



**CYBSEC** SA  
Security Systems

# SAP Penetration Testing

Mariano Nuñez Di Croce  
[mnunez@cybsec.com](mailto:mnunez@cybsec.com)

April 16, 2009  
Black Hat Europe 09 Briefings and Training



## Who is CYBSEC ?

- Provides Information Security services **since 1996**.
- More than 300 customers, located in LatinAmerica, USA and **Europe**.
- **Wide range of services**: Strategic Management, Operation Management, Control Management, Incident Management, PCI Services, **SAP Security**.

## Who am I?

- **Senior Security Researcher / Project Leader** at CYBSEC.
- Degree in Computer Systems Engineering.
- Originally devoted to **Penetration Testing** and **Vulnerability Research**.
- Discovered **vulnerabilities** in Microsoft, Oracle, SAP, Watchfire, ...
- **Speaker/Trainer** at Black Hat, Hack.lu, DeepSec, Ekoparty, Sec-T, CIBSI, ...
- Developed **sapyto**, the first **SAP Penetration Testing Framework**.
- CYBSEC's "**SAP (In)Security Training**" instructor.



## Agenda

- Introduction to the SAP World
- Why SAP Penetration Testing?
- PenTest Setup
- SAP PenTesting
  - Discovery Phase
  - Exploration Phase
  - Vulnerability Assessment Phase
  - Exploitation Phase
- Conclusions



# Introduction to the SAP World

*Basic concepts for deep knowledge*



## So... what is SAP?

**SAP** (*Systems, Applications and Products in Data Processing*) is a german company devoted to the **development of business solutions**.

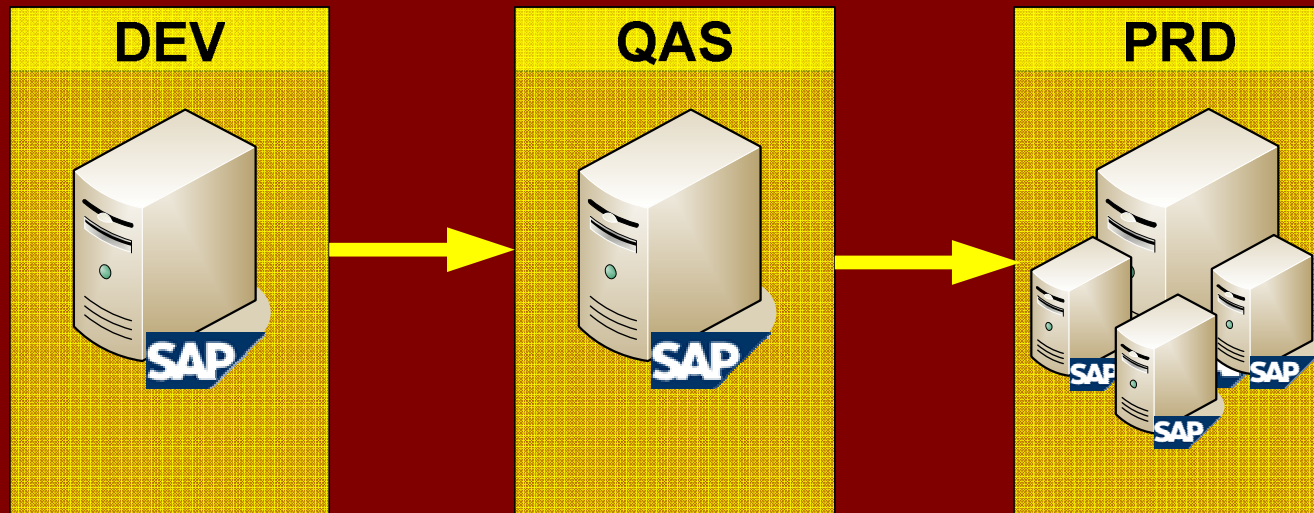
- More than 41.600 customers in more than 120 countries.
- More than **121.000 SAP implementations** around the globe.
- Third biggest independent software vendor (ISV).
  
- Provides **different solutions**:  
**CRM, ERP, PLM, SCM, SRM, GRC, Business One, ...**
  
- The ERP solution is composed of **different functional modules** (FI, CO, SD, HR, MM, etc) that implements organization **business processes**.
- Modules are **linked** together, integrated by the **Netweaver** platform.
- SAP runs on multiple Operating Systems and Databases.



## SAP Basic Concepts

### ▪ Instance & System

- An **instance** is an administrative entity which groups related components of an SAP system, providing one or more **services**.
- Systems are identified by **SAP System ID (SID)**.
- System (instance) parametrization is done in **Profiles**.





## SAP Basic Concepts

### ▪ Client

- Legally and organizationally **independent unit** in an SAP system (company group, business unit, corporation).
- Identified by a **three-digit number**.
- **Default clients**: 000, 001 and 066.

### ▪ Transaction

- Related sequence of steps (**dialog steps**) aimed to perform an operation in the SAP database.
- Identified by a **transaction code** (ej: SU01, SE16, FK01, PA20,...)

### ▪ Authorizations

- Users are assigned roles/profiles which contain authorizations.



## SAP Basic Concepts

### ▪ ABAP

- ABAP is the SAP high-level programming language used to develop business applications.

### ▪ Reports / Programs

- ABAP programs that receive user input and produce a report in the form of an interactive list.

### ▪ Function Modules

- Independent ABAP modules. Can be called locally or **remotely**.

### ▪ The RFC (Remote Function Call) Interface

- Used to call function modules **on remote systems**.





## Some “Low-level” Knowledge

- **SAP\_ALL** profile = **SAP GOD**.
- **Many other profiles** may enable a user become a **god too**.
- Each SAP System uses its own Database.
- SAP processes run under the **<sid>adm** or **SAPService<SID>** user accounts.
- Direct **access to the Database** means complete **SAP compromise!**
- Connections between systems often based on **Trust Relationships** (r\* services).
- Many customer's **interfaces** are implemented through **FTP** (cleartext, usually weak passwords).



# Why SAP Penetration Testing?

*Or why you and your CFO should care*



## Why do you Need an SAP Penetration Test?

But we haven't secured the systems yet...you know, there is something called "Security"

The new SAP system must be running on April 21<sup>st</sup>, no excuses.

Security? Hmm...is it French?

But we should take care of User authorizations to

Whatever, I don't care... business \*must\* go on!

@#-\*!#&\$%!!

give everyone full access for three months, then we'll lock it down





## Why do you Need an SAP Penetration Test? (cont.)

What the CFO doesn't realize:

Alert 

**Weak SAP Security configuration can definitely result in Business Frauds!**

What the CSO doesn't realize:

Alert 

**SAP Security is much (\*much\*) more than User roles and authorizations!**



## Why do you Need an SAP Penetration Test? (Wrappin' up)

- Security configurations of SAP systems are usually left by default.
- By default, many configurations are not secure.
- Conclusion: Many SAP systems out there are not secure!
  
- **Is yours secure?** A Penetration Test to these systems will help you know how your SAP implementation can be attacked and which is the real impact of this.
  
- It will help you discover the weaknesses, secure them, and increase the security level of your systems (a.k.a decreasing fraud risk).
  
- In this talk, we'll see some of the activities that make up the different phases of an SAP Penetration Testing (no way of covering them all).



# PenTest Setup

*Gentlemen, start your engines...*



## Preparation

- What do you need? **The OpenSource “Shopping” List**
  - sapyto
  - nmap
  - r\* tools (rsh, rlogin, rexec)
  - SQL client tools
  - NFS client tools
  - SMB client & security tools
  - BurpSuite / w3af
  - Nessus
  - john (patched)
  - hydra
- Try to get as much information as possible about **target platforms**, usage and **policies** before starting the assessment.
- Remember that **everything that breaks** while you are pentesting **\*will\* be your fault** (even if someone breaks his leg).

## sapyto

- First **SAP Penetration Testing Framework**.
- Support for activities in **all phases of the pentest**.
- Open-source (and free).
- **Plugin based**.
- Developed in Python and C.
- Version 0.93 released at **Blackhat Europe 07** (nice to see you again!)





## Back in December: sapyto v0.98-Public\_Edition



- Core and architecture **fully re-built**.
- Based on targets & **connectors**.
- The SAPRFC\* connectors and the RFCSDK.
- Plugins are now **categorized** in Discovery, Audit and Exploit.
  - **Discovery** plugins find new targets.
  - **Audit** plugins carry out the vulnerability assessments.
  - **Exploit** plugins are used as proof of concepts for discovered vulns.
- **sapytoAgents** deployment.
- New plugins for auditing SAProuters, find clients, bruteforcing, ...

*Free download: <http://www.cybsec.com/EN/research/sapyto.php>*

## Now: sapyto v0.99-Public\_Edition !



- **Windows** support (!)
- **Automatic discovery and configuration** of target SAP systems.
- **New plugins:**
  - getServers – Find new SAP instances.
  - oraAuth – Take advantage of weak SAP/Oracle implementations.
  - oraAgent – Remote Interactive SQL shell with elevated privileges.
  - and more...

## Future

- **Version 1.00** is just around the corner...
- Is your mouse bored? The **sapyto GUI** is under heavy development!



# Performing an SAP PenTest

*Showtime*



# Discovery Phase

*Finding SAP targets*



## Discovering SAP Systems and Applications (Targets)

- Available Options:
  - Traffic sniffing.
  - **SAP portscanning.**
  - Checking SAPGUI configurations.
  
- SAP Systems use a “**fixed**” range of ports.
- Most ports follows the **PREFIX + SYS. NUMBER** format.
- **Common ports:** 32XX, 33XX, 36XX, 39XX, 3299, 81XX, ...
  
- **Nmap:** Watch Timings (-T3) and don't use version detection.
  
- New sapyto is shipped with **automatic discovery of SAP systems and configuration** of targets & connectors for auditing!



# Exploration Phase

*Getting as much information as possible*



## Getting Information from SAP Application Servers

- The **RFC\_SYSTEM\_INFO** function module returns information about remote SAP Application Servers (implemented in sapyto's sapinfo plugin)
- Can be **called remotely** (and anonymously) by default.

```
sapinfo(target#0) {  
Remote System Information:  
  RFC Log Version: 011  
  Release Status of SAP System: 700  
  Kernel Release: 700  
  Operating System: Linux  
  Database Host: sap101  
  Central Database System: ORACLE  
  Integer Format: Little Endian  
  Daylighth Saving Time:  
  Float Type Format: IEEE  
  Hostame: sap101  
  IP Address: 192.168.3.4  
  System ID: TL1  
  RFC Destination: sap101_TL1_00  
  Timezone: -18000 (diff from UTC in seconds)  
  Character Set: 4103  
  Machine ID: 390
```



## Getting Information from SAP Application Servers

- The **RFC SYSTEM INFO** function module returns information about remote SAP Application Servers (implemented in sapyto's sapinfo plugin)
- Can be **called remotely** (and anonymously) by default.

```
sapinfo (target#0) {  
  Remote System Information:
```

### Protection / Countermeasure

- **Restrict connections to the Gateway at the network level.**
- **For more information, refer to SAP Note 931252.**

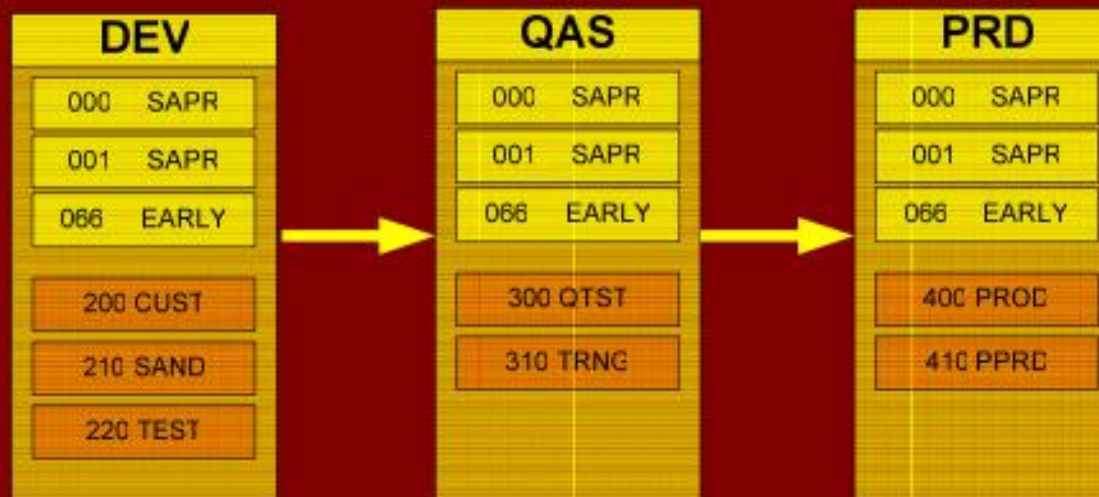
```
Float Type Format: IEEE  
Hostame: sap101  
IP Address: 192.168.3.4  
System ID: TL1  
RFC Destination: sap101_TL1_00  
Timezone: -18000 (diff from UTC in seconds)  
Character Set: 4103  
Machine ID: 390
```





## Finding Available Clients

- Users are client-dependent.
- Default clients: 000, 001, 066.



```
getClients(target#0) {  
    Client 000 is available.  
    Client 001 is available.  
    Client 066 is available.  
    Client 101 is available.  
    Client 200 is available.  
} res: Ok
```





## Analyzing Shared Resources

- The **Common Transport Directory (CTD)** is the directory where changes (**transports**) are exported to and imported from in an SAP landscape.
- This directory **must be shared** for all systems in the landscape.
- In most cases, **kernel files and profiles are shared to dialog instances.**

```
$ showmount -e sapserver  
  
/export/usr/sap/trans (everyone)  
/export/sapmnt/NP1 (everyone)  
/export/informix/NP1 (everyone)  
/export/interfacesNP1 (everyone)  
/export/interfsrcNP1 (everyone)
```





## Analyzing Shared Resources

- The Common Transport Directory (CTD) is the directory where changes (transports) are exported to and imported from in an SAP landscape.
- This directory is shared across all SAP systems.
- In most cases, the CTD is located in the following paths:

### Protection / Countermeasure

**Restrict access to shared resources, allowing connections only from SAP related systems and users.**

```
/export/usr/sap/trans (everyone)
/export/sapmnt/NP1 (everyone)
/export/informix/NP1 (everyone)
/export/interfacesNP1 (everyone)
/export/interfsrsrcNP1 (everyone)
```



# Vulnerability Assessment Phase

*Analyzing the discovered components*



## SAP Default Users


- There is **public information** regarding the existence of **default SAP user accounts**.
- Many of these accounts are configured with **high privileged profiles**.

User ID	Description	Clients	Password
SAP*	Super user	000,001, 066 <b>new clients</b>	06071992 PASS
DDIC	ABAP Dictionary super user	000,001	19920706
EARLYWATCH	User for the EarlyWatch Service	066	SUPPORT
SAPCPIC	Communication User	000, 001	ADMIN



## SAP Default Users

- There is **public information** regarding the existence of **default SAP user accounts**.
- Many of these accounts are configured with **high privileged profiles**.

User ID	<b>Protection / Countermeasure</b> 		
SAP*	<ul style="list-style-type: none"><li>▪ <b>Default users must be secured.</b></li><li>▪ <b>SAP* should be deactivated.</b></li><li>▪ <b>Use report RSUSR003 to check the status of default users.</b></li></ul>		
DDIC	EarlyWatch Service		
EARLYWA			
SAPCPIC	Communication User	000, 001	ADMIN



## SAP User Account Bruteforcing

- Usernames are up to **12 characters long**.
- As part of the PenTest, you can try **guessing/cracking user credentials**.

	Old Passwords ( $\leq 6.40$ )	New Passwords ( $> 6.40$ )
Max. Length	8	40
Case	Insensitive	Sensitive

- **WARNING!** User **locking** is implemented! (usually, between 3-12 tries)
- **Ops!** In versions  $\leq 6.20$ , lock counter is **not incremented** through RFC.
- **sapyto's bruteLogin** plugin can work in different modes:
  - Try default users only and SAP\*:PASS in detected clients.
  - Specific credentials wordlist.
  - Username and Password wordlists.



## Getting Credentials from the Wire – RFC Sniffing

- **RFC** (Remote Function Call) is the most widely used interface in the SAP world.
- In order for a system to connect through RFC, it must **provide login information** for the remote system.
- RFC is clear-text, but you won't be able to **see the password** in the wire...
- **Password is obfuscated!** -> Use **sapyto's getPassword** plugin

```

...
01a0  00 00 00 00 00 00 06 05 14 00 10 5f 22 ea 45 5e  ....._.E^
01b0  22 c5 10 e1 00 00 00 c0 a8 02 8b 05 14 01 30 00  .....0.
01c0  0a 72 66 63 5f 73 65 72 76 65 72 01 30 01 11 00  .rfc_server.0...
01d0  06 42 43 55 53 45 52 01 11 01 17 00 0b 81 bb 89  .BCUSER.....
01e0  62 fc b5 3e 70 07 6e 79 01 17 01 14 00 03 30 30  b..?w.oy.....00
01f0  30 01 14 01 15 00 01 45 01 15 05 01 00 01 01 05  0.....E.....
0200  01 05 02 00 00 05 02 00 0b 00 03 36 34 30 00 0b  .....640..
0210  01 02 00 0e 5a 43 55 53 54 5f 47 45 54 4d 4f 4e  ....ZCUST_GETMON
0220  45 59 01 02 05 14 00 10 5f 22 ea 45 5e 22 c5 10  EY....._.E^"..
0230  e1 00 00 00 c0 a8 02 8b 05 14 02 01 00 09 43 4c  .....CL
0240  49 45 4e 54 5f 49 44 02 01 02 03 00 08 43 55 53  IENT_ID.....CUS
0250  54 30 30 31 00 02 03 ff ff 00 00 ff ff 00 00 01  T001.....
0260  c7 00 00 3e 80  .....>.
    
```

```
for CHAR in CLEAR_TEXT_PASS:
```

```
OBFUSCATED_PASS[i] = CHAR XOR KEY[i]
```





## Getting Credentials from the Wire – RFC Sniffing

- RFC (Remote Function Call) is the most widely used interface in the SAP world.
- In order for a system to connect through RFC, it must provide login information for the remote system.

▪ RFC is cle...  
wire...

▪ Password

### Protection / Countermeasure



Enable SNC, protecting the confidentiality and integrity of the traffic.

```
...
01a0  00 00 00 00 00
01b0  22 c5 10 e1 00
01c0  0a 72 66 63 5f 73 65 72 76 65 72 01 30 01 11 00   .rfc_server.0...
01d0  06 42 43 55 53 45 52 01 11 01 17 00 0b 81 bb 89   .BCUSER.....
01e0  62 fc b5 3e 70 07 6e 79 01 17 01 14 00 03 30 30   b..?w.oy.....00
01f0  30 01 14 01 15 00 01 45 01 15 05 01 00 01 01 05   0.....E.....
0200  01 05 02 00 00 05 02 00 0b 00 03 36 34 30 00 0b   .....640..
0210  01 02 00 0a 5a 43 55 53 54 5f 47 45 54 4d 4f 4e   ....ZCUST_GETMON
0220  45 59 01 02 05 14 00 10 5f 22 ea 45 5e 22 c5 10   NY.....".E^"...
0230  e1 00 00 00 c0 a8 02 8b 05 14 02 01 00 09 43 4c   .....CL
0240  49 45 4e 54 5f 49 44 02 01 02 03 00 08 43 55 53   IEMP_ID.....OUS
0250  54 30 30 31 00 02 03 ff ff 00 00 ff ff 00 00 01   TOOL.....
0260  c7 00 00 3e 80   ...>.
```

```
for CHAR in CLEAR_TEXT_PASS:
    OBFUSCATED_PASS[i] = CHAR XOR KEY[i]
```



# Exploitation Phase

*Getting access and beyond*



## But... why do we need Exploitation anyway?

- Vulnerability Assessments reports **enumerate discovered vulnerabilities** with the associated risk estimate.
- A security aware individual would easily see the problems.
- But, **what about the people from the Financial areas?**
- For them to get involved, **they need to see the facts!** You must show them how “their” information can be compromised -> screenshots, live-demos...

*In the end, it's all about risk management - and they can't do that if they don't know which are the involved threats.*

- *Moreover, it will enable the review of the “security in-depth” principle in the customer, impossible without exploitation.*



## Exploits in the SAP World

- Traditional **memory corruption exploits** are not suitable for SAP PTs.
- **Main reasons:**
  - Variety of Operating Systems.
  - Variety of Processor architectures.
  - Reliability (!)
- Most of the exploit plugins in **sapyto** could be classified under “abuse of functionality”, so are safe\* to use in production environment.
- However, **never forget you are testing a business-critical infrastructure** and take proper precautions...

\* Certain limitations apply. Use exploit plugins only if you know exactly \*what\* you are doing.



## SAP Password Considerations & Cracking

- SAP has implemented different **password hashing mechanisms**.
- Passwords hashes are stored in table **USR02** (BCODE, PASSCODE) and **USH02**.

Code Vers.	Description
A	Obsolete
B	Based on MD5, 8 characters, Uppercase, ASCII
C	Not implemented
D	Based on MD5, 8 characters, Uppercase, UTF-8
E	Reserved
F	Based on SHA1, 40 characters, Case Insensitive, UTF-8
G	Code Version F + Code Version B (2 hashes)

- On June 26 2008, a **patch for John The Ripper** for CODVN B and G was publicly released.



## SAP Password Considerations & Cracking

- SAP has implemented different password hashing mechanisms.
- Passwords hashes are stored in table **USR02** (BCODE, PASSCODE) and **USH02**

### Protection / Countermeasure



- Access to tables **USR02** and **USH02** should be protected.
- Password security should be enforced through profile configuration (login/\* parameters).
- Table **USR40** can be used to protect from trivial passwords.
- For more information, refer to **SAP Note 1237762**.

G

Code Version F + Code Version B (2 hashes)

- On June 26 2008, a patch for **John The Ripper** for CODVN B and G was publicly released.



## Abusing of Sensitive ABAP Reports

- There are many **dangerous** reports/programs.
- Reports can be run through **many** transactions (SE80, SE38, SA38, ...)
- An example of such a dangerous program: **RSBDCOS0**.

```
Operating system command  Edit  Goto  Environment  System  Help  Scripts ?
Execute OS Command (Logged in SYSLOG and Trace Files)
Reset list  Change current directory

R/3  TL1 001      User    SAP*      Date  28.06.2005 Time 12:24:55
Host  sap101      User    t11adm
Path  /usr/sap/TL1/DVEBMGS00/work

Execute history command number with next command
Execute last history command with next command ..
$(name) replaced by logical OS commands and profile parameters

[1]whoami
t11adm
[2]ls -l
total 1976
-rw-r--r-- 1 t11adm sapsys  5724 2005-06-28 12:24  available.log
lrwxrwxrwx 1 t11adm sapsys    35 2005-06-28 11:29  co.sapTL1_DVEBMGS00 -> /usr/sap/TL1/DVEBMGS00/exe/rs1gcol1
-rw-r--r-- 1 t11adm sapsys  2598 2008-04-07 17:13  CPICTRC13440
-rw-r--r-- 1 t11adm sapsys  3311 2008-05-01 13:18  CPICTRC3621
-rw-r--r-- 1 t11adm sapsys  2712 2008-03-04 22:25  CPICTRC3823
-rw-r--r-- 1 t11adm sapsys  2628 2008-05-01 18:20  CPICTRC3824
-rw-r--r-- 1 t11adm sapsys  2662 2008-05-04 15:00  CPICTRC3825
-rw-r--r-- 1 t11adm sapsys  2605 2008-02-29 10:15  CPICTRC3850
-rw-r--r-- 1 t11adm sapsys  2556 2005-06-28 10:55  CPICTRC3855
-rw-r--r-- 1 t11adm sapsys  4398 2008-02-13 23:32  CPICTRC3863
-rw-r--r-- 1 t11adm sapsys  3569 2008-02-29 10:20  CPICTRC3870
-rw-r--r-- 1 t11adm sapsys  2602 2008-04-29 14:00  CPICTRC3875
-rw-r--r-- 1 t11adm sapsys  2552 2008-05-06 20:03  CPICTRC3879
```



## Abusing of Sensitive ABAP Reports

- There are many **dangerous** reports/programs.
- Reports can be run through **many** transactions (SE80, SE38, SA38, ...)
- An example of such a dangerous program: **RSBDCOS0**.

### Protection / Countermeasure



- **Access to transaction SA38 should be restricted.**
- **Reports should not be run directly, they should be linked to transactions.**
- **Use authorization object S\_PROGRAM to protect sensitive reports.**

```
new report 1 f11adm sapsys 2511 2008-05-01 10:18:00 C11CTR0261
new report 1 f11adm sapsys 2512 2008-05-01 10:25:00 C11CTR0262
new report 1 f11adm sapsys 2520 2008-05-01 10:20:00 C11CTR0264
new report 1 f11adm sapsys 2563 2008-05-01 10:30:00 C11CTR0275
new report 1 f11adm sapsys 2655 2008-05-29 10:15:00 C11CTR0280
new report 1 f11adm sapsys 2556 2008-06-25 10:55:00 C11CTR0265
new report 1 f11adm sapsys 4530 2008-02-13 10:30:00 C11CTR0363
new report 1 f11adm sapsys 2595 2008-02-29 10:30:00 C11CTR0370
new report 1 f11adm sapsys 2502 2008-04-29 14:00:00 C11CTR0375
new report 1 f11adm sapsys 2552 2008-05-01 10:30:00 C11CTR0270
```





## Exploiting SAP/Oracle Authentication Mechanism

- Discovered by me in 2007.
- Discovered by Jochen Hein in 2002 (D'oh!)
- Target: Default SAP/Oracle installations.

### The SAP+Oracle Authentication Mechanism

- SAP connects to the database as the OPS\$<SID>ADM (e.g: OPS\$TL1ADM)
- Retrieves encrypted username and password from table SAPUSER.
- Re-connects to the database, using the retrieved credentials.



## Exploiting SAP/Oracle Authentication Mechanism

- There is a special Oracle configuration parameter named **REMOTE\_OS\_AUTHENT**.
- If set to TRUE, Oracle “**trusts**” that the remote system has authenticated the user used for the SQL connection (!)
- The user is created as “identified externally” in the Oracle database.
- Oracle recommendation: **remote\_os\_authent = false**
- SAP **default** and **necessary** configuration: **remote\_os\_authent = true**
  
- **What do you need?**
  - Database host/port.
  - SAP System ID.
  - Oracle Instance ID ( = SAPSID?)



## Exploiting SAP/Oracle Authentication Mechanism

- There is a special Oracle configuration parameter named **REMOTE\_OS\_AUTHENT**.
- If set to TRUE, Oracle “**trusts**” that the remote system has authenticated the user used for the SQL connection (!)
- The user is **not** prompted for a password.
- Oracle records the connection in the **listener log**.
- SAP default configuration is **REMOTE\_OS\_AUTHENT=TRUE**.
- **What do you need to exploit this?**
  - Database user with **sysdba** privilege.
  - SAP System ID.
  - Oracle Instance ID (= SAPSID?)

### Protection / Countermeasure

#### Restrict who can connect to the Oracle listener:

```
tcp.validnode_checking = yes  
tcp.invited_nodes = (192.168.1.102, ...)
```



## Remote sapyto shells and Beyond

- sapyto **exploit plugins** will generate shells/agents.
- For example, by abusing weak RFC interfaces, a **remote shell** can be spawned:

```
Starting EXPLOIT plugins
-----

rfcexec(target#1-3) {
    Trying to connect...
    Creating new SHELL...
    SHELL created.
} res: Ok
Finishing sapyto execution - Fri Apr 1 22:37:42 2009
sapyto> shells
sapyto/shells> show
Shell ID: 0 [RFCEXECShell]
Target information (#1):
Host: sap101

Connector: SAPRFC_EXT
SAP Gateway Host: sap101
SAP Gateway Service: 3300
Tpname: sap101.rfcexec

sapyto/shells> start 0
Starting shell #0
RFCEXECShell - Run commands & read files through rfcexec.
The remote target OS is: Linux.
sapyto/shells/0> run whoami
Command was run successfully.
tlladm
```





## Remote sapyto shells and Beyond

- sapyto **exploit plugins** will generate shells/agents.
- For example, by abusing weak RFC interfaces, a **remote shell** can be spawned:

### Protection / Countermeasure



- **Starting of External RFC Servers is controlled through the file specified by the *gw/sec\_info* profile parameter.**
- **This file should exist and restrict access to allowed systems to start specific programs in the Application Servers.**
- **The *gw/reg\_info* file protects Registered Servers and should also be configured.**
- **For more information, refer to SAP Note 618516.**

```
sapyto/shells> start 0
Starting shell #0
          RFCEXECShell - Run commands & read files through rfcexec.
          The remote target OS is: Linux.
sapyto/shells/0> run whoami
          Command was run successfully.
          tlladm
```

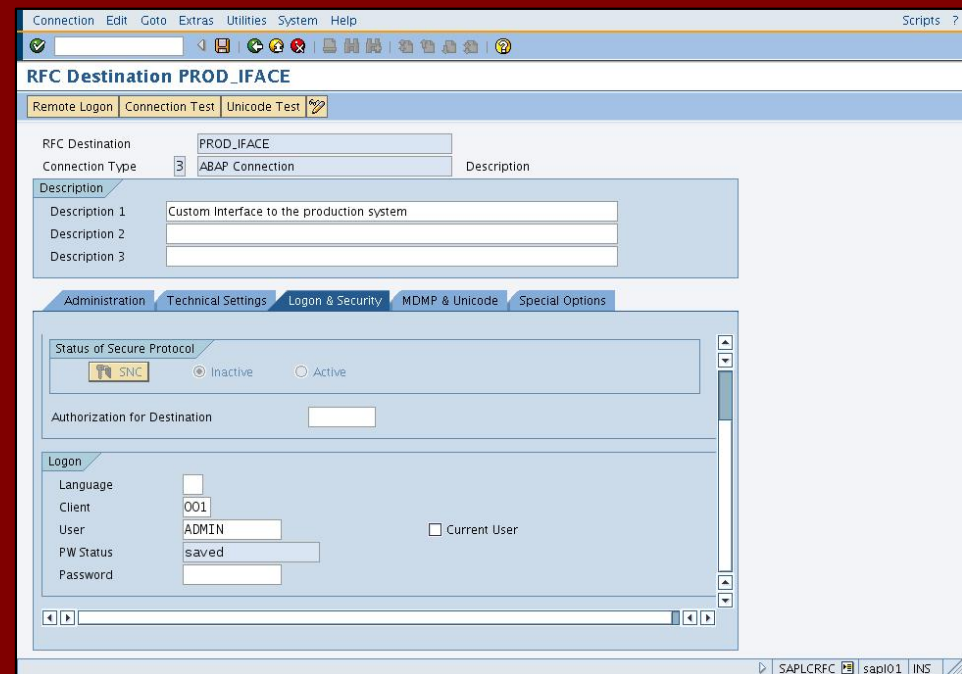


## Expanding Influence

- SAP Application Servers **may need to communicate** with other systems.
- Several types of interfaces exist: RFC, HTTP, etc.
- Interfaces are managed centrally, stored in table **RFCDES**, administered through transaction **SM59**.
- Some of these interfaces need to provide logon information to access the remote system.

### *From the trenches*

- It is very common to find connections with “stored credentials”.
- Usually, designed users have **\*high privileges\*** (e.g: SAP\_ALL)





## Expanding Influence

- SAP Application Servers **may need to communicate** with other systems.
- Several types of interfaces exist: RFC, HTTP, etc.
- Interfaces are managed centrally, stored in table **RFCDES**, administered through transactions **SM59**.

- Some of the interfaces connect to remote systems.

### From the tree

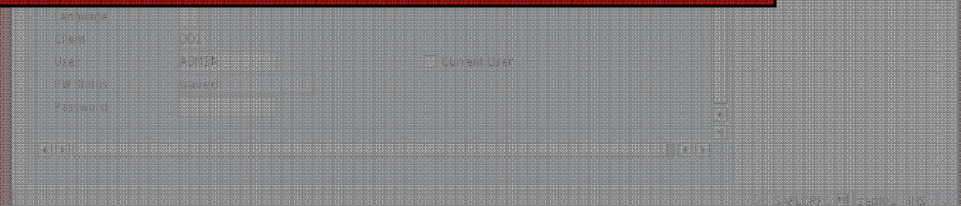
- It is very common to find interfaces with "stored credentials".

- Usually, designed users have **\*high privileges\*** (e.g: SAP\_ALL)

## Protection / Countermeasure



- **Access to transactions SM59 should be restricted.**
- **Access to table RFCDES should be restricted.**
- **Avoid using "stored credentials" -> Trusted systems**
- **Avoid using Dialog users for interfaces.**
- **Restrict user privileges in target systems.**





# Conclusions

*Wrapping up*





## Conclusions

- It's impossible to cover all the activities of an SAP Pentest in a one hour talk! :P
- SAP systems deal with sensitive business information and processes. The integrity, confidentiality and availability of this information is critical.
- SAP systems security is often overlooked during the implementation phase, in order to avoid “business delays”.
- By default, some configurations would expose the systems to high risk threats.
- SAP security is much more than User Roles/Profiles and Authorizations! All presented vulnerabilities can be exploited despite having perfect SoD and SOX compliance.
- SAP provides many ways to secure systems and communications. Administrators should enable security settings as soon as possible.
- SAP security is indeed getting better in each new release. The problem is that many customers are still running old versions.



## Conclusions

- Pentesting your SAP systems will let you know the **current and effective security level of your implementation.**
- If you are a CSO, it can **help you show your managers** why you need more resources to secure it ;)
- If you are a CFO, you can not tell we didn't warn you! :P
- SAP Penetration Tests should be carried out in **controlled environments**, performed by **qualified experts** in the subject.
- CYBSEC's **sapyto** supports activities of all phases of the project.
- If you are taking SAP PenTest seriously, ask us about ***sapyto Advanced\_Edition.***
- *New research coming soon...*



# ¿Questions?



# Thank you!



CYBSEC<sup>SA</sup>  
Security Systems

[www.cybsec.com](http://www.cybsec.com)