Penetration Testing of OPC as part of Process Control Systems

Maria B. Line[1], Martin G. Jaatun[1],
Zi Bin Cheah[2], A. B. M. Omar Faruk[2], Håvard H. Garnes[2], Petter Wedum[2]

[1]SINTEF ICT, N-7465 Trondheim, NORWAY
{maria.b.line, martin.g.jaatun}@sintef.no
[2]SISLAB
{redteam,blueteam}@sislab.no

Abstract. We have performed penetration testing on OPC, which is a central component in process control systems on oil installations. We have shown how a malicious user with different privileges – outside the network, access to the signalling path and physical access to the OPC server – can fairly easily compromise the integrity, availability and confidentiality of the system. Our tentative tests demonstrate that full-scale penetration testing of process control systems in offshore installations is necessary in order to sensitise the oil and gas industry to the evolving threats.

Keywords: Information Security, Process Control, Penetration Testing, OPC

1 Introduction

Process control systems (PCS) are used to support and control a specific industrial process. The setting for the kind of system discussed in this paper is oil production on offshore installations. Traditionally, hardware and software components in such systems were proprietary; designed for use in a specific context. The current trend is heading towards commercial off-the-shelf technologies because of the goal of integrated operations, which means extensive cooperation between onshore and offshore staff, and the possibility of controlling installations completely from onshore.

This transition results in exposure to an entirely new set of security threats. Many components and applications in use were developed before this transition without any thought of making them intrusion-proof, because the access used to be more limited, both in the physical and the logical way.

With respect to security, the ideal solution would be to build all parts of a process control system from scratch, taking all new threats into account. In reality this is not achievable; economically it would be a disaster for the operators, and a set of threats relevant today may change during the period of development. Also, there is the fact that new security flaws are usually introduced when developing new systems. Hence the brand new systems would not be perfect anyway

The more realistic solution is to add security functionality or security measures where they are needed. In order to identify where the needs are, several components and applications already in use must be analyzed in detail. This paper presents how a test network was set up to simulate a process control system, and how intrusion tests were performed towards OPC. The concept of OPC is described in the following section.

1.1 OPC – OLE for Process Control

OLE for Process Control (OPC) [1] is a mechanism for exchanging process control data among numerous data sources in process control system. OLE stands for Object Linking and Embedding. OPC is implemented as a client-server-architecture. The OPC server aggregates data from available control units in the system. It is capable of reading and writing values from/to these units, and offers mechanisms for interaction with these values to an OPC client. OPC is widely used in process control systems due to its stability and reliability. Also, it allows transmission of double precision real numbers, which other protocols usually do not allow without rewriting [2].

The OPC interface is defined on Microsoft's OLE/COM (where COM stands for Component Object Model) [3], and most OPC servers run on MS Windows servers, even though there exist some implementations for other operating systems, such as OpenSCADA. The OPC interface is implemented as a set of function calls to COM objects defined by the OPC server, and can be accessed by DCOM (Distributed COM) from other computers. Hence, OPC does not implement security, but makes use of the security offered by DCOM.

We have used OPC Tunneller and OPC Simulation Server from Matrikon[1] in our simulated process control system. The reason for choosing Matrikon is that it is a freely available and fully functioning OPC-server for use in a non-production setting. The simulation server is normally used for testing OPC connections and services before production servers are put to use. It is therefore considered to be a suitable alternative for a real OPC server.

1.2 Penetration Testing

Penetration testing is a way of evaluating the security of a system by simulating attacks by a malicious user against the system. Such testing can be performed in different ways; by running known vulnerability scanners like Nessus[2] and Nmap[3], or more as research and testing of new and unknown vulnerabilities.

Nessus is well-suited for testing systems in production, and it is low-cost. It is continuously updated with new modules for newly discovered vulnerabilities, and running Nessus gives a good indication for how secure a system is, without giving a 100% complete result, which is quite impossible to achieve in any ways.

[1] http://www.matrikonopc.com

[2] http://www.nessus.org

[3] http://insecure.org/nmap/

Testing of unknown vulnerabilities requires extensive knowledge of the system in focus and is hence out of the question for our tests this time, as time and resources puts limitations on our work. In the future we might be able to move on to deeper analyses of OPC including testing of unknown vulnerabilities.

1.3 Our Tests in Brief

Our aim is to show how a malicious user with varying privileges – outside the network, access to the signalling path and physical access to the OPC server – can fairly easily compromise the integrity, availability and confidentiality of the system. Resources available to us include a lab where we can set up a test network that simulates a process control system, and software freely available on the Internet.

We define a blue team and a red team, where the blue team is responsible for setting up the test network and detecting intrusions, and the red team will act as malicious hackers and run different sets of exploits/attacks.

1.4 Paper Outline

Section 2 describes how the test network was setup. Section 3 is about vulnerabilities that apply to OPC, which will be exploited. Section 4 describes the performed attacks together with achieved results, while section 5 discusses our findings. We offer concluding remarks and directions for further work in section 6.

2 Test Network Setup

Blue team set up a simulated process control system in a test lab [4]. The topology reflects a typical offshore network inspired by the SeSa method [5]. The network is divided into three subnets, as follows:

- DMZ subnet
- Admin subnet; with an OPC client
- Process Network subnet; with an OPC server

DMZ is an abbreviation for demilitarized zone. In common computer networks, a DMZ contains an organization's services that are available for access from untrusted networks, typically the Internet. Services in a DMZ are usually needed by external hosts in order to communicate successfully with hosts inside the network and vice versa. The purpose of the DMZ is to add an additional layer of security to an organization's network by having services that are frequently accessed housed in this layer.

The Admin layer models a typical process control system setup where hosts in the Admin Network are privileged users that usually have access to the Process Network layer. In order for users in the Admin network to access the Process Layer services, they should have the proper credentials to do so. For example, a manager in the Admin layer might use OPC client software such as a Matrikon OPC Client to access information from an OPC server (which is placed in the Process Layer). In order for the manager to successfully do this, he has to have the proper login/password credentials.

The Process Network layer is the deepest layer of the system and houses the most critical services in an offshore network. If an attacker manages to take control over this network, he or she has succeeded in compromising the entire network.

The specifications of the computers used in the network are listed in Table 1 below, and the topology is illustrated in Fig. 1.

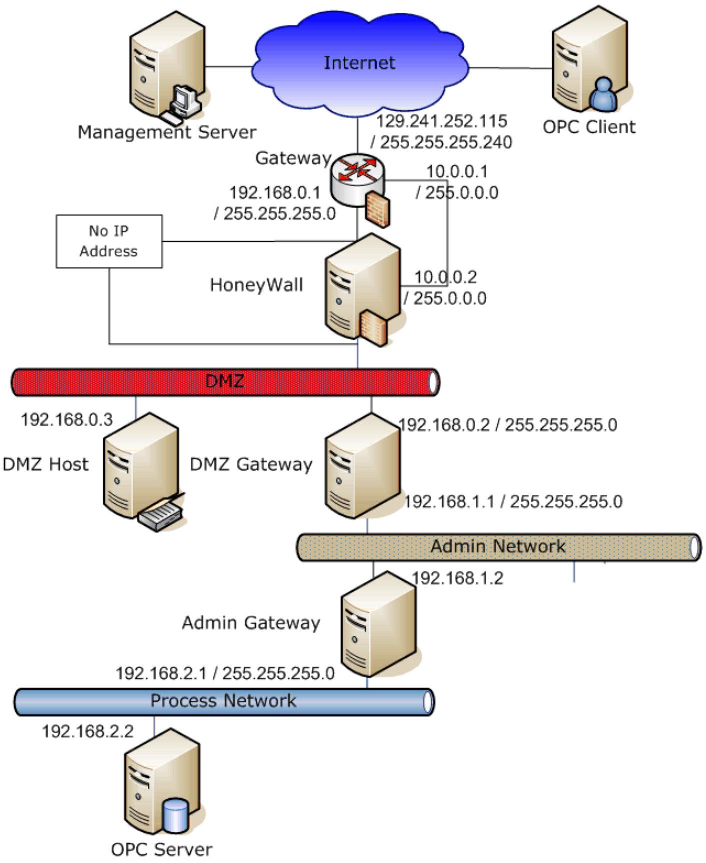

Fig. 1. The network topology as set up by the blue team.

Table 1. Computer hardware specification.

Host Name	Operating System	# NICs
Gateway	Linux Fedora Core 6	3
Honeywall	Linux Fedora Core 6	3
DMZ Gateway	Linux Fedora Core 6	2
DMZ Host	Linux Fedora Core 6	1
Admin Gateway	Linux Fedora Core 6	2
OPC Server	Windows XP SP2	1

The honeywall and the router are for administrative and monitoring purposes. The OPC client is placed on the Admin subnet, while the OPC server resides on the process network subnet.

2.1 The Honeywall

A honeypot is a host set up to detect, monitor and/or trap attackers, and a honeywall is a host that is usually placed between honeypot(s) and non-honeypot components in a network. It is regarded as a bridge between honeypots and regular networks segments. The two main tasks for a honeywall are traffic monitoring and intrusion prevention.

Our honeywall has an IP address on only one interface, and we use the gateway to block anyone trying to access it. This interface is used as a remote management interface, while the other two interfaces (without IP addresses) act as the bridge. The bridge operates like a hub; passing traffic along from one interface to the next. As a honeywall, it also reads, copies, and analyzes all the passing traffic.

A honeywall should be “non-existent” from a logical view. No one will know there is a honeywall host present unless that person enters the laboratory and sees our network.

2.2 The Gateway

All traffic that enters or exits the network has to go through the gateway. If the gateway crashes, it means that the connection between our test network and the Internet is broken. In addition, the gateway performs firewall filtering and Network Address Translation (NAT). The purposes of the main gateway are:

- Internet entry-exit point
- Firewall packet filtering
- Firewall NAT

Our network provides an open port (21379) for the OPC client to connect to our OPC server using OPC tunnelling. Furthermore, the services SSH (port 22) and HTTPS (port 443) are also open for remote connection. An attacker might wish to exploit such open services and try to attack remotely from the Internet.

We choose to use Iptables[4] as our choice of firewall implementation due to its flexibility. The firewall not only blocks traffic, but shapes traffic, too. Traffic can be denied or allowed based on port numbers, host names and IP addresses. For our case we use the *filter table* and *NAT table* in iptables.

In the *filter table* of iptables, we used ALLOW ALL. Any traffic may come in and out of the network. The only exception is that we have configured our gateway to deny access to the private IP address of 10.0.0.X. The 10.0.0.X is the private network between the gateway and the honeywall; we want this subnet to be logically invisible so that no one can access this subnet. Usually, ICMP (Internet Control Message Protocol) is used to ping a host to determine if it is alive. We have configured the firewall to reply with "`icmp port unreacheable`" when anyone tries to ping the 10.0.0.X subnet. This will let others think that the subnet does not exist. The only way to access this subnet is via HTTPS from the Internet. In this case, the outside host will not know that the subnet exists because NAT has already translated the addressing.

The *NAT table* in iptables allows us to forward traffic to certain ports to ports that we decide. Since we only have one public IP address, NAT directs the traffic to the appropriate destinations. In our configuration we forward traffic meant for the ports 3389 (RDP)[5], 21379 (OPC Tunneller) and 135 (RPC)[6] to the OPC server, and 443 (HTTPS)[7] to the honeywall.

2.3 Intrusion Prevention

We do not want an attacker that has successfully compromised our network to launch attacks from our network against others. However, we do not want to deny all outbound traffic, only malicious traffic. Outbound informational queries, such as ICMP ping, finger query, or a simple HTTP GET command should be allowed. To realize this we deploy a special snort function called snort_inline[8] that is integrated in the Honeywall. This function lets us limit outbound, but welcome inbound, malicious traffic.

One way to limit outbound traffic is to drop all outbound malicious packets; another is to limit the amount of outbound traffic per time unit – we decided to do both. Limiting outbound traffic is especially important regarding preventing Denial-of-Service attacks from our network towards others. We do not set the outbound traffic to zero, as this will arouse the attacker's suspicion as to why there is no outbound connections allowed.

[4] http://www.netfilter.org

[5] 3389: Needed for remote desktop management of the OPC server.

[6] 21379 and 135: Needed to run the OPC server.

[7] 443: Needed for configuration of the honeywall via a browser.

[8] http://snort-inline.sourceforge.net

2.4 OPC

By default, Matrikon OPC Tunneller does not use encryption or authentication to connect with another OPC Tunneller. We can, however, use a shared secret to encrypt and authenticate communication between client and server side OPC Tunnellers. This option was not employed in our tests. Throughout the penetration testing we have assumed that the IP-address of the OPC server is known to the attackers (red team).

2.5 Red team's equipment

Red team set up a Matrikon OPC Simulation Server with Windows XP SP2 and all available updates. The server was set up as described in the guide published by CERN [6]. The password for this server was known by the red team, as the login service was not to be tested. This server is in addition to the blue team simulation OPC-network.

Besides this test server, red team operated with two regular computers and a switch that connected their units with "the Internet", where "the Internet" is represented by a switched Ethernet LAN.

3 Vulnerabilities in OPC

In order to map vulnerabilities related to OPC, we have to consider vulnerabilities related to DCOM and RPC. This is because OPC is based on DCOM, which uses RPC, as mentioned in the Introduction.

Frequently used vulnerability databases like the National Vulnerability Database (NVD)[9], the Open Source Vulnerability Database (OSVD)[10], and the database provided by United States Computer Emergency Readiness Team (US-CERT)[11] reveal 555 vulnerabilities in RPC. 71 vulnerabilities related to DCOM are listed, and 40 OPC-specific vulnerabilities are reported. Here are some examples of vulnerabilities:

- NETxAutomation NETxEIB OPC Server fails to properly validate OPC server handles
- MS Windows RPC DCOM Interface Overflow
- MS Windows DCOM RPC Object Identity Information Disclosure
- MS Windows DNS RPC buffer overflow
- MS Windows RPC service vulnerable to denial of service
- MS Windows RPCSS Service contains heap overflow in DCOM request filename handling
- MS Windows 2000 RPC Authentication Unspecified Information Disclosure

[9] http://nvd.nist.gov/

[10] http://osvdb.org/

[11] http://www.kb.cert.org/vuls/

These vulnerabilities open for buffer overflow, Denial-of-Service, and information disclosure; especially in the case of RPC. Carter et al [7] describe OPC-specific vulnerabilities in detail, and the main points of their discussion are summarized below.

Lack of Authentication in OPC Server Browser: Configuration guidance from many vendors recommends allowing remote Anonymous Login so that OPCEnum will work when DCOM Authentication is sent to "None". If a buffer overflow is discovered in the OPC Server Browser code, the result could be arbitrary code execution or Denial-of-Service attack against any computer running the OPC Server Browser. Fortunately, such an overflow has not been discovered yet.

Lack of Integrity in OPC Communications: The default DCOM settings do not provide message integrity for OPC communication. If the underlying network is compromised and the attacker can sniff and insert traffic, it is likely that rogue messages could be injected once the client and server are authenticated during the initial connection establishment. A number of "Man-in-the-Middle" tools and techniques are available, and it is likely that these could be modified or enhanced to conduct attacks against OPC communication.

Lack of Confidentiality in OPC Traffic: Although DCOM supports message encryption, most OPC vendors do not recommend enabling Packet Privacy for their OPC Server or the OPC Server Browser. Some vendors recommend VPN tunnelling as a means of providing secure remote access. Matrikon uses client- and server-side tunnelling component with encryption for this purpose.

In a different article, Lluis Mora [8] discusses the vulnerabilities of OPC servers. These include attacks using invalid server handle, invalid or crafted configuration file, resource starvation, etc. They have developed the tool "OPC Security Tester"/OPCTester[12] for testing these vulnerabilities.

4 Exploits and Results

Red team's task was to act as attackers towards the test network and the OPC server specifically. Their initial knowledge included the IP address of the OPC server. In the following, the different types of mappings and attacks they performed are described, together with the results they achieved. More details about their work can be found in a separate report [9].

4.1 Initial Network Mapping – Scanning and Probing the Network

Many tools are available for gathering information about networks, operating systems and services offered by a network. Network administrators use such tools to manage network services, monitor hosts, service uptime etc. But as such tools are freely available; attackers can also use them for their own purposes. Nmap and Nessus can be used for scanning a process control system. Nessus also supports testing a list of

[12] http://www.neutralbit.com/en/rd/opctest/

vulnerabilities on a specific remote host. This can be thought of as the first step for any attacker in order to explore the list of services and ports exposed by a network.

The introduction of a packet sniffer in the network clearly showed that in the case of setting the DCOM security level for the OPC server to "connect", all the OPC traffic is sent over the network in plain text, both from our test server and from the simulation network. Without any knowledge of the packet layout, we could still easily read out string values of the packets. Closer inspection and cross-checking with a proper OPC client lead us to also be able to identify other types of values in the packets, especially numerical values. This experience indicates that CERN does not value confidentiality of OPC data, as they have recommended this configuration setting.

Further examinations of the network were done with the network mapper Nmap and the vulnerability scanner Nessus. Neither of these gave any indications of easily exploitable vulnerabilities in either our test server or in the test network. The information that was obtained was correct from both scanners on the operating system and other information known for our test server, and indicated Linux Fedora as the operating system running on the front end of the simulation network. However, these tools do not test OPC in particular, but rather the general setup of the servers with their operating systems and services.

OPCTester performs automated tests of around 30 known vulnerabilities and access faults. We ran OPCTester to scan both the test server and the simulation network and did not get any useful results for either of the OPC servers. Run locally on our test server, OPCTester showed all available tests as passed without known vulnerabilities.

To sum up, network mapping did not yield any information on exploitable technical issues on the simulation network nor on our test server. Network sniffing did clearly show that the confidentiality of a standard set up OPC-server is void.

4.2 Entering the Signalling Path

As “the Internet” in our setup is a switched Ethernet, we utilized the well known technique ARP-spoofing in order to be able to realistically perform network sniffing. We used the tool ARPspoof for this purpose, which is included in the Linux software package dsniff[13].

With ARPspoof we continuously sent out gratuitous ARP-packets to two given IP addresses informing them that the other IP-address is located at the attacker’s MAC address. This way all traffic between the two hosts is switched onto the attacker’s port on the switch, effectively putting us (the attacker) in the middle.

4.3 Packet Sniffing

A network packet analyzer, e.g. Wireshark[14], can be used to capture network packets and analyze them for future attacks. In the case of OPC traffic communication

[13] http://www.monkey.org/~dugsong/dsniff/

[14] http://www.wireshark.org/

between the server and client side tunnellers is by default not encrypted. Information about OPC server and groups and items added or updated in OPC server can hence be read. A Man-in-the-Middle attack can be used due to the lack of access control mechanisms. An ARP spoofing attack allows an attacker to spoof the MAC address to sniff, modify, redirect or stop the traffic to the IP address of the target system.

By entering the signalling path we were able to read the packets sent between the client and the server. By monitoring the authentication process of the client we were able to read in clear text the user name and machine name of the client and the machine name of the server. The client response to the server challenge includes both the NTLM version 1 response and the LM response. There is no need for both of the responses which only leads to further reduction of the security. Both of these two protocols use DES, which is known to be a weak encryption algorithm. NTLM version 2 has been accepted as a de-facto internet standard as of 2000 [10], is widely used, and is deemed much more secure than NTLM version 1.

4.4 Denial-of-Service Attacks

We used SYN flooding and ARP spoofing to launch a Denial-of-Service attack. TCP uses three-way handshake (SYN, SYN-ACK, ACK) to establish a session. In the case of SYN flooding the attacker sends several packets, but does not send the last ACK back to the server, or the attacker spoofs the source IP address of the SYN message, server sends SYN-ACK to the false IP address and never receives the last ACK. ARP spoofing can stop the traffic to the spoofed target system. With the previously mentioned middleman-status we performed a Denial-of-Service attack by simply dropping all the packets destined for both the spoofed hosts, and thereby acting as a black hole in the signal path. This attack totally destroyed the communication between the test server and the client as expected.

A second successful Denial-of-Service attack was performed as a SYN-flood attack run by a single attacker. On the test server, this attack not only disabled incoming communication from a client, but also slowed down the system so much that the server was practically unresponsive during the attack. The effect of the attack lasted until about one minute after the attack was called off by the attacker. As this attack used fake source addresses in the packets, this attack can potentially be very difficult to distinguish from genuine heavy load.

As seen, both a black-hole method and a SYN-flood method of attacks were able to destroy the availability of the test system; the SYN-flood even resulted in lost local availability.

4.5 Man-in-the-Middle Attack

A tool was written to do string replacement and forwarding of packets in the ARP spoofed network setup between the OPC client and the OPC test server. As the recommended setup of DCOM did not seem to contain any packet integrity mechanisms, we expected this attack to be able to compromise the integrity of the entire OPC network setup.

The tool we wrote simply replaced the string "Simulation" as sent from the OPC server A with the string "Real" and forwarded the modified packets to the OPC client B. Likewise the tool changed the string "Real" in the packets from the client to the server into the string "Simulation" and again forwarded these packets. Our tool was written as a proof-of-concept in the sense that the strings replaced were arbitrary. Any value and/or string can be replaced in the packets in the same way. The same tool can also be easily modified to shield the client from the server and in effect hijack the session. The resulting setup is described in Fig. 2.

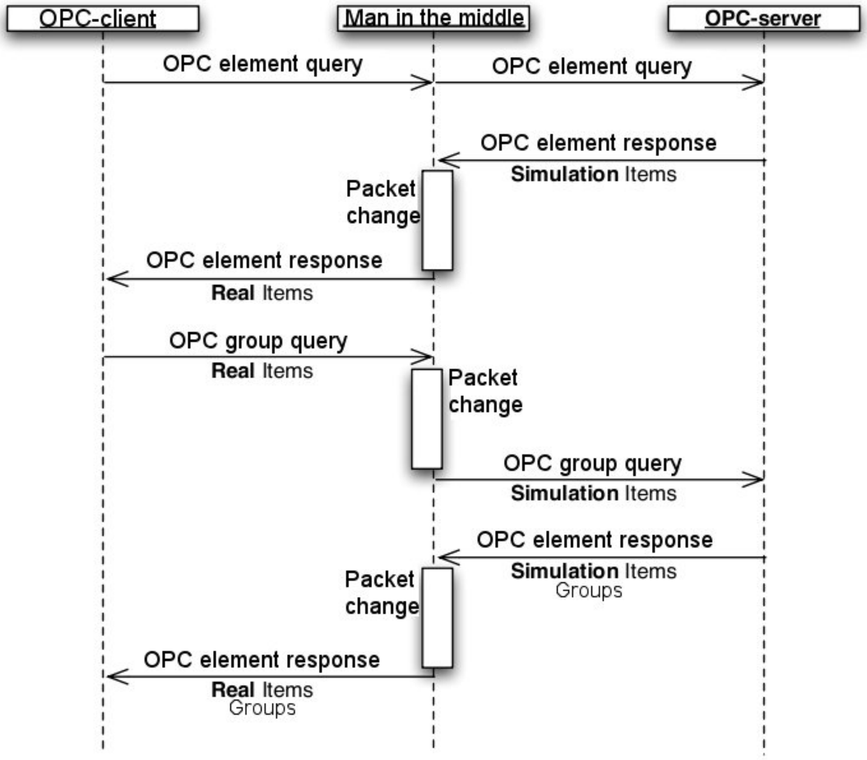

Fig. 2. Man-in-the-middle attack on OPC; packets between server and client are modified.

Table 2. Excerpt from printouts from OPC client from two different OPC sessions.

Without packet modification	**Man-in-the-middle attack**
2007-12-05 14:00:05,061 [main]	2007-12-05 14:02:25,345 [main]
DEBUG org.openscada.opc.lib.da. browser.BaseBrowser – Browsing with a batch size of 10	DEBUG org.openscada.opc.lib.da. browser.BaseBrowser – Browsing with a batch size of 10
Leaf: Clients [Clients]	Leaf: Clients [Clients]
Branch: *Simulation* Items	Branch: *Real* Items
Branch: Bucket Brigade	Branch: Bucket Brigade
Leaf: ArrayOfReal8 [Bucket Brigade.ArrayOfReal8]	Leaf: ArrayOfReal8 [Bucket Brigade.ArrayOfReal8]
Leaf: ArrayOfString [Bucket Brigade.ArrayOfString]	Leaf: ArrayOfString [Bucket Brigade.ArrayOfString]

As we can see from the sampled output in **Table 2**, we have a situation that to the eye looks genuine, but in reality is completely false.

With the Man-in-the-Middle attack we were able to gain complete control of the system with the user privileges and access rights of the user in session. We were able to read, drop, change, and create packets as we saw fit in both directions, to the server and to the client.

4.6 Configuration Errors

In the process of setting up the simulation network, the blue team installed an OPC Tunneller. As we connected to this OPC Tunneller, there were no access control mechanisms in place se we had complete access to the OPC server in the simulation network. The blue team also gave us instructions on how to set up access control on the OPC client, but these instructions led to the setup screen for the Windows XP DCOM run-as-user configuration management. We assume the instructions for the server setup was misunderstood such that the blue team in reality had set their OPC Tunneller to run as a specific user on the server rather than setting access control credentials needed for a client to log onto it.

We can see that a small misunderstanding in the setup of the server led the red team, and in reality anyone, to take complete control of the simulation OPC server while the blue team was certain that access control was in place.

5 Discussion

Process control networks in the oil and gas industry have traditionally had a very strong focus on safety, but safety-related risks are quite different from security-related risks. One way of differentiating between security and safety is to say that safety consists of protecting the world from the system, while security is protecting the system from the world [11]. Another aspect is that safety-related incidents normally have clearly defined statistical properties that relate to empirically established values for e.g. Mean Time Between Failures (MTBF) for components and systems. The same thing cannot be said for security-related incidents, since they predominately involve a human attacker – and we have yet to develop a useful statistical model for *Homo Sapiens*.

In the early days of the internet, there was a Usenet newsgroup known as alt.folklore.urban[15], devoted to the topic of urban legends. A recurring joke (or mantra) in this newsgroup was: "It could have happened, so it must be true." Ludicrous as this statement may be in "real life," this is exactly how we must think in the realm of computer security. The professional paranoia this reflects also testifies to a distinct difference in mindset compared to the safety domain.

Our main contribution in this paper is to highlight the "could have happened" aspect of computer security in the Integrated Operations context. We have no empirical evidence that the specific attacks in our laboratory tests have been successfully wielded in a real PCS environment, although there are certainly news reports of other attacks against PCS's in the past [12]. In our contact with representatives from the Norwegian oil & gas industry, we have been confronted with the sentiment "We don't care what they say they can do over in the States; unless you can show us something that is directly applicable to our setup, we're not interested." This tells us that there is a need for full-scale penetration testing activities – and our tests indicate that such activities would yield interesting results.

The most conclusive security breach that we successfully performed was achieved due to configuration error, which may at first glance seem to be an insignificant result. However, it is clear that the human factor plays a major role in the defence of systems as well as in their attack, and the safety concept of "fail safe" should clearly be applied to security settings,

It may be argued that also the other attacks successfully implemented in our tests have limited value, since they all require physical access to the communications medium on some level, and these networks are all protected by multiple layers of firewalls and other protective mechanisms [5]. However, previous breaches tell us that not only is the insider threat an important factor in PCS environments [12], but the increasing integration of offshore and land-based networks also mean that the attack surface is increasing. This, in turn, implies that it is important to follow the doctrine of defence in depth, ensuring that there are sufficient (and independent) security mechanisms every step of the way from the external barrier to the heart of the PCS. Clearly, a protocol that uses outdated cryptography for authentication, and transmits data in plain text is not a good match for our doctrine. It is our opinion that the OPC implementation should implement NTLM version 2 or other more secure authentication protocols like Kerberos.

[15] It's still out there – see http://tafkac.org/

6 Conclusion

We have shown that confidentiality in OPC networks is non-existent in the default setup as recommended by CERN. Furthermore, the authentication process of the client reveals confidential information in clear text and gives a weak encryption of the client password. We have seen that DoS attacks can easily be accomplished, not only in making the server unavailable over the network, but also leading to denial of local control over the server. We have demonstrated a to the eye perfect breach of the integrity of the OPC server as a consequence of lacking DCOM packet integrity measures in the setup recommended by CERN, and we have demonstrated how fragile an OPC network is to configuration errors.

Acknowledgements

The research presented in this paper was performed as part of project assignments at the Norwegian University of Science and Technology (NTNU).

References

[1] "OPC Overview 1.0," OPC Foundation 2008-01-17 1998, http://www.opcfoundation.org/Downloads.aspx?CM=1&CN=KEY&CI=282

[2] "Understanding OPC and How it is deployed," Byres Research 2008-01-17 2007, http://csrp.inl.gov/Recommended_Practices.html.

[3] "DCOM Technical Overview," Microsoft Developer Network 2008-01-17 1996, http://msdn2.microsoft.com/en-us/library/ms809340.aspx.

[4] Z. b. Cheah and A. B. M. O. Faruk, "Identifying and Responding to External Threats in a PCS Network," Norwegian University of Science and Technology Project Assignment, Trondheim 2007-12-19 2007.

[5] T. O. Grøtan, et al., "The SeSa Method for Assessing Secure Remote Access to Safety Instrumented Systems," SINTEF Report A1626, Trondheim June 2007, http://www.sintef.no/content/page1____16321.aspx.

[6] M. B. J.-P. Puget and R. Barillere, "IT-CO recommended DCOM settings for OPC," CERN, Geneva 2005.

[7] J. Carter, et al., "OPC Security," Digital Bond 2007,

[8] L. Mora, "OPC Server Security Considerations," presented at SCADA Security Scientific Symposium 2007, Miami, FL, 2007.

[9] H. H. Garnes and P. Wedum, "Innbruddstesting på prosesskontrollsystemer på oljeplattform," Norwegian University of Technology and Science Project Assignment, Trondheim 2007-12-19 2007.

[10] G. Zorn, "RFC 2759: Microsoft PPP CHAP Extensions, Version 2," The Internet Society 2000,

[11] M. B. Line, et al., "Safety vs security?," presented at Eighth International Conference on Probabilistic Safety Assessment and Management, New Orleans, USA, 2006.

[12] "GAO-07-1036 Critical Infrastructure Protection: Multiple Efforts to Secure Control Systems Are Under Way, but Challenges Remain," United States Government Accountability Office 2007, http://www.gao.gov/htext/d071036.html.