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

D7.1 Preliminary report on forensic analysis for industrial systems

Contract No. FP7-SEC-285477-CRISALIS

Workpackage	WP7 - Vulnerability discovery
Author	Heiko Patzlaff
Version	1.0
Date of delivery	M12
Actual Date of Delivery	M12
Dissemination level	Public
Responsible	SIE

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:

Symantec Ltd.	Project coordinator	Ireland
Alliander		Netherlands
Chalmers University		Sweden
ENEL Ingegneria e Innovazione		Italy
EURECOM		France
Security Matters BV		Netherlands
Siemens AG		Germany
Universiteit Twente		Netherlands

Contact information:
Dr. Corrado Leita
2229 Route des Cretes
06560 Sophia Antipolis
France

e-mail: corrado_leita@symantec.com
Phone: +33 673 41 91 27

Contents

Abstract

The ability to analyze security breaches in industrial control systems (ICS) is vital for the fast remediation of compromises and thereby helps to improve the robustness of these systems in the face of attacks. Current forensic practice lacks specific tools and procedures for performing investigations in these environments. Furthermore, the specifics of industrial control systems in most cases prevent the application of existing standard forensic tools and approaches.

In this paper we discuss the challenges posed by industrial control systems for forensic investigations. We derive design criteria for a forensic framework for such systems and describe the preliminary work that has been performed for implementing such a framework.

1 Introduction

1.1 Motivation

A number of trends have resulted in a gradual convergence of the traditional information technology field (IT) and various other fields which are collectively referred to as Operational Technology (OT) ([1]). Operational Technology in the above sense refers to hardware and software directly responsible for the monitoring and controlling of physical devices. In a industrial settings these systems are usually referred to as Industrial Control Systems (ICS). Among those trends are the increasing digitalization of control systems, the growing use of standardized components and protocols, and the increasing connection of formerly separate subnets to the wider internet.

These trend have sparked interest in the security of ICS, resulting in a significant increase in reported vulnerabilites of and exploits for ICS devices in recent years, as can be seen in figure 1.1. Additionally, due to the convergence of conventional IT and OT, a host of new security challenges have come to light as evidenced by a growing number of security incidents related to ICS that are reported.

The attacks performed using the Stuxnet worm and other high-profile security incidents have demonstrated not only that ICS are vulnerable, but also that there currently is a lack of tools and procedures to react to these types of security breaches. Therefore, the ability to not only detect but also to quickly analyze a compromise is a fundamental capability in order to effectively react and remediate security incidents and to increase the robustness of ICS in the face of attacks. The discussion and design of such *forensic analysis* is the goal of this deliverable.

1.2 Layout of this document

The layout of this document is as follows. We begin by describing the scope of our forensic research and by discussing the current state of the art in ICS forensics. In chapter 3 we detail the challenges specific to ICS. To better understand the limitations of this project we briefly discuss the attack scenarios that we consider relevant in our study in chapter 4 and describe a model architecture which will act as a reference point for our work in chapter 5. The last part of this document is dedicated to the discussion

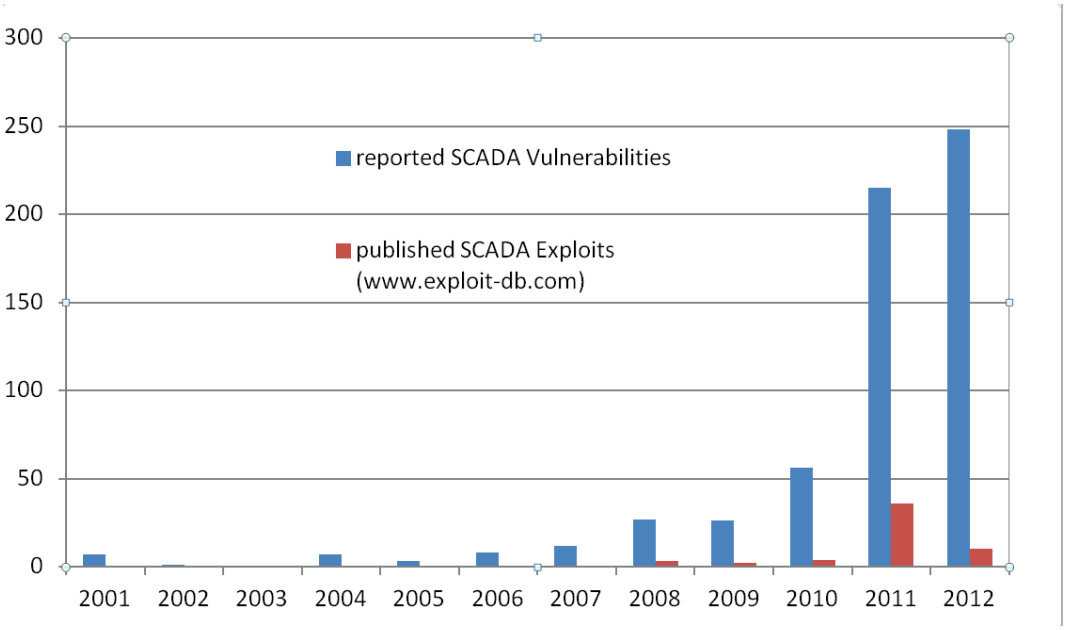

Figure 1.1: Development of the number of SCADA security vulnerabilities and exploits

of the current state of the main goal of this project, the development of a suite of tools for the acquisition and analysis of forensic data of industrial control systems.

1.3 Scope of the project

The objective of the *forensic analysis of industrial systems* subtask of WP7 is to research ways to perform forensic investigations of system compromises in ICS. The compromise scenarios that are most relevant in the context of Crisalis are malware infections, hacking incidents, and targeted attacks which are described in more detail in section 4.

In particular, we will address the following topics

1. Investigate requirements, challenges and opportunities for forensic investigations of system compromises that are specific to ICS. ICS pose specific challenges and requirements for forensic investigations, both in the acquisition and in the analysis of data. ICS are typically proprietary systems with undocumented internal protocols, data structures and interfaces. They are usually distributed, potentially over

wide areas, they are often not directly connected to the internet, they are resource constrained, and they need to run continuously.

2. Identify data sources in ICS that can be used in forensic investigations. Relevant data can be obtained for example from log files, databases, filesystem objects and timestamps which can be collected from different sub-components which make up an ICS. Of particular interest are the programmable logic controllers (PLCs) and the code and data running on them. ICS are unique because of the large amounts of process data that is generated. Whether it is possible to utilize this data for forensic purposes is one of the open questions this project intends to address.

3. Identify possible solutions that can be implemented in ICS which would aid a forensic investigation in case of an incident. Apart from the identification of existing mechanisms that can be used in forensic investigations, we also want to identify additional measures that can be implemented in ICS that would aid a forensic investigation. Examples of possible measures are the creation of forensic baselines on clean systems, the synchronization of all time sources or the implementation of logging of relevant system events.

4. Develop a methodology for the forensic acquisition and analysis of data as part of an incident response methodology in ICS. The forensic analysis of system compromises is just one step in a complete incidence response process. We need to identify practical incidence response procedures that honor the specific requirements of forensic investigations, in particular the preservation of traces and the order of volatility, while enabling a quick reaction to and resolution of incidents.

5. Develop tools and techniques for the extraction and the analysis of forensic data. The Stuxnet incident has demonstrated that we lack forensically sound tools to extract and analyze relevant data from ICS. We are going to develop tools and test them in industrial settings, ideally as part of realistic incidents.

6. Develop tools and techniques to support reverse-engineering and analysis of malware artifacts in the context of security incidents in control systems.

2 State of the art

Digital forensics is concerned with the recovery and investigation of digital evidence ([2]). It is used as an umbrella term to unite different fields in which forensic methods are applied. Not all of these fields are relevant to the investigation of compromises in industrial control systems.

The forensic investigation of ICS and SCADA systems is a comparatively new and undeveloped domain. Few contributions exist from academia. It has also so far spurred little interest in commercial forensic companies, ICS vendors or the end user community. Consequently, little material exist that is specific to IC systems and previous work has largely concentrated on the investigation of network traces (see for example [3]). However, many contributions from other fields can be applied and are relevant to the question of forensic investigations in industrial control systems.

In this section we provide a review of the current state of the forensic practice in those fields that have a bearing on the work done in WP7 of the Crisalis project. These are in particular the classical computer forensics of standard host systems and the forensic investigation of embedded systems.

2.1 Host forensics on standard PCs - Windows, MacOS, Linux

Computer forensics on standard PCs is a mature field and a plethora of documentation and tools exist both for the extraction and the analysis of host forensic data. Basic operating system subsystems and mechanisms relevant for the generation and preservation of traces are well understood and standard procedures have been developed for the secure acquisition of forensic data.

Two general books that cover these topics are [4] and [5]. Operating system specific material is e.g. covered in [6] for Windows, [7] for Mac OS and [8] for UNIX and Linux systems. Specialized topics such as filesystem forensics ([9]), Windows registry forensics ([10]) and malware forensics ([11]) are also well documented.

Several commercial toolkits exist for the analysis of host data of standard PCs such as EnCase ([12]), FTK ([13]) and X-Ways Forensics ([14]).

Computer forensics of standard operating systems has a high relevance for the forensic analysis of industrial control systems since many central components of IC systems such

as the DCS or the engineering workstation are usually based on standard operating systems.

2.1.1 Analysis of live systems and volatile data

Traditional computer forensics is concerned with the analysis of persistent host data. This approach, sometimes also called "dead forensics", extracts data from a powered down system.

A comparatively new field in the forensic analysis of host systems is concerned with the acquisition and analysis of volatile memory data. This is in particular relevant for the detection of system compromises since the live analysis provides a forensic analyst with a more complete picture of run-time system activities such as active processes and network connections. However, the acquisition of data from a live system poses several challenges. In particular, how to extract data without negatively impacting the system stability and run-time behavior and how to solve the challenge posed by rootkits that actively conceal objects or prevent access to them.

Most of the recent activity in this field has been centered around the open-source volatility framework ([15], see also [16]).

2.1.2 Agent based compromise detection

Traditional host computer forensics has several shortcomings when being applied to the analysis of system compromises in enterprise networks. The acquisition and analysis of the data is time consuming. The transfer of large amounts of forensic data such as hard-drive images around the globe for analysis in a forensic lab poses its own challenges. As a response, light-weight approaches have been developed that use agent based solutions on the host combined with backend processing and analysis capabilities.

Two general classes exist. The first uses a pre-installed and constantly running forensic agent that provides remote access to a live system by the forensic analysis environment. Typical examples of this approach are EncaseEnterprise ([17]), Google rapid response ([18]) and RSA ECAT ([19]).

The second class uses a data acquisition agent that is deployed on-demand and collects selective forensic data from a host. The size of the extracted data is usually small enough to be transferred electronically to a human analyst that uses backend processing and review software to analyze the collected data. A typical example for this approach is Mandiant Redline ([20]).

2.2 Embedded forensics

Embedded devices are utilized in different industries. Examples for embedded systems range from gaming consoles, mobile phones and home entertainment devices to network equipment, medical devices and train control systems. Of particular interest for the current work are programmable logic controllers (PLCs) used in industrial control systems.

The forensic investigation of such systems has to solve two main problems. The first is the question of how to extract data. Embedded systems usually don't employ detachable storage devices for size, cost and robustness reasons. Instead, on-board flash memory chips are often used. Commercial tools for mobile phone forensics such as [21] and [22] exist that perform data extraction on a logical level. However, in many devices external logical access to the memory content is restricted.

Several ways for accessing the flash content on a lower level have been documented. A good overview is provided in [23]. Breeuwsma ([24]) discusses the use of the JTAG (Joint Test Action Group) test port. The use of so-called flasher tools and the physical extraction (de-soldering) of the flash chips together with the extraction of the content with a suitable memory chip programmer are two other, less generic, approaches.

The second problem that needs to be solved is the logical reconstruction of objects from the extracted memory content. This is usually highly vendor specific both on the filesystem level and on the file object level. For USB and solid-state-disk media this has been discussed eg. in [25], [26], [27] and [28].

3 Challenges

Industrial control systems present some unique challenges for forensic investigations. We will discuss these challenges in the following paragraphs.

3.1 Proprietary components, interfaces, data formats and protocols

Many of the components used in ICS environments are vendor specific and proprietary. They contain vendor specific software including firmware and operating systems, use proprietary data formats, data structures and cryptographic primitives, implement proprietary interfaces and make use of proprietary or vendor augmented protocols. For example, while there exist about 150-200 SCADA protocols, only few of them are open and documented.

This has several consequences for forensic investigations.

3.1.1 Lack of documentation

A large amount of training material and documentation is available that details how to set up, engineer and operate ICS devices. However, such documentation typically only describes the steps required for using the devices and software for the intended purpose. Moreover, there are private standardized protocols and interfaces, where it is possible to purchase the specification for a fee. However, since the fee might reach a few thousand dollars (e.g., protocol specification from Open DeviceNet Vendors Association (ODVA)) it will present a barrier for many independent researchers.

In contrast, there is almost no low-level documentation available for most proprietary components with serious implications for forensic investigations.

As one consequence of the lack of documentation, it is difficult to know what kind of data that can be used in an investigation is available in an ICS installation. The availability of data has a huge impact on the types of system compromises and attacks that can be detected and investigated. Therefore it is one of the first and also most important steps to clarify what information can be collected in a specific ICS environment.

Moreover, the lack of documentation about the available APIs makes it difficult to collect the data because the only reliable way to collect data is to use the same APIs and functions as the vendors programs.

Furthermore, some ICS devices might have hidden functions and interfaces that were implemented for maintenance or debugging purposes. Such additional information sources could be very valuable for an investigation, but the uncovering of such sources is often a matter of luck and there are no guarantees wether such functions and interfaces will remain available over time. However these can not be simply ignored. When considering advanced attackers one has to assume that extensive research is performed in preparation for an attack and that knowledge of these types of interfaces is gained, and possibly subsequently employed in a attack. As such, a forensic investigation which does not consider these undocumented features has a clear disadvantage.

After the relevant data has been collected, a further challenge arises in parsing and understanding undocumented data structures in order to utilize the information for a forensic investigation.

Another impact of the lack of documentation is that if a vendor releases a new firmware version, undocumented changes could also affect already developed and tested forensic tools. So additional work might be required to analyze the implemented changes and to adjust the existing tools for each new firmware revision.

3.1.2 Legal constraints

It is very likely that vendors do not want researchers to publish details about internals of proprietary components, because this would allow competitors to copy unique features. This makes the acquisition and exchange of new expertise difficult and results in a lack of forensics knowledge.

In most cases, there are legal constraints regarding reverse engineering, which is a common method to reveal internals and to understand the operation of proprietary components. Legal constraints also restrict the publication and sharing of knowledge about such components.

When access to documentation about system internals is granted, it is usually accompanied by strict legal bounds that limit both how this documentation can be used and how it can be shared.

3.2 Lack of domain specific knowledge and experiences

For the field of traditional IT forensics, a vast body of knowledge has been accumulated over the years that is widely available through books, articles, blog entries and course

material. Additionally, practitioners can receive education through various outlets and educational facilities. Best practice advise is available on how to conduct forensic investigations and standard operating procedures exist for the acquisition, handling and processing of digital evidence.

Almost no such knowledge exists for performing forensics in the context of ICS. While most control systems incorporate, build on and make use of traditional PC equipment, as far as the vendor specific components are concerned any research into this area has to start from scratch.

The most promising course of action seems to be to try to adapt already existing forensic methods and best practices from the traditional forensics realm and apply these to the ICS world. The constraints in these two environments are obviously different but certain steps should be more or less applicable.

There exists no comprehensive knowledge about possible or common attack vectors. This is because documentation on attacks on ICS is relatively scarce and typically only covers high profile, high impact attacks. However, different attack methods leave different kinds of traces and knowledge about attack techniques should inform a forensic investigation and determine what data needs to be collected and analyzed and what to look for. For example, Stuxnet has demonstrated how adversaries can make use of standard vendor APIs and interfaces to infect PLCs. For this it needed to hook into the call-chain by replacing a vendor supplied DLL with its own copy which left a modification in the filesystem. The exploitation of vulnerabilities (e.g. buffer overflows) would also be a possible attack method and would leave characteristic traces in volatile memory.

3.3 Lack of security mechanisms

The design of traditional IT systems has been shaped by the need of preserving the integrity and availability of the system in the face of attacks. Due to the interconnected nature of traditional IT systems, the principle of defense-in-depth is often applied which consist of multiple protection layers and several separate security boundaries.

This has implications for the investigation of security incidents. The various mechanisms that exist in most common IT components to prevent unauthorized access very often generate and preserve traces of intruder activity. Time stamps and logs allow the reconstruction of events and are an indispensable tool in forensic investigations.

Since ICS environments were in the past isolated from other networks by the use of proprietary protocols and dedicated connections, the need to integrate security mechanisms was not as urgent. For example, by default most logging systems store only information that is useful for operational purposes.

It needs to be investigated if, and to what extend, existing operational mechanisms in ICS can be configured in such a way as to become valuable sources for the purpose of a forensic investigation (e.g. by increasing the log level).

3.4 Lack of forensic tools

To the best of our knowledge there are no currently available dedicated forensic tools for ICS infrastructures that allow an analyst to perform a comprehensive investigation. Nevertheless, there exist tools (e.g. LibnoDave [29], libOpenSRTP [30], PLC-Logger [31]), which have promising functionality, but it is unclear whether it is possible to use these tools for a forensic investigation.

For the construction of comprehensive forensic investigation tools it is probably necessary to develop many completely new functionalities that are based upon the gathered knowledge and the gained experiences. The efforts for the development will strongly depend on how quickly the system internals of property components can be understood with the help of in most parts insufficient documentation.

The usage of existing tools for conducting an investigation might be very risky, because it is unknown how reliable such tools are. It should be kept in mind that these tools where not developed for a forensic purpose, therefore extensive testing and reviewing of the code is absolutely mandatory. Otherwise the usage of these programs in an operational environment could put the infrastructure availability at risk. To evaluate the reliability and the usability of such tools it is also important to consider how active is the project development, to evaluate if the project is still active and necessary changes will be implemented.

3.5 Availability/ Always-On

For most ICS environments (e.g. factories or plants) the availability of the whole infrastructure is a top priority. Unavailable components can lead to a decreased production rate or, in a worst-case scenario, to injured or dead people. Therefore it is important that the forensic investigation does not put at risk the availability of the whole infrastructure. As an ICS process is always on, it will not be shutdown but for exceptional circumstances, which results in some side effects that need to be dealt with in the context of a forensic investigation.

A widely known forensic method, the so-called post mortem forensic analysis, seems not to be applicable for most ICS environments because of these being always-on. To

perform such an investigation it is necessary to turn off a component, which would lead to an availability issue for the infrastructure.

The ICS availability requirements also results in a lack of strict patch management for this class of devices. The reason for this is that typically it is necessary to reboot devices to successfully apply a software update, which would in turn negatively impact the availability. Furthermore, it is mandatory in many industries to extensively test or even certify the infrastructure with the updated components to ensure the availability of the environment. The result is that tools must support various firmware and operating system versions for the same device, because it cannot be expected that all devices run the latest software version.

The always-on challenge has also impact on the planning and development of forensic tools. Due to the fact that such tools might be executed in a live production system, it is necessary to be aware of the possible risks and side effects that may occur when running these tools within a complex ICS environment. Vendors might not certify third party forensic tools as compatible with their own products, particularly if the functionality is based upon the knowledge of reverse engineered proprietary functions and components.

In very critical infrastructures the always-on challenge could result in that only passive information gathering will be allowed because of the non-tolerable risk that other methods will possibly have on the availability.

The ability to reconstruct events in a forensic investigation requires the preservation of data as quickly as possible after the security incident. However, it needs to be considered that it might require a lengthy process to get the permission to integrate forensic sensors or to execute forensic tools because of the unknown side effects on the critical infrastructure. Therefore it is desirable that an ICS environment is prepared for a forensic examination beforehand, so that no relevant information for an investigation gets lost after the incident.

Furthermore, the always-on challenge has also an impact on the methods that are available and suitable for the collection of information. For example, some devices permit the dumping of the firmware to a local storage medium. To perform this operation it is necessary to stop the component which in turn again affects at least the device itself if not the whole ICS infrastructure. Therefore this collection method would not be usable for a forensic investigation.

3.6 Real time/ Resource constraints

For the reliable operation of an ICS environment it is often mandatory that the various components are able to respond to occurring events in a fixed time window. To satisfy

these needs, a so called real time operating system (RTOS) is used.

PLCs and RTUs are essential components in ICS environment and typically are very limited with respect to their processing power and memory capacity.

Therefore, tools used in a forensic investigation in industrial control systems should not negatively impact the real-time requirements and should work inside these resource constraints.

These two properties of an ICS infrastructure are relevant especially during the data acquisition phase. In most cases it will neither be possible nor desirable to retrieve all available data from all components. If too much information is gathered for a forensic investigation, the real time operation of the different devices might be affected because of the high number of information gathering requests. Instead, only the key information sources need to be identified.

It is also important to consider that the data collection also generates additional network traffic which might result in a transmission delay of the packages that are needed for the plant operation. Therefore, special care must be taken so that the data acquisition itself does not unnecessarily strain the process network.

3.7 Distributed nature

Large ICS environments consist of a multitude of components, which may be distributed over a large geographical area. For a comprehensive forensic investigation, it is important to also include these distributed components. Conducting a forensic investigation in such a distributed infrastructure is more challenging than in a non-distributed environment.

To conduct an investigation it is mandatory to have a good understanding of the ICS infrastructure. In a distributed environment it is much more difficult to get such an understanding because of the additional complexity. The distributed components might be slightly different from the devices that are used within a plant or factory because they have to be able to communicate with special purpose protocols and withstand rough conditions. Moreover, the dependencies between components in the system might be harder to understand because of the number of devices that are communicating with each other.

For a distributed environment it has to be taken into account that the collection of information from all components at a specific time might not be possible. The reason for this is, for example, that some distributed components are connected with the central infrastructure via a WAN connection with a small bandwidth and high delay. Therefore it might not be possible to get a snapshot of all needed information of the whole environment at a specific point in time.

Moreover, the clock of the different components might not be synchronized. Therefore, the correlation of events in these distributed environments during a forensic investigation requires to detect and account for any time shift intrinsic to the components or introduced during the data collection.

The connectivity of the distributed components also impact the amount of data that can be acquired and the intervals how often this information can be gathered. Therefore the forensic tools used for data acquisition need to be configurable so that they can be used in different environments with various bandwidth properties. Some distributed components might not allow remote access at all and even physical access might be difficult. Tools for data acquisition need to take these situations into account.

3.8 Accessibility

The accessibility and bandwidth constraints of individual components also apply to ICS systems as a whole. Industrial control systems are often used in remote locations like off-shore installations. Data transfer speeds are sometimes low (eg. only satellite uplink or service technician traveling by boat). Furthermore, for security reasons, remote access to these installations might be restricted or not possible at all.

This puts specific requirements on the data collection. In particular, the data collection needs to be decentralized and agent based. Furthermore, all data required for an investigation should be collected in one go. Since the nature of the incident, which largely determines the type of information that is required to solve the case, is often not known from the outset, the last requirement means that some processing and filtering has to be done by the agent. Since ICS installations are frequently not connected to the internet, the transfer of the collected data needs to be possible via offline methods.

3.9 Diverse / Complex

Another challenge typical for ICS environment is the diversity of such infrastructures. This results from the fact that many different ICS components are available from different vendors and are often combined within one ICS installation.

Furthermore, it is not unusual that industrial installations include some rather old components, because the lifetime of ICS devices can be up to 20 years.

Due to the challenge of upgrading and patching ICS components, multiple OS and firmware versions might be in use in the same installation.

This mixture of new and old components from different vendors and running different operating systems and firmware versions results in a high complexity of many industrial

installations.

For the development of tools and the understanding of system internals it is necessary to build appropriate test environments but because of the complexity of real world infrastructures and the high diversity of components it is very complicated to model realistic test environments. The configuration of different components might be very time-consuming and labor-intensive, especially if the test environment is to produce realistic data and behave like a real world infrastructure. In addition to this it is infeasible to purchase all available ICS components from all vendors because some devices and software packages are very expensive, especially when it comes to the top-models of the vendors. Therefore a comprehensive tool support for all kind of components might be very difficult.

4 Attack Scenarios

Traditional computer forensics is used in a variety of contexts and scenarios. The investigations of financial fraud, compliance cases, or hacking incidents have little in common with respect to the goal of the investigation. However, similar tools, approaches and methods are used in these diverse scenarios.

For the Crisalis project we are concerned with the investigation of security incidents in industrial control systems. To gain a better understanding of the requirements we consider the following scenarios:

1. Mass malware infection
2. Hacking
3. Malfunction
4. Targeted attack

The forensic tools and methodologies to be developed as part of the Crisalis project will largely concentrate on the investigations of the various host components and subsystems of industrial control installations.

4.1 Mass malware infection

Infections of standard Windows PC components by mass-malware are by far the most common group of incidents that affect industrial control systems. Common infection vectors are infected USB sticks or laptops of service technicians. The malware in these cases doesn't target the control installation and also don't contain any features specific to control systems. Once inside the process control subnet, the infection might spread to other Windows based components.

The forensic tools and methodologies need to be able to reliably detect these threats. For this, they need to concentrate on the executable file content and the running processes of all the standard Windows based components in the ICS network.

4.2 Hacking

Manual hacking of control installations typically exploits weaknesses in the setup, configuration and maintenance of these systems. The most common scenario are remote access capabilities that are insufficiently protected. An example could be a remote VNC connection protected by a weak default password that is discovered by an intruder and used to gain access to the system. The intruder could then employ the standard mechanisms available to the operators to manipulate the plant operation.

A forensic investigation in these cases needs to concentrate on the available logs, in particular the access logs.

4.3 Malfunction

A third group of incidents commonly encountered in industrial control systems are malfunctions of components. They might display certain features of security incidents and from the outset it is usually not clear whether the malfunction was accidental or caused by an intruder.

An example could be the erratic effects caused by a faulty keyboard or mouse connected to a control terminal. The forensic investigation in these cases needs to be able to rule out with a sufficient degree of certainty any malicious context and help pinpoint the cause of the observed behavior.

4.4 Targeted attack

The Stuxnet incident was the most publicized and best researched example of a cyber sabotage attack against an industrial control system [32],[33].

Targeted attacks like Stuxnet have the potential to affect all elements of control installations. They might employ anti-forensic measures to increase their chance of survival. They share certain characteristics both with the standard-malware infection and the hacking scenarios but also have additional features.

They are therefore the most challenging incidents to analyze. The tools and methodologies to be developed need to be able to detect and locate the intrusion and help to identify the extend of the manipulations.

5 Model Architecture

The term "Industrial Control System" (ICS) encompasses a wide range of different types of control system configurations that vary in their architectures, components and connections. Examples range from supervisory control and data acquisition systems (SCADA) that are used in distribution networks for energy, water, oil or gas, to process control systems (PCS) used for the control of continuous processes such as power generation or chemical plants, to factory automation systems (AS) used to control the manufacturing of goods and the delivery of services.

As a precondition to the development and testing of the forensic tools, methods, and procedures we therefore need to specify a model environment that we will use as a reference point.

5.1 Siemens SIMATIC PCS7

As the base model we choose a simple, best-practice installation of the Siemens Simatic PCS7 process control system suite. PCS7 is the Siemens main distributed control system (DCS) offering and it was chosen as our model environment due to its relevance for the objectives of the Crisalis projects for three main reasons:

1. PCS7 is currently adopted in *critical infrastructure* installations such as oil refineries and power generation plants
2. PCS7 has a large worldwide market share and is therefore *relevant for real-world scenarios*
3. a PCS7 installation was the likely target of the *Stuxnet incident* in 2010 that demonstrated the vulnerability of industrial control systems to targeted attacks

Siemens recommends the implementation of a cell concept as the basis of the security architecture of a plant installation with the logical separation of different sub-networks.

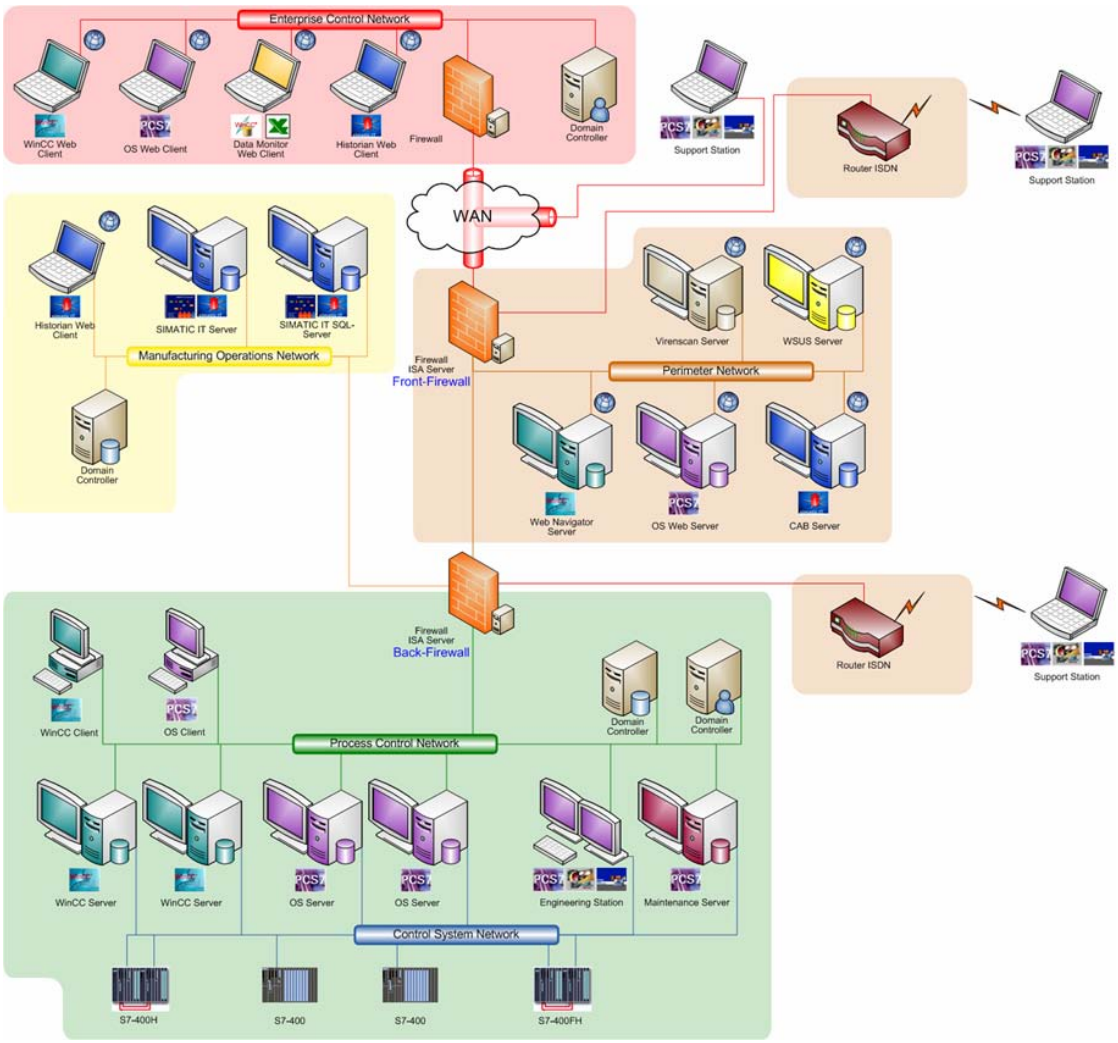

Figure 5.1: The Siemens SIMATIC cell security concept

The main control components are located in the lower green cell of Figure 5.1, that hosts both the process control network and the control system network.

In particular, the model system we implemented contains the following components (see Figure 5.2):

1. OS Client - this is the human machine interface (HMI) operator station. It is a

Microsoft Windows XP PC running the Simatic HMI software. A human operator can supervise and control the operation of the plant from this machine.

2. OS Server - this is the central DCS server. It is a PC running the Microsoft Windows Server 2003 operating system with the MSSQL database software and the PCS7 software packages hat implement the DCS server functionality.

3. Engineering Station - this is the central station from which the layout of the plant, the connections between the different components, and the logic running on the PLCs are configured. It is a PC running Microsoft Windows Server 2003.

4. Archive server - this is the historian server that stores historical process data. It is a PC running the Microsoft Windows Server 2003 operating system.

5. PLCs - the programmable logic controllers are connected to the control system network and control the operation of the actuators and sensors connected to the field network.

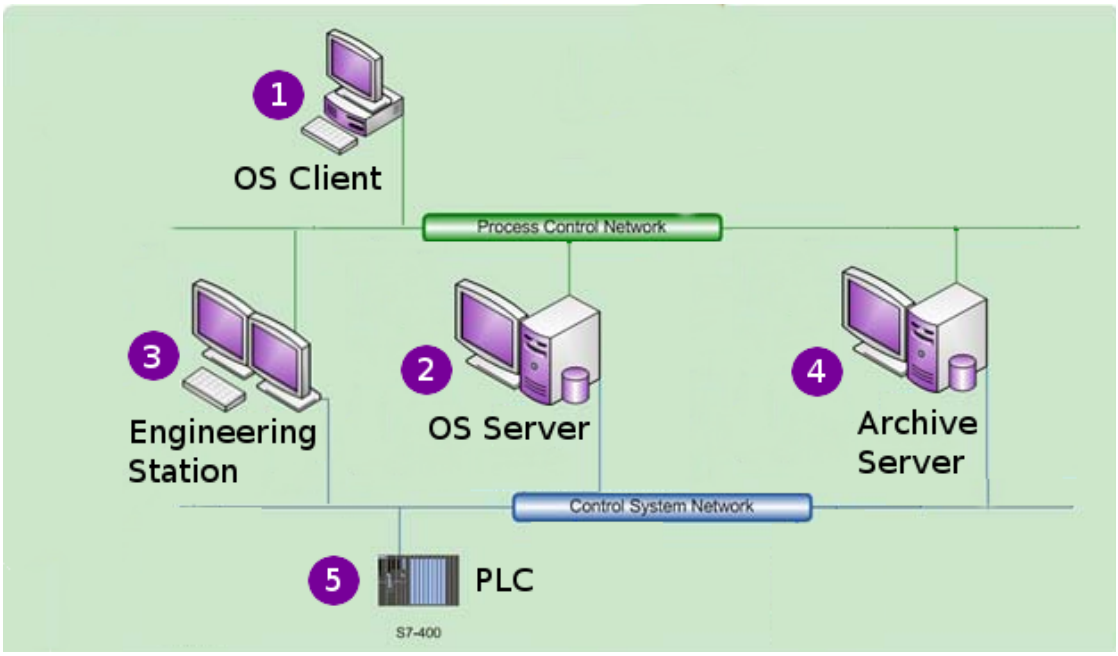

Figure 5.2: The PCS7 model installation

6 Forensic analysis framework

The specifics of many control system installations prevent the application of classical offline forensic methodologies that require the powering down of the equipment, the subsequent dismantling and removal of all storage devices, the creation of bit-images of the storage devices using write-blockers to prevent accidental changes and the analysis of the acquired filesystem images.

Instead, the distributed nature, the continuous operation, the lack of connectivity to the internet and the remote location of many control system installations that we described in chapter 3 require a different approach. In particular, they impact the way that forensic data has to be collected.

In CRISALIS we therefore propose a solution based on the combination of a software agent that is temporarily deployed on these systems and extracts selected forensic data, with the processing of the data by back end software and the final review of the processed data by a forensic analyst.

These components, the forensic agent, the back end processing software and the review environment constitute the framework for the forensic analysis of industrial control systems. In the Crisalis project we set out to develop such a framework as a proof-of-concept.

6.1 Goals and requirements

Based on the discussions in sections 3, 4 and 5, the framework should satisfy a number of goals.

- Ability to extract and process forensic data from standard Windows based PCs. Most components of industrial control systems, in particular those that we consider as part of our model PCS7 architecture, are based on standard PCs running variants of the Microsoft Windows operating system. The framework should be able to extract and process forensic data maintained by the operating systems such as filesystem information, registry information and system events.

- Ability to extract and process control specific data. In addition to the information maintained by the operating system we need to extract and process information

that is specific to the control system and maintained by vendor specific control software. In particular, the tool should be able to

  - enumerate PLC components connected to the process control network and to perform the extraction of the user code and data blocks running on them over the network
  - extract engineering data from the engineering station
  - extract process data from the historian server

- Flexibility. Industrial control software is used in a variety of settings. From power generation installations to transportation systems and chemical plants, we envision the framework being used in different contexts. Although these systems share basic common characteristics, the framework needs to allow to be adjusted to these different environments. This might mean that a specific log file needs to be collected and included in the analysis, that specific credentials need to be used for the access to certain components or that specific file formats and data structures need to be analyzed.

- Quick reaction to security incidents. The framework should enable an analyst to quickly solve an incident case. This includes all stages ranging from the deployment of the agent, the collection and transfer of the data, to the processing and review stage.

- Ability to solve different compromise scenarios. The framework should enable an analyst to solve at least the four different types of scenarios that we discussed in 4: standard malware infections, hacking incidents, malfunctions and targeted sabotage attacks. These different scenarios require the collection and analysis of different types of data. The extracted data should therefore be comprehensive enough and should include both historical data such as log files, run-time data such as process and network connection lists, and meta-data from the filesystem and the registry.

- High fidelity and protection of the extracted data. The extraction should prevent interception and tampering of the data by the attacker using rootkit like mechanisms. Since the extracted data might be sensitive, the confidentiality and integrity of the data needs to be protected during transit.

- Low run-time footprint. Since many devices used in existing control systems have very limited processing power and since the extraction process should impact the

operation of the plant as little as possible, the extraction step should only do a very limited amount of processing of the data.

- Limited amount of extracted data. Since the connectivity of remote plant installations is occasionally limited with only low-bandwidth connections available for the exfiltration of the collected data, the amount of the data collected should be small.
- Data reduction. To review large amounts of data manually is difficult and time consuming. The processing step should therefore employ various filtering mechanism to achieve a high amount of data reduction.
- Machine intelligence. We envision the framework being used by analyst of different experience and skill levels. Ideally the forensic processing would yield automated answers to the various possible questions posed: Is the machine compromised? How did it happen? When did it happen? Although a fully automated intelligence is likely unattainable, the framework should at least flag and prioritize suspicious findings.
- Ease of use. The data presentation for the review step should be intuitive and easy to work with. The framework should be able to combine, filter and augment data in such a way that a human analyst is quickly able to review the data and draw conclusions.

These abstract goals lead to the following design requirements:

1. The agent should be a universal binary for IA32 and IA64 architectures and should only use APIs and operating system mechanisms available on the WindowsXP, WindowsVista, Windows7, Windows Server2003 and Server2008 operating system variants.
2. The agent should include code to enumerate PLCs visible on the network and collect data from them. It should also be able to extract other control specific data available on the host the agent runs on.
3. The framework should use a plugin architecture combined with a scripting language. A config file should be used to adjust the operation of the agent.
4. The agent should be optimized for a quick deployment and for fast collection of the data. The processing of the data should be largely automated. The data presented for review should be relevant and easy to review.

5. The collection and analysis should include run-time data, meta-data and log file data.
6. The agent should use low-level APIs and mechanisms for the extraction of data when possible. The data should be encrypted during transfer.
7. The amount of processing required on the host should be minimized.
8. The data extraction should balance the requirement of reconstructing a comprehensive picture of the incident with limited bandwidth availability. This should be configurable via a configuration file such that in situations that are severely bandwidth-constraint certain data sources are not extracted.
9. The framework should implement filtering mechanisms such as whitelisting to reduce the data size that needs to be reviewed
10. The framework should use anomaly detection and heuristic methods to identify and prioritize suspicious objects for review
11. The framework should be able to combine data values from different subsystems and components into a common timeline to present an integrated view of the plant operation to the analyst

6.2 Forensic agent

We started implementing a proof-of-concept forensic agent that incorporates previously listed requirements. The agent is responsible for the collection of various data sources. The extraction logic is written in the Python programming language and compiled into a single executable *agent.exe* that runs on 32-bit and 64-bit Windows PCs. The Python code can easily be adjusted and extended for different scenarios. The main parts of the data extraction are controlled by a configuration file embedded in the executable.

We also set up a build environment that consists of two virtual machines running 32- and 64-bit variants of the Windows operating system with build scripts that automate the generation of the agent executable.

The agent collects the following data sources:

- Master file table (MFT) of all fixed drives
- Registry hive files
- Event log files

- Other log files such as firewall logs, anti-virus logs, VNC access logs, etc.
- Run-time information such as running processes, active network connections, loaded drivers etc.

The agent also contains logic to identify suspicious executable files that are present on the system and extracts those executables. It is able to extract locked and hidden files as well as data contained in NTFS alternate data streams by directly interacting with the physical device corresponding to the filesystem partition, identifying the inode attached to the file and the corresponding sectors and reconstructing the file content from the data contained in the sectors.

We also wrote experimental code to enumerate PLCs on the network and collect code and data blocks that are available on them. This code uses the Siemens S7 protocol to communicate with the PLCs and consequently is highly Siemens specific.

6.3 Analysis back-end

We also started developing the analysis back-end component. The back-end is written in the Python programming language. It performs the following steps:

- Convert various binary format objects into a parseable ASCII/Unicode format. In particular the master file table (MFT) objects, the registry hives and the eventlog files need to be converted
- Reconstruct deleted objects. Deleted files and registry entries are reconstructed.
- Extract time stamps from the different objects and combine them into a universal timeline. Time information is extracted from the file system (MFT) for every file, from the registry for every key and from the event log files for each event log entry. The time stamps from the filesystem include creation, modification, access and entry modification times both from the $StandardInformation attribute and from the $FileInformation attribute.
- Perform anomaly detection and heuristic checks to identify suspicious objects. Currently a small number of basic heuristic checks are implemented to identify potentially suspicious files. Also, work has started on implementing a local outlier factor (LOF) anomaly detection algorithm to cluster filesystem events and detect outliers.

7 Conclusion

In this document we described the challenges of performing forensic investigations in industrial control systems and derived requirements for the implementation of forensic tools. We detailed the design of a forensic framework suitable for IC systems that was derived from these requirements. And we layed the groundwork for the implementation of such a framework, starting with the development of basic functionality for a forensic agent and accompanying back end.

In the second part of this work package of the Crisalis project we intend to implement this framework and test it in different scenarios and environments. We plan to augment it with an incidence response methodology that incorporates the forensic analysis of industrial control systems using the developed tools, and derive lessons learned and suggestions for a better preparation and increased readiness of IC systems in the face of security incidents.

Bibliography

[1] Gartner. IT and operational technology alignment. `http://www.gartner.com/technology/research/it-ot-alignment/`, 2013. [Online; accessed 20-April-2013].

[2] Wikipedia. Digital forensics — Wikipedia, the free encyclopedia. `http://en.wikipedia.org/wiki/Digital_forensics`, 2013. [Online; accessed 20-April-2013].

[3] T. Kilpatrick, J. Gonzalez, R. Chandia, and M. Papa. An architecture for SCADA network forensics. In *IFIP Int. Conf. Digital Forensics'06*, 2006.

[4] Douglas Schweitzer. *Incident Response - Computer Forensics Toolkit*. Wiley, 2003.

[5] Chris Prosise, Kevin Mandia, and Matthew Pepe. *Incident Response and Computer Forensics, Second Edition*. McGraw-Hill, 2001.

[6] Harlan Carvey and Eoghan Casey. *Windows Forensic Analysis*. Syngress, 2009.

[7] Jesse Varsalone, Ryan R. Jubasiak, and Sean Morrissey. *Mac OS X, iPod and iPhone Forensic Analysis DVD Toolkit*. Syngress, 2009.

[8] Chris Pogue, Cory Altheide, and Todd Haverkos. *UNIX and Linux Forensic Analysis DVD Toolkit*. Syngress, 2008.

[9] Brian Carrier. *File System Forensic Analysis*. Addison-Wesley, 2005.

[10] Harlan Carvey. *Windows Registry Forensics*. Syngress, 2011.

[11] James M. Aquilina, Eoghan Casey, and Cameron H. Malin. *Malware Forensics - Investigating and Analyzing Malicious Code*. Syngress, 2008.

[12] Encase. `http://www.guidancesoftware.com/encase-forensic.htm`. [Online; accessed 20-April-2013].

[13] Forensic toolkit. `http://www.accessdata.com/products/digital-forensics/ftk`. [Online; accessed 20-April-2013].

[14] X-ways forensics. `http://www.x-ways.net`. [Online; accessed 20-April-2013].

[15] Volatility. https://code.google.com/p/volatility, 2013. [Online; accessed 20-April-2013].

[16] Nick L. Petroni, Aaron Walters, Timothy Fraser, and William A. Arbaugh. Fatkit: A framework for the extraction and analysis of digital forensic data from volatile system memory. *Digital Investigation, Volume 3, Issue 4*, 2006.

[17] EncaseEnterprise. http://www.guidancesoftware.com/encase-enterprise.htm. [Online; accessed 20-April-2013].

[18] Google Rapid Response. https://code.google.com/p/grr. [Online; accessed 20-April-2013].

[19] RSA ECAT. http://www.siliciumsecurity.com/. [Online; accessed 20-April-2013].

[20] Mandiant Redline. http://www.mandiant.com/resources/download/redline. [Online; accessed 20-April-2013].

[21] Xry. http://www.msab.com/. [Online; accessed 20-April-2013].

[22] Paraben device seizure. http://www.paraben.com/index.html. [Online; accessed 20-April-2013].

[23] Marcel Breeuwsma, Martien De Jongh, Coert Klaver, Ronald Van Der Knijff, and Mark Roeloffs. Forensic data recovery from flash memory. *Small Scale Digital Device Forensics Journal, Volume 1, Issue 1*, 2007.

[24] M.F. Breeuwsma. Forensic imaging of embedded systems using jtag (boundary-scan). *Digital Investigation, Volume 3, Issue 1*, 2006.

[25] Norrie Bennie. Forensic analysis of a usb flash drive. http://computer-forensics.sans.org/community/papers/gcfa/forensic-analysis-usb-flash-drive_201, 2005. [Online; accessed 20-April-2013].

[26] Krishnun Sansurooah. A forensics overview and analysis of usb flash memory devices. In *Proceedings of the 7th Australian Digital Forensics Conference, Edith Cowan University, Perth Western Australia*, 2009.

[27] G.B. Bell and R. Boddington. Solid state drives: The beginning of the end for current practice in digital forensic recovery? *Journal of Digital Forensics, Security and Law, Volume 5, Issue 3*, 2010.

[28] B. J. Phillips, C. D. Schmidt, and D. R. Kelly. Recovering data from usb flash memory sticks that have been damaged or electronically erased. In *Proceedings of the 1st international conference on Forensic applications and techniques in telecommunications, information, and multimedia and workshop*, 2008.

[29] Thomas Hergenhahn. Libnodave – exchange data with siemens plcs. `http://libnodave.sourceforge.net`, 2013. [Online; accessed 20-April-2013].

[30] Thomas Hergenhahn. Libopensrtp – exchange data with ge fanuc plcs. `http://libopensrtp.sourceforge.net`, 2013. [Online; accessed 20-April-2013].

[31] Various. Plc-logger. `http://sourceforge.net/projects/plclogger`, 2013. [Online; accessed 20-April-2013].

[32] Nicolas Falliere, Liam O Murchu, and Eric Chien. W32.stuxnet dossier. `http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/w32_stuxnet_dossier.pdf`, 2011. [Online; accessed 20-April-2013].

[33] Geoff McDonald, , Liam O Murchu, Stephen Doherty, and Eric Chien. Stuxnet 0.5 - the missing link. `http://www.symantec.com/content/en/us/enterprise/media/security_response/whitepapers/stuxnet_0_5_the_missing_link.pdf`, 2013. [Online; accessed 20-April-2013].