Noname manuscript No.
(will be inserted by the editor)

Priming: Making the Reaction to Intrusion or Fault Predictable

Martin Drozda · Sven Schaust · Sebastian Schildt · Helena Szczerbicka

Received: date / Revised version: date

Abstract We propose and evaluate an immuno-inspired approach for misbehavior detection in ad hoc wireless networks. Misbehavior is the result of an intrusion, or a software or hardware failure. Our misbehavior detection approach is inspired by the role of co-stimulation and priming in the Biological immune system (BIS). We translate priming into a computational paradigm that can increase robustness as well as stimulate energy efficiency of misbehavior detection. We provide a detailed energy consumption analysis with respect to the IEEE 802.11 and IEEE 802.15.4 protocols. We analyze the efficiency of misbehavior detection with co-stimulation and priming. This analysis is complemented with experimental results. We show that co-stimulation and priming introduce new options such as the ability to choose a trade-off between detection performance and energy efficiency. We provide a summary of the challenges related to the design of co-stimulation and priming based architectures. We argue that co-stimulation and priming are rather general paradigms with possible applications in other areas than misbehavior detection.

Keywords Ad Hoc Wireless Network, Sensor Network, Misbehavior Detection, Energy Efficient Design, Co-stimulation, Artificial Immune System

A preliminary version of this paper appeared in: Andrews PS, Timmis J, Owens NDL, Aickelin U, Hart E, Hone A, Tyrrell AM (eds) ICARIS '09: Proceedings of the International Conference on Artificial Immune Systems, York, UK, Springer Berlin / Heidelberg, Lecture Notes in Computer Science, vol 5666, pp 260-273, 2009, URL http://dx.doi.org/10.1007/978-3-642-03246-2_25

M. Drozda · S. Schaust · H. Szczerbicka
Simulation and Modeling Group, Faculty of Electrical Engineering and Computer Science, Leibniz University of Hannover, Welfengarten 1, 30167 Hannover, Germany.
E-mail: {drozda,svs,hsz}@sim.uni-hannover.de

S. Schildt
Institute of Operating Systems and Computer Networks, Technische Universität Braunschweig, Mühlenpfordtstr. 23, 38106 Braunschweig, Germany.
E-mail: schildt@ibr.cs.tu-bs.de

1 Introduction and Motivation

Ad hoc and sensor wireless networks can become an object of attacks and intrusions. The motivation for attacking an ad hoc network can range from a desire to benefit from the network's services to an intent to make it non-functional. Faults that are a result of software or hardware failures can be equally damaging. Correcting the consequences of some faults or attacks might only be possible by a costly human intervention, or not at all. Even though secure protocols address issues connected with data integrity and user authentication, the experience with the Internet shows that flaws in these protocols are continuously being found and exploited (Yegneswaran et al 2003).

This establishes the basic motivation for designing autonomous detection and response systems that aim at offering an additional line of defense to the employed secure protocols. Such systems should provide several layers of functionality including the following: (i) distributed self-learning and self-tuning with the aspiration to minimize the need for human intervention and maintenance and (ii) active response with focus on attenuation and possibly elimination of negative effects of faults or attacks on the network.

In many scenarios, ad hoc and sensor networks are expected to be based on wireless devices with limited (battery) resources. In order to stimulate the survivability of such networks (Sterbenz et al 2002), it is essential that autonomous detection and response systems reflect these resource constraints.

Best current practices for misbehavior detection in ad hoc wireless networks are almost exclusively done on a domain knowledge basis; see (Anantvalee and Wu 2007) and references therein. Although such an approach allows us to find a good predictor for a specific type of misbehavior, it fails to provide a misbehavior detection framework that would offer good extensibility towards new misbehavior types. Furthermore, the area of energy efficient misbehavior detection remains to be an open problem.

We assume that upon deployment of an ad hoc network, an enforcement of operational strategies in an energy efficient way is desired. Such operational strategies may impose performance limits in the form of e.g. the maximum data packet loss at a node. One possibility for determining whether a node keeps forwarding data packets is monitoring by other nodes. This is however a very costly approach since the monitoring nodes cannot enter sleep mode to preserve their energy resources. It is therefore necessary that monitoring will be kept to a minimum.

Our contribution is an approach offering a reasonable trade-off between energy efficiency and misbehavior classification performance. Our solution is an algorithm that allows for energy efficient misbehavior detection that can also be applied to just deployed ad hoc networks. Since the nodes in such ad hoc networks were not given the time required to observe and analyze the events and states of their neighboring nodes, they lack the data basis necessary for an efficient learning of the normal behavior and misbehavior.

To stimulate energy efficiency, we exploit an inherent property of ad hoc networks. This property characterizes the relationship between continuous and time window based data traffic monitoring. In the former case, each data packet is overheard and analyzed by a neighboring node, whereas in the latter case a data traffic statistic using a fixed size time window is being computed by two neighboring nodes. The statistics computed by these two nodes are then compared and analyzed. Energy efficiency as well as capability of our approach to suppress false positives is achieved by combining continuous and time window based data traffic monitoring. This mixed design also facilitates tuning detection rate and false positives rate in an independent way. This is due to the fact that the detection rate is influenced by the detection performance of time window based data traffic monitoring, whereas the false positives rate is influenced by the efficiency of continuous data traffic monitoring. Combining these two types of data traffic monitoring is inspired by the role of the innate and adaptive immune systems and by their ability to communicate.

This document is organized as follows. In Section 2, we discuss the basic principles of the Biological immune system. In Section 3, we review the related work. In Section 4, we introduce ad hoc networks. We also summarize several relevant communication protocols and their assumptions. In Section 5, we introduce several types of misbehavior and explain our approach to classification performance evaluation. In Section 6, we introduce co-stimulation and priming as computational paradigms for ad hoc networks. In Section 7, we present our experimental setup. In Section 8, we discuss the performance of the proposed priming approach. In Section 9 we summarize the research challenges. In Section 10, we summarize the results and conclude.

2 The Biological Immune System

The Biological immune system (BIS) (Murphy et al 2008) can quickly recognize the presence of foreign microorganisms in the human body. It is remarkably efficient in correctly detecting and eliminating pathogens such as viruses, bacteria, fungi or parasites. When confronted with a pathogen, the BIS often relies on a coordinated response from both of its two vital parts:

- the *innate system*: the innate immune system is able to recognize the presence of a pathogen or tissue injury, and is able to signal this to the adaptive immune system.
- the *adaptive system*: the adaptive immune system can develop during the lifetime of its host a specific set of immune responses and provide immunological memory. Immunological memory serves as a basis for a stronger immune response, should a pathogen re-exposure happen.

The form and amplitude of immune responses is pathogen dependent. Often, an immune response within the BIS is based on a feedback mechanism between the innate and adaptive immune systems. Such a feedback can result in a feedback loop, in which the innate immune system further stimulates the adaptive immune system, and vice versa (Frauwirth and Thompson 2002). For example, a pathogen gets eliminated, if it was recognized by the adaptive immune system as a pathogen, and at the same time, the innate immune system signals that this pathogen causes some damage to the human organism. Under this specific immune reaction, only damage inflicting or *infectious* cells get eliminated by the BIS.

This demonstrates that a two-way communication, hereafter referred to as *co-stimulation*, between the innate and adaptive immune systems is common. Immunologists such as Frauwirth and Thompson describe co-stimulation as the involvement of *"reciprocal and sequential signals between cells"* in order to fully activate a lymphocyte (Frauwirth and Thompson 2002). The role of lymphocytes is to recognize a specific pathogen, to trigger a corresponding immune reaction, in some forms they are also capable of pathogen elimination.

Priming in the BIS describes the effects of a first encounter of an immune cell with a pathogen. More specifically, immunologists define priming as the activation and clonal expansion of certain immune cells into effector cells that are then capable of inducing a full immune response against a specific pathogen.

Communication capabilities within the BIS received an increased interest from the Artificial immune systems (AIS) community and evolved into an independent research direction. Several different types of danger, safe and amplifying signals were proposed within the Danger theory due to Aickelin et al. (Aickelin et al 2003).

In our work, we were motivated by the ability of the BIS to act in a coordinated way when confronted with a pathogen. Our misbehavior detection approach is inspired by the role of co-stimulation and priming in the BIS. We translate priming into a feasible computational paradigm. We show that co-stimulation and priming in ad hoc networks not only increases the overall misbehavior detection robustness but also introduce new options such as the ability to choose a trade-off between detection performance and energy efficiency.

3 Related Work

Co-stimulation as an approach for misbehavior detection received thus far only a limited interest from the research community.

In one of the first BIS inspired works, Hofmeyr and Forrest described an AIS able to detect anomalies in a wired TCP/IP network (Hofmeyr and Forrest 1999). Anomaly detection was based on the negative selection algorithm (Forrest et al 1994). After this mechanism detected an anomaly, a message was sent to a human operator. He was given 24 hours to confirm a detected attack. This means, two qualitatively different sorts of classification were used: negative selection and human expertise. The second approach is only applied, if a co-stimulation in the form of a message is received.

Sarafijanović and Le Boudec introduced an AIS for misbehavior detection in mobile ad hoc wireless networks (Sarafijanovic and Le Boudec 2004). A local mechanism applied by each node in the network required a co-stimulation in order to classify a neighbor as misbehaving. The origin of co-stimulation was a TCP connection source that perceived data losses. The information about a perceived data loss was forwarded along the connection. The authors observed that their form of co-stimulation reduces the false positives rate. The disadvantage of their approach is its tight coupling with a transport layer protocol (TCP), thus negatively influencing energy consumption.

Drozda et al. proposed in (Drozda et al 2010) a co-stimulation based approach to misbehavior detection in ad hoc wireless networks. Their observations were two-fold: co-stimulation can help reduce the energy consumption of misbehavior detection and it also significantly reduces the false positives rate. Their approach does not rely on a transport protocol as a source of co-stimulation. Instead, it benefits from the differences in energy efficiency and detection performance, when misbehavior classification is done in a cooperative way and a centralized way. The less energy efficient approach is applied only if the other one detects a misbehavior, i.e. the first type of classification provides co-stimulation for the other type of classification.

Even though our research focus stays on co-stimulation and its suitability for misbehavior detection, we would also like to offer insight into related approaches not taking advantage of this technique.

Marti et al. introduced in (Marti et al 2000) two techniques for data dropping detection in ad hoc networks: watchdog and pathrater. A watchdog allows for collecting traffic information about neighboring nodes using promiscuous radio mode. Based on this information, a pathrater can assess route reliability, thus maximizing the chance that data packets get correctly delivered. Due to the energy consumption connected with operation in promiscuous node, the watchdog technique is not suitable for energy constrained ad hoc networks, even though, it offers high data packet dropping detection rates (Drozda et al 2010).

Hu et al. introduced two approaches using *packet leashes* for wormhole detection in mobile wireless networks (Hu et al 2003, 2006). A wormhole is an out-of-band connection between two devices which allows an attacker to manipulate the network topology and thus the routing of packets. A leash either consists of an authenticated timing or location based information. A metric calculating whether a packet has traveled further than allowed (or physically possible) is applied to the leashes. While the first approach requires a tightly time synchronized network, the other one requires GPS (Global Positioning System) based geographical information. The disadvantage of the former approach is the increased message and authentication complexity, the disadvantage of the latter approach is the increased energy consumption connected with GPS device operation.

Huang and Lee introduced a cooperative approach for intrusion detection in ad hoc networks (Huang and Lee 2003). It takes advantage of a comprehensive set of 141 data traffic features. The sampling rate for the feature computation was 5 seconds. It applies a decision tree for the classification of traffic samples based on these features. It thus requires a learning period in which the classifier gets trained. To improve the energy efficiency of their approach, a cooperative approach is considered. Under this approach a clique of nodes elects a cluster head. This cluster head does network monitoring on behalf of other clique members until a new cluster head is elected. The disadvantage of such an approach is the necessity to overhear the data traffic in promiscuous mode and to reelect new cluster heads so that the monitoring load gets evenly distributed among the clique members.

Bhuse et al. proposed an approach for data packet dropping detection that does not rely on promiscuous mode (Bhuse et al 2005). In their approach each connection destination node sends to the source connection node a statistic about received data packets. This is done over an alternative route that is computed by the routing protocol. The received statistic is then compared with a similar statistic computed by the source node. Based on this, the source node can decide whether all nodes on the connection cooperate in data packet forwarding. This approach does not allow for detecting the individual misbehaving node. The energy consumption overhead of this approach when the DSR protocol is used for alternative route discovery is 0.6-2.6% for paths of length 3 to 13 hops. The ability of this approach to find an alternative route decreases as the number of data dropping nodes increases as well as with the connection length. For example, when 10% nodes are dropping data packets the reported success rate is about 75% and 25% for route lengths of 3 and 13 hops, respectively.

Gonzales et al. proposed an approach for data dropping detection in ad hoc networks (Gonzalez et al 2007). Their approach is based on the principle of flow conservation. Under this approach, it is assumed that the number of data packets, that a node receives, equals the number of data packets that it forwards excluding the cases when the given node is also the destination node. Whether a node forwards data packets is determined by its neighbors in promiscuous mode. The focus of their experimental evaluation was to determine the ratio of data packets that a node is allowed to drop without being detected. This ratio is measured relative to the maximum number of data packets that a node is allowed (or expected) to drop. Their experimental evaluation shows that this ratio is about 10-15%.

Krishnamurthy et al. introduced a machine learning misbehavior detection approach for wireless sensor networks (Krishnamurthy et al 2009). They considered several misbehavior types: continuous signal jamming, signal jamming applied only if there is other radio transmission detected, data packet redirecting and data packet dropping. The latter misbehavior type required data traffic observation in promiscuous mode. Classification was done using a linear discriminant analysis and/or a fixed-width clustering with different distance measures. They tested their approach on a sensor network based on Crossbow MicaZ motes and TinyOS. The memory footprint of their approach was 31,342 bytes and 3,500 bytes of flash memory and data memory, respectively. The reported detection accuracy for data packet dropping was nearly 100% with linear discriminant analysis applied.

With respect to the above reviewed work, our goal was to benefit from the capability of co-stimulation to suppress false positives. At the same time, due to energy efficiency concerns we aimed at minimizing any overhearing in promiscuous mode. When evaluating the energy efficiency of our approach, we compare against watchdog based misbehavior detection. Watchdog based misbehavior detection offers solid detection performance in scenarios where the ambition is to identify a specific node executing data dropping, data delaying or a similar misbehavior type (Drozda et al 2010). Such a solid detection performance makes it straightforward to understand the trade-off between detection performance and energy efficiency, i.e. to understand whether any decrease in detection performance was matched by an increase in energy efficiency.

The main discerning factor between our approach and the approaches reviewed above is that our approach develops a conceptual framework for misbehavior detection in ad hoc wireless networks. The reviewed approaches concentrate on achieving an acceptable misbehavior detection performance, whereas we attempt to provide a means that offers the possibility to influence the detection rate, the false positives rate and the energy efficiency in a controlled manner. This allows us to provide insight on how these three objectives influence each other.

4 Protocols and Assumptions

We now summarize the definitions, protocols, mechanisms and assumptions relevant to our misbehavior detection approach and the related performance evaluation.

We assume that an *ad hoc network* is a net $N = (n(t), e(t))$ where $n(t), e(t)$ are the set of nodes and edges at time t, respectively. Nodes correspond to wireless devices that wish to communicate with each other. An edge between two nodes A and B is said to exist when A is within the radio transmission range of B and vice versa. A *sensor network* is a static ad hoc network deployed with the goal to monitor environmental or physical conditions such as humidity, temperature, motion or noise.

The acquisition of a route between two arbitrary nodes in an ad hoc network is done by a routing protocol. We use the AODV routing protocol (Perkins and Royer 1999). AODV is an on-demand routing protocol that starts a route search only when a route to a destination is needed. This is done by flooding the network with RREQ (= Route Request) control packets. The destination node or an intermediate node that knows a route to the destination will reply with a RREP (= Route Reply) control packet. This RREP follows the route back to the source node and updates routing tables at each node that it traverses. A RERR (= Route Error) packet is sent to the connection originator when a node finds out that the next node on the forwarding path is not replying. Each node maintains a routing table that contains the information about destination nodes with known routes, their distance in terms of hop count and the next hop node for a given destination.

In order to avoid data transmission by several neighboring nodes at once, the access to the wireless medium needs to be coordinated. This is often done on a medium contention basis. When attempting to transmit a data packet, the IEEE 802.11 MAC protocol (IEEE Std. 802.11 2007) uses carrier sensing followed by an RTS-CTS-DATA-ACK handshake (RTS = Ready to send, CTS = Clear to send, ACK = Acknowledgment). The RTS-CTS-DATA-ACK handshake can be optionally disabled. This happens if the data packet size is equal or smaller than the RTS threshold. The default value for this threshold is 2432 bytes. The threshold can be adjusted as required by the data traffic pattern. The max. data rate for IEEE 802.11b and IEEE 802.11g protocols is 11Mbit/s and 54Mbit/s, respectively. They both operate in the 2.4GHz frequency band. Other MAC protocols such as the 802.15.4 MAC protocol (IEEE Std. 802.15.4 2003) avoid using the RTS-CTS-DATA-ACK handshake, only relying on carrier sensing to access the medium, in order to decrease energy consumption at nodes. This protocol is aimed at low data rate wireless networks such as sensor networks. The max. data rate is 250kbit/s.

In *promiscuous mode*, a node listens to the on-going traffic among other nodes in the neighborhood and collects information from the overheard packets. Promiscuous mode is energy inefficient because it prevents the wireless interface from entering sleep mode, forcing it into either idle or receive mode. There is also extra computational overhead caused by analyzing all overheard packets. According to (Feeney and Nilsson 2001), power consumption in idle and receive modes is about 12-20 higher than in sleep mode.

We do not assume any node location knowledge or time synchronization among nodes. We assume that packets are authenticated, i.e. the originator of any packet as well as changes in the packet body can be easily identified. This is a reasonable assumption in-line with e.g. the ZigBee specification (ZigBee Alliance 2005). This requirement can be relaxed in cases when the goal is to detect software or hardware failures rather than some more severe instances of malicious misbehavior executed after a node intrusion.

5 Misbehavior Modeling and Classification Performance Evaluation

Node misbehavior can be the result of an intrusion, or a software or hardware failure. For the purpose of our experimental evaluation we consider three misbehavior types: data packet dropping (qualitative misbehavior), data packet delaying (quantitative misbehavior) and wormholes (topology related misbehavior) (Hu et al 2006). In data packet dropping, the misbehaving node drops a given data packet randomly and uniformly with probability α. In data delaying, the misbehaving node delays the forwarding of a given data packet randomly and uniformly with probability β by a fixed delay amount δ. Wormholes are private (out-of-band) links between one or several pairs of nodes. They are added by an attacker in order to redirect data traffic and thus gain control over packet routing. Wormholes are an example of misbehavior done by several nodes in *collusion*, whereas the other two types are done by individual nodes.

The approach that we discuss herein can also be extended to other misbehavior types. Since any additional misbehavior type increases the complexity of experimental evaluation, we had to keep their number at a reasonable level. However, in view of the fact that solutions that allow for detection of a specific misbehavior type are not uncommon, it was our goal to demonstrate that our approach can deal with several misbehavior types at once.

The necessary traffic samples for the three misbehavior classes and the normal class were obtained through network simulation by applying one of the above misbehavior models or running a misbehavior free simulation. The detailed experimental setup is discussed in Section 7.

Classification performance in our experiments was evaluated in terms of detection rate and false positives (FP) rate. These two measures were computed as follows:

$$det.\ rate_{c_j} = \frac{c_{c_j}}{n_{c_j}} \times 100.0\% \qquad FP\ rate_{c_j} = \frac{FP_{c_j}}{FP_{c_j} + c_{c_j}} \times 100.0\% \tag{1}$$

where $c_j = \{normal, dropping, delaying, wormhole\}$. n_{c_j} is the number of vectors (samples) labeled with the j-th class c_j; note that $n_{c_j} > 0$ in all our experiments. c_{c_j} is the number of vectors that were correctly classified as belonging to the class c_j. FP_{c_j} is the number of samples incorrectly predicted as belonging to c_j. 95% confidence intervals ($CI_{95\%}$) were computed for each measure. Classification performance with respect to a vector set was evaluated by means of the classification error ϵ:

$$\epsilon = \frac{\sum_{c_j} FP_{c_j}}{\sum_{c_j} n_{c_j}} \times 100.0\% \tag{2}$$

6 Co-stimulation in Ad Hoc and Sensor Networks

Drozda et al. introduced and evaluated a co-stimulation architecture inspired by the interplay between the innate and adaptive immune system (Drozda et al 2010). An important task when implementing this architecture was to identify several feature sets, each offering a different perspective on a node's behavior. Next we present the considered feature sets and the proposed architecture. Later we turn our attention to our contribution, a priming approach based on this architecture. Our priming approach takes advantage of the capability of this architecture to decrease the false positives rate.

24 features suitable for misbehavior detection from several layers of the OSI protocol stack were considered. These features were divided into several subsets with respect to their energy requirements and protocol assumptions. A wrapper approach (Kohavi and John 1997) was used to identify features with a *statistically significant contribution* to the detection of the three considered misbehavior types. More specifically, each of the 24 features was tested whether it significantly decreases the classification error with respect to these misbehavior types. The feature that decreased the classification error the most was chosen for the final feature set. The process was then repeated with the remaining features until there was no other feature that could significantly decrease the classification error. This was coupled with cross-validation in order to obtain a robust estimate of the classification error in each round. The features that were identified through this process are listed below.

These features can be computed without much computational overhead. Our misbehavior detection results could have been better if a more complex Fourier or wavelets analysis of the packet stream had been done. As the results by Barford et al. point out (Barford et al 2002), this could lead to good anomaly detection rates.

6.1 The Features

Let $s_s, s_1, ..., s_i, s_{i+1}, s_{i+2}, ..., s_d$ be the path between s_s and s_d determined by a routing protocol, where s_s is the source node, s_d is the destination node. The features in the following feature set f are averaged over a time window. We use the feature labels (M3, M4, ...) as they were introduced in (Drozda et al 2010).

MAC Layer Features:

M3 **Forwarding index (watchdog) (Marti et al 2000):** Ratio of data packets sent from s_i to s_{i+1} and then subsequently forwarded to s_{i+2}.

M4 **Processing delay index (quantitative watchdog):** Time delay that a data packet accumulates at s_{i+1} before being forwarded to s_{i+2}.

Routing Layer Features:

R5 **Average distance to destination:** Average number of hops from s_i to any known destination.

R9 **Connectivity index:** Number of destinations with known routes as recorded in the routing table of node s_i.

R12 **Diameter index:** Number of hops to the furthermost destination as recorded in the routing table of node s_i.

Transport Layer Features:

T1 **Out-of-order packet index:** Number of DATA packets that were received by s_i out of order. This assumes that the connection source uses a predictable scheme for computing data packet IDs. Normalized by the time window size.

T2 **Interarrival packet delay index 1:** Average delay between DATA packets received by s_i. The delay was computed separately *for each connection* and then a master average was computed.

T3 **Interarrival packet delay variance index 1:** Variance of delay between DATA packets received by s_i. The variance was computed separately for *each connection* and then a master average was computed.

T4 **Interarrival packet delay index 2:** Average delay between DATA packets received by s_i.

T5 **Interarrival packet delay variance index 2:** Variance of delay between DATA packets received by s_i.

All the above features can be locally computed. The features $T2$, $T4$ and $T3$, $T5$ are identical, if only DATA packets belonging to a single connection are received by s_i. $M3$ and $M4$ require operation in promiscuous mode and therefore can be considered energy inefficient. We consider the following two subsets of the feature set f:

1. $f_0 = \{M3, M4\}$
2. $f_1 = \{R5, R9, R12, T1, T2, T3, T4, T5\}$

To distinguish between a feature set and its numerical *instance* computed in a given time window, we introduce the following "hat" notation: $\hat{f}_0$ and $\hat{f}_1$. In our experimental setup, we apply vector representation to $\hat{f}_0$ and $\hat{f}_1$.

Algorithm 1 Co-stimulation based misbehavior detection at node s_i

```
Require: Sufficient data traffic in current time window at node $s_i$
Require: $\hat{f}_1^{s_{i+2}}$ from 2-hop downstream node
 1: procedure DETECT_MISBEHAVIOR
 2:     $\hat{f}_1^{s_i} \leftarrow$ COMPUTE_$f_1$_FEATURE_SAMPLE($s_i$)
 3:     SEND_UPSTREAM($\hat{f}_1^{s_i}$)
 4:     $\hat{\mathcal{F}}_1 \leftarrow \hat{f}_1^{s_i} \circ \hat{f}_1^{s_{i+2}}$
 5:     if CLASSIFICATION($\hat{\mathcal{F}}_1$) == misbehavior then
 6:         suspicious ← True
 7:     end if
 8:     if suspicious == True then                    ▷ Co-stimulation: $\mathcal{F}_1 \rightarrow f_0$
 9:         $\hat{f}_0 \leftarrow$ COMPUTE_$f_0$_FEATURE_SAMPLE($s_i$)
10:         if CLASSIFICATION($\hat{f}_0$) == misbehavior then
11:             MARK_AS_MISBEHAVING($s_{i+1}$)          ▷ Misbehavior confirmed
12:         end if
13:     end if
14: end procedure
```

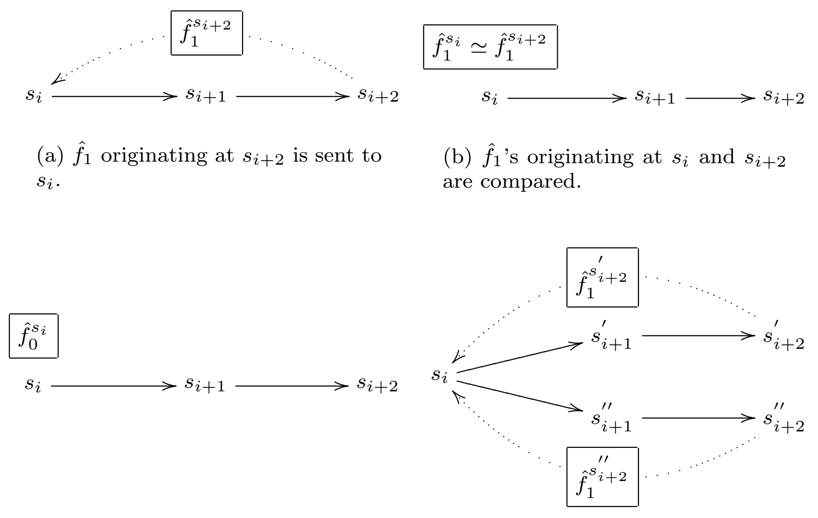
6.2 A Co-stimulation Based Approach

With the goal to achieve a misbehavior detection performance with low false positives rate, a co-stimulation inspired mechanism is considered in (Drozda et al 2010). This mechanism mimics the ability of communication between various players of the innate and adaptive immune system. More specifically, the inspiration was based upon the observation that a cell will not be marked as infectious and thus eliminated as long as no co-stimulation from other BIS players is received. Thus by analogy, a node should not mark another node as misbehaving as long as it does not receive a co-stimulatory signal from a third-party node. Such a signal should be preferably based on a distinctly different form of misbehavior detection than that applied by the primary node.

The co-stimulation inspired approach is depicted in Fig. 1. Each node in the network computes $\hat{f}_1$. This is then proliferated upstream (towards the connection source); see Fig. 1(a). Each $\hat{f}_1$ travels exactly two hops; in our example from s_{i+2} to s_i. If s_i receives $\hat{f}_1$ originating at one of its neighbors, it will forward this $\hat{f}_1$ in the upstream direction. If this $\hat{f}_1$ is not originating at one of its neighbors, it is used but not forwarded.

Since the computation of $\hat{f}_1$ is time window based, the frequency with which $\hat{f}_1$ gets sent depends on the time window size. Upon receiving $\hat{f}_1$ from a two-hop neighbor, the node s_i compares it with its own $\hat{f}_1$ sample. This is done by evaluating the differences between corresponding features in these two vectors (e.g. $T1$ computed by s_i is compared with $T1$ computed by s_{i+2}); see Fig. 1(b). Based on this, a behavior classification with respect to the node s_{i+1} is done.

If s_i classifies s_{i+1} as misbehaving, then it computes a sample based on the f_0 feature set. This means, a *co-stimulation* from the f_1 based classification approach is needed in order to activate the less energy efficient f_0 based classification approach; see Fig. 1(c). If misbehavior gets confirmed, the node s_{i+1} is marked as misbehaving. Note that s_i can receive $\hat{f}_1$ samples from several two-hop neighbors, if multiple connections are running over this node; see Fig. 1(d) for an example. More formally, we assume $|s_i \bullet| \geq 1$ and $|s_{i+1} \bullet| \geq 1$, where $s_k\bullet$ is the set of all successor nodes of s_k.



(a) $\hat{f}_1$ originating at s_{i+2} is sent to s_i.

(b) $\hat{f}_1$'s originating at s_i and s_{i+2} are compared.

(c) *Co-stimulation:* if (b) indicates that s_{i+1} is misbehaving, $\hat{f}_0$ will be computed and evaluated.

(d) If s_i forwards data packets for several connections, multiple $\hat{f}_1$'s are computed and sent.

Fig. 1 Immuno-inspired misbehavior detection approach.

If the MAC layer protocol takes advantage of the RTS-CTS-DATA-ACK handshake then the proliferation of $\hat{f}_1$ can be implemented without adding any extra communication complexity by attaching this information to CTS or ACK packets. As long as there are DATA packets being forwarded on this connection, the feature set can be propagated. If there is no DATA traffic on the connection (and thus no CTS/ACK packets exchanged), the relative necessity to detect the possibly misbehaving node s_{i+1} decreases. Optionally, proliferation of $\hat{f}_1$ can be implemented by broadcasting it with a lower time-to-live value or by using a standalone packet type.

If the node s_{i+1} decides not to cooperate in forwarding the feature set information, the node s_i will switch, after a time-out, to the f_0 feature set computation. In this respect, not receiving $\hat{f}_1$ can be interpreted as a form of *negative co-stimulation.* If the goal is to detect a node that misbehaves due to an intrusion, it is important that the originator of $\hat{f}_1$ can be unambiguously identified, i.e. strict authentication is necessary. This requirement for message authentication can be lifted when the goal is to detect software or hardware failures. An additional requirement is the use of sequence numbers for $\hat{f}_1$. Otherwise, the misbehaving node s_{i+1} could interfere with the mechanism by forwarding an outdated cached $\hat{f}_1$.

We introduce the following notation in order to keep track of composite feature sets f_1 applied at the nodes s_i and s_{i+2}: $\mathcal{F}_1^{s_i} = f_1^{s_i} \cup f_1^{s_{i+2}}$. Similarly, $\hat{\mathcal{F}}_1^{s_i} = \hat{f}_1^{s_i} \circ \hat{f}_1^{s_{i+2}}$, where $\circ$ is the operator of vector concatenation. For simplicity, we omit any superscript, whenever the node identity is clear. The co-stimulation misbehavior detection approach is formally presented in Alg. 1. With respect to our notation, it can be also succinctly expressed as:

$$\mathcal{F}_1 \xrightarrow{co-stimulation} f_0 \tag{3}$$

The mechanisms of the innate immune system bear a certain resemblance to the f_0 based classification phase; see Fig. 1(c). The innate system is for example very efficient in signaling tissue injury or damage to the adaptive immune system. To a certain degree it relies on some very rudimentary methods such as recognizing an unusually high level of dead or damaged self cells (e.g. blood cells). This can be directly compared with the very straightforward functionality of watchdogs. Similarly, the more learning extensive classification approach based on $\mathcal{F}_1$ (Fig. 1(b)) can be compared with the adaptive immune system.

6.3 Properties of the f_0 and $\mathcal{F}_1$ Feature Sets

The features in these sets are averaged over a time window of size *win. size*. The experimental results in (Drozda et al 2010) show that for the three considered misbehavior types, the properties of $\mathcal{F}_1$ and f_0 can be summarized with respect to the classification error ϵ and the energy cost ξ as follows:

1. For *win. size* $\rightarrow 0$, it holds:

$$\lim_{win.\ size \rightarrow 0} \epsilon(\mathcal{F}_1) \approx \epsilon(f_0) \tag{4}$$

This characterizes the relationship between watchdog and $\mathcal{F}_1$ based misbehavior detection. It points out that instead of observing *each* data packet's delivery in promiscuous mode by the node s_i, it can be equally well done in a cooperative way by s_i and s_{i+2}, if *win. size* $\rightarrow 0$.

2. For *win. size* $\gg 0$, it holds:

$$\epsilon(\mathcal{F}_1) > \epsilon(f_0) \tag{5}$$

$$\xi(\mathcal{F}_1) < \xi(f_0) \tag{6}$$

The measure of energy cost ξ includes feature computation costs as well as all induced communication costs. The communication costs for f_0 are related to overhearing in promiscuous mode. The communication costs for $\mathcal{F}_1$ are related to the necessity to transmit $\hat{f}_1^{s_{i+2}}$ over two hops to s_i.
The first inequality reflects the fact that overhearing each data packet in promiscuous mode gives a better base for classification than a classification based on features computed by two distinct nodes and aggregated over a time window. The other inequality reflects the fact that operation in promiscuous mode is inherently energy inefficient; see the comments in Section 4.

6.4 Co-stimulation and its Misbehavior Detection Efficiency

The misbehavior detection results reported in (Drozda et al 2010) show that for a scenario applying 500-second time window, the f_0 based detection rate is 97.19±0.85% and the FP rate is 1.77±0.66%. In scenarios, where communication costs are negligible, only the f_0 based classification approach would get applied. Unfortunately, in the case of ad hoc networks (and similar distributed computing environments), the cost of overhearing in promiscuous mode is very high; see the comments in Section 4.

Let Ω be the set of all vectors subject to misbehavior classification. The vectors in Ω represent the behavior of monitored nodes (or in general, any other type of objects). In our case, a vector $v_m^{s_i} \in \Omega$ has two components: $v_m^{s_i} = \hat{\mathcal{F}}_1^{s_i} \circ \hat{f}_0^{s_i}$, where m identifies the time window in which $\hat{\mathcal{F}}_1^{s_i}$ was computed. Dependent upon the evaluation of $\hat{\mathcal{F}}_1^{s_i}$, the computation of $\hat{f}_0^{s_i}$ is started in the time period following the time window m. $\mathcal{F}_1$ and f_0 based classification of $v_m^{s_i}$ is done using the $\hat{\mathcal{F}}_1^{s_i}$ component and the $\hat{f}_0^{s_i}$ component, respectively.

Let $\Omega_{\mathcal{F}_1} \subseteq \Omega$ be the subset of vectors that were marked as representing suspicious behavior, after the $\mathcal{F}_1$ based classification was done. Let us first assume that $\epsilon^{\Omega}(f_0) = 0$, where $\epsilon^{\Omega}(f_0)$ is the classification error of f_0 based classification applied to the vector set Ω. If f_0 based classification is applied to $\Omega_{\mathcal{F}_1}$, it clearly holds:

$$\epsilon^{\Omega_{\mathcal{F}_1}}(f_0) = 0 \tag{7}$$

This implies, for the final FP rate and a given misbehavior class c_j, after co-stimulation is applied, it holds:

$$FP\ rate_{c_j}^{\Omega}(\mathcal{F}_1 \rightarrow f_0) = 0 \tag{8}$$

In other words, the conditional application of f_0 based classification removes all misclassified vectors. Furthermore, the final detection rate for the given class is determined by the detection rate after the $\mathcal{F}_1$ based classification:

$$det.\ rate_{c_j}^{\Omega}(\mathcal{F}_1 \rightarrow f_0) = det.\ rate_{c_j}^{\Omega}(\mathcal{F}_1) \tag{9}$$

Since to achieve $\epsilon^{\Omega}(f_0) = 0$ may not be possible, Eqs. 8 and 9 translate for $\epsilon^{\Omega}(f_0) \neq 0$ to:

$$FP\ rate_{c_j}^{\Omega}(\mathcal{F}_1 \rightarrow f_0) = FP\ rate_{c_j}^{\Omega}(f_0) \tag{10}$$

$$det.\ rate_{c_j}^{\Omega}(\mathcal{F}_1 \rightarrow f_0) \leq det.\ rate_{c_j}^{\Omega}(\mathcal{F}_1) \tag{11}$$

This means, the final FP rate is only depending on the efficiency of f_0 based classification. Eq. 10 is based on the assumption that when classifying Ω and $\Omega_{\mathcal{F}_1}$ using f_0 based classification, the following holds:

$$FP\ rate_{c_j}^{\Omega_{\mathcal{F}_1}}(f_0) = FP\ rate_{c_j}^{\Omega}(f_0) \tag{12}$$

To what extent such an assumption is reasonable, is one of the goals of our experimental analysis. Notice that co-stimulation introduces an implicit class of "undecided" vectors. These are the vectors that were marked as suspicious, but this was not confirmed through the f_0 based classification. Such undecided vectors are *not* reclassified according to f_0, since in our classification approach, the classification results based on f_0 and $\mathcal{F}_1$ must coincide.

The results reported in (Drozda et al 2010) show, for a scenario with $win.\ size = 500s$ and with respect to the general misbehavior class mis (in bimodal normal-misbehavior classification), the following:

$$FP\ rate_{mis}^{\Omega}(\mathcal{F}_1 \rightarrow f_0) = 1.67 \pm 2.59\% \ \cong\ FP\ rate_{mis}^{\Omega}(f_0) = 1.77 \pm 0.66\% \tag{13}$$

$$det.\ rate_{mis}^{\Omega}(\mathcal{F}_1 \rightarrow f_0) = 78.89 \pm 1.71\% \ \cong\ det.\ rate_{mis}^{\Omega}(\mathcal{F}_1) = 76.40 \pm 2.53\% \tag{14}$$

The achieved $FP\ rate_{mis}^{\Omega}(\mathcal{F}_1)$ was $15.09 \pm 2.27\%$. Comparing the results for the $\mathcal{F}_1$ and $\mathcal{F}_1 \rightarrow f_0$ based classification, it is clear that $\mathcal{F}_1 \rightarrow f_0$ offers a detection rate comparable to $\mathcal{F}_1$, however, a much lower false positives rate.

Algorithm 2 Initialization and Error Propagation at node s_i

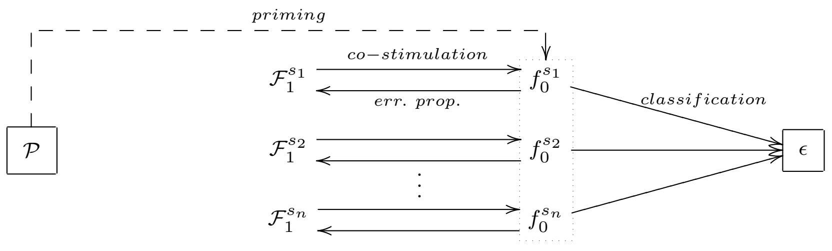
```
$\bar{f}_0$: an ordered set of cardinality $|\bar{f}_0|$ of $\hat{f}_0$ feature vectors
$\bar{\mathcal{F}}_1$: an ordered set of cardinality $|\bar{\mathcal{F}}_1|$ of $\hat{\mathcal{F}}_1$ feature vectors
$\hat{f}_0[w]$, $\hat{\mathcal{F}}_1[w]$: the feature vectors $\hat{f}_0$ and $\hat{\mathcal{F}}_1$, respectively, computed in the time window $w$
Require: Sufficient data traffic at nodes $s_i$ and $s_{i+2}$
Require: Set of priming thresholds $\mathcal{P}$
 1: $\bar{f}_0 = \bar{\mathcal{F}}_1 = \emptyset$
 2: repeat for each window $w$
 3:     $\hat{f}_0 \leftarrow$ COMPUTE_$f_0$_FEATURE_SAMPLE($s_i$)
 4:     $\hat{f}_{1L} \leftarrow$ COMPUTE_$f_1$_FEATURE_SAMPLE($s_i$)
 5:     SEND_UPSTREAM($\hat{f}_{1L}$)
 6:     $\hat{f}_{1R} \leftarrow$ RECEIVE_$f_1$_FEATURE_SAMPLE($s_{i+2}$)
 7:     if $\hat{f}_{1R} \neq \emptyset$ then
 8:         $\bar{f}_0 \leftarrow \bar{f}_0 \cup \{\hat{f}_0\}$
 9:         $\hat{\mathcal{F}}_1 \leftarrow \hat{f}_{1L} \circ \hat{f}_{1R}$
10:         $\bar{\mathcal{F}}_1 \leftarrow \bar{\mathcal{F}}_1 \cup \{\hat{\mathcal{F}}_1\}$
11:     end if
12: until $|\bar{f}_0|$ == target_size
13: for all windows $w$ in $\bar{f}_0$ do
14:     if $\hat{f}_0[w]$ violates $\mathcal{P}$ then
15:         LABEL_VECTOR_AS_MISBEHAVIOR($\hat{\mathcal{F}}_1[w]$)
16:     else
17:         LABEL_VECTOR_AS_NORMAL($\hat{\mathcal{F}}_1[w]$)
18:     end if
19: end for
20: TRAIN_CLASSIFIER_USING_LABELED_SET($\bar{\mathcal{F}}_1$)
```

Algorithm 3 Co-stimulation based misbehavior detection at node s_i

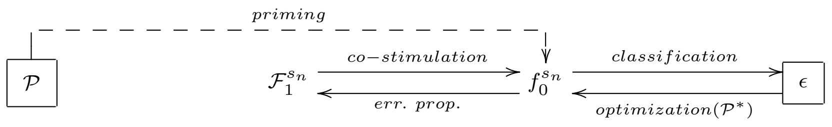
```
Require: Sufficient data traffic in current time window at node $s_i$
Require: $\hat{f}_1^{s_{i+2}}$ from 2-hop downstream node
Require: Set of (optimized) priming thresholds $\mathcal{P}$
 1: procedure DETECT_MISBEHAVIOR
 2:     $\hat{f}_1^{s_i} \leftarrow$ COMPUTE_$f_1$_FEATURE_SAMPLE($s_i$)
 3:     SEND_UPSTREAM($\hat{f}_1^{s_i}$)
 4:     $\hat{\mathcal{F}}_1 \leftarrow \hat{f}_1^{s_i} \circ \hat{f}_1^{s_{i+2}}$
 5:     if CLASSIFICATION($\hat{\mathcal{F}}_1$) == misbehavior then
 6:         suspicious ← True
 7:     end if
 8:     if suspicious == True then                 ▷ Co-stimulation: $\mathcal{F}_1 \rightarrow f_0$
 9:         $\hat{f}_0 \leftarrow$ COMPUTE_$f_0$_FEATURE_SAMPLE($s_i$)
10:         if $\hat{f}_0$ violates $\mathcal{P}$ then
11:             MARK_AS_MISBEHAVING($s_{i+1}$)            ▷ Misbehavior detected
12:         end if
13:     end if
14: end procedure
```

6.5 Error Propagation Algorithm for Ad Hoc Networks

Co-stimulation, as presented in the previous sections, requires that two distinct $\mathcal{F}_1$ and f_0 based classifiers get computed. This implies that two sets of $\hat{\mathcal{F}}_1$ and $\hat{f}_0$ feature vectors, labeled with the classes under consideration, must be available for training. Next, we discuss how this can be avoided.



(a) An approach with error propagation and co-stimulation.



(b) An approach extended with optimization.

Fig. 2 Error propagation algorithm.

In this section, we delve into the mechanisms of error propagation. Error propagation is the opposite of co-stimulation, i.e. in a short form it can be expressed as:

$$\mathcal{F}_1 \xleftarrow{error\ propagation} f_0 \tag{15}$$

Eq. 4 states that for *win. size* $\to 0$, the $\mathcal{F}_1$ and f_0 based classification approaches offer the same classification accuracy. This implies, if $\epsilon^{\Omega}(f_0) = 0$, for the $\hat{\mathcal{F}}_1$ and $\hat{f}_0$ feature vectors stemming from the same time window, the classification outcome will be (nearly) the same. This motivates the following strategy: compute $\hat{\mathcal{F}}_1$ and $\hat{f}_0$ feature vectors, classify the $\hat{f}_0$ feature vector according to a predefined threshold, and then, label the $\hat{\mathcal{F}}_1$ feature vector with the same label as the $\hat{f}_0$ based vector. This allows us to build a labeled $\hat{\mathcal{F}}_1$ based feature vector set necessary for the computation of an $\mathcal{F}_1$ based classifier.

The application of such thresholds is hereafter referred to as *priming.* The general goal of priming is to introduce a well-defined level of consistency, when detecting misbehavior in an ad hoc network. With respect to the above said, we define misbehavior as a violation of the priming thresholds $\mathcal{P} = \{p_1, p_2, ..., p_l\}$, where l is the number of priming thresholds. A priming threshold can be for example the maximum allowed data packet loss at a node or the maximum allowed data packet processing delay at a node. If any priming threshold is violated, the corresponding $\hat{\mathcal{F}}_1$ and $\hat{f}_0$ feature vectors will be labeled as representing misbehavior.

In order to achieve a good level of energy efficiency, it is desirable to apply *win. size* $\gg 0$. Since however a larger time window size introduces a loss in misbehavior detection precision, error propagation must be followed by co-stimulation:

$$\begin{tikzcd} \mathcal{F}_1 \arrow[r, shift left, "co-stimulation"] & f_0 \arrow[l, shift left, "error\ propagation"] \arrow[r] & \epsilon \\ & & \mathcal{P} \arrow[ul] \end{tikzcd} \tag{16}$$

Co-stimulation is achieved by computing a fresh $\hat{f}_0$ feature vector and comparing it with $\mathcal{P}$. The initialization phase of this approach is formally presented in Alg. 2. The detection phase is presented in Alg. 3. This approach is also schematically depicted in Fig. 2(a).

Error propagation and co-stimulation is executed within the same node; see Fig. 3. Notice that some nodes in the example network, for the shown data flows, are unable to compute $\hat{\mathcal{F}}_1$ since they do not have any two-hop neighbor s_{i+2}. On the other hand, several nodes receive multiple $\hat{f}_1^{s_{i+2}}$, e.g. s_1 from s_3 and s_5.

Let us now formulate the effects of these two phases in more detail. Let us assume that $win.\ size \gg 0$. For simplicity, let us also assume that $\epsilon^{\Omega}(f_0) = 0$.

1. *After the error propagation phase:* it holds that $\epsilon^{\Omega}(\mathcal{F}_1) > 0$, i.e. the precision of $\mathcal{F}_1$ based classification is for $win.\ size \gg 0$ lower than the precision of f_0 based classification. This is a direct consequence of the property expressed in Eq. 5. $\epsilon^{\Omega}(\mathcal{F}_1) > 0$ implies that there exists a class c_j for which at least one of the following holds:

$$det.\ rate_{c_j}^{\Omega}(\mathcal{F}_1) = a < 100\%$$

$$FP\ rate_{c_j}^{\Omega}(\mathcal{F}_1) = b > 0\%$$

2. *After the co-stimulation phase:* co-stimulation applied after error propagation decreases the FP rate of the class c_j to zero, while keeping the detection rate unchanged:

$$det.\ rate_{c_j}^{\Omega}(\mathcal{F}_1 \rightarrow f_0) = a$$

$$FP\ rate_{c_j}^{\Omega}(\mathcal{F}_1 \rightarrow f_0) = 0\%$$

This is a direct consequence of the properties expressed in Eqs. 8 and 9.

Since to achieve $\epsilon^{\Omega}(f_0) = 0$ may not be feasible, Eqs. 10 and 11 apply. This means that the final FP rate for a class c_j is determined by $FP\ rate_{c_j}^{\Omega_{\mathcal{F}_1}}(f_0)$. Similarly, the final detection rate has an upper bound equal to $det.\ rate_{c_j}^{\Omega}(\mathcal{F}_1)$. Notice that $win.\ size$ also determines the time delay for detecting a misbehavior, i.e. it should reflect the requirements prescribed for the given misbehavior detection system.

6.6 Error Propagation Algorithm with Optimization

The error propagation algorithm can be extended with an optimization phase; see Fig. 2(b). With respect to the classification outcome, the individual priming threshold values for each node can be optimized, i.e. the classification error can be minimized. This can be done by a repeated application of the error propagation and co-stimulation phases, while adjusting the priming thresholds for each node, until a termination condition is met; see Fig. 2. This approach can be described as priming with influences of e.g. noise being considered locally. More formally, new optimized priming thresholds

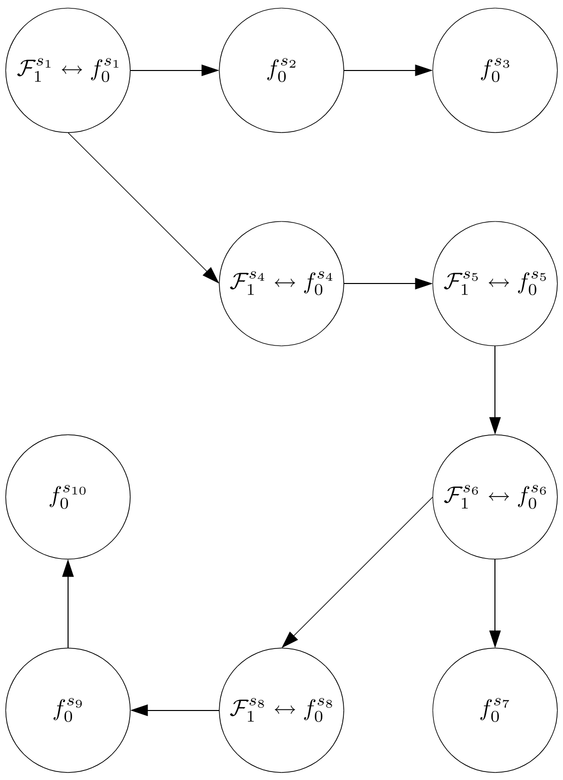

$\mathcal{F}_1^{s_1} \leftrightarrow f_0^{s_1}$
$f_0^{s_2}$
$f_0^{s_3}$
$\mathcal{F}_1^{s_4} \leftrightarrow f_0^{s_4}$
$\mathcal{F}_1^{s_5} \leftrightarrow f_0^{s_5}$
$f_0^{s_{10}}$
$\mathcal{F}_1^{s_6} \leftrightarrow f_0^{s_6}$
$f_0^{s_9}$
$\mathcal{F}_1^{s_8} \leftrightarrow f_0^{s_8}$
$f_0^{s_7}$

Fig. 3 A 10-node ad hoc network with priming. s_1 is the only data flow source node. s_3, s_7 and s_{10} are sink nodes for three distinct data flows.

$\mathcal{P}^* = \{p_1^*, p_2^*, ..., p_l^*\}$ for each node s_i will be found, so that the final classification error for each node s_i is minimized. The repeated application of the error propagation and co-stimulation phases is inspired by the feedback loop between the innate and adaptive immune systems as discussed in Section 2.

Our extended approach bears a certain similarity to the *backpropagation algorithm* for artificial neural networks (Alpaydin 2004). The backpropagation algorithm takes advantage of two steps, feed-forward and error backpropagation. These two steps are repeated in order to minimize an error function. A notable difference between our extended approach and the backpropagation algorithm isthat the co-stimulation phase is designed to revert the FP rate to the levels before any error propagation was done. Since the FP rate is expected to stay unchanged, the search for suitable values for priming thresholds is less complex.

7 Experimental Setup

Even though, we were able to derive the basic rules governing the efficiency of co-stimulation and priming (see Eqs. 10 and 11), an experimental analysis is necessary in order to offer a quantitative insight into their efficiency. Therefore, we designed several experiments based on network simulation using the JiST/SWANS network simulator (Barr et al 2005).

JiST/SWANS was chosen over other alternative such as ns2 due to its better scalability with the experiment size, both in terms of event processing speed and memory overhead (Barr et al 2005). JiST/SWANS is about two orders of magnitude faster than ns2 with Tcl. The other considered alternative was Glomosim (Bajaj et al 1999). We opted not to use Glomosim since its development stopped in 2001, when its commercial version Qualnet (Scalable Networks 2010) was introduced. The JiST/SWANS deficiency, that we faced, was its simplified physical layer implementation, most notably JiST/SWANS does not implement any modulation schemes.

JiST/SWANS does not offer any ready-to-use misbehavior implementation. Therefore we had to implement the three considered misbehavior types. Additionally, it was necessary to increase the number of packet events that get logged. This resulted in large packet event logs, therefore it was also necessary to implement an on-the-fly data compression. For this purpose we used the BZIP2 library (Seward 2010).

Performance evaluation was based upon a standard machine learning experiment, where the classification efficiency is estimated using stratified k-fold cross-validation (Alpaydin 2004), in our case with $k = 20$. The classification based on $\mathcal{F}_1$ was done using a decision tree classifier; see Alg. 2, line 20. To decide whether a node within the decision tree should be further split (impurity measure), we used the information gain measure. As the decision tree classifier is a well-known algorithm, we omit its discussion. We refer the interested reader to (Alpaydin 2004). We used the decision tree implementation from the Rapidminer tool (Mierswa et al 2006).

Experiments: The misbehavior detection performance of our approach depends on $\epsilon^{\Omega}(f_0)$, the classification error of f_0 based classification. This is due to the fact that f_0 based classification provides a basis for the final decision about whether a node misbehaves. It is also used in the initial phase to correctly label the $\mathcal{F}_1$ based feature vectors. Due to its importance, we decided to use $\epsilon^{\Omega}(f_0)$ as one of the input parameters in our performance evaluation. We consider three levels of $\epsilon^{\Omega}(f_0)$: *level*$_1$, *level*$_2$ and *zero*. Our goal is to have $\epsilon^{\Omega}(f_0)$ for *level*$_2$ equal or greater than $\epsilon^{\Omega}(f_0)$ for *level*$_1$. Unfortunately, it is only possible to set $\epsilon^{\Omega}(f_0)$ indirectly. This can be done by identifying three scenarios with three different levels of misbehavior severity, each scenario yielding a substantially different $\epsilon^{\Omega}(f_0)$.

We executed three distinct experiments with the following values for α, β and δ; see Section 5:

- Experiment 1: $\epsilon^{\Omega}(f_0) = level_1$ with $\alpha_1 = 30\%$, $\beta_1 = 30\%$ and $\delta_1 = 100ms$.
- Experiment 2: $\epsilon^{\Omega}(f_0) = level_2$ with $\alpha_2 = 10\%$, $\beta_2 = 30\%$ and $\delta_2 = 20ms$.
- Experiment 3: $\epsilon^{\Omega}(f_0) = 0$. The same parameters as in Experiment 2 were applied, however, $\epsilon^{\Omega}(f_0)$ was set to 0. This was done by labeling all vectors according to the priming thresholds $\mathcal{P}$, i.e. if a threshold was violated, the vector was labeled as representing an undesired operational state and vice-versa.

To put the applied parameters α, β, δ into perspective, the data packet dropping rate under "normal" traffic was $0.01 \pm 0.18\%$. The data packet delaying rate under "normal" traffic was 1.90 ± 0.48 ms.

The rationale for the first experiment was to consider conditions with a lower classification complexity, so that an upper bound on the energy efficiency of our approach can be established. The second experiment was designed to gain some insight into how

	Experiment 1		
Win. size [s]	Total	Normal	Dropping&Delaying
50	6847.85±142.90	4603.75±88.94	2200.50±66.20
100	3477.15±72.37	2340.15±44.95	1115.25±33.68
250	1465.80±30.87	985.15±19.21	471.25±14.20
500	785.40±15.87	530.00±9.91	250.45±7.50
	Experiment 2		
Win. size [s]	Total	Normal	Dropping&Delaying
50	6238.30±118.96	4385.30±85.14	1812.20±46.62
100	3012.90±57.77	2124.80±41.57	868.20±22.67
250	1078.75±20.65	767.20±15.07	304.70±7.96
500	444.75±8.73	323.85±6.66	118.20±3.19
	Experiment 3		
Win. size [s]	Total	Normal	Misbehavior
50	6238.30±118.96	4498.10±90.11	1740.20±44.31
100	3012.90±57.77	2153.20±43.62	859.70±22.38
250	1078.75±20.65	778.60±15.64	300.15±7.96
500	444.75±8.73	328.50±6.89	116.25±3.27

Table 1 Sample set size for Experiment 1-3.

our approach performs under less severe misbehavior levels, i.e. when the difference between normal behavior and misbehavior is less pronounced. The aim of the last experiment was to offer an estimate on detection performance when the goal is to detect a violation of the priming thresholds $\mathcal{P}$ without assuming that such a violation must be the result of a misbehavior. It also offers a quantitative insight into Eqs. 8 and 9.

$M3_D$ and $M4_D$ Priming Thresholds: In each of the three experiments, we used two priming thresholds $\mathcal{P} = \{M3_D, M4_D\}$. In Experiment 1 we set M3_D to 97.5% and M4_D to 4ms. In Experiment 2 and Experiment 3 we set M3_D to 99.5% and M4_D to 2.6ms. These values were found through the extended approach shown in Fig. 2(b) using the "trial and error" method (as a substitute for a formal optimization approach). Notice that we apply identical priming thresholds to all nodes. Allowing for node specific priming thresholds could lead to an improved detection performance of our priming approach.

Misbehavior: There were 236 nodes randomly chosen to execute data dropping or delaying misbehavior. Our intention was to model *random failure occurrences*, assuming a uniform failure distribution in the network. As it is hard to predict the routing of data packets, many of these nodes could not execute any misbehavior as there were no data packets to be forwarded by them. In our case, about 20-30 misbehaving nodes were concurrently active.

There were 20 wormholes simulated; each wormhole was designed to bypass 15 hops, i.e. the source and sink were 15 hops away (with respect to the unmodified topology before any wormhole was added). We considered this number of wormholes largely due to the necessity to obtain enough data for a later statistical analysis of the experimental results.

Simulation Setup: We did 20 independent simulation runs for each misbehavior type and 20 misbehavior free (normal) simulation runs. The simulation time for each run was 4 hours. We used a non-overlapping time window approach for features' computation. We used four different time window sizes: 50, 100, 250 and 500 seconds. In the case of a 500-second time window, there were 28 non-overlapping windows in each run (4 hours/500 seconds = 28.8). This gave us $4 \times 20 \times 28 = 2{,}240$ vectors (samples) for each node. This also determined the max. target sample set size for the error propagation algorithm.

Alg. 2 and 3 require "sufficient" data traffic in a given time window. This is a technical requirement aimed at avoiding time windows with no or very little data traffic. We only considered time windows with the data traffic equal or larger than $\frac{\eta}{\alpha} \times \frac{win.\ size}{50}$, where η is the minimum expected number of data packets to be dropped in a 50-second time window. We considered $\eta = 2.0$, i.e. the minimum number of data packets in a 50-second time window was 6.66 and 20 for $\alpha = 0.3$ and $\alpha = 0.1$, respectively. To simplify the experiments we only considered 20 nodes with the highest amount of data traffic. The resulting sample set size per node is reported in Table 1.

The complexity of our experimental setup can thus be summarized as follows: 3 distinct experiments, 20 independent simulation runs, 3 misbehavior types and 1 normal traffic behavior, and 4 time window sizes. Additionally, when estimating the classification efficiency of our approach, k-fold cross-validation with $k = 20$ was done. Due to the experimental setup complexity, we only considered a single network topology, a single model for data packet injection and a fixed number of data connections. The used hardware was 20× Linux (SuSE 11.2) PC with 2GB RAM and a Pentium 4 3GHz microprocessor.

Topology, Connections, Data Traffic and Protocols: We used a topology based on a snapshot from the movement prescribed by the Random waypoint movement model (Johnson and Maltz 1996). There were 1,718 nodes simulated; the physical area size was 3,000m × 3,000m. A topology of this size was necessary since we needed to accommodate 20 wormholes. We used this specific topology because it had been quite extensively studied by Barrett et al. in (Barrett et al 2005); the results reported therein include many graph-theoretical measures that were helpful in finding suitable parameters for our experiments.

We modeled data traffic as Constant bit rate (CBR), i.e. there was a constant delay when injecting data packets. This constant delay in our experiments was 2 seconds (injection rate of 0.5 packet/s); the packet size was 68 bytes. CBR data packet sources correspond to e.g. sensors that transmit their measurements in predefined constant intervals. CBR can be considered due to its synchronized nature an extreme model for data packet injection. In fact, the results published in (Schaust and Drozda 2008) show that when using a stochastic injection model, such as the Poisson traffic model, one can expect a better performance of the misbehavior detection system.

In our simulations we used 50 concurrent data connections. The connection length was 7 hops. In order to represent a dynamically changing system, we allowed connections to expire. An expired connection was replaced by another connection starting at a new random source node. Each connection was scheduled to exist 15 to 20 minutes. The exact connection duration was computed as

$$\tau + r_U \lambda \tag{17}$$

where τ the minimum duration time of a connection, r_U a random number from the uniform distribution $[0, 1]$ and λ the desired variance of the connection duration. In our experiments, we used $\tau = 15$ *min* and $\lambda = 5$ *min*.

We used the AODV routing protocol. We used this protocol with JiST/SWANS default settings. Most notably, the net diameter was set to 19 hops and the routes in any routing cache were not allowed to expire. Since in our approach to misbehavior

	Normal		Dropping&Delaying		Any Misbehavior	
Win. size [s]	Det. rate	$CI_{95\%}$	Det. rate	$CI_{95\%}$	Det. rate	$CI_{95\%}$
	f_0					
50	97.23	0.67	99.10	0.25	99.10	0.25
100	97.37	0.66	99.62	0.18	99.62	0.18
250	97.34	0.69	99.83	0.14	99.83	0.14
500	97.38	0.70	99.91	0.11	99.91	0.11
	$\mathcal{F}_1$					
50	94.68	0.98	95.56	0.88	95.44	0.91
100	93.70	1.02	94.37	0.81	94.40	0.81
250	91.06	1.12	89.65	1.33	89.84	1.38
500	88.47	1.35	85.39	2.25	85.64	2.30
	$\mathcal{F}_1 \rightarrow f_0$					
50	94.38	1.01	94.72	0.97	94.57	1.02
100	93.57	1.06	93.55	0.97	93.52	0.98
250	89.08	1.31	88.29	1.71	88.29	1.76
500	88.43	1.36	83.42	2.55	83.60	2.50

Table 2 Experiment 1: Detection rate.

	Normal		Dropping&Delaying		Any Misbehavior	
Win. size [s]	FP rate	$CI_{95\%}$	FP rate	$CI_{95\%}$	FP rate	$CI_{95\%}$
	f_0					
50	0.41	0.13	4.15	1.29	5.96	1.32
100	0.17	0.10	3.88	1.27	5.69	1.30
250	0.08	0.06	3.87	1.34	5.74	1.35
500	0.05	0.06	3.85	1.43	5.73	1.36
	$\mathcal{F}_1$					
50	2.04	0.48	9.97	1.69	11.55	1.83
100	2.43	0.40	11.92	1.85	13.85	1.95
250	4.41	0.78	18.44	1.88	19.74	2.19
500	5.85	0.89	24.95	2.91	26.04	2.70
	$\mathcal{F}_1 \rightarrow f_0$					
50	0.10	0.07	3.37	1.16	5.16	1.24
100	0.04	0.04	3.10	1.07	4.92	1.18
250	0.03	0.03	2.64	0.86	4.35	1.08
500	0.01	0.02	2.15	0.89	4.00	1.08

Table 3 Experiment 1: FP rate.

detection, we only look at a two-hop segment of a connection, a variety of other routing protocols could have been used instead. For example, the DSR protocol (Johnson and Maltz 1996) could have been equally well used.

We used the IEEE 802.11 MAC protocol. The RTS-CTS-DATA-ACK handshake was enabled for all data communication. This minimized the influence of the hidden terminal phenomenon on the overall performance and thus simplified the evaluation of experimental results. We used UDP transport protocol and IPv4. The channel frequency was set to 2.4 GHz. The transmission rate was set to 54 Mbit/s. We used the two-ray signal propagation model (Rappaport 2001). Antenna and signal propagation properties were set so that the resulting radio radius equals 100 meters.

	Normal		Dropping&Delaying		Any Misbehavior	
Win. size [s]	Det. rate	$CI_{95\%}$	Det. rate	$CI_{95\%}$	Det. rate	$CI_{95\%}$
			f_0			
50	95.88	0.73	89.59	1.62	95.60	1.00
100	97.49	0.70	96.79	1.00	96.79	1.00
250	97.95	0.62	98.40	0.90	98.40	0.90
500	98.15	0.54	98.74	1.00	98.74	1.00
			$\mathcal{F}_1$			
50	92.97	1.30	88.50	1.33	88.77	1.25
100	92.85	1.35	86.79	1.66	86.56	1.64
250	90.22	1.46	81.38	1.78	82.00	1.69
500	88.35	0.99	73.40	2.52	73.49	2.34
			$\mathcal{F}_1 \rightarrow f_0$			
50	91.84	1.51	88.12	1.47	88.77	1.25
100	91.99	1.59	85.69	1.69	85.35	1.71
250	89.80	1.64	80.49	1.88	80.94	1.79
500	87.95	1.16	72.21	2.66	71.64	2.57

Table 4 Experiment 2: Detection rate.

	Normal		Dropping&Delaying		Any Misbehavior	
Win. size [s]	FP rate	$CI_{95\%}$	FP rate	$CI_{95\%}$	FP rate	$CI_{95\%}$
			f_0			
50	1.76	0.48	14.46	3.36	10.01	1.64
100	1.34	0.49	4.12	1.40	6.13	1.59
250	0.67	0.45	3.13	1.16	5.17	1.41
500	0.50	0.47	3.10	1.34	5.18	1.52
			$\mathcal{F}_1$			
50	4.36	0.51	15.27	3.06	17.23	2.96
100	5.23	0.60	16.19	2.98	17.69	3.08
250	6.56	0.69	23.72	3.51	25.38	3.62
500	8.83	0.80	32.12	4.17	33.20	3.70
			$\mathcal{F}_1 \rightarrow f_0$			
50	0.59	0.18	15.09	3.72	6.42	1.40
100	0.44	0.18	3.10	1.13	5.01	1.35
250	0.23	0.16	2.34	0.84	4.26	1.24
500	0.09	0.10	2.09	0.99	3.57	1.25

Table 5 Experiment 2: FP rate.

8 Performance Evaluation

8.1 Misbehavior Detection Performance

The results achieved by applying the error propagation algorithm in Experiment 1 and Experiment 2 are reported in Tables 2-5. The results for the three misbehavior classes are reported separately for the packet dropping and delaying classes (primed misbehavior classes) as well as for all three misbehavior classes together. For the sake of thoroughness, we also report the results after co-stimulation ($\mathcal{F}_1 \rightarrow f_0$) for the normal class. Notice that our approach enters the co-stimulation phase, only if $\mathcal{F}_1$ based classification detects a possible misbehavior; see Alg. 3, line 8.

Excluding the results for Experiment 2 with 50-second time window, it can be seen that as the time window size decreases, the performance of f_0 and $\mathcal{F}_1$ based classification becomes more and more comparable. This is expected and in-line with Eq. 4. It can also be seen that the final detection rate after co-stimulation is in-line with Eq. 11. The final FP rate is in-line with Eq. 10 or lower. This is an important observation pointing out that in our case the task of classifying $\Omega_{\mathcal{F}_1}$ was no more complex than the task of classifying Ω, i.e. Eq. 12 can be reformulated with respect to our results as follows:

$$FP\ rate_{c_j}^{\Omega_{\mathcal{F}_1}}(f_0) \le FP\ rate_{c_j}^{\Omega}(f_0) \tag{18}$$

The f_0 based classification error $\epsilon^{\Omega}(f_0)$ is reported in Table 6. It is in the range of $1.85 - 2.19\%$ and $1.73 - 4.17\%$ for Experiment 1 and Experiment 2, respectively. It is equal to the fraction of vectors that were incorrectly labeled after the priming thresholds $\mathcal{P}$ were applied in the initial phase of our algorithm (Alg. 2, lines 15 and 17). This did not only decrease the precision of labeling in the initial phase, but it also decreased the efficiency of co-stimulation (Alg. 3, line 10). This fact especially negatively impacted the results for Experiment 2 with 50-second time window. This problem could be less pronounced, if priming thresholds optimized locally with respect to each node were used.

The final FP rate for "normal" behavior could be significantly decreased in comparison to $FP\ rate(\mathcal{F}_1)$. It is in the range of $0.02-0.07\%$ and $0.10-0.18\%$ for Experiment 1 and Experiment 2, respectively. The FP rate for the two primed misbehavior classes is in the range $2.15 - 3.37\%$ and $2.09 - 15.09\%$ for Experiment 1 and Experiment 2, respectively. The high FP rate value for Experiment 2 and 50-second time windows is connected with $\epsilon^{\Omega}(f_0)$ being notable higher than in the other cases. The performance gap, in terms of FP rate, between the "normal" class and the two primed classes originates from the fact that learning "normal" is simpler than learning "misbehavior". In the latter case, samples which belong to "normal" are often misinterpreted as representing misbehavior.

The performance difference between the two primed classes and the "any misbehavior" class is the result of two phenomena: i) there was no priming applied with respect to the wormhole misbehavior, and ii) the number of vectors representing the wormhole misbehavior is smaller compared to data dropping and delaying. The latter argument is connected with the fact that the number of wormholes in a network must not exceed a certain limit, otherwise the induced topological changes become extreme.

The results for Experiment 3 are reported in Table 7. It can be seen that they are in-line with Eqs. 8 and 9. Most notably, it can be seen that the final FP rate is as expected 0%.

In Section 6.4 we pointed out that co-stimulation introduces an implicit class of "undecided" vectors. These are the vectors that were marked as suspicious after $\mathcal{F}_1$ based classification, however, this prediction could not be confirmed by the subsequent f_0 based classification. There are two cases possible: i) a false positive was not confirmed by the f_0 based classification and ii) a true positive was not confirmed by the f_0 based classification. The first case represents a desirable phenomenon since it decreases the false alarm rate. The other case decreases the final detection rate. The number of vectors that become "undecided" in the latter case can be estimated as follows:

$$\kappa = (det.\ rate_{c_j}(\mathcal{F}_1) - det.\ rate_{c_j}(\mathcal{F}_1 \to f_0)) \times sample\_size_{c_j}$$

where $sample\_size_{c_j}$ is the sample set size for a given class c_j as reported in Table 1. For example, for Experiment 1, $win.\ size = 100s$ and the combined dropping and

Win. size [s]	Experiment 1	Experiment 2
50	2.19±0.11	4.17±0.17
100	1.93±0.11	2.72±0.16
250	1.89±0.11	1.94±0.15
500	1.85±0.11	1.73±0.14

Table 6 $\epsilon^{\Omega}(f_0)$ for Experiment 1 and 2.

	Normal		Any Misbehavior	
Win. size [s]	Det. rate	$CI_{95\%}$	Det. rate	$CI_{95\%}$
	$\mathcal{F}_1$ or $\mathcal{F}_1 \rightarrow f_0$			
50	94.50	1.04	86.39	1.20
100	94.03	1.14	86.79	1.53
250	91.19	1.26	81.25	2.06
500	89.36	1.06	72.47	2.88

	FP rate	$CI_{95\%}$	FP rate	$CI_{95\%}$
	$\mathcal{F}_1$			
50	5.17	0.50	14.36	2.54
100	5.08	0.59	15.22	2.60
250	6.62	0.71	23.92	3.38
500	8.99	0.79	31.35	3.32
	$\mathcal{F}_1 \rightarrow f_0$			
50-500	0.00 ± 0.00%			

Table 7 Experiment 3: Detection rate and FP rate.

delaying class, $\kappa = (94.37 - 93.55) \times 2340.15 = 19.19$ vectors or 0.55% of the sample set size for this class. For the same time window size and class, the corresponding rate in Experiment 2 equals 0.78%. Notice that in Experiment 3, since $det.\ rate_{c_j}(\mathcal{F}_1) = det.\ rate_{c_j}(\mathcal{F}_1 \rightarrow f_0)$, κ equals 0. This is expected and in-line with Eq. 9. Notice also that κ reflects the cost of co-stimulation, i.e. the cost of removing false positives through co-stimulation.

8.2 Energy Consumption Analysis

When detecting a misbehavior, the more costly f_0 based detection will only get used, if (i) a true positive was detected by $\mathcal{F}_1$ or (ii) a false positive was mistakenly detected by $\mathcal{F}_1$. This means, for a misbehavior free ad hoc network, the energy saving over an exclusive f_0 approach is related to $FP\ rate_{mis}(\mathcal{F}_1)$, the rate at which f_0 based misbehavior detection is mistakenly applied. We focus on the energy efficiency analysis in a misbehavior free ad hoc network, since it is reasonable to assume that an ad hoc network will work reliably, most of the time. The energy consumption can be thus modeled as:

Let us first derive the models for the energy consumption with respect to f_0 and $\mathcal{F}_1$ based misbehavior detection. The energy cost $\xi_t(n)$ related to receiving n bytes can be expressed as:

$$\xi_r(n) = n \times data\_rate^{-1} \times V_{cc} \times I_r \tag{19}$$

where $data\_rate$ is the data rate in $bytes/s$, V_{cc} is the supply voltage of the wireless device in V and I_r is the current drawn while receiving in A. Similarly, the energy cost $\xi_t(n)$ related to transmitting n bytes can be expressed as follows:

$$\xi_t(n) = n \times data\_rate^{-1} \times V_{cc} \times I_t \tag{20}$$

where I_t is the current drawn while the device is in receive mode.

The f_0 based detection relies on data packet overhearing. The energy cost $\xi_{f_0}(t)$ when f_0 based detection is applied for t_0 seconds can be expressed as:

$$\xi_{f_0}(t_0) = \xi_r(t_0 \times inj\_rate \times size(data)) \tag{21}$$

where $size(data)$ is the data packet size in bytes and inj_rate is the injection rate in s^{-1}. Eq. 21 reflects the fact that the energy consumption in promiscuous mode grows linearly with the number and size of the overheard data packets. We remind the reader that we considered $inj\_rate = 0.5s^{-1}$ in our experiments. The energy cost of $\mathcal{F}_1$ based detection applied for t seconds can be expressed as:

$$\xi_{\mathcal{F}_1}(t) = \lfloor \frac{t}{win.\ size} \rfloor \times 2 \times \left(\xi_t(size(\hat{f}_1)) + \xi_r(size(\hat{f}_1))\right) \tag{22}$$

We assume that 24 2-byte features are being transported in $\hat{f}_1$. We thus assume $size(\hat{f}_1) = 48 + size(header)$ bytes, where $size(header)$ is the data packet header size. Eq. 22 reflects the cost of sending $\hat{f}_1$ from the node s_{i+2} over two hops to the node s_i.

As we stated above, the f_0 based misbehavior detection in a misbehavior free network is applied with the probability $FP\ rate_{mis}(\mathcal{F}_1)$. The total energy cost of our approach being applied for t seconds can be expressed for $t \geq win.\ size$ as:

$$\xi(t) = \xi_{\mathcal{F}_1}(t) + FP\ rate_{mis}(\mathcal{F}_1) \times \xi'_{f_0}(t) \tag{23}$$

where

$$\xi'_{f_0}(t) = \lfloor \frac{t - win.\ size}{win.\ size} \rfloor \times \xi_{f_0}(t_0) + \xi_{f_0}(\mathbf{min}\{t \ \mathbf{mod}\ win.\ size, t_0\}) \tag{24}$$

t_0 is the duration of f_0 based classification. In our experiments, we set $t_0 = 50s$. Under our data traffic model, if s_i forwards data packets for a single connection, 25 data packets get overheard in promiscuous mode. There is no energy consumed for $t < win.\ size$, therefore we omit this case. The energy cost model formulated in Eq. 23 assumes that feature computation and decision tree query costs are negligible, compared to communication costs. This is a reasonable simplifying assumption, since the maximum depth of the decision tree is five. Zhao et al. (Zhao et al 2003) reported that for Sensoria sensors and Berkeley motes the ratio between the energy consumption related to communication and computation is in the range of $10^3 - 10^4$.

In order to present a quantitative energy cost analysis, it is necessary to consider a specific wireless device. We chose an IEEE 802.15.4 compliant device and two IEEE 802.11 WLAN devices. To make the results for these devices comparable, we apply an estimate of MAC Layer overhead and adjust the expected data rate for each device accordingly. Our choice are the following three devices, each requiring a 3.3V DC power supply:

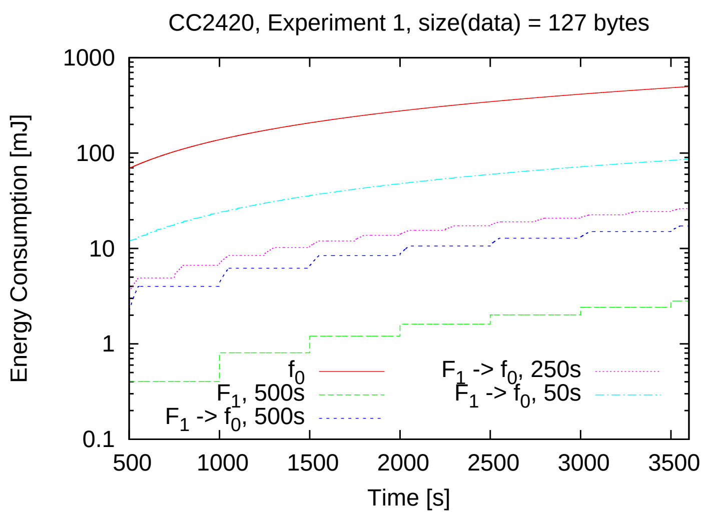


Fig. 4 TI CC2420 energy consumption for $win.\ size = \{50s, 250s, 500s\}$.

- *TI C2420* (Texas Instruments 2007) is a widespread IEEE 802.15.4 compliant transceiver. TI CC2420 supports data packet sizes up to 127 bytes. We assume the use of short addresses and PAN ID compression[1], i.e. 11 bytes are needed for the MAC header (IEEE Std. 802.15.4 2003). This translates to an effective data rate of 228kbit/s, i.e. in our computations we set $data_rate = 228$kbit/s. TI C2420 consumes 18.8mA when in receive mode and 17.4mA when in transmit mode.
- *Wistron CM9 miniPCI card* (Wistron NeWeb Corp. 2010) is an example for a consumer grade WiFi device. Wistron CM9 is based on the Atheros AR5213 chipset. It is compliant with IEEE 802.11a/b/g. When operating in 54Mbit 802.11g mode, it consumes 410mA when transmitting and 310mA when receiving data. For a 54Mbit connection we assume an effective data rate of 22Mbit/s (Ahlers 2009).
- *Ubiquiti XR2 card* (Ubiquiti Networks 2010) is an Atheros AR5414 based long-range WiFi device. Ubiquiti XR2 consumes 900mA when transmitting and 300mA when receiving in 802.11g 54Mbit mode. Similarly as for the previous wireless device, we assume an effective data rate of 22Mbit/s.

Considering these three wireless devices, we investigated our priming approach using $FP\ rate_{mis}(\mathcal{F}_1)$ for "any misbehavior" from Experiment 1 and 2; see Tables 3 and 5. We assumed that $\hat{f}_1^{s_{i+2}}$ is transmitted as a standalone data packet. Should some other approach be used, e.g. piggybacking on CTS/ACK packets, the associated energy cost could be lower.

Fig. 4 shows the energy cost of TI CC2420 when transmitting 127-byte data packets assuming the conditions of Experiment 1. For 500-second time window, the energy cost reduction is 96.8%, if compared to an exclusive use of f_0 based misbehavior detection. More specifically, f_0 based detection consumes $69.11mJ$, whereas our priming approach consumes $2.20mJ$. For the smaller 50-second time window, the energy cost reduction

[1] PAN ID = Personal Area Network ID.

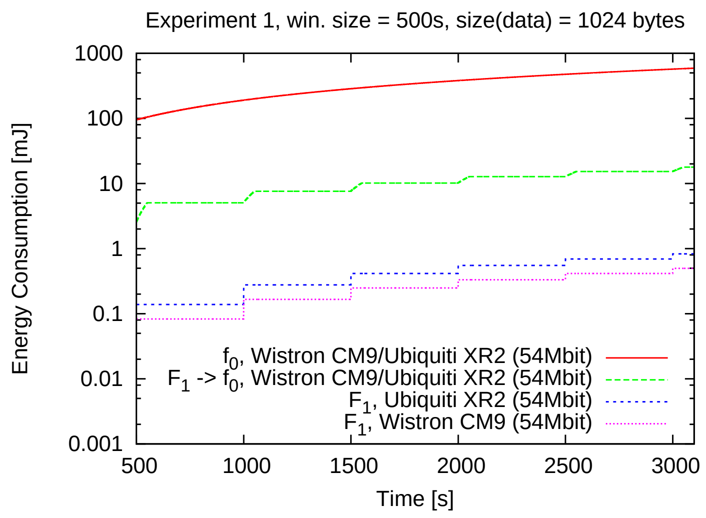


Fig. 5 Wistron CM9 and Ubiquiti XR2 energy consumption.

is 82.6%, with $1.20mJ$ per time window consumed by our approach and $6.91mJ$ by an exclusive f_0 based detection.

Under the conditions of Experiment 2, due to the higher $FP\ rate_{mis}(\mathcal{F}_1)$, the energy cost increases to $2.70mJ$ for 500-second time window and to $1.60mJ$ for 50-second time window, yielding an improvement of 96.1% and 76.9%, respectively, in comparison to an exclusive use of f_0 based misbehavior detection.

In Fig. 5 we report the results for the two WLAN wireless cards: Wistron CM9 and Ubiquiti XR2. We applied the FP rates obtained in Experiment 1. The time windows size was set to 500 seconds and the data packet size was set to 1024 bytes. The energy consumption reduction for Ubiquiti XR2 is 97.2%. f_0 based detection consumes $92.16mJ$ per time window, whereas our priming approach consumes $2.54mJ$ per time window. The energy consumption reduction for Wistron CM9 is 97.31%. f_0 based detection consumes $95.23mJ$ per time window, whereas our priming approach consumes $2.56mJ$ per time window. The difference in energy consumption for these two WLAN wireless cards is rather small. This is due to the fact that the energy consumption while receiving is almost identical. Additionally, the energy consumption while sending has only a limited impact, since the cost of $\mathcal{F}_1$ based classification is much lower than the cost of f_0 based classification.

Notice that the cost of sending a single $\hat{f}_1$ packet over two hops is small, if compared to overhearing data packets in promiscuous mode for the same time period; see Fig. 4 and compare the results for f_0 and $\mathcal{F}_1$. Therefore, in the initial phase of the error propagation algorithm, the necessity to compute $\hat{\mathcal{F}}_1$ increases the energy consumption only slightly.

9 Summary of Results and Research Challenges

Since co-stimulation and priming are approaches that until now received relatively little attention, we discuss the challenges connected with their application. We also summarize our key results.

1. *Independence of detection rate and FP rate:* The co-stimulation enables that the final detection rate and false positives rate are influenced by two distinct mechanisms: $\mathcal{F}_1$ and f_0 based classification. This fact is documented in Eqs. 10 and 11. This property allows for tuning the detection system with respect to these two measures in isolation. We showed that in scenarios, where the only goal is to detect whether a specific performance threshold was violated, it is feasible to achieve $\epsilon^{\Omega}(f_0) = 0$. For this reason, with respect to Eq. 8, it was possible to decrease the final false positives rate to zero; see Experiment 3 and the related results in Table 7.
2. *Energy efficiency:* The costly f_0 (watchdog) based classification is only applied if the more energy efficient $\mathcal{F}_1$ classification detects a misbehavior. In a misbehavior free ad hoc network, the frequency of f_0 based classification is decreased proportionally to $1 - FP\ rate_{mis}(\mathcal{F}_1)$. Under our experimental setup, the peak energy consumption reduction was 92.7% compared to the energy cost of exclusive f_0 based classification. Co-stimulation allows us to choose a *trade-off* between detection performance and energy efficiency. A higher level of energy efficiency can be achieved by increasing the time window size. This results in a lower detection rate. Increasing energy efficiency has however only a limited influence on the FP rate.
3. *Source of co-stimulation:* The co-stimulation approach due to Sarafijanović and Le Boudec (Sarafijanovic and Le Boudec 2004) benefits from the information about data packet delivery provided by TCP (Transmission Control Protocol). If a data packet is not delivered, then the connection source does not receive the corresponding packet acknowledgment. This can then be used as a co-stimulation for another form of data packet loss detection. The transport layer can thus serve as a potent source of co-stimulation for any lower layer detection mechanisms. This is however not always possible, since e.g. sensor networks are expected to operate with a reduced set of transport layer services. The reason for such a reduction is also the energy consumption related to transport layer services. The goal to increase energy efficiency of misbehavior detection was our motivation for investigating a distinctly different form of co-stimulation as discussed in Section 6.2.
4. *Co-stimulation avoiding watchdog (f_0) features:* Watchdog features can only be computed, if omnidirectional antennas are being used. Using a directional antenna by s_{i+1} could preclude any packet overhearing by s_i. A similar effect can be observed, if s_{i+1} is capable of dynamic radio radius adjustment. As Eq. 4 suggests, watchdog features can be substituted by $\mathcal{F}_1$ based features, if $win.\ size \to 0$. This implies, a co-stimulation based on $\mathcal{F}_1$ with a small window size could be a viable substitute for an f_0 based co-stimulation. For example, an $\mathcal{F}_1$ based classification with a 25-second time window could be used instead:

$$\mathcal{F}_1(500s) \xrightarrow{co-stimulation} \mathcal{F}_1(25s) \tag{25}$$

   This sort of co-stimulation would require an adaptive approach for requesting an $\hat{f}_1^{s_{i+2}}$ sample based on a smaller time window size. It also limits the option of negative co-stimulation; see Section 6.2. Negative co-stimulation is executed, if s_{i+1} does not cooperate in forwarding $\hat{f}_1^{s_{i+2}}$ feature vectors. This means, either f_0

based classification is reserved for negative co-stimulation or another mechanism is needed.

We next discuss a few challenges related to co-stimulation or priming:

1. *Convergence of the extended procedure with optimization:* In Section 6.6 we introduced priming extended with an optimization phase. This optimization phase allows for finding the optimal thresholds for $\mathcal{P}^*$. It is currently unclear, which optimization approach would suit best this purpose, i.e. delivering the best performance with respect to the optimization convergence.
2. *Minimizing the rate of undecided vectors:* Undecided vectors are the vectors that were after $\mathcal{F}_1$ based classification marked as representing a misbehavior, but this could not be confirmed by f_0 based classification. An especially severe case happens, if $\mathcal{F}_1$ based classification correctly classifies a vector as representing a misbehavior, but this classification result will get incorrectly rejected by f_0 based classification. One of the possibilities how the decrease the rate of undecided vectors is to present f_0 based classification with vectors that this classification approach can correctly classify with a high success rate. This implies that $\epsilon^{\Omega_{\mathcal{F}_1}}(f_0) \ll \epsilon^{\Omega}(f_0)$, i.e. the task to classify the vectors in $\Omega_{\mathcal{F}_1}$ is much less complex than the task to classify the vectors in Ω. The challenge is to tune $\mathcal{F}_1$ based classification in such a way that it outputs a vector set $\Omega_{\mathcal{F}_1}$ such that $\epsilon^{\Omega_{\mathcal{F}_1}}(f_0) \rightarrow 0$.

10 Conclusions

We proposed and evaluated a priming approach for misbehavior detection in ad hoc networks. The inspiration for our priming approach came from the BIS priming mechanisms that enable the activation of immune cells. Such activated immune cells are then capable of detecting a specific type of pathogen as well as triggering a specific type of immune reaction. Our general goal could be thus described as providing priming capabilities within an ad hoc network, so that a node can recognize whether a set of operational conditions is not violated. Such operational conditions may require that a certain minimum or maximum level of data packet delivery or data packet delay is achieved. Our priming approach takes advantage of co-stimulation. A key characteristic of co-stimulation is its ability to suppress FP rates as well as to stimulate energy efficiency. We showed that applying our priming approach, an improvement in energy efficiency of about 1.5 order of magnitude is possible in certain scenarios, if compared to a direct node monitoring using watchdog features.

Our additional goal was to show that our priming approach can provide a high degree of independence between the resulting detection and FP rates. The resulting false positives rate could be decreased to the same levels that were achieved by the extremely costly approach relying only on watchdog features. This allows for tuning the detection performance with respect to these two basic measures in isolation. Our approach also allows for choosing a trade-off between energy efficiency and misbehavior detection performance. This can be done through adjusting the time window size of the primary $\mathcal{F}_1$ based detection mechanism.

We pointed out that our priming approach can be extended with optimization. This allows for adjusting the priming thresholds $\mathcal{P}$ to local conditions. We also pointed out that the co-stimulation with f_0 based misbehavior classification can be substituted with $\mathcal{F}_1$ based classification with a small time window size. This is especially useful in environments benefiting from the use of directional antennas.

Even though, we concentrated on the applicability of co-stimulation and priming to misbehavior detection in ad hoc networks, we believe these methods are of general interest. An important characteristics of application scenarios, that could take advantage of these two methods, is the high cost connected with the relocation of information, services or resources, and at the same time, the feasibility to improve their control or allocation through one or several cost related parameters (such as time window size).

An example where co-stimulation and priming could be applicable is information broadcasting. When broadcasting information to users, the demand for any type of information can only be estimated (Vaidya and Hameed 1999). Since the users cannot send feedback directly to the broadcast source (e.g. satellite), the feedback is sent intermittently using other paths for delivery (often a mix of wireless and wired delivery paths). The feedback frequency can increase the accuracy, it is however connected with a certain cost. It thus gives sense to track the demand with some prediction accuracy exchanged for cost efficiency.

Acknowledgments

We would like to thank the reviewers for their very constructive and detailed comments. This work was supported by the German Research Foundation (DFG) under the grant no. SZ 51/24-3 (Survivable Ad Hoc Networks – SANE).

References

Ahlers E (2009) Funk-evolution. c't Magazin für Computer und Technik 13:86–89

Aickelin U, Bentley P, Cayzer S, Kim J, McLeod J (2003) Danger theory: The link between ais and ids? In: Timmis J, Bentley PJ, Hart E (eds) ICARIS '03: Proceedings of the International Conference on Artificial Immune Systems, Springer Berlin / Heidelberg, Edinburgh, UK, Lecture Notes in Computer Science, vol 2787, pp 147–155, URL http://dx.doi.org/10.1007/978-3-540-45192-1_15

Alpaydin E (2004) Introduction To Machine Learning. MIT Press

Anantvalee T, Wu J (2007) A survey on intrusion detection in mobile ad hoc networks. In: Xiao Y, Shen XS, Du DZ (eds) Wireless Network Security, Signals and Communication Technology, Springer, pp 159–180, URL http://dx.doi.org/10.1007/978-0-387-33112-6_7

Bajaj L, Takai M, Ahuja R, Tang K, Bagrodia R, Gerla M (1999) GloMoSim: A Scalable Network Simulation Environment. UCLA Computer Science Department Technical Report 990027

Barford P, Kline J, Plonka D, Ron A (2002) A signal analysis of network traffic anomalies. In: IMW '02: Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurement, ACM, Marseille, France, pp 71–82, DOI http://doi.acm.org/10.1145/637201.637210

Barr R, Haas Z, van Renesse R (2005) JiST: an efficient approach to simulation using virtual machines. Software Practice and Experience 35(6):539–576

Barrett C, Drozda M, Engelhart D, Kumar V, Marathe M, Morin M, Ravi S, Smith J (2005) Understanding protocol performance and robustness of ad hoc networks through structural analysis. In: Proceedings of the IEEE International Conference on Wireless and Mobile Computing, Networking and Communications (WiMob'2005), Montreal, Canada, vol 3, pp 65–72

Bhuse V, Gupta A, Lilien L (2005) DPDSN: Detection of packet-dropping attacks for wireless sensor networks. In: Proceedings of the Fourth International Trusted Internet Workshop, Goa, India

Drozda M, Schildt S, Schaust S, Szczerbicka H (2010) An Immuno-Inspired Approach to Misbehavior Detection in Ad Hoc Wireless Networks. Computing Research Repository (CoRR) URL http://arXiv.org/abs/1001.3113

Feeney L, Nilsson M (2001) Investigating the energy consumption of a wireless network interface in an ad hoc networking environment. In: INFOCOM 2001: Proceedings of Twentieth Annual Joint Conference of the IEEE Computer and Communications Societies, Anchorage, Alaska, vol 3, pp 1548–1557

Forrest S, Perelson A, Allen L, Cherukuri R (1994) Self-nonself discrimination in a computer. In: Proceedings of IEEE Computer Society Symposium on Research in Security and Privacy, Oakland, CA, USA, pp 202–212

Frauwirth KA, Thompson CB (2002) Activation and inhibition of lymphocytes by costimulation. The Journal of Clinical Investigation 109(3):295–299, DOI 10.1172/JCI14941, URL http://www.jci.org/articles/view/14941

Gonzalez O, Howarth M, Pavlou G (2007) Detection of packet forwarding misbehavior in mobile ad-hoc networks. In: Boavida F, Monteiro E, Mascolo S, Koucheryavy Y (eds) Proceedings of the International Conference on Wired/Wireless Internet Communications, Springer Berlin / Heidelberg, Coimbra, Portugal, Lecture Notes in Computer Science, vol 4517, pp 302–314, URL http://dx.doi.org/10.1007/978-3-540-72697-5_26

Hofmeyr S, Forrest S (1999) Immunity by design: An artificial immune system. In: GECCO '99: Proceedings of Genetic and Evolutionary Computation Conference, Morgan Kaufmann, Orlando, Florida, USA, vol 2, pp 1289–1296

Hu Y, Perrig A, Johnson D (2003) Packet leashes: a defense against wormhole attacks in wireless networks. In: INFOCOM 2003: Proceedings of the Twenty-Second Annual Joint Conference of the IEEE Computer and Communications Societies, IEEE, San Francisco, CA, USA, vol 3, pp 1976–1986

Hu Y, Perrig A, Johnson D (2006) Wormhole attacks in wireless networks. IEEE Journal on Selected Areas in Communications 24(2):370–380

Huang Ya, Lee W (2003) A cooperative intrusion detection system for ad hoc networks. In: SASN '03: Proceedings of the 1st ACM workshop on Security of ad hoc and sensor networks, ACM, Fairfax, Virginia, pp 135–147, DOI http://doi.acm.org/10.1145/986858.986877

IEEE Std. 802.11 (2007) Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications. IEEE Standard for Information technology. DOI http://dx.doi.org/10.1109/IEEESTD.2007.373646

IEEE Std. 802.15.4 (2003) Part 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (WPANs). IEEE Standard for Information technology. DOI http://dx.doi.org/10.1109/IEEESTD.2003.94389

Johnson DB, Maltz DA (1996) Dynamic source routing in ad hoc wireless networks. In: Imielinski T, Korth HF (eds) Mobile Computing, The Kluwer International Series in Engineering and Computer Science, vol 353, Springer, pp 153–181, URL http://dx.doi.org/10.1007/978-0-585-29603-6_5

Kohavi R, John G (1997) Wrappers for feature subset selection. Artificial Intelligence 97(1-2):273–324

Krishnamurthy S, Thamilarasu G, Bauckhage C (2009) Malady: A machine learning-based autonomous decision-making system for sensor networks. In: Proc. of IEEE International Conference on Computational Science and Engineering, IEEE Computer Society, Vancouver, Canada, vol 2, pp 93–100, DOI http://doi.ieeecomputersociety.org/10.1109/CSE.2009.246

Marti S, Giuli TJ, Lai K, Baker M (2000) Mitigating routing misbehavior in mobile ad hoc networks. In: MobiCom '00: Proceedings of the 6th annual international conference on Mobile computing and networking, ACM, Boston, Massachusetts, United States, pp 255–265, DOI http://doi.acm.org/10.1145/345910.345955

Mierswa I, Wurst M, Klinkenberg R, Scholz M, Euler T (2006) Yale: Rapid prototyping for complex data mining tasks. In: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, Philadelphia, PA, USA, pp 935–940

Murphy K, Travers P, Walport M (2008) Janeway's immunobiology. Garland Pub

Perkins CE, Royer EM (1999) Ad hoc On-Demand Distance Vector Routing. In: Proceedings of the 2nd IEEE Workshop on Mobile Computing Systems and Applications, IEEE Press, New Orleans, Louisiana, USA, pp 90–100

Rappaport T (2001) Wireless communications: principles and practice. Prentice Hall PTR Upper Saddle River, NJ, USA

Sarafijanovic S, Le Boudec JY (2004) An artificial immune system for misbehavior detection in mobile ad-hoc networks with virtual thymus, clustering, danger signal, and memory detectors. In: Nicosia G, Cutello V, Bentley PJ, Timmis J (eds) ICARIS '04: Proceedings of the International Conference on Artificial Immune Systems, Springer Berlin / Heidelberg, Catania, Sicily, Lecture Notes in Computer Science, vol 3239, pp 342–356, URL http://dx.doi.org/10.1007/978-3-540-30220-9_28

Scalable Networks (2010) Qualnet Simulator. URL http://www.scalable-networks.com, accessed: August 12, 2010

Schaust S, Drozda M (2008) Influence of Network Payload and Traffic Models on the Detection Performance of AIS. In: Procedings of International Symposium on Performance Evaluation of Computer and Telecommunication Systems (SPECTS), IEEE Press, Edinburgh, UK, pp 44–51

Seward J (2010) BZIP2 data compression library. URL http://www.bzip.org, accessed: August 12, 2010

Sterbenz JPG, Krishnan R, Hain RR, Jackson AW, Levin D, Ramanathan R, Zao J (2002) Survivable mobile wireless networks: issues, challenges, and research directions. In: WiSE '02: Proceedings of the 1st ACM workshop on Wireless security, ACM, Atlanta, GA, USA, pp 31–40, DOI http://doi.acm.org/10.1145/570681.570685

Texas Instruments (2007) CC2420 - 2.4 GHz IEEE 802.15.4 / ZigBee-ready RF Transceiver

Ubiquiti Networks (2010) XTREMERange2 - Carrier-Class 2.4GHz 802.11b/g Radio Module Datasheet. URL http://www.ubnt.com/xr2, accessed: August 12, 2010

Vaidya N, Hameed S (1999) Scheduling data broadcast in asymmetric communication environments. Wireless Networks 5(3):171–182

Wistron NeWeb Corp (2010) Wistron CM9 Datasheet. URL http://www.wneweb.com/, accessed: August 16, 2010

Yegneswaran V, Barford P, Ullrich J (2003) Internet intrusions: global characteristics and prevalence. In: SIGMETRICS '03: Proceedings of the 2003 ACM SIGMETRICS international conference on Measurement and modeling of computer systems, ACM, San Diego, CA, USA, pp 138–147, DOI http://doi.acm.org/10.1145/781027.781045

Zhao F, Liu J, Liu J, Guibas L, Reich J (2003) Collaborative signal and information processing: an information-directed approach. Proceedings of the IEEE 91(8)

ZigBee Alliance (2005) ZigBee Specification. URL http://www.zigbee.org, accessed: August 10, 2010