

Discovering Flaws in IDS through Analysis of their Inputs

Raphaël Jamet, Pascal Lafourcade

▶ **To cite this version:**

Raphaël Jamet, Pascal Lafourcade. Discovering Flaws in IDS through Analysis of their Inputs. The Sixth International Symposium on Foundations and Practice of Security FPS'2013, 2013, La Rochelle, France. hal-01707722

HAL Id: hal-01707722
https://hal.science/hal-01707722

Submitted on 13 Feb 2018

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L'archive ouverte pluridisciplinaire **HAL**, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d'enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.

Discovering Flaws in IDS through Analysis of their Inputs

Raphaël Jamet and Pascal Lafourcade

Université Grenoble 1, CNRS, VERIMAG, France
firstname.lastname@imag.fr

Abstract. To secure Wireless Ad-hoc Networks (WANET) against malicious behaviors, three components are needed: prevention, detection, and response. In this paper, we focus on Intrusion Detection Systems (IDS) for WANET. We classify the different inputs used by the decision process of these IDS, according to their level of cooperation, and the source of their data. We then propose a decision aid which allows automated discovery of attacks for IDS, according to the inputs used. Finally we apply our framework to discover weaknesses in two existing IDS.

Keywords: Wireless Ad-hoc Networks, Formal Methods, Intrusion Detection Systems, Security.

1 Introduction

Wireless Ad-hoc Networks (WANET) are networks built from several devices, which use multi-hops radio communications to form a mesh network. For instance, such networks can be used to provide low-cost internet in developing countries [1], or to spread messages and make calls in a smartphone-equipped group of people when the infrastructure is unavailable [14]. Wireless Sensor Networks (WSN) are a subset of these networks. They are built from a large quantity of inexpensive motes, which have little computing power and wireless communication abilities, in order to generate data covering a large area. These motes form an ad-hoc network, usually centered around a base station, also called a *sink*.

Due to their intrinsic properties and their applications, these networks are vulnerable to attackers. In order to guarantee the security of ad-hoc wireless networks, several complementary layers of protection are needed. First, *intrusion prevention methods* strive towards blocking eventual attacks. Since these systems are seldom infallible, *intrusion detection* should also be used, to ensure that an attacker bypassing the prevention layer will be noticed. The last level is the *intrusion response system*, whose goal is to mitigate the effect that detected attackers have on the network.

By consequence, intrusion detection systems (IDS) are critical for the security of WANET. Network-based IDS are often evaluated against a few attack scenarios based on specific intruder node behavior. We argue that such an analysis can be dangerous, as they do not specify how the intruder node became part of the network, what is the attacker achieving with that attack, nor what exactly the IDS is

trying to prevent. This lack of precision can lead to undiscovered flaws in the network. It is therefore important to formulate the properties that they try to achieve in a clear way, to define the intruder model, to model the protocols and to state the network assumptions, in a formal framework. Then, we can use formal methods, in order to sytematically find the flaws in an IDS. A similar process has been done in the last years in cryptographic protocol analysis [3], and our goal is to provide the first components to enable the use of such methods in the context of IDS.

Contributions: Our contribution is threefold: i) We survey different inputs that an IDS decision process can use. We base our analysis on two axis: the degree of cooperation, and what is being monitored. We also give examples of mechanisms from existing IDS to illustrate our classification. ii) We develop a formal model to evaluate such systems, based on *anomalies*, which are the results of attacker behavior. We propose some deduction rules (around 40) expressing how the combination of certain anomalies allows an attacker to build more complex attacks. These rules model the logical steps needed for constructing a specific attack. Then, depending on inputs used by an IDS, we determine whether an attacker can mount an attack without being detected. We also provide a prototype of our formal framework. iii) Using that prototype, we analyze two IDS from the literature, [9] and [21]. We show that it is not possible to fool the first IDS in presence of restricted intruders. However, by relaxing some of the hypothesis, we discover some weaknesses in the IDS. For the second IDS, we found undetected attacks using intruder nodes with directional antennas. Finally, we propose some modifications for these IDS in order to prevent these attacks. Overall, using our approach, it is easy to compare the inputs of an IDS to others from the literature, and to find its flaws and possible improvements.

Related work: Intrusion detection systems are usually classified with two main characteristics, which evolved from the seminal work in [10]. The first characteristic expresses what is being observed, and contains two categories: *host-based IDS* monitor their node only, while *network-based IDS* search for signs of malicious activity in the network. The second characteristic is based on the method used to detect intrusions. We do not consider this second aspect. Our classification can be seen as a refinement of the first category. Instead of having two broad categories, we separate the different inputs that an IDS uses and we determine the cooperation level and the sources of the data being monitored.

In [16], the authors provide a specification-based IDS for the AODV routing protocol, based on an extended finite state automaton for modeling the protocol. In their paper, attacks cause various *anomalous basic events*, which are defined as the segments of a routing process that do not follow the routing specification. These events are then classified in two categories: those that can be detected directly, and those that require statistical analysis. They also give a correspondence between

some classical attacker models and the related anomalous basic events. Finally, they propose an IDS which is built to detect all of the anomalous basic events they identified. Our work is different from theirs in two ways. First of all, instead of building an IDS specifically for AODV, we focus on the evaluation of any IDS for a wireless ad-hoc network. To achieve this, we built a model which is not protocol dependent and adjustable depending on the assumptions. The second important difference is our concept of anomaly. It is based on their anomalous events, but instead of isolating them, we add a model of their dependencies. This allows us to describe attacks taking into account the whole process, instead of just taking the end results of the attack into account.

Outline: In section 2, we provide our classification of IDS inputs and illustrate it through examples. In section 3, we formally define our model, which we apply on two existing IDS using our prototype in section 4.

2 Decision Process Inputs

IDS build their decision process over a multitude of *inputs* that we classify along two axis. The first one is made of three categories, that express the level of cooperation needed to use an input:

a. **Local** inputs are accessible by a node, without help from their neighbors.
b. Inputs requiring **k-neighborhood-wide cooperation**.
c. Inputs requiring **global cooperation**.

The other axis corresponds to how the IDS collect their inputs. We identified five categories. The first category is independent on the network, then the next categories depend on the protocols.

1. **Offline** inputs, which can be created even if the node is not part of a network.
2. **Topological** inputs, related to the positions of the nodes.
3. **Radio** inputs, linked to the medium access control protocol.
4. **Routing** inputs, related to the routing protocol directly or indirectly.
5. Inputs extracted from the application **data**, as opposed to all the previous categories which analyze how the nodes and the network behave.

Using the second axis, we now describe and illustrate with references each of those inputs according the level of involvement or cooperation needed to use an input, followed by a summary of this classification.

2.1 Offline inputs

Offline inputs do not depend on the network: the object of their monitoring is internal to a node.

a) The first family of offline inputs we identified are **local**: host-based IDS (with for instance [30]). This family of IDS are looking for a partial compromise of the node running the detection, for instance through viruses or vulnerable applications. b) To allow compromise detection by third-party nodes, we need **neighborhood cooperation**. In [31], the authors proposed a scheme where free memory in nodes is pre-loaded with random noise, the knowledge of which is shared among the node's neighbors. They make the hypothesis that a compromised node would have deleted some of that noise to include its rogue algorithms. Therefore, to detect whether a suspected node has been compromised, its neighbors can collaboratively query it to check the integrity of its random noise. c) Finally, **global** offline inputs would examine something independent of the network on a network-wide scale, which would not add anything significantly more useful than local or neighborhood-scale analysis, which explains the absence of such systems in the literature.

2.2 Inputs Based on the Network Topology

This category contains mechanisms that use distances, or neighborhoods.

a) First, this analysis can be **local** to a node. For instance, in the IDS described in [9], nodes have a list of verified neighbors, so that messages coming from unverified neighbors trigger alerts. In [11, 20, 21], nodes remember the strength of the signal received for the last transmissions from each neighbor. The signal strength is related to distance: if an intruder node impersonates some honest node, then the receiver may notice the change of received signal strength. In [24], instead of using previous measurements of signal strength to detect anomalies, the expected value is deduced from distance data.

b) The IDS described in [25] requires **neighborhood-wide cooperation** to measure distances between nodes in a static network, which allows to detect unexpected changes in localization that are characteristic of a node being manipulated. A similar neighborhood-wide effort can identify Sybil intruders (single nodes that use multiple identities) through signal strength, as described in [11]. By computing the ratios of the signal strength of different nodes, the authors show that it is feasible to triangulate the position of other nodes, allowing detection of nodes that share the same exact position who may be Sybil intruders.

c) Finally, such an analysis can be done from a **global** point of view. In [12], the authors argue that under some hypothesis on the underlying network, it is feasible to detect wormholes (two distant intruder nodes, linked with a special communication channel) using only topological data. In [7], the authors propose a mechanism to detect nodes sharing identities in the network. It is a global algorithm which reports a suspect node's identity and location to a specific third-party node, using the suspect identity. Then, if that third-party node receives several reports containing different locations for a single identity, an alert is raised.

2.3 Inputs from the Medium Access Control (MAC) Protocol

This category contains all the inputs which use data originating from the wireless transmission medium.

a) This category is well-suited to **local** analysis, but increasing the cooperation will not significantly improve these techniques. By looking at indicators in the MAC such as collision rate or the number of retransmissions requests, a node may be able to tell if there are degradations of the radio medium, whether they are environmental or caused by an attacker. In all of [4, 9, 22, 26], there is a part of the intrusion detection being done based on the collision rate. This mechanism is close to the definition of specification-based IDS, which detects whether a node respects the protocol they should run. Another approach based on the radio layer is to use characteristic features of the radio emitters to identify them. This technique is called *fingerprinting* [5, 15], and would allow detection of some Sybil attackers and impersonation-based attacks.

2.4 Data Based on the Routing Layer and Traffic

This category is based on the analysis of the way messages are routed through the network. Some of the IDS mechanisms we present here are tied to a specific routing protocol, which allows them to monitor the protocol compliance of suspected nodes at the cost of genericity. On the other hand, some of them are built upon generic properties that are common to most routing protocols, such as immediate packet retransmission or each node using only one public identity.

a) A **local** input common to several IDS is that the nodes monitor the variations in the volume of incoming traffic. Then, if there are significant differences when comparing to some reference measure, the node will raise an alert. We observe several variants such as separating traffic streams per message type [8], per neighbor [9], or depending on the messages source and destination [18]. Some IDS also monitor the intervals between reception of messages of different types [8]. Finally, instead of looking at the traffic flow, the authors of [4] suggest observing the data types which are expected on each route. Another category of local inputs require using promiscuous listening, so that a monitor node can observe its neighbor's behavior. By designing an IDS more specific to the routing protocol, it becomes possible to detect deviations from the routing protocol, using an analysis that can span from simple rules or features to a complete specification-based monitoring. Both [9] and [26] propose some IDS which uses promiscuous listening. The IDS in [27, 29] monitor the routing process of neighboring nodes using finite state machines, respectively built for AODV[1] and OLSR[2]. Another specification-based IDS built on top of an extended finite state machine for AODV is described in [16]. These examples are still local to a node.

[1] Advanced On-Demand Distance Vector, a routing protocol described in [23].
[2] Open Link-State Routing, another routing protocol [6].

b) By extending to a **neighborhood-wide** effort, specification-based IDS can be more efficient and accurate. For instance, in [28], the authors present DEMEM, a specification-based IDS with new messages overlayed on the routing protocol. These messages allow nodes to verify the claims of their neighbors, to detect more attacks on the routing protocol.

c) Finally, the **global** scale inputs can be illustrated with Lipad [2]. This IDS does traffic analysis in a centralized way, which allows to locate precisely where the anomaly occurred.

2.5 Inputs Based on the Application Data

The last category concerns all the inputs which use the application data, and assumptions about it.

a) A good illustration of **local** application data analysis can be found in [17]. This IDS is designed for general-purpose networks, and operates by statistical analysis of the application data being transmitted in the captured packets, with a different model for each application. b) Using **neighborhood** collaboration, the authors of [13] present an IDS based on analyzing the data delivered to the application using hidden Markov models. Such a data analysis allows them to detect altered data, depending on what is being monitored. This example is on a global scale, but the idea of application data modeling can also be applied with a lower cooperation level. For instance, in a network monitoring earthquakes, neighborhood-scale cooperation makes sense to detect falsified data insertion. On the other hand, if the network measures are purely local (such as motion sensors in a building), global-scale analysis may not be useful. We only found [13] to illustrate such an analysis for WANET in the literature. We conjecture that this is because this technique is strongly tied to the application.

2.6 Summary

Our classification allows us to categorize the various inputs a network-based IDS uses in its decision process. In Figure 1, we recapitulate our classification, and we provide a list of the different inputs we identified from existing IDS, plus the categories in which they belong. Note that the same IDS can use different inputs, and will therefore be in several different categories. We also include a few mechanisms built to detect specific intruders such as [7, 11], as they are compatible with our definition of IDS input.

We denote by a 'X' the categories which do not bring any new significant information when compared to the lower cooperation levels. Since the MAC protocol is by essence local to a link, nodes have little interest in relying on the declarations of distant nodes to detect intruders. A similar situation appears for global offline inputs. We also denote with a '?' the categories of data analysis where we did not find any example, but we think this combination can be relevant to detect malicious nodes.

Data source	Offline	Topology	Radio	Routing	Data
Local	[30]	[9, 11, 20, 21, 24]	[4, 5, 9, 15, 22, 26]	[4, 8, 9, 16, 18] [19, 26, 27, 29]	[17]
Neighborhood	[31]	[11, 25]	X	[28]	[13]
Global	X	[7, 12]	X	[2]	?

Fig. 1. Classification of IDS data sources

3 Automating Vulnerabilities Discovery

We now introduce our model that finds whether an attacker can reach a certain goal without its actions being noticed by an IDS. This is done through modelization of the different steps necessary to mount an attack. We call those steps *anomalies*. We model the IDS using the different inputs to its detection process, which gives us a set of anomalies that could cause detection of the attack. If an attacker can reach its target, without using any of the steps monitored by the IDS, then this attack will be undetected. We start by giving some definitions then we present differents components of our model: facts, anomalies and our attack detection mechanism.

3.1 Definitions

An *identity* is the different items a node needs to join and operate legitimately in the network. For example, in a network where nodes only use their hardware identifier to identify themselves and where there are no other forms of security, the identity is composed of that identifier only (which is trivial to copy or create). Alternatively, the identity can be a combination of various things such as frequency hopping schedules, pre-shared cryptographic material, or a radio which has the right fingerprints.

An *associated* node is a node who is able to use an identity. If this node is an intruder node, the identity is said to be *compromised*. Finally, *valid* messages are messages whose data would be delivered to the application if they reach their destination. Moreover, the wireless network is composed of honest nodes. These nodes generate data messages, which are then routed to one or several destinations, in order to be delivered to the application. These nodes all possess an identity. Also, the attacker can deploy several *intruder nodes*, which are attacker-controlled nodes. These also have memory, computing power, and wireless communication capabilities using an omnidirectionnal antenna unless otherwise specified. To denote the logical transitions between the attack steps, we use inference rules and axioms.

Definition 1 (Axioms and Rules). *Given* $T_1, \ldots, T_n$*, a* rule (R) *concluding* C *is denoted by* $(R)\dfrac{T_1 \quad \ldots \quad T_n}{C}$*. An* axiom *is a rule without any conditions. Such an axiom named* (A) *concluding* C *is denoted by* $(A)\dfrac{}{C}$.

3.2 Facts and Relationships

We now describe formally our model, beginning with the notion of facts. A fact is an assumption about the network, the protocols used or the attacker.

Definition 2 (Facts). *Assumptions about the network, protocols and attackers are called facts. They are represented by keywords in italic. We denote the set of all facts* **F**.

We now detail all facts contained in **F**, ordered by category. A fact holds if the topology, attacker or protocol described has the associated property.
• The first category contains facts related to the attacker. *CompromisableNodes*: An attacker is able to take full control of legitimate nodes, and recover their identity and knowledge. *TxPowAdjust*: Intruder nodes are able to adjust their radio transmission power. *DirAntenna*: Intruder nodes are equipped with directionnal antennas. *CanImpersonate*: The attacker is able to impersonate honest nodes (but this does not assume anything about validity).
• The second category contains the facts related to the protocols in use in the network. *SimpleValidity*: An attacker can alter or create a message and keep it valid. *ValidityCheckedEachHop*: An attacker cannot alter a given message in a way that keeps it valid, and originating from an uncompromised node. Validity of a message is checked at each hop. *NoConfidentiality*: Any passive listener is able to recover the contents of messages. *HopConfidentiality*: Nodes outside of the message route cannot recover the contents of a message. *EndToEndConfidentiality*: Only the source and destination(s) of a message can recover its contents. *OpenNetwork*: Nodes have access to new identities at will. Thus, any node can associate, regardless of initial knowledge or pre-existing relationships.

Some of these facts are related. We define that a fact F_1 is more restrictive for an attacker than F_2 when any possible attack when F_1 holds is also possible when F_2 holds.

Definition 3 (Factual relationships $\mathcal{F}$). *If a fact $F_1 \in \mathbf{F}$ is more restrictive for an attacker than a fact $F_2 \in \mathbf{F}$, we express that relation using the following rule named (FR): $(FR)\dfrac{F_2}{F_1}$. We denote by $\mathcal{F}$ the set of rules (F-Conf1),(F-Conf2) and (F-VC1).*

We now describe the contents of $\mathcal{F}$. Having validity checks at each hop blocks any invalid message before it gets retransmitted by an honest node, which only limits what messages an attacker can usefully send. Therefore, the fact *ValidityChecksAtEachHop* is more restrictive for an attacker than *SimpleValidity*.

$$(F-VC1)\frac{SimpleValidity}{ValidityChecksAtEachHop}$$

Regarding the confidentiality-related facts, the relationship is similar. *HopConfidentiality* strictly restricts attacker knowledge when compared to *NoConfidentiality*, as it only prevents listeners outside of the route from being able to read the data. Therefore, *HopConfidentiality* is more restrictive for an attacker than *NoConfidentiality*.

$$(F - Conf1)\frac{NoConfidentiality}{HopConfidentiality}$$

If an attack is possible when considering the *EndToEndConfidentiality* fact, then removing the confidentiality aspect for intermediate nodes does not change that possibility, as it merely adds more possibilities for the intruder. Therefore, we say that *EndToEndConfidentiality* is more restrictive for an attacker than *HopConfidentiality*, and the rule, named (F-Conf2), is written as follows:

$$(F - Conf2)\frac{HopConfidentiality}{EndToEndConfidentiality}$$

The set of facts we want to use for the analysis is denoted $\mathbf{F}_I \subseteq \mathcal{F}$. Using this set of facts, we build a set of axioms, named the selected hypothesis.

Definition 4 (Selected hypothesis $Hyp(\mathcal{H}, \mathbf{F}_I)$). *Let $\mathcal{H}$ be the set of all possible axioms deducing a fact from $\mathbf{F}$, and let $\mathbf{F}_I$ be a subset of $\mathbf{F}$ we want to assume for the verification. The set of selected hypothesis is a set of axioms, denoted by $Hyp(\mathcal{H}, \mathbf{F}_I)$, and defined by:*

$$Hyp(\mathcal{H}, \mathbf{F}_I) = \left\{ (AF)\frac{}{F} \,\middle|\, F \in \mathbf{F}_I \right\}$$

3.3 Anomalies

Anomalies are components used to describe the different steps in an attack, which are linked together using rules. We first define them, then we present the contents of $\mathbf{A}$, separated in categories.

Definition 5 (Anomalies). *Anomalies are the results of the attacker's behavior. We denote the set of all anomalies $\mathbf{A}$.*

• The following anomalies are related to impersonation.

– *OmniImpersonation*: An intruder node transmits packets as if they were emitted by an honest neighbor, but the message can be received by any neighbor, and the received signal strength may differ from the one from legitimate transmissions.

- *DirImpersonation*: An intruder node transmits packets in a directed fashion, such that only the attacker and the receiver know the transmission happened. The signal strength of this transmission may however be different from what is expected from the impersonated node.
- *TxPowImpersonation*: An intruder node impersonates an honest node, while adjusting its transmission power so that the signal strength at the receiver corresponds to the signal strength which would be expected from the impersonated node.
- *DirTxPowImpersonation*: An intruder node transmits packets in a directed fashion, such that only the attacker and the receiver know the transmission happened. Furthermore, the intruder adjusted its transmission power so that the signal strength at the receiver corresponds to the signal strength which would be expected from the impersonated node.
- *Impersonation*: The attacker can communicate with any node, as if the communication happened from an honest neighbor. This anomaly is a generic version of the previous ones.

• The next category are anomalies related to node compromise and identities.

- *VirusCompromise*: The attacker compromises some part of a node, whose identity is now compromised. The attacker is also able to control this node.
- *TotalCompromise*: The attacker takes full control of a node, whose identity is now compromised.
- *AttackerAssociated*: The attacker uses intruder nodes which are associated.
- *RoutingMisbehavior*: Intruder nodes deviate from the routing protocol.

• Finally, the last category of anomalies are related to the application data.

- *ApplicationDataAltered*: The attacker alters the data which is delivered to the application, either by adding, altering or substracting data.
- *Snooping*: The attacker reads some of the application data going through the network.
- *Alteration*: The attacker alters data in messages.
- *ValidAlteration*: The attacker alters data in messages, while keeping them both valid and appearing to be from an emitter whose identity is uncompromised.
- *ImmediateAlteration*: An intruder node is able to alter the data in a message it received.
- *NeighborVisibleAlteration*: An intruder node alters the data in a message it received, in a way that can be overheard by neighbors.
- *NeighborVisibleValidAlteration*: An intruder node alters the data in a message it received, while keeping them both valid and appearing to be from an emitter whose identity is uncompromised, in a way that can be overheard by neighbors.

- *ImmediateValidAlteration*: An intruder node is able to alter the data in a message it received, while keeping it both valid and appearing to be from an emitter whose identity is uncompromised.
- *NeighborVisibleSuppression*: The attacker drops data messages, which prevents them from being delivered to their destinations. This action can be overheard by the node's neighbors.
- *Suppression*: The attacker drops data messages at some point in the network, which prevents them from being delivered to their destinations.
- *ImmediateSuppression*: An intruder node is able to drop valid data-bearing messages.
- *Insertion*: The attacker creates valid data messages.
- *NeighVisibleInsertion*: An intruder node creates valid data messages, which may be overheard by neighbors.
- *ImmediateInsertion*: An intruder node can create valid data messages.

Anomalies follow a logical progression, with certain anomalies and facts being prerequisites to other anomalies. To model these dependancies, we use rules.

We denote by $\mathcal{R}$ the set of all our rules. The set $\mathcal{R}$ contains 39 rules, given in Figure 2. Due to space constraints, their justifications are available in our technical report [**?**], Section 6. We now explain how those anomalies and rules are used to search for undetected attack.

3.4 Attack Detection

An IDS has a certain number of inputs, each of these noticing a certain number of anomalies. To know if an IDS is adequate to prevent an attacker from reaching a given goal, we need to check if there is a way to reach a target anomaly from accepted facts, in a way that do not use any of the anomalies covered by the set of this IDS's inputs. To model this, given an IDS, we remove all the rules going to anomalies detected by that IDS, leaving only anomalies which do not trigger detection.

Definition 6 (IDS). *We denote the anomalies monitored by an IDS by* $\mathbf{A}_I \subseteq \mathbf{A}$. *We define the set of rules* $IDS(\mathcal{R}, \mathbf{A}_I)$ *which is the allowed set of rules for the attacker given the base rules. This set is defined as:*

$$IDS(\mathcal{R}, \mathbf{A}_I) = \left\{ (R) \frac{T_0 \quad \dots \quad T_n}{A} \in \mathcal{R} \;\middle|\; A \notin \mathbf{A}_I \wedge \forall i, T_i \notin \mathbf{A}_I \right\}$$

To build $\mathbf{A}_I$, one should examine which are the inputs the IDS uses, and for each of them, which are the anomalies that may be detected by such an input. For instance, message addition or substraction can be detected by traffic analysis. Then, the set $IDS(\mathcal{R}, \mathbf{A}_I)$ is used to build the set of rules which will be used by the analysis.

Name	**Prerequisite(s)**	**Conclusion**
CompV	*CompromisableNodes*	*VirusCompromise*
CompT	*CompromisableNodes*	*TotalCompromise*
VAssoc	*VirusCompromise*	*AttackerAssociated*
TAssoc	*TotalCompromise*	*AttackerAssociated*
Open	*OpenNetwork*	*AttackerAssociated*
Misbehave	*AttackerAssociated*	*RoutingMisbehavior*
OmnI	*CanImpersonate*	*OmniImpersonation*
DirI	*DirAntenna* $\wedge$ *CanImpersonate*	*DirImpersonation*
PowI	*TxPowAdjust* $\wedge$ *CanImpersonate*	*TxPowImpersonation*
DirPowI	*TxPowAdjust* $\wedge$ *DirAntenna* $\wedge$ *CanImpersonate*	*DirTxPowImpersonation*
OtoI	*OmniImpersonation*	*Impersonation*
DtoI	*DirImpersonation*	*Impersonation*
TtoI	*TxPowImpersonation*	*Impersonation*
DTtoI	*DirTxPowImpersonation*	*Impersonation*
IStoS	*ImmediateSuppression*	*NeighVisibleSuppression*
DirIStoS	*ImmediateSuppression* $\wedge$ *DirAntenna*	*Suppression*
IIntoNVIn	*ImmediateInsertion*	*NeighVisibleInsertion*
AssoIns	*AttackerAssociated*	*ImmediateInsertion*
ImpInser	*Impersonation* $\wedge$ *SimpleValidity*	*ImmediateInsertion*
IAlt	*AttackerAssociated* $\wedge$ *RoutingMisbehavior*	*ImmediateAlteration*
VIAlt	*ImmediateAlteration* $\wedge$ *SimpleValidity*	*ImmediateValidAlteration*
VIaltIAlt	*ImmediateValidAlteration*	*ImmediateAlteration*
ValtAlt	*ValidAlteration*	*Alteration*
NVIaltAlt	*ImmediateAlteration*	*NeighVisibleAlteration*
NVIValtVAlt	*ImmediateValidAlteration*	*NeighVisibleValidAlteration*
NVIaltAlt	*ImmediateAlteration* $\wedge$ *DirAntenna*	*Alteration*
NVIValtVAlt	*ImmediateValidAlteration* $\wedge$ *DirAntenna*	*ValidAlteration*
AssoISup	*AttackerAssociated* $\wedge$ *RoutingMisbehavior*	*ImmediateSuppression*
InSupAlt	*ImmediateSuppression* $\wedge$ *ImmediateInsertion*	*ImmediateAlteration*
S	*NeighVisibleSuppression*	*Suppression*
DirIIntoIn	*ImmediateInsertion* $\wedge$ *DirAntenna*	*Insertion*
NVIntoIn	*NeighVisibleInsertion*	*Insertion*
IaltAlt	*NeighVisibleAlteration*	*Alteration*
IValtVAlt	*NeighVisibleValidAlteration*	*ValidAlteration*
SApp	*Suppression*	*ApplicationDataAltered*
AApp	*ValidAlteration*	*ApplicationDataAltered*
IApp	*Insertion*	*ApplicationDataAltered*
HopSnoop	*AttackerAssociated* $\wedge$ *HopConfidentiality*	*Snooping*
ConfSnoop	*NoConfidentiality*	*Snooping*

Fig. 2. List of all the rules

Definition 7 (Setting). *The set of rules obtained by the union of selected hypothesis, the rules allowed given a specific IDS, and factual relationships is called a setting. We denote it by :* $\mathcal{S} = IDS(\mathcal{R}, \mathbf{A}_I) \cup Hyp(\mathcal{H}, \mathbf{F}_I) \cup \mathcal{F}$

Once the setting is determined, we can search for undetected attacks, by looking at which anomalies are reachable using the rules.

Definition 8 (Undetected attack). *Let* $\mathcal{S}$ *be a setting describing an IDS together with assumptions about the network, protocol and attacker. Let* $G \in \mathcal{A}$ *be an anomaly. We say that there exists an attack resulting in* G *which can not be detected by the IDS described in* $\mathcal{S}$ *if* G *is reachable using* $\mathcal{S}$*.*

To summarize, in our model, an attack is a chain of anomalies, linked together by rules. Starting from facts, the attacker mounts his attack using only the rules in the setting (*i.e.* the rules that allow him to stay undetected). Each of those rules allow him to progress to further anomalies. Therefore, analyzing whether the attacker can reach a certain anomaly in a specific setting allows us to know whether an undetected attack is possible against that IDS. Also, we focus only on attacks which change the data to the application, impersonations, and node compromise. All considerations linked to the performances and availability of the network are not captured by our model, and constitute a natural future extension of this work.

We built a prototype which automatically goes through all the reachable anomalies, given an IDS and facts. It is available online, along with instructions on how to use it, at the following url: `http://www-verimag.imag.fr/~rjamet/IDS/`. The examples in the next section were analyzed using that tool, and the analysis took less than a second for each of them on a regular laptop.

4 Modeling Existing IDS

We now use the inputs and the intruder model previously described to evaluate two existing IDS, [21] and [9], and show the weaknesses and the possible improvements we discovered. All the outputs of the tool are available in our technical report [**?**] Section 7.

4.1 A Real-Time Node-Based Traffic Anomaly Detection Algorithm [21]

In [21], Ilker Onat and Ali Miri present an anomaly-based IDS based on two inputs, received signal strength, and packet arrival rates. They make several hypothesis: the routing protocol is based on a tree (such as GBR), nodes are static and can uniquely identify neighbors, all nodes use the same hardware and software, and all nodes use constant transmission power. Our model does not take into account movement of nodes, and assumes that neighbors can be identified. Thus, the only assumption we need to transpose is the constant transmission power. This is modeled by not adding *TxPowAdjust* to the hypothesis set.

We now build the set $\mathbf{A}_I$ of anomalies the IDS can detect. The first input used by this IDS is a packet arrival rate analysis. This input is able to discover any attacker that stops or inserts more messages than expected from an honest node. The corresponding anomalies in our model are *Suppression* and *Insertion*. Each node running this IDS also observe the received signal strength, which will detect the anomalies *OmniImpersonation* and *DirImpersonation*. We therefore have our set of anomalies $\mathbf{A}_I = \{$ *Suppression, Insertion, OmniImpersonation, DirImpersonation* $\}$.

The next step is to set the attacker's goal. In their paper, the authors address node impersonation, and resource depletion by excessive generation of traffic. As the latter is not covered by our model, we will focus on impersonation first, and then consider a more general anomaly, *ApplicationDataAltered*.

In order to find if an attacker in our model is able to impersonate honest nodes, we need to choose the facts modeling the attacker. The IDS supposes that attackers have the ability to impersonate honest nodes (fact *CanImpersonate*), and we also suppose that they have directionnal antennas (fact *DirAntenna*) as no assumptions were made about this in the paper. We therefore set $\mathbf{F}_I = \{$ *CanImpersonate, DirAntenna* $\}$.

From the facts and the IDS description, we compute the set of rules $\mathcal{S}$, and search for a way to reach *Impersonation* using $\mathcal{S}$. We see that in this case, the tool cannot apply any rules, and so our system did not find weaknesses on this aspect of the IDS.

However, the assumption that the attacker cannot modify its intruder nodes's transmission power is strong, as such hardware is readily available. If we relax that assumption by removing *TxPowAdjust* from $\mathbf{F}_I$, we find that the attacker can now reach *Impersonation* by doing an impersonation with adjusted transmission power, effectively bypassing the IDS input. To prevent this, the IDS would need a way of detecting this behavior.

We can also consider other intruder goals. For instance, we wonder if an attacker would be able to alter the data going to the application (*ApplicationDataAltered*). Let us assume that nodes can be compromised by an attacker (fact *CompromisableNodes*), and that validity is easy to fake for the attacker (fact *SimpleValidity*). As we assumed that nodes can be compromised, and there are no protections against this in the IDS, the intruder can take control of some nodes, and make them alter the data they retransmit. As we assumed that an attacker can fake the validity of a message, the results of that alteration is valid, thus the altered data will get delivered without any alert.

This attack path uses the fact that traffic flow analysis does not protects against message alterations. To prevent this, the IDS would need countermeasures preventing message alterations, such as choosing the right protocols to have integrity and authenticity of the messages, or using more IDS inputs such as the ones used

by [9]. The other way to prevent this attack would be to prevent the compromise of nodes, either through specific inputs (see for instance [31] or [25]).

4.2 Decentralized Intrusion Detection [9]

In [9], the authors propose an IDS for WSNs based on promiscuous listening. The network contains monitoring nodes, which observe their neighbor's behavior. If their neighbors break one of a series of rules, an alert is raised. To avoid confusion with the rules from our model, we will call the rules from this IDS *behavior rules*. They are the following: A node must receive messages regularly (the interval rule), neighbors must retransmit packets quickly (the delay rule), without altering them, nor repeating them. Also, transmissions must come from a sensible distance, and there must not be too many collisions.

We first build the set of monitored anomalies $\mathbf{A}_I$. The interval rule detects *Suppression* and *Insertion*, as both addition or suppression of messages alters downstream traffic flows. The integrity rule detects *NeighVisibleAlteration* and *NeighVisibleValidAlteration*, as they are based on promiscuous monitoring. The delay rule detects *NeighVisibleSuppression*. The last three behavior rules are not considered in our model, as we do not model any sort of distance measurements for the range rule, nor availability-related anomalies regarding the collision rule. We therefore have our set of anomalies $\mathbf{A}_I = \{$ *Suppression, Insertion, NeighVisibleAlteration, NeighVisibleValidAlteration, NeighVisibleSuppression* $\}$.

There are no specific hypothesis about the nodes in the paper. Regarding facts, we include *CompromisableNodes* to allow the attacker to compromise nodes. We also add *DirAntenna* to model the attacker's access to advanced hardware. Regarding the protocols, we add *EndToEndConfidentiality* to model a protocol ensuring that the data stays confidential, and *SimpleValidity* to be able to find an attack. Thus, we have $\mathbf{F}_I = \{$ *CompromisableNodes, DirAntenna, EndToEndConfidentiality, SimpleValidity* $\}$. For the intruder goal, we select *ApplicationDataAltered* as it encompasses most of what this IDS aims to prevent. The tool's output shows that there is an undetected attack.

Similarly to the previous IDS, this attack stems from the assumption that the attacker can compromise honest nodes, as there are no countermeasures regarding these anomalies. With an associated intruder node, the attacker can therefore modify the data being routed, as we assumed intruder nodes can alter data while keeping packets valid. However, the IDS makes honest nodes monitor their neighbors for such a behavior. This is where we use the directionnal antennas: with these, the intruder node is able to send the altered packet to its destination, while sending the initial version of that packet to the monitors. This way, the attacker can alter data, while appearing to satisfy the behavior rules triggering the intrusion detection. Then, that altered packet will be forwarded to its destination and delivered, effectively altering application data, which is the goal we set.

The straightforward way to prevent this specific attack in our model is to use a protocol that guarantees the validity of the transmitted message. Indeed, when removing the *SimpleValidity* fact, our tool was not able to find an undetected attack. Alternatively, one may try to prevent any association of the intruder, through for instance tamper-resistant nodes and secure software. This way, attackers will not be able to alter the traffic going through the network, and this would also prevent an attacker from reaching *ApplicationDataAltered* in our model.

5 Conclusion

We presented a characterization and modeling of the different data sources used in network-based intrusion detection systems, focusing on wireless ad-hoc networks. We found that a lot of IDS from the literature base their decisions on a small set of distinct inputs, which we have listed in this paper. We then used those inputs to build a decision aid in order to help IDS designers to locate some of the weak points of their algorithms, depending on the protocols used in their network.

In the future, we would like to further refine that model and take into account various families of protocols, especially when looking at the low levels of the protocol stack, such as the medium access protocols. Also, we would like to extend our model to include availability attacks.

References

1. M. Adeyeye and P. Gardner-Stephen. The village telco project: a reliable and practical wireless mesh telephony infrastructure. *EURASIP Journal on Wireless Communications and Networking*, 2011(1):1–11, 2011.
2. F. Anjum and R. Talpade. Lipad: lightweight packet drop detection for ad hoc networks. In *Vehicular Technology Conference, 2004. VTC2004-Fall. 2004 IEEE 60th*, volume 2, pages 1233–1237. IEEE, 2004.
3. D. Basin, C. Cremers, and C. Meadows. Model checking security protocols. *Handbook of Model Checking. Springer, Heidelberg (to appear, 2011), http://people. inf. ethz. ch/cremersc/publications/index. html*, 2011.
4. V. Bhuse and A. Gupta. Anomaly intrusion detection in wireless sensor networks. *Journal of High Speed Networks*, 15(1):33–51, 2006.
5. K. Bonne Rasmussen and S. Capkun. Implications of radio fingerprinting on the security of sensor networks. In *Security and Privacy in Communications Networks and the Workshops, 2007. SecureComm 2007. Third International Conference on*, pages 331–340. IEEE, 2007.
6. T. Clausen and P. Jacquet. Optimized Link State Routing Protocol (OLSR). RFC 3626 (Experimental), Oct. 2003.
7. M. Conti, R. Di Pietro, L. V. Mancini, and A. Mei. A randomized, efficient, and distributed protocol for the detection of node replication attacks in wireless sensor networks. In *Proceedings of the 8th ACM international symposium on Mobile ad hoc networking and computing*, pages 80–89. ACM, 2007.
8. J. Cucurull, M. Asplund, and S. Nadjm-Tehrani. Anomaly detection and mitigation for disaster area networks. In S. Jha, R. Sommer, and C. Kreibich, editors, *Recent Advances in Intrusion Detection*, volume 6307 of *Lecture Notes in Computer Science*, pages 339–359. Springer Berlin Heidelberg, 2010.

9. A. P. R. da Silva, M. H. Martins, B. P. Rocha, A. A. Loureiro, L. B. Ruiz, and H. C. Wong. Decentralized intrusion detection in wireless sensor networks. In *Proceedings of the 1st ACM international workshop on Quality of service & security in wireless and mobile networks*, pages 16–23. ACM, 2005.
10. H. Debar, M. Dacier, and A. Wespi. Towards a taxonomy of intrusion-detection systems. *Computer Networks*, 31(8):805–822, 1999.
11. M. Demirbas and Y. Song. An rssi-based scheme for sybil attack detection in wireless sensor networks. In *Proceedings of the 2006 International Symposium on on World of Wireless, Mobile and Multimedia Networks*, pages 564–570. IEEE Computer Society, 2006.
12. D. Dong, M. Li, Y. Liu, X.-Y. Li, and X. Liao. Topological detection on wormholes in wireless ad hoc and sensor networks. *IEEE/ACM Transactions on Networking (TON)*, 19(6):1787–1796, 2011.
13. S. S. Doumit and D. P. Agrawal. Self-organized criticality and stochastic learning based intrusion detection system for wireless sensor networks. In *Military Communications Conference, 2003. MILCOM'03. 2003 IEEE*, volume 1, pages 609–614. IEEE, 2003.
14. P. Gardner-Stephen and S. Palaniswamy. Serval mesh software-wifi multi model management. In *Proceedings of the 1st International Conference on Wireless Technologies for Humanitarian Relief*, pages 71–77. ACM, 2011.
15. J. Hall, M. Barbeau, and E. Kranakis. Enhancing intrusion detection in wireless networks using radio frequency fingerprinting. In *Proceedings of the 3rd IASTED International Conference on Communications, Internet and Information Technology (CIIT)*, pages 201–206, 2004.
16. Y.-a. Huang and W. Lee. Attack analysis and detection for ad hoc routing protocols. In *Recent Advances in Intrusion Detection*, pages 125–145. Springer, 2004.
17. C. Krügel, T. Toth, and E. Kirda. Service specific anomaly detection for network intrusion detection. In *Proceedings of the 2002 ACM symposium on Applied computing*, pages 201–208. ACM, 2002.
18. Y. Liu, Y. Li, and H. Man. Short paper: A distributed cross-layer intrusion detection system for ad hoc networks. In *Security and Privacy for Emerging Areas in Communications Networks, 2005. SecureComm 2005. First International Conference on*, pages 418–420. IEEE, 2005.
19. H. Nakayama, S. Kurosawa, A. Jamalipour, Y. Nemoto, and N. Kato. A dynamic anomaly detection scheme for aodv-based mobile ad hoc networks. *Vehicular Technology, IEEE Transactions on*, 58(5):2471–2481, 2009.
20. I. Onat and A. Miri. An intrusion detection system for wireless sensor networks. In *Wireless And Mobile Computing, Networking And Communications, 2005.(WiMob'2005), IEEE International Conference on*, volume 3, pages 253–259. IEEE, 2005.
21. I. Onat and A. Miri. A real-time node-based traffic anomaly detection algorithm for wireless sensor networks. In *Systems Communications, 2005. Proceedings*, pages 422–427. IEEE, 2005.
22. J. Parker, A. Patwardhan, and A. Joshi. Cross-layer analysis for detecting wireless misbehavior. In *Proceedings of the IEEE Consumer Communications and Networking Conference (CCNC 2006)*, pages 6–9, 2006.
23. C. Perkins, E. Belding-Royer, and S. Das. Ad hoc On-Demand Distance Vector (AODV) Routing. RFC 3561 (Experimental), July 2003.
24. W. R. Pires Jr, T. H. de Paula Figueiredo, H. C. Wong, and A. A. Loureiro. Malicious node detection in wireless sensor networks. In *Parallel and Distributed Processing Symposium, 2004. Proceedings. 18th International*, page 24. IEEE, 2004.
25. H. Song, L. Xie, S. Zhu, and G. Cao. Sensor node compromise detection: the location perspective. In *Proceedings of the 2007 international conference on Wireless communications and mobile computing*, pages 242–247. ACM, 2007.
26. G. Thamilarasu, A. Balasubramanian, S. Mishra, and R. Sridhar. A cross-layer based intrusion detection approach for wireless ad hoc networks. In *Mobile Adhoc and Sensor Systems Conference, 2005. IEEE International Conference on*, pages 7–pp. IEEE, 2005.

27. C. Tseng, T. Song, P. Balasubramanyam, C. Ko, and K. Levitt. A specification-based intrusion detection model for OLSR. In *Recent Advances in Intrusion Detection*, pages 330–350. Springer, 2006.
28. C. Tseng, S.-H. Wang, C. Ko, and K. Levitt. Demem: Distributed evidence-driven message exchange intrusion detection model for manet. In *Recent Advances in Intrusion Detection*, pages 249–271. Springer, 2006.
29. C.-Y. Tseng, P. Balasubramanyam, C. Ko, R. Limprasittiporn, J. Rowe, and K. Levitt. A specification-based intrusion detection system for AODV. In *Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks*, pages 125–134. ACM, 2003.
30. D. Wagner and P. Soto. Mimicry attacks on host-based intrusion detection systems. In *Proceedings of the 9th ACM Conference on Computer and Communications Security*, pages 255–264. ACM, 2002.
31. Y. Yang, X. Wang, S. Zhu, and G. Cao. Distributed software-based attestation for node compromise detection in sensor networks. In *Reliable Distributed Systems, 2007. SRDS 2007. 26th IEEE International Symposium on*, pages 219–230. IEEE, 2007.