SEVENTH FRAMEWORK PROGRAMME

Theme SEC-2011.2.5-1 (Cyber attacks against critical infrastructures)

CRISALIS
SECURING CRITICAL INFRASTRUCTURES

D3.4 Return on Experience

Contract No. FP7-SEC-285477-CRISALIS

Workpackage	WP 3 - Validation Support
Authors	Magnus Almgren (CHA), Valentino Angeletti (ENEL), Davide Balzarotti (IEU), Jens-Uwe Bußer (SIE), Marco Caselli (UT), Vincenzo Gulisano (CHA), Michael Munzert (SIE), Heiko Patzlaff (SIE), Jan Stijohann (SIE), Valentin Tudor (CHA), Emmanuele Zambon (SM)
Editor	Jens-Uwe Bußer
Version	1.0
Date of delivery	M40
Actual Date of Delivery	M40
Dissemination level	Public
Responsible	ENEL

The research leading to these results has received funding from the European Community's Seventh Framework Programme (FP7/2007-2013) under grant agreement n°285477.

SEVENTH FRAMEWORK PROGRAMME

Theme SEC-2011.2.5-1 (Cyber attacks against critical infrastructures)

The CRISALIS Consortium consists of:

Siemens AG	Project coordinator	Germany
Alliander		Netherlands
Chalmers University		Sweden
ENEL Ingegneria e Innovazione		Italy
EURECOM		France
Security Matters BV		Netherlands
Universiteit Twente		Netherlands
University of Ulm		Germany

Contact information:
Michael Munzert
Siemens AG
Otto-Hahn-Ring 6
81739 Munich
Germany

e-mail: michael-munzert@siemens.com

Contents

Abstract

In this deliverable we provide return of experience with regard to the experiments performed within workpackage 3 in order to improve the security level of the future critical infrastructures.

1 Introduction and Scope

Several experimental environments have been set up within CRISALIS workpackage 3. These environments were used for validation of applicability and efficiency of the security mechanisms, tools and frameworks developed in CRISALIS. The validation was performed in form of a multitude of experiments.

In chapter 2, we present for each set of experiments a brief description and the main results as well as practical conclusions and lessons learnt from the outcome of the experiments and evaluations. Furthermore, we present some recommendations for further improvements of the tools and recommendations for security improvement of critical infrastructures.

Building upon these results and on the acquired experience in the detection and analysis of cyber attacks, we derive a set of recommendations for the improvement of the security of critical infrastructure environments in chapter 3.

2 Experiments

In this chapter, we present the experiments carried out within the context of the CRISALIS project:

- Testing fingerprinting techniques in Industrial Control Systems (UT)
- Using open source smart fuzzers (SM/CHA)
- Semi-automated vulnerability detection using vulnerability indicators (SIE)
- Firmware Analysis (IEU)
- Protocol Learning and Whitelisting (SM)
- Sequence-aware Network Intrusion Detection (UT)
- Critical Event Detection in AMI (CHA)
- Scalable and Online Intrusion Detection for AMI (CHA)
- Analysis of encrypted traffic in the Advanced Metering Infrastructure (CHA)
- The Influence of Smart Grid Data Characteristics on Privacy (CHA)
- Data Validation in Advanced Metering Infrastructures (CHA)
- Forensic Agent FERRET (SIE)
- Measurement of forensic drift in ICS (SIE)
- Host-Based Protection: Anomaly detection, heap spraying protection (IEU)

For each experiment, we first provide a detailed description of the performed experiment. The main results are described next, and practical conclusions are drawn. Furthermore, recommendations for improvement of the tools are given. In addition, also improvements of critical infrastructures are recommended.

2.1 Testing fingerprinting techniques in Industrial Control Systems

This set of activities is included into CRISALIS Working Package 4 and relates to the research performed by CRISALIS partners on network discovery.

Experiments consisted in testing standard fingerprinting techniques in Industrial Control System environments and, eventually, designing and testing custom solutions.

2.1.1 Detailed Description

We perform two different kinds of experiments. First, we used standard fingerprinting tools commonly used in IT. Among all, we especially focused on two fingerprinters:

- **P0f** is one of the most comprehensive passive TCP/IP fingerprinting tools. Among its characteristics there are: high scalability and fast identification, measurement of system uptime and network hookup, automated detection of connection sharing, and detection of malicious hosts.
- **Nmap** is the most important and well known tools for TCP/IP fingerprinting. The main features include: host discovery, port scanning, version detection, OS detection and scriptable interaction with the target.

P0f and Nmap well represent the two opposite aspects of fingerprinting, namely active and passive fingerprinting. The first relies on the traffic captured on the network to recognize active devices. The second performs network discovery by probing devices and opening new communication channels. This last approach usually improves the chances to correctly fingerprint devices.

Furthermore, we developed a further fingerprinting tool specifically customized for Industrial Control Systems. The **Flow Fingerprinter** attempts to recognize ICS devices by looking at network traffic and communication patterns. The tool observes the "behavior" devices have on the network and fingerprints the roles such devices have within a specific Industrial Control System.

We tested standard fingerprinters and the Flow Fingerprinter against several targets. We used both Enel testbed (Idrolab) and University of Twente testbed described in CRISALIS Deliverable 3.1 [2]. Other than the testbeds, we tested aforementioned fingerprinting techniques against a number of different devices and datasets.

To broaden the set of experiments, we performed further tests on different PLCs:

- ABB AC800M

- DATAwatt D05-MCU 60870-5-104
- Janitza Electronics UMG 604

Moreover, to improve the quality of the experiments, we tested some passive fingerprinting techniques against three network traffic traces captured on real critical infrastructures. These facilities are:

- a water treatment and purification plant
- a gas storage facility
- a gas distribution facility

All utilities were running normally while capturing traffic and operators were free to perform any scheduled or unscheduled operation on the infrastructures.

2.1.2 Main Results

Our tests showed that no standard fingerprinting tool was able to correctly fingerprint PLCs. This was mainly due to the lack of information related to ICS in tools' datasets. However, from a practical point of view, this was not the only problem. For example, comprehensive tools such as Nmap were able to correctly identify some network services (e.g., some open TCP connections) used by PLCs. However, there was no smart mechanism aiming to solve ambiguities in responses. This caused correct information to be discarded due to a simple "majority voting" decision (wrong responses always outnumbered correct ones). Another issue affects tools using information that ultimately invalidates fingerprints. This is the case of TCP/IP fingerprinters. Tests showed that focusing on some characteristics of devices' network stack implementations does not work well for PLCs due to incomplete or customized implementations.

The Flow Fingerprinter improved results obtained with standard fingerprinting techniques by exploiting different characteristics of the network. Focusing on traffic flows and devices' behaviors allowed the tool to effectively recognize devices' roles on the network with a higher degree of accuracy.

2.1.3 Practical Conclusions / Lessons Learned

These experiments allowed us to define a set of fingerprinting "challenges and opportunities" within Industrial Control System environments.

Challenges

A set of ICS network characteristics makes device fingerprinting more challenging compared to regular Internet or company LANs. These characteristics can be summarized as follows:

- **Active vs. Passive Fingerprinting**: ICSs often monitor or control processes in which a failure may have disastrous consequences. For this reason active probing (such as scanning for open ports and then opening arbitrary TCP connections) should generally be avoided and it is always preferable to use only passive monitoring techniques.
- **Device Heterogeneity**: ICS can monitor and manage many different physical processes. In each of them devices perform a wide range of actions such as: data collection, sharing of information, coordination control, etc. Moreover, ICS vendors usually develop their own devices for each one of the previous activities. Creating several unique and reliable fingerprints can be difficult as this would need accessing several different installations in order to reach ubiquitous coverage.
- **Proprietary protocols**: ICS devices mainly implement vendor-specific protocols. Several of these implementations are not published or documented and it is difficult to tune device fingerprinting on unknown communication.
- **Long-running TCP Sessions**: Most popular tools for TCP/IP fingerprinting (both active and passive) takes advantage of SYN and SYN-ACK packets exchanged during connection establishment. Unfortunately, once a PLC and a control server set up a TCP connection, the session is likely to remain open for a very long time (days or even weeks). This issue makes standard passive fingerprinting almost impossible.

Opportunities

Despite the aforementioned challenges, we observed some properties fingerprinting tools can leverage to effectively recognize devices on an ICS network. These are:

- **Long life-cycle of devices**: industrial control systems are used since the '70s and several system operators do not continuously update their hardware and software components. Devices' life cycles are usually long enough to guarantee a fingerprint to remain valid for years.

- **Stable topology and communications**: the number of devices as well as temporal and communication patterns do not usually change in ICS networks. Fingerprinting can exploit this stability linking such patterns to component signatures (as the Flow Fingerprinter does).

- **Protocol Specification**: some ICS protocols, like Modbus, Profinet and MMS provide a way to query components in order to have information about their hardware and software. Fingerprinters can exploit such functionalities and enrich signatures with more comprehensive and precise information.

2.1.4 Recommendations for Improvement

Fingerprinting is an important building block of many security related activities and it is often used in unknown IT environments. For example, several penetration testing methodologies state that system discovery, including device fingerprinting, is often the first step towards vulnerability identification. Moreover, fingerprinting is used to support security tools in order to increase the accuracy of the assessments (i.e. together with intrusion prevention or detection systems).

In ICS environments, a reliable technique to recognize devices can be useful to improve industrial penetration testing techniques and security checks. To achieve this result we emphasize the following recommendations:

- This set of experiments aims to provide a broad overview of different fingerprinting techniques. Each one of them leverages different network characteristics and contributes to a more complete and clear picture of an industrial control system. We believe that, the more information and properties a fingerprinter is able to analyze, the more comprehensive and meaningful the result will be. *Integration* of different fingerprinting solutions is a goal that further research on the topic will need to deepen.

- Lack of real data is one of the main concerns for ICS security researchers. Thanks to CRISALIS partnerships we had the possibility to dig deep into ICS traffic and understand challenges and opportunities towards effective fingerprinting techniques. Despite this, we emphasize the need for further information from different ICS environments to design and develop even more comprehensive tools. We believe that stronger relationships between ICS vendors, operators and researchers should be encouraged. This should especially include free sharing of data and resources that will eventually improve the ICS security tool market.

- Finally, Critical Infrastructures operators should support the use of open standards and protocols. This would improve any security tool that bases its analysis on large datasets (as fingerprinters do). Acquiring information on open standards and protocols means being able to use such information not just in one but in several critical infrastructure environments. This solution will also help ICS operators to fairly compare tool results and, ultimately, to improve security assessments of their infrastructures.

2.2 Open source smart fuzzers

The open source smart fuzzers were developed as part of Workpackage 5, in the context of automated vulnerability discovery techniques. These are to be used to proactively find possible vulnerabilities for equipment and network owners. The validation of the developed fuzzers consisted of using them in different laboratory environments to test the robustness of a number of ICS and AMI devices (PLCs and smart meters).

2.2.1 Detailed Description

The open source fuzzers were developed to bridge the gap where on one hand there is a need to analyze critical components in ICS and AMI and on the other hand, the methodology or the toolsets are missing. The full details of the development of the open source fuzzers are described in Deliverable D5.2 and D5.3.

We carry out the tests for ICS devices in three environments: a local laboratory environment available at SecurityMatters, the University of Twente testbed and the ENEL laboratory described in the CRISALIS deliverable D3.1 [2]. We carry out the tests for AMI devices in two environments: a local testbed at Chalmers and the Alliander testbed described in the CRISALIS deliverable D3.1 [2]. It is important to have a range of different environments for the testing of the fuzzers. In the case of the smart meters, for example, our tests covered two important types of meters.

The following experiments were run.

- Modbus/TCP PLC: with an initial investigation of supported commands for the PLC.
- OPC Server: a more complex case than the above as proper authentication needs to be implemented
- PP PLC: using a proprietary protocol that we also analyzed in the protocol learning Workpackage

- PP-OPC gateway: translating commands received on a PP interface to an OPC interface
- UPP PLC: fuzzing done through a mutative approach
- ITOT interface of PLCs: mutation-based fuzzing of the COTP connection
- M-Bus Smart Meters: fuzzing the older standard M-Bus that are used by some European Smart Meters
- DLMS/COSEM Smart Meters: fuzzing of the widely used standard for a range of European Smart Meters

2.2.2 Main Results

We run experiments using our open source smart fuzzers to discover vulnerabilities in the different test environments built in the context of the project. During these experiments we discover vulnerabilities in different types of equipment (primarily embedded devices such as PLCs and smart meters).

Most of the identified vulnerabilities have been already disclosed to the respective vendors and were confirmed. In more details, one of the vulnerabilities was already known to the vendor, who confirmed to us it was fixed in a newer version of the firmware, with regards to the one we had available in the laboratory. The remaining ones were previously unknown. One of the vendors announced that a fix for the reported vulnerabilities was planned for the first quarter of 2014. The same vendor also showed interest in the tools we used to discover the vulnerabilities.

For the vulnerabilities found in the ENEL laboratory, we created a detailed report explaining their nature and possible consequences. The report was distributed to the managers of the sites which employ the same technology present in the laboratory in order to allow site managers to take appropriate countermeasures (e.g., to promptly install the patched firmware once the vendor makes it available).

2.2.3 Practical Conclusions / Lessons Learned

Summarizing the lessons learnt during our fuzzing experiments we notice that, not surprisingly, it is relatively easy to find (new) vulnerabilities in legacy devices. This confirms the widespread notion that ICS equipment was not built with cyber-security in mind. From our talks with ICS and AMI vendors, we learn that robustness security testing procedures are starting to be adopted by vendors, especially after the importance of cyber-security was brought to the general attention by the Stuxnet [6] case and by the publication of dozens of new vulnerabilities [3] in ICS systems. However, the number and diversity of devices and protocols makes it hard to achieve proper coverage. In fact, in some of our tests we discovered vulnerabilities even in devices (or device firmware versions) that were released after the adoption of such procedures.

In the majority of the cases, the vulnerabilities discovered with our fuzzing experiments cause a denial of service in the system under test (SUT). However, a more in-depth monitoring of the SUT during fuzzing might allow to discover more subtle vulnerabilities, which might affect the correct functioning of the devices without necessarily causing a crash. This is achieved in traditional IT software fuzzing by instrumenting the SUT with debuggers or memory profiling tools during the tests and logging code exceptions or excessive memory usage. The Avatar tool discussed in deliverable D5.3 [1] allows for such instrumentation in embedded devices.

2.2.4 Recommendations for Improvement

Software / Component View

- More robustness is needed. ICS and AMI devices should be at least as “good” as regular IT, if not better. At the moment they are much less robust than regular IT.
- When a PLC crashes it is not immediately clear to operators. Although some PLCs (e.g., safety critical PLCs) are equipped with additional monitoring capabilities to detect power outages, crashes, etc., monitoring the health of process control equipment can be improved.
- Regular IT security is guaranteed by quick and easy updates. Updating the firmware is at the moment a limiting factor for the security of ICS devices (not sure about meters). Vendors should make it easier to update firmware.

Technical Environment / Architectural View

- We noticed that it is very hard (even for vendors) to debug the cause of certain crashes in embedded devices. This because sometimes even the vendor has no access to a debug interface on the device. This should be improved.
- Logging should be improved too. Certain health conditions of the devices can be monitored (e.g., by means of SNMP), but this is not currently possible for all devices.

Organisational Aspects

- Our recommendation is thus that European industries producing or using smart meters request the developed tools in the CRISALIS deliverable D5.2 (classified as "EU RESTRICTED") to comprehensively test their particular brand for vulnerabilities.
- Same for ICS vendors.
- Make robustness testing part of the standard testing methodology for all ICS equipment. Include legacy devices as well, since they are still widely used.
- It is a long way from reporting a vulnerability to a patched component because several stakeholders have to be involved: The security researcher who finds a vulnerability has to report it to the device vendor. The vendor has to analyze it, and to provide a patch. The patch has to be published, but it is not possible for vendors to inform all operators to install an announced patch because vendors usually do not know where their components are used. When the operator is ready to install the patch after comprehensive testing for no undesired side effects, he may have to wait for the next maintenance interval of the plant if the patch installation is not possible during ongoing plant operation, e.g. because of a service disruption of the component. Furthermore in some critical infrastructures (e.g. railroad), safety reasons require for any (even very minor) change carrying out a new sophisticated, protracted and expensive admission procedure with the respective regulatory agency.

2.3 Building a semi-automated Indicator-based Vulnerability Detection System

The experiments were performed in WP 5 and WP 7 and accompanied the design and implementation of an indicator-based vulnerability detection system as described in D 5.3.

2.3.1 Detailed Description

The experiments included the following activities:

- Evaluating different platforms for the analysis system: Windows, Linux, 32 or 64 bit

- Evaluating and integrating 3rd party tools and libraries
- Identifying a set of suitable test samples to verify the functioning of the system. The samples should include real world samples (e.g. the Linux tool "curl") but also self-crafted samples with and without certain vulnerabilities
- Validating the correct functioning the implemented system components given selected samples
- Prototyping a new approach for the integration of the analysis system into a (distant) SUT's environment

2.3.2 Main Results

The main results were

- an architecture and design of the system,
- the prototypical implementation of major system components, among others, an automated control flow path finder,
- a concept for a new approach to integrate the analysis components into a (distant) SUT's environment, based on packer and vagrant[1].

2.3.3 Practical Conclusions / Lessons Learned

We draw three main conclusions from the experiments:

1. A low false positive (FP) rate is a key factor for industry acceptance and a real-world impact. The use cases in which a low FP rate is preferred even at the costs of higher 'miss rates' and lower 'hit rates' turned out to be much more important than we initially thought. Examples of such use cases with a high importance for CI security are:
    - System tests performed by non-security-experts:
      Here, the tester has neither the time nor the expertise to deal with false positives. The results have to be unambiguous to provide a concrete benefit to him.

[1]See `http://packer.io/` and `https://www.vagrantup.com/`

- Fully automated security analysis solutions:
  The advantages of a fully automated security analysis (costs, time) are lost if the results need to be manually triaged.

  Yet, we discovered a lack of tools and approaches which support this specific setting of priorities.

2. Developing technology-independent analysis systems is a challenge. By technology-independent we mean a system working on different operating systems and being able to analyze samples of different types (see below). To demonstrate the exploding complexity, the following list gives some exemplary factors that determine the actual technical realization of a binary analysis system[2]:
   - Operating System, e.g. Windows or Linux, 32 or 64 bit
   - Binary Format, e.g. ELF, PE, or Mach-O
   - CPU architecture, e.g. X86, ARM, MIPS
   - Dynamically or statically linked
   - Compiled as position independent or non-position independent code

   Even after deciding to first focus on ELF binaries (most commonly found in embedded systems integrated in CIs), we discovered a set of more than 15 CPU-dependent specification 'supplements'[3], especially relevant for dynamically linked ELF binaries.

3. For a large scale (semi-)automated security analysis of complex software systems, the creation of the SUT's runtime environments is a bottle-neck: Currently, this step involves the manual installation and set-up of a new virtual machine - a process that it tedious, error-prone and time-consuming.

2.3.4 Recommendations for Improvement

The security of critical infrastructures can be improved in the future if we manage to develop automated security testing solutions with extremely low FP rates. The system designed in WP 5 (see D5.3) and continued in WP 7 and WP 3 could be adapted for a higher automation degree and the different analysis modules could be used as overlapping checks to filter out all but extremely solid results.

[2]whose scope is already narrowed to native executables, i.e. not comprising managed code such as .NET, Java, or Python

[3]see `http://en.wikipedia.org/wiki/Executable_and_Linkable_Format#Specifications`

Technical Environment / Architectural View

Automated solutions to create a working SUT-runtime environments are the key to large-scale automated testing. The more automated (e.g. via shell scripts) the installation, configuration, and deployment of a CI is, the better it can be integrated in automated testing solutions. If both, the testing and the SUT, use the same technology for this purpose, there is even less overhead to realize an automated testing during development and operations.

Organisational Aspects

As discussed in the previous recommendation, we suggest to automate the installation, setup, deployment of software components used in critical infrastructures. Without organizational support, e.g. via corresponding policies, trainings, and provisioning of the required infrastructure, this will not be possible at a large scale.

2.4 Firmware Analysis

The goal of these experiments was to test the platform that we developed to perform sophisticated analysis of firmware images. The tool, named Avatar and explained in more details in deliverable D5.3, enables complex dynamic analysis of embedded devices by orchestrating the execution of an emulator together with the real hardware.

Firmware analysis is a fundamental component in two CRISALIS workpackages: WP5, as a tool to detect or investigate vulnerabilities, and WP7, to support the forensic examinations of compromised or backdoored devices.

Avatar was tested on a number of heterogeneous embedded devices, to emphasize the different use cases and the different characteristics of the tool.

2.4.1 Detailed Description

The list of test devices on which Avatar was tested include:

- *A Commercial Zigbee Device*: an all-in-one device for experimenting with low power wireless protocols based on the IEEE 802.15.4 standard, such as Zigbee. The device is built around the MC13224v System on a Chip from Freescale, that is in turn built upon an ARM7TDMI microcontroller and includes several memories, peripherals and has an integrated IEEE 802.15.4 compatible radio transceiver.

- *A Common Feature Phone*: a Motorola C118 GSM feature phone that, in contrast with most recent and advanced mobile phones and smartphones, has one single embedded processor for both the network stack (i.e., GSM baseband capabilities) and the Human-to-Machine Interface. The phone board makes use of the Texas Instruments "Calypso" digital baseband, which is composed of a mask- ROM, a DSP for GSM signal decoding, and a single ARM7TDMI processor.

- *A Hard Disk Drive*: a commercial off-the-shelf SATA drive from a major hard disk manufacturer. It contains an ARM 966 processor (that implements the ARMv5 instruction set), an on-chip ROM memory which contains the masked ROM bootloader and some library functions, an external serial flash that is connected over the SPI bus to the processor, a dynamic memory (SDRAM) controller, a serial port accessible through the master/slave jumpers, and some other custom hardware that is necessary for the drive's operation. The drive is equipped with a JTAG connection.

- *A Programmable Logic Controller (PLC)*: a very common PLC device, for which we have been asked to anonymize the model and the manufacturer. This test is particularly important for critical infrastructures, where PLC play very important roles and can be the target of cyber attacks. It was also very important for Avatar, since this was the only device not equipped with dedicated monitoring ports – and therefore required a considerable amount of manual work to connect it to our tool.

2.4.2 Main Results

Our tests by no means cover all the possible scenarios in which Avatar can be applied. However, we believe they are different enough to show a wide range of possible uses of our tool and of dynamic analysis of firmware images in general.

For each of the devices listed in the previous section we performed a number of different experiments, ranging from simple reverse engineering tasks, to the automated analysis communication protocols using symbolic and concolic executions. In other experiments we implanted a vulnerability in the firmware code and then use Avatar to detect its presence. We also tested a complex setup in which Avatar was used to pilot an external fuzzer, thus showing how this tool could be integrated with other software developed as part of the CRISALIS project.

As a result, we can consider all tests were successfully executed.

2.4.3 Practical Conclusions / Lessons Learned

The different setup and goals of our experiments showed the flexibility of Avatar in different environments. However, Avatar is a framework that does not produce any results per se, but instead needs to be operated by expert analysts. Similarly to a common software debugger (which does not automatically find any bug), Avatar does not perform vulnerability analysis or forensics of embedded devices. However, it is the first tool that – in the right hands – makes this analysis possible.

2.4.4 Recommendations for Improvement

Software / Component View

Avatar is currently an open source tool that has attracted a lot of interest from many companies and researchers. The tools is still a prototype, and much work is needed to improve its functionality and in particular its usability. We are aware of several limitations that need to be addressed, and we are currently working to make Avatar a more solid and stable dynamic analysis tool.

Technical Environment / Architectural View

If tools need to be used to test the firmware of PLCs and other embedded devices in a critical infrastructure, these devices need to be designed with accessibility in mind. In other words, the PLCs should have either an hardware functionality to dump the firmware, or a debugging port to connect to the live device.

2.5 Protocol Learning and Whitelisting

The Protocol Learning and Whitelisting (PLW) approach was developed and built in SecurityMatters SilentDefense intrusion detection system as part of Workpackage 6, in the context of network-based and protocol-aware intrusion detection for ICS/SCADA environments. The validation of the approach consisted of using the tool in two different laboratory environments to test its ability to detect previously unknown and sophisticated attacks against these systems and to provide visibility into the network operations of process automation and control systems.

2.5.1 Detailed Description

The PLW approach was developed to provide visibility into the network operations of ICS/SCADA control systems and to allow security operators to detect (even previously unknown) attacks against these systems. The full details of the PLW approach are described in deliverables D6.1 and D6.4.

We carry out two sets of tests in two different laboratories: the ENEL cybersecurity laboratory, and another laboratory set up to study the Stuxnet worm.

In the ENEL cybersecurity laboratory we tested the detection capabilities of the PLW approach with respect to the detection goals defined in the context of work package 2 (see D2.2). In more detail, we developed and tested attacks against DCS server, suspicious or anomalous traffic between DCS server, Engineering workstation and PLCs, and critical commands sent to PLCs. The attack techniques ranged from attempting unauthorized access, crashing devices through malformed messages, scanning devices for information and replaying valid commands to trigger unwanted functionalities.

In the laboratory set-up to study the Stuxnet worm we tested the ability of the PLW approach to both provide visibility into the network operations of ICS/SCADA control systems and to detect the Stuxnet worm without leveraging any signature. In more detail, the data at our disposal was divided in two parts: a clean dataset, containing normal operations and a virus dataset, captured while the hosts in the environment were being infected with Stuxnet. We proceeded with the following main steps:

1. We built for reference a diagram of the control system network in the laboratory from the network traces of the clean dataset using standard tools (e.g., Wireshark).

2. We deployed three PLW profiles: one for the SMB protocol, one for the DCOM/OPC protocol, and one for the Siemens Step7 protocol.

3. We processed the clean dataset with the three PLW profiles in learning mode and analyzed the information collected in the profile models.

4. We processed the virus dataset with the three PLW profiles in detection mode and analyzed the collected alerts.

2.5.2 Main Results

The tests carried out at the ENEL cybersecurity laboratory demonstrated the ability of the PLW approach in detecting both attacks against Windows components and services (such as the DCS server and the Engineering WorkStation) and attacks targeting specific ICS/SCADA protocols and components (such as PLCs) with a very low false positive rate. The lowest false positive rate could be achieved after tuning the detection to specific byte fields in the model.

The tests carried out with the Stuxnet laboratory dataset showed that the PLW approach can provide useful insights about the type of operations carried out in a ICS/SCADA control systems. For example, we discovered what type of file sharing operations were carried out between the Windows computers, we discovered what data sharing operations were carried out between the different OPC servers and we discovered that the PLC logic was being commissioned (i.e., programmed) at the time the clean dataset was being captured.

The tests also demonstrated that the PLW approach is capable of detecting the anomalous network communications resulting from complex targeted attacks such as the ones carried out by Stuxent at almost every observable phase of its life: propagation, peer-to-peer communication, PLC monitoring, and infection.

2.5.3 Practical Conclusions / Lessons Learned

Many dangerous attacks can be detected by monitoring the network of ICS/SCADA systems. Our experiments with the Stuxnet dataset, for example, show that even advanced and targeted attacks meant to go unnoticed leave traces on the network that differ in many aspects from normal traffic. In addition, by baselining the network operations of an ICS/SCADA system one can learn important insights about how the network and control system is used, and about possible misconfigurations or malfunctioning.

Although some of the most simple attacks we tested could in principle be detected through other less accurate approaches, only a deep analysis of the application-layer protocol allows detecting sophisticated targeted attacks. For instance, some of the operations of Stuxnet (e.g., PLC monitoring and infection) can only be detected because of the unusual value of some message parameters. For this, detection tools need the ability to inspect all parts of each message at full depth.

Many dedicated protocols are used in ICS/SCADA environments, with a large amount of functionality duplication motivated by historical and commercial reasons. This has an obvious impact on security: higher chances of design flaws and implementation vulnerabilities. It is therefore crucial for successful intrusion detection to develop a wide coverage of these protocols. With the increase of detection capabilities for standard ICS/SCADA protocols we foresee that motivated and resourceful attackers wishing to go unnoticed on the network will target proprietary protocols to avoid detection. Fortunately, our efforts with proprietary protocols showed that it is feasible for intrusion detection systems to achieve such coverage.

Our tests at the ENEL laboratory pointed out the need for detection algorithms to be flexible. For example, not all parts of a protocol message are equally useful or significant for intrusion detection. In the same way, the importance of monitoring certain protocol fields may depend on the specific environment the monitoring system is deployed on. For this reason, it is important for intrusion detection systems to provide a transparent and flexible way of tuning detection algorithms and models.

2.5.4 Recommendations for Improvement

Software / Component View

- Proprietary ICS/SCADA protocols have proven to be as vulnerable to attacks as open protocols, with the disadvantage that it is more difficult to build intrusion detection support for them and to build a solid understanding of the key aspects to be monitored for security. We therefore encourage an evolution towards a fewer standard protocols.
- Ongoing efforts for securing ICS/SCADA protocols should take into account the specific security requirements of ICS/SCADA systems (Availability and Integrity more important than Confidentiality). Current standardization efforts are considering retrofitting these protocols with standard IT security solutions (TLS, IPsec). However, encrypting data on the network would have a negative impact on the ability of security operators to monitor the traffic, without a real need for confidentiality. Other solutions enforcing machine to machine authentication and message/-data integrity should be adopted instead, that still allow network communications to be monitored for operations and security.

Technical Environment / Architectural View

- While establishing strong perimeter security (e.g., by means of firewalls or network data diodes) is a key first step to secure ICS/SCADA networks, we believe monitoring the inner parts of the network for security is equally important, since motivated attackers have proven to be able to circumvent perimeter protection measures.
- When setting up network monitoring for field network segments (e.g., the segments that connect PLCs and other low-level equipment) we noticed that, in order to achieve fault tolerance, network administrators set up Ethernet networks in ring configurations. In this configuration, switches are connected to each others to form a ring, so that packets can flow in any direction in case one of the switches breaks. Unfortunately, this makes it hard to monitor the network traffic, since in order to

capture all the traffic one should set up a monitoring port in every switch of the ring. We believe effort should be spent to analyze this issue and find technical solutions for it.

Organisational Aspects

- Our experiments show how network monitoring can improve the awareness and visibility of network operations in ICS/SCADA networks. We therefore recommend that ICS/SCADA specific intrusion detection and monitoring techniques become standard practice in the security strategy of utilities.
- To monitor ICS/SCADA systems for security one needs first to understand the environment and the technology being used. In many critical infrastructure organizations the technical knowledge of protocols used for network communications and their possible misuse is missing or concentrated in few key people who are not involved in cyber-security. Our PLW approach can help operators to gain the understanding of network operations needed to successfully establish a monitoring process.
- Visibility and awareness alone are not enough to protect against cyber-attacks. Critical infrastructure organizations should establish processes and procedures to use the early detection capabilities being developed in projects such as CRISALIS to become resilient against the attacks.

2.6 Sequence-aware Intrusion Detection

This set of activities is included into CRISALIS Working Package 6 and relates to the research performed by CRISALIS partners on network intrusion detection systems.

The experiment consisted in testing a custom sequence-aware intrusion detection system on traffic traces captured within real critical infrastructures.

2.6.1 Detailed Description

Performed tests involved the use of the ENEL testbed described in CRISALIS Deliverable 3.1 [2]. As in the fingerprinting case (Section 2.1), we also used three network traffic traces captured on real critical infrastructures. Such utilities were:

- a water treatment and purification plant

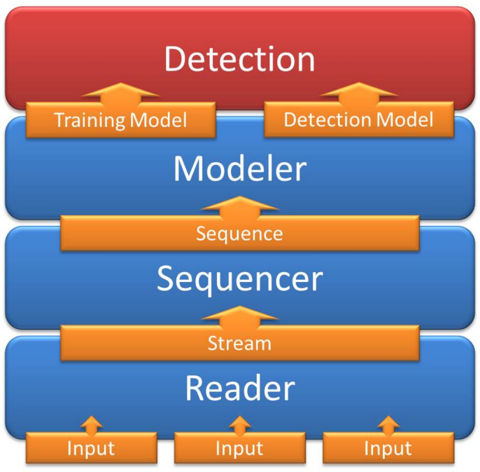

Figure 2.1: S-IDS general architecture

- a gas storage facility
- a gas distribution facility

One experiment included two sequence attacks being injected into traffic traces. We crafted the two malicious sequences within a Modbus communication of the water treatment and purification plant network. The attacks consist of:

1. inverting two write messages concerning the control of two different pumps
2. triggering several start and stop commands on the same pump

To detect the attacks we used a “Sequence-aware” Intrusion Detection System (S-IDS). The tool was designed based on the framework shown in Figure 2.1.

The Modeler implements algorithms that transform sequence of network events (e.g., network messages) into discrete-time Markov chains (DTMC).

2.6.2 Main Results

Our first tests showed that the S-IDS is able to correctly identify sequence attacks injected into traffic traces. However, the number of false positive was too high to make S-IDSs effectively deployable in industrial networks. We believe that a feasible rate of alerts given to the operators should be in the order or 1-2 per day. The rate obtained with the first tests would have forced an operator to check several alerts every hour.

To solve this issue, we proposed several enhancing techniques. One of them, called "Event Importance Enhancement" allowed us to substantially decrease the number of false positives without losing any correct alert. The "Event Importance Enhancement" relies on the possibility to assess the importance of all states in the discrete-time Markov chain and focus the detection on a specific subset states. The rate of alerts ultimately decrease to the value of one alert every two hours. Despite still unfitting the aforementioned assumption of 1-2 alerts per day this result, combined with the other enhancing techniques, could eventually decrease to an acceptable rate.

2.6.3 Practical Conclusions / Lessons Learned

The value of this set of experiments is twofold. First, we showed the feasibility of sequence attacks by crafting actual malicious sequences and injecting them into real traffic traces. Second we showed how it is possible to detect sequence attacks and, eventually, decrease the number of false positives.

In particular, the framework designed for Sequence Detection is rather general to deal with a large set of possible scenarios. This allows the development of flexible solutions that respond to different requirements. For example, the Reader allows to abstract from a specific source of information (e.g., network traffic traces, log files, memory dumps). In the same way, the Sequencer can work on different data (e.g., in case of a network traffic trace: Modbus, Manufacturing Message Specification, IEC104) and the Modeler can implement different modeling techniques. This special attention on flexibility was due to the high variety of technologies and protocols found in ICS.

Finally, we focused our experiments on anomaly-based intrusion detection. This was primarily due to the effort needed to deploy security solutions that require heavy configurations. Operators working on large critical infrastructures cannot check signatures needed to signature-based intrusion detection to ensure the quality of the alerts. Moreover, signature-based solutions do not target unknown attacks (e.g., Zero-day attacks). This issue limits the applicability of such solutions into environments subjected to targeted attacks where the likelihood of crafting highly-specialized attack is significant.

2.6.4 Recommendations for Improvement

As discussed in the introduction, critical infrastructures are now increasingly connected among them and to the Internet. This setup makes them a feasible target of cyber-attacks. More than standard IT threats, "semantic attacks" such as Stuxnet are particularly dangerous for critical infrastructures. This is because they rely on knowledge about the control processes (or even the physical systems controlled by ICSs) and use such knowledge to maximize damages inflicted to the physical world.

To defend infrastructures against this kind of attacks ICS security researchers and operators should work on converging IT security and process automation together into comprehensive software solutions. This concerns both detection mechanisms and results. The first need as much information as possible to correctly recognize semantic attacks. The second need the same information to effectively advice operators on how to deal with the identified concerns and finally solve the problems.

2.7 Critical Event Detection in AMI

The work has been conducted as part of WP 6 and in particular in the task dealing with large scale distributed systems. In the case of large infrastructures with a large number of devices, it becomes a challenge to efficiently process events. In particular, we have considered clustering of "power outage" events, that in a major outage in a city result in a very large number of almost simultaneous events.

2.7.1 Detailed Description

This work was developed to better understand the constraints on processing of events from large scale advanced metering infrastructures, be it alarms from the meters or other types of events. By considering a baseline algorithm in literature, we investigate improvements to scale to large sets of almost simultaneous events, where also some events may be lost. In case of alarming situations in AMI, e.g. power outage, affected meters suffer the power failures almost at the same time and the alarm messages are sent within a short time interval (e.g. a couple of seconds). To detect such a power outage, we need to identify a *temporal-spatial related set* of records from a continuous data stream of alarm messages. We formalize the problem as follows. Messages indicating the events of interest are reported by the sensors over time, forming a data stream $\mathcal{S} = s_1, s_2, \cdots$, where s_i is a data record, i.e., a message. We use t_i to denote the time-stamp of s_i, which is the message-generation time. For a pair of data records s_i and s_j, we use $d(s_i, s_j)$ to denote the geographical distance between the corresponding sensors.

We implemented G-SLC (see Deliverable 6.4 for details) in JDK 1.7 on an Intel Xeon E5645 2.4GHz machine with 48GB memory. The operating system is Ubuntu 12.04 with Linux kernel version 3.2.0-53. In addition, we implemented the SLC method [9] whose performance acts as the baseline in the evaluation. The evaluation utilizes a data-set from an Advanced Metering Infrastructure (AMI) of a city with over 500,000 inhabitants and about 450 km^2 in size. The AMI contains around 300,000 smart meters which wirelessly report faults and errors in power lines. The goals of the validation was to especially study

- memory usage requirements
- computation time
- clustering performance
- detection ratio
- detection precision

2.7.2 Main Results

The study showed for the newly proposed algorithm G-SLC, high accuracy with a couple of orders of magnitude improved detection time compared with the baseline algorithm. Even in cases when many events happen at the same time (e.g. power outage, large attack), G-SLC scales appropriately without losing significant detection ratio or precision. In the evaluation of its detection accuracy, we compared the detected power outages with the ground truth based on information from a call center where customers call in about power outages. We also investigated the tradeoff between cell sizes (allows fast computation) with detection accuracy. With larger cells, the location of the critical events given by G-SLC may be very coarse, whereas smaller cell sizes can offer fine-grained localization performance. This would be a parameter to be tuned by the utility.

2.7.3 Practical Conclusions / Lessons Learned

Across Europe, different types of meters are deployed. In some cases, the events (consumption and alerts) must be pulled from the meters. In other cases, a push-based model is used where the meter attempts to send the information upwards in the network topology in case of disturbances. The study shows that it is possible to have stream-based, near real-time detection of critical events in AMI in a scalable manner. In turn, this implies that the AMI infrastructure can be more useful than just for reporting consumption measurements.

2.7.4 Recommendations for Improvement

Given the successful study, it is shown that it is possible to process critical events in AMI in a scalable manner. This in turn means that a clear recommendation is to build on AMI for "lower voltage SCADA" functionality in electricity networks (i.e. SCADA-type functionality at the lower tiers of the electricity networks, which is currently not available), enabling further grid control measurements to ensure good power quality also in the distribution network, and, in the future, tie this system together with other monitoring systems in the smart grid. Of course this needs to go hand-in-hand with securing the infrastructure against malicious use, hence the matching work on IDS and monitoring of traffic in the same WP [7, 11]. Allowing the meters to monitor themselves and report additional security events will by itself make the infrastructure more secure.

2.8 Scalable and Online Intrusion Detection for AMI

As part of WP 6, in task 6.3 focused on large-scale distributed approaches, we develop new methods for attack detection in advanced metering infrastructures. The experiments run to evaluate the proposed intrusion detection technique (METIS) aim at not only studying its detection capability, but also its possible use in real-world applications.

2.8.1 Detailed Description

One of the challenges in the advanced metering infrastructure, as opposed to many SCADA networks, is that all traffic is encrypted to, among other purposes, protect the privacy or confidentiality of the collected energy consumption readings. However, this in turn makes it challenging to monitor the network for attacks. Another challenge is that AMI across Europe differ in its deployment details. Any detection framework should be able to adapt to the local network topology and be able to efficiently run the detection rules that are important for the particular utility in question. In this particular experiment, we have two goals. First, we develop a general scalable framework for detection of attacks and other unwanted events, that the *utility owner* can adapt to his own needs depending on his particular type of infrastructure. The detection engine is two-tier and built to be easily distributed depending on the network infrastructure. Second, we show that even with information gained from the network communication we can detect attacks.

We carry out different experiments aiming at a comprehensive evaluation of the proposed intrusion detection scheme. More concretely, the evaluation covers the following aspects:

- **Detection Accuracy.** Focusing on the detection capabilities (i.e., true-positive and false-positive rates).
- **Parameters Sensitivity.** Studying how the different parameters (and the changes in them) can affect the resulting detection.
- **Processing Capacity**. Studying the performance (in terms of throughput and latency) of the proposed intrusion detection scheme.

2.8.2 Main Results

The evaluation shows that the proposed intrusion detection scheme has good potential, as shown by its resulting detection capabilities (observed around 90% in the experiments). At the same time, it results in a low false-positive rate, which could be indeed managed by a system expert in charge of analyzing the alarms being reported by the intrusion detection application.

At the same time, it also shows an excellent result in terms of performance (this in turn enables for such scheme to be leveraged in real-world scenarios). We observed processing throughput of up to 2000 tuples per second. In the given setup, this would translate to the ability of monitoring the data reported by more than 3 millions smart meters with off-the-shelf hardware. At the same time, the detection itself results in negligible processing times (in the realm of hundreds of milliseconds).

2.8.3 Practical Conclusions / Lessons Learned

The proposed architecture relies on different modules in order to perform intrusion detection. Different practical conclusions and lesson learned can be summarized looking at the different architecture's modules.

- The use of online processing paradigms such as data streaming (the one used in this evaluation) is of fundamental importance in order for any proposed intensive data analysis method to detect possible threats in a real-time fashion.
- The system expert knowledge (of both the system itself and of the nature of a possible threat) is also a fundamental piece of a comprehensive intrusion detection scheme. In this sense, as we did in our work, it is important to design and develop solution that ease the interaction with the system expert during the deployment and use of the application.

- Finally, distributed analysis is also important in order to spot possible threats by accessing data that, being local to the devices deployed in the field, would not be observed by a traditional centralized detection scheme.

2.8.4 Recommendations for Improvement

Software / Component View

- Improved and automatic translation of possible intrusion detection rules expressed by the system expert. In it's current shape, some parts of the translation process require the intervention of a programmer.

2.9 Analysis of encrypted traffic in the Advanced Metering Infrastructure

As part of WP 6, we develop new methods for attack detection in advanced metering infrastructures. The following considers building an intrusion detection system, monitoring the use of smart meters by, for example, building a model of the commands exchanged between the meters.

2.9.1 Detailed Description

One of the challenges in the advanced metering infrastructure, as opposed to many SCADA networks, is that all traffic is encrypted to, among other purposes, protect the privacy of the collected energy consumption readings. However, this in turn makes it challenging to monitor the network for attacks.

The purpose of this particular experiment is to test the efficiency of a module for an Intrusion Detection System (IDS) which can be deployed in the Advanced Metering Infrastructure (AMI) communication network. This module should be able to accurately determine the individual commands (but not their content) sent between AMI devices even when these commands are sent over an encrypted channel or in a protocol that the IDS cannot parse. The results are validated experimentally using two different datasets containing realistic traffic captured from two different AMI testbeds. The network traces for the encrypted protocol were collected from the Alliander testbed described in CRISALIS Deliverable 3.1 [2]. A secondary set of network traces for a different protocol were collected using a local testbed at Chalmers University of Technology.

As previously mentioned, the main goal of this module is the identification of exchanged AMI device commands, without access to clear text payload. More specifically, it harnesses a set of features that reflect the characteristics of AMI communication which are later used for classifying through supervised learning.

2.9.2 Main Results

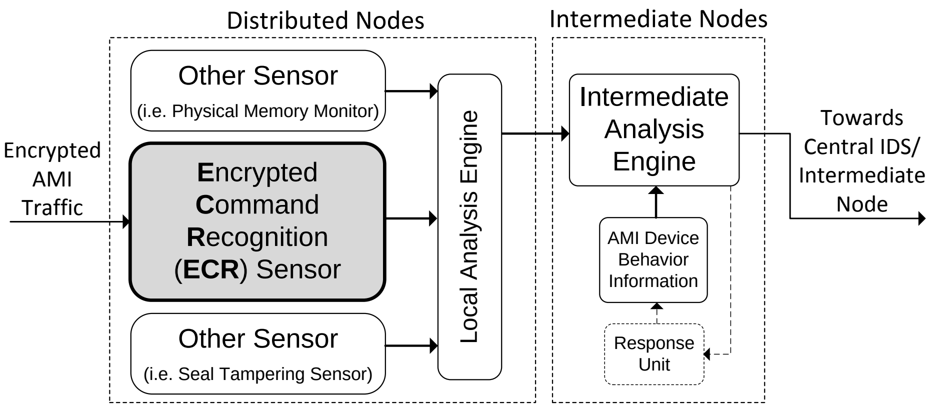
Distributed Nodes
Intermediate Nodes
Encrypted AMI Traffic
Other Sensor (i.e. Physical Memory Monitor)
Encrypted Command Recognition (ECR) Sensor
Other Sensor (i.e. Seal Tampering Sensor)
Local Analysis Engine
Intermediate Analysis Engine
AMI Device Behavior Information
Response Unit
Towards Central IDS/ Intermediate Node

Figure 2.2: The *ECR* sensor in an AMI IDS

Our results show that features extracted from the AMI communication which encapsulate important traffic characteristics can be successfully used to classify different types of commands that are run in an AMI environment. We have focused on two protocols, the *M-Bus protocol* [8] and the *DLMS/COSEM protocol* [4], with the first being relatively simple in its construction while the latter being more complex. Both of the protocols used TCP/IP at a lower communication layer.

We investigated a number of parameters typical for AMI commands, among them the more sensitive function "update firmware" (that was detailed in the requirements in deliverable 2.2 as being important to monitor).

One of the features that performs very well for both environments (testbeds and protocols) takes into consideration the number of bytes exchanged between the client and the smart meter and the number of interactions between those two during the session, as this reflects the way AMI commands are formed and the AMI protocol interactions.

2.9.3 Practical Conclusions / Lessons Learned

We have taken the first steps towards an encrypted command recognition component, as part of an Intrusion Detection System for the Advanced Metering Infrastructure (AMI). It uses statistical information and a classification method to identify AMI commands, even when they are sent over an encrypted channel or embedded in a protocol that is hard to parse.

The extracted command information can then be used as input to specification-based and anomaly-based intrusion detection systems. This is accomplished without decrypting confidential communication values, but at the same time enabling traffic monitoring.

2.9.4 Recommendations for Improvement

Creating a universal traffic analysis tool for all possible AMI deployments is almost impossible, due to the different types of protocols and network topologies that exist. We expect that the features identified for the two AMI protocols covered in this study and the analysis of their performance in identifying the different AMI commands will give useful insight on the internal characteristics of these types of protocols. We expect that our study will facilitate the analysis of other proprietary and/or encrypted AMI protocols as a step to build an IDS to protect this critical infrastructure.

Organisational Aspects

In todays deployment of AMI, all information is encrypted in the same manner. As there is an inherent balance between keeping values confidential and monitoring the network, it would be preferable if this was also accounted for in the standardization committees when designing the next generation AMI protocol. With todays protocols, it is challenging to find attacks even for the network owner; something an attacker can take advantage of in the future.

2.10 The Influence of Smart Grid Data Characteristics on Privacy

In a range of domains, it is important to consider the privacy implications of collections of large dataset. This is also true for the advanced metering infrastructure (AMI) as part of the smart grid. In the context of WP 6, we study how characteristics of datasets collected in the AMI influence the effectiveness of privacy-enhancing technologies.

2.10.1 Detailed Description

Preserving the privacy of the customers in the AMI is important as the lack of it has a direct effect on the individuals. Data gathered in the AMI in particular can be used to infer information about the living patterns of the residents. We investigate how characteristics of collected AMI datasets can influence the efficacy of previously proposed anonymization techniques for the AMI.

In particular, we study the case of two types of AMI data, one used mainly for billing and which should be safe with regard to privacy concerns (called Low-Frequency (LF) data [5]) and one that could be used for grid operations but which is privacy sensitive (called High-Frequency (HF) data [5]). The former is stored using the identity of the smart meter that generated them while the latter is stored with help of pseudonyms to enhance the privacy of the data.

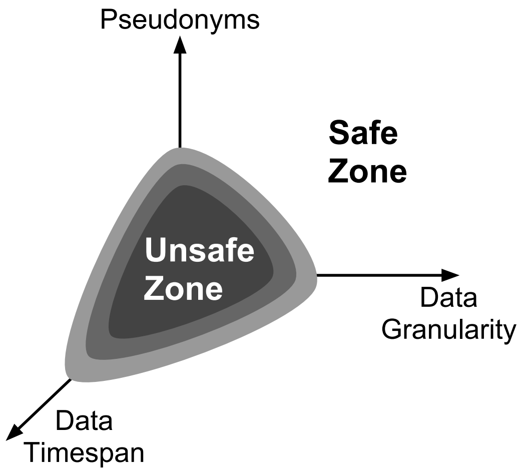

Figure 2.3: Characteristics of AMI datasets

Figure 2.3 [10] presents the three characteristics that we focus on: the usage of *pseudonyms* in the process of reporting/storing data for the same smart meter, the *timespan* of data stored by the utility provider under the same pseudonym for the same smart meter and the *granularity* of reported/stored data.

We investigate these three characteristics in two de-anonymization scenarios, based on two separate adversary models. In the first scenario we study how the granularity and the timespan of the data stored are influencing the adversary's capability of linking together two datasets produced by the same smart meters. In the second scenario we investigate an adversary whose goal is to use consumption features extracted from the datasets containing HF data in order to link together the different pseudonyms used for the same smart meter.

To see how well the adversarial strategy algorithm works in a real setting we use a dataset consisting of smart meter readings from a large number of consumers in a medium-sized city. The original data have hourly smart meter index readings for a period of seven non-contiguous months. The data originates from a range of smart meters serving very small consumers (summer cottages) to large consumers (industrial customers). The data have gone through a anonymization process, to make sure it is not possible to physically identify any customers in the set. The dataset contained $19,334$ unique smart meters with $99,355,998$ hourly energy consumption readings.

2.10.2 Main Results

Our experiments are validated by a large dataset, which means that we are able to revisit previously reported results in literature to see if they also hold for larger datasets. One of our results shows that varying the granularity under which data are reported can drastically reduce the fraction of identified smart meters. For example, reporting the index at a 10 kWh scale (that means without the last kWh digit) can reduce in this case the percentage of identified smart meters at under 10% for one period and under 30% for all periods. This result justifies a reporting scheme in which electrical energy consumed is rounded to the next 10 kWh value, before it is reported and billed, instead of being reported with 1 kWh accuracy. Such a simple change provides a good and cheap privacy solution for the other 70% of the customers, in the case that everyone opts for such a reporting scheme.

In the second experiment, we also observed that an increase in the number of smart meters contained in the dataset translates into a decrease in the ratio of successfully identified households. Due to the similarities in customers' energy consumption patterns it is harder to exactly match the pseudonyms under which data is stored for the same smart meter for different periods of time, especially for datasets containing a large number of customers. This is further elaborated in Deliverable D6.4 [12]. Also, as an interesting side effect, we notice that the effectiveness of the de-pseudonymization process is somewhat dependent on the season the data is collected.

2.10.3 Practical Conclusions / Lessons Learned

It is almost unquestionable that the smart grid will produce even more data, which includes the already collected consumption readings with other readings regarding the well-being of the electrical grid. Harnessing and processing these large quantities of data will make the electrical grid more resilient to faults, provide a better balance between the production and the consumption, but as we saw, these datasets also raise privacy concerns. For that reason, it is important to understand the limits on current privacy-enhancing technologies with the goal to improve them. We take into account two main properties of smart metering data: the granularity of the data reported and its timespan. We argue that these two, together with the number of pseudonyms used in the reporting process play a significant role in a three-way balance towards obtaining better customer privacy.

Similarly, in our second experiment we contribute to the study of the de-pseudonymization problem and we show how the number of households and the season influence the de-pseudonymization process. We show that the absolute number of re-identified households is not proportional with the size of the dataset as the percentage of re-identified households decreases as the size of the dataset increases.

2.10.4 Recommendations for Improvement

These results advocate for storing datasets containing values collected from a large number of anonymized households, as the de-pseudonymization process might benefit from smaller sets. Although an adversary might gain information from many anonymized households, re-identifying them will be harder if they are part of a larger dataset.

Our methodology and results can be used by electricity companies to better understand the properties of their smart metering datasets and the conditions under which such datasets can be released to third parties.

Organisational Aspects

Measuring systems by collecting datasets in any domain is important to gain knowledge of how the system is functioning and increasing its effectiveness. In the AMI, this is particularly true for research purposes, ensuring network stability, understanding trends of consumption, decision support to understand where renewables should be deployed, as well as a historical resource for future generations. From this perspective it is important to understand any potential privacy loss through the collection of such datasets and be able to increase the efficiency of privacy-preserving technologies to the domain.

2.11 Data Validation in Advanced Metering Infrastructures

The following work took place as part of WP 6, in task 6.3 focused on large-scale distributed approaches. It complements other work in the task in that it is a methodology to efficiently validate any type of data collected from within the AMI before it is further processed.

2.11.1 Detailed Description

The current generation of smart meters is able to report a number of different types of data, e.g. energy consumption, alarms, power quality measurements. However, due to a number of different reasons, these events need to be processed and validated before use (e.g. for attack detection or billing). We investigate a methodology for highly scalable validation of such events. The experiments run to evaluate the proposed data validation architecture have been conducted to study the scalability of the proposed approach. That is, to present evidence about the online high-throughput and low-latency processing enabled by distribute and parallel data streaming validation analysis. Different experiments have been conducted to assess the scalability of the proposed data validation architecture. These experiments cover the:

- **Processing throughput.** Focusing on the number of tuples (i.e., energy consumption readings) that can be consumed and validated per time-unit. The evaluation presents results in terms of tuples/second and overall number of smart meters that can be analyzed by a given server.
- **Processing Latency**. Focusing on the latency introduced to validate each individual tuple (or group of tuples). The latency is expressed in milliseconds.

All the experiments take into consideration different setups. One of the key parameters in such setups is the *batching size*, that is, the number of tuples that are delivered and validated together. As presented in the evaluation, the batching size is a key parameter in achieving high processing throughput while providing low processing latency.

At the same time, the different experiments, conducted for different validation rules, also provide evidence of the expressiveness of the data streaming processing paradigm, which can be leveraged to define a wide range of complex validation rules.

The experiments used data extracted from a real-world AMI, composed of 300,000 smart meters that cover a metropolitan area of 450 km2 with roughly 600,000 inhabitants. The dataset covers 13 months (From May 2012 to June 2013) of hourly consumption readings from a subset of 50 meters. The validation rules are implemented on top of

Storm, version 0.9.1. The evaluation has been conducted with an Intel-based workstation with two sockets of 8-core Xeon E5-2650 processors and 64 GB DDR3 memory.

2.11.2 Main Results

The evaluation gives evidence of very promising results for the proposed data validation architecture. For different batching sizes taken into account (of 100, 200, 300 and 400 meter readings, also referred to as tuples in the following), we have observed a throughput ranging from 2,000 tuples/second (batch size of 100 tuples) up to 7,000 tuples/second (batch size of 400 tuples). At the same time, we also observed low processing latencies, which do not exceed 56 milliseconds (and can be as low as 0.2 milliseconds depending on the validation rules deployed in the system). Such latency is very small and, subsequently, allows for validated data to be later used in different applications imposing a bound on the validation latency (such as demand-response or load prediction schemes).

2.11.3 Practical Conclusions / Lessons Learned

Based on the evaluation, we can conclude that the proposed data validation architecture is a clear way to address the challenges coming from the distributed nature of data validation in the context of smart grids and advanced metering infrastructures. For the observed throughput (up to 7,000 tuples/second) and a setup where smart meters are instructed to measure and report energy consumption readings every hour, the observed processing capacity would enable the validation architecture (deployed on a small server with 16 cores) to validate the readings produced by approximately 25 million meters for the validation rules we designed and developed. Hence, the proposed architecture would allow energy utilities to leverage and deploy data validation schemes with negligible deployment costs, even distributed within the AMI itself.

2.11.4 Recommendations for Improvement

Software / Component View

- A possible way to improve and automate the proposed data validation infrastructure is to ease the composition of validation rules via automatic translation of data streaming queries. It is common for Stream Processing Engines to rely on graphical interfaces to compose queries or to use widely-adopted languages such as SQL with adaptations to the streaming semantics. In this sense, specialized tools (with building blocks such as data streaming operators) tailored to Smart Grids and Advanced Metering Systems could be explored.

2.12 Forensic Agent FERRET

This set of experiments took place as part of Workpackage 7. We evaluated the forensic investigation framework FERRET in both operational technology (OT) as well as standard information technology (IT) environments. In particular we employed the tool in three real-world investigations:

1. OT case 1: malware infection in medical diagnostic system
2. OT case 2: anomaly in wind power installation
3. IT case: APT investigation

2.12.1 Detailed Description

A detailed description of the individual test cases can be found in deliverable D7.3. OT case 1 concerned a malware infection that was reported by a customer in an air-gaped network of medical diagnostic devices deployed at the customer side. OT case 2 was a malfunctioning off-shore wind turbine. The IT case was an APT investigation into manual hacking attacks against several standard IT assets (desktops and servers).

2.12.2 Main Results

As detailed in deliverable D7.3 all three test cases were concluded successfully. FERRET was able to collect and analyse the data that was relevant for the investigations. In particular the ability to deploy the forensic collector both in connected as well as in air-gaped networking scenarios and the ability to expand the collection set using a custom configuration was crucial.

2.12.3 Practical Conclusions / Lessons Learned

We drew several lessons from the practical deployments of the FERRET system.

- Need for flexibility: the variety of OT environments that are potential deployment scenarios for the FERRET system requires both a standard baseline with respect to the data collection, backend analysis and user interface representation capabilities and the ability to adjust or tweak these base capabilities to reflect the special circumstances of each environment. This mainly concerns additional data sources that needs to be integrated.

- Need for limited data: low bandwith situations frequently encountered in OT environments (eg. service connections via mobile data networks) put limits on the amount of data that can be collected off a system. It is of importance to select the most important data for a forensic investigation, to have strict caps on the total amount of data collected in place and to have the flexibility to tune the data collection to particular circumstances via policy files.
- Need for live response: investigating systems while they are active serves two important functions. It provides a natural filter to reduce the amount of data that needs to be collected and analysed based on the assumption that active content (such as running processes) is the most relevant for the investigation of security incidents. It also allows operators to leave machines running and thereby reduces the threshold of when an investigation can be performed. If the investigation is non-disruptive, even low-suspicious situation can be investigated.

2.12.4 Recommendations for Improvement

In a short- and medium-term outlook we see the most benefit in improvements of the FERRET framework in the following areas:

- Broaden support for different OT environments. Implement support for Linux and Solaris based systems. Also support embedded systems.
- Improve analysis capabilities. Support analyst with more automatically generated insight.

Apart from improvements to the FERRET framework, there are also deficiencies in the typical OT environments that should be addressed as part of forensic readiness.

Software / Component View

1. **Better ability to control impact of a forensic data collection on the continuous operation/Forensic data collection as part of acceptance tests:** The forensic data collection needs to run with the highest system permissions in order to be able to have read access to all required data. It has an non-negligible impact on the processing, I/O and potentially network bandwith (when transmitting the collected results). While on modern operating systems it is possible to put certain constraints on the resource consumption (eg. via the nice or ionice mechanism), these mechanisms are limited and lacking on older operating systems. The

impact of such a data collection on the continuous operation of an installation should ideally be validated as part of the acceptance testing. Alternatively there should be a way for the control system software to signal the forensic collection process its needs and thereby enable the forensic collection to limit its impact.

2. **Standardized format and access to forensic data/Improved availability and protection of forensic data:**
   While there are standard security logging mechanisms on modern operating systems, the detail of the information that is recorded via these mechanisms is usually insufficient for a forensic investigation. Operating systems employed in OT environments should provided standardized mechanisms for the recording of more system activities such as file system activities, should allow to better control the detail that is recorded for each activity, should better protect the recorded information against unauthoried manipulations and should provide standardized mechanisms to access the recorded information.

3. **Integration of rudimentary filtering and analysis capabilities in logging mechanism:**
   Improved forensic logging would result in an increase of the amount of logging data stored on a system. To counterbalance this, the logging mechanism should be augmented with rudimentary analysis capabilities to provide better selection of the data that is recorded and stored. This type of “smart” logging would also enable to provide scoring and classification of the logged data according to the relevance for forensic investigations.

Technical Environment / Architectural View

1. **Embedding of security monitoring and SIEM (Security Information and Event Management) capabilities into ICS systems**
   There are currently capabilities lacking for the active detection of security incidents that are taylored to industrial control systems or OT environments in general. A timely reaction to such incidents is therefore not guaranteed. There should therefore be security monitoring capabilities embedded into these environments that fulfill this role.

2. **Integration of SIEM with forensic investigation capabilities. Automation of response process**
   A tight coupling of security monitoring and forensic investigation capabilities would

allow a higher degree of automation of these tasks resulting in a more rapid response to security breaches.

Organisational Aspects

1. **Integration of security monitoring and operational monitoring**
   Security monitoring and first response capabilities should be integrated with the operational monitoring to enable operational staff to detect and react to security breaches.

2. **Forensic investigations as a service model**
   While first response capabilities can be provided by operational staff, more demanding forensic investigation capabilities are usually out of reach for most industrial control system operators. It is therefore important to enable a service model where forensic investigation capabilities can be provided by third parties. This introduces additional requirements such as anonymization of the collected data and the filtering of proprietary information.

2.13 Forensic Drift

This set of experiments took place as part of Workpackage 7. We used the forensic investigation framework FERRET developed as part of this work package to measure the "forensic drift" in various industrial control systems in use in the ENEL test lab. We define the "forensic drift" as the set of persistent modifications to objects on a host as part of the normal operation of the system.

In particular the drift on the following industrial control system components was evaluated:

1. DigiTool: a Windows PC running the DigiTool industrial control system by ABB

2. Simatic: a Windows PC running the Simatic industrial control system by Siemens

3. DeltaV: a Windows PC running the DeltaV industrial control system by Emerson

2.13.1 Detailed Description

While there are various ephemeral changes during the operation of a computer system such as data flows accross the network in this experiment we are only interested in modifications that are permanent and can be detected in a forensic investigation. Furthermore, not all persistent changes are relevant for a forensic investigation in a security context. We are therefore concentrating on those changes that can be measured by the FERRET tool. Lastly, since the FERRET tool is largely based on the collection of meta-information it has limitations in the available resolution. While we can for example detect that a file was modified, we can not determine what modification was made or even the extend of the change.

We initially performed a compehensive test that included all data sources that are collected and analysed by the FERRET system to better understand the system changes during normal operation. We then exluded those data sources that were irrelevant. On the one hand these were volatile information such as process IDs, network connections or the amount of free memory. On the other hand we excluded data that was static in the context of these ICS systems for obvious reasons such as the list of visited URLs (the ICS systems did not have a direct connection to the internet).

We therefore limited the experiment to the following objects:

1. files - for each file we compared the following meta-information available in the Windows NTFS file system: name, path, size, access time, creation time, change time and entry modification time
2. processes - for each process we compared the path of the executable file and the process arguments
3. object handles - for each process we compared the list of open object handles such as mutexes, RPC ports, open registry keys and open files

We ran the FERRET collection agent three times on each system. First to establish a baseline, a second time after one week of normal operation and a third time after four weeks of normal operation.

2.13.2 Main Results

File System

We compared the list of files and their meta-information after one and again after four weeks against the baseline using different filters.

- filter1 - exclude entry modification timestamp from comparison
- filter2 - exclude entry modification and access timestamps from comparison

- filter3 - exclude all meta-information except file path from comparison

Digitool: 58493 files

test	filter	unchanged	only baseline	new and modified
one week	none	7.72%	46.12%	46.15%
one week	filter1	79.47%	10.24%	10.29%
one week	filter2	99.49%	0.23%	0.28%
one week	filter3	99.86%	0.04%	0.09%
four weeks	none	7.72%	46.12%	46.16%
four weeks	filter1	79.30%	10.32%	10.39%
four weeks	filter2	99.31%	0.31%	0.38%
four weeks	filter3	99.84%	0.04%	0.12%

Table 2.1: Digitool

DeltaV: 65544 files.

test	filter	unchanged	only baseline	new and modified
one week	none	20.68%	39.72%	39.60%
one week	filter1	82.10%	9.05%	8.85%
one week	filter2	98.86%	0.67%	0.47%
one week	filter3	99.43%	0.39%	0.18%
four weeks	none	20.68%	39.73%	39.59%
four weeks	filter1	82.09%	9.06%	8.85%
four weeks	filter2	98.81%	0.71%	0.48%
four weeks	filter3	99.35%	0.44%	0.21%

Table 2.2: DeltaV

Simatic: 46739 files

test	filter	unchanged	only baseline	new and modified
one week	none	4.31%	47.87%	47.82%
one week	filter1	78.76%	10.66%	10.58%
one week	filter2	97.29%	1.40%	1.31%
one week	filter3	97.89%	1.10%	1.01%
four weeks	none	4.29%	47.61%	48.10%
four weeks	filter1	78.02%	10.58%	11.40%
four weeks	filter2	96.25%	1.42%	2.33%
four weeks	filter3	96.85%	1.12%	2.03%

Table 2.3: Simatic

Looking in detail at the changes we found that the vast majority were due to operating system mechanisms. In particular, the 948 files (2.03%) that were found new after four weeks of operation in the Simatic data set above belonged to the following classes.

A similar situation is found with respect to the files in the baseline that were no longer present after four weeks of operation. This result was not specific to the Simatic data set but also applied to the Digitool and DeltaV data sets.

number	class
691	system restore points
211	anti virus signature updates
22	prefetch files
17	Windows system information (pchealth)
5	files in temp folder
2	other files

Table 2.4: Newly created files

Processes

We compared the list of running processes. This included the path of the corresponding executable file and their arguments. We found very little change overall. On all systems two new processes were running related to the distributed transaction coordinator msdtc service:

```
C:\WINDOWS\system32\dllhost.exe /Processid:{02D4B3F1-FD88-11D1-960D-00805
FC79235}
C:\WINDOWS\system32\msdtc.exe
```

On the digitool system the process `C:\ICUBESYS\BIN\GMSRCLN.EXE` was found running without an argument compared to the baseline were it was running with the argument `C:\ICUBESYS\BIN\GMSRCLN.EXE IDROLAB_MAIN.GRC`

Object handles

For each running process we compared the list of open object handles such as mutexes, RPC ports, open registry keys or files. On a typical system, the list of object handles as for example reported by the Microsoft/Sysinternals `handles.exe` tool is on the order of 10.000 entries. This prohibits a manual review of these handles. The question is, if a filtering using a baseline can result in a sufficient reduction in the number of entries to enable a manual evaluation.

Object handles are volatile memory objects. They are created and dstroyed during program execution when certain API functions (such as CreateFile/CloseFile, CreateMutex/ReleaseMutex etc.) are called. Some of them such as mutexes or semaphores often get assigned random names that are generated by the operating system. In order to better filter those random objects, a filter (filter1) was used to normalize object handle names by removing strings constisting of consecutive digits.

Digitool: 15693 handles

test	filter	unchanged	only baseline	new and modified
one week	none	95.71%	2.07%	2.22%
one week	filter1	99.53%	0.22%	0.24%
four weeks	none	95.51%	2.13%	2.36%
four weeks	filter1	99.41%	0.26%	0.33%

Table 2.5: Digitool

DeltaV: 31228 handles

test	filter	unchanged	only baseline	new and modified
one week	none	95.62%	2.19%	2.19%
one week	filter1	99.80%	0.07%	0.13%
four weeks	none	94.24%	2.32%	3.43%
four weeks	filter1	99.76%	0.09%	0.15%

Table 2.6: DeltaV

Simatic: 10784 handles

test	filter	unchanged	only baseline	new and modified
one week	none	91.80%	3.91%	4.29%
one week	filter1	98.56%	0.65%	0.79%
four weeks	none	91.80%	3.84%	4.36%
four weeks	filter1	98.60%	0.62%	0.78%

Table 2.7: Simatic

2.13.3 Practical Conclusions / Lessons Learned

We drew several lessons from the practical deployments of the FERRET system.

- While ICS environments are not used interactively and are therefore considered comparatively static there is still considerable forensic drift due to the normal operation of the systems.

- The majority of the forensic drift can be modeled and subsequently filtered out with some simple static filtering rules.
- The use of baselines together with static filters allows the reduction of the forensic data to a point where a manual review of the remaining data is feasible. While it is for example unfeasible to manually review a list of 10.000 object handles for suspcious entries, bringing this list down to 50 entries allows a manual processing in a reasonable amount of time. This opens up new avenues and approaches for forensic investigations.

2.13.4 Recommendations for Improvement

The results suggest that the use of a forensic baseline is a highly effective mechanism for the filtering of results in a forensic investigation. The "forensic drift", that is the deviation from a baseline, is generally small due to the limited use and consequent variability of these systems and can be further reduced through simple static filters.

A practical recommendation would be to generate such baselines of all components of an ICS system during deployment.

Besides aiding on-demand forensic investigations as part of an incident response, such a baseline could enable regular automated forensic checks to detect anomalous malicious behaviour as part of a continuous monitoring.

2.14 Host-Based Protection Systems

This set of experiments was designed to test two protection techniques developed as part of Workpackage 7. Both techniques were designed as standalone thin hypervisors that run below the traditional operating system in order to provide a system-wide protection against sophisticated attacks that can subvert and tamper with the OS itself.

The two proposed solutions are:

1. **AccessMiner** – An anomaly-based system that models the interaction between programs and OS resources (filesystem, network, registry, ...) to detect violations of the "normal" behavior.
2. **Graffiti** – A rule-based system to detect any kind of heap-spraying attacks performed in the Operating System.

Both systems were tested to verify the overhead (very important for hypervisor-based solutions), the detection capabilities, and the false positives generated when running benign software.

2.14.1 Detailed Description

We run a very extensive set of tests to verify the effectiveness of the two techniques. A detailed description of the individual test cases can be found in deliverable D7.4.

The performed tests included:

1. *Accessminer-1*: in which we tested AccessMiner ability to detect anomalous behaviors when running 6,000 unique malware samples on MS. Windows.
2. *Accessminer-2*: in which we tested AccessMiner false positive rate on 114.5 GB of execution traces collected from 10 different real-world computers - running 362,600 processes and 242 distinct applications for a period of a week.
3. *Accessminer-3*: in which we compared the overhead introduced by AccessMiner on different workloads by using four of the Pass-Mark performance tests: memory operations read and write, and sequential disk read and write operations.
4. *Accessminer-4*: in which we conducted a fine-grained experiment to measure the time needed to perform a context switch between the OS and AccessMiner thin hypervisor.
5. *Graffiti-1*: in which we used the stress suite to simulate a program that intensively allocate memory to measure the overhead introduced by the tracking component of Graffiti.
6. *Graffiti-2*: in which we measured Graffiti overhead on Microsoft Windows when using Internet Explorer 8 to visit GMail, watch videos on YouTube, browse Facebook, and check hundreds of pictures on 9gag.
7. *Graffiti-3*: in which we measured the Graffiti overhead on Linux when using Acrobat Reader to open 100 benign PDF files including conference papers, books, Ph.D. dissertations, and large manuals.
8. *Graffiti-4*: in which we tested the ability of Graffiti to detect different kinds of heap spraying attacks. For this test we selected a number of real exploits against Adobe Reader, Flash Player, Internet Explorer, and Firefox.
9. *Graffiti-5*: in which we used Graffiti to analyze three datasets: a set of 1000 malicious PDF documents, a set containing 1000 benign web pages, and a set of 1000 benign PDF documents.

All experiments were performed in a lab on Windows XP, Windows 7, and Debian Wheezy 32 bit operating systems.

2.14.2 Main Results

The results of all experiments were very promising.

The experiments on AccessMiner show that the system, after a proper training, could be deployed in a real environment with only a limited impact on the performance of the system. Overall, AccessMiner provided a high level of protection (around 90% detection rate with zero false positives) with an overhead similar to the one that can be experienced in a state of the art virtual machine environment.

Graffiti was able to offer a complete, OS independent protection against all spraying techniques known to date. All attacks were successfully detected and only few false alarms were raised when browsing complex web applications whose Javascript contained a large number of potential code pointers. Moreover, Graffiti's overhead can be tuned according to the application behavior and to the desired level of protection.

See Deliverable D7.4 for more details about the individual results.

2.14.3 Practical Conclusions / Lessons Learned

All our experiments proved that the host-based protection system designed as part of Workpackage 7 are effective in a large set of environments, ranging from lab machines, to servers, to user laptops. The more stable environment of a critical infrastructure represents therefore a best-case deployment scenario. However, critical infrastructures introduce two relevant challenges:

- The computers hardware is often outdated and not compatible with the most recent technologies required to support virtualization. Even though the main manufacturers had introduced this support several years ago, computers that are 10 years old or more may not be able to run the software presented in our experiments.

- Performances are very important in a critical infrastructure, especially in old hardware. While is indisputable that hypervisor-based solutions offer a better protection against sophisticated attacks, in certain conditions the overhead they introduce may not be acceptable in an operational environment.

2.14.4 Recommendations for Improvement

OS-based protection are a fundamental component to detect attacks and protect critical systems against malware and targeted attacks. In this section we summarize the recommendations to improve the tools themselves and the critical infrastructures as a whole.

Software / Component View

1. **Support to create anomaly-based models on the field**
AccessMiner relies on the knowledge of a model of the interaction between programs and the operating system. Such model was extracted from a training set collected on a number of different "clean" computers. However, the model would change over time and between different operational environments. Therefore, it would be beneficial to have an automated tool to build a model for a target system. This tools should be usable by the system administrator of a critical infrastructure without requiring a detailed knowledge of the AccessMiner system itself.

2. **Implementation of a robust, self-contained Graffiti system**
At the current stage, Graffiti is a prototype software that requires an expert to be operated. Some of the components needs to be operated separately and a considerable manual expertize is required to install the software. Therefore, some engineering effort should be dedicated to develop a more refined tool and an installer for it. This effort has already started and it is now under active development.

Technical Environment / Architectural View

1. **Keep hardware up-to-date with current technologies**
Both Intel and AMD are very active in introducing new security-related technologies in their processor. This innovation plays a very important role to support modern OS defense mechanisms as well as new protection tools. In particular, the support for hardware-assisted virtualization is nowadays required by many software.

3 Recommendations for future critical infrastructures

In this chapter, we present recommendations for the improvement of the security of future critical infrastructure systems. Therefore, we summarize the most important recommendations from the experiments and add further recommendations from the project experience as well as from our daily work. We especially recommend the following (only partially automation networks specific) measures:

Securing the system:

- A closed and secure engineering chain is essential for secure plant operation. This includes especially secure firmware / software updates, based on software signatures or whitelists, and accurate local device and configuration inventories.
- We encourage the use of well-tested standard protocols (instead of proprietary ones) if possible.
- Security testing of devices has to be improved. Generally recommended test methods such as fuzzing and code analysis shall become a standard feature of security testing at product vendors.
- Patching traditional components often results in a service discontinuation of the system which is usually not acceptable during plant operation in many industrial sectors. Therefore, we recommend to develop new components for these sectors in such a way that patching without service discontinuation is possible.
- Firmware / software updates for security reasons have to be considered in safety approval procedures. Therefore, regulatory procedures have to be adapted in order to allow especially security patching.
- A higher security level (see e.g. IEC 62443) for more security critical (e.g. more exposed) systems is recommended.

Detecting intrusion:

- Security surveillance solutions (SIEM, IDS) shall be integrated within automation networks. Centralized logging of security relevant events may support this.
- For all alarming systems and tools, a very low false positive rate is essential for usage in operational environments.
- Reaction on security breaches should include graceful degradation; automated reaction must be evaluated carefully on a case by case basis.
- System discovery tools shall be used for network surveillance (even in the case of accurate inventories), e.g. for discovery of legacy devices, network topology updates in dynamic networks, and detection of potential attacker devices.
- Need for encryption (especially of not confidential traffic) has to be balanced with requirements of security monitoring.
- Health monitoring shall be available not only for safety critical, but also for security relevant devices.

Analyzing successful intrusion:

- Quality of stored log data used for forensics (which may be used also for intrusion detection purposes), and easeness of access shall be improved, e.g.
  - Standardized format
  - Standardized access
  - Increased volume
  - Increased storage time
  - Increased configuration options to allow high flexibility
  - Protection against manipulation (change, deletion)
- Analysis of forensic data shall be enhanced by workflow automation, e.g.
  - Collection of known vulnerabilities from available databases
  - Comparison of vulnerabilities with device inventories and system discovery results
  - Preparation / preprocessing of forensic data, including automated forensic data collection from system components and the system's SIEM/IDS.

- Collection of forensic data shall be possible during system operation without service impact.
- Collection of forensic data as part of a system's acceptance test and later at regular intervals allows definition of a "forensic baseline profile" which can be used for forensics and even for intrusion monitoring.

From our view, the most important returns of experiences are:

- A lot of security technologies and best practices available in classical IT are not yet (widely) used in automation networks. Some examples for those security measures are malware scans, address space layout randomization, and data execution prevention. The usage of measures known from classical IT should be stronger considered in automation networks, too.
- Also new technologies and solutions are needed, in particular for anomaly detection in OT environments, fingerprinting and forensic analysis of industrial devices, as well as not invasive real-time network and device monitoring. Ongoing research on these topics shall be continued.
- Security processes in the lifecycle of industrial components should be further improved. Important topics are:
  - Security is becoming more and more a key feature of successful plant operation. Therefore, security has be be considered from the beginning in the development process.
  - Security needs to be user-fiendly. Security processes have to be automated as much as possible.
  - Security awareness of vendors, integrators, and operators is essential for secure ICSs.
  - Measures for close monitoring not only of safety, but also of the security status during operation is strongly recommended to allow quick detection of ongoing attacks, and in-time deployment of appropriate countermeasures.
  - Remediation plans should be defined, implemented, and tested with security issues and measures in mind. These plans should particularly contain options for forensic analysis of IT and OT devices after security incidents.
- Finally, we would like to underline the importance of sharing information among the different trusted stakeholders regarding new attacks and incidents, vulnerabilities, best practices, countermeasures and security technologies available, etc. The

collaboration, in particular considering the even more connected world of smart grid and smart cities, is essential for security improvements and effective security management. The current global efforts in sharing threat intelligence information show that exploiting synergies can provide big advantages.

The CRISALIS project confirms the high value-add given by the experiences in laboratories and testbeds where also invasive experiments on new tools and technologies can be performed without the issues, limitations, functional constrains and requirements of the real plants. In this context the availability of research institutes, software houses, tools developers, and universities, driven by the knowledge of end-users, can generate good contributions to the development of innovative, advanced and precise technological solutions responding to the market and to infrastructure owners' needs.

Bibliography

[1] M. Almgren, D. Balzarotti, J. Stijohann, and E. Zambon. CRISALIS D5.3 Report on automated vulnerability discovery techniques. Technical report, European Community's Seventh Framework Programme, 2014.

[2] V. Angeletti, E. Kooi, M. Caselli, T. Limmer, and C. Leita. CRISALIS D3.1 Setup detailed description. Technical report, European Community's Seventh Framework Programme, 2013.

[3] L. Auriemma. Advisories. `http://aluigi.altervista.org`, November 2011.

[4] DLMS User Association. DLMS/COSEM protocol `http://www.dlms.com/index2.php`.

[5] C. Efthymiou and G. Kalogridis. Smart grid privacy via anonymization of smart metering data. In *Smart Grid Communications (SmartGridComm), 2010 First IEEE International Conference on*, pages 238 –243, Oct. 2010.

[6] N. Falliere, L. Murchu, and E. Chien. W32.Stuxnet Dossier. Technical report, Symantec, September 2010.

[7] V. Gulisano, M. Almgren, and M. Papatriantafilou. Metis: a two-tier intrusion detection system for advanced metering infrastructures. In *10th International Conference on Security and Privacy in Communication Networks (SecureComm)*, 2014.

[8] M-Bus Usergroup. M-Bus protocol `http://www.m-bus.com/`.

[9] R. Sibson. Slink: an optimally efficient algorithm for the single-link cluster method. *The Computer Journal*, 16(1):30–34, 1973.

[10] V. Tudor, M. Almgren, and M. Papatriantafilou. Analysis of the impact of data granularity on privacy for the smart grid. In *Proceedings of the 12th ACM Workshop on Workshop on Privacy in the Electronic Society*, WPES '13, pages 61–70, New York, NY, USA, 2013. ACM.

[11] V. Tudor, M. Almgren, and M. Papatriantafilou. Harnessing the unknown in advanced meter infrastructure traffic. In *the ACM Symposium on Applied Computing (SAC 2015)*, 2015.

[12] E. Zambon, M. Caselli, and M. Almgren. Deliverable D6.4: Network-Driven Analysis tools. `http://www.crisalis-project.eu`, August 2015.

Nomenclature

AMI	Advanced Metering Infrastructure
APT	Advanced Persistent Threat
CI	Critical infrastructure
COSEM	Companion Specification for Energy Metering
COTP	Connection oriented Transport Protocol
DLMS	Device Language Message Specification
DTMC	Discrete-time Markov chains
FN	False negative
FP	False positive
G-SLC	Grid-based Single Linkage Clustering
HF	High frequency
ICS	Industrial Control System
IDS	Intrusion Detection System
IP	Internet Protocol
IPS	Intrusion Prevention System
IT	Information Technology
ITOT	Information Technology - Operational Technology
JDK	Java Development Kit
LF	Low frequency
MMS	Manufacturing Message Specification

NAT	Network Address Translation
OLE	Object Linking and Embedding
OPC	OLE for Process Control
OT	Operational Technology
PAT	Port Address Translation
PLC	Programmable Logic Controller
PP	Proprietary protocol
S-IDS	Sequence-aware Intrusion Detection System
SIEM	Security Information and Event Management
TCP	Transport Control Protocol
TP	True positive
UPP	UDP proprietary protocol