Welcome to

7. Network Management

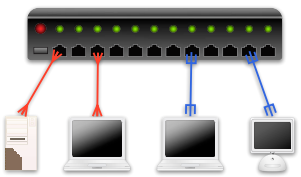
Communication and Network Security 2019

Henrik Lund Kramshøj hlk@zencurity.com

Slides are available as PDF, kramse@Github

7-Network-Management.tex in the repo security-courses

Plan for today



Subjects

- Network Management
- SNMP version 2 vs version 3
- Bruteforcing network devices SSH vs SNMP
- Centralized management SSH, Jump hosts
- Monitoring

Exercises

- Run SNMP walk
- Try brute-force SNMP

Reading Summary

Network management is the process of administering and managing computer networks. Services provided by this discipline include fault analysis, performance management, provisioning of networks and maintaining the quality of service. Software that enables network administrators to perform their functions is called network management software.
`https://en.wikipedia.org/wiki/Network_management`

PPA chapter 9,10,11 - 94 pages
Skim:
`https://nsrc.org/workshops/2015/sanog25-nmm-tutorial/materials/snmp.pdf`

Very common network security task to follow guides like the ones from NSRC.org

What is a core network service?

Core Network Services

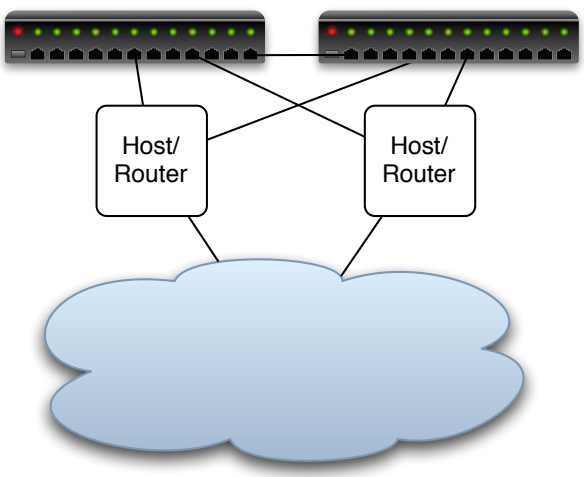
- These are critical for the network to operate correctly. IP packets may flow in the network, but if these services don't reply or aren't configured correctly, users and devices won't be able to connect, authentication services may fail, and network applications won't be accessible.
- They are:
  1. DNS
  2. DHCP
  3. NTP

UNIVERSITY OF OREGON

NSRC
Network Startup Resource Center

Source: https://nsrc.org/workshops/2018/tenet-nsrc-cndo/networking/cndo/en/presentations/Campus_Operations_BCP.pdf

Goals



Show the most important components for secure networks through network management protocols, techniques, systems

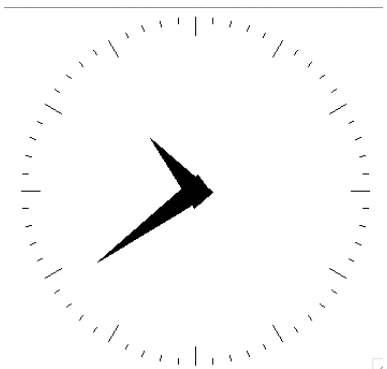
Introduce essential tools for network management

Network Management

Network management is the process of administering and managing computer networks. Services provided by this discipline include fault analysis, performance management, provisioning of networks and maintaining the quality of service. Software that enables network administrators to perform their functions is called network management software.

Source: `https://en.wikipedia.org/wiki/Network_management`

NTP Network Time Protocol



Vigtigt at netværksenheder bruger korrekt tid, sikkerhed og drift

Server NTP foregår typisk i `/etc/ntp.conf` eller `/etc/ntpd.conf`

det vigtigste er navnet på den/de servere man vil bruge som tidskilde

Brug enten en NTP server hos din udbyder eller en fra `http://www.pool.ntp.org/`

Eksempelvis:

```
server 0.dk.pool.ntp.org
server 0.europe.pool.ntp.org
server 3.europe.pool.ntp.org
```

What time is it? - spørg ICMP

ICMP timestamp option - request/reply

hvad er klokken på en server

Slayer icmpush - er installeret på server

viser tidstempel

```
#  icmpush -v -tstamp 10.0.0.12
ICMP Timestamp Request packet sent to 10.0.0.12 (10.0.0.12)

Receiving ICMP replies ...
fischer          -> 21:27:17
icmpush: Program finished OK
```

Stop - NTP Konfigurationseksempler

Vi har en masse udstyr, de meste kan NTP, men hvordan

Vi gennemgår, eller I undersøger selv:

- Switche (managed)
- OpenBSD - check `man rdate` og `man ntpd`
- Mac OS X / Windows / Linux - jeres laptops

BIND DNS server

BIND 9 is transparent open source. If your organization needs some functionality that is not in BIND 9, you can modify it, and contribute the new feature back to the the community by sending us your source. Download a tarball from the ISC web site or ftp.isc.org, or a binary from your operating system repository. BIND 9 has evolved to be a very flexible, full-featured DNS system. Whatever your application is, BIND 9 most likely has the required features.
`https://www.isc.org/downloads/bind/`

Berkeley Internet Name Daemon server

Konfigureres gennem `named.conf`

det anbefales at bruge BIND version 9

BIND konfiguration - et udgangspunkt

```
acl internals { 127.0.0.1; ::1; 10.0.0.0/24; };
options {
        port 53; version "Dont know"; allow-query { any; };
};
view "internal" {
   match-clients { internals; };
   recursion yes;
   zone "." {
       type hint;   file "root.cache"; };
   // localhost forward lookup
   zone "localhost." {
        type master; file "internal/db.localhost";   };
   // localhost reverse lookup from IPv4 address
   zone "0.0.127.in-addr.arpa" {
        type master; file "internal/db.127.0.0"; notify no;   };
...
}
```

Små DNS tools bind-version - Shell script

```
#! /bin/sh
# Try to get version info from BIND server
PROGRAM=`basename $0`
. `dirname $0`/functions.sh
if [ $# -ne 1 ]; then
   echo "get name server version, need a target! "
   echo "Usage: $0 target"
   echo "example $0 10.1.2.3"
   exit 0
fi
TARGET=$1
# using dig
start_time
dig @$1 version.bind chaos txt
echo Authors BIND er i versionerne 9.1 og 9.2 - måske ...
dig @$1 authors.bind chaos txt
stop_time
```

http://www.kramse.dk/files/tools/dns/bind-version

Små DNS tools dns-timecheck - Perl script

```
#!/usr/bin/perl
# modified from original by Henrik Kramshøj, hlk@kramse.dk
# Original from: http://www.rfc.se/fpdns/timecheck.html
use Net::DNS;

my $resolver = Net::DNS::Resolver->new;
$resolver->nameservers($ARGV[0]);
my $query = Net::DNS::Packet->new;
$query->sign_tsig("n","test");

my $response = $resolver->send($query);
foreach my $rr ($response->additional)
  print "localtime vs nameserver $ARGV[0] time difference: ";
  print$rr->time_signed - time() if $rr->type eq "TSIG";
```

http://www.kramse.dk/files/tools/dns/dns-timecheck

Unbound and NSD

Unbound is a validating, recursive, caching DNS resolver. It is designed to be fast and lean and incorporates modern features based on open standards.

To help increase online privacy, Unbound supports DNS-over-TLS which allows clients to encrypt their communication. In addition, it supports various modern standards that limit the amount of data exchanged with authoritative servers.

`https://www.nlnetlabs.nl/projects/unbound/about/`

My preferred local DNS server. We will now stop and look at this configuration file and function.

Also check out uncensored DNS and his DNS over TLS setup!
Even has pinning information available:
`https://blog.censurfridns.dk/blog/32-dns-over-tls-pinning-information-for-unicastcensurfridnsdk/`

DHCPD server

Dynamic Host Configuration Protocol Server

Mange bruger DHCPD fra Internet Systems Consortium

`http://www.isc.org` - altså Open Source

konfigureres gennem `dhcpd.conf` - næsten samme syntaks som BIND

DHCP er en efterfølger til BOOTP protokollen

```
ddns-update-style ad-hoc;
shared-network LOCAL-NET {
    option  domain-name "zencurity.com";
    option  domain-name-servers 10.0.45.1;
    subnet 10.0.45.0 netmask 255.255.255.0 {
            option routers 10.0.45.1;
            range 10.0.45.32 10.0.45.127;
    }
}
```

Rogue DHCP servers

Common problem in networks is people connecting devices with DHCPD servers

In general make sure to segment networks

Start to use port security on switches, including DHCP snooping
`https://en.wikipedia.org/wiki/DHCP_snooping`

Example port security

```
[edit ethernet-switching-options secure-access-port]
set interface ge-0/0/1 mac-limit 4
set interface ge-0/0/2 allowed-mac 00:05:85:3A:82:80
set interface ge-0/0/2 allowed-mac 00:05:85:3A:82:81
set interface ge-0/0/2 allowed-mac 00:05:85:3A:82:83
set interface ge-0/0/2 allowed-mac 00:05:85:3A:82:85
set interface ge-0/0/2 allowed-mac 00:05:85:3A:82:88
set interface ge-0/0/2 mac-limit 4
set interface ge-0/0/1 persistent-learning
set interface ge-0/0/8 dhcp-trusted
set vlan employee-vlan arp-inspection
set vlan employee-vlan examine-dhcp
set vlan employee-vlan mac-move-limit 5
```

Source: Overview of Port Security, Juniper

https://www.juniper.net/documentation/en_US/junos/topics/example/overview-port-security.html

Simple Network Management Protocol

SNMP er en protokol der supporteres af de fleste professionelle netværksenheder, såsom switche, routere

hosts - skal slås til men følger som regel med

SNMP bruges til:

- *network management*
- statistik
- rapportering af fejl - SNMP traps

sikkerheden baseres på community strings der sendes som klartekst ...

det er nemmere at brute-force en community string end en brugerid/kodeord kombination

SNMP version 2 vs version 3

- SNMP versions 1 and 2c are insecure
- SNMP version 3 created to fix this
- Authenticity and integrity: Keys are used for users and messages have digital signatures generated with a hash function (MD5 or SHA)
- Privacy: Messages can be encrypted with secret-key (private) algorithms

Example for Juniper can be found at:

`https://www.juniper.net/documentation/en_US/junos/topics/example/snmpv3-configuration-junos-nm.html`

SNMP - *hacking*

Simple Network Management Protocol

sikkerheden afhænger alene af en Community string SNMPv2

typisk er den nem at gætte:

- public - default til at aflæse statistik
- private - default når man skal ændre på enheden, skrive
- cisco
- ...

Der findes lister og ordbøger på nettet over kendte default communities

Systemer med SNMP

kan være svært at finde ... det er UDP 161

Hvis man finder en så prøv at bruge **snmpwalk** programmet - det kan vise alle tilgængelige SNMP oplysninger fra den pågældende host

det kan være en af måderne at identificere uautoriserede enheder på - sweep efter port 161/UDP

snmpwalk er et af de mest brugte programmer til at hente snmp oplysninger - i forbindelse med hackning og penetrationstest

snmpwalk

Typisk brug er:

`snmpwalk -v 1 -c secret switch1`

`snmpwalk -v 2c -c secret switch1`

Eventuelt bruges `snmpget` og `snmpset`

Ovenstående er en del af Net-SNMP pakken, `http://net-snmp.sourceforge.net/`

Bruteforcing network devices SSH vs SNMP

hvad betyder bruteforcing?
afprøvning af alle mulighederne

```
Hydra v2.5 (c) 2003 by van Hauser / THC <vh@thc.org>
Syntax: hydra [[[-l LOGIN|-L FILE] [-p PASS|-P FILE]] | [-C FILE]]
[-o FILE] [-t TASKS] [-g TASKS] [-T SERVERS] [-M FILE] [-w TIME]
[-f] [-e ns] [-s PORT] [-S] [-vV] server service [OPT]
Options:
  -S        connect via SSL
  -s PORT   if the service is on a different default port, define it here
  -l LOGIN  or -L FILE login with LOGIN name, or load several logins from FILE
  -p PASS   or -P FILE try password PASS, or load several passwords from FILE
  -e ns     additional checks, "n" for null password, "s" try login as pass
  -C FILE   colon seperated "login:pass" format, instead of -L/-P option
  -M FILE   file containing server list (parallizes attacks, see -T)
  -o FILE   write found login/password pairs to FILE instead of stdout
...
```

Eksempler på SNMP og management

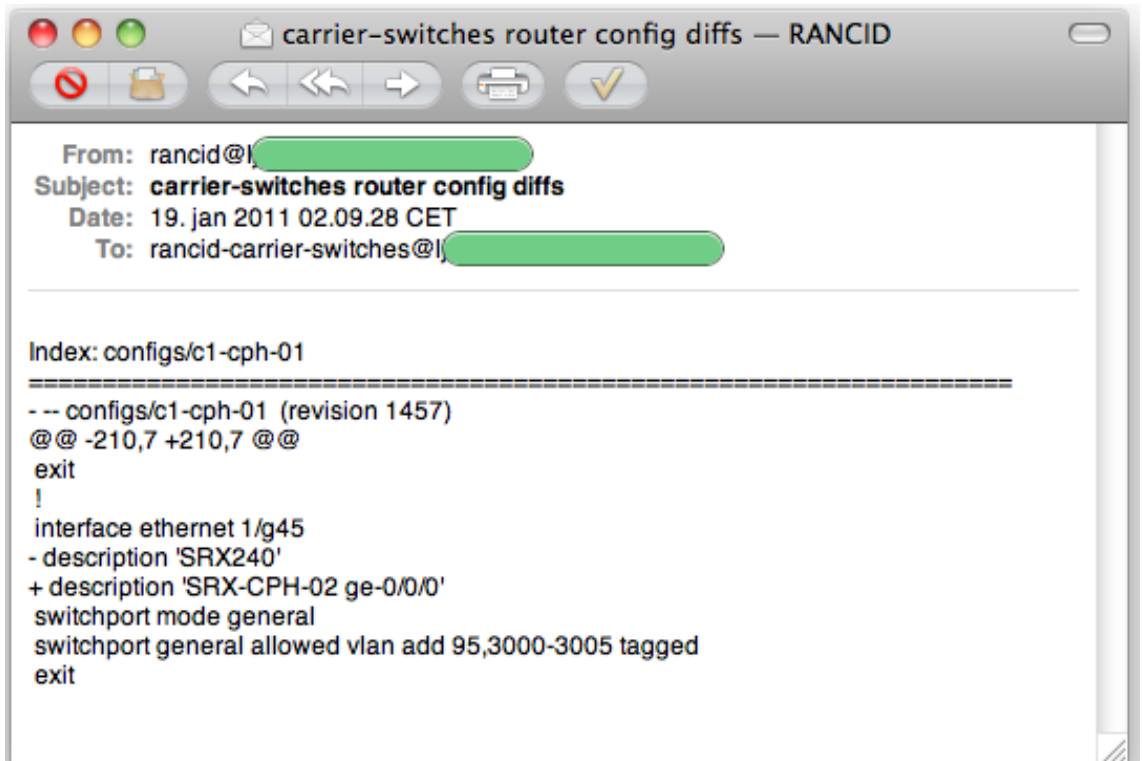
Ofte foregår administration af netværksenheder via HTTP, Telnet eller SSH

- små dumme enheder er idag ofte web-enablet
- bedre enheder giver både HTTP og kommandolinieadgang
- de bedste giver mulighed for SSH, fremfor Telnet

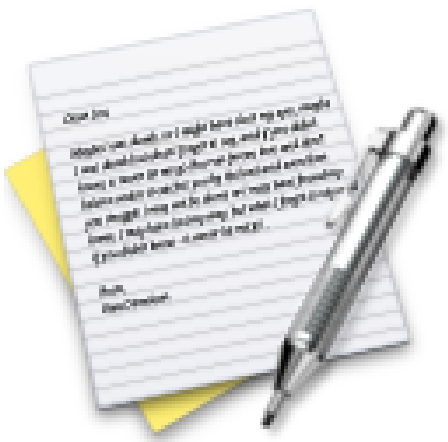
Det er idag muligt at bruge scripting, som:

- RANCID `http://www.shrubbery.net/rancid/`
- Ansible `https://www.ansible.com/`
- Python
- Also make sure to note down `https://github.com/ytti/oxidized`
  Oxidized is a network device configuration backup tool. It's a RANCID replacement!

RANCID output



Exercise

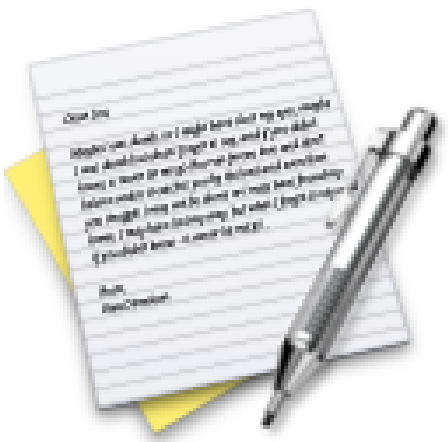


Now lets do the exercise

SNMP walk 15min

which is number **25** in the exercise PDF.

Exercise



Now lets do the exercise

Try Hydra brute force 30min

which is number **26** in the exercise PDF.

Step 1: configure devices properly

You should always configure your devices properly

Turn on SNMP, probably SNMPv2

Turn on LLDP Link Layer Discovery Protocol – vendor-neutral

`http://en.wikipedia.org/wiki/Link_Layer_Discovery_Protocol`

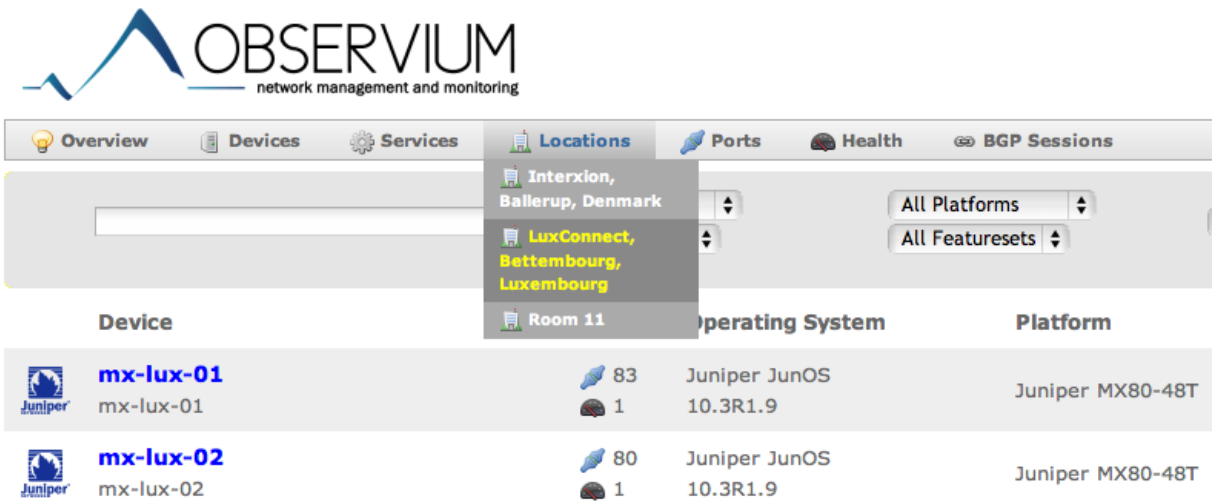
Centralized syslog

And updated firmware, HTTPS and SSH only etc. the usual stuff

Config example: SNMP

```
snmp {
    description "SW-CPH-02";
    location "Interxion, Ballerup, Denmark";
    contact "noc@zencurity.com";
    community yourcommunitynotmine {
        authorization read-only;
        clients {
            10.1.1.1/32;
            10.1.2.2/32;
        }
    }
}
```

Location, location, location

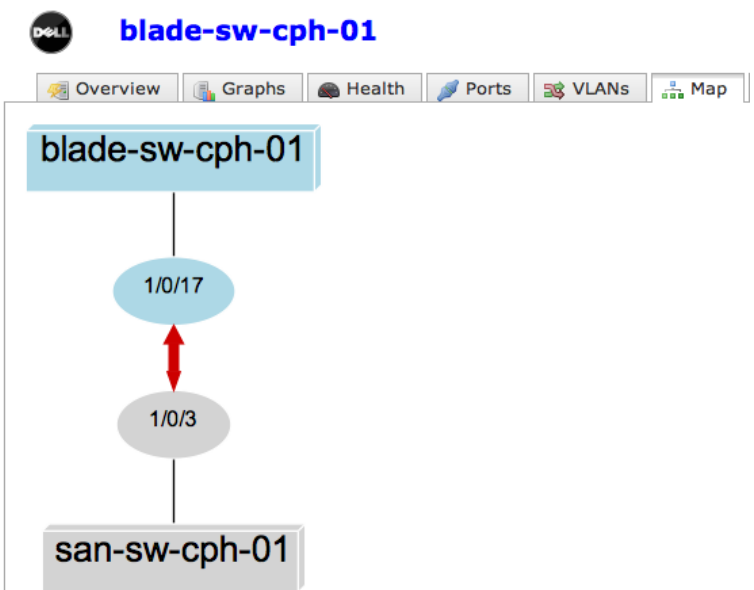


Observium picks up the location from SNMP :-)

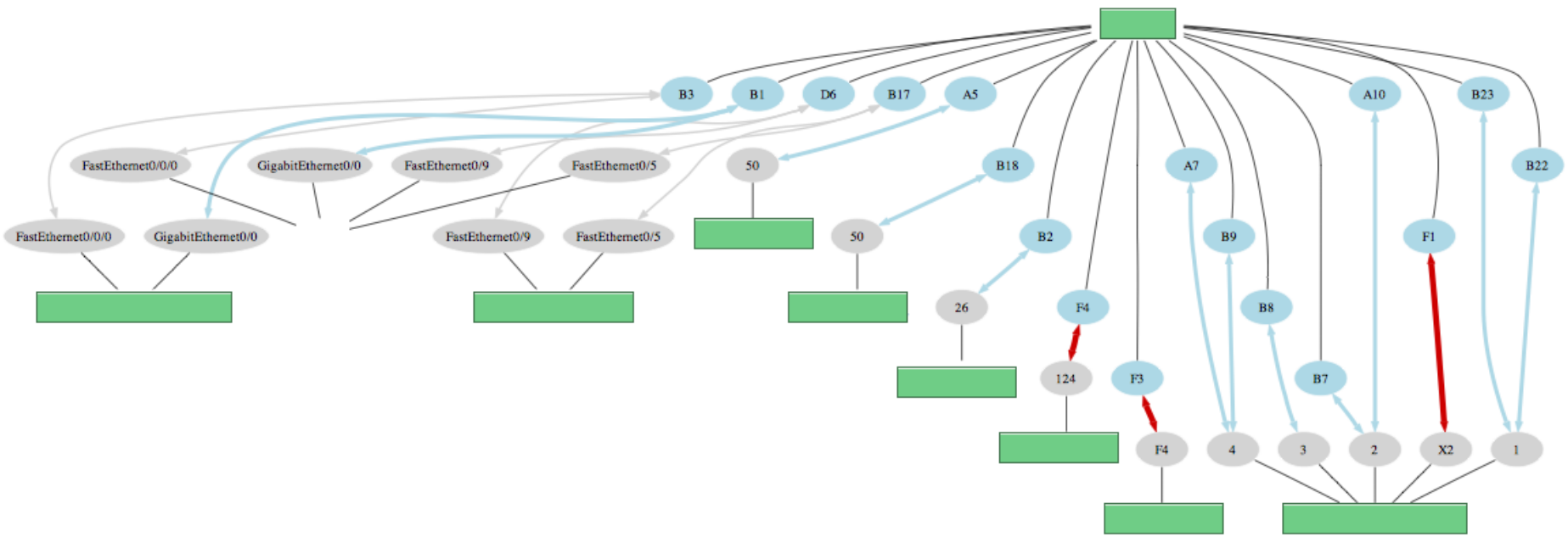
Config example: LLDP

Dell 8024F switch LLDP

```
interface ethernet 1/xg17
mtu 9216
lldp transmit-tlv port-desc sys-name sys-desc sys-cap
lldp transmit-mgmt
exit
```



LLDP spaghetti?



LLDP is needed!

LLDP trick using tcpdump

```
[hlk@ljh ~]$ sudo tcpdump -i eth0 ether proto 0x88cc
tcpdump: verbose output suppressed, use -v or -vv for full protocol decode
listening on eth0, link-type EN10MB (Ethernet), capture size 96 bytes
.... wait for it ....
11:03:55.395064 00:1c:23:80:49:ff (oui Unknown) > 01:80:c2:00:00:0e (oui Unknown),
ethertype Unknown (0x88cc), length 60:
0x0000:  0207 0400 1c23 8049 fd04 0705 312f 302f  .....#.I....1/0/
0x0010:  3331 0602 0078 0000 0000 0000 0000 0000  31...x..........
0x0020:  0000 0000 0000 0000 0000 0000 0000       ..............

1 packets captured
3 packets received by filter
0 packets dropped by kernel
```

I know **for sure** that this server is in Unit 1 port 31!

Basic stuff - consoles

Conserver is an application that allows multiple users to watch a serial console at the same time. It can log the data, allows users to take write-access of a console (one at a time), and has a variety of bells and whistles to accentuate that basic functionality.

Watch the console!

A network device rebooted - what happened?

I accidently the whole network, what now?

Serial consoles are not dead, and still very useful

`http://www.conserver.com/`

Hardware and software

Soekris, 4-port serial card EUR59 / 430DKK + OpenBSD + conserver

Conserver is easy

```
### set the defaults for all the consoles
# these get applied before anything else
default * {
        # The '&' character is substituted with the console name
        logfile /var/consoles/&;
        # timestamps every hour with activity and break logging
        timestamp 1hab;
        # include the 'full' default
        include full;
        # master server is localhost
        master localhost;
}
...
console portS1 {
        type device;
        device /dev/cua02; parity none; baud 57600;
        idlestring "#";
        idletimeout 5m;         # send a '#' every 5 minutes of idle
        timestamp "";           # no timestamps on this console
}
```

You will actually be able to say what happened at that device

Centralized management SSH, Jump hosts

A jump server, jump host or jumpbox is a computer on a network used to access and manage devices in a separate security zone. The most common example is managing a host in a DMZ from trusted networks or computers.

`https://en.wikipedia.org/wiki/Jump_server`

OpenSSH client config with jump host

My recommended SSH client settings, put in `$HOME/.ssh/config`:

```
Host *
    ServerAliveInterval=30
    ServerAliveCountMax=30
    NoHostAuthenticationForLocalhost yes
    HashKnownHosts yes
    UseRoaming no

Host jump-01
  Hostname 10.1.2.3
  Port 12345678

Host fw-site-01 10.1.2.5
  User hlk
  Port 34
  Hostname 10.1.2.5
  ProxyCommand ssh -q -a -x jump-01 -W %h:%p
```

I configure fw using both hostname and IP,
then I can use name, and any program using IP get this config too

What is Ansible

AUTOMATION FOR EVERYONE

Ansible is designed around the way people work and the way people work together.

Ansible has thousands of users, hundreds of customers and over 2,400 community contributors.

750+ Ansible modules

`https://www.ansible.com/`

How Ansible Works: inventory files

List your hosts in one or multiple text files:

```
[all:vars]
ansible_ssh_port=34443

[office]
fw-01 ansible_ssh_host=192.168.1.1 ansible_ssh_port=22
ansible_python_interpreter=/usr/local/bin/python

[infrastructure]
smtp-01     ansible_ssh_host=192.0.2.10
ansible_python_interpreter=/usr/local/bin/python
vpnmon-01   ansible_ssh_host=10.50.60.18
```

- Inventory files specify the hosts we work with
- Linux and OpenBSD servers shown here
- Real inventory for a site with development and staging may be 500 lines
- office and infrastructure are group names

How Ansible Works: ad hoc commands

Using the inventory file you can run commands with Ansible:

```
ansible -m ping new-server
ansible -a "date" new-server
ansible -m shell -a "grep a /etc/something" new-server
```

- Running commands on multiple servers is easy now
- This alone has value, you can start
- Checking settings on servers
- Making small changes to servers

How Ansible Works: Playbooks

The benefit comes with tasks listed in playbooks - do something:

```
- hosts: smartbox-*
  become: yes
  tasks:
  - name: Create a template pf.conf
    template:
      src=pf/pf.conf.j2
      dest=/etc/pf.conf owner=root group=wheel mode=0600
   notify:
      - reload pf
    tags:
      - firewall
      - pf.conf
```

- Specify the end result, more than the steps, also restarts daemons
- Use the modules from `https://docs.ansible.com/ansible/modules_by_category.html`
- Jinja templates - oooooooh so great!

How Ansible Works: typical execution

```
ansible-playbook -i hosts.cph1 -K infrastructure-firewalls.yml -t pf.conf --check --diff

ansible-playbook -i hosts.cph1 -K infrastructure-firewalls.yml -t pf.conf

ansible-playbook -i hosts.cph1 -K infrastructure-nagios.yml -t config-only

ansible-playbook -i smartboxes -K create-pf-conf.yml -l smartbox-xxx-01
```

- Pro tip: check before you push out changes to production networks ☺
- Check will see if something needs changing
- Diff will show the changes about to be made

How Ansible Works: atypical execution / gotchas

```
ansible -i ../smartboxes.osl1 --become --ask-become-pass -m shell
-a "pfctl -s rules" -l smartbox01

ansible -i ../smartboxes.osl1 --become --ask-become-pass -m shell
-a "nmap -sP 185.161.1xx.123-124 2> /dev/null| grep done" all
```

- Sometimes you need a trick or persistence
- Ansible moving from *sudo* to *become*
- The normal -K did not work, but the above does for ad hoc commands

Life of a server

- Create VM
- Network install - with pxeboot
- Standard settings: hostname, LDAP, SSH, timezone, ...
- Configure this server: application installation, settings, etc.
- Configure monitoring: like Smokeping

Get ready, Up and running with Ansible

Prequisites for Ansible - you need a Linux machine:

- python language - Ansible uses this
- ssh keys - remote login without passwords
- Sudo - allow regular users to do superuser tasks
- Recommended tool: `ssh-copy-id` for getting your key on new server
- Recommended Change: `sshd_config` - no passwords allowed, no bruteforce
- Recommended to use: jump hosts/ProxyCommand in `ssh_config`
- Highly recommended: Git and/or github for version control

Official docs:
`https://docs.ansible.com/ansible/intro_installation.html`

Get set, Installation options

Options:

1. use your laptop, easy if you run Mac or Linux
2. install and use a local virtual machine, like Kali Linux and use graphical editor like leafpad for playbooks
3. you also need Git installed

We will use:

```
git clone https://github.com/kramse/ansible-workshop
```

A goal is also to work in team on a server!

We will not do this workshop now in this course

Install Ansible on Mac and Ubuntu Linux clients

Official instructions!
`http://docs.ansible.com/ansible/latest/intro_installation.html`

- Mac OS X `brew install ansible` or use pip

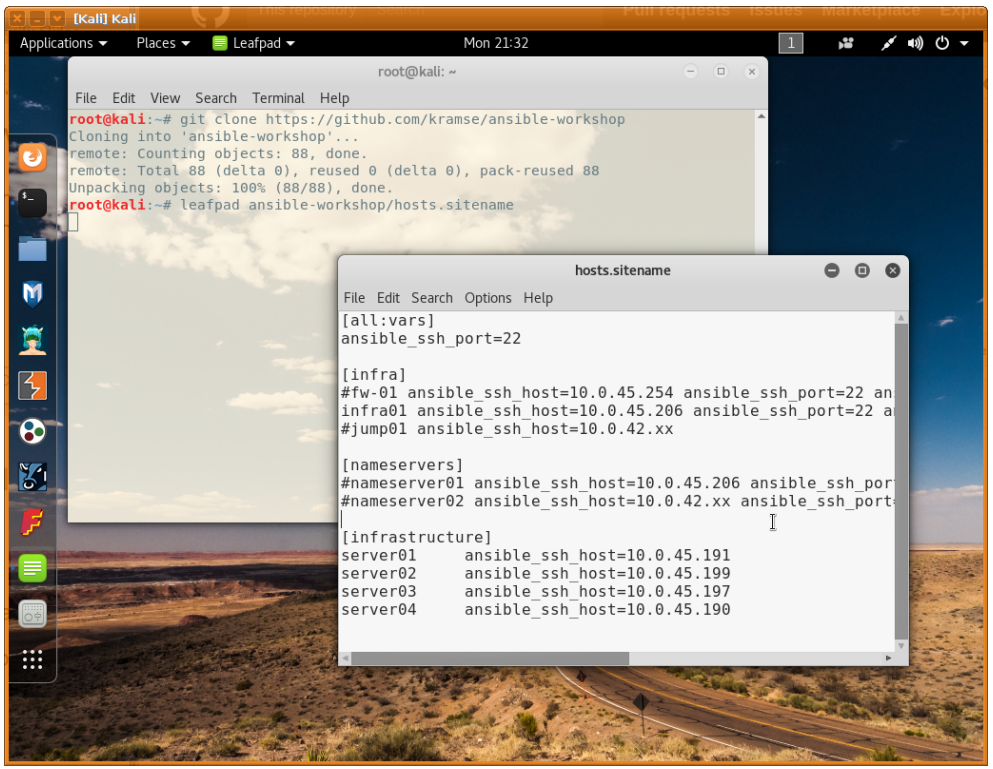
```
$ sudo easy_install pip
$ sudo pip install ansible
```

- Ubuntu Linux try something like:

```
$ sudo apt-get update
$ sudo apt-get install software-properties-common
$ sudo apt-add-repository ppa:ansible/ansible
$ sudo apt-get update
$ sudo apt-get install ansible openssh-client
```

Yes, I expect OpenSSH client is also installed :-D

Kali Linux as Ansible client



Install python on servers

You can use Kali as a client, no problem

- Ubuntu server: `apt install python openssh-server`
- OpenBSD: `pkg_add python`
  Requires `PKG_PATH` set, see below

OpenBSD package path can be set in `/root/.profile`

```
PKG_PATH=ftp://mirror.one.com/pub/OpenBSD/`uname -r`/packages/`uname -m`
PKG_PATH=https://stable.mtier.org/updates/$(uname -r)/$(arch -s):$PKG_PATH
export PKG_PATH
```

Yes, I expect OpenSSH server is also installed ☺

Create OpenSSH compatible private / public key pair

```
hlk@generic:~$ ssh-keygen -f .ssh/kramse
Generating public/private rsa key pair.
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in .ssh/kramse.
Your public key has been saved in .ssh/kramse.pub.
The key fingerprint is:
SHA256:chCjaP6BHaoPy/EMDlP6xKAP4aGAX2mknGA/ZoAzU3o hlk@generic
The key's randomart image is:
+---[RSA 2048]----+
|  .   o          |
|.o . . o         |
|BoE + .          |
|oXoB o .         |
|=.O=* . S        |
|=O++.. o         |
|X++ .            |
|+X=              |
|.o+o             |
+----[SHA256]-----+
```

/home/hlk/.ssh/kramse.pub is the public key in this example

SSH utility ssh-copy-id

You need to copy your SSH public key to the server to use SSH+Ansible:

```
hlk@kunoichi:hlk$ ssh-copy-id -i .ssh/kramse hlk@10.0.42.147
/usr/local/bin/ssh-copy-id: INFO: Source of key(s) to be installed: ".ssh/kramse.pub"
The authenticity of host '10.0.42.147 (10.0.42.147)' can't be established.
ECDSA key fingerprint is SHA256:DP6jqadDWEJW/3FYPY84cpTKmEW7XoQ4zDNf/RdTu6M.
Are you sure you want to continue connecting (yes/no)? yes
/usr/local/bin/ssh-copy-id: INFO: attempting to log in with the new key(s),
to filter out any that are already installed
/usr/local/bin/ssh-copy-id: INFO: 1 key(s) remain to be installed -- if you
are prompted now it is to install the new keys
hlk@10.0.42.147's password:

Number of key(s) added:        1

Now try logging into the machine, with:   "ssh -o 'IdentitiesOnly yes' 'hlk@10.0.42.147'"
and check to make sure that only the key(s) you wanted were added.
```

This is the best tool for the job!

Exercise: trying Ansible

Create inventory file, and then:

```
ansible -m ping new-server
ansible -a "date" new-server
ansible -m shell -a "grep a /etc/something" new-server
```

Lets try running Ansible!

- Hopefully there is a small getting started repo to clone from Github ☺
- Install software: ansible and openssh-client
- Generate SSH key pair - see previous slides
- Server to use should be shown on the whiteboard (or similar)
- You can override user with `ansible -u manager`
  very useful if you are bringing up a server from PXE boot using predefined user `manager`
- Trouble? Try running with `-vvv`, try manual ssh, is Python installed on server and ready?

Success looks something like this

```
$ ssh-keygen -f .ssh/kramse
... generates key pair and saves public key in .ssh/kramse.pub
$ ssh-copy-id -i .ssh/kramse manager@10.0.42.147
... asks for password "henrik42" and installs key
$ cd ansible-workshop
$ leafpad hosts.sitename # add the host server01 for your group

$ ansible -i hosts.sitename -u manager -m ping server01
server01 | success >> {
    "changed": false,
    "ping": "pong"
}
```

Congratulations you are now running Ansible!

Exercise: try fetching facts

```
$ ansible -i hosts.cph1 -u manager -m setup server01 | grep hostname
        "ansible_hostname": "cph1-fw-cph1-01",
```

- Facts are fetched by default from servers
- Can be fetched / investigated using the setup module
- Returns JSON
- Try saving the output in a file and look at it:
  `ansible -i hosts.cph1 -u manager -m setup server01 > facts.txt`
  `less facts.txt`

Goal is to learn basics of Ansible by seeing some server facts

Exercise: try adding you own user

```
---
- hosts: all:!*openbsd*
  become: true
  serial: 10
tasks:
  - group: name=yourusername state=present
  - user: name=yourusername shell=/bin/bash group=sudo
```

- Copy above or edit the create-user.yml
- Replace "hlk"with your username, the one you want
- Run this task / playbook so your own user is created
  `ansible-playbook -i hosts.sitename -K -u manager create-user.yml`
- Dont forget to install your key on this user!
- Try running multiple times, and try adding check and diff:

```
ansible-playbook -i hosts.sitename -K -u yourusername create-user.yml --check --diff
```

Congratulations: you can now do real work with Ansible!

Monitor your network

MRTG The Multi Router Traffic Grapher - simple, great, fast
`http://oss.oetiker.ch/mrtg/`

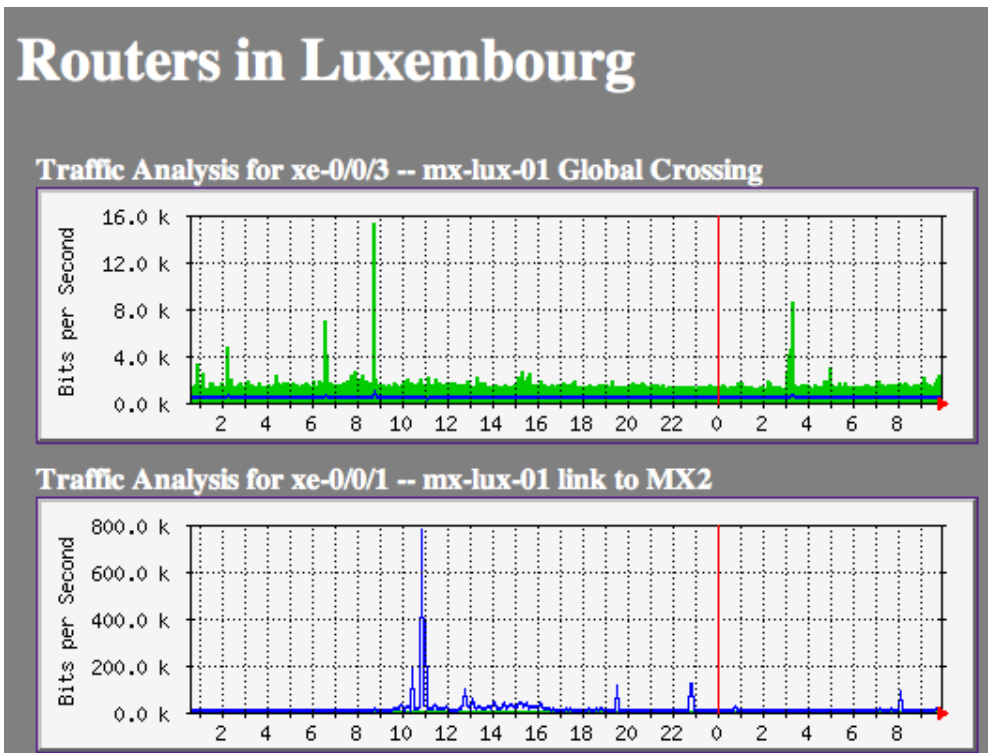
Smokeping Network Latency measurements - network quality
`http://oss.oetiker.ch/smokeping/`

NFsen Netflow monitoring - turn on at selected routers/switches

LibreNMS `https://www.librenms.org/`

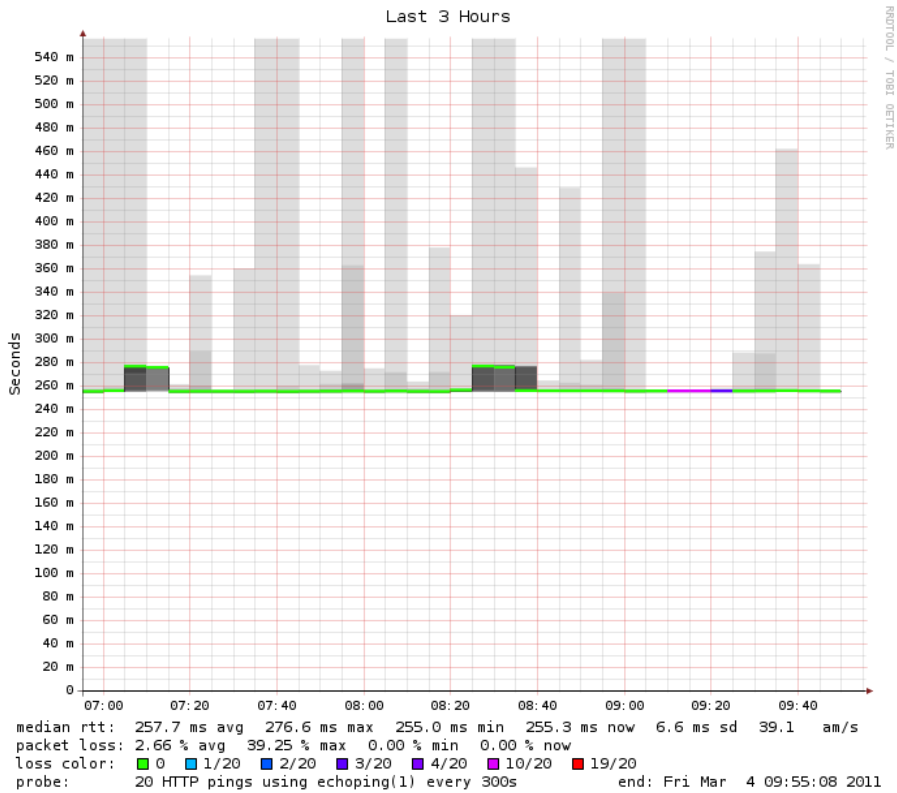
Manual tools, My Traceroute, Nping

MRTG SNMP monitoring made easy

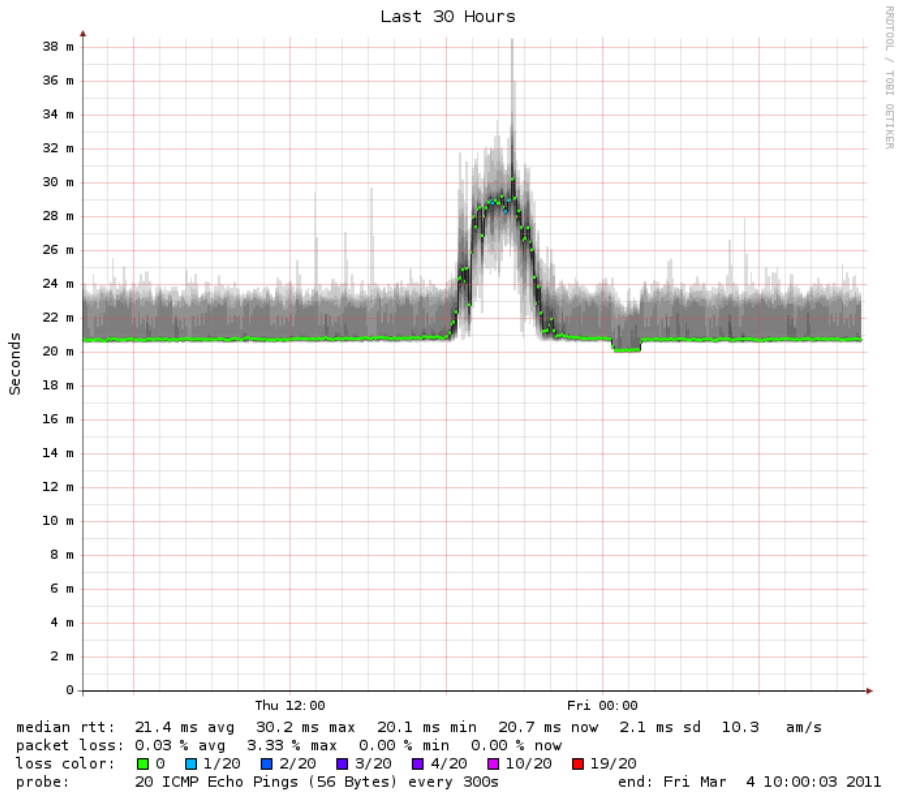


Run configmaker, indexmaker - almost done

Smokeping packet loss



Smokeping latency changed



Netflow

Netflow is getting more important, more data share the same links

Accounting is important

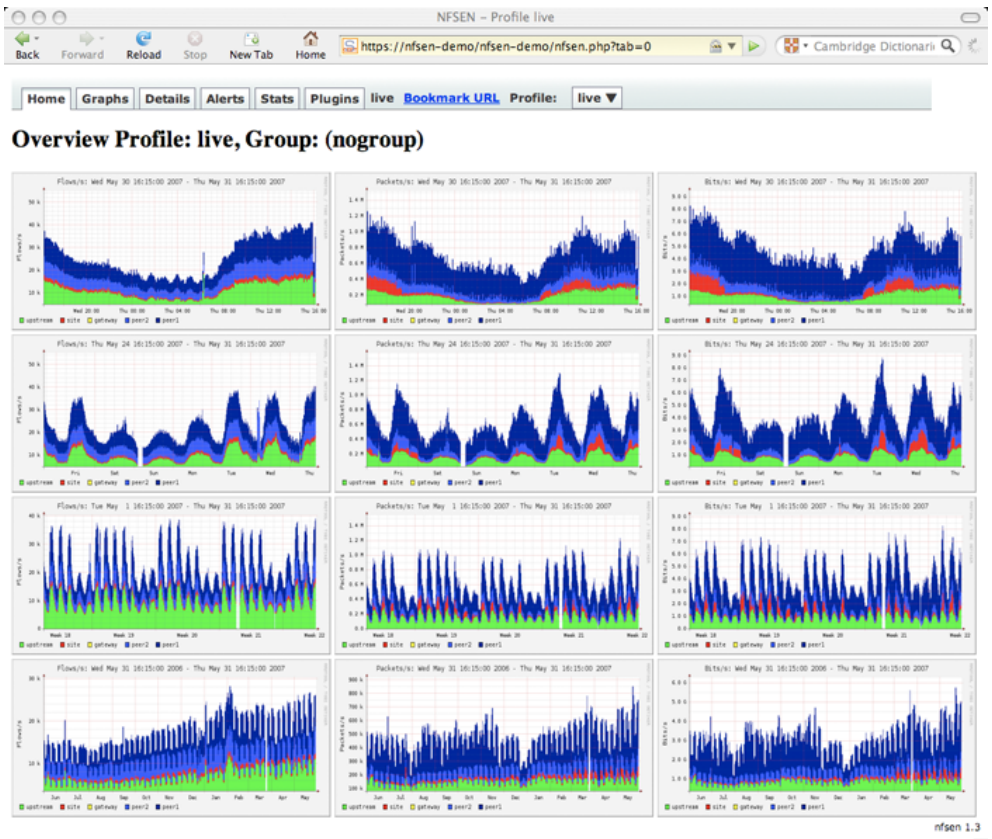
Detecting DoS/DDoS and problems is essential

Netflow sampling is vital information - 123Mbit, but what kind of traffic

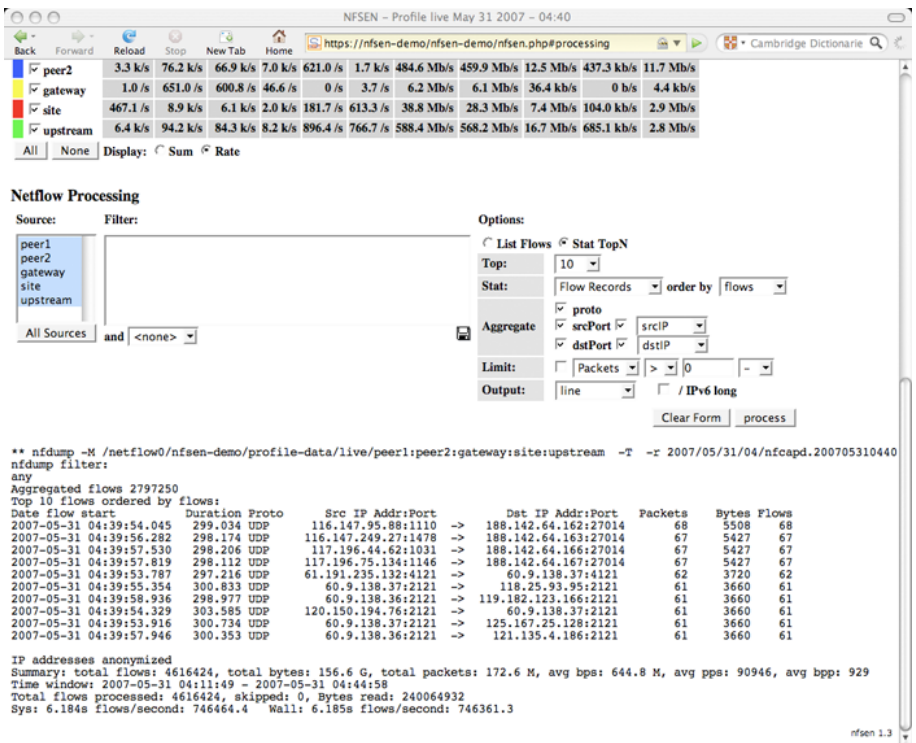
We use mostly NFSen, but are looking at various software packages `http://nfsen.sourceforge.net/`

Currently also investigating sFlow - hopefully more fine grained

Netflow using NFSen

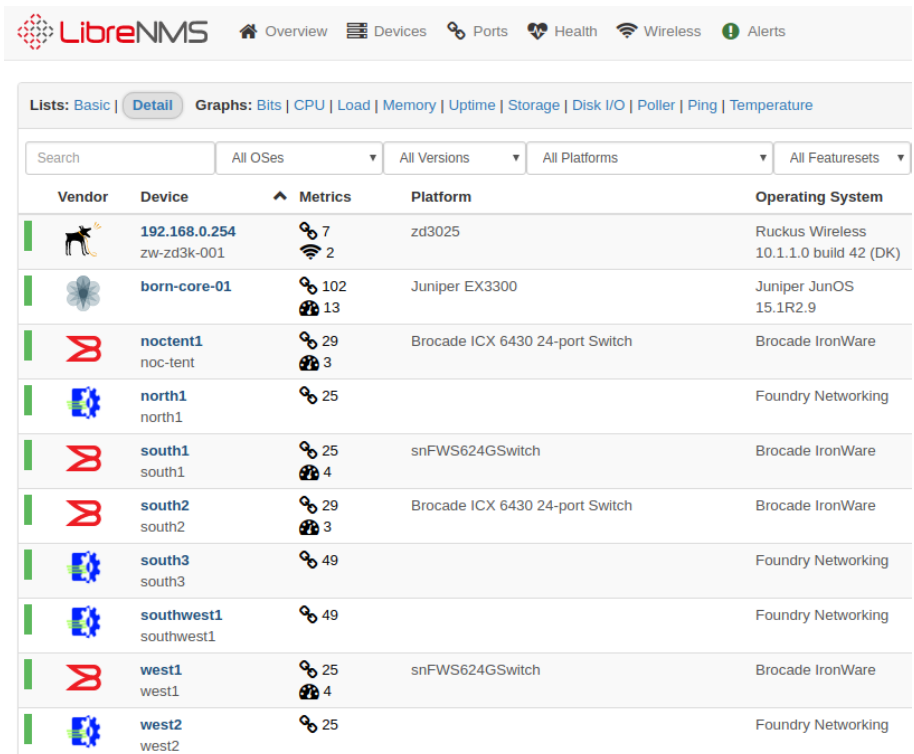


Netflow processing from the web interface



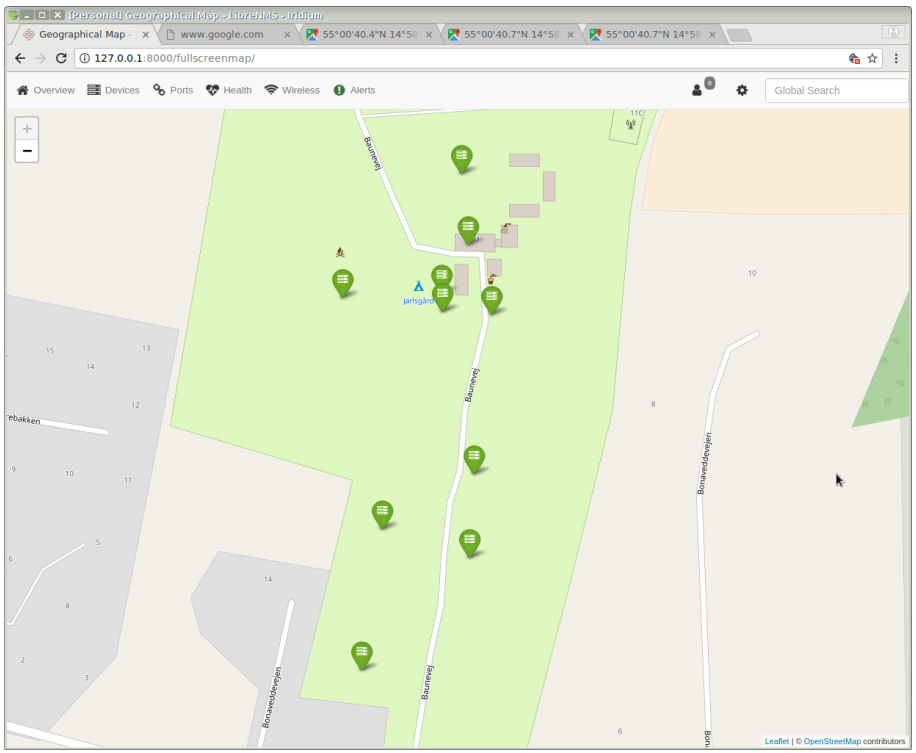
Bringing the power of the command line forward

LibreNMS Automatic discovery

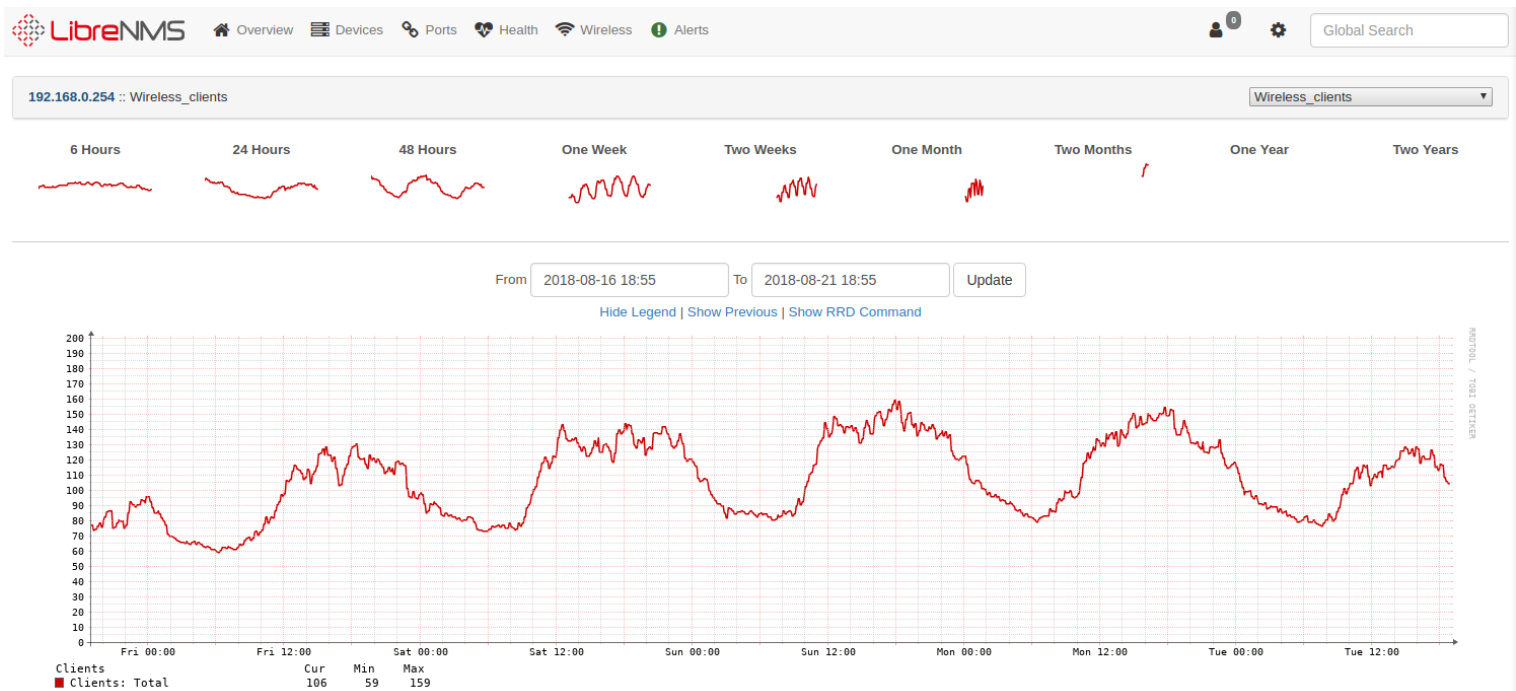


Automatically discover your entire network using CDP, FDP, LLDP, OSPF, BGP, SNMP and ARP.

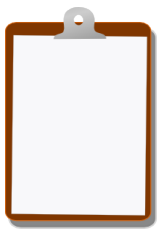
LibreNMS Geo Location



LibreNMS wireless clients



For Next Time



Think about the subjects from this time, write down questions

Check the plan for chapters to read in the books
Most days have about 100 pages or less, but one day has 4 chapters to read!

Visit web sites and download papers if needed

Retry the exercises to get more confident using the tools