UNIVERSITY OF PIRAEUS

Security Attacks and IDS solutions in Mobile Ad Hoc Networks

Msc of Digital Systems Security

Platon-Pantelis Kotzias

Supervisor: Xenakis Christos

December 2011

Abstract

Mobile ad hoc networks (MANETs) are wireless networks established by autonomous nodes without the aid of any established infrastructure. In MANETs, the nodes themselves implement the network management in a cooperative fashion and thus, all of them are responsible for this. Due to their wireless-mobile nature MANETs are susceptible to various attacks. This thesis focuses on exploring the attacks performed at the two most vulnerable layers, namely the Data Link and the Network layer. The ability to effectively detect these attacks is considered crucial for the survival of MANETs. Therefore, we provide a critical analysis of the most recently proposed detection schemes for both layers which can be used as bedrock for the creation of novel intrusion detection engines.

In addition, this thesis proposes an Intrusion detection engine that is capable of detecting the majority of attacks performed at the MAC layer. The engine utilizes a specification-based mechanism, although It also uses some threshold based rules.

Contents

1. Introduction

The field of wireless networks has been experiencing great growth over the past decade. We have witnessed great advances in network infrastructures and in the evolution of wireless devices such as laptops, PDAs and cell phones. These devices are getting smaller, cheaper, more powerful and they are also capable of running more applications and network services. These technological achievements create a debate over existing architectural models based on server-client model and have lead to the creation of more groundbreaking models based on more autonomous features. In fact, a significant part of the research community's interest shifts from wireless LANs (WLANs) to mobile ad hoc networks (MANETs). MANETs came to revolutionize the way of delivering connectivity among nodes, thereby, providing new capabilities along with new challenges on many levels. Now, for the first time MANETs have given the opportunity to create complex network structures capable of sustaining many different applications without the need of infrastructure which is both complex and expensive. Future applications of MANETs are very promising including the creation of networks after a disaster (storm, earthquake etc.), the installation of digital sensors in places unreachable by humans, the deployment of Vehicular ad hoc networks (VANETs) which allows connectivity among vehicles or simply sharing information among users during conferences. A MANET is a collection of autonomous nodes that form a dynamic, purpose-specific, multi-hop radio network in a decentralized fashion. In a MANET, the nodes themselves implement the network management in a cooperative fashion and thus, all the network members share the responsibility for this. This decentralized nature of MANETs make them susceptible to a variety of attacks performed in different layers of TCP/IP. Therefore, features such as lack of central infrastructure, wireless connectivity, mobility of users and limitation of nodes' resources creates a very challenging environment where conventional security approaches fail to support viable solutions. Thus, a great research effort is carried on to make MANET applications both efficient and safe.

The main purpose of this thesis is to study the foremost attacks in MANETs and to explore security solutions based on intrusion detection. Our goals include studying the attacks in Data Link and Network layer, as they are the most problematic layers in MANETs. We provide a detailed description of the performed attacks, to be used as a guide for the deployment of more effective detection schemes. In addition, we propose a lightweight specification-based detection engine for the MAC layer that manages to detect the majority of the attacks performed at this layer.

Following this introduction, in Chapter 2 we discuss the main characteristics of MANETs along with

their applications on several fields. In Chapter 3 after presenting the main functionality of protocols at the Data Link layer we describe the majority of the attacks at this layer with a critical analysis of the most recently proposed detection schemes. Chapter 4 focuses on the most sophisticated attacks conducted at Network layer along with a critical analysis of the detection methods that have been proposed. Chapter 5 focuses on Intrusion Detection Systems (IDS) discussing their main classification and features. The core of this chapter is the presentation of a lightweight specification-based detection engine for MAC layer. Last but not least Chapter 6 discusses the conclusions of our research and suggestions for future research activity.

2. MANETs and their characteristics

A Mobile Ad Hoc Network (MANET) is an autonomous system of mobile nodes with routing capabilities connected by wireless links, the union of which forms a communication network [43]. Therefore, it can be considered as a temporary infrastructureless network formed by a set of wireless mobile hosts that dynamically establish their own network on the fly without relying on any central administration [44]. All participants at these networks act as both hosts and routers forming an autonomous network heavily depended on the belief that all participants give and take resources in a fairly manner. The nodes are usually devices with limited CPU, storage and energy resources such as laptops, PDAs and other mobile devices. Moreover, we can easily understand the serious challenges that exist in the implementation of MANETs. The foremost features of MANETs, which have a significant impact on both the quality of services and the security, are presented in [48] and are:

- *Infrastructureless*

There is an absence of central servers and fixed routers, therefore, all solutions should rely on distributed cooperative schemes instead of a centralized scheme.

- Wireless Link Use

MANET inherits the security issues that arise from the wireless link usage. Wireless local area networks (WLANs) face the same problems, however, the usage of Access Points (APs) gives the opportunity of applying effective security solutions. Moreover, both types of network need to overcome the issues engendered by the fact that in contrast to wired networks, wireless links allows attackers to access third nodes' data either those are encrypted or unencrypted.

- *Multi-hop*

As mentioned before all nodes that participate in MANET need to act as host and routers at the same time. In fact, the existence of such networks heavily depends on this feature and at some level the philosophy of MANET is based on this feature, namely the trust among nodes which it is not always guaranteed. Data is transferred from node to node allowing the connection of distant nodes by the creation of multi-hop routes. However, since nodes are usually devices with stringent resources some may not willing to act as servers in order to save resources resulting in connectivity problems.

- *Node movement autonomy*

Mobile nodes are generally autonomous units that are capable of roaming independently. This means that both routing protocols and security solutions need to be robust to increased mobility.

- *Amorphous*

Node mobility and wireless connectivity allow nodes to enter and leave the network spontaneously, to form and break links unintentionally. The network topology is not fixed but instead it changes regarding its size and shape. Therefore, security solutions such as Intrusion Detection Systems (IDS) need to consider this feature when they are built.

- *Power limitations*

Ad hoc enabled mobile hosts are small and lightweight, and they are often supplied with limited power resources, such as small batteries. This limitation causes a vulnerability, namely, attackers may target some nodes' batteries to disconnect them, which may lead to a network partition. On the other hand, security solutions such as cryptographic protocols and embedded IDS need also to be lightweight and energy conservative in order to be considered as vital solutions.

- *Memory and computation power limitation*

Ad hoc enabled mobile nodes have limited storage devices and weak computational capabilities. Consequently, high complexity security solutions, such as symmetric or asymmetric data encryption are difficult to implement.

2.1 Applications of MANETs

Although implementing a MANET is a very challenging task, several applications exist that try to surmount the aforementioned difficulties. The most widely discussed application scenario for pure general purpose MANET is a battlefield or a disaster-recovery network, however these kinds of networks have not achieved the envisaged impact in terms of real world implementation and industrial development [45]. On the other hand, many humanitarian application have been built using wireless communication based on MANETs. Wildlife monitoring focuses on tracking wild species to deeply investigate their behavior and understand the interactions and influences on each other, as well as their reaction to the ecosystem changes caused by many activities.

Another example is the DakNet Project [49] which provide intermittent Internet connectivity to rural and developing areas. In fact, the project established a very low-cost asynchronous ICT (Information and Communication Technology) infrastructure to provide connectivity to rural villages in India, where it is not cost-effective to deploy standard Internet access.

Moreover, another subcategory of MANETs which received great research attention is Vehicular Ad Hoc Networks (VANETs) which use ad hoc communications for performing efficient driver assistance and car safety. The communications include data from the roadside and from other cars. VANET research aims to supply drivers with information regarding obstacles on the road and emergency events, mainly due to line-of-sight limitations and large processing delays. The European FleetNet project [50] is an example of a VANET project. It aims in inter-vehicle communications and its key factors are the capability to distribute locally relevant data where generated or needed and to satisfy the vehicle drivers' and passengers needs for location-dependent information and services.

Last but not least, wireless sensor networks (WSN), another subcategory of MANETs, received great attention due to their many applications. A sensor network is created by a number of small, wireless, battery powered, smart sensor nodes and are devices which have computing and communication capabilities that not only sample real world phenomena but also can filter, share, combine and operate on the data they sense. An example of such an application is the Great Duck Island Habitat Monitoring project which is a pilot application for monitoring migratory seabirds on Great Duck island [51]. The WSN in this case was used to monitor the micro-climates in and around nesting burrows. Moreover another application of WSN is for health monitoring. Codeblue [52], a

system developed at Harvard University exploits a WSN to raise an alert when a vital sign fall outside of the normal parameters. The system monitors heart rate, oxygen saturation.

In conclusion, it is clear that MANET and its subcategories (VANET, WSN) are here to shape the existing field of wireless networks in various manners. We expect that as long as MANET applications become a part of our daily life the need for security would increase due to the exchange of valuable assets.

Attacks in MANETs

MANETs are susceptible to various attacks due to their nature and features discussed in paragraph 1.1. These attacks can be divided according to their origin or their nature. In [44] the authors classify the attacks according to their nature to *external* and *internal* attacks. In addition, a nature-based classification splits them into *passive* and *active* attacks.

External attacks includes attacks launched by a node that does not belong to the logical network, or is not allowed to access it. On the other hand, *internal* attacks are launched by an internal compromised or malicious node. This is a more severe type of threat since the proposed defense toward external attacks is ineffective against compromised and internal malicious nodes.

Passive attacks can be considered as actions where the attacker passively collects information that might be used later when launching an active attack. For that, the attacker eavesdrops packets and analyzes them to pick up required information. On the contrary, *active* attacks include almost all other attacks launched by actively interacting with victims such as jamming, Denial of Service (DoS).

Surveys on describing all the possible attacks in MANET in a layer perspective can be found in [46] [44] . In Table 1 we briefly summarize all the possible attacks related to the layer that are implemented. However, our research is confined to analyze only the attacks implemented in the two foremost layers in MANET, the Data Link and the Network layer, as these are the most problematic layers.

Layer	Attacks
Application layer	Repudiation, data corruption
Transport Layer	Session Hijacking, SYN flooding
Network Layer	Wormhole, Blackhole, Rushing, byzantine, flooding, resource consumption
Data Link Layer	Backoff manipulation, IFS manipulation, Data dropping, RTS/CTS Spurious attacks
Physical layer	Jamming, interceptions, eavesdropping

Table 1 Security attacks on each layer in MANET

3. MAC Layer attacks and detection methods

After introducing the basic features of MANETs, we move on describing the foremost attacks implemented in the Data Link layer. In particular, we discuss the formation of the attacks and describe the most recent detection methods proposed to detect them.

Our contribution in this chapter is a comprehensive description of the MAC layer functionality that is needed in order to follow the chapter. Moreover, we analyze the attacks in order to provide the bedrock for the creation of more accurate detection methods. In addition, the critical analysis of the current detection methods will enlighten the research community for the strengths and the weaknesses of these methods.

3.1 The Data Link Layer (DLL)

The Data-Link Layer (DLL) is responsible for one-hop connectivity between neighboring nodes. It consists of the Logical Link Control (LLC) and the Media Access Control (MAC) sub-layers. The IEEE 802.11 MAC protocol is a standard for MANETs, responsible for the coordination of transmissions on a common communication medium. It utilizes a distributed contention resolution mechanism for sharing the wireless channel among multiple wireless nodes. In this mechanism, when a node wants to transmit data, it initiates the process by sending a request to send (RTS) frame, and the destination node replies with a clear to send (CTS) frame. Any other node receiving the RTS or CTS frames retreats from transmitting any data for a certain time. The protocol is vulnerable to a variety of attacks such as DoS, traffic analysis, monitoring, MAC disruption, etc. IEEE 802.11 supports two

different MAC schemes, Distributed Coordination Function (DCF) and Polling Coordination Function (PCF). DCF operates solely in the ad hoc network, and either operates solely or coexists with the PCF in an infrastructure mode. In DCF all users with data to transmit have an equally fair chance of accessing the network. DCF specifies the use of CSMA/CA (Carrier Sense Multiple Access Collision Avoidance) protocol used to minimize collisions. In order to prioritize access to the wireless medium, DCF specifies three time windows, SIFS (Short InterFrame Space), DIFS (DCF InterFrame Space) and EIFS (Extended InterFrame Space). Typical parameter values for the MAC protocol depend on the physical layer that IEEE 802.11 uses. Table 1 shows the parameters used when the physical layer is using direct sequence spread spectrum (DSSS) [38].

Parameter	***Parameter value***
SlotTime	20 μs
SIFS	10 μs
DIFS	50 μs
EIFS	263 μs
ACK	112 bits + PHY_header = 203 μs
RTS	160 bits + PHY_header = 207 μs
CTS	112 bits + PHY_header = 203 μs
DATA	MAC_header (30b) + DATA (0-2312b) + FCS (4b)
CWmin	32 time slots
CWmax	1024 time slots
Timeouts	300-350 μs

Table 1: CSMA/CA parameters if DSSS is implemented

Every node with data to transmit has to sense the network for DIFS period of time. If the medium stays idle for such period the node can send an RTS (Request to Send) message to the receiver (Figure 1a). The receiver has to reply with a CTS (Clear to Send) message after sensing the network for a SIFS period of time. RTS/CTS messages are used to address the hidden and exposed terminal problem and inform any node that hears the exchange, the duration of the whole DATA frame transmission. Another advantage of using RTS/CTS messages is to minimize the amount of bandwidth wasted when collision occur. A node that overhears the exchange of RTS/CTS messages adjust its NAV (Network allocation vector) field, which indicates the amount of time that the node should wait before sensing the medium

again. A receiver that successfully receives a DATA frame must reply with an ACK (Acknowledgement) message.

In case the channel is busy during the DIFS period of time a node wishing to transmit must use its backoff mechanism (Figure 1b). CSMA/CA adopts a backoff mechanism to resolve channel contention. The transmitter after the expiration of NAV period of time has to sense the medium for DIFS time plus a random backoff time from range [0,CW] where CW is the contention window size maintained by each node. As long as the medium stays idle the backoff timer decrease and stop decrementing when the channel is busy. If the channel become busy during the backoff time the node must sense the channel for an additional DIFS/EIFS time before it can decrease the backoff timer again [39]. Once the backoff counter reaches zero the node can transmit the RTS to the receiver and follow the standard procedure. To guarantee fair access to the shared medium, a station that has just transmitted a packet and has another packet ready for transmission must perform the backoff procedure before initiating the second transmission.

There is also the possibility for the transmitter to send the RTS message but not receive the corresponding CTS (Figure 1c). This means that a collision occurred at the receiver end. In that case the transmitter performs the backoff mechanism after doubling its CW, thus he must now choose a value between [0,2CW]. After each retransmission the transmitter must double its CW until it reaches CWmax and remain the same until the packet is finally transmitted or the packet is discarded and CW is reset. When a collision occurs the transmitter has to wait for EIFS time instead of DIFS in normal circumstances. The transmitter has the same reaction if an ACK message is not received.

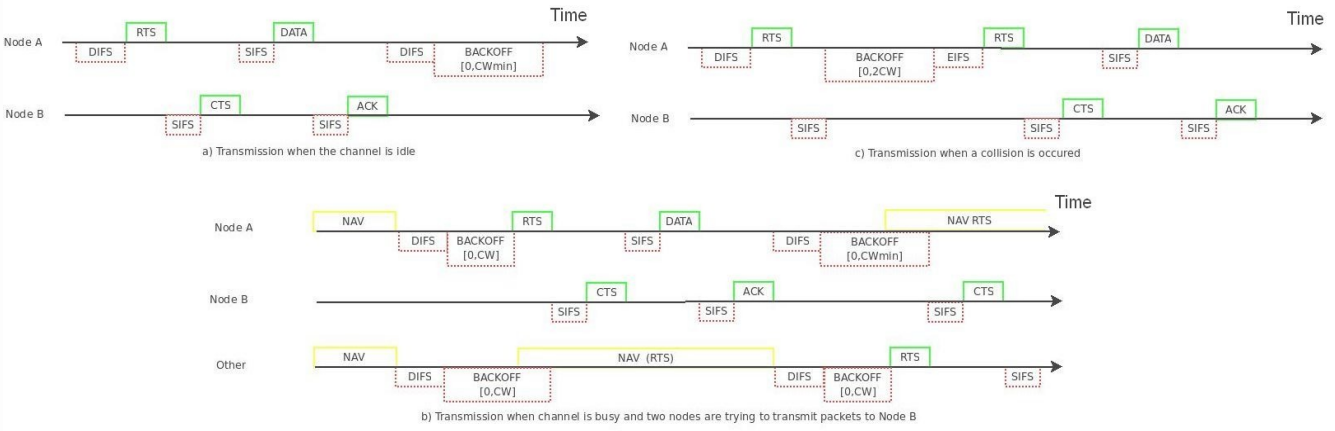

Figure 1: CSMA/CA Functionality

3.2 *NAV attack*

NAV attack is an efficient and easily implemented attack. Every time a node sends an RTS/CTS it also includes the time interval necessary to transmit the whole data frame and the acknowledgement related to it. This way every node that listens to the channel during the transmission of RTS/CTS can be informed and set its NAV value in accordance with the duration field. The NAV field specifies the earliest point at which the other stations can try to access the medium again. The attacker asserts a larger duration field while advertising the RTS packet, thereby preventing well-behaved clients from gaining access to the channel. Attacker can benefit because he can start decreasing his backoff counter faster, thus increasing his chance to capture the channel again. The maximum time that the channel can be reserved for in a single frame is limited by the size of the duration field, a maximum of 32767 μs [32]. Simulations in [34] have shown that the decrease in the throughput of the well-behaved nodes is proportional to the value of the NAV field.

3.2.1 Methods of NAV attack detection

DOMINO is an intrusion detection system proposed for 802.11 infrastructure networks [33]. Among others, DOMINO suggests a mechanism for detecting NAV attacks. Although the indicators implemented for detecting the attack are used in infrastructure networks, where the Access Point plays the role of the IDS, these indicators can also be used in MANETs without any modification. DOMINO adopts a simple and straightforward solution to NAV attack. The monitor node is measuring the actual duration of a transmission (including the DATA, ACK) and comparing it with the NAV value in the RTS or DATA frame headers. The monitoring node can detect nodes that regularly set the NAV to very large values. In DOMINO the monitor node is the AP of the network but in MANETs due to the infrastructureless nature of the network the monitor node can be any node in the vicinity of the attacker.

In [32] the authors created an anomaly based detection mechanism for NAV attack. The detection mechanism monitors and inspects the network activities and study the network performance statistics under normal condition. The statistics used in this work is the packets received from each transmitting node. Any fluctuations from normal condition are used to detect misbehaving nodes. This indicator is useful and does not cause delay in the network since every node can monitor the nodes from which it receives packets. According to the writers, under normal packet transfer the distribution pattern of packets is found to be uniform for all transmitting nodes. During the NAV attack the distribution does not follow a uniform pattern. The attacker on the other hand maintains the uniform

packet distribution under normal and attack conditions thus the attacker can be easily detected. However, this detection method presents some drawbacks. Firstly, the simulations tested only the scenario in which the attacker uses the maximum size of NAV value. However, attackers who want to be more careful or have identified the usage of an IDS mechanism may use smaller values of NAV that can cause significant damage to the network performance. In addition, as is the case of anomaly-based detection, the mechanism must monitor the network for a noteworthy period before it can create a normal behavioral pattern. Last but not least, the mechanism identifies misbehavior by monitoring the throughput of nodes. However, the throughput is mostly related to the application running on each station and may not be used to accurately detect the attacks [33].

3.3 RTS/CTS Spurious Attack

This attack has similar effects to the network performance as NAV attack but it cannot be detected by the methods described previously. In this attack a node transmits RTS/CTS messages but with no intention of sending any data packet afterwards. Therefore, nodes that overhear these messages update their NAV field accordingly and wait before sensing the medium again. An attacker that sends these spurious packets periodically is virtually jamming all the nodes in attacker's vicinity thus resulting in Denial of Service (DoS) state. The attack can be implemented using either RTS or CTS packets, however using an RTS message the attack is more efficient since the receiver replies by sending a CTS message which results in expanding the results of the attack. Spurious attack exploits the fact that the protocol does not check if a node's data transmission is actually happening during the expected period advertised in NAV field. Finally the attack can be combined with NAV attack to increase the time the medium is idle by manipulating the NAV field.

3.3.1 Methods of Spurious attack detection

The linchpin of detecting this attack is to ascertain that a node that sends an RTS/CTS message is actually completing the data transmission. Authors of [40] address this problem by introducing a mechanism named Carrier Sensing Discarding (CSD) mechanism. The scheme suggest the usage of several CSD points which are randomly chosen among the entire expected data packet transmission time. These points whenever they overhear a CTS packet, they assess the shared medium at the time when the corresponding data packet transmission should start. If the medium, within the range of a point, is idle the CTS packet is treated as spurious and the corresponding NAV value is discarded. On

the contrary, if the channel is sensed busy, the CTS is treated as normal. The greatest advantage of this method is that the probability of false alarm is 0 since the scheme is using the sensing mechanism of CSMA/CA which is assumed to be perfect. However, the scheme is only used to detect CTS spurious packets and not RTS which are the most commonly used due to their efficiency. Moreover, since the detection points are chosen randomly attackers also would be chosen as detectors. Therefore, the mechanism's efficiency may decrease in scenarios where attackers number is increasing.

Other proposals such as [41] suggest protocol modification to surmount this attack. The authors argue that 802.11 should have a provision to reset NAV value after a fixed period of time if the channel is found idle. However, a solution like this is considered impractical because it requires the redesigning of the protocol.

3.4 IFS Manipulation

When a node wants to transmit an RTS message it must sense the channel idle for a DIFS period of time, according to the CSMA/CA. A misbehaving node in order to increase its chances of accessing the channel it modifies the protocol parameters and decrease its DIFS or even transmit after SIFS. This can be done due to the implementation of protocol in software rather than in hardware [33]. The same manipulation can be done to SIFS and EIFS time periods.

3.4.1 Detection of IFS Manipulation

DOMINO [33] manages to effectively detect the DIFS manipulation by comparing it with the appropriate one. The Access Point calculate the idle period after the last ACK and distinguish any station that transmits before the required DIFS period. After several observations the AP can make a reliable decision. Although DOMINO, as mentioned before, is implemented in infrastructure networks the same detection mechanism can also be used in MANETs. Each node in the network could monitor the nodes in its vicinity using the above rule. However, this solution requires much energy thus, it is not the optimal in environments with stringent resource constraints such as MANETs.

3.5 Backoff Manipulation

Backoff manipulation is the foremost attack to the CSMA/CA protocol. This stems not only from the fact that most of the detection mechanisms that had been proposed try to surmount this attack but also from the fact that simulations have shown that the attacker can gain a significant increase in its

throughput by carrying it out [34]. As we seen in paragraph 3.1, there are situations where a transmitting node must select a random backoff value from the range of [0,CW], where CW (Contention Window) is a variable maintained by each node. IEEE 802.11 MAC protocol favors the node that selects the smallest back-off value among a set of contending nodes. Therefore, a selfish node may choose not to comply with the protocol rules by selecting small back-off intervals to gain significant advantage in channel sharing over well-behaved nodes [28]. A selfish node can manipulate the backoff mechanism implementing one or more of the following changes to the protocol parameters. Firstly, it can decrease the CWmin value which results in decreasing the range of the CW values. This allows the node to choose smaller values thus, having greater chances of capturing the shared medium. Secondly, a node can refuse to double its CW after each collision. Therefore, the node's performance would not be affected by collisions as the rest of the nodes. Thirdly, a node can select its CW values in a non-random way. Nodes that are capable of doing this manipulation can always select the smallest CW value thus increasing significantly their chances of capturing the channel.

3.5.1 Detection of Backoff Manipulation

The difficulty of detecting backoff manipulation lies on the fact that there is no straightforward way of distinguishing between a legitimate node that happens to select small backoff intervals and a selfish node [28]. In fact all detection mechanisms that had been proposed follow two distinct steps. The first one is the calculation of CW values. The detection mechanism needs somehow to calculate the CW values that are actually chosen by the node. This is an intricate process due to the difficulty of measuring the CW value just by observing a node. Two main perspectives are followed for this process, the one is the estimation of the CW value by direct observation such as [33] and the other is the estimation of CW by the observation of throughput like in [29]. The second step is to choose a rule to distinguish misbehaving from well-behaved nodes. After collecting the CW values that are selected by a node the detection mechanism needs to make a reliable decision on its behavior. The rules that have been proposed follow different perspectives, each with its own merits and demerits.

DOMINO manages to detect backoff manipulation using three backoff tests [33]. The *maximum backoff test* is an auxiliary to the others because it can be easily tricked. It is used to suspect stations whose maximum backoff over a set of samples is smaller than a threshold value. The *actual backoff test* is used to measure the average actual backoff of the observing node and compare it with the nominal backoff value (which is equal to the average backoff of the AP) and a parameter configurable

according to the desired true positive and false positive percentages. This test fails to detect the misbehavior case when the cheater has interframe delays (e.g., a TCP source using congestion control). The consecutive backoff test is used as a solution to the above problem. The average of the collected values is compared to a fraction α of the nominal value of the AP. The backoff detection mechanism adopted by DOMINO can easily implemented in 802.11 networks with AP but implementing the same mechanism in MANETs environment would create problems. The fact is that the mechanism is too complicate, namely three different checks are needed to detect backoff manipulation, thus, too much energy is required from nodes, a resource that is very limited in such environments. Moreover, the presence of interference could create difficulties in the estimation of the backoff of a neighboring node [29]. At last, DOMINO's usage of threshold-based detection reveals that it is prone to adaptive cheating. Advesaries who have knowledge on the thresholds that have been selected can modify their misbehior accordingly so it always stays under the specified threshold, therefore being undetected.

Authors of [30] proposed a scheme to detect backoff manipulation. This scheme requires modification to the existing backoff mechanism in order to be implemented. Except from a neighbor list each node needs to maintain entries for the degree of selfishness called *d*, the expected backoff value for the transmission named *ExpBOV* and the access values for sender and receiver informing the number of medium access of the node. All these entries are required for every neighbor in the vicinity of a node. The degree of selfishness along with the access values are used to determine if a node is misbehaving. By examining the neighbor list a node can ascertain if a neighbor's degree of selfishness is below the appropriate threshold and if the neighbor's access values are above the delegated thresholds. In this scenario the node is characterized as selfish and penalized. In order for a node to fill the neighbor list entries information related to backoff mechanism (sender backoff value, receiver backoff value) are exchanged through the RTS/CTS messages. Nodes that are marked as malicious are desolated by neighbors by not responding to their RTS messages. Simulations have shown that although the diagnosis accuracy is very high, namely 80%, in situations where the percentage of misbehavior increases significantly, the mechanism fails to detect misbehavior when the percentage of misbehavior is less than 20%. In addition, the proposed scheme requires modifications to the existing CSMA/CA protocol, thus it is considered to be impractical.

In [29] the proposed mechanism also tries to address the problem of backoff manipulation but it follows a different perspective. The estimation of the backoff value is done by observing the throughput of the neighboring node. Each node collects and record the number of successful neighbor's

transmissions over a period of time. A node then can examine if a node in it's vicinity is transmitting much more than him. The detection is based on a threshold related to the number of nodes and a constant dependable on the network parameters. The neighbor list of each node is enhanced with a misbehaving score which is incremented every time a node exceeds the above threshold. The proposed scheme offers an aging mechanism based on a weighted mean of the misbehavior score in order to provide redemption to nodes that misconduct in the past but now are well behaved. However the foremost weakness of the mechanism is that it can only detect cheating with a much lower CWmax than the other users have. In addition, simulations have shown that identification using this mechanism gets harder with the increase of nodes. Finally, the simulations tested the mechanism only in situations where the nodes are static, however, mobility is one of the basic characteristics of MANETs and a detection mechanism needs to sustain high nodes' mobility without great accuracy loss in order to be effective.

The mechanism proposed in [28] is dedicated in detecting backoff manipulation. It is based on measuring the backoff time between consecutive successful transmissions. The mechanism collects a backoff time sequence of a node using another mechanism presented in [47]. The latter mechanism describes a method for both 1-hop and 2-hop neighbors to derive the backoff time between two consecutive successful transmissions. Therefore, [28] focuses on how to determine the non-randomness in the selection of CW values by a node. The authors use a technique called CUSUM (Cumulative Sum) which detect the distribution change of observations as quickly as possible. They argue that IEEE 802.11 DCF is a random access protocol so it is considered as a stationary random process. It is assumed, that in the normal case, the average value of the random sequence should be negative and it becomes positive after change. The proposed mechanism has a short detection delay since it can start detecting misbehavior right after the 50^{th} transmission. In addition, the detection accuracy is high and the mechanism can operate without modifying the protocol implementation. However, as in the case of DOMINO [33] the mechanism accuracy is expected to deteriorate in case of interference. Last but not least, the calculation of CW values are not considered sufficient since it does not take into account the protocols used in session layer (TCP or UDP) that play a significant role in the accuracy of CW values as presented in [33].

In [36] the authors suggested modifications to the 802.11 DCF mode in order to mitigate the negative impacts in the presence of selfish nodes. In fact, they proposed a Predictive Random Backoff algorithm (PRB) that tunes the Binary Exponential Backoff algorithm to generate reproducible backoff

values [42]. That way, receivers can monitor and detect possible misbehavior resulted by the manipulation of backoff parameters. As argued for other solutions of the same perspective, this scheme although it alleviates the impact of selfish behavior, it is found to be impractical due to the need of changes to the protocol itself.

Authors of [38] are approaching the problem by focusing on the effect of misbehaving senders in the absence of an arbiter. More specifically, they adopt the minimax robust detection approach where the goal is to optimize performance for the worst case instance of uncertainty. In fact, the sender and the receiver strike a mutual agreement on the expected backoff values at the beginning of the communication. This model works on the logic that as long as one of the two nodes is honest, the system is free of cheaters.

Detection Mechanism	Basic Characteristics	Strengths	Weaknesses
DOMINO [33]	Comparing average backoff values to given threshold	Detailed observation of backoff value	Interference may deteriorate detection's accuracy
	Designed for 802.11 infrastructure networks	Small implementation overhead	Vulnerable to adaptive cheating
		High detection rates	Energy demanding if implemented in MANETs
New backoff scheme [30]	Modifications to the backoff scheme by extending neighbor lists	High detection rates when misbehavior rates increase	Requires modifications to the protocol, thus it is impractical
		Small implementation overhead	Low detection rates in cases where misbehavior stays in low rates
Throughput-based detection [29]	Detection of misbehavior with nodes' throughput observation	Observation of throughput can be done easily	Throughput is greatly depended on above layer protocols and fluctuations of it does not necessarily indicates attack state
		Introduce an aging mechanism based on weighted misbehavior scores	The mechanism is not tested in nodes' mobility scenarios
Detection using CUSUM statistics [28]	Detection through observation of distribution changes on the backoff time sequences of a node	Short detection delay	The calculation of backoff time sequences is not accurate since it does not take into account factors that affect the backoff observation
		No protocol modifications	
		High detection accuracy	
Predictable Random Backoff (PRB) [36]	Modifications to the backoff scheme in order to produce predictable backoff values	Mitigates significantly the misbehavior impact	Requires modifications to the protocol, thus it is considered impractical
Minimax robust	Method based on	High detection rates	It is prone to adaptive

detection [38]	sequential detection procedures		cheating

Table 2 Strengths and weaknesses of backoff manipulation detection schemes

3.6 Time-out attack

Time-out attack is a new type of attack described in [31] which exploits the three timeout intervals that DCF defines (CTS, DATA and ACK timeout). CTS timeout is computed by the sender after transmitting an RTS and during this interval the sender expects a CTS from the destination. Accordingly DATA timeout is computed from the destination node after transmitting a CTS. ACK timeout is computed by the source after sending a DATA frame. This attack can be implemented in the following ways:

A) A malicious destination that selects a larger SIFS value deliberately delays the arrival of the CTS/ACK until its corresponding timeout has expired at the source. The source then is forced to timeout every time until it drops the data packet and reports the link breakage.

B) A malicious source selects a smaller SIFS value and timeouts before the CTS/DATA arrives back from the destination. After several unsuccessful attempts the malicious source drops the packet and reports the link breakage to the routing layer.

This attack can disrupt the route discovering process from discovering routes through them; forcing packets going through non optimal routes.

3.6.1 Detection of Time-out attack

Although the attack has been presented as a new type of attack according to our classification it falls down the IFS manipulation attacks, as it is based on SIFS manipulation. In more detail, time-out attack focuses on the side-effects of SIFS manipulation in the transmitting process, thus it cannot be considered as new.

The only proposed detection mechanism, in our concern, is the one introduced by the authors of [31] who also had first described this type of attack. The detection mechanism distinguishes two scenarios where the transmitter or the receiver plays the role of the malicious node. In both scenarios, the philosophy behind the detection mechanism is the same. As long as a node does not respond

according to the protocol rules is considered malicious. However, depending on specific rules, related on packet arrival times, its *badCredit* parameter is either heavily or slightly increased. This parameter is used by every node to evaluate the trustworthiness of every node that it communicates with. The most important factor of the mechanism is that it also includes a reaction scheme which allows a node to adjust its timers in order to mitigate the impact of the misbehaving node.

3.7 Selective Dropping

Selective dropping mainly aims to disrupt the normal communication and force the victims to retransmit their data causing severe delays. When a malicious node is receiving an RTS/DATA packet it does not reply by sending an CTS/ACK message. The sender believes that a collision occurred and he doubles his CW and after performing the backoff procedure tries to retransmit the DATA packet. Apart from imposing great delays to the victims an attacker who drops a large percentage of CTS/ACK messages can gain a significant advantage in capturing the channel because all the adjoining nodes will have to use their backoff mechanism.

3.7.1 Detection of Selective Dropping

DOMINO [33] manages to detect this kind of attack by monitoring the fluctuations on the number of retransmissions among the nodes of the 802.11 infrastructure network. The idea behind this test is that the average number of retransmissions of a selfish node would be less than that of the other stations. Inserting a tolerance parameter allows the designers to tune it according to their strictness of their policy. Although this solution can be effective in infrastructure networks, in MANETs the high mobility of nodes combined with the hidden terminal problem can create situations in which some parts of the network may face high drop packet ratio. This phenomenon may cause increase on the false alarm ratio of the mechanism.

4. Network layer attacks and detection methods

After examining the MAC layer attacks and the current detection methods, we move on exploring the foremost attacks implemented in the Network layer by following the same process as in Chapter 2. In particular, we discuss the formation of the attacks and describe the most recent detection methods that have been proposed to detect them. Although, several attacks can be launched against network layer our research focus on the most sophisticated of them, namely the wormhole, rushing and blackhole.

Our contribution in this chapter is a comprehensive description of AODV protocol functionality that is needed in order to follow the chapter. Moreover, we analyze only the most complicate attacks in this layer in order to provide the base for the creation of more accurate detection methods. In addition, the critical analysis of the current detection methods will enlighten the research community for the strengths and the weaknesses of these methods.

4.1 AODV protocol functionality

AODV (Ad Hoc On-Demand Distance Vector) is a reactive routing protocol, meaning that every node sets up a route on-demand at the start of a communication session and uses it till it breaks. The algorithm's foremost aim is to broadcast discovery packets only when necessary. AODV introduces three types of packets, the RREQ (Route Request), the RREP (Route Reply) used during route discovery phase and the RRER (Route Error) used during the route maintenance phase. Every node of a network, according to AODV, can be classified as a source node, an intermediate node or a destination node.

To begin with, we discuss the route discovery phase using an example illustrated in Figure 2. Suppose node A wants to communicate with node O and it does not have any path for it. Since no path exists, source node A initiates the path discovery phase by broadcasting a RREQ packet to its neighbors. The RREQ contains the following fields <*source_address, source_sequence, broadcast_id, destination_address, destination_sequence, hop_count*>. The source sequence number is used to maintain freshness information about the reverse route to the source. Moreover, the pair <*source_address, broadcast_id*> uniquely identifies a RREQ. The broadcast id is a counter incremented each time a node sends a RREQ message. When a node receives a RREQ it first check the pair <*source_address, broadcast_id*> of the packet to see if it has already received the same RREQ. If this is the case the packet is discarded and the RREQ is not forwarded (see node D in Figure 2). In any

other case the intermediate node checks if it has a route for the destination. It searches its routing table and if it finds a route to the destination it compares the destination sequence number found in the RREQ packet with the destination sequence number stored in the routing table. If the one stored in the table is equal or bigger than the one in the RREQ it means that the route stored by the intermediate node is fresh enough, thus the node sends a RREP message to the source. In any other case the intermediate node rebroadcasts the RREQ message to its own neighbors after increasing the hop count by one. In our example all intermediates do not have a valid route for node O, therefore, the RREQ finally reaches the destination, namely node O. Node O generates a RREP message and sends it to node A as illustrated in Figure 2. Every node that forwarded a RREQ message records the address of the neighbor from which it received the first copy of RREQ and maintains it for at least enough time for the RREQ to traverse the network and produce a reply to the sender. This way as the RREQ travels from source node A to destination node O it automatically sets up a reverse path from all nodes back to the source. It is not rare for a node to receive multiple RREPs, in this case, it will select the RREP whose destination sequence number is the largest amongst all previously received RREPs. But if all the destination sequence numbers are the same, the node will select the RREP whose hop count is the smallest.

When a node tries to forward a packet to the next node and fails it means that the next node became unreachable. In this scenario the node generates and broadcasts a RRER message indicating that a link breakage occurred. Upon notified of a broken link, source nodes can restart the discovery process if they still require a route to the destination.

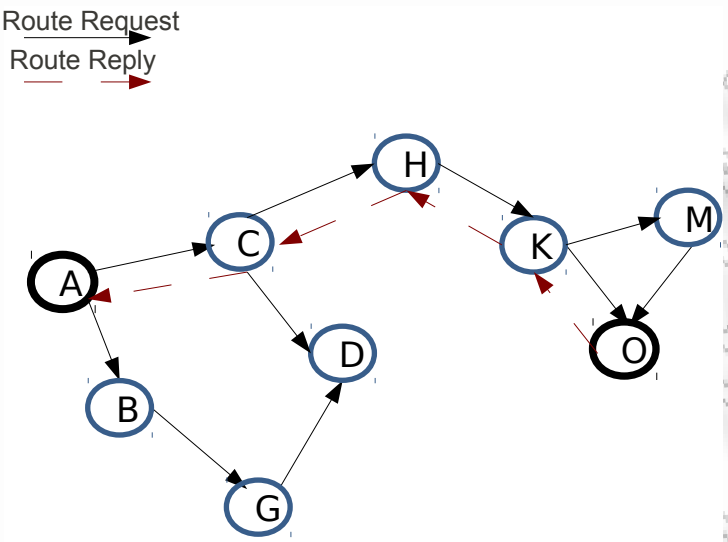

Figure 2: Route Discovery phase in AODV

4.2 Sinkhole of blackhole attack

A black hole attack is a denial of service attack where a malicious node can attract all packets and then absorb them without forwarding them to the destination. The attacker can attract all packets, and eventually invade in a lot of routes, either by forging RREQ and RREP packets either using other attacks such as rushing and wormhole attacks (Figure 2). As presented in our attack tree blackhole attack has two phases, the first one is the route invasion phase, in which the attacker tries to invade routes and the second is the data dropping phase in which the attacker does not forwards the accepted data. Blackhole attack affects the data packet forwarding phase and not the route maintenance. It is crucial to know the procedures followed by an attacker to modify RREQ and RREP messages in order to invade in as much routes as possible. Route invasion is considered a more intricate procedure than data dropping and therefore it is examined thoroughly.

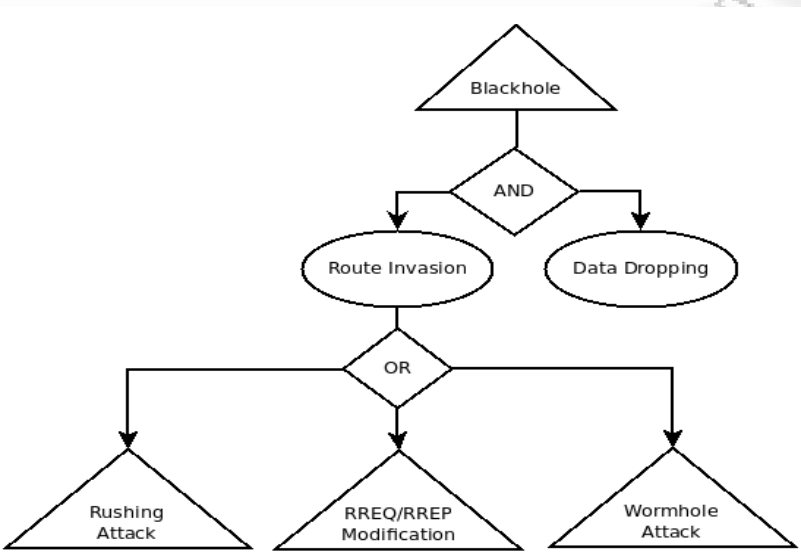

Figure 1: Blackhole attack tree

In fact we present in detail all the ways of invading a route that will help us to build more effective detection mechanisms. More specifically, an attacker can intrude a route by modifying or forging RouteRequest or RouteReply messages in many ways as presented in [1].

At first we examine the Route Request modification. In this scenario we assume that the attacker is inside the transmission range of the source who begins the route discovery. In the first scenario (Figure 3) we suppose that node X broadcasts a RREQ to establish a route to node Z and the attacker A is in the transmission range of X and receiving the RREQ. The attacker modifies the RREQ by increasing the RREQ_ID by at least one, increasing the originator sequence number and the destination sequence number by at least one, and broadcasts it to its neighbors. As a result the neighbors will accept the faked RREQ messages (due to new RREQ_ID) and they will update their next hop to the originating node as the attacking node. When X receives the modified RREQ it just drops it. When Z also receives the fake RREQ it updates his next hop to the one that he received the faked RREQ and then updates its own sequence number to the destination sequence number in RREQ. When Z unicasts the RREP through the inverse path it contains the attacker node.

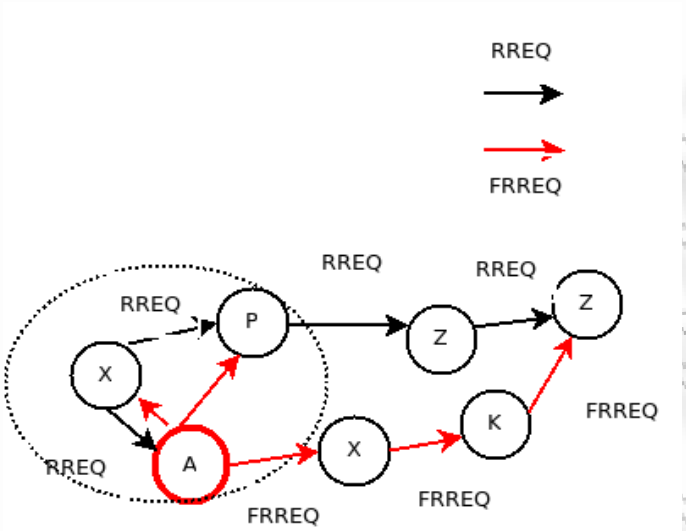

Figure 2: Route Request Modification Example

An attacker can also forge Route Request packets except from modifying them. In the second scenario we assume that the attacking node is not in the transmission range of the source and he is not receiving any RREQ. The attacker tries to invade a route between two communicating nodes through RREQ active forging (Figure 3). Suppose nodes 0 through 5 are normal nodes, and node A is a malicious node. Further assume there is a route form node 0 to node 5. In order to invade the route, node A forges the first RREQ message by setting the originator IP address as node 5, the destination IP address as node 0, originator sequence number to a number greater than node 5's current sequence number and the source IP address (in the IP header) as node A. Node A then broadcasts the faked RREQ message. After receiving this message, nodes 2 and 3 will both set node A as the next hop to node 5 as in Figure 3b. To further establish the route from node A to node 5, the attacker generates this second RREQ message by setting the originator IP address as node A, the destination IP as node 5, the destination sequence number to a number greater than node 5's current sequence number and the source IP address (in the IP header) as node A. Node A then broadcasts this RREQ message, which helps node A establish a route to node 5, as shown in Figure 3c.

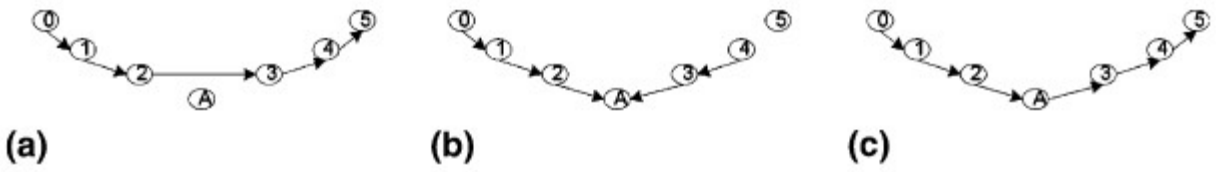

Figure 3: Route invasion using two fake RREQ messages

Another way for an attacker to invade routes is by manipulating Route Reply messages. However, the attacker must already be in a reverse route involving a victim node so that it can receive a RREP message, or send a forged RREP through some other nodes. Due to this fact the attacks have limited impact. As in the case of RREQs we first examine the scenario of modifying Route Replies messages. This attack make sense only if more than one RREP messages exist for the source to choose for. The attacker suppresses the other RREP messages by increasing the destination sequence number of the RREP message by a small number. The originating node will update its route table by the faked RREP message that has the greatest destination sequence number thus, choosing the route involving the attacker.

Attackers can also forge Route Replies to cause the same results. In the AODV protocol, normal node trusts all other nodes, and updates its routing table according to the received RREP messages, even if it has not generated or forwarded a corresponding message before. An attacker can invade a route by sending a faked RREP actively. The assumption here is that the attacker has already a route to both the originating and the destination nodes of an existing route (Figure 4a). The attacker can invade the route by sending a fake RREP message to the originating node, In Figure 4 we assume that node A is the attacking node, which already has a route to nodes 0 and 3, respectively. Node A can forge a RREP message by setting the originator IP address to the originating node (node 0), the destination IP address to destination node (node 3), the destination sequence number to destination node's sequence number plus at least one, the source IP address (in the IP header) to the attacking node (node A) and the destination IP address (in the IP header) to one intermediate node (node 1). Node A then sends the faked RREP message to node 1 which forwards the faked RREP message to node 0 (Figure 4b). When nodes 0 and 1 receive the faked RREP message, they will update the sequence number of node 3 in their routing tables to the destination sequence number in the faked RREP message. Node 0 will still use node 1 as the next hop to node 3, but node 1 will update node A as the next hop to node 3. As a result, node A successfully becomes a part of the route from node 0 to node 3 (Figure 4c).

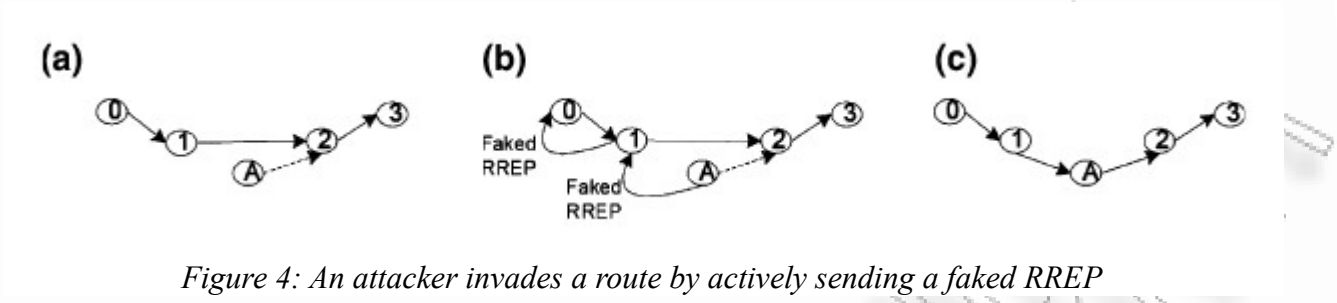

Figure 4: An attacker invades a route by actively sending a faked RREP

4.2.1 Methods of detecting Blackhole attack

One primitive solution to vanish the RREP forging is to disable the ability to reply in a message of an intermediate node, so all reply messages should be sent out only by the destination node [5]. This will result in great delay especially in large networks and in addition the attacker can fabricate a reply message on behalf of the destination node.

Gisung Kim et al. [3] proposed a solution in which the network can identify a malicious node which is acting as a blackhole. Their mechanism detects fistly a blackhole activity in the network and then an algorithm allows the detection of the exact blackhole node. The detection of malicious activity is based on a simple rule applied on each RREQ message. More specifically, this rule states that if a node receives a RREQ whose source id is equal to the id of the receiving node, it checks the sequence number. Then, if the sequence number of the RREQ is greater than the current sequence number of the node, then the node recognizes the existence of sinkhole and understands that this RREQ is from a malicious node. Afterwards, an algorithm is activated that tries to identify the exact blackhole node in a co-operative way. The nodes that received the bogus RREQ can be compared to find a common part between them. The last node of the common path is determined as a suspicious node. The nodes communicate through two types of messages, SDP (Sinkhole Detection Packet) and SNP (Sinkhole Node Packet) in order to find the sinkhole node. If a node receives a RREQ whose source id is equal to the id of the receiving node, it checks the sequence number. Simulations showed that the scheme manages to detect both strong and weak blackhole attacks with the same efficiency. Moreover, it offers good detection time and acceptable rates of false positive. One important characteristic though is the robustness regarding the number of the malicious nodes. On the other hand, the rule being used identifies blackhole activity only by examining RREQ messages. However, as we described above, according to [1] route invasion can also be accomplished by manipulating RREP messages, although with smaller impact. Therefore, we believe that because of the mechanism incapability of detecting all

kinds of route invasion it can be characterized as incomplete. In addition, we consider as complete detecting mechanisms only those that try to identify both phases of the attack, namely the route invasion and data dropping phase.

Mehdi Medadian et al. [4] proposed that every node should use a number of rules to inference about the honesty of the RREP sender. If a node is the first receiver of a RREP packet, it forwards packets to source and initiates judgment process for the specific RREP sender. The judgment process is based on the opinion of network's nodes about the RREP sender. The activities of a node are logged by its neighbors and when it is requested they send their opinion about a node. When a node collects all different opinions from neighbors, it decides if the replier is a malicious or not. The decision is based on four rules described in their paper. These rules are:

- Rule 1: if a node delivers many data packets to destinations, it is assumed as an honest node.
- Rule 2: if a node receives many packets but don't send the same amount of data packets, it's possible that the current node is a misbehavior node.
- Rule 3: When the rule 2 is correct about a node, if the current node has sent a number of RREP packets;therefore surely the current node is misbehavior.
- Rule 4: When the rule2 is correct about a node, if the current node has not sent any RREP packets; therefore the current node is a failed node.

Simulations have shown that the detection mechanism functions well with no significant overhead and manages to preserve throughput rates in decent levels even in situations where the network is under attack. However, no simulations had been taken regarding the false positive and false negative levels of the mechanism which are considered essential for judging a detection mechanism. Moreover, the rules that have used are considered incomplete due to the fact that they fail to detect blackhole activity caused by the manipulation of RREQ and RREP messages presented in [1]. This is mainly due to the fact that the detection mechanism focuses only on the second phase of the attack, namely the data dropping phase. Finally, the rules imply that thresholds are used to identify sinkhole nodes. Threshold-based detection though, has the essential weakness of being deceived by clever adaptive cheating adopted by users that are already aware of the detection mechanism.

Homgei Deng, Wei Li, and Dharma P. Argwal proposed a method [5] to surmount the blackhole problem. The scheme assumes that every node that sends a RREP adds also the extra information of the

nexthop which allows the source to identify the replier's honesty. Therefore, when a source of a RREQ receives a RREP from an intermediate node the source sends an extra request called *Further-Request* to the next node (information that is known from the RREP that is already received) and examine if the replier has actually a path to the destination. Due to the great overhead that the mechanism introduces the authors suggest its usage only in cases whenever the network finds a suspected node. The authors have not made any simulations of the mechanism's usage thus, factors such as detection time, false positive and false negative are not provided. However, we believe that as in the case of [4] the authors have not considered the possibility of an attacker manipulating RREQ packets in order to invade in the route process.

H,C Tseng and B.J. Culpepper proposed two rules in [2], sequence number discontinuity and route add ratio in order to detect sinkhole activity. The sequence number discontinuity is the quality perceived by nodes receiving out-of-order, missing, or duplicated sequence numbers in the routing messages emitted by a neighboring node. It is measured by the overall average difference between the current and the last sequence number from each node, plus a penalty that is proportional to the number of observed duplicate sequence numbers. The route add ratio is the proportion of routes that traverse a particular node to the total number of routes added to this node's routing table. Here we try to analyze the routes being added to each route cache. By its very nature, a sinkhole attack causes nodes in the network to add routes that pass through the sinkhole. It's important to mention that both variables must have values greater than the corresponding threshold values before the system issues an intrusion alert. Simulations shows that the mechanism is efficient and has no false positives. In addition, the authors describe as a problem to be solved a way to choose an optimal threshold value. However, as stated in the case of [4] the threshold-based detection are incapable of detecting adaptive cheating. Moreover, if the attackers are aware of the detection mechanism they can choose the right parameters accordingly and the final outcome is below the threshold but still having an impact to the network's performance.

Method	Basic characteristics	Phase Detected	Strengths	Weaknesses
Co-operative sinkhole detection [3]	Exchange of special messages called SAP,SNP,SDP	Route Invasion	Efficiency	Does not detect modification of RREP messages
			Low false positive	
			Low detection time	
			Robust to number of attacking nodes	
4-rule detection [4]	Introduces the judgment process in which the different opinions are affecting the final decision	Data dropping	No overhead	No simulations for false positive and false negative
			Preserve throughput rates under attack	Does not detect modification of RREQ nor RREP messages
				Vulnerable to adaptive cheating due to the usage of thresholds
Method proposed in [5]	It is used to ascertain the malicious activity in the network	Route Invasion	-	Based on the assumption of no group attacking
				Requires modifications to the existing routing protocol
				It is not test via simulations
				Creates significant overhead
				Does not detect RREQ modification
SIIS method [2]	The method is based on two parameters that are constantly monitored the sequence number discontinuity and the Route add ratio	Route Invasion	Provides efficiency	Vulnerable to adaptive cheating due to the usage of thresholds
			No false positives	

Table 3 Strengths and weaknesses of blackhole detection mechanisms

4.3 Rushing attack

This attack takes place during the route discovery phase and can be seen as a "race" between the legitimate RREQs and the adversarial variant of them. If an adversary successfully reaches some of its neighbors with its own version of RREQ message before they receive a version through a legitimate route, then those nodes will ignore the legitimate version and will propagate the adversarial version [10]. Rushing attack exploits the fact that on-demand routing protocols only forward the RREQ that arrives first from each route discovery.

An example of a rushing attack can be seen in Figure 6 [9] where node A initiates a route discovery for node G and C as the attacking node receives the RREQ and broadcast it immediately. The RREQ through B or any other good node take some time to check whether the particular route is available. The request from malicious node first reaches the node G. As a result the initiator will be unable to discover any usable routes.

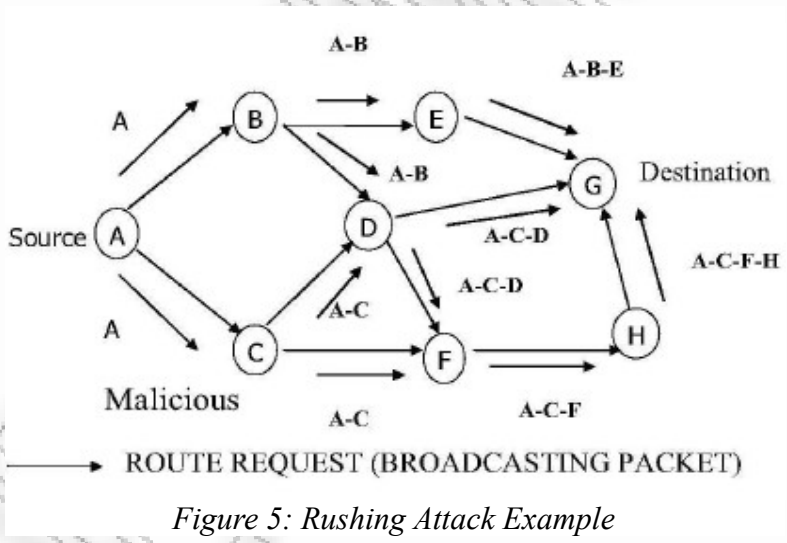

Figure 5: Rushing Attack Example

Rushing attack, like blackhole attack, it is considered as a two phase process. At first, the attacker needs to invade to as many routes as possible and then at phase two, he/she can start dropping data packets, or manipulating with any other malicious way. Therefore, both phases are required to be accomplished in order to characterize a node as a rushing attack or blackhole node.

Some interesting observations are made in [8] where the authors examined the impact of rushing attack regarding the location of the attacker. Through simulation they observed that the best

position to launch rushing attack, namely having the highest success rates, is near the receiver of the RREQ. Near the sender of the RREQ attack success rate is lower and anywhere else in the network the success rate is limited.

Figure 7 shows us the attack tree of the rushing attack. The attack tree describes only the route invasion phase discussed previously. Nevertheless, the rushing attack follows the structure of blackhole attack tree presented previously. In order to beat regular nodes the attacker must acquire relatively small latency or relatively large link speed. The first can be done imposing delays in 802.11 MAC protocol (disrupt back-off scheme in CSMA/CA). Routing protocols also impose a randomized delay in RREQ forwarding, for collision avoidance, that can be disrupted. Large link speed can be reached using higher transmission power levels, or through a wormhole tunnel or by keeping the neighbor's transmission queues full.

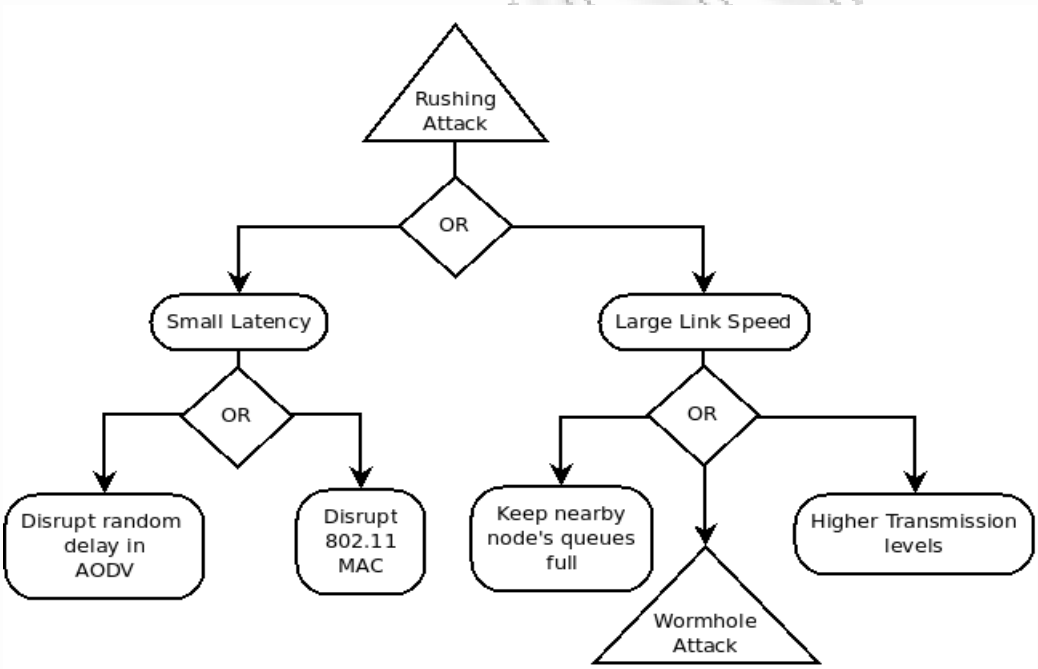

Figure 6: Attack tree of rushing attack

4.3.1 Detection methods of Rushing attack

As we mentioned before one way of implementing rushing attack is by disrupting 802.11 MAC layer mechanisms. For these attacks (back-off scheme attack, NAV attack) indicators have been proposed in MAC layer section (Section 2). There is lack of detection mechanism capable of detecting disruption of random delay in AODV or nodes that are using higher transmission levels.

Yih-Chun Hu et al. in [6] proposed a generic defense mechanism, the Rushing Attack

Prevention (RAP). The authors introduce a set of mechanisms integrated in the new secure protocol to thwart rushing attacks. More specifically, the define the usage of a secure *Neighbor Detection* scheme, a secure *route delegation mechanism* and a *randomized Route Request forwarding*. However, this solution tries to prevent rushing attack through modification of the existing protocols and through creation of new protocols, thus is is considered impractical for direct implementation purposes.

Latha Tamilsevan and Dr.V. Sankaranarayanan proposed a method to prevent rushing attack [9].In their solution every node doesn't forward the first RREQ it receives, instead it waits for some amount of time, collects all the request via different nodes and randomly selects a request to forward. The time for which every node waits is proportional to its distance from the source. The probability that the request selected to forward belongs to the attacker is greatly reduced. On the other hand, this solution creates great deal of network overhead from the extra information exchanged by nodes. In addition, as simulations have shown, the mechanism also introduces greater end to end delays in comparison to DSR routing protocol. Finally, we believe the mechanism to be ineffective if the adversary has compromised more nodes, thus increasing their chances of their RREQs messages being selected by innocent nodes.

Moitreyee Dasgupta, S.Choudhury and N. Chaki have presented a new protocol to obstruct rushing attack but also other types of DoS attacks [7]. Their mechanism called S-HTMRP (Secure Trust Model & Rushing attack Prevention) introduce the notion of *weight* which replaces the hop-count parameter. This new parameter is a value that indicates dynamically the reliability of each route path. Every node maintains this value for all its neighbors. In addition, the traditional packet forwarding method has also been replaced by a Randomized Route Request Forwarding method with the same philosophy of [6] and [9].

4.4 Wormhole attack

We define as wormhole attack the attack where two malicious nodes that are separated by a large distance of several hops build a direct communication link called a tunnel and communicate with each other through this tunnel (Figure 8). This communication among attackers can be established through an out-of-band channel (e.g. wired link or with the usage of high powered transmission) or through an in-band channel using encapsulation [12] . Affected nodes do not have a true picture of the

network, which may disrupt the localization-based schemes, which results in wrong decisions. This type of attack is possible even if the attacker has not compromised any hosts and even if all communication among network nodes provides authenticity and confidentiality.

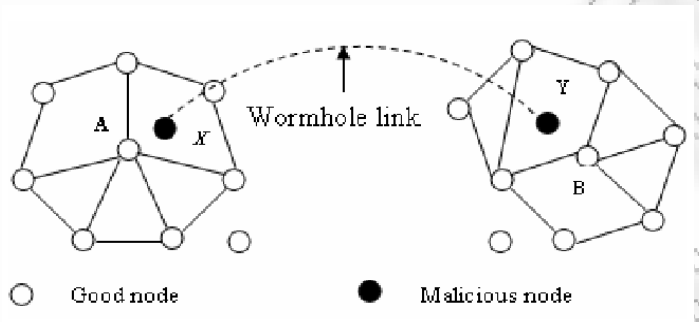

Figure 7: A network under wormhole attack

A significant part in identifying wormhole activity plays the detection of the link that colluding nodes use to transfer data. In out-of-band wormhole usage the link that is established between the colluding nodes is a wired link or a long range wireless transmission [14]. In in-band wormhole the attackers does not use an external communication medium to develop the link. They develop a covert overlay tunnel over the existing wireless medium [13]. Usually in this scenario it also exists a third attacking node that acts as an application-layer relay for wormhole traffic between the wormhole end points. Using an in-band link the attackers can create an extended or a self contained wormhole [11]. In the first case the colluding nodes creates a wormhole that extends beyond the attackers forming the tunnel endpoints. A false link is advertised between two nodes that are not the attacker nodes. In the second case, the attackers creates a stealthier wormhole, advertises a false link between the attacker nodes themselves.

Studying papers [11] [13] we derived the following attack tree (Figure 9). Like blackhole and rushing attack, wormhole attack is also perceived as a two phase process starting with the route invasion phase and ending with either data dropping or any other malicious manipulation of data. However the attack tree presented in Figure 9 refers to the data invasion phase. The purpose of the attack tree is to help us understand all the possible ways to implement a wormhole attack thus create an effective detection mechanism that can foresee all these possible situations.

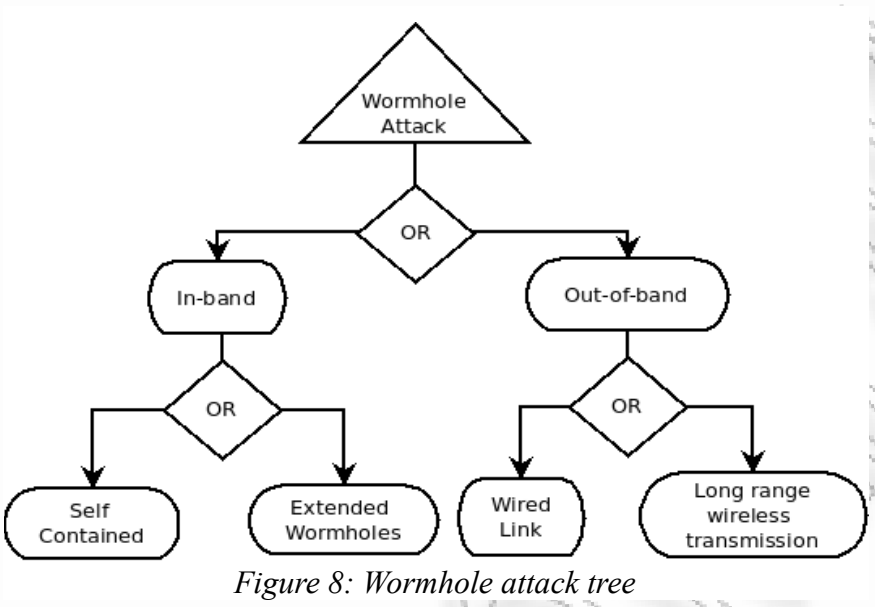

Figure 8: Wormhole attack tree

4.4.1 Detection methods of Wormhole attacks

Indeed there are plenty of detecting mechanisms that have been proposed to surmount the problem of wormhole attack each of one with different requirements. In this paragraph we will try to survey the most recently proposed mechanisms. We categorize the schemes according to the type of wormhole attack that try to detect.

4.4.1.1 Out-of-band wormhole detection mechanisms

One way to detect out-of-band wormhole attacks is by using the notion of packet leashes. According to Yih-Chun Hu et al. in [16] a leash is any information that is added to a packet designed to restrict the maximum allowed transmission distance. At their paper they proposed two different kinds of packet leashes, geographic leashes and temporal leashes. In geographic leashes, each node knows its precise position and all nodes have a loosely synchronized clock. Each node, before sending a packet, appends its current position and transmission time to it. The receiving node, on receipt of the packet, computes the distance to the sender and the time it took the packet to traverse the path. The receiver can use this information to deduce whether the received packet passed through a wormhole or not. In temporal leashes, all nodes are required to maintain a tightly synchronized clock but do not rely on GPS information. When temporal leashes are used, the sending node append the time of transmission to each sent packet t_s in a packet , and the receiving node uses its own packet reception time t_r for verification. The sending node calculates an expiration time t_e after which a packet should not be

accepted, and puts that information in the leash. Geographic leashes is a robust straightforward solution but it inherits general limitations of GPS technology. Temporal leashes on the other hand is an impractical solution because it requires time synchronization level that is not currently achievable in neither MANETs nor sensor networks.

Another proposed solution by L. Hu and D. Evans [17] suggest all nodes to be equipped with directional antennas. In this technique, nodes use specific 'sectors' of their antennas to communicate with each other. Each couple of nodes has to examine the direction of received signals from its neighbor. Hence, the neighbor relation is set only if the directions of both pairs match. The extra bit of information makes wormhole discovery and introduces substantial inconsistencies in the network, and can easily be detected. This solution is very good for networks relying on directional antennas, but it is not directly applicable to other networks.

Khin Sandar Win et al. proposed a wormhole detection scheme [13] capable of detecting out-of-band exposed wormhole attacks. Every node go into promiscuous mode and monitors its neighbors to see if they are forwarding the receiving packets. Each node creates a table in which every row representing a different neighbor and every column is containing the packets sent and dropped at a specific time from a neighbor. Whenever a node receives a RREP it performs Link Frequency analysis (abnormally high frequency of a link could suggest that it can be a wormhole) on links. If there is no suspicious link it continues routing process. If there is a suspicious link, it uses trust information from RREP to examine the suspicious link. Every RREP contains trust information added from intermediate nodes while forwarding the RREQ. This trust information is drilled from the array that each node keeps. The performance of the scheme in terms of precision of alarms, amount of false positive is considered to be good. The mechanism relies only on the data dropping phase of the attack and the fact that wormhole nodes will be included in a lot of routes. We believe that the mechanism cannot distinguish between different attacks in network layer. On the other hand the mechanism can successfully detect malicious nodes in the network.

Yet another mechanism is proposed by Maria A. Gorlatova et al., which is based on the idea of anomaly detection performed locally from each node [Wor04]. The mechanism tries to identify the delay in packet processing and retransmission associated with the attack steps. Each node use its own periodic messages as a metric. It builds a PSD (power spectrum density) profile based on its hello messages and compare it to the profile of incoming HELLOs. If the incoming messages show PSD

differences above some predetermined threshold function there is a cause for alarm. This method is not easily circumvented by attackers, it is topologically robust and can detect “short range” wormholes. Although the detection accuracy depends on how accurately the tunnel's jitter is distinguishable from the jitter caused by other sources, such as node's processing medium.

Yun Wang et al. described a detection algorithm for out-of-band hidden wormhole attacks [18]. The writers suggest that the hidden wormhole detection lies in identifying the nodes which are under attack by wormhole nodes. The algorithm is distributed and each node is equipped with the algorithm locally. The algorithm consists of the following distinct steps. At first each node discovers its 1-hop neighbors and then broadcasts a message of “1-hop neighbor information” to the nodes within 2 hops. The 2-hop neighbors are receiving this information and records them into its neighbor's information table. Then every node runs the Lookup algorithm, the heart of the detection algorithm. The Lookup algorithm is based on the following rule: If the algorithm identifies at least 3 nodes, which are non 1-hop neighbors, in the intersection of the neighbor sets of u and v where u and v are 2-hop neighbors, u and v have been attacked by a wormhole. At last if the lookup algorithm returns TRUE then the node launches the processing to remove all the suspected nodes from the network.

Majid Khabbazian et al. proposed a detection mechanism based on timing analysis [19]. Timing analysis techniques are based on the fact that a packet can travel at most at the speed of light. Therefore, a node can estimate its distance to a sender by multiplying Packet Travel Time (PTT) by the light speed. PTT is computed using a few rounds of message exchanges. In the proposed mechanism PTT is computed in a new way that does not require any specialized hardware. It is only assumed that each node is able to record the time at which a packet is fully sent/received. Each node can validate vicinity of all its neighbors in two rounds of communication. In the first round, each node sends a signed HELLO message containing its ID and a nonce, and records the time at which the message is fully sent. After the first round each node has a list of all its potential neighbors. In the second round, each node signs and sends a follow-up packet. The follow-up packet includes the time at which the node's Hello message was sent (in the first round), the list of all the ID's in the received Hello messages together with their corresponding nonces and the times at which they were received. After the second round, each node has a list of all its 2-hop neighbors. The proposed mechanism is employing the Maheswari's algorithm [21] to further check the existence of a wormhole.

Wassim Znaidi et al. presented in [20] a wormhole detection mechanism which uses local

neighborhood information. Their algorithm is decentralized, distributed and executed locally for each node of the network and consists of three steps. The first step is the neighborhood discovery phase in which every node i creates a list of 1 and 2-hop neighbors. After that, node i computes the CS, the number of "cyclic structures". The idea here is that if two nodes are declared as 1-hop neighbors and if the network is sufficiently dense, those two particular nodes must have some common 1-hop neighbors whereas it is not the case if the corresponding link is a wormhole. For example a node α could say that its 1-hop neighbor w1 is a wormhole node if it could not reach w2, the other wormhole node and a 1-hop neighbor of w1 through its "own" 2-hop neighbor-list. At last there is the isolating phase when a node finds a wormhole node and broadcasts an alert with the node id and adds him to the red list. When a node w gets enough alert messages higher than a given threshold σ sends black alert messages to all its direct neighbors to isolate the malicious node.

Ritesh Maheshwari et al. proposed in [21] a wormhole detection algorithm that looks for forbidden substructures in the connectivity graph. The main idea is that the placement of wormhole influences the network connectivity by creating long links between two sets of nodes located potentially far away. The resulting connectivity graph thus deviates from the true connectivity graph. The key notion that the writers exploit is a packing argument – inside a fixed region, one cannot pack too many nodes without having edges in between. The forbidden substructures that the algorithm look for, are actually those that violate this packing argument. The forbidden structures that are used in the algorithm are based on the following rule. Two neighboring nodes having Fk independent common k-hop neighbors. Fk is the forbidden parameter of the wormhole detection algorithm and must be more than the packing number for unit distance inside the lune of two disks of radius k placed at distance 1. When nothing is known about the node distribution or communication model, it becomes harder to estimate Fk. The algorithm is localized and distributed. Each node searches for forbidden structures in its k-hop neighborhood. The algorithm can also remove wormhole by dividing nodes to corrupted (can hear wormhole nodes) and uncorrupted (not in transmission range of the wormhole nodes). Each corrupted node verifies its neighbor list with neighbor lists of uncorrupted nodes.

Xia Wang and Johny Wong introduced in [22] an end-to-end detection mechanism for out-of-band hidden wormhole attack. The mechanism assumes that each node is equipped with GPS technology and the network is using shared pairwise secret keys. It is also assumed that a modified AODV is used in which only the receiver can respond the RREQ packet. According to the algorithm at first the source node estimates the minimum hop count to the destination based on the geographic

information of the two end hosts in which the receiver's location is piggy-backed by the RREP during the route discovery. After that the source compares the hop count value received from the RREP with its estimated value. If the received value is less than the estimation, the corresponding route is marked as if a wormhole is detected. At last if a wormhole was detected in the previous step the source launches wormhole TRACING in which the two end points of the wormhole would be identified in a small area meaning that there are multi-paths exist between the source and the destination.

4.4.1.2 In-band wormhole detection mechanisms

Viren Mahajan et al. proposed a scheme which tries to detect self-contained in-band wormhole attacks [11]. The scheme is using delay statistics as a criterion for identifying wormholes. These statistics can generate anomalies on the launch of a wormhole attack. These anomalies can be incompatible hop delays or end-to-end delays. In the first case the advertised routes are much shorter than the actual routes which go through the wormhole tunnel. In the second case a large part of the end-to-end delay for a path consists of hop delays at each hop. The end-to-end delay for such a path will not be explained by the sum of hop delays of the hops present on its advertised path. According to the simulation of this scheme the wormholes with high strength show a higher detection ratio as compared to the wormholes with lower strength.

4.4.1.3 Detection mechanisms capable of detecting both kinds of wormhole attack

Sun Choi et al. suggested the use of WAP: Wormhole Attack Prevention algorithm based on neighbor node monitoring that acts during the route discovery phase [12]. The algorithm is able to detect all types of wormhole attack. The only requirements are the capability of nodes turning in promiscuous mode and a slight modification on routing protocols in a way that an intermediate node cannot reply to RREQs messages. All nodes of the network monitor the activities of their neighbors in their tables and check for malicious behavior among their neighbors. When a node A sends a RREQ the WPT (Wormhole Prevention Timer) starts. WPT is based on the fact that if a hidden wormhole attack is launched, the packet transmission time between two fake neighbor nodes may be longer than the normal transmission time of the hop. If A receives the message after the timer expires, it suspects B or one of B's next nodes to be wormhole nodes. In another example suppose a source node S broadcasts RREQ at time Ta and then receives a RREP at time Tb, the source node can calculate the time delay per hop in the route by using hop_count field in the RREP. The scheme uses a wormhole node list indexed by a wormhole node and a colluding node. Every time a node finds out the existence of a wormhole

and colluding node it broadcasts this information to the rest of the network. Out-of-band exposed detection relies on the length and transmission speed of the wormhole link. As the length decreases or the speed increase, the algorithm fails to detect the attack.

Dang Quan Nguyen and Louise Lamont in [15] described a detection mechanism for all kinds of wormhole attack. The basic idea is that neighbor nodes compare their local reception times of a same broadcast message (called reference message) sent by a common node (called reference node) to determine the offset between their clocks. Any difference in the absolute reception times is due to the different distances between the neighbors and the reference node. Each node needs to keep track of the offsets between neighbors. If α and b are true neighbors, their time offset is constant. If the variation of the time offsets to a particular neighbor α exceeds a certain threshold δ, then node b may consider its communications with α subject to wormhole attack. The idea behind this is that offsets variation between two true neighbors could be non-zero due to their relative velocities. But we expect this variation to be small in normal circumstances (variations of ten nanoseconds of the order of magnitude). Whereas under the fastest wormhole attacks, the amount of time tunneling from one end of the wormhole to the other end and back is not smaller than micro-seconds.

Phuong Van Tran et al. built a transmission time base detection mechanism for all kinds of wormhole attack [23]. The detection mechanism is executed each time a route is requested. The source node is in charge of collecting all RTT (Round Trip Time) values between every two successive nodes. In order the mechanism to have little overhead as RTT is considered the time between an intermediate node sending the RREQ and receiving RREP. Every intermediate node along the route needs to send the RTT between them and the destination back to the source node. The RTT will be sent along with the RREP back to the source node. At last the source node after collecting the RTTs searches for a wormhole among them. A considerably higher RTT value between two successive nodes indicates a wormhole link between those two nodes. To make the decision a threshold is used. The threshold must be selected in order to minimum both false positive rate and false negative rate.

Detection Mechanism	Type of wormhole attack that can detect	Requirements/Limitations	Comments
Geographical packet leashes [16]	Out-of-band hidden	GPS coordinates for every node	Inherits general limitations of GPS technology;
		Loosely synchronized clocks (ms)	Expensive to use widely
Temporal packet leashes [16]	Out-of-band hidden	Tightly synchronized clocks (ns)	Impractical
			Expensive to use widely
Directional antennas [17]	Out-of-band hidden	Directional antennas on all nodes	Good solution for networks relying on directional antennas but not directly applicable to other networks.
Protocol breaking and packet timing analysis [14]	Out-of-band hidden	Nodes capable of building power spectral density profiles (PSD profiles)	The detection accuracy depends on how accurately the tunnel's jitter is distinguishable from the jitter caused by other sources such as node's processing medium contention.
		Usage of HELLO messages at the routing protocol	
Neighborhood Information [18]	Out-of-band hidden	UDG model is applied	Straightforward detection;Approach more suitable for dense networks.
		Wormhole will not replay messages	
		Private/Public key have been deployed	
Time-based detection [19]	Out-of-band hidden	Each node is able to record the time at which a packet is fully send/received	-
End-to-end detection [22]	Out-of-band hidden	GPS coordinates for each node	Method's accuracy is doubted especially when comforting long distance paths.
		Shared pairwise secret keys or PKI	
		Modification in AODV so only the destination reply to a RREQ	
Using			Difficulty in computing Fk for realistic models;

Detection Mechanism	Type of wormhole attack that can detect	Requirements/Limitations	Comments
connectivity information [21]	Out-of-band hidden and exposed	No limitations	Lacks theoretic detection probability analysis;
			Doesn't work when the attacker selectively forwards HELLO messages.
LiteWorp [24]	Out-of-band hidden and exposed	Requires a pre-distribution pair-wise key management protocol	Introduce other attacks such blackmail attack through impersonation;
			Applicable only to static stationary networks.
Link Frequency analysis and Trust based model [13]	Out-of-band exposed	Nodes always on promiscuous mode	The method cannot distinguish between wormhole and sinkhole attack
			Power consuming method, impractical for MANET nodes with restricted resources
Using local neighborhood information [20]	Out-of-band exposed	Topology is static	Practical for sensor networks but not for MANETs where the topology changes dynamically
Detection through delays [11]	In-band	No limitations	-
Detection through monitoring neighbors and delays (WAP) [12]	Both kinds	Nodes always on promiscuous mode.	Detection of out-of-band exposed attack relies on the size and transmission speed of the wormhole link.
		Intermediate nodes cannot reply to RREQs	
Transmission time-based mechanism	Both kinds	No limitations	-

Detection Mechanism	Type of wormhole attack that can detect	Requirements/Limitations	Comments
(TTM) [23]			
Reference broadcasts [15]	Both kinds	Each wormhole node attacks at least two victims that can hear each other	-
		Clocks homogeneous	
		Usage of HELLO messages as reference messages	

Table 4: Comparison of the majority of the detection mechanisms that has been proposed to thwart wormhole attack

5. Intrusion Detection Systems

An important aspect of network security is the ability of detecting and preventing attacks. The detection is feasible through the deployment of Intrusion detection systems (IDS). This chapter examines the usage of IDS in MANETs environments.

At this chapter a classification of IDS is presented. However our main contribution is the proposal of a specification-based detection engine to protect the MAC layer in MANETs. The engine is capable of detecting the majority of the attacks performed at this layer.

5.1 Intrusion Detection classification

Due to their nature MANETs are susceptible to a variety of attacks implemented mainly in MAC, Network and Session layer. In previous chapters (Chapter 2,3) we discussed the majority of the attacks which exploit both MAC and Network layer protocols. It is widely believed that intrusion prevention techniques, such as encryption and authentication, per se are not enough to protect a network. As systems become more intricate, more weaknesses exist which leads to more security problems. Therefore, Intrusion Detection Systems (IDS) can be used as a second wall of defense against these type of attacks [25]. IDS is a sensor mechanism that monitors network activity in order to detect malicious actions. When detecting an intruder, the IDS reacts in various ways from a simple alert notification of the network to more comprehensive defensive actions. Recently, a lot of research is done in intrusion reaction schemes in MANETs that are focus on the most efficient ways of reacting to malicious attacks [26].

IDS are divided in two main parts, the architecture which describes the operational structure of the IDS and the detection engine which is the mechanism used to detect malicious behavior. Thus, IDS can be classified according to their architecture and according to their detection engine they use. In relation to their architecture all existing IDS fall under three basic categories: 1) *stand-alone* 2) *cooperative* and 3) *hierarchical* [25]. In *stand-alone* architectures, all nodes are equipped with an intrusion detection engine which uses local audit data to detect malicious behavior. There is no collaboration among nodes, therefore, these architectures are not capable of detecting some types of attacks and provide limited detection accuracy. For these reasons this architecture is not usually used in MANETs [26]. On the other hand, in *cooperative architecture* all nodes process data locally but they also communicate these data among them resulting in a more accurately and widely detection of attacks. Finally, *hierarchical architecture* as the previous architecture also disseminates audit data

among nodes. The basic difference though, is that the network is divided in clusters and in each cluster some nodes are selected as cluster-heads and undertake various responsibilities in intrusion detection. Cluster-heads are responsible of running comprehensive detection engines that acts as a second layer of detection to the already lightweight detection engines established on each cluster member.

Intrusion detection systems except from their architecture are also divided according to their detection engine they use. In fact we have three different categories: 1) *signature-based* engines, 2) *anomaly-based* engines and 3) *specification-based* engines. The *signature-based* engine compares known attack signatures with current system activities. Although it has a low false positive rate, this type of engine fails to detect new attacks. In addition, it needs a frequently updated signature database. On the other hand, *anomaly-based* engines are capable of detecting new attacks due to the usage of profiles based on nodes' normal behavior. The engine creates profiles based on specific parameters such as the network throughput, the cpu usage and others. It detects intrusions as anomalies from the normal behavior. A major disadvantage is that the engine needs time to be trained on a specific network before it creates the profiles and normal profiles can change over time and the engine should also change accordingly. At last, specification-based engines define a set of constraints on a program or a protocol and intrusions are detected as runtime violations of these specifications. This type of engines came as an alternative which manages to combine the merits of both anomaly-based and signature-based techniques, as it provides detection of known and unknown attacks. However, creating detailed specifications for each protocol and program can be very time-consuming [27].

5.2 A lightweight detection engine for MAC Layer misbehavior

In this section we present a detection engine capable of detecting the majority of attacks performed at the Data Link layer. Studying MAC layer attacks in MANETs and the existing detection engines (see Chapter 2) we concluded that there is a lack of an engine that manages to detect the foremost attacks. As far as we know, the only such engine is the DOMINO [33] which however is implemented in wireless infrastructure networks and therefore, cannot be used unmodified in MANETs due to the lack of a central authority such as the Access Point (AP). Except DOMINO all other engines focus on detecting a single or a small group of attacks. Therefore, our efforts concentrated on creating an engine that is also lightweight in terms of energy and process resources. This can be considered as the first step of building a lightweight engine that detects attacks in the three more important layers, namely the MAC, the Network and the Session layer of TCP/IP.

The proposed scheme defines that each node deploys a MAC layer specification-based detection engine. This engine performs detections using a set of specifications, which describe the normal node's operations, monitoring the most important node functionality. The advantage of this approach is twofold: (i) the overhead of specifications development can be reduced, since aggregated specifications are developed that focus on the most important attacks at layer 2; and (ii) the proposed engine detects the majority of attacks that occur in MANETs, protecting the most critical/significant network operations.

Figure 9 and 10 illustrates the usage of the CSMA/CA protocol for a transmitting and a receiving node respectively. In both figures we can see all the actions that an honest node should do upon sending or receiving packets according to the 802.11 CSMA/CA protocol (a detailed description of the protocol can be found in Chapter 2.1). In green color boxes, we present all the checks that are needed from the engine to detect possible misbehaviors. All the controls are performed when the protocol reaches the specific steps. These steps are considered hazardous of performing malicious actions and therefore are the moments where controls are needed. In addition every control is related with a specific attack that the engine tries to detect. Thus, each attack will be discussed separately explaining the checks performed. Detail for all the attacks in MAC layer can be found in Chapter 2.

5.2.1 Detecting IFS Manipulation

Manipulating IFS parameters a node can decrease its waiting time between transmissions or retransmissions. Our mechanism is capable of detecting this alteration by inspecting the actual protocol parameters, namely the DIFS, SIFS and EIFS time periods using three different checks. If there is a difference between the expected values and those that are used by a node then an alert message is created. The expected values for these parameters are related to the physical layer (see Table 1 for an example). In Figure 9 and 10 we can see in green all three different checks (DIFS, SIFS and EIFS check) dedicated on detecting IFS manipulation. Following this method of detecting IFS manipulation has also another significant advantage, namely the detection also of another sophisticated attack called time-out-attack [31]. This attack is based on exploiting the SIFS parameter (see Paragraph 2.6), and therefore can be detected by the SIFS check.

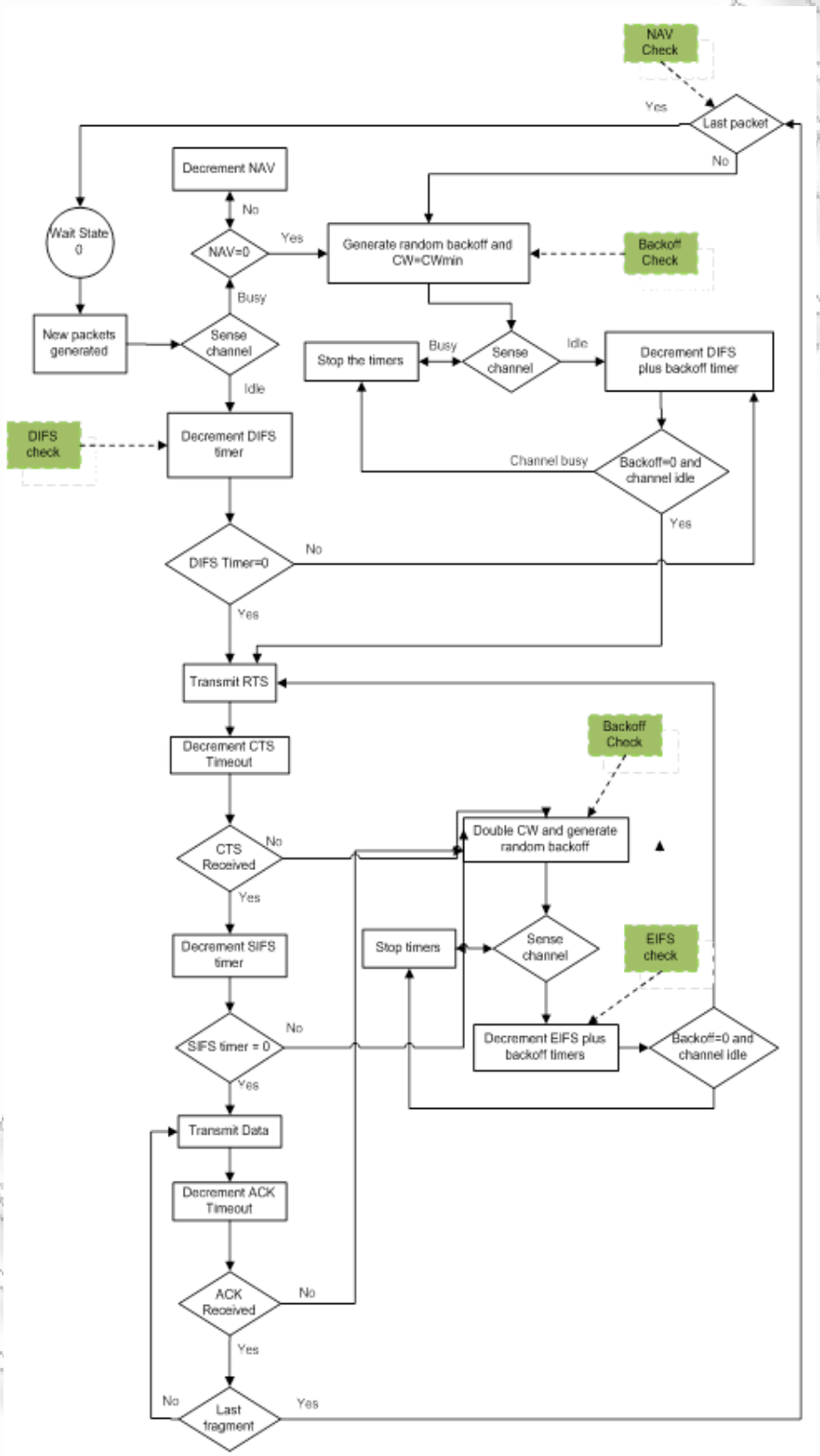

Figure 9: Transmitter's State Diagram with the proper controls

5.2.2 Detecting NAV attack

Our system follows the approach of DOMINO [33] on detecting the NAV attack. We use a simple NAV check to detect nodes that deliberately advertise false NAV values. This check is performed after each packet transmission (see Figure 9) and is based on measuring the actual transmission time of a packet and comparing it with the advertised NAV value. A misbehavior counter enumerates the times a node advertises a false NAV. When the counter exceeds a specific threshold the node is characterized as malicious (Figure 11).

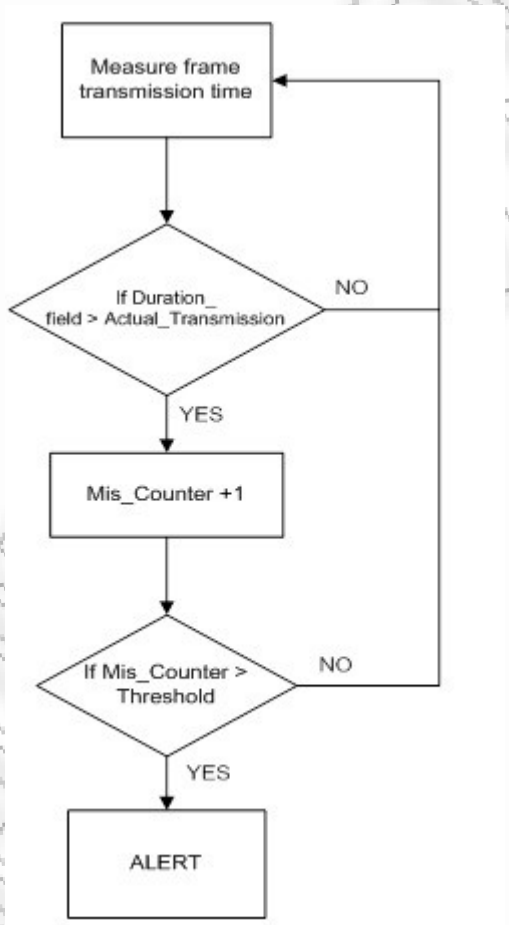

Figure 11: NAV check in detail

5.2.3 Detecting backoff manipulation

Our mechanism detects backoff manipulation using two checks, the CW check and the Backoff Randomness Check. Both checks are performed whenever the backoff mechanism is used by a node (see Figure 9). CW check is used to identify manipulation of the CW parameter, namely changes of CWmin value and denial of doubling the CW value after each collision. This check monitors the

changes of the CW parameter and ascertains that these are performed according to the protocol. Figure 12 illustrates the expected changes of CW and the checks performed to assure the correct function of the protocol. Our mechanism draws its CW values by monitoring the usage of the CSMA/CA protocol. More specifically, at CW value initial state the CW value is equal to Cwmin. The first check here ascertains that the Cwmin selected by the node is the appropriate according to physical layer. If a collision occur the CW value moves state 1 where it must double its CW value. Here the control guarantees that the node doubles the CW value after each collision accordingly. After a successful transmission or when CW reaches the Cwmax the node returns to the initial state where CW resets to Cwmin. On the other hand the Backoff Randomness check identifies if a node chooses its CW values randomly or not.

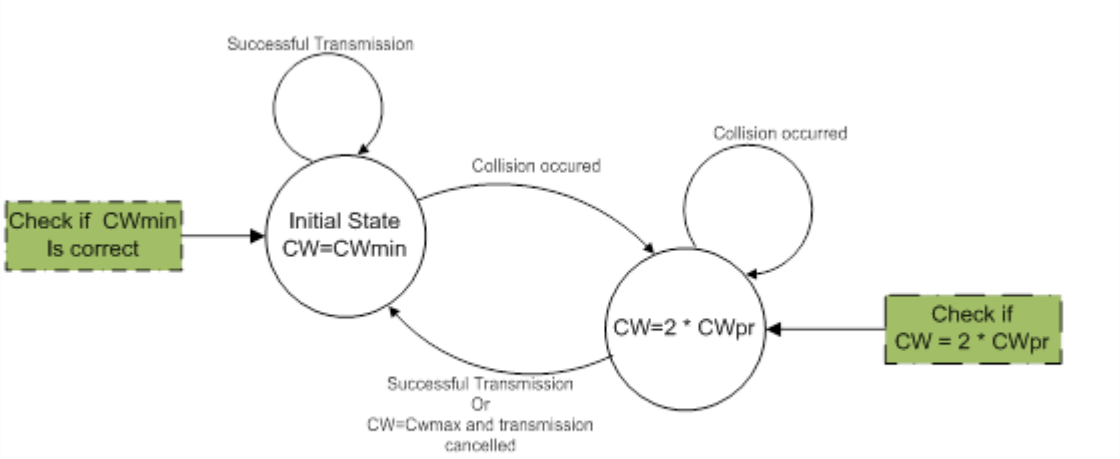

Figure 12: CW check in detail

5.2.4 Detecting Selective Dropping

In Figure 13 we illustrate a set of specifications that facilitate the engine to monitor for selective dropping. It observes whether the monitored node answers to RTS and DATA packets. At the state So it waits for RTS messages, when an RTS arrives it moves to S1 where it sends a CTS message as response. If the node does not transmit the CTS it moves to S3 where it is considered malicious. If the node sends the CTS, as specified by the protocol, it moves to state S2 where it waits to receive the DATA packet. Upon reception of the DATA packet the node moves to state S3 where it needs to

transmit an ACK packet for the data it received. If the node does not send an ACK it moves to state S4 designating that the node is dropping packets. Otherwise, the node will send the ACK packet and move to the initial state So.

5.2.5 Detecting RTS/CTS Spurious Attack

During this attack the attacker send RTS or CTS messages without the purpose of transmitting any data afterwards. Its aim is to cause virtual jamming to all the nodes that receive the RTS/CTS messages. The control here is performed every time the node transmits either an RTS or a CTS message. The RTS control checks if the network layer has forwarded any data to the data link layer for transmission. If no data is forwarded from above layers and the nodes transmits an RTS is considered malicious and the detection engine generates an alarm. Moreover, the CTS control guarantees that no CTS is transmitted if there is no corresponding RTS is received. If the node sends a CTS message but no RTS message is previously received the detection engine as before generates an alarm. These two controls ascertains that RTS and CTS messages are sent only when a normal communication between nodes occurs.

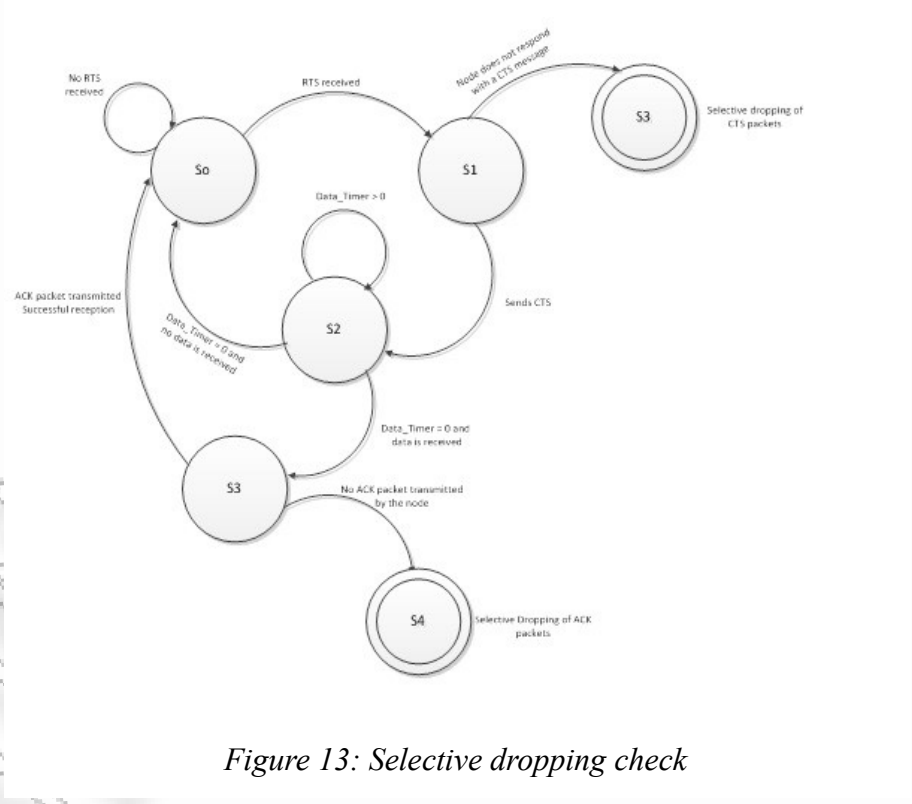

Figure 13: Selective dropping check

6. Conclusions

MANETs are prone to several different types of attacks described at chapters 4 and 5. In fact the most severe types of attacks target the Data Link and Network layers as being the two most vulnerable layers when deploying MANETs. Therefore, the bulk of research effort focuses on protecting the protocols of these two layers.

At the Data Link layer the attacks aim at the MAC sub-layer and especially at the CSMA/CA protocol which is responsible for providing fair chance of accessing the network for all users and mitigating collisions. Virtual jamming attacks such as NAV manipulation and RTS/CTS spurious attacks aiming in disrupting the connectivity among nodes by exploiting the CSMA/CA vulnerabilities. These attacks are easily detectable by detection mechanism [33] [32] with great accuracy although they can be cheated by carefully deployed attacks that stay under the IDS threshold. At the same category they fall two more attacks namely the time-out and the selective dropping attacks. In addition, another category of attacks is selfish behavior which is the most common threat in MANETs since attackers have a significant gain and the resources are limited. Attacks like IFS and backoff manipulation have a severe impact on communications. Backoff manipulation have been addressed by several detection schemes that try to mitigate its impact. However, solutions like DOMINO [33] are complicated, detection by examining a nodes throughput like [29] are not very accurate and solutions like [30] and [38] are considered impractical because they require protocol modifications. It is clear that although several detection mechanisms are capable of detecting an attack or a small group of them, none mechanism is designed to detect all of them. DOMINO [33] is an exception because although it manages to detect the majority of attacks implemented at MAC layer it is designed for infrastructure 802.11 networks where the AP acts also as an IDS. In Chapter 5 we described a lightweight intrusion detection mechanism that is feasible to detect all of the aforementioned attacks with great accuracy. In addition, the rules are very simple compared with other schemes since the mechanism is established at each station and monitors only the activity of this node. Moreover, the IDS uses a specification-based engine which examines the protocol and detects any differentiation from normal protocol behavior. Due to its simplicity it is expected not to create any computational overhead thus, it can be implemented in stringent environments like MANETs.

Network layer is vulnerable to several attacks because the routing protocols rely on the collaboration among nodes. Attacks here are mainly different types of denial-of service attacks and aim in disrupting the network connectivity. In order for an attacker to increase the damage to the network

first needs to invade to as many routes as possible. The distinction of the attacks is based on the way the attacker chooses to invade to routes. Blackhole attack exploits the different features of RREQ and RREP messages to acquire access to different routes. Although several detection mechanisms presented [5,3,4] to mitigate blackhole attack none of them is capable of detecting all the possible ways of implementing the attack. More specifically, some mechanisms detect only the RREQ modification/forging [3], others only the RREP modification/forging [5] and others only the data dropping phase of the attack [4]. None of the mechanisms that we have been examined manage to detect all the possible attack variations. In addition, rushing attack tries to invade to routes by disrupting MAC and network protocols in order to reply faster to all the accepted RREQs. Not much research is done to detect this type of attack since it includes misbehavior on both protocols. Solutions here include protocol modification [6], creation of new protocol and techniques based on random RREQ selection. At last, wormhole attack is one of the most sophisticated attack and thus, very difficult to detect. Attackers can create in-band and out-of-band wormholes in order to transfer data in ways that can mislead the legitimate nodes. Presented detection methods can be expensive due to the need of extra hardware like GPS or Directional antennas [16,17,22]. In addition, some require tight synchronization [19,15,23] or complex computations like the creation of power spectral density profiles [14]. Moreover, other solutions require heavy cryptographic schemes [24,18] which cannot be guaranteed in such restricted environments. For the same reason, solutions that rely on constant monitoring [13,12] cannot be accepted because nodes that are in promiscuous mode can drain their energy resources very fast.

7. Future Work

We proposed a lightweight detection engine for MAC layer misbehavior that detect the majority of attacks carried out at this layer. Although the engine is mainly a specification-based some of the controls, namely the NAV control and the Selective dropping cannot be accurately detected through specifications and some threshold-based rules have been used. Therefore, further analysis for the selection of the thresholds is needed for better configuration of the engine. In addition, the CW randomness test for the detection of backoff manipulation attack needs to be deployed in future revisions. Furthermore, we plan on implementing our engine and testing its accuracy through simulations.

Moreover, network layer solutions are not yet capable of detecting the large spectrum of attacks presented in Chapter 4 and their many variations. Attacks such as rushing attack are not yet extensively researched so further investigation is required. Moreover, blackhole detection engines are require dedicated hardware or are based on unrealistic assumptions. Thus, we intend to build a lightweight specification-based engine that manages to detect the majority of the network layer attacks. We expect that implementing an IDS on each node and by monitoring only this node's activity we can detect accurately all the aforementioned attacks. Therefore, this approach can be very energy and computational efficient and at the same time based on simple rules.

Last but not least, we plan on combining the data link and the network layer engine into one two-layer engine that manages to mitigate attacks with collaboration between the two layers. Our approach can be very helpful in detecting complex attacks like rushing attack in which the attacker exploits vulnerabilities on both data link and network layer protocols to deploy its attack. Furthermore, this cross-layer feature can help to more accurate detection since attacks effects can be spread to several layers.

References

1. Peng Ning, Kun Sun, "How to misuse AODV: A case study of insider attacks against mobile ad-hoc routing protocols", Proceedings of 2003 IEEE Systems, Man and Cybernetics Society Information Assurance Workshop, June 18-20, pp. 60-67, 2003.
2. H.Chris Tseng, B. Jack Culpepper, "Sinkhole intrusion in mobile ad hoc networks: The problem and some detection indicators", Elsevier Computers and Security, vol. 24, pp. 561-570, 2005.
3. Gisung Kim, Younggoo Han, Sehun Kim, "A cooperative-sinkhole detection method for mobile ad hoc networks", Int J Electron Commun (AEU), 2009.
4. Mehdi Medadian et. al., "Combat with Black Hole Attack in AODV routing protocol in MANET", IEEE, 2009.
5. Hongmei Deng, Dharma P. Argawal, "Routing Security in Wireless Ad Hoc Networks", IEEE Communications Magazine, October 2002.
6. Yih-Chun Hu, Adrian Perrig, David B. Johnson, "Rushing attacks and defense in wireless ad hoc network routing protocols", ACM, September 2003.
7. Moitreyee Dasgupta, S.Choudhury, N. Chaki, "Routing Misbehavior in Ad hoc Network", International Journal of Computer Applications,Volume 1-No 18, 2010.
8. V. Palanisamy, P. Annadurai, "Impact of rushing attack on multicast in Mobile Ad Hoc network", International Journal of Computer Science and Information Security, Vol. 4, No. 1 & 2, 2009.
9. Latha Tamilselvan and Dr. V. Sankaranarayanan, "Solution to Prevent rushing attack in wireless mobile ad hoc networks",IEEE, 2006.
10. Baruch Awerbuch et. Al, "Mitigating Byzantine attacks in ad hoc wireless networks", Technical report, March 2004.
11. Viren Mahajan, Maitreya Natu, and Adarshpal Sethi, "Analysis of wormhole intrusion attacks in MANETs", IEEE, 2008.
12. Sun Choi et al., "WAP: Wormhole Attack Prevention Algorithm in Mobile Ad Hoc Networks", IEEE 2008.
13. Khin Sandar Win, "Analysis of Detecting Wormhole Attack in Wireless Networks", World Academy of Science, Engineering and Technology, 2008.
14. Maria A. Gorlatova et al., "Detecting Wormhole Attacks in Mobile Ad Hoc Networks through protocol breaking and packet timing analysis", IEEE, 2006.
15. Dang Quan Nguyen, Louise Lamont, "A simple and efficient detection of wormhole attacks", IEEE 2008.
16. Yih-Chun Hu, Adrian Perrig, David B. Johnson, "Wormhole detection in wireless Ad Hoc Networks", 2002.
17. L.Hu, D. Evans, "Using Directional Antennas to prevent wormhole attacks", 14 Proceedings of the 11th Network and Distributed System Security Symposium, pp. 2003.
18. Yun Wang, Zhongke Zhang and Jie Wu, "A distributed Approach for Hidden wormhole detection with neighborhood Information", Fifth IEEE International Conference on Networking, Architecture, and Storage, 2010.
19. Majid Khabbazian, Hurgues Mercier, and Vijay K. Bhargava, "Severity analysis and Countermeasure for the Wormhole attack in Wireless Ad Hoc Networks", IEEE Transactions on wireless communications, vol. 8, no.2, February 2009.
20. Wassim Znaidi, Marine Minier and Jean-Philippe Babau, "Detecting Wormhole attacks in wireless Networks using local neighborhood information", IEEE, 2008.
21. Ritesh Maheshwari, Jie Gao and Samir R Das, "Detecting wormhole attacks in wireless

Networks using connectivity information", IEEE, 2007.

22. Xia Wang and Johny Wong, "An end-to-end detection of wormhole attack in wireless ad-hoc networks", 31st annual international computer software and applications conference, 2007.
23. Phuong Van Tran et al., "TTM: An efficient mechanism to detect wormhole attacks in wireless ad-hoc networks", IEEE, 2007.
24. Issa Khalil, Sauraph Bagchi, Ness B. Shroff, "LiteWorp: A lightweight countermeasure for the wormhole attack in multi-hop wireless networks", Proceedings of the International conference on dependable systems and networks, 2005.
25. Tiranuch Anantvalee, Jie Wu, "A survey on Intrusion Detection in Mobile Ad Hoc Networks", Wireless/Mobile Network Security, Chapter 7, pp. 170-196, 2006, Springer.
26. Christos Xenakis, Christoforos Panos, Ioannis Stavrakakis, "A Comparative Evaluation of Intrusion Detection Architectures for Mobile Ad Hoc Networks",Computers & Security, Elsevier Science, Vol. 30, Issue 1, pp: 63-80, Jan. 2011.
27. Sevil Sen, John Andrew Clark, Guide to Wireless Ad Hoc Networks, Computer Communications & Networks, Chapter 7, pp. 427-454, 2009.
28. Chunfeng L., Yantai S., Mingyuan L., Yang O.: A new mechanism to detect selfish behavior in IEEE 802.11 Ad Hoc Networks, IEEE ICC, 2009.
29. Nagel N., Shokranian R., Bordim J. : Mac layer misbehavior on Ad Hoc Networks, International Journal of Computer Science and Network Security, Vol.9 No. 11, November 2009.
30. R.Gunasekaran, Dr.V.Rhymend Uthariaraj, R.Sudharsan, S.Sujitha Priyadarshini, U.Yamini, "Detection and prevention of selfish and misbehaving nodes at mac layer in mobile ad hoc networks",2008, IEEE.
31. Guang L., Assi C., : A self-adaptive detection system for MAC misbehavior in Ad Hoc Networks, IEEE ICC, 2006.
32. Sugantha K., Shanmugavel S., : Anomaly Detection of the NAV attack in MAC layer under non-time and time-constrained environment, IEEE, 2006.
33. Raya M., Hubaux J., Aad I., : DOMINO: A system to detect greedy behavior in IEEE 802.11 hotspots, MobiSYS, 2004.
34. Djahel S., Abdesselam F., Ahsan F., : Highlighting the effects of joint MAC layer misbehavior and virtual Link attack in Wireless Ad Hoc Networks.
35. Guang L., Assi C., : Vunlerabilities of Ad Hoc network routing protocols to MAC misbehavior, IEEE, 2005.
36. Guang L. and Assi C., "Mitigating Smart Selfish MAC layer Misbehavior in Ad Hoc Networks",IEEE, 2006.
37. Toledo A., Wang X., : Robust Detection of MAC layer Denial-of-Service attacks in CSMA/CA wireless networks, IEEE Transactions on Information Forensics and Security, Vol.3 , No. 3, September 2008.
38. Radosavac S., Cardenas A.A., Baras J.S., "Detecting IEEE 802.11 MAC layer misbehavior in ad hoc networks: Robust strategies against individual and colluding attackers", Journal of Computer Security 15, pp. 103-128, 2007.
39. IEEE 802.11 Standard,: Part 11 – Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, 2007.
40. Bansal R., Tiwari S., Bansal D.,: Non Cryptographic methods of mac spoofing detection in wireless LAN, IEEE, 2008.
41. Al-Hemairy M., Amin S., Trabelsi Z.,: Towards more sophisticated ARP spoofing

detection/prevention systems in LAN networks, IEEE, 2009.
42. Venkatarama A., Corbett C., Beyah R., "A wired-side approach to MAC misbehavior detection", IEEE Communications Society, 2010.
43. Tavil B., Heinzelman W., "Mobile Ad Hoc Networks – Energy-efficient real-time data communications ", Chapter 1.1 Characteristics of MANETs, Springer, 2006.
44. Djenouri D., Khelladi L., Badache N., "A survey of security issues in mobile ad hoc and sensor networks" , IEEE Communications Surveys, Volume 7, No. 4, 2005.
45. Lobzhanidze A., "Wireless ad hoc networks and its vulnerabilities", HAKIN9, Issue 6, No. 6, 2011.
46. Yang H., Luo H., Ye F., Lu S. and Zhang L., "Security in mobile ad hoc networks: challenges and solutions", IEEE Wireless Communications, pp. 38-47, February 2004.
47. S. Radosavac, J. Baras and I. Koutsopoulos, "A framework for MAC protocol misbehavior detection in wireless networks", Proceedings of ACM workshop on Wireless Security,pp. 33-42, 2005.
48. Djamel Djenouri, Lyes Khelladi, "A survey of security issues in mobile ad hoc and sensor networks", IEEE Communications Surveys, Fourth Quarter 2005, Volume 7, No. 4.
49. Carol Chyau, Jean-Francois Raymond, "What works: First mile solutions' daknet takes rural communities online", World Resources Institute, October 2005.
50. Andreas Festag, Holger Fubler, Hannes Hartenstein, Amardeo Sarma, Ralf Schmitz, "FleetNet: Bringing car-to-car communication into the real world", Proceedings of the 11th World Congress on ITS, Nagoya, Japan, October 2004.
51. Alan Mainwaring, Joseph Polastre, Robert Szewczyk, David Culler, John Anderson, "Wireless Sensor Networks for Habitat Monitoring", WSNA'02, Atlanta September 2002.
52. David Malan, Thaddeus Fulford-Jones, Matt Welsh, Steve Moulton, "Codeblue: An Ad Hoc Sensor Network Infrastructure for Emergency medical care", MobiSys 2004 Workshop on Applications of Mobile Embedded Systems (WAMES 2004), June, 2004.
53. C.Perkins, E.Rover, S.Das, RFC 3561 Ad hoc On-demand Distance Vector (AODV) routing, 2003.

Bibliography

1. Mathew Gast, "802.11 Wireless Networks The definitive guide", O'Reilly, April 2005.
2. Praphul Chandra, "Bulletproof Wireless Security – GSM, UMTS, 802.11 and Ad Hoc Secirty", Elsevier, 2005.
3. Carlos de Morais Cordeiro, Dharma Prakash Agrawal, "Ad Hoc & Sensor Networks Theory and Applications ", World Scientific Publishing, 2006.
4. Stefano Basagni, Marco Conti, Silvia Giordano, Ivan Stojmenovic, "Mobile Ad Hoc Networking", IEEE Press, 2004.
5. Bulent Tavil, Wendi Heinzelman, "Mobile Ad Hoc Networks – Energy Efficient Real-Time Data Communications", Springer, 2006.
6. Aftab Ahmad, "Wireless and Mobile Data Networks", Willey-Interscience, 2005.
7. Shih-Lin Wu, Yu-Chee Tseng, "Wireless Ad Hoc Networking: Personal area, Local area, and the sensory-area networks", Auerbach Publications, 2007.