INFORMATION SECURITY ASSESSMENT OF CERN ACCESS AND SAFETY SYSTEMS

T. Hakulinen, X.B. Costa Lopez, P. Ninin, P. Oser
CERN, Geneva, Switzerland

Abstract

Access and safety systems are traditionally considered critical in organizations and they are therefore usually well isolated from the rest of the network. However, recent years have seen a number of cases, where such systems have been compromised even when in principle well protected. The tendency has also been to increase information exchange between these systems and the rest of the world to facilitate operation and maintenance, which further serves to make these systems vulnerable. In order to gain insight on the overall level of information security of CERN access and safety systems, a security assessment was carried out. This process consisted not only of a logical evaluation of the architecture and implementation, but also of active probing for various types of vulnerabilities on test bench installations.

INTRODUCTION

Information Security of Control Systems

While network accessible server and desktop systems are nowadays out of necessity fairly well secured, control systems have until recent times been largely ignored when it comes to information security. This is due to the fact that most such systems are by their nature private and disconnected. Control system vendors also tend to strongly recommend that their systems be kept in isolation of publicly available networks, for which reason neither the vendors nor the clients may have seen the need for rigorous security. However, tendency nowadays is to increase information exchange between control systems and the rest of the organization to allow, e.g., supervision and remote control, and to facilitate operation and maintenance. Recent events, such as the infamous Stuxnet episode [1], have also served to demonstrate that even well isolated systems are not necessarily immune to security problems.

Access and Safety Systems

CERN GS/ASE group is responsible for the management of CERN personnel access and safety systems. This includes specification, design, implementation, contracting, operation, and maintenance of these systems. The current principal access and safety systems under our responsibility are LHC Access Control System (LACS), LHC Access Safety System (LASS), PS Access Control System (PACS), PS Access Safety System (PASS), Personnel Safety System of the SPS complex, Surveillance of Sites system (SUSI), CERN Safety Alarm Monitoring system (CSAM), Gas detection and alarm system (Sniffer), Site Information Panels / Simple Access Messages system (SIP/SAM), and Safety System Atlas (SSA).

CERN access and safety systems typically consist of many different kinds of devices, such as Windows and Linux servers, operator posts, panel-PCs, PLCs, video cameras, interphones, card readers, biometry scanners, etc. These devices come from many different vendors, and they are nowadays mostly directly network connected. Access systems reside mainly in the restricted CERN Technical Network (TN) but some devices reside also in the CERN-public General Purpose Network (GPN). The most important systems have their own private networks. Most systems also have their respective test platforms, which aim to replicate the production systems in sufficient detail, albeit in a smaller scale, in order for development and testing to be possible without endangering production.

We carried out an information security assessment of two of our most visible access systems, LACS and PACS. The work consisted first of creating an inventory of all the devices and services, which included establishing dependencies between systems, and then carrying out active penetration testing of the various services using available security tools or writing specific utilities, whenever necessary. The goal was not only to find any existing vulnerabilities, but also to gain a larger understanding of the principal risks involved and to develop procedures for managing such risks.

INVENTORY

The first task in a security assessment of a complex system is to create a comprehensive inventory of the entire system: types of devices, operating systems they run, services they use and provide, interdependencies they have. After all the devices were identified and categorized, representatives of the most interesting devices within a category were selected and focused on. Fifteen generic groups were identified within the test systems: servers, access point PCs, operator post, gateway PLCs, access control units, biometry scanners, webcams, network devices, interphone master stations, intercom servers, interphone client stations, UPS monitoring devices, virtual machine servers, generic windows PCs and generic PLCs. Every device was also reviewed for brand, model, operating system, installed software, and running services, as shown in Table 1.

ISBN 978-3-95450-148-9

Table 1: Part of the Inventory of LHC Test Bench Devices. Device Names Are Grayed Out

Type	Name	Brand	Model	OS	Software/ Services
Access point PC	LHC0	IEI	PPC-5150	WINDOWS 7	MS Terminal Service, MS Windows RPC
Windows devices	LHC0	HP	COMPAQ DC7100	WINDOWS 7	MS-DS Active Directory, MS Terminal Service
Windows devices	LHC0	HP	PAVILION	WINDOWS 7	MS-DS Active Directory, MS Terminal Service
Windows devices	LHC0	HP	PROLIANT	WINDOWS 2008	MS-DS Active Directory, MS Terminal Service, Oracle MTS Service
PLCs	LHC0	SIEMENS	S7-1200	STEP7	Industrial Port, HTTP

Interdependencies between devices were identified. These include the following:

- Access point PLCs depend on access point PCs.
- Biometry scanners depend on biometry servers.
- Webcams depend on video servers.
- Access control devices and operator posts depend on access control servers.
- Intercom devices depend on intercom servers.
- Everything depends on network devices.

Results from an interdependency analysis focus attention to the important testing targets: which devices are critical for the correct functioning of the entire system and what the consequences of a breach of a certain device could be to those that depend on it.

SECURITY ASSESSMENT

Methodologies

The different probing methodologies used in this assessment were deterministic (local or remote) and fuzz testing (fuzzing) attacks. Deterministic techniques, like local and remote attacks compromise a system precisely as specified. For example a local exploit may need a particular operating system version and a remote exploit may need a specific service or application running.

The idea of fuzzing is to try to proof the software against incorrectly implemented code, by testing it with non-deterministic (fuzzy) techniques. Fuzzing is used by software developers and auditors to find exceptions, which are not properly handled. Fuzzing techniques don't usually pose precise requirements, because they make use of different services or operating systems.

Preliminary Testing

After the initial inventory, a period of preliminary pilot testing of a small number of devices was carried out to determine the scope of the project, any special conditions, and the tools needed. The scope was determined based on typical attack vectors and types in information systems as presented in Table 2.

Table 2: Typical Attack Vectors and Types [2]

Attack Vectors	Attack Types
Code Injection	Buffer Overflow Buffer Underrun Viruses Malware
Web Based	Defacement Cross-Site Scripting (XSS) Cross-Site Request Forgery (CSRF) SQL Injection
Network Based	Denial of Service (DoS) Distributed Denial of Service (DDoS) Password and Sensitive Data -Interception Stealing or Counterfeiting Credentials
Social Engineering	Impersonation Phishing Spear Phishing Intelligence Gathering Tailgating

Tools

Several security-testing frameworks were evaluated, and the following tools were chosen for the project:

- **Metasploit Framework** is an open source penetration testing software project for security officers and administrators.
- **Armitage** is a GUI for the Metasploit Framework.
- **nMap** is a network mapping tool for discovering services, open ports, operating systems, and subnets.
- **Wireshark** is a versatile protocol analysis tool that displays network packets in a human readable form allowing sophisticated filtering and analysis.
- **Backfuzz** is a multi-protocol fuzzing toolkit supporting the most important network protocols.
- **W3af** (Web Application Attack and Audit Framework) is an open source tool for finding vulnerabilities in web-applications.
- **Nikto** web scanner detects outdated software, dangerous files and CGIs on web servers.
- **BeEF** is a penetration-testing tool that exploits web browsers. It creates a dedicated server on the attacker system to listen for connections.
- **THC Hydra** is a password-cracking tool that uses dictionary attacks against servers and databases using various protocols.
- **THC flood_router26** is a denial-of-service script that floods the network with router advertisements.
- **THC smurf6** is an IPv6 tool for DDoS (Distributed Denial-of-Service) attacks.

Penetration Testing

The tools were managed using a special Linux distribution, Kali Linux [3], which comes standard with a suite of security tools. This system was used as the platform for all penetration testing as well as network and system monitoring. As the network segment of the PS access system test platform is private to us, we were able to carry out tests, which normally might have been risky on publicly accessible segments.

ISBN 978-3-95450-148-9

Findings

The findings were classified using probability and criticality rating defined in the testing guide of the Open Web Application Security Project (OWASP) [4]. The classification contains 8 categories, of which two first are shown as an example in Table 3.

Table 3: Probability and Criticality Rating of Issues

	Probability Rating		Criticality Rating
1	**Skill level of hackers**	1	**How much data is affected that could be disclosed**
	(1) No technical skills (3) Some technical skills (4) Advanced computer user (6) Network & programming skills (9) Security penetration skills		(2) Minimal non-sensitive data disclosed (6) Minimal critical data disclosed (6) Extensive non-sensitive data disclosed (9)Extensive critical data disclosed or all data disclosed
2	**How motivated they are**	2	**How sensitive is the data that could be disclosed**
	(1) Low or no reward (4) Possible reward (9) High reward		(2) Minimal non-sensitive data disclosed (6) Minimal critical data disclosed (6) Extensive non-sensitive data disclosed (9)Extensive critical data disclosed or all data disclosed

Among the issues discovered were things like missing or vendor-default passwords in embedded devices, unpatched bugs in Windows machines causing them to crash or freeze, unsecured web-interfaces, open ports and unnecessary services on devices. Most of these issues can be fixed by configuration and patching. However, sometimes vendor patches for some embedded devices are not available despite requests, and the only options are to either isolate them or decide to ignore the risk (the latter option applicable to non-critical devices only).

A particular worry is the sensitivity of PLCs to intrusion and denial-of-service attacks. A simple network scan can crash a PLC and even sending commands to access PLC memory or to reboot it is possible using existing toolkits [5]. Network interfaces of old generation PLCs are often badly implemented and very intolerant of disturbances. Newer generation units can often be better secured. Of particular importance is protecting the Siemens 400 series PLCs acting as gateway devices between more restricted safety and more relaxed access systems. These units must be run in the password-protected mode.

Tools for Best Practices

Best practices of the computer security industry are available and tools exist for their implementation, auditing, and enforcement. We tried two: Lynis [6], for auditing Unix and Linux systems, and Open Vulnerability Assessment System (OpenVAS) [7], which is a combination of several services and tools. These are principally frameworks for gaining information and understanding of the systems and to be used in connection with other tools.

OTHER FINDINGS

While carrying out the security assessment, other security issues were also tested besides those directly connected with the target systems. Interesting observations were made on how a seemingly private network could be compromised, what risks are involved in the deployment of IPv6 in a network segment, and in particular, what the importance of physical security of installations is.

Tunnelling Out of Private Networks

One of the main reasons for isolating access and safety systems within their own private networks is security. Reducing or even completely blocking information exchange between networks greatly reduces the probability of successful remote attacks. This becomes particularly important considering that critical systems may not be able to follow the same update and patching cycles as less critical ones. Older systems may also be running obsolete hardware and software, which is known to be insecure, but which cannot be upgraded without a complete redesign of the system – a lengthy and expensive process possibly even requiring revalidation by national authorities.

While the critical safety systems are normally in strict isolation, we have also implemented some less critical access systems within their own private network segments but with controlled routing to other CERN networks. In this case only a few hosts can be accessed from outside of the private segment, but all devices enjoy basic central network services, DHCP, DNS, and NTP. It turns out that it may be possible to exploit DNS for unauthorized, difficult-to-detect tunnelling from the private network through firewalls all the way to the public Internet. This is based on the observation that DNS protocol allows inserting arbitrary data in the query and response packets, which can hence be used to transfer data. All that is needed is that DNS be allowed to resolve outside host and domain names from inside the private network. Iodine software [8] was used for this test:

1. A special DNS client is installed on a machine in the private network.
2. A special DNS server is set up in the Internet with its own top domain.
3. The client makes a DNS query to a subdomain of the top domain with a data payload attached.
4. Server answers with its own data-stuffed packet.
5. Client makes another DNS query to a different systematically named subdomain in order to avoid DNS caching by intermediate servers, etc.

While DNS service is useful for internal name resolution, blocking this exploit requires that hosts from private networks not be able to make DNS queries to the outside domains. Firewall could also be configured to inspect DNS packets for suspect payloads.

IPv6 Issues

IPv6 [9] has been right around the corner for the last 20 years. Only recently has it really started to be implemented, as the IPv4 address space is finally getting scarce. In addition to a much larger address space, IPv6 provides a number of new functionalities to facilitate network management, but which can also be used to subvert network security. As the protocol has only been in widespread use for a relatively short time, return of experience on it is still limited. As an example, we tested two particular is-

ISBN 978-3-95450-148-9

sues having to do with the new Stateless Address Auto Configuration (SLAAC) feature of IPv6:

1. A malicious host can freeze a Windows host by sending a large number of router advertisements with a different route prefix thus simulating different subnets. Until recently, Windows didn't limit the size of IPv6 routing table, which allowed this attack to fill up the system virtual memory and crash the system. After Microsoft update MS14-006, which limits the system routing table size, the system no longer crashes, but stays at 100% CPU while an attack is running and recovers after. The same behaviour is seen with Linux systems during an attack.
2. A man-in-the-middle attack is possible in mixed IPv4 and IPv6 networks. If the routers in the network are IPv4, a malicious host may use IPv6 router advertisements to hijack connections because an IPv6 router has a higher priority than an IPv4 one.

There are ways to mitigate these kinds of problems on the host level: firewall rules can be used to allow traffic only to known hosts and routers, router discovery can often be turned off on the host network stack, or IPv6 can be completely disabled on the host. The last option is particularly reasonable for control systems, where IPv6 support is in any case missing or often poorly implemented.

Importance of Physical Access

When considering any system, and in particular a critical isolated one, restricting physical access to it is of prime importance. An expert able to access a device can in principle do whatever he wants with it. It may not even be necessary to access the device in person, but to have someone else do it instead, e.g., by tricking the person to connect an infected USB key into a restricted system using social engineering. This is in fact very likely the way the Stuxnet worm was first introduced into the classified Iranian nuclear installations.

In supervised areas it may not be possible for an intruder to work with a device without being noticed even if he could access the facilities. There may also not be enough time to carry out compromising changes to a system even if a lapse in security could be exploited. However, devices exist that make this kind of an intrusion a lot faster and easier. We tested a USB keyboard injection device Rubber Ducky [10], which installs itself as a keyboard device to a system and runs a script that can do whatever a user could from the console. The device looks like any USB key as seen in Figure 1.

Figure 1: USB keyboard injection device Rubber Ducky.

Preparing a script for the injection device obviously requires good knowledge of the system in question. However, as most control system computers use commodity operating systems like Windows, this is often not a difficult thing to figure out. Once inserted into the system, the device is able to play a script very fast minimizing the exposure time of the intruder.

A Simulated Intrusion Scenario

It is instructive to conceive a scenario for an intrusion based on the above findings. Assume a semi-critical system running in its own private network but allowing basic network services from outside for convenience:

1. The intruder finds out basic information of the target: operating system, possibly any SCADA systems running. This kind of information is often surprisingly easily available.
2. The intruder accesses the facility by some pretext (cleaning crew, commercial representative, etc.).
3. At a suitable opportunity, using a keyboard injection device, the intruder installs a DNS tunnelling client to an operator console machine with an open session in the private network.
4. Now the intruder has full transparent access to the infected machine. There's a good chance that the account used to run the operator console has admin privileges and can be used to install low-lever network sniffers, password crackers, fake routers, etc.

CONCLUSIONS

We have presented results of an information security assessment carried out on some of CERN access and safety systems. A number of equipment were found vulnerable and subsequently patched or otherwise secured whenever possible following the best practices in the industry. We also had some surprises in learning about recent intrusion techniques and devices, which make the life of an intruder a lot easier than it needs to be. Lessons learned include ways to mitigate the risk of intrusions:

- Strict access controls to sensitive areas to know who enters and when.
- Devices in locked racks away from manipulation.
- Disabling of any unnecessary network protocols.
- Updated firewalls and monitoring of suspect traffic.
- Defense-in-depth: keep even isolated devices updated and patched as much as possible.

REFERENCES

[1] https://en.wikipedia.org/wiki/Stuxnet
[2] B.J. and B. Andrew, "Hacking with Kali", Elsevier, (2013).
[3] http://www.kali.org
[4] https://www.owasp.org
[5] http://libnodave.sourceforge.net
[6] http://sourceforge.net/projects/lynis
[7] http://www.openvas.org
[8] http://code.kryo.se/iodine/
[9] https://en.wikipedia.org/wiki/IPv6
[10] http://usbrubberducky.com

ISBN 978-3-95450-148-9