc Networks”, *IEEE Network Magazine*, Nov/Dec 1999

7 V. Gupta; “Denial of service attacks at the MAC layer in wireless ad hoc networks”, proc. *Milcom 2002*, October 2002.

8 S. Wong, “The evolution of wireless security in 802.11 networks: WEP, WPA and 802.11 standards”, May 20, 2003.

9 N. Moore, “Optimistic Duplicate Address Detection for IPv6”, draft-ietf-ipv6-optimistic-dad-02.txt, September 2004

10 Y. Hu, A. Perrig, et al., “Ariadne: A secure on-demand routing protocol for ad hoc networks”, *MobiCom 2002*, September 2002

11 S. Thomson et al., “IPv6 Stateless Address Autoconfiguration”, RFC 2462, December 1998.

12 Y. Zhang, W. Lee. “Intrusion Detection in Wireless Ad hoc Networks”. *Proceedings of MOBICOM*. Pages: 275-283. ACM. 2000.

13 C. Ko, M. Ruschitzka, and K. Levitt, “Execution Monitoring of Security Critical Programs in Distributed Systems: A Specification-based approach”, *In Proceedings of Symposium on Security and Privacy*, 1997.

14 C. Ko, P. Brutch, J. Rowe, Tasfnat and K. Levitt, “System Health and Intrusion Monitoring using a Hierarchy of Constraints”, *In Proceedings of 4th International Symposium on Recent Advances in Intrusion Detection*, 2001.

15 D. Johnson, D. Maltz, Y-C. Hu, The Dynamic Source Routing Protocol for Mobile Ad Hoc Networks (DSR) , draft-ietf-manet-dsr, internet draft, work in progress

16 National institute of standards and technology, *Finite state machine*, http://www.nist.gov/dads/HTML/finiteStateMachine.html (accessed 050206)

17 Finite state machines, http://spinroot.com/spin/Doc/Book91_PDF/ch8.pdf (accesses 050206)

18 T. Clausen, P. Jacquet, “Optimised Link State Routing Protocol (OLSR)”, RFC 3626, October 2003

19 S. Bhargava and D.P. Agrawal. “Security Enhancements in AODV Protocol for Wireless Ad Hoc”, VTC 2001, Fall, vol. 4, Oct 7.-11.2001, pp 2143-47

20 R. Ramanujan et al., “Techniques for Intrusion-Resistant Ad Hoc Routing Algorithms (TIARA)” MILCOM 2000, vol 2, Oct. 22-25, 2000, pp. 660-664.

21 Marti et al., “Mitigating Routing Misbehavior in Mobile Ad Hoc Networks,” *In Proc. 6th Annual int’l Conf. Mobile Comp. and Net*., Boston, MA, pp. 255-65.

22 R. Ogier et al, “Topology Dissemination Based on Reverse-Path Forwarding (TBRPF),” RFC 3684, Feb. 2004.

23 W. Stallings, *Network Security Essentials - Applications and Standards*, Prentice Hall 2003, ISBN 013035128.

24 A. B. Smith, “An Examination of an Intrusion Detection Architecture for Wireless Ad Hoc Networks”, In Proc. *5th Nat’l. Colloq. For Info Sys. Sec. Education*, May 2001.

25 P. Albers et al., “Security in Ad Hoc Networks: a general Intrusion Detection Architecture Enhancing Trust Based Approaches,” *In Proc. 1st Int’l Wksp. Wireless Info. Sys.*, Ciudad Real, Spain, Apr. 3-6, 2002.

26 Y. Huang and W. Lee, “Attack Analysis and Detection for Ad Hoc Routing Protocols”, RAID, 2004

27 C-Y Tseng et al., “A specification-Based Intrusion Detection System for AODV”, In Proc. Of ACM Workshop on Security of ad hoc and sensor networks, 2003.

28 M. Guerrero, SAODV, Internet draft, draft-guerrero-manet-saodv, September 2005, work in progress

29 J. Kohl and B. Neuman, “The Kerberos Network Authentication Service”, RFC 1510, Sept. 1993

30 B. Asp, G. Eriksson, and P. Holm, “DetVag-90 -- Final Report”, Scientific report FOA-R--97-00566--SE, Defence Research Establishment, Linköping, Sweden, Sept. 1997.

31 E. Hansson and A. Hansson, “Evaluation of wireless Intrusion Detection tools for Mobile Ad Hoc Networks”, FOI-R—1374—SE, Linköping, FOI, November 2004.

32 A. Perrig et al., “Efficient authentication and signing of multicast streams over lossy channels”, In *IEEE Symposium on Security and Privacy*, pp 56-73, May 2000.