Advanced Networked Agents for Security and Trust Assessment in CPS/IoT Architectures

D2.6

Attacks Threats Analysis and Contingency Actions Final Report

This deliverable presents the results of the first 26 months of work concerning the task 2.6.

Distribution level	CO
Contractual date	28.02.2019 [M26]
Delivery date	28.02.2019 [M26]
WP / Task	WP2 / T2.2
WP Leader	CNR
Authors	I. Vaccari (CNR), E. Cambiaso (CNR), S. Scaglione (CNR), E. Punta (CNR), S. Vuppala (UTRC), P. Sobonski (UTRC), D. Rivera (MONT), R. Trapero (ATOS), M. Bagaa (AALTO)
EC Project Officer	Carmen Ifrim carmen.ifrim@ec.europa.eu
Project Coordinator	Softeco Sismat SpA Stefano Bianchi Via De Marini 1, 16149 Genova – Italy +39 0106026368 stefano.bianchi@softeco.itstefano.bianchi@softeco.it
Project website	www.anastacia-h2020.eu

ANASTACIA has received funding from the European Union's
Horizon 2020 research and innovation programme under Grant Agreement N° 731558
and from the Swiss State Secretariat for Education, Research and Innovation.

Table of contents

PUBLIC SUMMARY

This document represents an extension of the work reported on the ANASTACIA D2.2 deliverable document. Concerning the ANASTACIA platform, infrastructure protection is a critical task, since it is crucial to defend the underlying system to prevent malicious activities on the network. In this context, our work reported in this document reports information on how to manage the use cases introduced in the D1.2 deliverable and not managed in the D2.2 document, hence focusing on the study of novel cyber-threats and on the protection from cyber-attacks for the ANASTACIA scenario. The aim of this document is indeed to analyse and to address two innovative threats able to exploit IoT and smart devices. Such two novel cyber-attacks are referred as IoT 0-day and Slow DoS Attack (SDA).

The first attack focuses on the IoT context, in particular IoT networks making use of the ZigBee protocol, and exploits a novel vulnerability of ZigBee enabled devices, providing an attacker the ability to control network nodes. For this threat, the attack is described in deep, as well as the proposed defence system implemented. Instead, concerning the SDA scenario, we propose a novel attack, by describing it in detail, report the obtained results after perpetrating the attack against real test servers, and describe algorithms and techniques we propose to protect a network system from this kind of attacks.

The ANASTACIA framework also includes a behavioural analysis system: in this document, the features developed and how it is integrated within the phase of identification of threats is also described. Furthermore, we also focus on the current state of the mitigation and contingency systems of the ANASTACIA framework. Finally, we describe appropriate approaches to integrate these two innovative attacks within the ANASTACIA framework, to provide detection and mitigation capabilities.

1 Introduction

1.1 Aims of the document

The aim of this document is mainly to analyze innovative threats considered for the ANASTACIA platform. For each threat, the document reports detailed information about the innovative attack, a detection and mitigation approaches are proposed in order to be implemented and deployed on the ANASTACIA platform. The document also reports a detailed description of the innovative attack with preliminary results. Finally, the document proposed algorithms and protection system able to mitigate the innovative threats analyzed in the document.

1.2 Applicable and reference documents

This document refers to the following documents:

- ANASTACIA deliverable D1.2 “User-centered Requirement Initial Analysis”
- ANASTACIA deliverable D2.2 “Attacks Threats Analysis and Contingency Actions Initial Report”
- ANASTACIA deliverable D4.1 “Initial Monitoring Component Services Implementation Report”
- ANASTACIA deliverable D4.3 “Initial Agents Development Report”
- ANASTACIA deliverable D6.2 “Initial use cases implementation and tests Report”

1.3 Revision History

Version	Date	Author	Description
v0.1	10/01/2019	I. Vaccari	Initial draft
v0.2	11/01/2019	E. Cambiaso	First revision and integration
v0.3	25/01/2019	E. Cambiaso	Inclusion of technical implementation details
v0.4	05/02/2019	I. Vaccari	Revision activities
v0.5	06/02/2019	E. Cambiaso	Minor readjustments
v0.6	08/02/2019	E. Cambiaso	Review of the chapters
v0.7	08/02/2019	I. Vaccari, P. Sobonski	Re-arrangement of the structure of the document
v0.8	11/02/2019	E. Cambiaso	Minor fixes
v0.8.1	14/02/2019	S. Vuppala, P. Sobonski	Adding UTRC contribution
v0.8.2	15/02/2019	D. Rivera	Adding MONT contribution
v0.8.3	19/02/2019	R. Trapero	Adding ATOS contribution

v0.9	19/02/2019	I. Vaccari	Addressing comment and adding public summary and conclusions
v1.0	21/02/2019	M. Bagaa	Adding AALTO contribution
v1.1	21/02/2019	E. Cambiaso	Minor revisions and amendments
v1.2	26/02/2019	E. Cambiaso, I. Vaccari	Amendments after internal review, plus consideration of use cases
v1.3	28/02/2019	E. Cambiaso, I. Vaccari	Added final contributions by UTRC and MONT
v1.4	28/02/2019	I. Vaccari	Updated document template

1.4 Acronyms and Definitions

Acronym	Meaning
CP	Constraint Programming
DDoS	Distributed Denial of Service
DoS	Denial of Service
DPI	Deep packet inspection
HIDS	Host-based Intrusion Detection System
IDS	Intrusion Detection System
IoT	Internet of Things
IPS	Intrusion Prevention System
MITM	Man-in-the-middle
MMT	Montimage Monitoring Tool
NIDS	Network Intrusion Detection System
SDA	Slow DoS Attack
SQLi	SQL injection

2 ATTACKS SCENARIOS

By analyzing the attacks scenario described in the D1.2 and the threats considered in D2.2, some of the attacks were considered in the D1.2 document, although not integrated in the ANASTACIA platform, hence, not exhaustively described in the D2.2 document.

Concerning the previously considered attacks, Table 1 reports an update of detection and mitigation strategies adopted.

Attack scenario	Detection plan	Mitigation plan
MEC.3 "DoS and DDoS attacks using smart cameras and IoTs"	• Traffic analysis inside the IoT network • Detection of unusual ICMP traffic.	• Deployment of firewall rule to filter the malicious attack.
BMS.2 "Insider attack to a fire suppression system"	• Token validation	• VNF deployment • IoT honeynet • Transparent forwarding
BMS.3 "Remote attack to an energy microgrid"	• Traffic analysis on the incoming link of the server using DPI. • Detection of SQL queries in the requests	• Deployment of firewall rule to filter the traffic from the attacker.
BMS.4 "Cascade attack on a megatall building"	• Buffered sensor data from smart buildings • Detection of misbehaviour of the system • Enabling of continuous and integrated monitoring of multivariate signals, event logs, heartbeat signals, status reports, operational information, etc.	• Block of the adversary, based on VDSS feedback • Restore of sensors data (back process)

Table 1: Detection and mitigation strategies adopted for the attacks considered in the D2.2 deliverable document

The aim of this section is to consider the remaining threats, that are related to the following use cases:

- O.1 "Overarching use case"
- MEC.1 "Spoofing attack to a smart security camera system"
- MEC.2 "Man-in-the-middle attack to a smart security camera system"
- MEC.4 "IoT-based attack in the MEC Scenario"
- BMS.1 "Cyber-attack to a hospital building management system"

By considering such threats, we will now describe how to protect from them.

2.1 USE CASE O.1

This scenario is focused on the execution of a malicious attack exploiting CCTV security cameras to compromise data confidentiality. In this case, once the attack is accomplished, the network is characterized by the presence of anomalous traffic, representing the effect of the attack.

2.1.1 Detection plan

Detection of the attack can be accomplished by monitoring the network traffic generated by the CCTV cameras to identify anomalous conditions. This can be done by implementing anomaly-based detection algorithms.

2.1.2 Mitigation plan

Once the attack is detected, mitigation activities may be deployed at two different levels: at the network level, it is possible to temporarily block communications (or connections involving external unwanted nodes/services) of the affected devices. A similar approach working at SDN level may isolate the affected nodes and redirect their network traffic to a secure internal location. At the host level, it is possible to reconfigure the hosts to temporarily interrupt communications (by keeping images and videos retrieval from the environment). Although a more practical solution may trigger self-reboot or self-shutdown activities on the hosts, this may not be a good solution, since it would induce some sort of denial of service, hence, promoting a cyber-physical threat.

2.2 Use Case MEC.1

This scenario is focused on the execution of a spoofing attack, aimed to impersonate (at the IP layer) a smart security camera data collector system to retrieve sensitive videos. The aim of the attacker is indeed to collect videos coming from the security cameras distributed on the territory, for blackmail purposes.

2.2.1 Detection plan

It is possible to detect the attack by also accomplishing SNMP data analysis and by designing proper routes configuration aimed to simplify detection of traffic from suspicious sources. In particular, by spoofing the IP address of the server, the attacker is supposed to adopt a different (and potentially suspicious) route.

2.2.2 Mitigation plan

In order to protect the system from the attack, it is possible to adopt the following approaches:

- Data and communication encryption, coupled with authentication methods, adopted for communications between the smart security cameras and the collector service
- Proper routes configuration and packets filtering techniques to block traffic from suspicious sources
- Update of the IP address of the collector server, with consequent update of the related references on the hosts communicating with the server

2.3 Use Case MEC.2

This scenario is focused on the execution of a man-in-the-middle attack against security cameras. The attacker is in this case an insider/employee of the targeted company, and his aim is to retrieve sensitive videos to store them illegally and/or share them outside of the network (e.g. on personal servers located in the house of the insider).

The attack is accomplished by exploiting credentials, certificates and video decryption keys owned by the employee/attacker. In addition, the attack exploits a man-in-the-middle approach to impersonate the smart camera management server system, to the eyes of the security cameras. Such server is supposed to retrieve videos from the security cameras.

2.3.1 Detection plan

Detection of the MEC.2 attack may be accomplished by adopting and possibly combining different approaches:

- By logging certificates adoption to check validity, origin and owner of the certificates
- By monitoring host-to-host communications of the security cameras (e.g. at IP or MAC levels), in order to identify unexpected data shares
- By analyzing network communications (e.g. through appropriate NIDS) to identify man-in-the-middle attacks exploiting for instance ARP tables
- By analyzing of communications and related flows (we suppose that videos are shared/exfiltrated outside of the organizations by exploiting the network, and by avoiding low-rate transfers), it is possible to identify anomalous traffic generated from the host operating as the MITM; similarly, behavioural analysis activities may be adopted to detect man-in-the-middle attacks through the use of constraint programming techniques

2.3.2 Mitigation plan

It is possible to mitigate the attack by working on the firewall devices, by limiting by design the communications of the security cameras, only allowing them to communicate with the legitimate smart security camera server.

Regarding data exfiltration/sharing outside of the organization, it's possible to block connectivity of the affected devices, once transfers are identified. Also, it is possible to avoid packets encapsulation (used for tunneling purposes), by accomplishing deep packet inspection.

2.4 Use Case MEC.4

This scenario is focused on the exploitation of vulnerabilities affecting IoT camera systems, aimed to make the camera perpetrate malicious cyber-attacks against third parties. Such malicious attacks include DoS, scanning, or other well-known threats.

2.4.1 Detection plan

Detection of the attack may involve two different temporal periods:

- At exploitation time, in case a known-vulnerability is exploited, it is possible to identify (and block) exploitation
- At post-exploitation time, hence, only after the IoT cameras are exploited, it is possible to identify running attacks (generated by the cameras themselves), by analyzing network traffic flows and communications, in order to identify known threats (for instance, through signature-based detection, combined with DPI), or unknown threats (for instance, by adopting network anomaly-based IDS)

2.4.2 Mitigation plan

Mitigation may be accomplished by blocking outgoing traffic from the affected IoT cameras, or by diverting malicious traffic to harmless locations (under the control of the network administrator) through network reconfiguration accomplished through SDN/NFV approaches.

Moreover, it is important to consider that, in order to identify known threats, periodic vulnerability assessment activities may be executed to identify potential threats and targets of an attack, in order to patch vulnerabilities.

2.5 Use Case BMS.1

This scenario is focused on the execution of an advanced and combined attack based on the exploitation of a 0-day vulnerability, granting the attacker access to the target network. Once access is obtained, different "actions" are accomplished (activation of emergency in several floors of the building, switch-off of emergency units, overwrite of heating and cooling configurations, etc.), also including the gaining of physical unauthorized access to the facilities, needed to install malicious applications on specific network nodes, making them exfiltrate sensitive data outside of the organization and, simultaneously, perpetrate network attacks (e.g. SQLi) against document management services.

2.5.1 Detection plan

In this case, the following statements aspects can be considered for detection purposes:

- By definition, it is not possible to detect 0-day vulnerabilities exploitation. Nevertheless, it is possible to deploy anomaly-based NIDS (making use for instance of machine learning methods) to identify anomalies on the network
- The different "actions" executed by the attacker can be detected by implementing and deploying appropriate logging systems
- The possibility to obtain physical unauthorized access to the facilities can be detected by implementing proper authentication and access control lists on the services adopted for access management and access to physical locations, combined with a physical identification of intrusions through the adoption of physical security systems
- Exfiltration outside the organization of sensitive data may be identified by deploying anomaly-based NIDS, designed to detect anomalous traffic on the network
- Running attacks (e.g. SQLi) can be identified by NIDS through DPI approaches

2.5.2 Mitigation plan

Mitigation of the advanced attack considered can be accomplished as follows:

- The different "actions" executed by the attacker can be mitigated by restoring systems and configuration to previous states
- The possibility to obtain physical unauthorized access to the facilities can be prevented by implementing proper authentication and access control lists on the services (for instance, also considering timing accesses), adopted for access management and access to physical locations
- Exfiltration outside the organization of sensitive data may be mitigated by blocking or redirecting network communications
- Running attacks (e.g. SQLi) can be mitigated at the network level, by NIDS and/or by redirecting the network traffic to harmless nodes

3 THREATS ANALYSIS PROCESS

By continuing the work carried out in the ANASTACIA deliverable D2.2, we have worked to deeply investigate the cyber-security field. In this context, once the approach for each of the considered tests cases has been defined (in the deliverable D2.2), we focused on analyzing last generation cyber-threats that are interesting to the ANASTACIA scenarios, due to the considered context, the dangerousness of the threats, or the sensitivity of the targets.

In particular, given the emergence of the Internet of Things context, related to the ability to provide network connectivity to environmental sensors, IoT based networks will be present in many and many networks in the next future. Thanks to IoT, simple objects have the ability to transmit, elaborate and collect data so they can communicate with each other or with human beings, monitoring and controlling the surrounding environment. IoT devices are adopted in different contexts (smart home, smart building, healthcare, smart city and mobility, etc.) to monitor the environments in which they are installed, so the information recovered are very sensitive and for this reason it is important to analyze the security aspects associated with this technology [Vaccari, 2019]. In this case, our work was focused on deeply analyze the topic, by first investigating the field at a higher layer, hence focusing on wireless communications, widely adopted, for instance, in home automation contexts. In this case, although several different protocols are available, we focused on the ZigBee protocol and its security, by studying the protocol and analyzing communication flows, considering the different categories of packets that ZigBee is able to manage. Our deep analysis led to the identification of an innovative attack exploiting the MAC layer of the ZigBee protocol stack and affecting specific categories of devices supporting ZigBee [Vaccari, 2017]. Although the scenario is limited to a specific scenario, our work led to the identification of a novel threat in the IoT security topic and to design appropriate protection systems. This investigation is also important for the ANASTACIA platform development, since it can provide the platform the ability to protect IoT networks from the proposed last generation attack.

In parallel, we have also worked to investigate the network security topic, concerning in particular the design and development of last generation denial of service attacks. The CNR team involved in the ANASTACIA project is a pioneer in the Slow DoS Attack context, related to innovative DoS attacks characterized by low-rate attack bandwidth requirements. By exploiting the previously acquired knowledge, further work on the topic was accomplished, contextualizing it on the ANASTACIA platform and service applications. In particular, we proposed a novel cyber-attack called SlowComm [Cambiaso, 2017], adopting minimum amounts of bandwidth to accomplish its purposes. Such characteristic is particularly interesting and makes it more challenging to efficiently identify an attack, due to the low amount of traffic generated.

By adopting such approach, we achieved important results both from the scientific and project point of views. By expanding the work carried out concerning test cases focused on existent threats (and described in the deliverable D2.2), we have now the possibility to integrate innovative attacks and to design and develop appropriate countermeasures providing ANASTACIA the ability to protect network-based systems not only from state-of-the-art threats, but also from extremely innovative attacks.

4 Overview of the Considered Innovative Attacks

Motivated by the evolution of threats against system and infrastructure, the ANASTACIA consortium recently started to explore new threats on cyber physical devices and systems. The current ANASTACIA use cases tackle the threats from other perspectives. In fact, our use cases need to consider the recent threats of cyber physical systems. For example, the adoption of future communication i.e. 5G significantly poses new threats in the system. Similarly, the advanced persistent threats have not been analyzed yet in current context. Therefore, the following advanced threats have been selected from the context of ANASTACIA use cases, and will be addressed using ANASTACIA platform. For this purpose, two innovative threats were studied and analyzed: IoT 0-day vulnerability and Slow DoS attack, described in detail in this section.

Generally speaking, a 0-day vulnerability is a security flaw which doesn't have a patch in place or a fix available. It has the potential to be exploited by cyber-criminals. The term 0-day refers indeed to a newly discovered vulnerability without any patch or fix available. Once the vulnerability becomes publicly known, the vendor usually has to quickly fix the issue to protect its users. This is not always accomplished, especially considering low-cost devices (like the ones we have analyzed) and the dynamic behavior/stability that contextualizes the companies behind IoT products.

Concerning instead the Slow DoS Attack scenario, it refers to a denial of service (DoS) attacks, they are executed with the aim to make a network service unavailable to its intended users. Such threats are considered particularly dangerous, due to the limited amount of network traffic generated.

4.1 IoT 0-day attack

Internet of Things (IoT) is a recent and emerging phenomenon that allows common use devices to communicate on the Internet. Although IoT devices are not widely adopted yet, it is assumed that, by 2020, approximately 24 billion of IoT devices will be online [Gubbi, 2013]. Such devices provide the ability to automate daily life activities such as turning lights on automatically when the user reaches home, considering a domestic context, in smart-city scenarios, or to improve productivity, on an Industry 4.0 environment. Being a pervasive technology embedded on critical locations, the IoT phenomenon is often coupled with privacy issues: as such sensors often process sensitive information, security becomes a very important topic.

Considering the domestic and smart-city contexts in particular, Internet of Things devices may communicate through standard networks, such as Wi-Fi or Ethernet, or build a dedicated network to communicate with other sensors, hence creating a network called Wireless Sensor Network (WSN). In this regard, a real common and universal standard has not yet been defined [Yick, 2008]. Currently, there are different protocols providing communication between sensors: some of them are based on pre-existing protocols (Wi-Fi, 6LowPAN or LoRa), while others provide the creation of a new ad-hoc infrastructure (ZigBee, Z-Wave).

Although different IoT protocols may be adopted, IoT devices are often exposed to security attacks, due to their limited functionalities (e.g. power consumption, computational capabilities). Therefore, the IoT security topic is extremely critical: let's think for instance to a temperature sensor installed on a home environment. Although at first sight, security may not appear a crucial element (in case of data leak, people may think that the house temperature may not be relevant to an external user), it is actually a critical topic (a malicious external user accessing leaked data may derive that if the house temperature during daylight is lower/higher than a specific threshold, then nobody is in the house).

Being exchanged information extremely sensitive, due to the nature of IoT devices and networks, security of IoT systems is a topic to be investigated in deep. The work behind the proposed attack goes in this direction, by investigating the home automation IoT context in particular and exploiting its components, in order to identify weaknesses that attackers may exploit.

4.1.1 Attack description

The proposed attack is part of the ZigBee security context. ZigBee is a wireless standard introduced by the ZigBee Alliance in 2004 and based on the IEEE 802.15.4 standard, used in the Wireless Personal Area Networks (WPAN) context [Ramya, 2011]. A typical ZigBee network is composed by three main devices: a Coordinator, a Router and End-device. The coordinator is the node in charge of create the ZigBee network and to elaborate each packet exchanged in the network. A router is adopted to extend the range of the ZigBee network (similar to the router adopted in Wi-Fi protocol). Finally, the end-device is the sensor installed in the field able to retrieve information about the areas monitored. The packets flow starts from the end-device that retrieves the information and sends to the coordinator, eventually through a router.

In order to properly investigate ZigBee security, we studied the protocol and analyzed communication flows, considering the different types of packets supported by ZigBee. While, at first, we focused on packets containing data sent from the coordinator to the end-device, later, we also analyzed other packets exchanged in the network. In this context, we found that, at the MAC layer of the ZigBee protocol stack, it is possible to send Remote AT Commands. By working at such lower layer, received packets are not processed at the application layer; hence, it may not be possible to access the packet content to avoid interpretation, except from the device manufacturer.

During this research work, we identified a particular vulnerability affecting AT Commands capabilities implemented in IoT sensor networks. The work focuses on the exploitation of such weakness. AT Commands are specific packets, historically adopted by old generation modems to interface with the device, today used by radio modules such as XBee [Piyare, 2013], ESP8266 [Thaker, 2016], or ETRX3 [Kirichek, 2015] to configure parameters such as connection type, network identifier, device name on the network, or destination address for a communication. AT Commands are today supported by many devices of different nature, providing different functionalities and hence commands. For instance, modules that provide connectivity support AT Command packets for network parameters configuration, while other modules may use these packets to alter light intensity of light bulbs. As an example, the following AT command is used to send a SMS message from a GSM phone.

```
AT+CMGF=1
OK
AT+CMGS="+31628870634"
> This is the text message.→
+CMGS: 198
OK
```

For this research, we focused on XBee modules. Such modules, widely adopted around the world, especially in Do-It-Yourself (DIY) contexts, implement two different AT Command packets, related to request and response operations, respectively. Concerning XBee modules, these packets can be sent remotely: we talk in this case of Remote AT Commands. Such packets belong to the (IEEE 802.15.4) MAC layer and they are interpreted by the (XBee) module automatically. Therefore, by being such interpretation demanded to the device firmware, and being such firmware provided by the manufacturer, Digi International, it is not possible to avoid implicit Remote AT Commands interpretation. In order to execute the proposed attack, the AT Command functionality of XBee has to be exploited. XBee supports several AT Command packets. Particularly, for our aim, we have used ATID commands to target sensors (in general, other commands/approaches may be used for different purposes: e.g., to make the sensor join a different network, to forward (sensitive) data to a different malicious receiver, and to disable data encryption). ATID is used by XBee modules to set the network identifier. During the proposed Remote AT Command attack, the malicious user sends an ATID packet with a bogus identifier in order to make it join a different (inexistent, in our case) network.

In order to maliciously exploit Remote AT Command, it is assumed that the attacker is connected to the network of the target. In this case, the enemy may, for instance, disconnect an end-device from the ZigBee network and make it join a different (malicious) network and hence forward potentially sensitive data to third malicious parties. Given the nature of IoT end-devices, often associated with a critical data and operations, it may be obvious how a Remote AT Command attack represents a serious threat for the entire infrastructure.

Early evaluation of the effects of the proposed attack on a real network led to validate the success of the proposed threat [Vaccari, 2017]. For our tests, as previously mentioned, the aim of the attacker is to disconnect the nodes of the network, by exploiting the identified threat to reconfigure them remotely, demanding them to join a different (not existent) network.

In order to evaluate the attack effects, we have configured a real network composed of real sensors communicating and spreading information to a coordinator node. Sensors are composed by an Arduino Uno microcontroller, connected to an XBee module and simulating data retrieval from the environment. Data are sent through the ZigBee protocol to a network coordinator, composed of a Raspberry PI device equipped with an XBee module. Under normal conditions, without any running attack, network nodes communicate on the network, for instance to spread information measured from the environment. Figure 1 reports a network traffic capture of a network node communicating normally on the network, in a situation without running attacks.

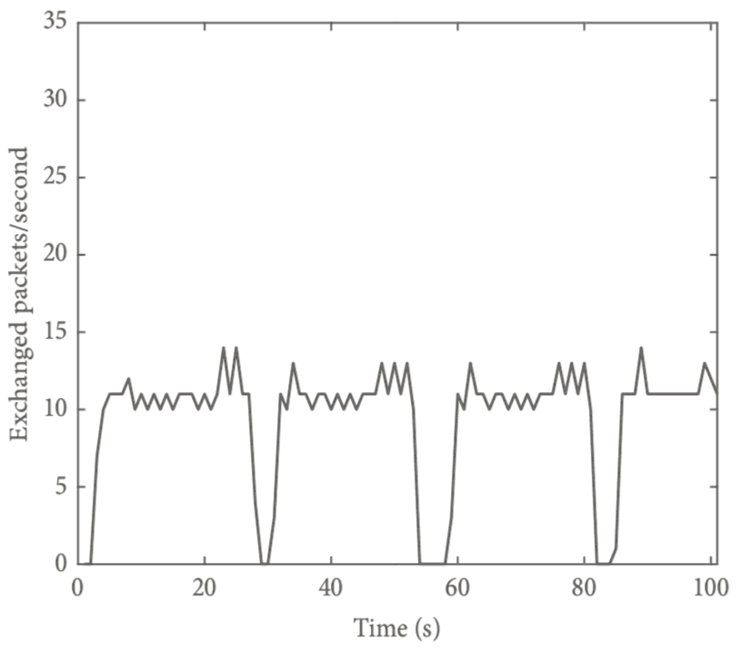

Figure 1 - Normal traffic flow

Instead, during a running attack, the end-device would implicitly interpret the received command, hence reconfiguring themselves as specified by the attacker. In this case, the effects are shown in the Figure 2, reporting the results of a network traffic capture during a running attack.

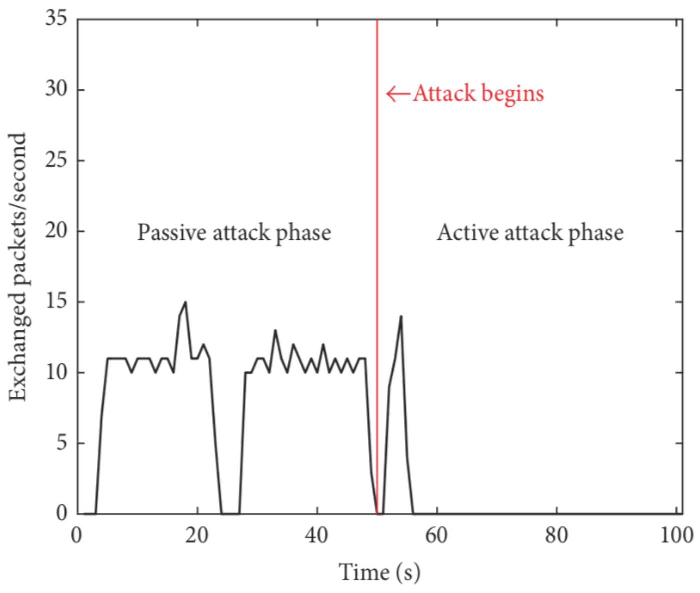

Figure 2 - Traffic flow with Remote AT Command in execution

As it is possible to notice, before the attacker starts communicating on the network, the node is working normally. Instead, just a few seconds after the beginning of the attack, the targeted node interrupts communications and it is therefore unable to communicate on the same channel.

These results prove the efficacy of the proposed attack. Moreover, since just a single packet is sent to the victim by the attacker in order to reconfigure it, the proposed attack should be considered as dangerous as scalable. Particularly, the time required to send such packet is minimal, so in case of multiple targeted sensors, the attack success is guaranteed.

4.2 SLOW DOS ATTACKS

Over the last few years, Internet has increasingly become the most important communication medium of the world. Because of this, Internet must remain a safe place, in order to provide a secure communication environment to its users.

Among all the methodologies used to successfully execute malicious cyber-operations, denial of service attacks (DoS) are executed with the aim of exhaust victim's resources, compromising the targeted systems' availability, thus affecting availability and reliability for legitimate users. These threats are particularly dangerous, since they can cause significant disruption on network-based systems [Beitollahi, 2011]. Considering old-style DoS threats, there are essentially two types of DoS attacks: while vulnerability/exploit-based attacks can exhaust the resources of a server by exploiting a vulnerability in the software, flooding based attacks simply send to the victim a high volume of data, which cannot be handled by the target.

Exploit based attacks (also known as vulnerability, software, or semantic attacks) may involve a few well-crafted packets in the attack. As they exploit a specific vulnerability of the software, they are quite simple to mitigate by patching the vulnerability on the victim host or by identifying the packets related to the attack and handling them separately, or simply dropping them. In general, signature-based approaches are therefore adopted in this case to counter such threats.

Instead, flooding based attacks overwhelm the victim's resources by sending a large amount of data. If DoS attacks are launched from a single machine, it is possible to protect from such attacks by equipping the victim with abundant resources. In this way, the attacker would need a better equipped machine in order to successfully perform the attack. Instead, if a distributed attack (DDoS) is accomplished, the attacker adopts a set of machines/bots under its control, whose resources overwhelm in total the resources of the server. In this case, it is more difficult for the victim to counter a running attack.

The term Slow DoS Attack (SDA), coined by the CNR Network Security research group involved in the project, concerns a DoS attack which makes use of low-bandwidth rate to accomplish its purpose. An SDA often acts at the application layer of the Internet protocol stack because the characteristics of this layer are easier to exploit in order to successfully attack a victim even by sending it few bytes of malicious requests. Note that the term slow does not necessarily imply that SDAs 'send data slowly': even if in some cases this may be true, the term slow historically comes from a famous attack known as slowloris [Cambiaso, 2012]. Similar definitions are given to low-rate DoS (LDoS), application layer DoS or (in case that either HTTP or XML protocols are exploited) HTTP/XML DoS attacks.

Relatively to the purpose of an SDA, this is often to cause unavailability of a victim host by seizing all the connections available at the application level of the victim. Under the SDA condition, all service queues are busy and any new incoming request from legitimate users is discarded (the saturation state is reached), therefore causing a DoS. The attacker attempts to force the server to process only his own requests by filling the service queues only with attack message requests.

Assume that somehow the attacker manages to make the server reach the saturation state. In this situation, whenever a position is freed in any of the service queues, the attacker attempts to take it again, before any other (legitimate) user can. Thus, the aim of the attacker is to achieve the maximum number of positions in the service queue, at a high-rate, thus maximizing the probability of seizing all the positions as quickly as possible. However, it is desirable to use low-rate traffic instead, for two main reasons: because this approach allows the attack to be carried out with much fewer resources, and also because it can more easily bypass protection mechanisms that rely on the statistical detection of high-rate traffic.

An SDA may also exhibit an ON-OFF nature [Cambiaso, 2013], which comprises a succession of consecutive periods composed of an interval of inactivity (called off-time), followed by an interval of activity (called on-time). This characteristic may add to the seriousness of the threat, since the attack traffic assumes a form which is statistically similar to the one of legitimate traffic; moreover, servers implicitly support users with slow or intermittent connection bandwidth.

Once the SDA has seized all available positions in the service queues, the attacker slows down the connections from or to the victim by exploiting the characteristics of either a specific protocol (i.e., HTTP, FTP, DNS, etc.) or the application software (i.e., PHP, SOAP, etc.). The connections are thus kept active as long as possible, by sending minimum amounts of data per time unit.

4.2.1 Attack description

The innovative attack proposed is called SlowComm [Cambiaso, 2017]. The attack belongs to the Long Request DoS attack category of Slow DoS Attacks [Cambiaso, 2013]. Particularly, SlowComm sends a large amount of slow (and endless) requests to the server, saturating the available connections at the application layer on the server inducing it to wait for the (never sent) completion of the requests. As example, we refer to the HTTP protocol, where the characters sequence `\r\n\r\n` represent the end of the request: SlowComm never sends such characters, hence forcing the server to an endless wait. Additionally, during a SlowComm the request payload is sent abnormally slowly. Similar behavior could be adopted for other protocols as well (SMTP, FTP, etc.). As a consequence, by applying this behavior to a large amount of connections with the victim, a DoS may be reached. The requests sent to the server must be considered slow in terms of "little amounts of bytes sent per second". Indeed, for the proposed attack this value approaches an extremely low value: the attack bandwidth requirements for the executed tests are less than 1 KB/s.

Analyzing how the attack works, since the attacker's purpose is to seize, as soon as possible, all the available connections on the targeted host (saturation state), we could assume that a DoS is reached some instants after the attack is launched. Nevertheless, the server may have already established active and legitimate connections with other clients. Those connections are working until they are closed. As soon as a connection closure happens, its relative resources on the server are released, thus allowing clients to establish new incoming connections. Because of this, the purpose of the attacker is to replace all the "just available" connections with malicious ones. While doing that, there could be a race condition between attacker and some other legitimate clients. Nevertheless, we could assume that sooner or later the attacker would obtain the connections, since it would repeatedly try to connect to the victim with an intelligent algorithm, tuning aggressiveness and stealthiness of the attack.

The SlowComm attacks works by creating a set of predefined connections with the victim host. For each connection, a specific payload message is sent (the payload is typically endless), one character at time (one single character per packet), by making use of the Wait Timeout [Cambiaso, 2012] to delay the sending. In this way, once the connection is established with the server (at the transport layer), a single character is sent (hence, establishing/seizing the connection at the application layer, hence, with the listening daemon). At this point, the Wait Timeout is triggered, in order to delay the sending of the remaining payload, and to prevent server-side connection closures. The value of such timeout depends indeed on the server capabilities, but it is even supposed to be equal to a few minutes. Hence, considering a single connection, the attacker would not communicate with the server (at the application layer, while at lower layers, keep-alive messages may be sent) for minutes (ON-OFF behavior). Although this may appear unusual, hence, easy to detect, this is a common behavior for several protocols. For instance, users connected to a remote shell may be inactive (hence, without sending any data) even for hours, keeping connection alive and the shell ready to receive commands. Because of the ON-OFF behavior of SlowComm, from one side, attack bandwidth requirements are reduced. From the other side, it may be difficult to identify a running attack, since it may be mixed with legitimate traffic that is supposed to allocate more network bandwidth.

By considering an attack execution against an Apache2 HTTP web server, it is important to consider that, by default, such server is configured to serve at most 150 simultaneous connections[1]. We have measured how it is possible to successfully lead a DoS on the victim. In particular, we have considered three different cases:

1. In case no modules are installed on the server, once the DoS is reached, it is continuously maintained.
2. If the `mod-security` module (a module aimed to limit the maximum number of simultaneous connections of the same client) is installed on the server, the DoS is never reached with a single host, since connections from the attacking node are limited.
3. If the `reqtimeout` module (a module aimed to close connections that do not respect minimum bandwidth requirements) is installed on the server, an alternation of DoS and non-DoS states occurs.

Results are visible in Figure 3.

[1] For last versions servers, this value is increased to 256. More information are available at the following address: https://httpd.apache.org/docs/current/mod/mpm_common.html#maxrequestworkers

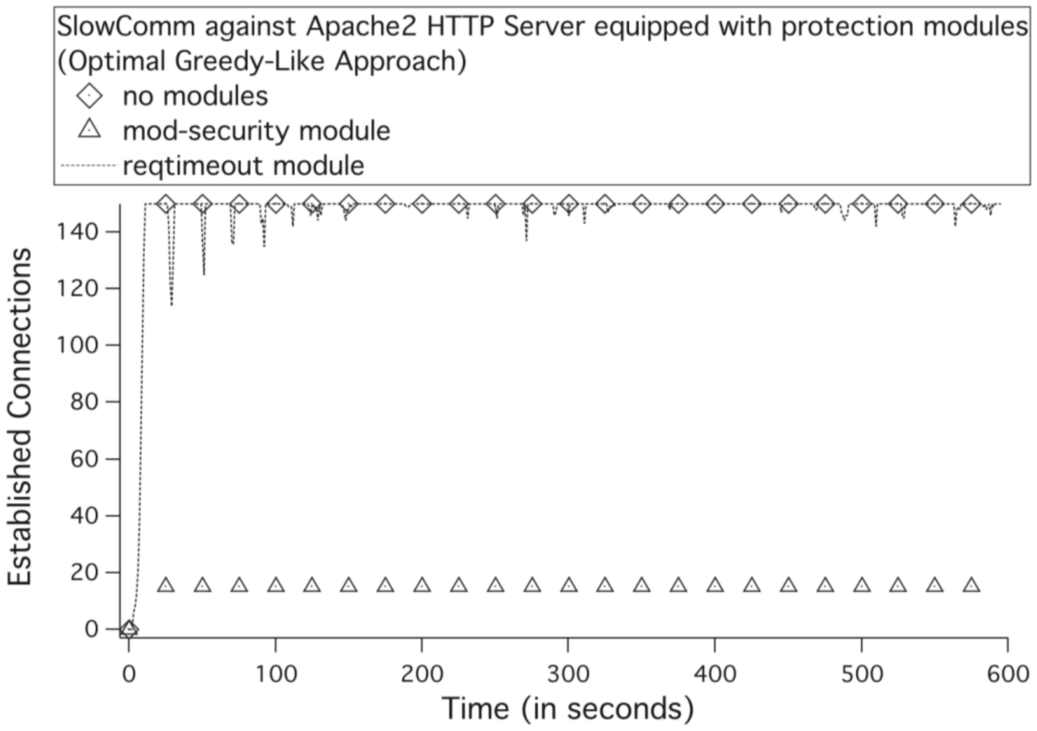

Figure 3 - Results of SlowComm attack against Apache2 HTTP Server

It's also important to notice that figure shows how the DoS is reached just after a few seconds from the beginning of the attack. Concerning instead the mod-security scenario, a distributed attack is able to lead a DoS with ease.

Since the SlowComm attack is characterized by a protocol independence feature [Cambiaso, 2013], we have also executed attacks on protocols different from HTTP. In this case, it is important to consider that the same payload of the previously mentioned tests is sent during the attacks. Results are reported in Figure 4.

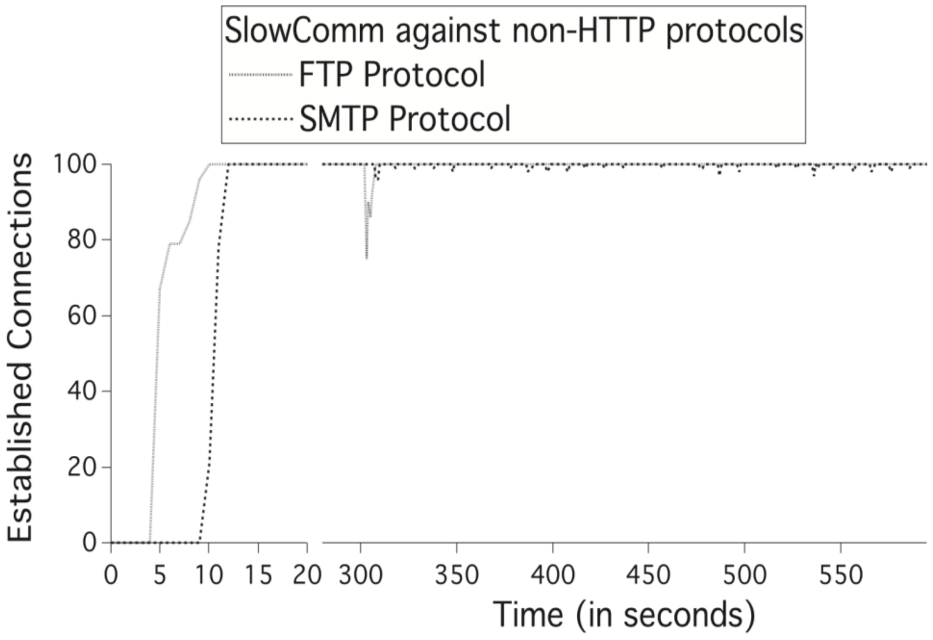

Figure 4 - Results of SlowComm against non-HTTP protocols

In this case, the attacks are both successful and the DoS state is reached after a few seconds from the beginning of the attack and it is maintained during the time, although a server-side timeout of `proftpd` automatically closes the connections after almost 300 seconds, returning a `421 Login Timeout` error.

Therefore, the proposed innovative threat represents an important attack in the cyber-security context, able to successfully perpetrate DoS attacks against different TCP based server systems.

5 Protecting from the Identified Innovative Threats

This section describes the defense methodologies we have identified concerning protection from the innovative attacks described in Section 4. For each attack, the protection methodology designed and implemented is described.

Regarding the IoT 0-day attack described in Section 4.1, in this section we report details on the protection system proposed and implemented, by describing how it works and by reporting preliminary results we have obtained during early tests, accomplished to validate the adopted approach. Concerning SDA attacks, instead, we report a description of the intrusion detection algorithm that we designed to identify the proposed attack. Finally, a framework able to identify cyber-security threats for IoT environment based on behavioural analysis model integrated into the ANASTACIA framework is described.

5.1 IoT 0-day attack protection

In general, since detection of a running threat may not be immediate, in order to protect a remote device from a Remote AT Command attack, it may be preferred to directly work on the (potentially vulnerable) nodes. In this context, three different approaches can be adopted, working at different levels:

1. **Firmware level**: creation of a modified version of the firmware, implementing Remote AT Commands filtering or allowing AT Commands elaboration at the application layer
2. **Device configuration level**: providing to the user the ability to configure a device to disable support to Remote AT Commands
3. **External level**: demanding protection capabilities to an external application program

Each approach provides an efficient solution to protect the device. Nevertheless, some approaches may not be adopted (e.g., device configuration, if not supported by the vendor). Suggested implementations provide a possible protection for this innovative threat.

The first proposed solution (firmware level protection) requires a device firmware upgrade to allow total AT Command packet management, such as the ability to process only packets received by the coordinator or secure devices. Such solution would provide the user with the possibility of configuring the device in order to avoid implicit Remote AT Commands interpretation. Nevertheless, since modifying a firmware may not be easy, and the source code may not be released as open-source software, it is suggested to have simpler but equally effective solutions.

The second solution (device configuration level protection) implements the ability to disable Remote AT Command support of the module, by implementing a specific setting able to disable automatic Remote AT Command interpretation (e.g., packets discard). In this way, the proposed threat would be ineffective. Nevertheless, to date, such solution is not supported by the provider, hence, implicit interpretation of remote AT commands is still accomplished.

Therefore, the last of the proposed solutions (external level protection) is the most interesting to us. Considering our scenario, the main purpose is to implement protection logics on the Arduino Uno device by implementing a function at application layer. The aim of the function is to verify if the XBee module may be communicating on the network. In this case, just before the sensor is ready to communicate on the network, an internal check is accomplished: if needed, an additional re-configuration is accomplished, in order to restore connectivity of the node.

By following this latest approach (external level protection), the protection system is directly employed on the nodes. In particular, the agents implemented on the IoT devices are responsible for monitoring the device status and verifying that all the parameters are correct. In case the device is affected by a remote AT reconfiguration command attack, such alert information is forwarded to the IoT coordinator, hence to the

ANASTACIA monitoring plane, and the device is designed to mitigate the attack (by autonomously reconfiguring itself, as previously described). Since not all the devices may embed a detection and mitigation system, the IoT coordinator is supposed to also monitor devices status periodically to identify disconnections, hence report them to the other ANASTACIA modules.

In order to validate the protection approach, a network of three components is implemented: a ZigBee coordinator and two nodes, vulnerable (hence, not protected from the proposed threat) and protected (hence, able to reconfigure itself autonomously in case a remote AT reconfiguration command is received) ones. In this case, while the protected node would keep communicating on the network, the vulnerable node would be disconnected from the network, and reconnection would not be possible (without physical operations on the node). Nevertheless, in both the cases, the attack attempts are identified and alert data is forwarded to the ANASTACIA Monitoring module, in order to analyze and manage the threat.

In order to validate the developed protection system, a series of tests are performed. In the considered scenario, a sensor has installed the protection system developed while another remains unprotected. Initially, we monitored the traffic for a certain time interval, to verify that both nodes communicate correctly. Subsequently, the attack was executed. As it is possible to observe in Figure 5, the protected node has an initial time in which it remains offline to reconfigure the parameters of the connection, then resumes to communicate correctly while the vulnerable sensor stops communicating with the main node.

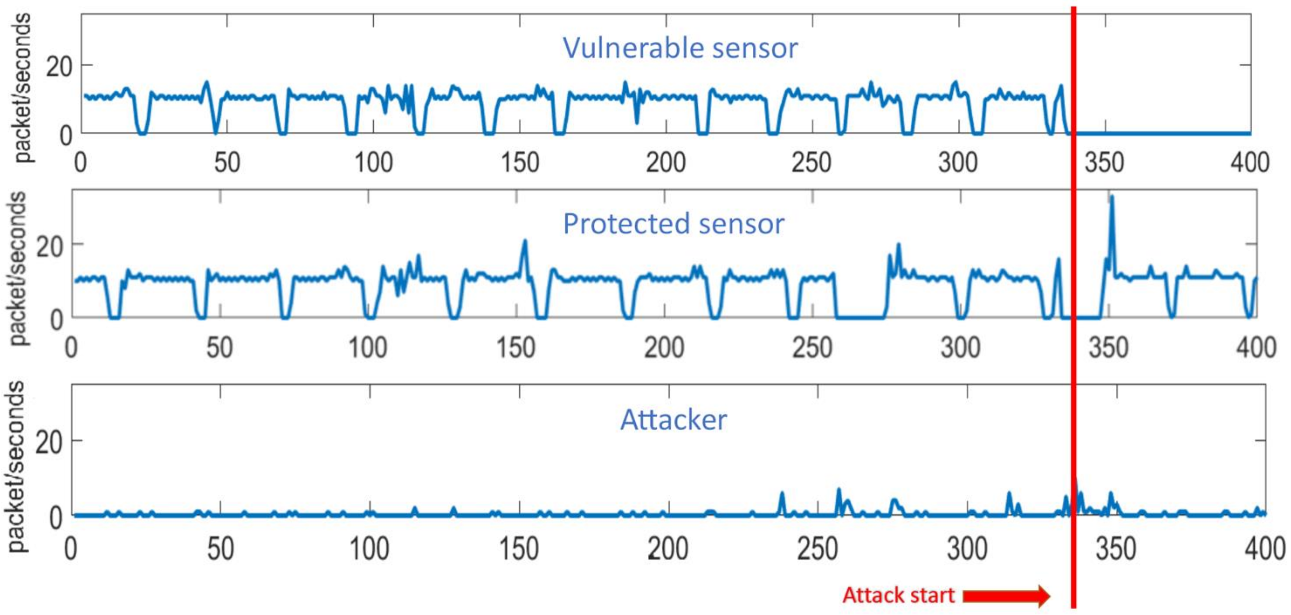

Figure 5 - Preliminary results of the protection system

It should be noted that the proposed attack should be considered a first step in the IoT security field. Indeed, although the integration and validation of the scenario on the ANASTACIA platform provides the ability to validate the attack and relative protection in a relevant environment, further work on the topic is needed in order to provide additional protection, from instance from flooding reconfiguration attacks, aimed to flood sensor nodes with high amounts of remote AT commands.

5.2 SLOW DOS ATTACKS PROTECTION

In order to protect from a Slow DoS Attack, it is important to consider the following fact: it is trivial to detect and mitigate a single attacking host, while it is extremely difficult to identify a distributed attack. This fact derives from the fact that IP address filtering may be applied to detect and mitigate a SlowComm attack (see, for instance, previous tests on mod-security), while in case of a distributed attack this concept may not be adopted with ease.

Moreover, from the stealth perspective, the proposed attack is particularly difficult to detect while it is active, since log files on the server are often updated only when a complete request is received or a connection is closed: being our requests typically endless, during the attack log files do not contain any trace of attack. Therefore, a log analysis can't produce an appropriate warning in reasonable times. Nevertheless, anomaly-based systems may reveal the malicious behavior of the threat, for instance through approaches based on statistic [Aiello, 2013], machine learning [Katkar, 2015; Duravkin, 2014; Singh, 2015], or spectral analysis [Brynielsson, 2015].

A possible protection approach, briefly described in the following, combines the algorithm proposed in [Aiello, 2013] and the methodology proposed in [Cambiaso, 2016] to detect running attacks. Early version of the algorithms has been tested in laboratory, while testing on relevant environments has not been accomplished to date. Concerning the ANASTACIA platform, a possible implementation of the described protection system would involve the Monitoring Module (of course, the other modules would be involved as a consequence of the detection).

The aim is to extract information from live network traffic, captured during a running attack against a specific ANASTACIA component. For our scenario, the target is represented by a web server (equipped for instance with the Apache2 daemon), while the attacker is represented by a single attacking node. As previously mentioned, the fact that a single node is targeting the server, instead of a set of nodes through a distributed attack, is not relevant in our scenario, since the detection algorithm does not distinguish on the attack source. In particular, after the identification of some protocol dependent parameters, their values are extracted and statistically analyzed, in order to compare and/or distinguish different network scenarios. According to [Cambiaso, 2016], the parameter related to an execution of the SlowComm attack is the Δ_{req} parameter, identifying the entire time of single requests. In our case, such time would assume values higher than expected, since SlowComm is designed to send endless requests to the server.

Briefly, once a specific parameter of the network connection has been defined (for example, the time related to the Δ_{req} parameter), a large data set of network connections is analyzed and all the values of the chosen parameter are measured for each connection. According to [Aiello, 2013], a histogram representing the occurrences of the parameter is then derived. This step is performed in order to execute a statistical analysis of events, eventually for forecasting purposes or for anomaly detection.

In order to retrieve such parameter from live network traffic, the following draft pseudo-code may be adopted.

```
var ht = {};
function process(pkt) {
    if(pkt.level7payload == null) return;
    var k = pkt.srcip+":"+pkt.srcport+"->"+pkt.dstip+":"+pkt.dsptport;
    if(ht[k] == null) ht[k] = { tstartrequest: pkt.time };
    ht[k].tendrequest = pkt.time;
}
```

In this way, connection data (each connection is univocally identified by `k`) are stored on a hash table `ht`. Once a packet arrives, if it does not contain application layer data, it is ignored. Otherwise, if not already present, it is inserted on a new hash table object. In any case, the `tendrequest` attribute of the corresponding object on the hash table is updated with the packet time. This means that for requests composed by a single packet, we have `tstartrequest = tendrequest`. This behavior does not limit the approach [Cambiaso, 2016]. At this point, the Δ_{req} value can be computed as the difference between `tendrequest` and `tstartrequest`. Such draft algorithm does not consider persistent connections, used to exploit already established connections to send additional requests to the server. Also, it does not require deep packet inspection that may be required for protocols allowing the sending of subsequent requests.

Let's consider two functions *f(x)* and *g(x)*, describing the probability distributions of the chosen parameter of two distinct network traffics. Specific characteristics of probability functions in general are that (i) the area under the distribution has value 1, after normalization is accomplished, and (ii) *f(x)*, *g(x) > 0*. Therefore, we initially build the sampling distribution of network traffic data set *f(x)* and *g(x)*, with the aim to compare them. Hence, our aim is to be able to determine if the events generating the above defined probability distributions are similar or, alternatively, have to be classified as different (i.e. in case of regular traffic or intrusion).

We therefore define and use the *L^2-norm* of the distribution difference:

$$N^2 = \int_0^\infty (f(x) - g(x))^2 \, dx$$

Intuitively, this norm represents an indicator (a pseudo-area) to highlight differences of the probability distributions describing network traffic; squaring is essential to emphasize the distinctions of *f(x)* and *g(x)*, regardless of the sign of the arithmetic difference.

Due to the above, one method to distinguish if *f(x)* and *g(x)* belong to different network scenarios is to evaluate how much the two functions differ: intuitively, if the underlying events are from the same network situation, *f(x)* and *g(x)* should be almost superimposed, and the area comprised between the two should tend to zero. Instead, if *f(x)* and *g(x)* describe different processes, the area comprised between the two functions is significantly different from zero.

From a mathematical point of view, N^2 equals zero if the two functions have the same distribution, while it may reach a maximum value when they are completely uncorrelated (not superimposed). The N^2 value corresponds to a numerical evaluation of the relationship between different network traffics. More the distributions are similar, more the N^2 value approaches zero.

In the context of Internet traffic analysis the process of characterizing traffic is executed on finite samples; therefore *f(x)* and *g(x)* belonging to the same network situation cannot be exactly identical but only similar, due to the randomness of network traffic.

This, it is crucial to establish a threshold N^{2*} beneath which *f(x)* and *g(x)* can be considered semantically equivalent. To this aim, under standard traffic conditions, several different measures of N^2 parameter have to be computed and its statistical distribution derived (in particular mean and standard deviation). Then a N^{2*} threshold is established according to a proper confidence level. Similarly, an average distribution, representative of legitimate scenarios, is computed.

Once the N^{2*} threshold is obtained, this algorithm allows to make a comparative analysis, using the equation reported above, between unknown traffic (in this case *g(x)*) and the mean legitimate traffic (in this case *f(x)*).

Then the so called H0 hypothesis is tested, and accepted if the unknown traffic belongs to the standard traffic distribution; conversely, if H0 is rejected, alarms are raised since f and g are semantically different (i.e. $N^2 > N^{2*}$).

The H0 hypothesis test can be conducted either on a single event (*g(x)*), or measuring the average of N^2 obtained analyzing different events. In the second case, the test takes more time to get the result (more PCAP packet capture files to be captured), although more accurate since:

$$\sigma_{mean} = \frac{\sigma}{\sqrt{n}}$$

with n the number of samples on which the average is executed.

Moreover, since N^2 compares two samples, two-sample test have to be executed, leading to the:

$$\sigma_{mean} = \sqrt{\frac{\sigma_1^2}{n_1} + \frac{\sigma_2^2}{n_2}}$$

where σ_1 is the standard deviation of the first distribution and n_1 is the number of samples used during the average calculation (similarly for σ_2 and n_2 values, related to the second distribution).

After computing the comparison between legitimate traffics, in order to sample them, it is possible to retrieve $\mu_{legitimate}$ and $\sigma_{legitimate}$ values. Hence, by adopting the 3-sigma rule [Cambiaso, 2013], it is possible to define a threshold $N^{2*} = \mu_{legitimate} + 3 \cdot \sigma_{legitimate}$, providing a confidence level equal to 99.7%. In particular, by comparing the potentially anomalous traffic distribution with the distribution representative of legitimate scenarios, it is possible to find N^2. By comparing such value to N^{2*} it is possible to identify if the traffic is anomalous ($N^2 > N^{2*}$) or not.

Through the proposed algorithm, early validated in laboratory to detect threats similar to SlowComm, it is theoretically possible to identify SlowComm attacks on the network, hence to provide an additional detection algorithm able to reveal attack attempts against network components. By potentially implementing such algorithm on ANASTACIA component, it may be possible to provide additional capabilities to the entire framework, in the context of cyber-security applied to counter last generation threats.

5.3 BEHAVIOURAL ANALYSIS

The framework identifies cyber-security attacks automatically in a given IoT environment. The framework uses system design and operational data to discover dependencies between cyber systems and operations in a cyber-physical domain. We predict potential security consequences of interacting operations among subsystems and generate threat alarms. Specifically, we are empowering ANASTACIA framework with new capabilities in form of behavioural analysis model to detect evasions or advanced attacks. Newly developed concept depicted on Figure 6 can help detect wide range of attacks including:

- Known attacks such as DDoS and MITM attacks (see ANASTACIA D2.2)[2],
- New zero-days attacks and slow DoS attacks that might pass undetected by normal IDS/IPS,
- Other attacks that affect monitored system characteristics.

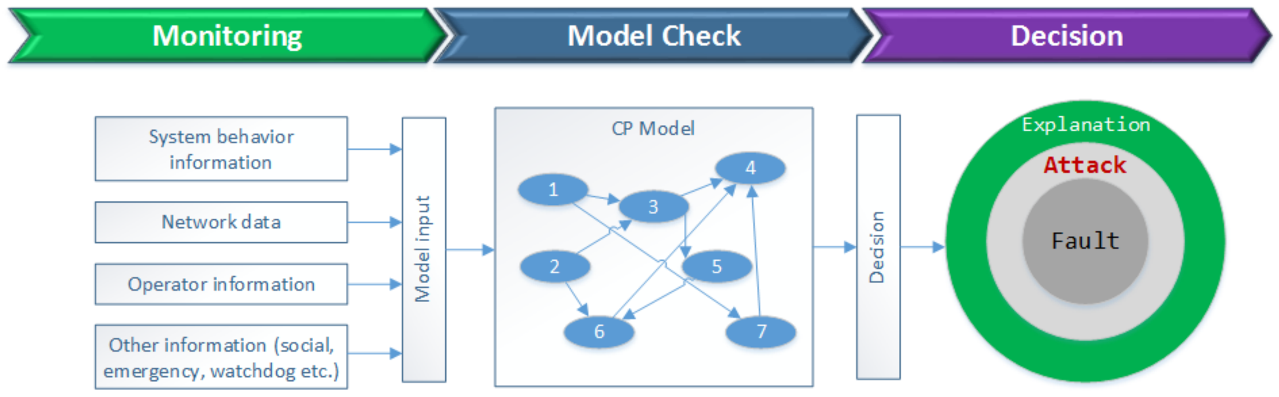

Figure 6 – Behavioural analysis concept.

Monitoring Framework developed for ANASTACIA includes component which is composed of messaging wrappers, constraint programming (CP) models and buffered sensor data from IoT networks. Mainly, CP model is core component of behavioural analysis engine. CP a subfield of Artificial Intelligence that solves combinatorial and optimization problems. It has been used widely in various application fields such as scheduling, automatic verification, timetabling, and planning problems. In contrast to this, we aim to use CP model to detect outliers in CPS/IoT data in real-time by using constraints.
First the information is gathered and analyzed for learning a CP model and then it is deployed in order to identify any intrusion. Moreover, CP model built on continuous stream of data (i.e. time-series) where the time interval between successive updates could vary from milliseconds to minutes. CP model consists of network of relations between building sensor data. Using this CP model, we aggregate the different types of

[2] More information can be found at the following address: https://www.bbc.com/news/technology-37738823

sensor data to truly model the normal behaviour of the system that is being supervised. This model is built for monitoring at system level, but it does not prevent from including in the model information about network performance if that is exposed to it. For an example, CPU consumption of a device can be included along its actual sensor data. The variety of data that we can aggregate allows the model to be as generic or as specific as the end-user required it to be. Since the model is built on relations, we can leverage from the fact that what data effects what other data type (features).

We developed an approach to learn a CP-based decision model consisting of a set of relations to detect misbehaviour of the system. More specifically, the idea is to learn a set of relations which together when satisfied defines the normal behaviour of the system. After learning important relations, the approach discards un-important relations, and consequently creates a model with best possible relations and features of sensor nodes. In each iteration, the relation between the sensor features and all other network features further is verified. Also, we identify the sensors which are involved in breaking the relation and what are the set of relations which are broken. Following this fashion, the model is further tuned. The workflow for learning the model is explained in ANASTACIA D4.1. The developed 'Monitoring' component enables continuous and integrated monitoring of multivariate signals, event logs, heartbeat signals, status reports, operational information, etc., emanating from various devices in multitude of building operational subsystems. This monitoring component also evaluates the security situation against known policies, models, threat signatures to detect abnormalities and outliers, e.g. high data download, external database or port accesses during an emergency. Such situations will be analyzed by the 'Reaction' component which will evaluate the severity of the situation. Isolation and predictive mechanisms are activated to ensure that the rest of the building operations system continues as normal. Policies and rules are activated, updated and enforced by the 'Security Enforcement' component, e.g. building emergency will lock-down the non-essential database accesses, and escalation of the emergency to the city fire brigade should be performed by any of the authorized personnel.

5.3.1 Behavioural detection model

CP is a declarative programming framework that consists of Constraint Satisfaction Problem (CSP). Generally CSP is composed of variables with set of domains which depicts the state of the system. Constraints are imposed on the domain of variables, called relations which define the problem. Our approach depends on depicting the desired CPS patterns or signatures through the constraints and then identifying a set of features that match the target CPS context in a given time window. Mainly, the model building consists of two phases. In first phase, we create a constraint state model with monitored features that consists of all features and observed relations among the features. The initial model serves as a base for tuning and learning a model that satisfies an acceptance criteria with respect to ground truth. In order to build the model, we frequently use the state and assignment cost of features.

Developed approach (Figure 7) to learn a CP-based decision model consisting of a set of relations to detect misbehaviour of the system. More specifically, the idea is to learn a set of relations which together when satisfied defines the normal behaviour of the system. After learning important relations, the model discards un-satisfied relations, and consequently update the model with best possible relations and features of IoT sensor nodes. In each iteration, *the relation between the sensor features and all other network features are further verified.* Also, *we identify the sensors which are involved in breaking the relation and what are the set of relations which are broken Following this fashion, the model is further tuned.* The workflow for learning the model is explained in ANASTACIA D4.1.

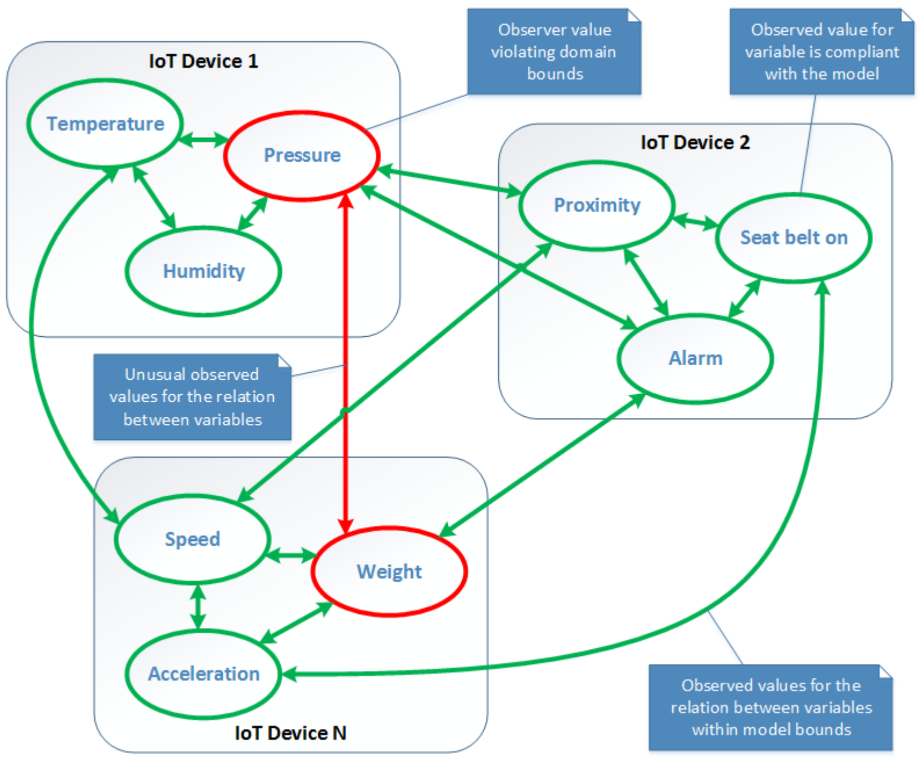
Observer value violating domain bounds
Observed value for variable is compliant with the model
IoT Device 1
Temperature
Pressure
Humidity
IoT Device 2
Proximity
Seat belt on
Alarm
Unusual observed values for the relation between variables
Speed
Weight
Acceleration
IoT Device N
Observed values for the relation between variables within model bounds

Figure 7 – CP Model with relations with highlighted sensor parameter.

Mainly detection in real deployment scenario will depend on how much data for training and validation is available and how many model constraints are predefined to enable high level model adherence to reality. The observed data in monitored agent is compared to anomaly detection model (CP model) and a decision is returned whether the model considered this behaviour as normal or abnormal. The decision (verdict) is sent to “UTRCVerdict” topic on Kafka broker for further analysis. In online usage in every predefined amount of minutes per time window a verdict will be returned whether the model has seen an anomaly or not. However, for simulation purposes we aggregate all verdicts and calculate the models accuracy in form of a confusion matrix. We consider attack windows and non-attacked points, where an attack window is detected if at any time point the model raised an alarm. Once a simulation is performed and we obtain the sequence of verdicts from the model we compare this with the expected verdict and calculate the following confusion matrices for man-in-middle attack at specified use cases.

5.3.2 Monitoring architecture

The developed monitoring component (Figure 8) performs system level monitoring by aggregating information from security enforcement plane (SEP) using Kafka broker. Details of data consumption scenario were described in detail in ANASTACIA D4.1. Behavioural engine is divided into two parts:

- Monitoring – received messages are processed, filtered and cleaned to enable data recording for future model training and attack verdict generation that will be sent to reaction components.
- Detection – system level analysis of current security state of SEP based on trained model and current information stored in monitoring buffer.

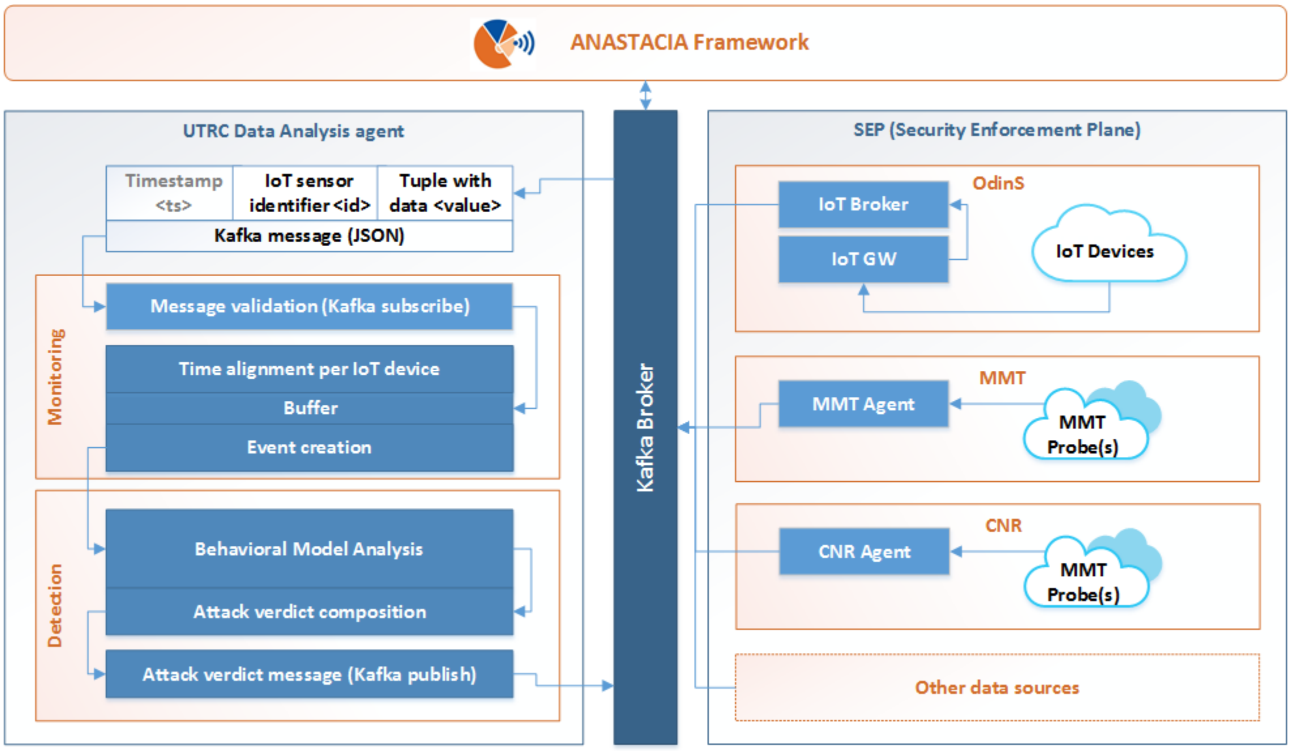

Figure 8 – Behavioural analysis engine's internal monitoring architecture.

It depicts two main parts of agent (monitoring and detection). Both sides are preconfigured to adjust to various system monitoring capabilities for anomaly detection (temperature, pressure, IR presence, etc.). The common interface between monitoring and detection part enables agent to consume and process various data sources that are available on SEP. The only condition is that message should contain IoT device identifier and value that will contain device current monitored state (both are marked blacked inside JSON message). Optionally if the security perimeter is not co-located with ANASTACIA framework, timestamp can be added for higher detection accuracy at detection level.

This mechanism enables other data sources to deliver more information that can be used in more accurate anomaly detection and IT system security evaluation. Once system state with historical data is created, it is passed to the model for detection evaluation and later the agent will generate appropriate attack verdict that will be sent to Verdicts and Decision Support System (VDSS) component via Kafka broker. All events are stored in time ordered fashion so when model requests them they are fetched without need to sort them out. The idea behind the scenes is to abstract model processing from various data sources and uniform system view (data cleaning) before sending information to detection process.

From data buffer configuration perspective, the buffer is initialized based on model configuration. This way when agent starts the buffer automatically adopts to model parameters (features) configuration to deliver required framesets for detection analysis.

6 Mitigation Plans and Contingency Actions

In this section of the document we report mitigation plans and contingency actions for the proposed attacks, applied in the ANASTACIA context.

6.1 Mitigation Plans

The selection of the most convenient mitigation is paramount to guarantee minimum impact on the infrastructure affected while maximizing the effectiveness of the remediation used to mitigate an ongoing incident. The VDSS is the component in charge of analyzing the incidents detected and deciding on the most convenient mitigation according to several criteria. Figure 9 represents a high-level overview of the inputs and outputs of the VDSS. The VDSS uses the information included in the alerts triggered by the incident detector, based on the information monitoring from the IoT infrastructure. Information about the assets affected by the incident is also required, including aspects such as the criticality of the assets in different aspects. Finally, it is also required information about the mitigations that are capable to be enforced by the infrastructure. The list of available mitigations is defined in the security policy although additional information is used by the VDSS in the analysis, such as cost of mitigating of the impact of such mitigation.

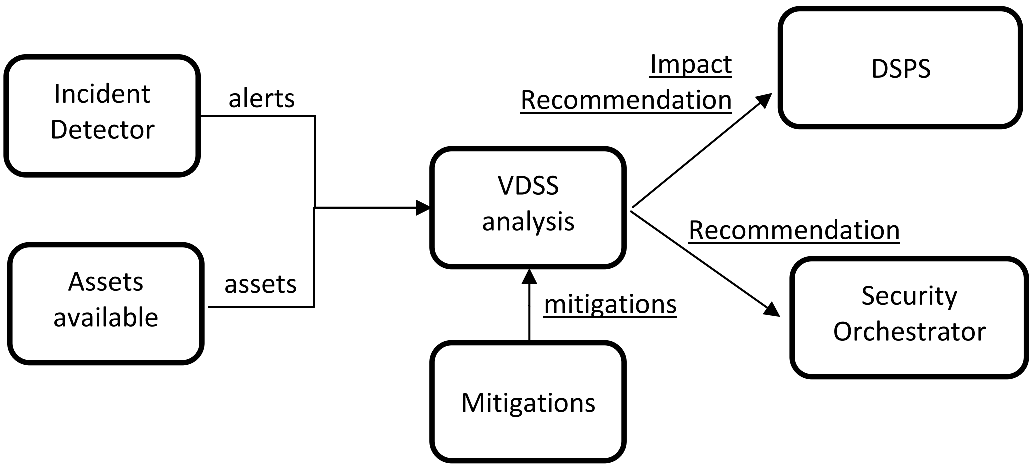

Figure 9 - Inputs and outputs of the VDSS

The VDSS combines information extracted from alerts, assets and mitigations to provide with quantitative scores representing the suitability of the mitigation for a given incident affecting a set of assets.

Alerts are generated by the incident detector by correlating events received from monitoring agents. Let N be the number of possible alerts, an alert a_i is characterized by three parameters, defined as:

$$a_i = a(\pi, \sigma, \rho)\,, \qquad i = 1 \ldots N$$

Being:

π the priority of the alert, representing the importance of the alert according to the severity of the incident detected by the Incident Detector. For example, in general terms, it is expected a higher priority for DoS alerts rather than port scanning ones.

σ the reliability of the alert, which indicates how sure the incident detector is about the alert triggered. Given the fact that the incident detector is capable of correlating events from different sources, this parameter takes into account the trustworthiness of the source of the event, considering the possibility of some uncertainty of the event sample collected from the infrastructure.

ρ the risk associated to the alert, computed as the impact of the incident over an infrastructure and associated to the device directly affected by the incident detected.

The information about the assets can be received either directly from the infrastructure (using some automatic mechanism for the creation of assets inventory such as OCS[3], or directly by some system administrator. Let M be the number of possible assets, an asset s_j is characterized by three parameters, defined as

$$s_j = s(\mu, \alpha, \gamma), \qquad j = 1 \dots M$$

Being:

μ the economic importance of the asset, including aspects such as the monetary cost of the device, the cost of repairing or maintenance or the economic lost in case of unavailability or malfunction of the device.

α the importance of the asset in terms of availability, this is how important is that the device is up and running for the service it is providing within an infrastructure. For instance, it is expected an α higher for a finger print access device rather than for an environmental sensor that measures pollution levels.

γ the societal criticality of the asset, this is how critical for human security and safety the device is. For example, it is expected a higher γ for devices managing traffic lights rather than for the environmental sensors mentioned above.

The information about the possible mitigations depends on the security policy, which determines the list of available security capabilities to be used by the security orchestrator to enforce the mitigations. Not all mitigations are the same, some of them are easy to deploy not using almost any resource (e.g., adding a new firewall rule) while others require more time and resources to be deployed (e.g., deploying a virtual honeynet). Therefore, every mitigation can be characterized by several parameters, which can be retrieved either automatically from the infrastructure or manually from system admins. Let K be the number of possible mitigations, a mitigation m_k can be described by:

$$m_k = m(\emptyset, \omega), \qquad k = 1 \dots K$$

Being:

$\emptyset$ the cost of the mitigation, understanding the cost not just in monetary terms but also in terms of resources required to mitigate.

ω the impact of the mitigation within the infrastructure, in terms of aspects such as the time that the infrastructure needs to be down when mitigating.

Having defined the inputs to be used by the VDSS, and having $M_K = (m_1, \dots, m_K)$ as the list of available mitigations, we can define V_{a_i} as a subset of mitigations capable of mitigating an incident represented by an alert a_i, being $V_{a_i} \subseteq M_K$.

The VDSS calculates two parameters to take the appropriate decision when choosing a mitigation:

- $\bar{I}$ is the impact of the incident over the infrastructure, defined as the average of the individual impacts of the incident over the assets of the infrastructure:

$$\bar{I} = \frac{\sum_1^j I(a_i, s_j)}{j}$$

- S is the suitability of a reaction for a given incident a_i. To calculate it we need to compute both the impact of the incident, given by $\bar{I}$ against every suitable mitigation m_k in V_{a_i}.

$$S_k = f(\bar{I}, m_k), \qquad m_k \epsilon\, V_{a_i}$$

[3] More information can be found at the following address: https://www.ocsinventory-ng.org/en/

Details of the calculation of the impact and suitability, and how they are used to decide on the most convenient mitigation will be given in WP4.

The final result provided by the VDSS is a list of mitigations with the corresponding suitability score, representing the suitability of the mitigation for the detected incident. It is decision of the orchestrator (or system admin) to automatically enforce the recommendation with the higher suitability score or decide on any other depending on other parameters obtained in the analysis (such as the global impact $\bar{I}$ of the incident).

6.2 Contingency Actions

Whenever the attack is successfully detected, contingency actions should be applied for preventing the attacks without affecting the service functionality. The security orchestrator has been designed in order to deploy and manage different security enablers according to the defined security policies. The later would be generated either by the end-user or received from the monitoring and reaction plane when some misbehavior in the network is detected. The security orchestration plane takes into account the policies requirements and the available resources in the underlying infrastructure in order to mitigate the different attacks while reducing the expected mitigation cost and without affecting the QoS requirements of different verticals. The resources in the underlying infrastructure refer to the available amount of resources in terms of CPU, memory, and storage in different cloud providers, as well as the bandwidth communication between these network clouds.

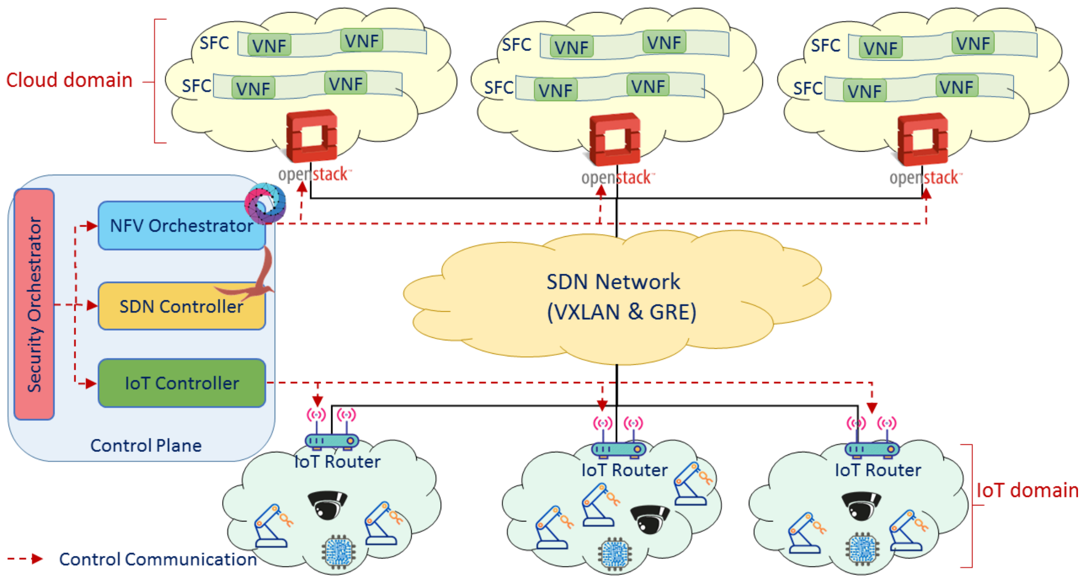

Figure 10 - Security orchestration plane

Figure 10 shows the main architecture of the security orchestration and enforcement plane suggested that used to apply the adequate contingency actions for preventing the attacks in an efficient way. Using SDN network, the IoT domain is connected to the cloud domain, whereby different IoT services are running.

Figure 11 depicts the different steps for applying contingency actions whenever misbehavior in the network is detected. After detecting the attack by the Mitigation Action Service (MAS) component. The later sends a mitigation request (MSPL file) to the security orchestrator (Fig. 10: 3). To mitigate the attacks, the security orchestrator interacts with three main actors, which are (Fig. 10: 4):

- **IoT controller**: In order to mitigate different attacks, the security orchestrator interacts with the IoT controller in order to mitigate the attacks at the level of the IoT domain and prevent the propagation of the attack to other networks (Fig. 10: 4). The IoT controller enforces different security rules at the IoT router (data plane) to mitigate the attack (Fig. 10: 5).

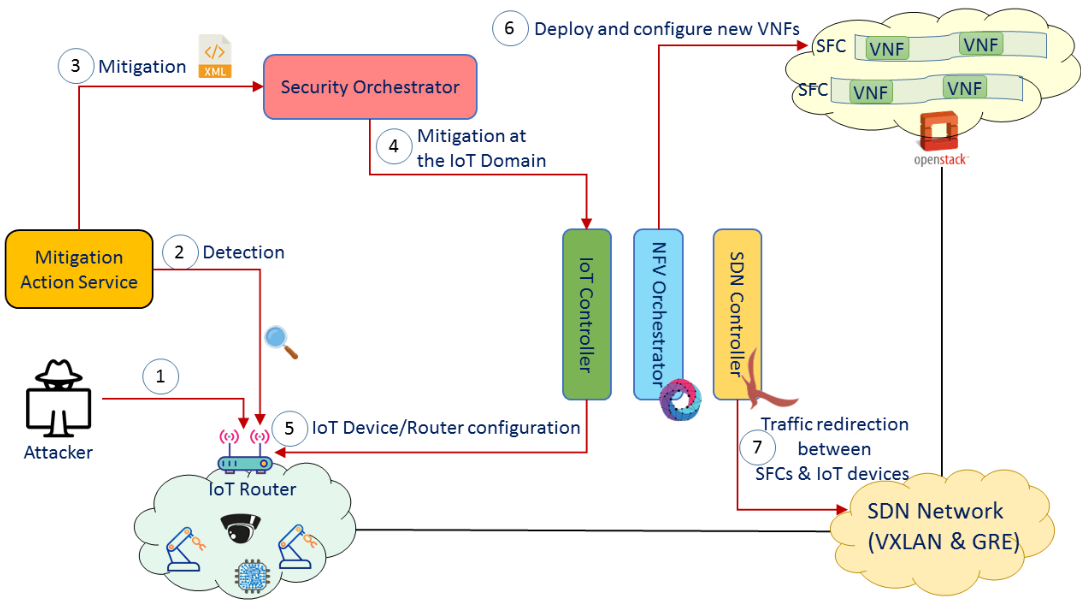
6 Deploy and configure new VNFs
3 Mitigation
XML
Security Orchestrator
SFC
VNF
VNF
SFC
VNF
VNF
openstack
4 Mitigation at the IoT Domain
Mitigation Action Service
2 Detection
IoT Controller
NFV Orchestrator
SDN Controller
1
Attacker
5 IoT Device/Router configuration
Traffic redirection between SFCs & IoT devices
7
IoT Router
SDN Network (VXLAN & GRE)

Figure 11 - Security orchestration and enforcement in case of a reactive scenario

- **NFV orchestrator:** After receiving the MSPL message from the MAS, the security orchestrator identifies the right contingency actions should be applied to prevent the attacks. If the mitigation plan requires the instantiation of new VNFs, the security orchestrator instructs the NFV orchestrator to instantiate and configure the required VNFs. In order to instantiate the required VNFs, the NFV orchestrator interacts with the VIM (Fig. 11: 6). Also, the security orchestrator interacts with the policy interpreter to translate the received MSPL to the low configuration (LSPL) needed for different VNFs. After the successful instantiation of a security VNF, the security orchestrator configures that VNF with the received LSPL (Fig. 11: 6).
- **SDN controller:** As depicted in Figure 11, when the mitigation action service notifies the orchestrator about an attack, the SFC would be updated by adding/inserting new security VNFs in the SFCs. The security orchestrator should push the adequate SDN rules to reroute the traffic between different VNFs in the SFC and the IoT domain (Fig. 11: 7). Also, according to the different situations, the security orchestrator can choose the SDN as security enabler. In this case, the contingency actions can be applied by leveraging SDN technology. If so, the security orchestrator can instruct the SDN controller to push some SDN rules to prevent, allow or limit the communication on specified protocols and ports between different communication peers (Fig. 11: 7).

7 Planned Activities

In this section of the document, we describe how the proposed innovative attacks and related protection systems are integrated inside of the ANASTACIA platform.

7.1 IoT-0 day planned activities

The network adopted to perform the tests on the innovative attack consists of three IoT nodes: a coordinator node that manages the ZigBee network and also is connected to an internet network, two ZigBee sensors adopted to implement and validate the innovative attack and the related protection system.

In order to integrate this scenario with the ANASTACIA framework, an interface between the Coordinator node and the monitoring module developed in ANASTACIA is implemented, as shown in the Figure 12. The interface is based on Kafka broker (part of the ANASTACIA platform), a software mainly used to exchange messages between different components of a system.

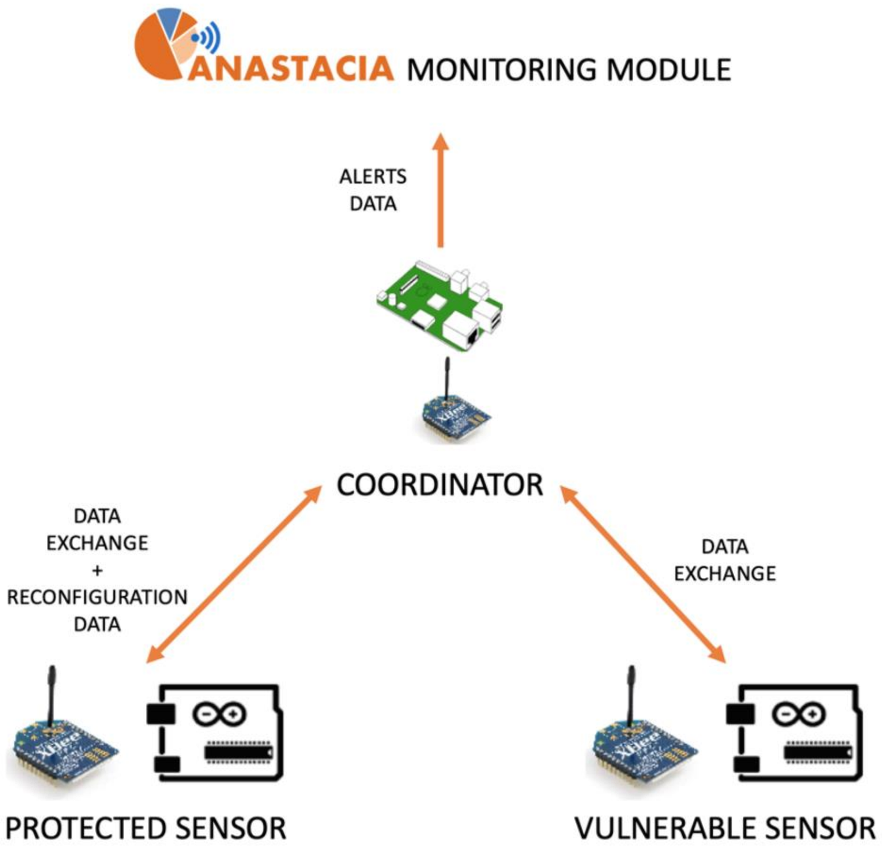

Figure 12 - IoT network

The integration system works as follows: when the protected sensor activates the implemented defense system, a particular packet is sent to the Coordinator node which will send an alert to the monitoring module through the Kafka Broker. In addition, within the Coordinator sensor, an algorithm was developed that analyzes the messages sent by the network sensors and in case a sensor does not send a message after a certain time interval, it sends an alert to the Monitoring module to alert the network to a possible attack on the sensor.

In conclusion, the innovative scenario communicates with the ANASTACIA framework Monitoring module through a channel hosted on Kafka Broker where the messages exchanged are alerts that allow identifying possible attacks on the IoT network.

7.2 Slow DoS Attack planned activities

Concerning protection from Slow DoS Attacks, the adopted methodology is based on the implementation of an intrusion detection system aimed to identify anomalous situation. In particular, for this aim, two different approaches will be followed:

1. A first approach is focused on the identification of a running attacks, by using an approach in some ways similar to the `reqtimeout` one [Cambiaso, 2017], that triggers a server-side connection closure if no enough request data is received within a specific time period. In particular, for "enough request data", we mean a certain amount of data championed after analysis of legitimate connections (through the same approach reported in Section 5.2). For instance, if, like in case of a running SlowComm attack, considering a single connection, the attacker sends a first portion of data to the server, and the second part of the request is sent after a specific time, once such second part is received (or even earlier), the server may compute the reception transfer speed and close the connection. As mentioned, this approach is similar to the `reqtimeout` one, although in this case the application is protocol independent (while `reqtimeout` is a software module specifically designed for the Apache2 web server).
2. A second approach can be combined with the previous one, in case the first approach is not enough satisfying after implementation. For instance, this may occur for specific protocols like SSH, characterized by extremely low traffic volumes, particularly similar to a running SDA. In this case, by adopting the intrusion detection algorithm reported in Section 5.2, we are able to identify anomalies on the network, related to the execution of a SlowComm attack. In this case, since off-line analysis is accomplished, real-time detection requirements may not be satisfied.

By combining the two approaches, it is possible to identify Slow DoS Attacks in an efficient way. Further work will be focused on the implementation of such approaches inside of ANASTACIA components. In particular, both the approaches require a network tap that, in the first case, has to be active, in order to intercept and actively respond to captured traffic. Differently, in the second case, the network tap could also passively capture network traffic to analyze it.

7.2.1 Slow DoS in the ANASTACIA MMT Tool

The Montimage Monitoring Tool is a security solution that directly analyses the network traffic and tests security properties using the information extracted with its Deep Packet Inspection (DPI) engine. Having this in mind, the MMT software can be effectively used to detect the aforementioned Slow DoS.

As mentioned in the previous section, the Slow DoS Attack can be detected using two strategies that rely on network-available information:

- Detection of low flow throughput. By using DPI engine, the MMT software is capable of detecting the exact size of the messages transmitted from client to server in each flow. Using this information, MMT is capable of determining the effective throughput of the flow at the application layer, which is crucial to detect the Slow DoS attack.
- Detection of long request transfer time. When access to the payload is not possible (for example, some scenarios where the payload of the packet is encrypted), the detection of the attack is still possible. The MMT software can easily be extended to embed the detection algorithm proposed in Section 4.2.

Considering these observations, Montimage plans to extend its tool in order to include both approaches to detect the Slow DoS attack. In particular, the related activities will include:

- Implementation of new protocol plugins to support the analysis of the IoT-specific protocols (ZigBee);
- Development of new security rules to be enforced that aim to detect the slow throughput of a flow;

- Development of a complex security rule capable of maintaining a live hash table with the last active connections to the server. This rule aims to keep track of the packets belonging to a single flow, and for how long the flow has been open, triggering an alert once a timeout has been reached.
- Performance and effectiveness of the detection algorithm. To complete the integration of the detection mechanism, it is planned to perform a set of performance and effectiveness tests to evaluate the approaches. In this way, we will determine the effectiveness (false positives and false negatives rates) and the applicability of the algorithms in real life environments. We aim to use the results of this evaluation to make eventual adjustments of the algorithm in order to have a working proof-of-concept validated in the ANSTACIA use cases.

The security rules mentioned above will enrich the set of available rules, and they will be deployed in the MMT-Probe instance integrated ANATSACIA platform to enrich the security analysis. Once this integration is completed, the MMT-Probe will be able to generate alerts about Slow DoS attack when one (or both) of the aforementioned properties are triggered, alerting the rest of the ANASTACIA processing chain and triggering the necessary reactions.

7.2.2 Slow DoS in the ANASTACIA Incident Detector

As any type of incident, the integration with the ANASTACIA Incident Detector very deeply relies on the security probes that are monitoring the infrastructure. In the case of Slow DoS Attacks, the detection can, in principle, be done either by having agents installed in the devices or by analyzing the traffic towards a targeted device. Agent based monitoring provides with information about the situation of the device targeted. This includes logs produced by the attacked service or installation of HIDS tools. In any case, the applicability of this approach depends on the capabilities of the devices targeted, which, in case of IoT devices, are scarce. However, a few solutions are already appearing[4], which would be able to report about opened connections, closed connections, time that a connection is open, or traffic exchanged by established connections. Additionally, network sniffers (such as conventional NIDS – MMT-Probe) can detect connection establishment patterns against certain devices. In this case, the ANASTACIA Incident Detector can combine such information to detect patterns that denote a potential Slow DoS incident. For instance, the combination of events that detect a high rate of open connections in a device (received through an HIDS) with a set of anomalous traffic targeting the device might represent a Slow DoS attack against it. In any case, the ANASTACIA Incident Detector would act as a correlator of events, applying the rules that identify the patterns of a Slow DoS. The higher the number of potential evidences (for example, the number of security probes capable of providing with events about Slow DoS information), the higher the reliability of the detection is.

Same as for the rest of the events received by the ANASTACIA Incident Detector, new plugins would be required to be created in order to properly interpret the events received by detectors that can provide with evidences about Slow DoS incidents (be them HIDS on device, external network sniffers or any other potential probe capable of providing such information). The complexity of the detection would be on the creation of appropriate rules, which, at the same time, depends on the identification of patterns, this is, similar events happening when a Slow DoS attack is triggered.

Figure 13 summarizes three possible sources of information to retrieve information relevant for detecting Slow DoS attacks:

- HIDS agents running within the IoT device, reporting information about number of stablished connections, bandwidth usage, amount of time connections are alive, etc. This option is only possible when the IoT device has enough resources both to run the HIDS agent and to report the events to the Incident Detector Agent, which, once normalized, are sent to the Incident Detector correlator. The mechanism used to report the events also depends on the connectivity capabilities of the IoT device (for example, whether it is able to report events through HTTP, TCP or just UDP).

[4] IDS designed for IoT systems. https://journalofcloudcomputing.springeropen.com/articles/10.1186/s13677-018-0123-6#Sec16

- HIDS agents reporting events as a proxy. This alternative is used for IoT devices with limited resources or with not sophisticated communication capabilities. In this case, a dedicated HIDS, acting as a proxy, would convert the information reported by the IoT device. IoT controllers could assume the role of HIDS proxy by providing to the Incident Detector Agent with relevant information about for Slow DoS incidents.
- Network monitoring, in this case, well known NIDS tools (e.g., Snort or Suricata) can be used to sniff and analyze traffic towards the targeted assets. In this case, the mentioned NIDS tools typically can't work at the physical layer. Unfortunately, many IoT devices, due to limited resources, are not able to work with full IP at the network layer (typically they are limited to 6LoWPAN), using just UDP at the transport layer and limited to CoAP at the application layer. However, the research community remains very active in what regards to intrusion detection in the IoT domain [Zarpelao17; Elrawy18].

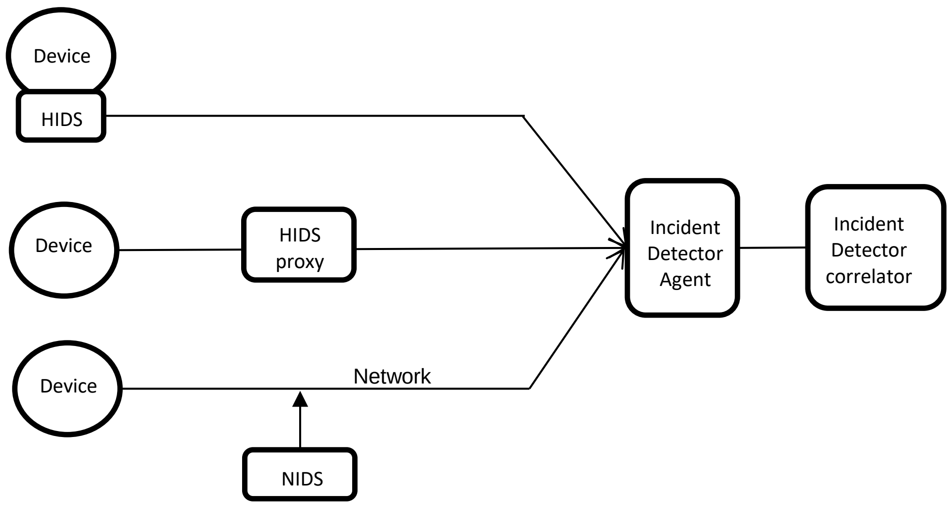

Figure 13 - Combination of events from different possible sources to detect Slow DoS attacks

8 Conclusions

This document describes information on how to detect and mitigate attacks previously introduced in D1.2 document and not considered in the scope of D2.2. Also, the document reports details on the innovative attacks integrated into the ANASTACIA framework and the actual state of the available mitigation and contingency methodologies. The attacks were initially described by exposing their aims and how they can be integrated into the project context. Subsequently, for each of the considered novel cyber-threat, the functioning is described in detail, hence reporting details on how to protect from the identified attacks. Regarding the IoT 0-day scenario, the proposed protection methodology is exhaustively described. Concerning the slow DoS scenario in particular, different approaches and an intrusion protection algorithm have been proposed, in order to detect the attack on the network. In addition, a possible solution to detect and mitigate the threat through the Incident Detector and the Montimage Monitoring Tool is described.

In addition, for each innovative attack, it has been described how it is involved in the ANASTACIA framework, and how the protection components are involved, in particular, the possible communication between the targeted components and the Monitoring and Reaction modules. The document also describes an innovative system that potentially provides ANASTACIA the ability to automatically identify attacks against IoT systems based, through behavioral analysis. It consists of a first component adopted to monitor network devices, plus a second component used to identify attacks.

Future work will be directed on the integration of the attacks on the ANASTACIA platform. Concerning the IoT 0-day attack, being the scenario tested only on isolated laboratory, integration with the ANASTACIA platform and related components will be accomplished. Regarding the SDA context, further work will focus on the implementation of a protection system, mainly on the Monitoring and Reaction module. In addition, SDA tests on the CoAP protocol will be accomplished, in order to evaluate the success of such threat when targeting a network protocol adopted on IoT environments.

9 REFERENCES

[Aiello, 2013] Aiello, M., Cambiaso, E., Scaglione, S., & Papaleo, G. (2013, July). A similarity based approach for application DoS attacks detection. In Computers and Communications (ISCC), 2013 IEEE Symposium on (pp. 000430-000435). IEEE.

[Beitollahi, 2011] Beitollahi, H., & Deconinck, G. (2011). A dependable architecture to mitigate distributed denial of service attacks on network-based control systems. International Journal of Critical Infrastructure Protection, 4(3-4), 107-123.

[Brynielsson, 2015] Brynielsson, J., & Sharma, R. (2015, August). Detectability of low-rate HTTP server DoS attacks using spectral analysis. In Advances in Social Networks Analysis and Mining (ASONAM), 2015 IEEE/ACM International Conference on (pp. 954-961). IEEE.

[Cambiaso, 2012] Cambiaso, E., Papaleo, G., & Aiello, M. (2012, October). Taxonomy of slow DoS attacks to web applications. In International Conference on Security in Computer Networks and Distributed Systems (pp. 195-204). Springer, Berlin, Heidelberg.

[Cambiaso, 2013] Cambiaso, E., Papaleo, G., Chiola, G., & Aiello, M. (2013). Slow DoS attacks: definition and categorisation. International Journal of Trust Management in Computing and Communications, 1(3-4), 300-319.

[Cambiaso, 2016] Cambiaso, E., Papaleo, G., Chiola, G., & Aiello, M. (2016). A Network Traffic Representation Model for Detecting Application Layer Attacks. International Journal of Computing and Digital Systems, 5(01).

[Cambiaso, 2017] Cambiaso, E., Papaleo, G., & Aiello, M. (2017). Slowcomm: Design, development and performance evaluation of a new slow DoS attack. Journal of Information Security and Applications, 35, 23-31.

[Duravkin, 2014] Duravkin, I. V., Carlsson, A., & Loktionova, A. S. (2014). Method of slow-attack detection. Системи обробки інформації, (8), 102-106.

[Elrawy18] Elrawy, M. F., Awad, A. I., & Hamed, H. F. (2018). Intrusion detection systems for IoT-based smart environments: a survey. Journal of Cloud Computing, 7(1), 21.

[Gubbi, 2013] J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, "Internet of Things (IoT): A vision, architectural elements, and future directions," Futur. Gener. Comput. Syst., vol. 29, no. 7, pp. 1645–1660, 2013.

[Katkar, 2015] Katkar, V., Zinjade, A., Dalvi, S., Bafna, T., & Mahajan, R. (2015, February). Detection of DoS/DDoS Attack against HTTP Servers Using Naive Bayesian. In Computing Communication Control and Automation (ICCUBEA), 2015 International Conference on (pp. 280-285). IEEE.

[Kirichek, 2015] Kirichek, R., Makolkina, M., Sene, J., & Takhtuev, V. (2015, October). Estimation quality parameters of transferring image and voice data over ZigBee in transparent mode. In International Conference on Distributed Computer and Communication Networks (pp. 260-267). Springer, Cham.

[Piyare, 2013] R. Piyare and S. r. Lee, "Performance analysis of xbee zb module based wireless sensor networks," International Journal of Scientific & Engineering Research, vol. 4, pp. 1615–1621, 2013.

[Ramya, 2011] C. M. Ramya, M. Shanmugaraj, and R. Prabakaran, "Study on ZigBee technology," in Proceedings of the 3rd International Conference on Electronics Computer Technology (ICECT '11), pp. 297–301, IEEE, Kanyakumari, India, April 2011.

[Singh, 2015] Singh, K. J., & De, T. (2015). An approach of DDOS attack detection using classifiers. In Emerging Research in Computing, Information, Communication and Applications (pp. 429-437). Springer, New Delhi.

[Thaker, 2016] Thaker, T. (2016, March). ESP8266 based implementation of wireless sensor network with Linux based web-server. In Colossal Data Analysis and Networking (CDAN), Symposium on (pp. 1-5). IEEE.

[Vaccari, 2017] Vaccari, I., Cambiaso, E., & Aiello, M. (2017). Remotely Exploiting AT Command Attacks on ZigBee Networks. Security and Communication Networks, 2017.

[Vaccari, 2019] Vaccari, I., Cambiaso, E., & Aiello, M. (2019, submitted). Evaluating Security of Low-Power Internet of Things Networks, 2019 (submitted).

[Yick, 2008] J. Yick, B. Mukherjee, and D. Ghosal, "Wireless sensor network survey," Comput. networks, vol. 52, no. 12, pp. 2292–2330, 2008.

[Zarpelao17] Zarpelao, B. B., Miani, R. S., Kawakani, C. T., & de Alvarenga, S. C. (2017). A survey of intrusion detection in Internet of Things. Journal of Network and Computer Applications, 84, 25-37.