Hindawi
Security and Communication Networks
Volume 2018, Article ID 1753897, 14 pages
https://doi.org/10.1155/2018/1753897

Research Article

An Intrusion Detection and Prevention Framework for Internet-Integrated CoAP WSN

Jorge Granjal and Artur Pedroso

CISUC/DEI, University of Coimbra, Coimbra, Portugal

Correspondence should be addressed to Jorge Granjal; jgranjal@dei.uc.pt

Received 5 January 2018; Accepted 14 March 2018; Published 22 April 2018

Academic Editor: Félix Gómez Mármol

End-to-end communications between Internet devices and Internet-integrated constrained wireless sensing platforms will provide an important contribution to the enabling of many of the envisioned IoT applications and, in this context, security must be addressed when employing communication technologies such as 6LoWPAN and CoAP. Considering the constraints typically found on sensing devices in terms of energy, memory, and computational capability, the integration of Wireless Sensor Networks (WSN) with the Internet using such technologies will open new threats and attacks that must be dealt with, particularly those originated at devices without the constraints of WSN sensors (e.g., Internet hosts). Existing encryption strategies for communications in IoT environments are unable to protect Internet-integrated WSN environments from Denial of Service (DoS) attacks, as well as from other forms of attacks at the network and application layers using CoAP. We may thus fairly consider that anomaly and intrusion detection will play a major role in the materialization of most of the envisioned IoT applications. In this article, we propose a framework to support intrusion detection and reaction in Internet-integrated CoAP WSN, and in the context of this framework we design and implement various approaches to support security against various classes of attacks. We have implemented and evaluated experimentally the proposed framework and mechanisms, considering various attack scenarios, and our approach was found to be viable, from the point of view of its impact on critical resources of sensing devices and of its efficiency in dealing with the considered attacks.

1. Introduction

As constrained wireless sensing and actuating devices are progressively integrated with the Internet communications infrastructure, the importance of detecting and dealing with attacks against its security and stability appears as a fundamental requirement. This integration is becoming a reality, thanks to a standardized communications stack being designed for the IoT [1], empowered by protocols such as the 6LoWPAN adaptation layer [2], RPL (IPv6 Routing Protocol for Low Power and Lossy Networks) [3], and the Constrained Application Protocol (CoAP) [4]. Other protocols could also be considered at the application-layer, such as MQTT (Message Queuing Telemetry Transport) [5], but our focus in CoAP is motivated by its support of low-energy wireless local communication environments, machine-to-machine (M2M) communications between constrained sensors and actuators and other external Internet devices, and its direct compatibility with HTTP. We find this last property to be of particular importance, since it allows to leverage existing web applications. Our focus is thus on the detection of attacks against sensing devices that are part of the Internet communications infrastructure, and CoAP is of particular relevance, since it was designed with the purpose of guaranteeing interoperability with the web. It is also important to note that, despite the current focus on IEEE 802.15.4 as the low-energy link-layer communication technology supporting Internet-integrated WSN environments, other technologies are being adopted by the 6LoWPAN adaptation layer, as is the case already with Bluetooth Low-Energy (BLE) [6].

Threats may appear in the form of Denial of Service (DoS) attacks and attacks at the network (6LoWPAN) and application layers, by subverting the usage rules of CoAP. Although security mechanisms such as IPSec and DTLS have

been adopted to secure 6LoWPAN and CoAP communications, encryption only protects against external attacks and is also not able to offer protection against DoS, because cryptography cannot detect attackers using legal keys but behaving maliciously. It is thus fair to consider that intrusion detection and prevention will play a significative role in enabling most of the envisioned applications on the IoT [1, 7]. Unlike in older isolated WSN environments, CoAP sensing networks employ IP (Internet Protocol) communications and may be directly connected to untrusted Internet devices, thus opening the door to new threats that must be prevented and dealt with.

Although we may find numerous proposals in the literature focusing on intrusion detection in WSN environments, we verify that most of such proposals are not applicable to the IoT, due to various reasons. One is that such works are usually designed to be decentralized, thus without a central manager or controller. In 6LoWPAN WSN environments, 6LoWPAN border routers (6LBR) are employed to support the integration of WSN with the Internet and, other than the forwarding of communications between the two domains, may also assist in the support of intrusion detection and prevention. It is also useful to consider that border routers may be less resource constrained than sensing and actuating devices and as such may be employed to support more resource-demanding security-related operations. Other limitations we may find on classic approaches to intrusion detection in isolated WSN environments is that such proposals do not assume that sensing devices may be identified globally, nor that messages may be exchanged between such devices for the purpose of managing security. Again, 6LoWPAN opens the door to the usage of IP addresses as global identifiers of sensors or actuators [1] and adopts technologies such as IPSec and DTLS that may provide a layer of encryption on top of which we may exchange security management messages.

We proceed by describing how the article is structured. In Section 2 we extend our analysis of intrusion detection and prevention in the IoT, with a particular focus on 6LoWPAN-related communication protocols. In Section 3 we describe the proposed intrusion detection and prevention framework, together with its modules and the format of messages for managing security. In Section 4 we focus on the intrusion detection and reaction mechanisms implemented and evaluated in the context of the proposed framework, which we evaluate experimentally in Section 5. Finally, in Section 6 we conclude the article and discuss future work in this area.

2. Intrusion Detection and Prevention in 6LoWPAN Communication Environments

Existing research proposals on intrusion detection and prevention for Internet communication environments belong in three main classes: signature-based, anomaly-based, and specification-based [8]. We find it useful to analyze the main difference between such approaches and how they are applicable to IoT environments, as we proceed to discussion. In short, in signature or misuse-based intrusion detection systems patterns of the known attacks are first configured, and this information is employed by the system to detect attacks among the analyzed communications [9]. In this class, systems are usually characterized by low false alarm rates, although they are required to store large data sets (the attack signatures and also the data to be analyzed) and are unable to detect attacks not described by signatures. In anomaly-based intrusion detection systems, normal network behavior is first classified and compared with the monitored communications, in order to detect anomalous activities [10]. This class of systems possess the ability to detect new attacks but can be characterized by a high false alarm rate. Finally, specification-based systems are a variant of anomaly-based systems and work by specifying normal network operations in detail and monitoring any breaking of that specification. Such systems decrease the false detection rate but, on the other hand, the operation patterns must usually be created by specialists. The current trend in IDS research for the IoT is to combine the aforementioned approaches and thus benefit from their detection capabilities [11, 12]. Another useful characterization of intrusion detection and prevention is in what respects the topology of the employed architecture, which may be either distributed, centralized, or hybrid. Distributed systems intrusion detection is supported by the various devices in the network, while in centralized systems a single system is responsible for this task. A hybrid system combines distributed and centralized intrusion detection, with a central device being responsible for more complex analysis and decisions operations. In this article, we consider the implementation of an *hybrid* intrusion detection and prevention architecture, employing signatures to detect attacks against 6LoWPAN and CoAP, as well as DoS.

Analyzing recent (less than five years) research proposals on intrusion detection and prevention in 6LoWPAN and CoAP environments, we find that existing proposals are mostly focused on supporting security against attacks on routing operations with RPL. A first approach towards IDS in IoT environments is presented in [13], in the form of SVELTE, a system designed to protect WSN from attacks against routing operations, in particular spoofed or altered routing messages, sinkhole, and selective-forwarding attacks. Attacks are detected by maintaining routing information in the 6LBR, constructed from RPL information detected and forwarded by the gateway, and also from information reported by the various sensors in the network. This information base thus serves the purpose of enabling the detection of inconsistencies in the routing tree. SVELTE does not focus on security against attacks at the network and upper layers (CoAP), or DoS or other types of attacks. The authors in [14] discuss an evolution of SVELTE focused on sinkhole attacks. In [15], again an evolution of SVELTE is proposed which adds a new parameter, in the form of a link reliability metric, with the goal of preventing the 6LBR and neighboring nodes to actively engage with malicious intruders. The implementation and evaluation are based on Contiki and COOJA is used for simulations, and the authors claim that their proposal improves the true positive rate of SVELTE. In [16] the authors focus again on threats against RPL and propose a two-layer IDS architecture designed to detect internal attacks on routing based on three components: an RPL specification-based monitor, anomaly-based used in cooperation with

specification-based detection to monitor node performance, and a statistical-based component to reveal the attacker source. Although this work performs a good job in discussing the applicability of WSN IDS systems to IoT 6LoWPAN environments, it is mostly focused on internal attacks against routing operations using RPL. Also, the proposed system is not materialized in the form of concrete detection and reaction mechanisms. In [17] the authors propose an intrusion detection method based on evaluating, over time, the energy consumption of sensing devices. The proposed system classifies sensing devices with irregular energy consumption as malicious attackers, compared with energy consumption models built for communications in a 6LoWPAN network, for both mesh-under and route-over IEEE 802.15.4 communications. From results obtained via simulation, the authors state that this strategy allows detecting misbehaving nodes and that such nodes may subsequently be excluded from operations in the network. One limitation of this approach is that IoT applications may not always present a homogeneous energy consumption profile over time, and other is that in this work the focus is not on attacks at the network or application layers. In [18] the authors again focus on discussing DoS attacks against routing operations using RPL. Despite their discussion on various type of attacks against routing, as well as recommendations on practical aspects such as the placement of the IDS system and the types of systems to employ and the parameters to consider in various scenarios, this article does not address nor evaluate any particular detection or prevention mechanisms. In [19] the authors propose a DoS architecture for 6LoWPAN, designed to integrate with the network framework developed within the EU FP7 project *ebbits*. This framework is focused on critical network environments, and the proposed architecture has been designed and evaluated for industrial environments. A preliminary implementation is also discussed and evaluated using a penetration testing system. The proposed system is focused on jamming attacks but is dependent on information modules available on the *eebits* architecture, and security-related notifications are dependent on the usage of a dedicated communications medium for security-related data. In [20] the authors address the design of a system based on detecting misbehaving nodes in a 6LoWPAN network, with the assumption that neighbor nodes in the network behave similarly, in terms of communications, during the lifetime of the application. The performance of this proposal is evaluated (in respect to its true positive and false positive rates) and found to perform well, the same applying to energy. We note that this work is focused again on attacks against routing operations in 6LoWPAN environments, in this case considering that all nodes in a given DODAG are supposed to operate similarly. In conclusion, from the previous analysis we may observe that, on the one hand, intrusion detection and prevention proposals addressing 6LoWPAN environments are very recent while, on the other hand, most of the existing proposals focus on attacks against routing. If it is true that we are able to find proposals dealing with attacks (in particular DoS) targeting 6LoWPAN operations, we are unable to find proposals combining detection and reaction to DoS attacks at the various layers of the communications stack, together with attacks at the application-layer using CoAP.

3. A Framework for Intrusion Detection and Prevention with CoAP

We proceed by analyzing the framework considered for distributed intrusion detection and prevention in CoAP Internet-integrated sensing environments. We start by identifying the system and security requirements considered for the design of this framework, and next we discuss the operation of its modules. Finally, we also address security management in the context of this framework.

3.1. System and Security Requirements. Applications for the IoT in areas as diverse as smart cities, surveillance, and smart energy will require fundamental assurances from the infrastructure in terms of security, and this certainly includes the ability of detecting and reacting to attacks against the availability of such devices and of the application itself, in a timely fashion. We are able to note that currently there is a lack of systems and mechanisms designed to enable intrusion detection in Internet-integrated CoAP communication environments and, with this goal in mind, we target the following goals:

(i) Cross-layer attack detection: detect attacks at the network (6LoWPAN), transport (RPL), and application (CoAP) layers.

(ii) Detect attacks originated at external (e.g., Internet) hosts and at internal devices (e.g., sensing and actuating devices located in the same or in a different WSN communication domain than the attacked host).

(iii) Intrusion prevention and filtering: react in a timely fashion to attacks and block communications from attackers.

(iv) Extensibility: support different intrusion and prevention mechanisms at the various layers, according to the requirements of the IoT application at hand.

(v) Reconfigurability: support reconfigurable detection and reaction policies, during the lifetime of the application.

(vi) Interoperability: support the integration of WSN domains employing different low-energy wireless communication technologies compatible with 6LoWPAN, in order to enable end-to-end communications established between devices in such domains and the Internet.

The intrusion detection and prevention framework we describe next is designed to materialize the previous goals. We start by analyzing the system architecture for intrusion detection and prevention in CoAP communication environments, and next we discuss how processing and filtering of communications is implemented, the same applying to security management related operations.

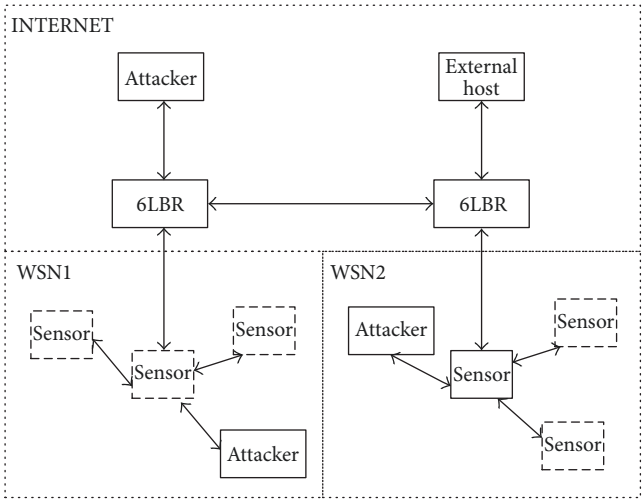

Figure 1: System architecture for intrusion detection and prevention in CoAP Internet-integrated networks.

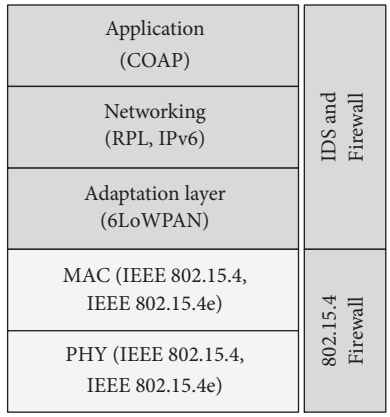

Figure 2: IDS and firewall modules in the communications stack (sensing device).

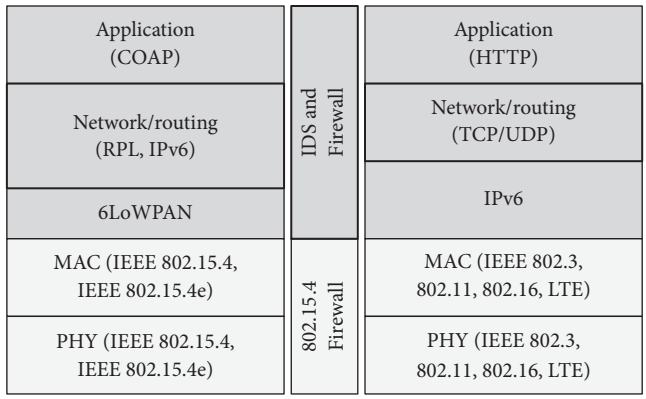

Figure 3: IDS and firewall modules in the communications stack (6LBR).

3.2. Intrusion Detection and Prevention Framework. The design of intrusion detection and prevention mechanisms for Internet-integrated WSN environments must consider a careful usage of the resources available in constrained sensing and actuating platforms while, on the other hand, it may adapt and benefit from the availability of more resources in other classes of devices, in particular in 6LoWPAN Border Routers (6LBR). We consider an hybrid approach to intrusion and detection, and thus such functionalities are implemented by all devices participating in CoAP communications. Figure 1 illustrates the system architecture considered for intrusion detection and prevention. We consider that devices performing intrusion detection are trusted, and as such both constrained sensing devices and 6LBR (gateways) support this role. Regarding the usage of the resources available for supporting intrusion detection and reaction, 6LBR devices support more intensive operations, with the various sensors in the WSN domains cooperating in the task of detecting and reacting to attacks.

As illustrated in Figure 1, attackers may be located in the same WSN domain as the attacked device or be external, thus located in another WSN domain or in the Internet. The cooperation of the various devices in attack detection and reaction is thus fundamental in order to stop attacks in a timely fashion, either at the end device being attacked or, if required, by blocking further communications between the Internet and WSN domains at the 6LBR. Other than blocking communications, the 6LBR can also send security management messages to sensing devices in the WSN, in order to start blocking communications received from the attacker. Next we describe the modules that implement our intrusion detection and reaction framework, in the context of the two classes of devices considered, sensing devices in Figure 2, and gateways (6LBR) in Figure 3.

Considering the communications stack illustrated in Figures 2 and 3, we start by noting that encryption can be enabled at either the physical and data-link layers, for communications in the WSN domain using IEEE 802.15.4 symmetric encryption, or on the other hand with DTLS or IPSec at higher layers of the communications stack. We enable detection of attacks by analyzing communications at the various layers, in particular at the 6LoWPAN adaptation layer, networking (routing with RPL), and application (using CoAP). We also implement a firewall to filter out undesired communications at the various stages (or layers) of processing of communications in the context of the networking stack of the device (sensing device or 6LBR). This strategy allows us to react to attacks by blocking communications, as soon as the attack is detected.

In Figures 4 and 5 we illustrate the flow of processing of a 6LoWPAN packet received on a sensing device and 6LBR, respectively. When a constrained device receives a 6LoWPAN packet, the first verification is on its source IP address. Other verifications are also performed on the packet to verify if the number of messages received from that particular source address is over a predefined threshold over a particular time frame, and we use this limit to drop further communications from devices attempting to perform DoS attacks. We are also able to distinguish between various preconfigured packet

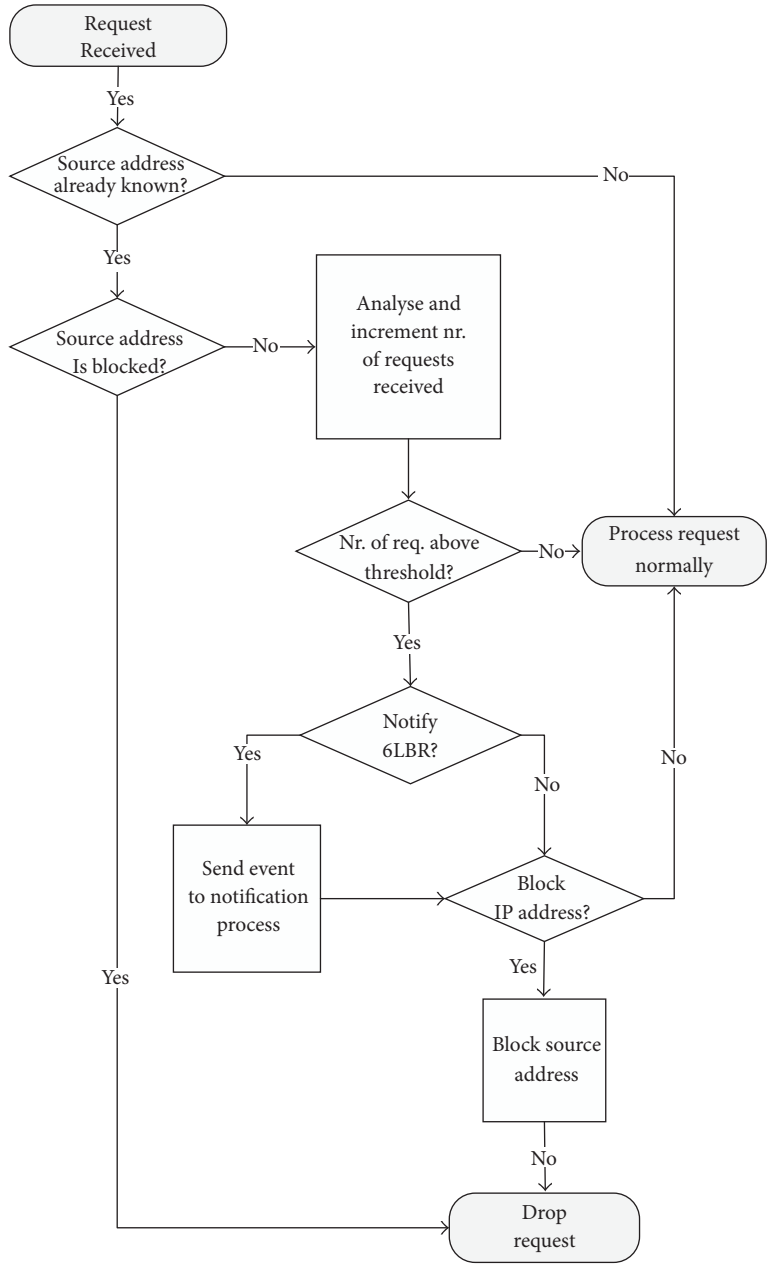

Figure 4: Processing of communications and security (sensing device).

types, by examining information available at the network, routing, and application layers.

We must also note that the dropping of undesired messages by the attacked device may occur simultaneously with the transmission of security warnings to the 6LBR, if this is required by the security policy for that particular communication, as we discuss later in the article. In fact, security management messages (discussed next) support the enabling of various types of security measures implemented by the 6LBR, in particular denying further messages originated at a particular source address from being forwarded and warning other sensors in the vicinity of the attacked device to also start blocking communications from the attacker.

In order to implement the previously described detection and reaction flow in constrained sensing platforms, devices store data regarding the source IP addresses and the signatures of known intrusions, along with the actions to be performed when such attacks are detected. In the 6LBR, we maintain data for the sensing devices in the WSN domain, particularly the IP address of each device, its CoAP resources, and the source IP addresses that are known by the device (enabling the device to autonomously detect attacks). Whenever a sensing device sends a notification to the 6LBR, the notification message transports an indication of the type of request received and of the origin IP address of the request. Upon receiving the notification, the 6LBR decides if an action

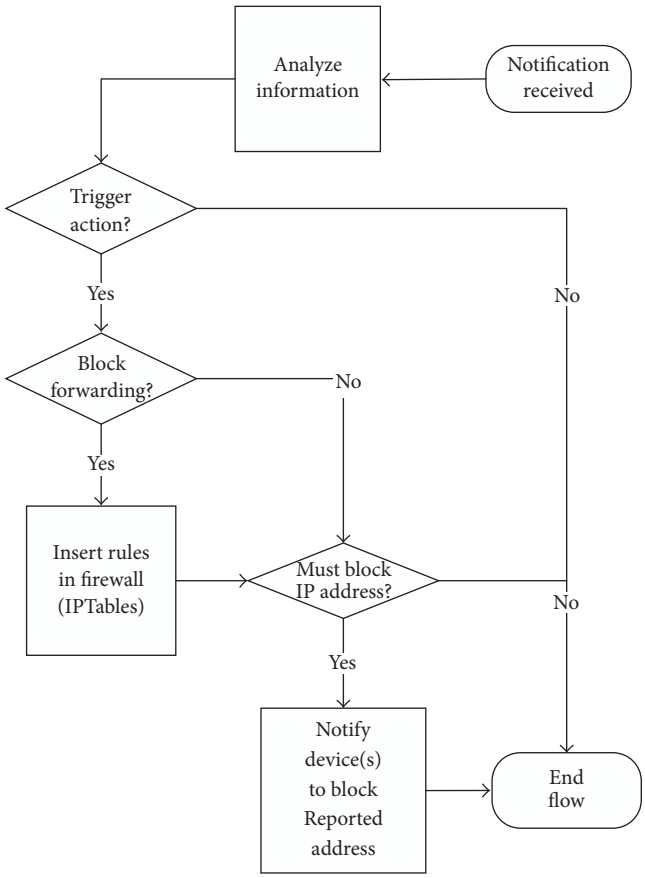

Figure 5: Processing of communications and security (6LBR).

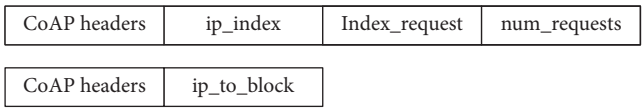

Figure 6: Format of security notification messages.

must be triggered, according to the security policy at hand. We proceed by analyzing our usage of security management messages in greater detail.

3.3. Security Management. As previously discussed, the security module is responsible for the transmission, reception, and processing of security management messages. Security management messages allow us to transmit information regarding the detected attacks, and about how the various devices must act in order to coordinately stop such attacks. As we focus on intrusion detection and prevention on CoAP IoT networks, security management messages are transported in the payload of CoAP confirmable messages, as such being inherently protected from packet losses [4]. In Figure 6 we illustrate the format of the security management messages exchanged between devices, and we also assume that communications may be protected via encryption, either at the network layer using IPSec or at the transport layer using DTLS. In Figure 6 we illustrate, at the top, the format for security messages originated at a sensing device and at the bottom those originated at the 6LBR.

As for security messages originated at a sensing device, the *ip_index* field stores a position to a vector holding structures which contain an IP address, the number of distinct requests received by the sensor from each source IP address and a flag indicating weather communications from that source address are already blocked or not. The *index_request* field indicates the type of request received by the constrained device (that subsequently has motivated the generation of the security notification message). Finally, in the *num_requests* field we transport the number of requests received, so far, by the sensing device, which are of type *index_request* and have been received from the source IP address identified by *ip_index*. Regarding the security notification messages originated at the 6LBR, *ip_to_block* is used again as an index to a position in a vector maintained in the memory of the destination sensing device, storing information about the blocked origin device. Finally, we note that the vector holding information about each entity known in the network, together with a counter of requests received from that entity and the corresponding blocking information, is stored in the various devices in the network in a synchronized fashion. This information, together with the identification of the type of request, allows for the exchange and updating of the information required for the security management module in each device to be able to detect and act upon received 6LoWPAN and CoAP communications.

4. Implementation Strategy

With the goal of evaluating experimentally the proposed approach, we have implemented the intrusion detection and reaction framework in the Contiki operating system [21]. The first goal is to integrate security with the processing of 6LoWPAN and CoAP communications in the uIPv6 [22] stack, as we proceed to discussion.

4.1. Counters and Thresholds for Attack Detection. As previously noted, the proposed framework is extensible, given that it allows for new filtering and analysis mechanisms to be integrated in the framework. Also, such mechanisms may distinguish between different types of requests at the network, routing, and application layers. The combination of different analysis and filtering mechanisms allows for supporting different intrusion detection strategies. We start by noting that one important motivation for our work is to detect and prevent resource exhaustion attacks at CoAP networks, particularly since network congestion control in CoAP is not controlled by the server (in fact being implemented via transmission parameters in CoAP messages sent by client [16]). With this aspect in mind, we consider a threshold for the number of requests received by the sensing device over a minute, above which we consider that the security and the stability of the WSN environment (and thus of the IoT application itself) may be at risk. Other than DoS attacks, we also consider attacks employing other types of messages, as

well as those subverting the usage rules of the CoAP protocol. In order to support the detection of the previous attacks, we maintain, in a given sensing device, separate counters for the following types of messages received by the system, during the previous time frame:

(i) Number of valid CoAP requests to resources (sensors and actuators) available in the device;

(ii) Number of invalid CoAP requests (malformed requests or requests to sensors or actuators not available in the device);

(iii) Number of messages of other protocols (e.g., ICMP and TCP);

The previous counters, together with the thresholds that may be activated for each type of security attack, allow us to detect and react to attacks via the security manager and firewall modules, as we have previously discussed.

4.2. Intrusion Detection and Prevention Policy. It is important to note that our framework supports different detection and blocking policies, in line with the requirements of the IoT application at hand. For the purpose of evaluating experimentally our proposal, we have considered the policy described next. This policy is implemented as a set of pre-configured rules in the memory of the participating devices (sensing devices and also the 6LBR gateway) and identifies the conditions that trigger reactions against attacks.

```
# Block requests to CoAP1 received from IP address orig1
If IP=orig1 and res=CoAP1 then block

# Notify the gateway (6LBR) when more than 5 requests per
# minute are received for the resource CoAP2, irrespective
# of the source
If IP=⋆ and res=CoAP2 and NReqMin>=5 then notify

# Security against attackers sending malformed CoAP requests
If IP=⋆ and res=malformed and NReqMin>=1 then notify
If IP=⋆ and res=malformed and NReqMin>=3 then block and notify

# Define a general threshold for the number of messages
# accepted by the device
If IP=⋆ and res=⋆ and NReqMin>=5 then notify
```

In the previous security policy, we start by considering the blocking of requests for a particular resource originated at IP address *orig1*, as well as the requests triggering a notification to the 6LBR above a particular threshold. We also enable security against undesired or malformed CoAP requests, by notifying and blocking further communications entering the device. Finally, we enable general security against DoS attacks, by establishing a threshold for the number of requests received (per minute). The following is the counterpart security policy, as defined in this case for the 6LBR.

```
# Block requests to resource CoAP2 on sensor1 received from
# the Internet or from a different WSN domain, if a
# notification is received from the device and the number
# requests received per minute is above the threshold
If NotifSource=sensor1 and IP=external and res=CoAP2 and NReqMin>=5 then blockdst=sensor1

# Notify all sensors in the WSN about a device making
# requests to resource CoAP2 when at least two alerts have
# been sent from sensing device
If NotifSource=⋆ and NNotif>=2 and IP=internal and res=CoAP2 then notifyblock=all
```

```
# Block malformed messages and notify other internal
# devices to also block such communications
If NotifSource=⋆ and NNotif>=3 and res=malformed and IP=⋆ then notifyblock=all and blockdst=
all

# Notify the internal devices of all communications
# exceeding a certain threshold of messages received per
# minute, irrespective of the origin, type and request
If NotifSource=⋆ and NNotif>=2 and res=⋆ and IP=⋆ and NReqMin>=7 then notifyblock=all
```

In the policy established for the 6LBR, we may observe that the gateway is able to control and block the forwarding of 6LoWPAN and CoAP communications according to the security warnings received from sensing devices and also to send security notifications instructing such devices of the necessity of acting (e.g., blocking) further communications received from the (suspect) origin IP origin. We proceed by discussing in detail how the proposed framework and mechanisms were implemented in the devices employed for the purpose of supporting our experimental evaluation study.

4.3. Implementation Details (Sensing Devices). We have modified the source code of the Contiki operating system [21], with the goal of experimentally evaluating the proposed intrusion detection and prevention approach. For this purpose of our experimental measurements, we have employed MTM-CM5000MSP TelosB sensing devices (illustrated in Figure 7). This device runs on a TI MSP430 micro controller, with 10 Kb of RAM and 48 Kb of ROM.

We do not currently support the management of addresses dynamically in the internal memory of the device, and as such addresses are statically preconfigured during the bootstrap phase of the application. The information required to perform analysis and filtering of the communications entering the device is maintained in the *CtkIPsVctr* vector, with the structure illustrated next. As illustrated, in this vector we hold information on the number of requests received from a particular IP address during the previous time frame (one minute in our current implementation), and also if that particular IP source address is currently being blocked by the device.

```
struct ip_requests:

    ip;
    num_requests [NUM_ENTRY_TARGETS];
    blocked;
```

We currently support the holding of filtering and blocking information for NUM_ENTRY_TARGETS different entries, maintained in an array. The first entry of this array holds information for requests that are not CoAP messages (e.g., TCP and ICMP), and from the second element to the one in position NUM_ENTRY_TARGETS-2 we maintain information for the various CoAP resources (sensors or actuators) available in the sensing device. Finally, the last entry of the array accounts for malformed CoAP requests. In each vector, the entry *num_requests* is updated whenever a request from a source IP that is not blocked is received and processed by the sensing device. The *blocked* parameter can also be configured during the device commissioning and bootstrapping phase, so that the sensing device starts with the correct security setting. If the parameter *blocked* contains the value 1, requests from the corresponding source IP address are dropped before further processing by the firewall module, in the context of the communications stack (please refer to Figure 1).

In our current implementation we also employ another vector, the *CtkD&RVctr* vector, in this case with the purpose of maintaining the information related to the rules of the security policy. Next, we illustrate the structure of the various elements stored in this data structure.

```
struct ids_rule:

    ip_index;
    request_type;
    threshold;
    action;
```

In the previous structure we start by holding the source IP address to which the rule can be applied, which acts as an index to the corresponding entry in the *CtkIPsVctr* vector. The *request_type* element holds the request type for which the rule applies, and the *threshold* element represents the number of requests above which the rule is activated and the corresponding action is applied. In line with the previously described security policy, in our current implementation the sensing device supports the following actions:

(i) 0: notify the 6LBR, but do not block the message.

(ii) 1: block the source IP address (dropping further requests received from that device).

(iii) 2: block the source IP address and also notify the 6LBR.

It is also important to note that, in order to support the verification and the application of the rules in the order in which they are defined in the security policy, information maintained in lower positions of Vector *CtkD&RVctr* has

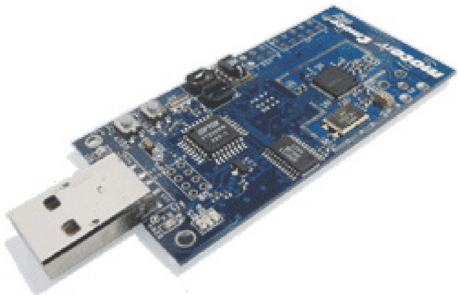

Figure 7: MTM-CM5000MSP (TelosB) sensing platform.

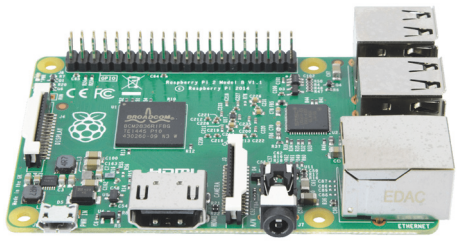

Figure 8: Raspberry Pi model B.

precedence over the rules in higher positions. The implementation of our security mechanisms is also required in the 6LBR, as we proceed to discussion.

4.4. Implementation Details (Gateway). For supporting the 6LBR, we have employed a Raspberry Pi model B device (illustrated in Figure 8). This device runs on a 1 GHz ARM processor, with 1 Gb of SDRAM. As per the security policy implemented in the context of our intrusion detection and prevention framework, this device is responsible for taking actions according to the security notifications received from sensing devices in the WSN domain. We use Raspbian Linux to support the network functionalities related to the role of the 6LBR, as well as to support intrusion detection, prevention, and security management.

Regarding the internal implementation of intrusion detection and reaction, the Vector *RPiIPsVctr* holds information associated with the sensing devices known in the WSN domain, and its structure is illustrated next. The fields *ip* and *ip_global* hold the sensors link-local and global IP addresses, respectively, and *num_resources* the number of CoAP resources hosted by that particular device. The field *resources_name* stores a pointer to the names of the hosted resources, while *resources_type* identifies the CoAP methods (GET, PUT, POST, or DELETE) supported by such resources.

```
struct sensor:

    ip;
    ip_global;
    num_resources;
    **resources_name;
    *resources_type;
    max_num_ips_in_ids_table;
    *ips_in_ids_table;
    *nnotif;
    *external_ip;
```

The field *max_num_ips_in_ids_table* corresponds to the size of *CtkIPsVctr* in the sensor (the table holding information regarding accesses received and filtered by the sensor), while *ips_in_ids_table* holds the IP addresses that also exist in the corresponding *CtkIPsVctr*. As we have previously discussed, this information is maintained synchronized among the various devices. The field *nnotif* holds the number of notifications received from a sensor notifying a given source IP address, while *external_ip* indicates if the address (maintained in the *ip* field) is internal (in the WSN domain) or external (in the Internet or in a separate WSN domain). We also hold a Vector *(RPiD&RVctr)* in the 6LBR, supporting the operationalization of the security rules in the policy, which we illustrate next.

```
struct rpi_rule:

    notify_source;
    reported_ip;
    reported_resource;
    notifies_threshold;
    requests_threshold;
    action;
    sensor_index_in_action;
```

In the previous structure, *notify_source* is a position in the *RPiIPsVctr* Vector holding the IP address of the constrained device that can trigger this rule. The rule also has a *reported_ip* and a *reported_resource*, which correspond to the position of the reported source IP address and the position of the reported resource, respectively. The fields *notifies_threshold* and *requests_threshold* represent the number of notifications received advertising the source IP address and the number of requests by the source IP for the specified resource, respectively. The *action* parameter defines the action to be taken (enforced) by the 6LBR in case the rule applies and may assume the following values:

(i) 0: notify one or more sensors (that have the reported IP in their CtkIPsVctrs Vectors) to block further communications from the reported source address.

(ii) 1: block (using IPTables in Linux) further forwarding of packets, originated at the reported source IP address.

(iii) 2: notify and also block further communications from the reported source IP address.

Finally, and referring again to the previous structure, in *sensor_index_in_action* we hold an index to the valid action, and if this number is equal to the number of sensing devices in the *RPiIPsVctr* Vector, the action is valid for all sensing devices.

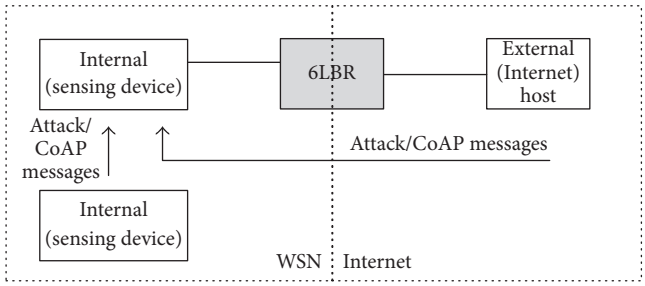

Figure 9: Experimental evaluation setup.

5. Experimental Evaluation

The proposed intrusion detection and prevention framework has been implemented and experimentally evaluated, with the goal of determining the effectiveness (in regard of its capability of detecting and reacting to attacks in a timely fashion) of the proposed mechanisms, as well as its impact in the critical resources available in constrained sensing platform. We start our discussion by describing our experimental evaluation setup.

5.1. Experimental Evaluation Setup and Goals. In Figure 9 we illustrate the experimental evaluation scenario. As illustrated, CoAP requests (as well as other types of messages targeting a constrained sensing device) may be originated either at another WSN device or at an external (Internet) host.

In this setup, we transmit different types of request messages to a CoAP sensing device and at different rates, with the goal of evaluating the effectiveness of the implemented mechanisms in dealing with DoS, as well as with attacks against the CoAP protocol. As previously discussed, our main goal is to experimentally evaluate the impact of intrusion detection and prevention on the resources of constrained devices and on the operations of IoT applications employing 6LoWPAN and CoAP, in particular:

(i) Evaluate the impact of intrusion detection and reaction on the *memory* of the sensing device. Memory is in fact a scarce resource on such devices, and as such this is important in order to ascertain on the effectiveness of our proposal.

(ii) Evaluate the *energy consumption* on the constrained sensing device in the presence of attacks, in comparison with normal operations. Energy consumption directly influences the achievable lifetime of IoT applications.

(iii) Evaluate the *computational effort* required from the sensing device in the presence of attacks (when dealing with the detection and blocking of such attacks), as this directly influences the maximum achievable communications rate of the application.

(iv) Evaluate the *delay* in reacting to attacks and blocking or filtering further communications, when such actions are performed directly at the sensing device or, on the other hand, centrally managed by the 6LBR. Our goal here is to verify the impact of the delay involved in notifying and reacting to the attack on the capability of the network filtering dangerous messages.

With the previous goals in mind, we have considered three complementary experimental evaluation scenarios, as we proceed to discussion.

5.2. Experimental Evaluation Scenarios. For the purpose of our evaluation, we have considered the following three complementary experimental evaluation scenarios:

(i) Scenario *E1*, filtered CoAP external requests: the sensing device is already blocking requests to resource *coap1* originated at a known external IP address.

(ii) Scenario *E2*, notify and block at the 6LBR (external attacker)*:* a known device in the Internet is sending CoAP requests to resource *coap2* on the sensing device. According to the considered security policy, this device is configured to notify the 6LBR when a given IP address transmits 5 or more requests per minute to a given CoAP resource. Upon receiving such notifications, 6LBR blocks further forwarding of communications.

(iii) Scenario *E3*, notify and block (internal attacker)*:* a known internal attacker (in the same WSN as the attacked device) sends requests to the resource *coap2*. The attacked sensing device is configured to notify the 6LBR when receiving 5 or more requests per minute directed to that resource, and upon receiving such notifications the 6LBR is configured to notify the devices in the WSN to block further requests received from the attacker.

For the purpose of establishing a baseline for comparison, other than the three previous scenarios we also consider the existence of a CoAP device fully exposed to internal and external communications. In this scenario, CoAP requests to resource *coap1* are received from an unknown external device, and the sensor does its best to process all messages, thus being fully exposed to DoS and attacks, as well as attacks against CoAP. We proceed by discussing the results obtained in our experimental evaluation.

5.3. Impact on Memory. In Figure 10 we illustrate the impact of the proposed intrusion detection and reaction mechanisms on the memory available in the TelosB. The impact is of 10.6% in the case of RAM (6830 bytes are used with IDS, against 6173 without security) and of 5.6% in the case of ROM (requiring 44926 bytes for the program code with security, against 42528 bytes without security).

Regarding its impact on memory, we may safely consider that the overhead of intrusion detection and prevention, materialized in the support of the firewall and security management modules of the framework, is acceptable, thus not compromising the employment of the previously discussed security functionalities on devices with the characteristics of the TelosB (MTM-CM5000MSP).

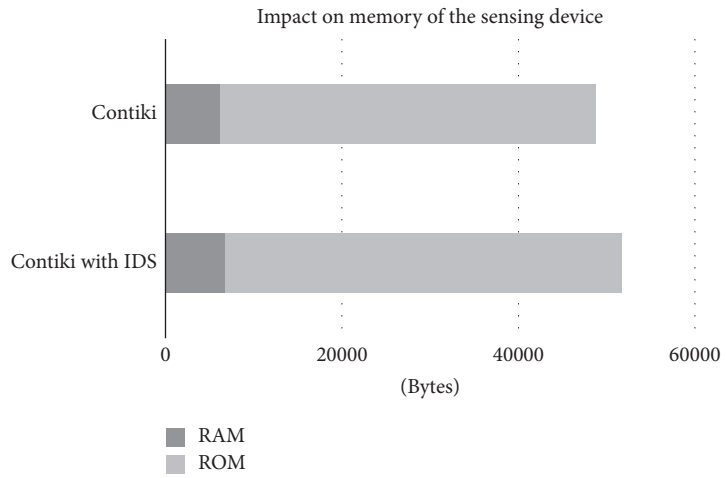

Figure 10: Impact of the proposed framework on the memory available on the sensing device.

5.4. Impact on Energy. As many sensing devices still depend on batteries, energy is a critical resource on IoT environments and thus must be used sparingly. It is thus important to guarantee that the communications and security protocols are efficient in respect to its usage of the available energy, in order not to compromise the lifetime of the IoT application at hand. With the purpose of measuring the energy required to support intrusion detection and reaction, we have used Powertrace [23] with Energest to profile power consumption in Contiki. Powertrace reports energy consumption with around 94% of reported accuracy and, using Energest, we are able to measure the CPU and radio cycles spent by the sensing device when receiving communications and processing security, taking measurements at each 20 seconds and for a period not less than 80 seconds. In each experiment, we start making requests to a sensing device in the beginning of the second period of 20 seconds, discarding the measures taken during the first period. We thus collect the next three 20-second periods measured by Energest and calculate the average cycles per second used by the sensing device in a 60 second period. Energest reports the CPU, LPM, Tx, and RX measurements, from which we are able to derive (analytically) the power usage. We consider 3V as standard voltage for our calculations and that a node is in low power mode (LPM) when the radio is off and the MCU (micro controller unit) is idle and calculate the CPU time when the MCU is on. In Figure 11 we illustrate the power required (in mW) to process CoAP requests, considering the previously discussed configurations (with and without security).

As can be observed in Figure 11, the impact of the implement security mechanisms provides evident energy savings, when compared against the scenarios where the sensor is fully exposed to attacks. For example, in the scenario E3 without security, the device is attacked by an internal attacker and up to 0,079 mW are required to process 480 CoAP requests in a minute, while with security (scenario E3 with IDS) this requirement decreases to 0,026 mW, or approximately 32% of the original value.

5.5. Lifetime of Sensing Applications. As previously discussed, another useful evaluation is on the impact of security on the lifetime of IoT applications. Considering our previous measurements on the energy required in the various evaluation scenarios, and the availability of two new AA-type batteries in the sensing platform employed in our experimental measurements, we are able to analytically derive the expected lifetime of IoT applications, which we illustrate in Figure 12.

We must note that the previously illustrated results consider only the energy required for processing communications and security, thus without considering the impact of other operations related to the IoT application at hand. Nevertheless, the illustrated measurements allows us to ascertain on whether intrusion detection and prevention may compromise the goals of the application in what respects the lifetime of sensing devices. The effectiveness of the proposed security mechanisms is again visible in the results illustrated in Figure 12. In the worst scenario (scenario E3 for 480 CoAP requests per minute), security still provides approximatively 71700 hours of lifetime, in contrast with only approximately 24060 without security. Also, if we consider the baseline measurements (scenario E1 without security), for 480 requests per minute the achievable lifetime is of approximately 21900 hours, less than one-third of the counterpart with security, which is of approximately 72240 hours.

5.6. Achievable Communications Rate. Our experimental evaluation allowed us also to measure the impact of security on the computational availability of the sensing device. In this context, it is important to consider that sensing devices as the TelosB are not capable of multiprocessing and thus, when busy processing communications and security, are unable to support other application-related operations. We thus find it important to consider also the impact of the attacks on the maximum achievable communications rate that a given IoT application can sustain. Similarly to our previous analysis, we consider the effort required solely for the processing of CoAP messages and the security-related

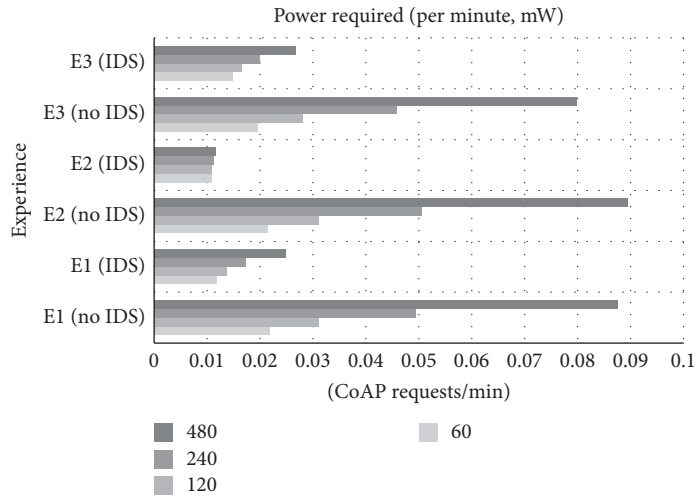

Figure 11: Power required to process CoAP requests (with and without IDS).

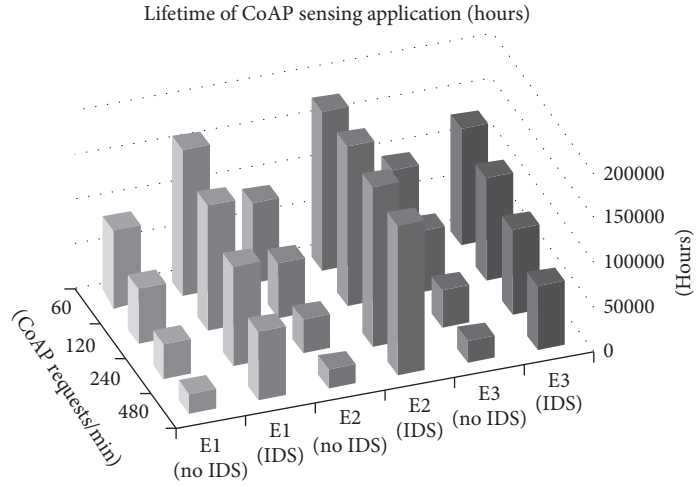

Figure 12: Lifetime of CoAP applications (with and without IDS).

operations. We consider the total available bandwidth to be of approximately 25700 bytes/sec, accounting for the overhead of 19.6% introduced by IEEE 802.15.4 on the total bandwidth available for CoAP communications. In Figure 13 we illustrate the maximum communications rate achievable, with and without the impact of security processing and of the transmission of security management messages.

As can be observed in Figure 13, the impact of security processing and communications, although higher for higher attack rates, does not compromise the viability of IoT CoAP applications in what respects the maximum communications rate sustained by the application. Considering modes with intrusion detection and reaction, in the worst scenario (scenario E1) the application can still make use of up to approximately 25306 bytes/sec, or 1.53% of the total available bandwidth.

5.7. Filtering and Blocking Efficiency. Regarding the efficiency in blocking (filtering) attacks, we also considered the effectiveness of the proposed mechanisms in what respects its

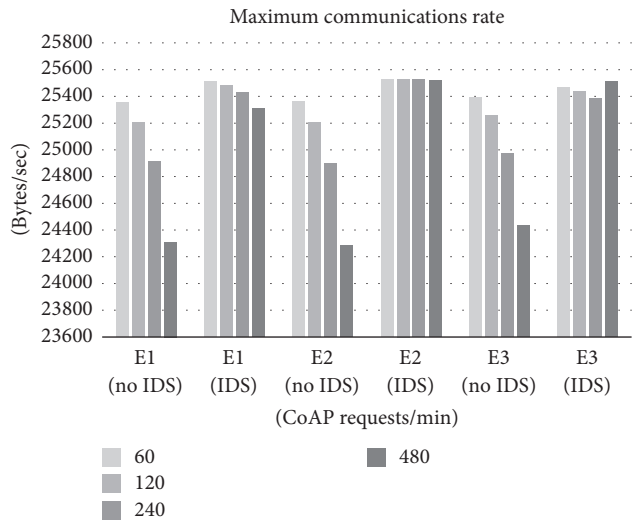

Figure 13: Maximum communications rate of CoAP applications (with and without IDS).

capability to react by blocking attacks on a timely fashion. Considering the scenario E2, we verify that communications are blocked immediately by the 6LBR, when up to 120 requests (attacks) per minute are received, upon receiving a notification from the attacked sensing device on the WSN domain. For 240 requests per minute, we verify that 1 CoAP request is able to reach the sensor, and this number rises to 2 in the case of 480 CoAP messages per minute. This is certainly due to the time required for security management messages sent by the sensing device to reach the 6LBR but is a limitation only when blocking is solely dependent on the 6LBR. We note that, in the remaining evaluation scenarios considered in our evaluation, the system is able to detect and react to attacks immediately.

6. Conclusions and Future Work

In this article we address the design of an architecture for distributed intrusion detection and reaction in Internet-integrated CoAP sensing environments and describe its implementation and the experimental evaluation of its effectiveness in detecting and reacting to attacks, as well as its impact on the fundamental resources of constrained wireless sensing platforms. As we have observed, the proposed framework is flexible and extensible, so that other attacks (which can target and be detected at the various layers of the communications stack) can be dealt with in the future. Focusing more closely on the application-layer, we plan to detect further attacks against the normal operation rules of the CoAP protocol. For example, CoAP acknowledgments may be sent to a device when no CoAP request exists, or requests can be sent to a CoAP server using invalid options such as Accept, Content-Format, and Max-Age, among others. Attacks can also be considered against the caching and proxy model defined in the protocol. As we have discussed, there is currently a lack of intrusion detection and reaction solutions focused on 6LoWPAN and CoAP environments and, as future work, we plan to extend the detection and filtering capabilities of the framework, namely, by supporting detection of new types of attacks against the CoAP application-layer protocol.

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

The work presented in this paper was partially carried out in the scope of the MobiWise project (P2020 SAICT-PAC/0011/2015), cofinanced by COMPETE 2020, Portugal 2020 Operational Program for Competitiveness and Internationalization (POCI), European Union ERDF (European Regional Development Fund), and the Portuguese Foundation for Science and Technology (FCT).

References

[1] M. R. Palattella, N. Accettura, X. Vilajosana et al., "Standardized protocol stack for the internet of (important) things," *IEEE Communications Surveys and Tutorials*, vol. 15, no. 3, pp. 1389–1406, 2013.

[2] G. Montenegro, N. Kushalnagar, J. Hui, and D. Culler, "Transmission of IPv6 packets over IEEE 802.15.4 networks," Tech. Rep., 2007.

[3] T. Winter, "Rpl: Ipv6 routing protocol for low-power and lossy networks," 2012.

[4] C. Bormann, A. P. Castellani, and Z. Shelby, "CoAP: an application protocol for billions of tiny internet nodes," *IEEE Internet Computing*, vol. 16, no. 2, pp. 62–67, 2012.

[5] B. Andrew and R. Gupta, "Mqtt version 3.1. 1. OASIS standard, 29," 2014.

[6] J. Nieminen, T. Savolainen, M. Isomaki, B. Patil, Z. Shelby, and C. Gomez, "Ipv6 over bluetooth (r) low energy," Tech. Rep., 2015.

[7] J. Granjal, E. Monteiro, and J. Sa Silva, "Security for the internet of things: a survey of existing protocols and open research issues," *IEEE Communications Surveys & Tutorials*, vol. 17, no. 3, pp. 1294–1312, 2015.

[8] H.-J. Liao, C.-H. R. Lin, Y.-C. Lin, and K.-Y. Tung, "Intrusion detection system: a comprehensive review," *Journal of Network and Computer Applications*, vol. 36, no. 1, pp. 16–24, 2013.

[9] I. Butun, S. D. Morgera, and R. Sankar, "A survey of intrusion detection systems in wireless sensor networks," *IEEE Communications Surveys & Tutorials*, vol. 16, no. 1, pp. 266–282, 2014.

[10] R. Mitchell and I.-R. Chen, "A survey of intrusion detection techniques for cyber-physical systems," *ACM Computing Surveys*, vol. 46, no. 4, article 55, 2014.

[11] R. Berthier and W. H. Sanders, "Specification-based intrusion detection for advanced metering infrastructures," in *Proceedings of the 17th IEEE Pacific Rim International Symposium on Dependable Computing, PRDC '11*, pp. 184–193, IEEE, Califorina, Calif, USA, 2011.

[12] P. Jokar, H. Nicanfar, and V. C. M. Leung, "Specification-based intrusion detection for home area networks in smart grids," in *Proceedings of the IEEE 2nd International Conference on Smart Grid Communications (SmartGridComm '11)*, pp. 208–213, IEEE, Brussels, Belgium, October 2011.

[13] S. Raza, L. Wallgren, and T. Voigt, "SVELTE: real-time intrusion detection in the internet of things," *Ad Hoc Networks*, vol. 11, no. 8, pp. 2661–2674, 2013.

[14] M. Surendar and A. Umamakeswari, "InDReS: an intrusion detection and response system for internet of things with 6LoWPAN," in *Proceedings of the 2016 IEEE International Conference on Wireless Communications, Signal Processing and Networking, WiSPNET '16*, pp. 1903–1908, IEEE, Chennai, India, 2016.

[15] D. Shreenivas, S. Raza, and T. Voigt, "Intrusion detection in the RPL-connected 6LoWPAN Networks," in *Proceedings of the 3rd ACM International Workshop on IoT Privacy, Trust, and Security, IoTPTS '17*, pp. 31–38, ACM, Abu Dhabi, UAE, 2017.

[16] A. Le, J. Loo, A. Lasebae, M. Aiash, and Y. Luo, "6LoWPAN: a study on QoS security threats and countermeasures using intrusion detection system approach," *International Journal of Communication Systems*, vol. 25, no. 9, pp. 1189–1212, 2012.

[17] T. Lee, C. Wen, L. Chang, H. Chiang, and M. Hsieh, "A lightweight intrusion detection scheme based on energy consumption analysis in 6lowpan," in *Advanced Technologies,*

Embedded and Multimedia for Human-centric Computing, pp. 1205–1213, Springer, Dordrecht, Netherlands, 2014.

[18] H. A. Abdullah, M. M. El-ajaily, E. E. Saad, A. A. Janga, A. A. Maihub, and J. Pharm, "Preparation, spectroscopic investigation and biological evaluation of schiff's base La(III), Zr(IV)and Ce(IV) chelates," *International Journal of Pharmaceutical and Chemical Sciences*, vol. 3, no. 2, p. 143, 2014.

[19] P. Kasinathan, C. Pastrone, M. A. Spirito, and M. Vinkovits, "Denial-of-Service detection in 6LoWPAN based Internet of Things," in *Proceedings of the 2013 IEEE 9th International Conference on Wireless and Mobile Computing, Networking and Communications, WiMob '13*, pp. 600–607, IEEE, Lyon, France, October 2013.

[20] A. Rghioui, A. Khannous, and M. Bouhorma, "Monitoring behavior-based intrusion detection system for 6lowpan networks," *International Journal of Innovation and Applied Studies*, vol. 11, no. 4, p. 894, 2015.

[21] A. Dunkels, B. Grönvall, and T. Voigt, "Contiki—a lightweight and flexible operating system for tiny networked sensors," in *Proceedings of the 29th Annual International Conference on Local Computer Networks (LCN '04)*, pp. 455–462, IEEE, Florida, Fla, USA, November 2004.

[22] M. Durvy, J. Abeillé, P. Wetterwald et al., "Making sensor networks IPv6 ready," in *Proceedings of the 6th ACM Conference on Embedded Networked Sensor Systems, SenSys '08*, pp. 421-422, ACM, North Carolina, NC, USA, November 2008.

[23] A. Dunkels, J. Eriksson, N. Finne, and N. Tsiftes, *Powertrace: Network-Level Power Profiling for Low-Power Wireless Networks*, 2011.