Attacks and Defenses Utilizing Cross-Layer Interactions in MANET

John S. Baras and Svetlana Radosavac
Electrical and Computer Engineering Department
and the Institute for Systems Research
University of Maryland College Park
{baras,svetlana}@isr.umd.edu

Cross-layer protocol design is one of the prevailing methodologies that have recently been adopted in networking research and leads to significant performance benefits. In this study, we assess the performance of cross-layer interaction and investigate its effects with regard to security and information assurance of mobile ad hoc wireless networks. Using attacks in realistic wireless networks as a prototype, we find that natural cross-layer interactions between physical, MAC and network layer protocols in MANET can turn out to be a weak point, causing various attacks and intrusions. However, by allowing a controlled synergy between layers affected by attacks, we facilitate timely detection of such attacks that are otherwise difficult to detect and may have devastating effects on network functionality and operation.

1 Introduction

A MANET is a collection of wireless mobile nodes that are capable of communicating with each other without the use of network infrastructure or any centralized administration [2]. In addition to the wide range of attacks that are similar to the ones performed in wired networks, mobility, limited bandwidth and limited battery life present opportunities for launching novel attacks. A new class of attacks, cross-layer attacks, emerges from lack of interaction between MAC and routing layers. These attacks propagate from the MAC layer, where they are manifested as Denial of Service (DoS) attacks, to the routing layer, causing serious degradation of network performance in terms of the achieved throughput, latency and connectivity. An attacker can cause congestion in the network by either generating an excessive amount of traffic [1, 3] or by generating specific traffic patterns that prevent certain nodes from communicating with other nodes [7].

In this paper we describe and investigate the effects of cross-layer interaction (collaboration), or the lack of it, for the security and information assurance of mobile ad hoc wireless networks (MANET). We demonstrate, by describing attacks in realistic MANET, that natural interactions between physical layer and MAC, as well as MAC and routing protocols in MANET can lead to a variety of attacks and intrusions. We also demonstrate that without purposeful collaboration between the layers affected by such attacks, they are very difficult to detect while at the same can have catastrophic effects on the MANET functionality and operation. For the majority of the paper we focus on attacks involving interactions between the MAC and routing protocols. We also describe detection and defense mechanisms we have developed for such attacks. MAC and routing layers interact in numerous ways. Although previous research has not addressed malicious cross-layer behavior of nodes, it is obvious that cross-layer interaction can be abused by malicious nodes to mount DoS attack in the MAC layer and propagate it to the routing layer. Contention at the MAC layer causes a routing protocol to respond by initiating new route queries. The same holds from routing abuses to cause malfunctions in the MAC. DoS attacks are difficult to prevent and protect against. We describe several DoS attacks in realistic MANET that explicitly exploit cross-layer interactions. All attacks include both malicious and misbehaving nodes. We use the realistic scenario, where each node initially employs legal communication patterns that prevent other nodes from communicating and after some time they start misbehaving in order to maintain priority in the network. Through simulations and analysis we answers and quantify answers to questions such as the number of attackers needed to cause serious interruption in the network, level of violation of protocol parameters needed for successful detection, detection process, ways of cross-layer collaboration that may increase the speed of attack detection, existence of ”stealthy” attacks, etc.

2 MAC layer issues in wireless networks and cross-layer interaction

As the results of [1] show, MAC and routing layers interact in numerous ways. Although the authors don't address malicious behavior of nodes, it is obvious that cross-layer interaction can be abused by malicious nodes to mount DoS attack in the MAC layer and propagate it to the routing layer. Contention at the MAC layer causes a routing protocol to respond by initiating new route queries. The same holds vice versa. Specific routes chosen by the routing protocol can significantly affect the performance of the underlying MAC protocols. The decision on the new routes doesn't depend on the MAC layer. Routing layer often has a choice to include several equally good nodes in the route and the decision on which one of the nodes to use is made randomly. MAC layer then delivers the packet to the next hop along the chosen path. Therefore, some of the routes may contain nodes that are not enabled for immediate transmission since channel conditions can cause data transmission to fail (due to the congestion in the MAC layer some nodes that are included in the new paths may not be available for immediate transmission, causing delays). Eventually, the routing layer will try to use alternate routes for retransmission, causing wastage of bandwidth and delay. Wasting of bandwidth and time can also happen when the node chooses an alternative route that is in the interference range of the path that is being attacked. Non-existence of cooperation between MAC and routing enables the intruder not only to break the existing routes, but also to maximize the probability of including himself in the new routes by maximizing the number of nodes he is disabling while minimizing the probability of being detected. In [6] the authors address the problem of selfish nodes, but the same scenario can be used by malicious nodes as well.

MAC layer has mechanisms to protect itself from congestions, but these mechanisms can be abused by attackers and used to disrupt communication in the MAC layer. Namely, the basic mechanism of MAC layer exchanges a series of control signals before it sends the data. If the control signals at either sender or receiver side are not received within a certain period of time, the signal is retransmitted (an upper bound on the number of transmission exists). All communication is done at the MAC level and there are no signals that are passed to the higher levels except the final ACK signal that notifies the routing layer that the data has been successfully forwarded to the next hop. Communication in the MAC layer has several problems that can cause severe degradation of network performance, even without attacks. However, the failure of service at the MAC layer causes route disruption at the routing level. As we will see in the later sections, the attacker can use the MAC layer properties to disable and isolate several key nodes and partition the network. Therefore, an attack-resilient MAC protocol should have communication with both routing layer and Intrusion Detection System, meaning that routing should not be allowed to decide about routes without previously knowing conditions in the MAC and when MAC detects intruders, it will notify the routing layer and the IDS. MAC with the help of IDS should detect if the congestion is an attack and based on that decision the routing/MAC decide on future actions: to create new routes or discard the activity of the node that is causing congestion and pass that information to the other nodes. Another problem is that when alternate routes are chosen, they should not be in interference range of each other in order to minimize the probability of collision. As an example of damage that can be caused by interfering paths, we present simple communication of nodes that belong to 3 different paths. The whole scenario is represented in Fig. 1.

In this scenario node S_1 is communicating with R_1, causing R_2 to be silenced. Meanwhile, S_2 tries to establish communication with R_2 by sending RTS. It silences its neighborhood, including R_3, but it never receives a reply from R_2 (it has already been silenced by S_1's communication) and backs off. When S_3 tries

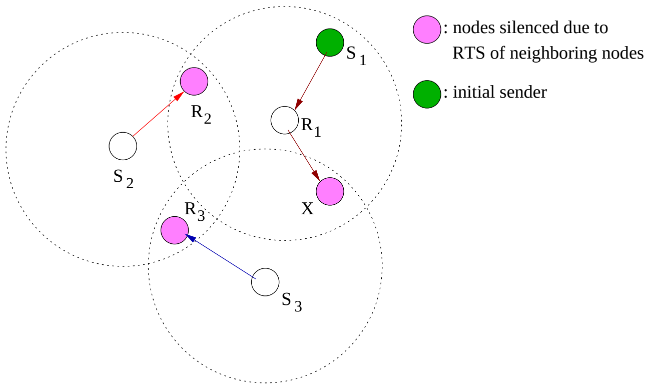

Figure 1: Example of interfering paths.

to communicate with R_3, it doesn't get any reply and backs off, just like S_2. Finally, when S_1 finishes its transmission to R_1 and R_1 tries to forward the packet to X, it doesn't get any reply because X was silenced by S_3's RTS. Therefore, R_1 enters its backoff. As we can see, MAC suffers from problems due to RTS/CTS silencing propagation even when the transmission is not successful and from interference problems that can eventually lead to route breaking. Another thing worth mentioning is that in case when S_2 is malicious, neither R_1 nor X can detect the attacker since it is not in their range. In that case, the time of attack detection increases and we may even isolate the wrong attacker. This implies that the system should monitor various parameters that are characteristic to MAC and routing protocols and based on their values make decisions about future actions. The general guidelines for parameters that can be exchanged between layers are given in [1]. However, that also raises another serious issue and that is how to distinguish an attack from congestion/interference.

It is obvious that several types of attacks can be performed in the MAC layer. First of all, an attacker can keep the channel busy so that regular nodes cannot use it for transmissions, which leads to DoS attack in that node. The nodes follow binary exponential backoff scheme that favors the last winner amongst the competing nodes. This leads to the capture effect where nodes that are heavily loaded tend to capture the channel by continuously transmitting data which makes lightly loaded neighbors to back off continuously.

Based on the previous analysis, we can distinguish three types of nodes:

1. ***Normal***
   This type of nodes obeys the rules of MAC layer protocols when both sending and receiving packets. This type of nodes will not behave selfishly and will reply to RTS requests from other nodes and will update their CW, NAV etc. according to the rules of the protocol.

2. ***Malicious***
   All communication is done following the MAC layer protocol. Nodes belonging to this group will employ legitimate communications which result in DoS in one or multiple nodes and attack propagation through the network.

3. ***Misbehaving***
   Nodes in this group misbehave in order to gain priority in the network or disrupt already existing routes. This group of nodes includes wide range of behavior: from malicious nodes that start misbehaving after a certain point in order to maintain the priority up to nodes that jam the network with large number of packets. Misbehaving nodes can change the value of CW, NAV value, Duration/ID field in the packet etc.

We now present the attacks represented in [7] and the results obtained. We then analyze the attacks from the point o view of our new IDS.

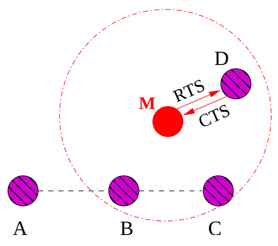

Figure 2: Attack scenario 1.

1. **Attack 1**
   Suppose that nodes A or D in figure 2 want to send data. Denote the malicious node as M. Node M captures the medium before node A/D decides to send data. Therefore, nodes A and D backoff for $T_{RTS/CTS}$. At the end of transmission node M will have to wait for t_{DIFS}+CW_{min} while A will wait fort_{DIFS}+CW, where $CW > CW_{min}$. When A tries to send a packet it will either sense the medium busy and stay in the same loop or it will eventually collide with RTS of node M. In this case its set of transitions is infinite loop. Node C, that wants to send a package through node D that is in the range of node M also cannot send any data. C sends a package to B, but B cannot receive any response from D because M has captured the medium. This attack addresses the unfairness of the 802.11 protocol since node that constantly fails to send data has worse chance to be enabled to send data as time passes. Hence, it is more likely that nodes with large CW that are backing off will be declared dead by other nodes than to get an opportunity to transmit.

   As a consequence, the throughput of the system is degraded. To be able to detect this kind of malicious behavior, cooperation of MAC and routing layers is required.

2. **Attack 2**
   By investigating traffic the attacker can find out which routes have higher priority. In the second step, mounting an attack from the MAC layer an attacker congests the channels and breaks multiple routes, increasing the possibility that in the new route search it is included in the new path. The network layer part of the attack could increase the probability of the node being included in the new path by false route advertisements or some other method that would increase the probability of node being included in the path in case multiple paths are left. In case of attack 1, the route $C \rightarrow B \rightarrow D \rightarrow E$ will be broken and the new route will be $C \rightarrow B \rightarrow A \rightarrow M$.

3. **Attack 3**
   We are observing a system that contains 2 malicious nodes. Those nodes are not directly cooperating are out of range of each other, but both are in the range of the attacked node D. The attack scenario can be performed as follows. Malicious node M_1 sends RTS to node A. RTS has information that the medium needs to be reserved for time t_1. At time t node D receives RTS from M_1 and defers its transmission for that period of time. Suppose that node M_2 needs to transfer data. It sends RTS to node B t_{DIFS} before the expiration of waiting period that was imposed by M_1's transmission. Node M_2 waits for t_{DIFS} and exactly at time when the first transmission stops this one starts and the medium is reserved. Since M_1, A and M_2, B are out of reach of each other but both can be heard by node D, they can continue their transmission infinitely many times unless additional fairness constraints aren't added. The described scenario is represented in figure 3.

At this point it is obvious that colluding malicious nodes are more difficult to detect. In cases when only one node is malicious the technique applied in [6] can be effective. However, in case of colluding nodes it is not possible to use the same strategy. In case when two colluding nodes are sending packets to each other

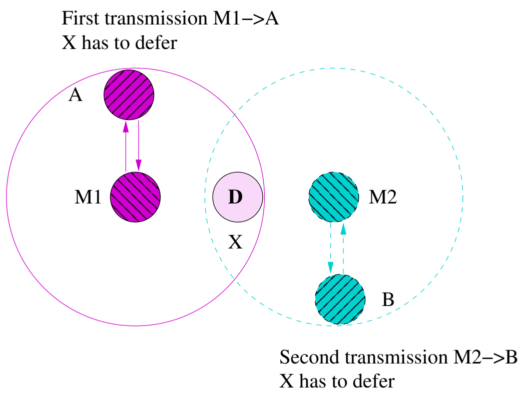

Figure 3: Attack scenario 3.

backoff needs to be observed by neighbors in order to be detected. That is achievable if at the beginning of each transmission sender is assigned a backoff and all one hop neighbors are noted about it. However, that can have devastating effects in case of normal nodes that are communicating and malicious nodes that are listening. The malicious nodes would then be notified of the exact backoff time of normal nodes and would be able to block further transmissions of normal nodes. The third attack presented in this section is even more difficult to detect since malicious nodes are not sending packets to each other and since they exchange in sending packets, they have enough time to back off without violating the protocol. In this case routing can help in detection. Routing can provide the information on whether nodes M_1 and M_2 are required to forward packets to the nodes they are trying to establish communication with.

3 MAC layer protocol representation

An Finite State Machine (FSM) is a five-tuple

$$(Q, \Sigma, \Delta, \sigma, q_0) \tag{1}$$

where Q denotes a finite set of symbols denoting states, Σ is a set of symbols denoting the possible inputs, Δ is a set of symbols denoting the possible outputs, σ is a transition function mapping $Q \times \Sigma$ to $Q \times \Delta$ and $q_0 \in Q$ is the initial state. In one reaction, an FSM maps a current state $p \in Q$ and an input symbol $a \in \Sigma$ to a next state $q \in Q$ and an output symbol $b \in \Delta$, where $\sigma(p,a) = (q,b)$. Given a sequence of symbols from Σ as input and an initial state, a sequence of reactions will produce a sequence of symbols from the output alphabet Δ. An FSM is *deterministic* if from any state there exists at most one enabled transition for each input symbol. An FSM is *reactive* if from any state there exists at least one enabled transition for each input symbol.

An Extended Finite State Machine (EFSM) is a finite automaton in which each transition is labeled by the name of a function (or a relation) defined on a set of typed input, output and internal variables. It is defined as a 5-tuple (S, I, O, D, T), where S is a set of states, I is a set of inputs, O is a set of outputs, D is an n-dimensional linear space and T is a transition relation, $T : S \times D \times I \rightarrow S \times D \times O$

Communicating Extended Finite State Machine (CEFSM) is 6-tuple: $CEFSM = (S, s_0, E, f, O, V)$, where S is a set of states, s_0 is an initial state, E is a set of events, f is a state transition function, O is a set of output signals and V is a set of variables. The function f returns a next state, a set of output signals and action list for each combination of a current state and an input event. A CEFSM can have predicates to control the behavior of the CEFSM so that some similar states can be grouped to reduce the total number of states. Upon receiving an event, the machine checks a predicate that is composed of variables, logical operators such as *AND*, *OR* and comparison operators such as *equal to*, *less than* and *greater than*. If the predicate is true, the entity performs actions and produces output signals if it has some information to transfer to the outside entities. The predicate is a pre-condition for the function execution.

The IEEE 802.11 DCF protocol specifies a Distributed Coordination Function (DCF) which is based on the same RTS/CTS message exchange as in MACA/MACAW. Unlike in MACA/MACAW, a node in IEEE 802.11 DCF defers only until the end of CTS frame reception. This solves both the hidden and exposed node problem. The only points where it differs from MACA are in the avoidance of collisions before transmitting RTS and its requirement of ACK transmission by the receiver after the successful reception of the data packet. The scheme follows the exponential backoff algorithm.

MAC protocols are easier to manage and represent than routing protocols. The nature of MAC protocol interactions, where event ordering and correct timing have crucial roles impose the necessity of using ordered models of execution with explicit timings. Explicit timing needs to be introduced in the model of event ordering due to the nature of event interactions in the MAC protocol (for example to describe timeouts). In this work we represent IEEE 802.11 protocol in the form of EFSMs.

Following the approach taken in [4] and modelling of PCF protocol in [9] it is straightforward to represent 802.11 MAC layer protocol using EFSMs.

Transmissions in 802.11 MAC layer are separated by inter packet gaps known as Inter Frame Spaces (IFS). Channel access is granted based on different priority access. The DIFS (DCF IFS) is used by STAs operating under the DCF for frame transmission. A station using the DCF shall be allowed to transmit if it determines that the medium is idle after a correctly received frame, and its backoff time has expired. It has the lowest priority. SIFS is the shortest of the interframe spaces. It is used when the stations have seized the medium and need to keep it for the duration of the frame exchange sequence. Using the smallest gap between transmissions prevents other stations, which are required to wait for the medium to be idle for a longer gap, from attempting to use the medium. This gives priority to completion of the frame exchange sequence in progress. Obviously, several timers need to be introduced in order to specify the exchange of messages between nodes i and j. We introduce:

1. T$_{DIFS}$ - DIFS timer
2. T$_{B}$ - backoff timer
3. T$_{SIFS}$ - SIFS timer
4. T$_{OUT}$ - set to a predetermined value when a node is waiting for a reply. If the reply doesn't arrive during the specified period, timer is set into time out mode (it has expired) and the node makes transition into corresponding error state (or initial state).
5. T$_{RTS/CTS}$ - set to a value that is defined in RTS/CTS message that the node overhears. This timer is activated when node makes a transition from state 0 to state 0' in figure 4.

All timers can be either active or inactive. Additionally, the timer can be expired (it's value has reached 0). EFSM representation of the node that is sending data is represented in figure 4.

In order to send data the node first needs to detect whether the channel is free or busy. In case it is busy, the node sets its idle time according to values given in NAV vector of the sender. It waits for that period of time and checks the status of channel. It either stays in state $0'$ in case when the channel is busy again or returns to state 0 where it waits for T_{DIFS}. If the channel stays idle for T_{DIFS}, node i sets its backoff timer T_B. In case when the node didn't have to defer transmission T_B is randomly chosen from the interval $[0, CW_{min}]$. Otherwise, CW is incremented every time the node defers and T_B is chosen from $[0, CW_{NEW}]$. In case when the channel becomes busy while T_B is being decremented, T_B is suspended and the process is resumed after the channel is sensed free. When T_B reaches zero, node i sends RTS to node j and other nodes defer according to $NAV(RTS_i)$. Node i then sets T_{OUT} and waits for CTS from node j. In case when CTS_j doesn't arrive, node

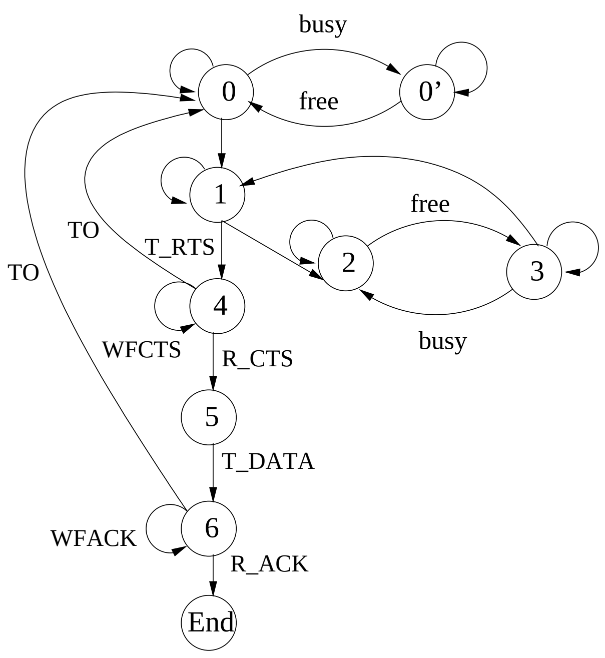

Figure 4: FSM representation of MAC protocol.

returns to state 0, increasing CW according to exponential backoff algorithm. Otherwise, when it receives CTS, it waits for T_{SIFS} and sends data. Again, T_{OUT} is set and if the acknowledgement doesn't arrive until T_{OUT} reaches zero, the node returns to state 0 and increases CW. Otherwise, it sets CW to CW_{MIN} and returns to idle state. The set of all transitions, predicates and events for sending node is represented in Table 1.

Where T_B=Random()*aSlotTime. Random()$\in$[0,CW] and aSlotTime=Transmission turn-on delay + medium propagation delay + medium busy detect response time. CW is Contention Window and it changes according to exponential back-off procedure. Finite State Machine representation of node j (receiver) is represented in figure 5.

If node j receives RTS$_{ij}$, it waits for T$_{SIFS}$ and transmits CTS and waits for data from node i. If the timeout timer T_{OUT} reaches 0, node j returns to the initial state. Otherwise, upon receiving data it makes transition to state 3, sends ACK to node i and returns to state 0 after waiting for T_{SIFS}. The set of all predicates, transitions and events for receiving node is represented in Table 2.

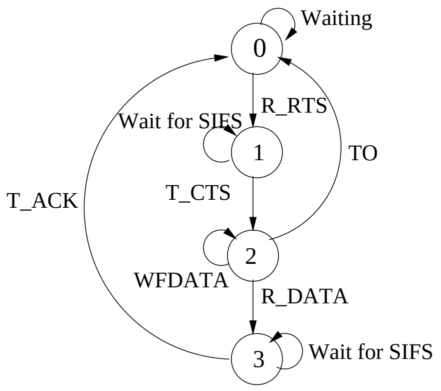

Figure 5: FSM representation of the receiving node in 802.11 MAC.

Predicate	Transition	Event
free=0	$0 \to 0'$	Set idle time for node i according to NAV vector of the sending node
free=0	$0 \to 0'$	wait until free
free=1	$0' \to 0$	set T_{DIFS}
free=1 $\wedge$ $T_{DIFS} \neq 0$	$0 \to 0$	$T_{DIFS} = T_{DIFS} - ASlotTime$
$T_{DIFS} = 0 \wedge free = 1$	$0 \to 1$	set $CW \wedge$ set T_B
free=1 $\wedge T_B \neq 0$	$1 \to 1$	$T_B = T_B - aSlotTime$
free=0 $\wedge T_B \neq 0$	$1 \to 2$	Suspend T_B
free=0	$2 \to 2$	Wait until free
free=1	$2 \to 3$	set T_{DIFS}
free=1 $\wedge T_{DIFS} \neq 0$	$3 \to 3$	$T_{DIFS} = T_{DIFS} - aSlotTime$
free=0	$3 \to 2$	wait until free
free=1 $\wedge T_{DIFS=0}$	$3 \to 1$	resume decrementing T_B and stay in 1 until T_B reaches 0
free=1 $\wedge T_B = 0 \wedge Data = 1$	$1 \to 4$	$RTS_{ij} = 1$, set $NAV(RTS) \wedge DA = j \wedge$ set $T_{OUT} \wedge$ activate T_{OUT}
free=1 $\wedge T_{OUT} \neq 0$	$4 \to 4$	$T_{OUT} = T_{OUT} - aSlotTime$
(free=1 $\wedge T_{OUT} = 0$) $\vee$ free=0	$4 \to 0$	set $CW_{NEW} \wedge T_{B_{NEW}} \in [0, CW_{NEW}]$
CTS=1 $\wedge$ DA=i	$4 \to 5$	set T_{SIFS}, activate T_{SIFS}
DA=j $\wedge$ Data=1 $\wedge T_{SIFS} = 0$	$5 \to 6$	Send data to j $\wedge$ set $T_{OUT} \wedge$ activate T_{OUT}
$T_{OUT} \neq 0 \wedge$ free=1	$6 \to 6$	$T_{OUT} = T_{OUT} - aSlotTime$
($T_{OUT} = 0 \wedge$ free=1) $\vee$ free=0	$6 \to 0$	set $CW_{NEW} \wedge T_{B'} \in [0, CW_{NEW}]$
ACK=1	$6 \to End$	CW=CW$_{MIN}$ $\wedge$ set i Idle

Table 1: Specification of node i (sender)

Predicate	Transition	Event
RTS=1	$0 \to 1$	Activate T$_{SIFS}$
$T_{SIFS} \neq 0$	$1 \to 1$	$T_{SIFS} = T_{SIFS} - aSlotTime$
T$_{SIFS}$=0	$1 \to 2$	CTS=1, set NAV(CTS), set T_{OUT}
$T_{OUT} \neq 0$	$2 \to 2$	$T_{OUT} = T_{OUT} - aSlotTime$
$T_{OUT} = 0 \wedge Data \neq 1$	$2 \to 0$	return to initial state
Data=1	$2 \to 3$	Activate T$_{SIFS}$
$T_{SIFS} \neq 0$	$3 \to 3$	$T_{SIFS} = T_{SIFS} - aSlotTime$
T$_{SIFS}$=0	$3 \to 0$	send $ACK \wedge$ set j to idle

Table 2: Specification of node j (receiving node)

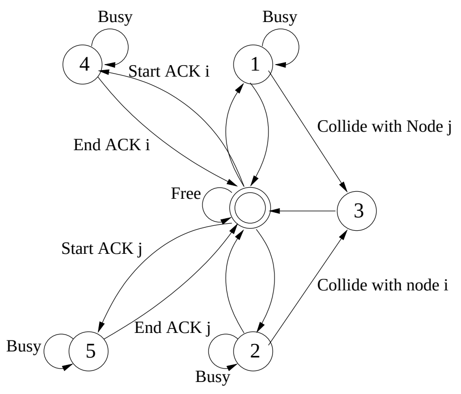

Figure 6: FSM representation of channel in 802.11 MAC.

The channel model is relatively simple and is presented in figure 6.

To complete the formal specification of EFSM we need a set of inputs, set of outputs and the initial state, which are obvious from the previously described model.

4 Suggested cross-layer cooperation scheme

As we have seen in previous sections, it is not meaningful to speak about neither MAC nor routing protocol in isolation. MAC layer protocols significantly influence routing protocols and vice versa. However, we have already mentioned that current interaction between MAC and routing protocols is limited to the exchange of ACK signals when the data is already sent. In order to mitigate the effects of congestion we need to design new dynamically adaptive protocols that can adapt to changing network and traffic characteristics by measuring and exchanging parameters that characterize cross-layer interaction and providing alternate routes with less traffic. However, in case of attacks that start in either MAC or routing layer, providing alternate routes may represent an opportunity for the attacker to include himself in the new routes. Hence, when incorporating cross-layer interaction we need to include interaction with an Intrusion Detection System. In case when IDS relies only on measuring traffic rates the number of false alarms rapidly increases. This implies the necessity of introducing more complex system that would observe both traffic rates and several other protocol-related parameters, such as CW, NAV, injection rate, etc. and impose timing constraints. MAC and routing layers would have to cooperate with each other in order to avoid points of congestion and reroute traffic and with the IDS in order to avoid inclusion of malicious nodes in the new routes or to isolate malicious nodes and propagate the information throughout the network. In this section we only refer to general cross-layer interaction scheme and we present the MAC layer detection scheme in Sec. 5. The proposed scheme for cross-layer interaction is presented in Fig. 7.

As we have stated in Sec. 2, routing decides on new routes independently of the conditions in the MAC layer, which can result in choosing routes with high interference or in the areas with high volume of traffic, which results in higher probability of collision and, therefore, in higher probability of multiple transmission retries and failure. Therefore, the routing layer should not decide on the final routes by itself. It initially chooses a subset of possible next hops, denoted as $r_1, \ldots, r_i$ in the route and forwards that information to the MAC layer. As we have already mentioned, one of the problems that arise in routing protocols is that most of "backup" routes are in vicinity of each other and, therefore, in interference range of each other. That can cause severe delays and breaking of routes since a node is allowed 7 RTS retransmissions until the packet is dropped. In this approach we refer to [8]. The conflict graphs or interference graphs can be derived from graph connectivity model using the protocol interference model. The nodes of the conflict graph represent

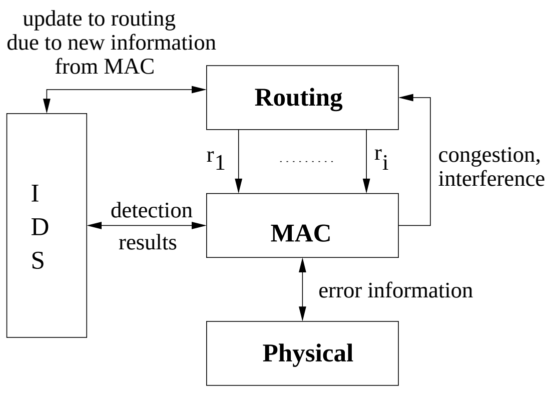

Figure 7: Cross-layer interaction.

links in the connectivity graph. A pair of nodes, l_{ij} and l_{kl} in the conflict graph are connected by an edge if they cannot have simultaneous transmissions according to the protocol interference model. In our case, a node assumes that all of the links that begin or end at it are in conflict with all links that end or begin at an interfering neighbor. We are interested in finding the number of *independent* conflict graph sets, which in our case represents the minimal number of attackers needed to partition the network. In general, the minimal number of attackers needed to partition the network is equal to the number of vertex disjoint paths from source to destination. However, due to interference that number is lower and can be determined by using conflict graphs. Finding all independent conflict graph sets is an NP complete problem, just like finding the number of vertex disjoint paths. We can also say that the minimal number of detectors needed for attack detection is equal to the number of independent conflict graph sets. Therefore, one strategy that would improve the performance of both MAC and routing would be that an alternate route is always chosen from another conflict graph set that is independent of the one the current node belongs to. MAC uses its own information that contains information about congestion and interference. It also contains a detection mechanism for misbehaving/malicious nodes and the results of the detection process influence the route selection as well. MAC forwards the intrusion detection information to *global IDS* which makes the final decision in case MAC is not able to do it with the current information. MAC also interacts with the physical level to determine the quality of suggested paths and choose the best next hop. One simplest way of interaction with the physical layer is to exchange the information about error correction since that is a measure of link quality. Interaction with IDS is extremely important due to the fact that the attackers goal is to include itself in the newly created routes. The attacker may also know the structure of the network and create its own attack graphs and make predictions of paths MAC could choose. MAC then forwards the final path selection to the routing layer. Therefore, using information from the MAC layer would minimize the appearance of failures caused by interfering routes, represented in Fig. 1.

MAC layer can also help in detection of certain routing attacks. For example, if a node in the routing layer claims it has the shortest route to the destination, while in reality it is trying to tunnel the packet, the MAC layer can detect the attack if it looks at the interference graph.

5 Attack detection

As we have already mentioned in Sec. 2, IEEE 802.11 MAC has several design problems that arise when the traffic level increases. Therefore, distinguishing normal from abnormal behavior in conditions of increased traffic represents a serious problem since our goal is to minimize the number of false alarms while maximizing the probability of detection. Most of MAC protocols assume by default that all nodes will respect rules of the

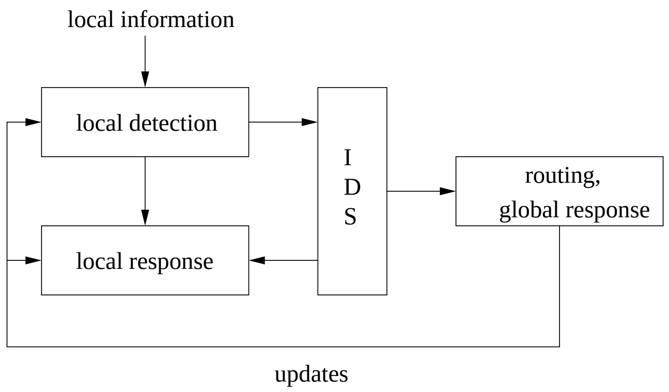

Figure 8: Detection mechanism in the MAC layer.

protocol. However, a user can easily modify several parameters of the protocol, gaining significant advantage. In [7] we classified nodes in three groups: *normal*, *misbehaving* and *malicious*. We are interested in detecting misbehaving nodes, more specifically, we are interested in creating a set of detection rules that maximize P_D while minimizing the probability of false alarm. We are also interested in creating a set of rules for detecting malicious nodes. Since the main goal of malicious nodes is to disable other nodes from communication, we are interested in their behavior in the long run. In particular, the malicious node will eventually collide with a real node and it will have to make a decision whether to violate the backoff algorithm and continue being malicious or to obey the assigned backoff. In the first case, the node will gain short-term advantage, but it will significantly raise the probability of detection. In the second case, the node will avoid detection, but the question is whether it will gain enough advantage (for example, larger throughput). Therefore, a malicious node has to make a tradeoff between maximizing its gain and minimizing P_D. Obviously, the length of malicious behavior of node M depends on the level of traffic in the neighborhood. If the number of neighbors is larger, the probability of node M being in interference range rises and, therefore, the probability of collision with a transmission of another node rises. In general, the malicious node is aware of the fact that the decisions he makes cause different backoff values and different P_D (they are inversely proportional). Hence, it has different preferences for different outcomes. Another issue is a network with multiple malicious nodes. If all the nodes are acting independently, this becomes a Prisoner's dilemma problem. Obviously, if all malicious nodes play the game in which they try to maximize their own gain, nobody will gain any advantage.

The MAC layer detection scheme is presented in Fig. 8. The input of the system is local information, such as the observed backoff window, Although we can also observe the value of NAV, whether the backoff counter stays idle while it senses the channel busy etc. That information is passed to the local detection system that is based on Neyman-Pearson detection. The system then decides whether the detected activity is anomalous or not with probability p'. If the result is below a certain threshold, the information is passed to the IDS. Otherwise a local response is issued. IDS cooperates with the other nodes in the network, with the routing layer and has a certain history of network behavior. It makes a final decision and forwards the result to the routing layer and also initiates a global response which then gives feedback to all nodes in the network. As mentioned, IDS updates routing with the current state in the MAC layer, helping it to make the initial selection of routes that will be passed to the MAC layer as well as the set of alternate routes that are used in case of route failure.

We refer to [5] for detailed description of IEEE 802.11 DCF protocol in normal mode (when all nodes obey rules of the protocol). We are interested in detection of misbehaving nodes. We first use a channel mode based on properties of 802.11. As we know from [5], it enables receiving of signals with low power in order to enable sensing of far away transmissions. Therefore, the receiver will send all transmissions sensed by the sender, but it also will sense transmissions not sensed by the sender. This results in the following expressions:

$$P(Receiver = busy|Sender = busy) = 1 \tag{2a}$$

and

$$P(Receiver = busy|Sender = idle) = p \tag{2b}$$

In order to be able to formulate a detection problem we assume that the receiver knows the backoff assigned to the sender. However, we face two problems in this formulation. The first one is that sender and receiver may not sense the channel in the same way and that may bring some delay in the detection process since Eq. (2b) states that we may still decide the sender is well behaved although the observed backoff at the receiver side is lower than the given threshold. Additional uncertainty is added with the fact that the sender will transmit in the interval $[0, 2^i W_0]$ with uniform probability. The problem becomes more difficult in the case of colluding nodes. Another solution would be that the backoff is assigned using some pre-determined function. This would result in all nodes knowing the length of backoff, as in [6] and that would bring additional problems for normal nodes. We believe that this problem can be solved using formal models. Therefore, if we set a threshold η and the total number of backoff slots is $2^i W_0$, the node will transmit in any of the slots in $[0, \eta)$ with probability $\frac{1}{2^i W_0}$. Therefore, we need to make a tradeoff regarding the threshold value. If it is set too low, we will miss attacks, but if it is set too high we will have a large number of false alarms. In our approach we use Neyman-Pearson criterion. We formulate two hypotheses:

- H_0: Sender is normal
- H_1: Sender is malicious

We define log-likelihood ratio as

$$L = \frac{P_{H_1}}{P_{H_0}} \underset{H_0}{\overset{H_1}{\gtrless}} \eta \tag{3}$$

where η is the threshold. Now we define B_r as the observed number of idle slots (backoff) at the receiver side, B_s as the number of idle (backoff) slots and B_t as the threshold. We are interested in observing two cases:

1. $B_r \geq B_t$
   In this case backoff at the receiver side is greater than the actual backoff. We are interested for conditions to decide on H_1 and H_0. Given the 802.11 model, we know that when $B_r \geq B_t$, $B_s > B_t$ with probability 1. Consequently, when $B_r \geq B_t$, $B_s < B_t$ with probability 0:

   $$P_{H_0} = 1, B_r \geq B_t \tag{4a}$$

   and

   $$P_{H_1} = 0, B_r \geq B_t \tag{4b}$$

2. $B_r < B_t$ In this case we will use Eq. 2b and the fact that the assigned backoff is M. Then we have the following equations

   $$P_{H_0} = P(B_s > B_t|B_r < B_t) = P(\text{making more than } B_t - B_r \text{ errors}) \tag{5a}$$

   and

   $$P_{H_1} = P(B_s < B_t|B_r < B_t) = P(\text{making } [0, B_t - B_r) \text{ errors}) \tag{5b}$$

Therefore, we can write the following expressions for the two hypotheses:

$$P_{H_0} = \begin{cases} 1 & \text{if } B_r \geq B_t, \\ P(\text{making } (B_t - B_r, M - B_r] \text{ errors}) & \text{if } B_r < B_t. \end{cases} \tag{6}$$

and

$$P_{H_1} = \begin{cases} 0 & \text{if } B_r \geq B_t, \\ P(\text{making } [0, B_t - B_r) \text{ errors}) & \text{if } B_r < B_t. \end{cases} \tag{7}$$

Therefore, the expression for log-likelihood ratio is

$$L = \frac{P_{H_1}}{P_{H_0}} = \begin{cases} 0 & \text{if } B_r \geq B_t, \\ \frac{P(\text{making } [0, B_t - B_r) \text{ errors})}{P(\text{making } (B_t - B_r, M - B_r] \text{ errors})} & \text{if } B_r < B_t. \end{cases} \tag{8}$$

Equation (8) for $B_r < B_t$ can be written as:

$$L = \frac{P(\text{making } [0, B_t - B_r) \text{ errors})}{P(\text{making } (B_t - B_r, M - B_r] \text{ errors})} = \frac{1}{P(\text{making } (B_t - B_r, M - B_r] \text{ errors})} - 1$$

$$P(\text{making } (B_t - B_r, M - B_r] \text{ errors}) = p^{B_r} \sum_{i=B_t+1}^{M} p^i$$

Therefore, Eq. (8) becomes:

$$L = \frac{1}{\sum_{i=B_t+1}^{M} p^{i-B_r}} - 1 \underset{H_0}{\overset{H_1}{\gtrless}} \eta \tag{9}$$

Therefore, we can see that NP criterion changes and we can compare B_r against $\eta' = f(\eta, B_t, M)$. Optimal threshold can be found and represents a tradeoff. Hence, the rule is:

$$p^{B_r} \underset{H_0}{\overset{H_1}{\gtrless}} \eta' \tag{10}$$

Now the decision rule is:

- Decide H_1 if $B_r < \eta'$
- Decide H_0 if $B_r > \eta'$
- Decide H_1 with probability γ if $B_r = \eta'$

Observing the Eq. (9), it is obvious that if B_r is increased, the number of errors is decreased (probability of correct, fastest detection increases). However, the log-likelihood ratio is decreasing with B_r increasing. When we increase B_r the probability of classifying the node as normal increases when the backoff is not fixed. We are concerned about the probability of false alarm since even normal nodes can transmit after a small number of idle slots. The probability that the node will transmit in a slot that is smaller than B_t is $\frac{B_t - 1}{2^i W_0}$. So, with raising B_t we are raising the probability of false alarm.

Now we discuss the scenario with distributed detection. Each node observes the backoff value and registers value R_{o_i} where $i = 1, \ldots, N$. Therefore, the vector of local observations is $\mathbf{B_o} = \{b_{o_1}, \ldots, b_{o_N}\}$. Each node makes decisions based on local observations and each node sends its log-likelihood ratio to the intrusion detection system. We denote the local decision vector as $\mathbf{u} = \{u_1, \ldots, u_N\}$ and the ID center arrives to global decision $u_0 = \gamma_0(\mathbf{u})$, where $u_0 = \{0, 1\}$, which corresponds to H_0 and H_1 respectively. Now we again formulate the NP rule for distributed detection as:

Definition 1 *NP rule for distributed detection*
For a predetermined probability of false alarm, $P_F = \alpha$, find optimum local and global decision rules $\Gamma = (\gamma_0, \gamma_1, \ldots, \gamma_N)$ that minimize the global probability of miss P_M.

Each node's observation and their decision processes are statistically independent and the log-likelihood ratio takes a simple form:

$$log(\Lambda(\mathbf{u})) = \sum_i \left(log \left(\frac{P(u_i|H_1)}{P(u_i|H_0)} \right) \right) \tag{11}$$

We denote global threshold as λ_0 and local thresholds as t_i, where $i = 1, \ldots, N$. Now we can represent local tests as:

$$\Lambda(b_{o_i}) = \frac{p(b_{o_i}|H_1)}{p(b_{o_i}|H_0)} = \begin{cases} > t_i & \text{then } u_i = 1 \\ = t_i & \text{then } u_i = 1 \text{ with probability } \varepsilon_i \\ < t_i & \text{then } u_i = 0. \end{cases} \tag{12}$$

Local thresholds t_i that enter the local tests represented in Eq. (12) need to be determined so as to maximize P_D for a given $P_F = \alpha$

The observation at the Intrusion Detection Center is $\mathbf{u}$ and according to the NP criterion, the optimal test is given by:

$$\Lambda(\mathbf{u}) = \frac{P(\mathbf{u}|H_1)}{P(\mathbf{u}|H_0)} = \begin{cases} > \lambda_0 & \text{then decide } H_1 \\ = \lambda_0 & \text{then decide } H_1 \text{ with probability } \varepsilon \\ < \lambda_0 & \text{then decide } H_0. \end{cases} \tag{13}$$

where threshold λ_0 and the randomization constant ε are chosen to achieve a desired $P_F = \alpha$.

Now we will consider a special case. First we observe the following equation:

$$\frac{p(u_i)|H_1}{p(u_i)|H_0} = \begin{cases} \frac{P_D^{(n)}}{P_F^{(n)}} & \text{if } u_i = 1 \\ \frac{1-P_D^{(n)}}{1-P_F^{(n)}} & \text{if } u_i = 0. \end{cases} \tag{14}$$

where $P_D^{(n)}$ and $P_F^{(n)}$ are the nth node's detection and false alarm probabilities. The simplest case is when all nodes have $P_D^{(n)} = P_D$ and $P_F^{(n)} = P_F$. If k denotes the number of nodes choosing H_1, which equals to $\sum_n u_n$, then the log-likelihood ratio becomes:

$$k \quad log \left(\frac{P_D}{P_F} \right) + (N-k) log \left(\frac{1-P_D}{1-P_F} \right) \underset{H_0}{\overset{H_1}{\gtrless}} log\eta \tag{15}$$
$$k \quad log \left(\frac{P_D(1-P_F)}{P_F(1-P_D)} \right) \underset{H_0}{\overset{H_1}{\gtrless}} log\eta + N log \left(\frac{1-P_F}{1-P_D} \right)$$

Since $P_D > P_F$, we know that $\frac{P_D(1-P_F)}{P_F(1-P_D)} > 1$. Hence, the optimal decision rule is:

$$k \underset{H_0}{\overset{H_1}{\gtrless}} \eta' \tag{16}$$

and $Pr(k) = \binom{N}{k} P^k (1-P)^{N-k}$. According to H_0 $P = P_F$ and to H_1, $P = P_D$.

.....Now talk about applying this to backoff.....then about malicious nodes and how to detect them (if possible).

6 Results

The experimental results do not incorporate any elements of cross-layer cooperation for now. We present the results of proposed attacks on IEEE 802.11 MAC. For better illustration of the above attacks, we have used OPNET to simulate the behavior of nodes under the attack.

The first scenario is presented in 2. We simulate network traffic with duration of 120 seconds. In the first half of the simulation, the malicious node is inactive and node *D* is able to send packets to its neighbor. After 70s, the malicious node starts attacking node *D*.

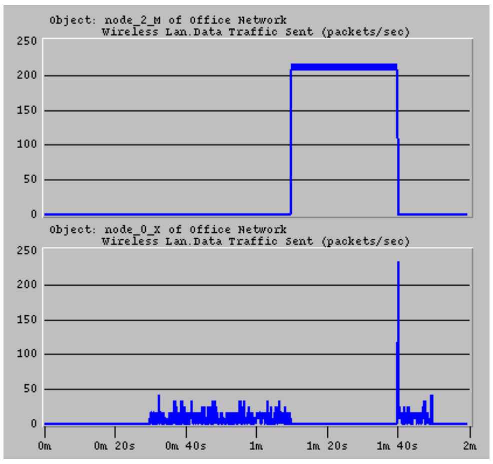

Figure 9: Data Traffic sent by nodes M and D.

The second scenario is presented in 3. It is important to realize that the malicious node does not need to send the traffic to node *D* in order to disrupt its traffic. The main intention of node *M* is to broadcast the RTS packet to node *D*, so it updates its Network Allocation Vector and doesn't send any traffic for the communication period indicated in the duration field of the RTS packet.

This attack scenario requires synchronization between two malicious nodes M_1 and M_2. The nodes need to alternate while sending traffic and therefore they need to generate packets at half the rate of the previous scenario. The major advantage of this attack is that it is more difficult to detect. Figure 10 shows the data traffic sent by the attacked node for 2 data traffic generation rates of the malicious nodes. In the first figure node *D* is still able to send its packets. However, when the attack is mounted, node *D* is completely disrupted during the 30s attacking period.

7 Conclusions and future work

In this work we have addressed several important issues regarding attack propagation and detection. We showed that cross-layer interaction can significantly improve the probability of attack detection as well as the speed of detection. We addressed the issue of malicious nodes and stealthy attacks and concluded that due to the mobility and randomness added by CW, stealthy attacks are not possible in the long term. However, that also depends on the level of traffic in the neighborhood of the malicious nodes. We also addressed the problems in the MAC layer such as RTS/CTS propagation and presented an example how that can be used for mounting an attack. We believe that the probability of detection can be increased by using formal models. More specifically, we refer to CTL theorems mentioned in [7] that can be used as an additional criterium in the IDS when we cannot guarantee with satisfying probability whether the attack is or isn't going on. In the next stage of our work we plan to extend theorems for violation of MAC protocol rules and examine parameters that need to be exchanged among MAC and routing layers. Given that event ordering and correct timing have crucial roles in MAC protocols we plan to introduce explicit timing constraints in our safety properties definitions. As in all model checking approaches, state space explosion represents a problem during transformation from a CTL formula to an EFSM. A potential approach that we are pursuing is a combination of model checking and theorem proving techniques. Regarding the latter we plan to use a combination of analytical techniques from graph theory, dynamic games, distributed detection, temporal logic, hybrid automata.

References

[1] C. Barrett, M. Drozda, A. Marathe, and M. V. Marathe, "Analyzing interaction between network protocols, topology and traffic in wireless radio networks," in *Proc. IEEE Wireless Communications and Networking Conference (WCNC'03)*, vol. 3, 2003, pp. 1760 – 1766.

[2] H. Deng, W. Li, and D. P. Agrawal, "Routing security in wireless ad hoc networks," *IEEE Communications Magazine*, vol. 40, pp. 70 – 75, October 2002.

[3] V. Gupta, S. Krishnamurthy, and M. Faloutsos, "Denial of service attacks at the MAC layer in wireless ad hoc networks," in *Proc IEEE MILCOM*, October 7-10, 2002.

[4] A. Helmy and S. Gupta, "STRESS: Systematic Testing of Protocol Robustness by Evaluation of Synthesized Scenarios."

[5] IEEE, "IEEE wireless LAN medium access control (MAC) and physical layer (PHY) specifications," 1999. [Online]. Available: http://standards.ieee.org/getieee802/

[6] P. Kyasanur and N. Vaidya, "Diagnosing and penalizing mac layer misbehavior in wireless networks," University of Illinois at Urbana - Champaign, Tech. Rep., December 2002.

[7] S. Radosavac and J. Baras, "Cross-layer attacks in wireless ad hoc networks," in *in Proceedings of CISS*, March 2004.

[8] J. M. Rajarshi Gupta and J. Walrand, "On-line capacity estimation for qos in ad-hoc networks."

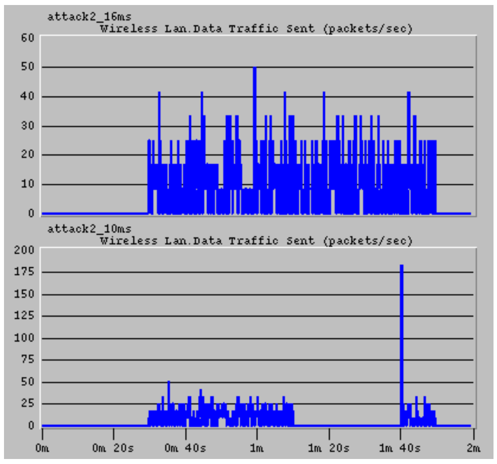

Figure 10: Data Traffic sent by node D.

[9] M. Youssef and R. Miller, “Analyzing the point coordination function of the ieee 802.11 wlan protocol using a systems of communicating machines specification,” University of Maryland, College Park, Tech. Rep. UMIACS-TR 2002-36 and CS-TR 4357, March 2002.