Welcome to

3. Traffic Inspection and Firewalls

Communication and Network Security 2019

Henrik Lund Kramshøj hlk@zencurity.com @kramse

Slides are available as PDF, kramse@Github

3-Traffic-Inspection-and-Firewalls.tex in the repo security-courses

Plan for today

Subjects

- Traffic inspection and firewalls
- Generic IP Firewalls stateless filtering vs stateful inspection
- Next Generation firewalls, Deep Packet Inspection
- IEEE 802.1q VLAN
- Common countermeasures in firewalls

Exercises

- Nmap scanning firewalls
- Nmap full scan - strategy
- Nmap reporting

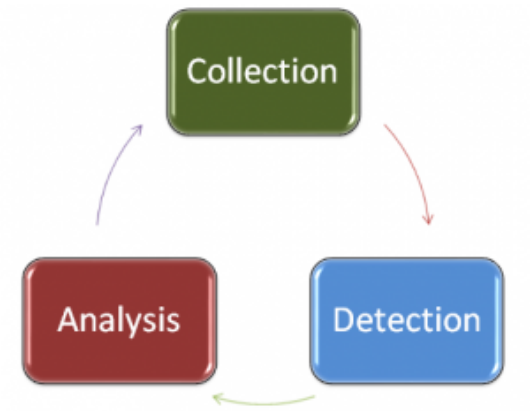
Introduce firewalls

Reading Summary

ANSM chapter 1,2,3 - 73 pages
`https://en.wikipedia.org/wiki/Firewall_(computing)`
`http://www.wilyhacker.com/` Cheswick chapter 2 og 3 PDF, ca 55 pages Skim chapters from 1st edition:
`http://www.wilyhacker.com/1e/chap03.pdf`
`http://www.wilyhacker.com/1e/chap04.pdf`

The next time you are at your console, review some logs. You might think. . . "I don't know what to look for". Start with what you know, understand, and don't care about. Discard those. Everything else is of interest.
Semper Vigilans, Mike Poor

Reading Summary, continued



ANSM chapter 1: The Practice of Applied Network Security Monitoring

- Vulnerability-Centric vs. Threat-Centric Defense
- The NSM cycle: collection, detection, and analysis
- Full Content Data, Session Data, Statistical Data, Packet String Data, and Alert Data
- Security Onion is nice, but a bit over the top - quickly gets overloaded

Security Onion

Security Onion is a Linux distro for IDS (Intrusion Detection) and NSM (Network Security Monitoring).

`http://securityonion.net`

Nice starting point for researching dashboards/network packets

Baseline Skills

- Threat-Centric Security, NSM, and the NSM Cycle
- TCP/IP Protocols
- Common Application Layer Protocols
- Packet Analysis
- Windows Architecture
- Linux Architecture
- Basic Data Parsing (BASH, Grep, SED, AWK, etc)
- IDS Usage (Snort, Suricata, etc.)
- Indicators of Compromise and IDS Signature Tuning
- Open Source Intelligence Gathering
- Basic Analytic Diagnostic Methods
- Basic Malware Analysis

Source: *Applied Network Security Monitoring Collection, Detection, and Analysis*, Chris Sanders and Jason Smith

Reading Summary, continued

ANSM chapter 2: Planning Data Collection

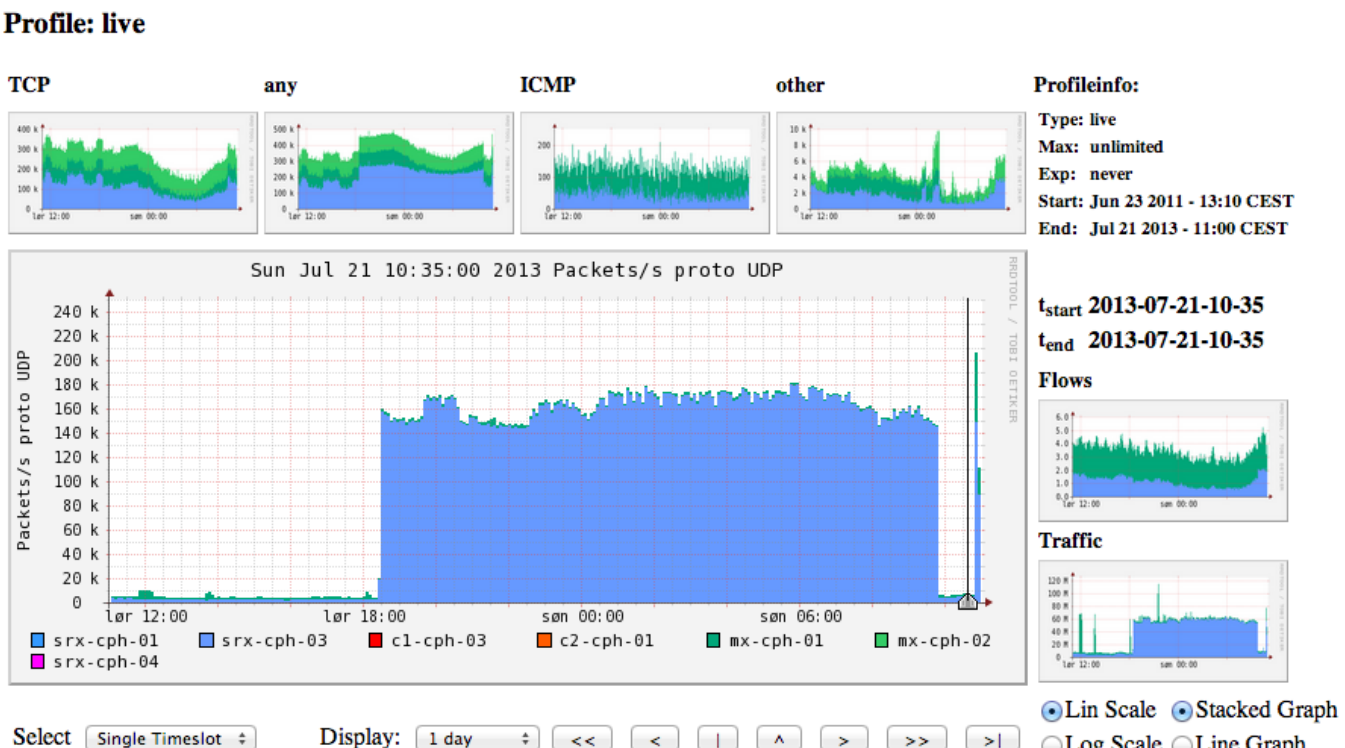
- The Applied Collection Framework (ACF)
- The ACF involves four distinct phases: Identify threats to your organization, quantify risk, identify relevant data feeds, and refine the useful elements
- Risk Analysis
- Lots of terms used, but only defined later in the book

Reading Summary, continued

ANSM chapter 3:The Sensor Platform

- Full Packet Capture (FPC) Data
- Session Data
- Statistical Data
- Packet String (PSTR) Data
- Log Data
- Sensor Placement, designing etc.

Network visibility: Netflow with NFSen



An extra 100k packets per second from this netflow source (source is a router)

How to get started

How to get started searching for security events?

Collect basic data from your devices and networks

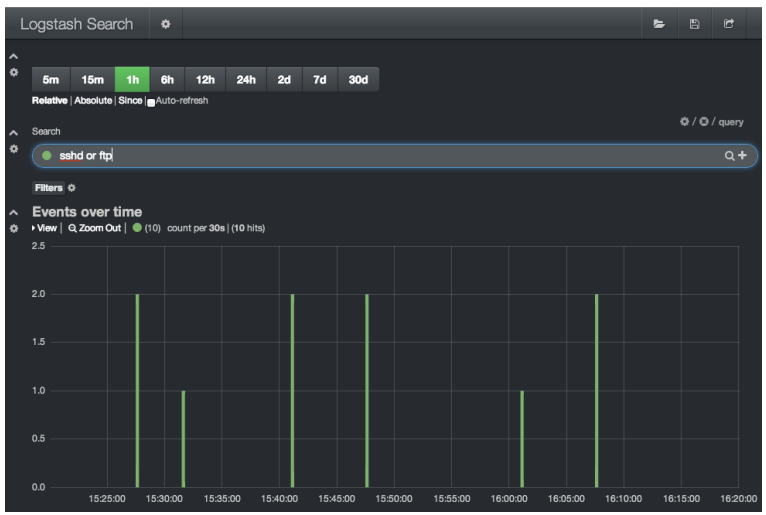
- Netflow data from routers
- Session data from firewalls
- Logging from applications: email, web, proxy systems

Centralize!

Process data

- Top 10: interesting due to high frequency, occurs often, brute-force attacks
- *ignore*
- Bottom 10: least-frequent messages are interesting

View data efficiently



View data by digging into it easily - must be fast

Logstash and Kibana are just examples, but use indexing to make it fast!

Other popular examples include Graylog and Grafana

Reading Summary, continued

> In computing, a firewall is a network security system that monitors and controls incoming and outgoing network traffic based on predetermined security rules.[1] A firewall typically establishes a barrier between a trusted internal network and untrusted external network, such as the Internet.[2]

Source: Wikipedia

`https://en.wikipedia.org/wiki/Firewall_(computing)`

`http://www.wilyhacker.com/` Cheswick chapter 2 PDF *A Security Review of Protocols: Lower Layers*

- Network layer, packet filters, application level, stateless, stateful

Firewalls are by design a choke point, natural place to do network security monitoring!

Reading Summary, continued

Source:

`http://www.wilyhacker.com/` Cheswick chapter 3 PDF *Security Review: The Upper Layers*

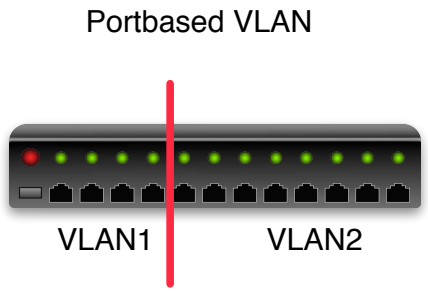
- How to configure firewalls often boil down to, should we allow protocol X
- If we allow SMB through an internet firewall, we are asking for trouble

Skim chapters from 1st edition:

`http://www.wilyhacker.com/1e/chap03.pdf`

`http://www.wilyhacker.com/1e/chap04.pdf`

Together with Firewalls - VLAN Virtual LAN



Nogle switche tillader at man opdeler portene

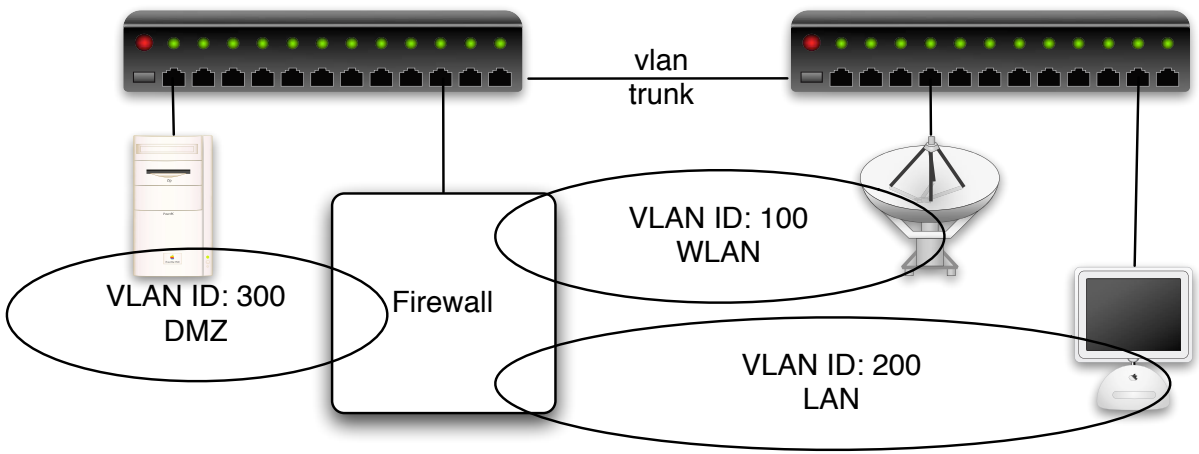
Denne opdeling kaldes VLAN og portbaseret er det mest simple

Port 1-4 er et LAN

De resterende er et andet LAN

Data skal omkring en firewall eller en router for at krydse fra VLAN1 til VLAN2

IEEE 802.1q



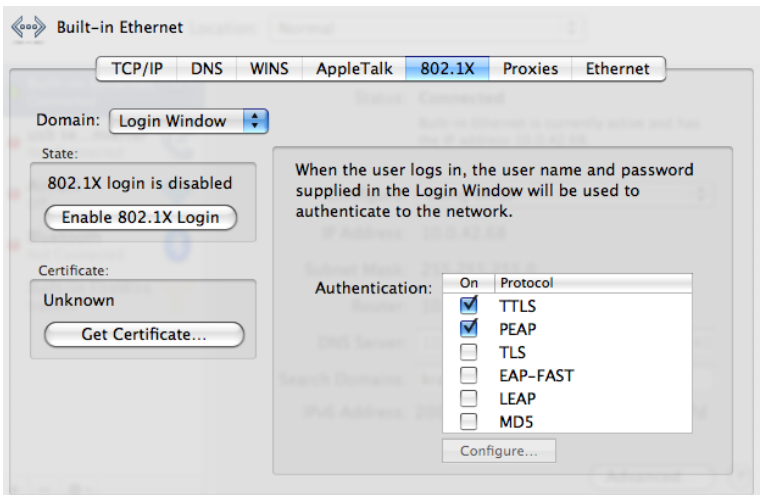
Med 802.1q tillades VLAN tagging på Ethernet niveau

Data skal omkring en firewall eller en router for at krydse fra VLAN1 til VLAN2

VLAN trunking giver mulighed for at dele VLANs ud på flere switches

Connecting clients more securely

IEEE 802.1x Port Based Network Access Control



Denne protokol sikrer at man valideres før der gives adgang til porten

Når systemet skal have adgang til porten afleveres brugernavn og kodeord/certifikat

802.1x og andre teknologier

802.1x i forhold til MAC filtrering giver væsentlige fordele

MAC filtrering kan spoofes, hvor 802.1x kræver det rigtige kodeord

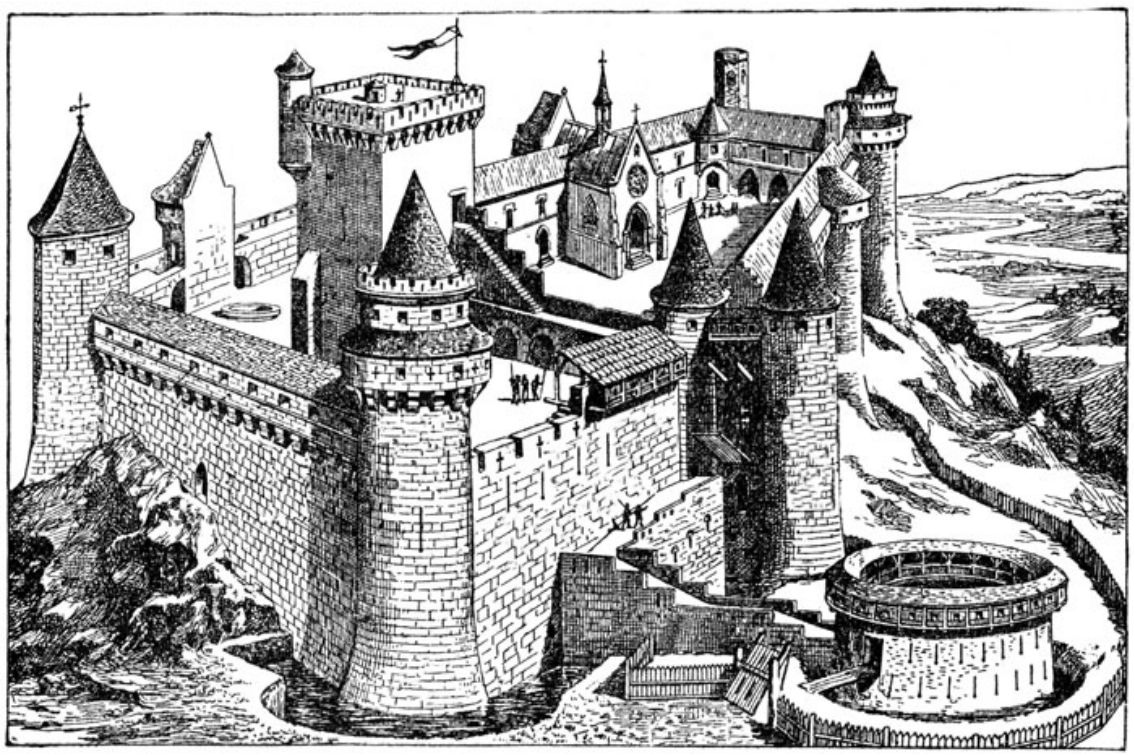
Typisk benyttes RADIUS og 802.1x integrerer således mod både LDAP og Active Directory

Generic IP Firewalls

En firewall er noget som blokerer traffik på Internet

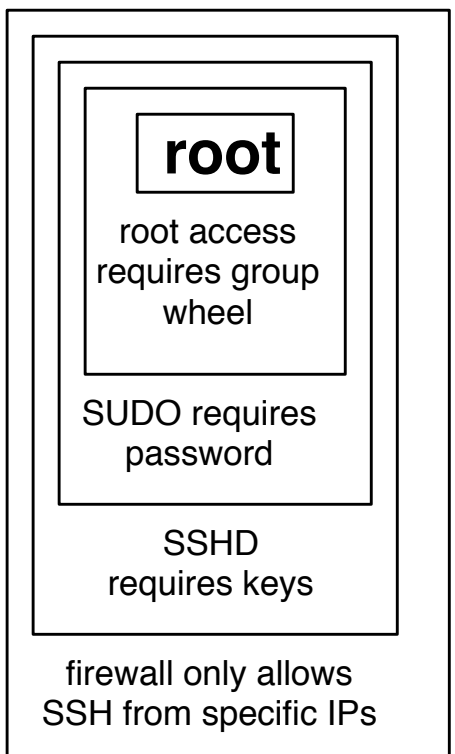
En firewall er noget som tillader traffik på Internet

Defense in depth



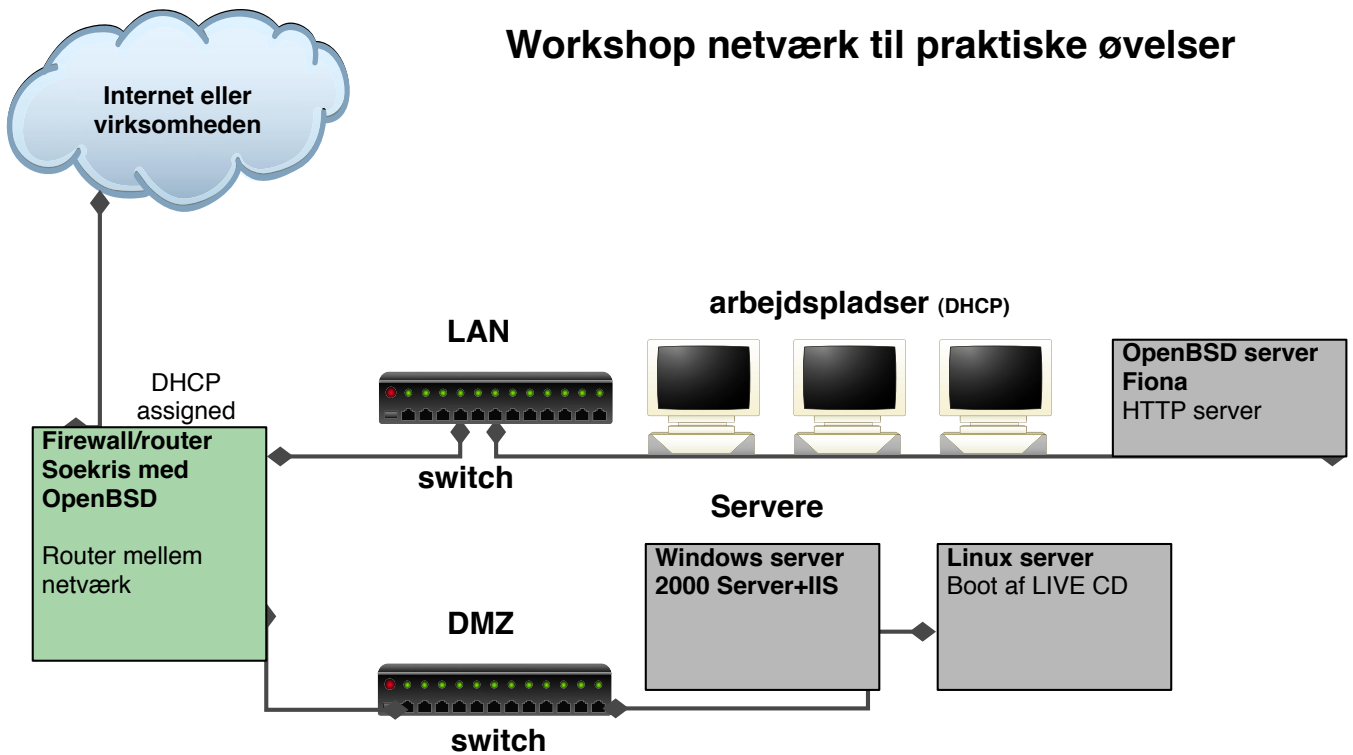
Picture originally from: `http://karenswhimsy.com/public-domain-images`

Defense in depth - layered security

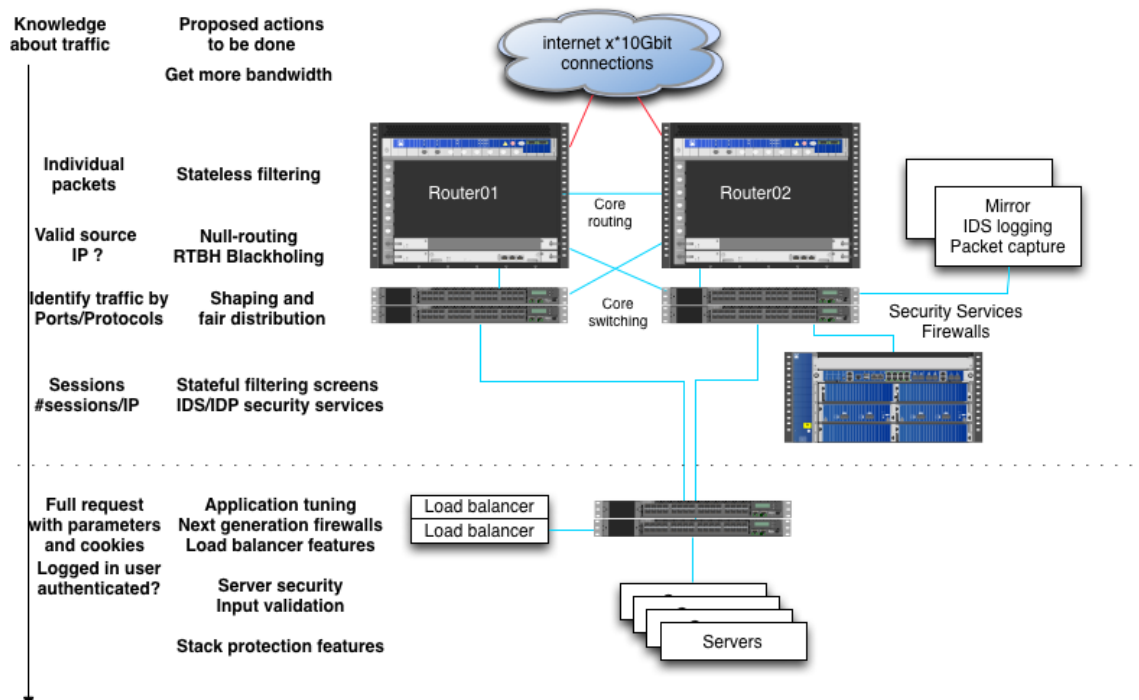


Multiple layers of security! Isolation!

Small networks

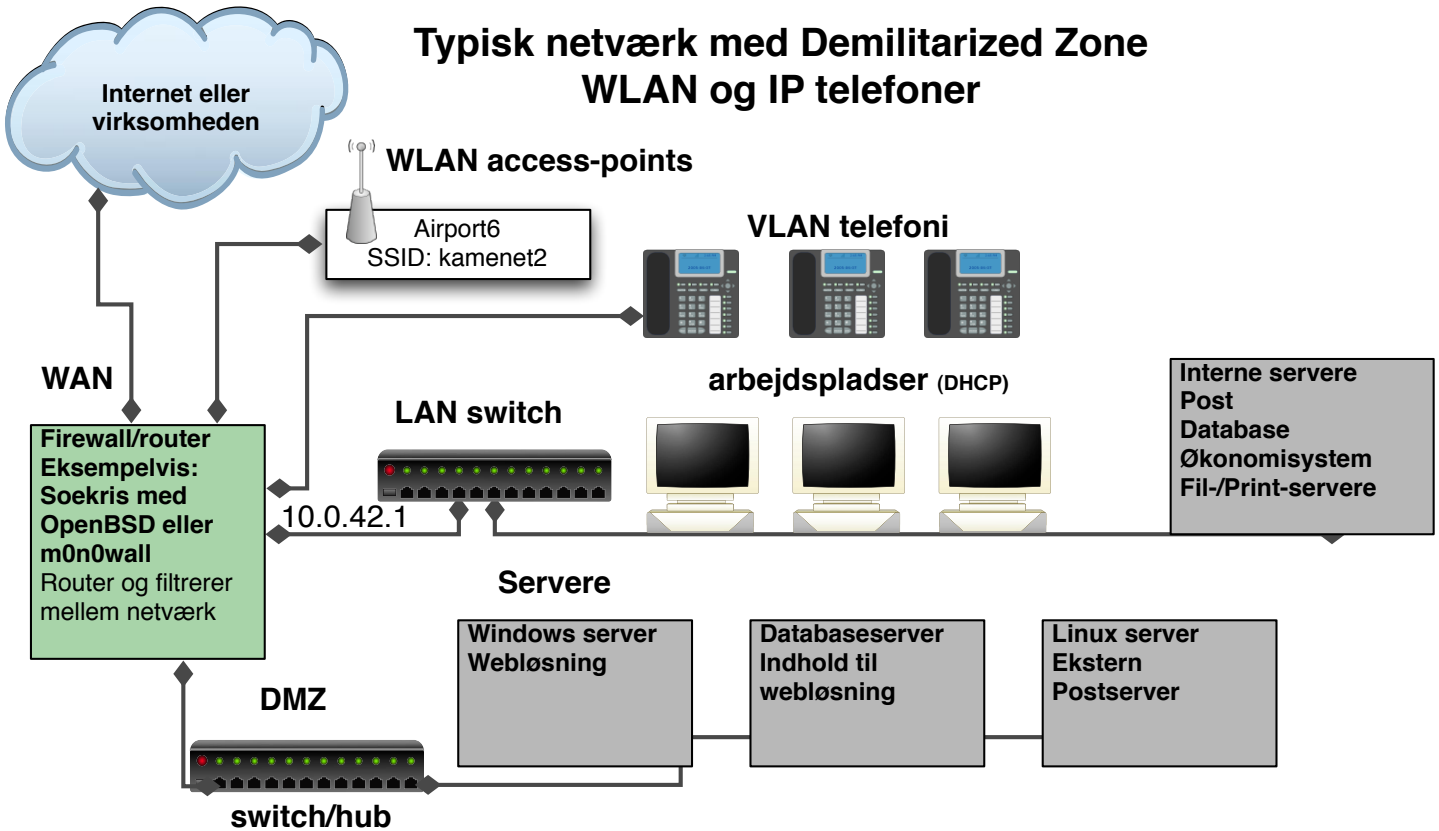


Firewall er ikke alene



Forsvaret er som altid - flere lag af sikkerhed!

Unified communications



Firewallrollen idag

Idag skal en firewall være med til at:

- Forhindre angribere i at komme ind
- Forhindre angribere i at sende traffik ud
- Forhindre virus og orme i at sprede sig i netværk
- Indgå i en samlet løsning med ISP, routere, firewalls, switchede strukturer, intrusion detectionsystemer samt andre dele af infrastrukturen

Det kræver overblik!

Modern Firewalls

Basalt set et netværksfilter - det yderste fæstningsværk

Indeholder typisk:

- Grafisk brugergrænseflade til konfiguration - er det en fordel?
- TCP/IP filtermuligheder - pakkernes afsender, modtager, retning ind/ud, porte, protokol, ...
- både IPv4 og IPv6
- foruddefinerede regler/eksempler - er det godt hvis det er nemt at tilføje/åbne en usikker protokol?
- typisk NAT funktionalitet indbygget
- typisk mulighed for nogle serverfunktioner: kan agere DHCP-server, DNS caching server og lignende

En router med Access Control Lists - kaldes ofte netværksfilter, mens en dedikeret maskine kaldes firewall

Sample rules from OpenBSD PF

```
# hosts and networks
router="217.157.20.129"
webserver="217.157.20.131"
homenet=" 192.168.1.0/24, 1.2.3.4/24 "
wlan="10.0.42.0/24"
wireless=wi0
set skip lo0
# things not used
spoofed=" 127.0.0.0/8, 172.16.0.0/12, 10.0.0.0/16, 255.255.255.255/32 "
```

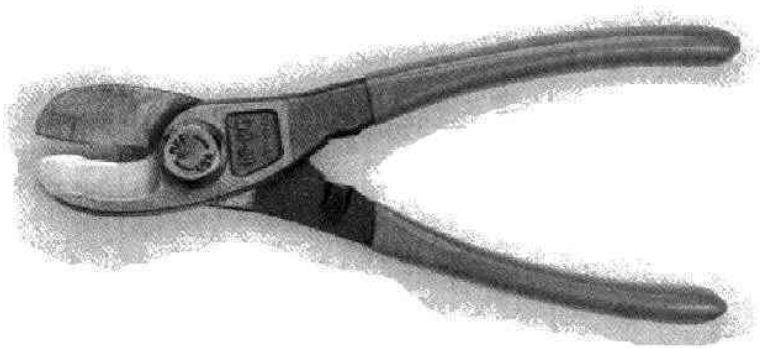
block in all # default block anything

```
# egress and ingress filtering - disallow spoofing, and drop spoofed
block in quick from $spoofed to any
block out quick from any to $spoofed

pass in on $wireless proto tcp from { $wlan $homenet } to any port = 22
pass in on $wireless proto tcp from any to $webserver port = 80

pass out
```

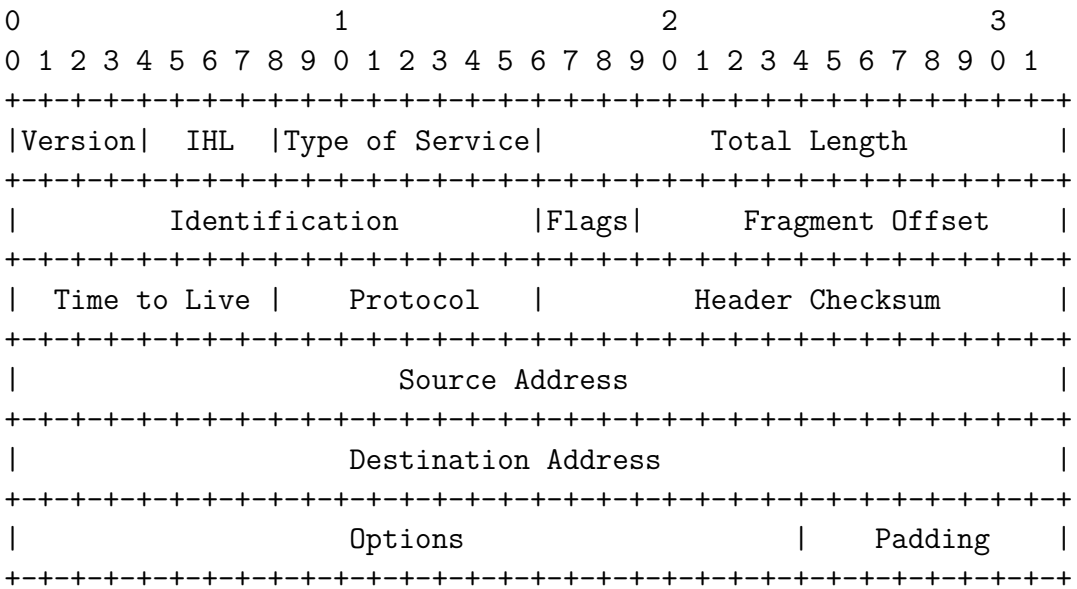
netdesign - med firewalls



- Hvor skal en firewall placeres for at gøre størst nytte?
- Hvad er forudsætningen for at en firewall virker?
  At der er konfigureret et sæt fornuftige regler!
- Hvor kommer reglerne fra? Sikkerhedspolitikken!

Kilde: `http://www.ranum.com/pubs/a1fwall/` The ULTIMATELY Secure Firewall

Packet filtering



Packet filtering er firewalls der filtrerer på IP niveau

Idag inkluderer de fleste stateful inspection

Kommercielle firewalls

- Checkpoint Firewall-1 `http://www.checkpoint.com`
- Cisco ASA `http://www.cisco.com`
- Clavister firewalls `http://www.clavister.com`
- Juniper SRX `http://www.juniper.net`
- Palo Alto `https://www.paloaltonetworks.com/`
- Fortinet `https://www.fortinet.com/`

Ovenstående er dem som jeg oftest ser ude hos mine kunder i Danmark

Open source baserede firewalls

- Linux firewalls IP tables, use command line tool ufw Uncomplicated Firewall!
- Firewall GUIs ovenpå Linux - mange! nogle er kommercielle produkter
- OpenBSD PF `http://www.openbsd.org`
- FreeBSD IPFW og IPFW2 `http://www.freebsd.org`
- Mac OS X benytter OpenBSD PF
- FreeBSD inkluderer også OpenBSD PF

NB: kun eksempler og dem jeg selv har brugt

Hardware eller software

Man hører indimellem begrebet *hardware firewall*

Det er dog et faktum at en firewall består af:

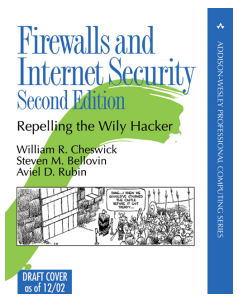
- Netværkskort - som er hardware
- Filtreringssoftware - som er *software*!

Det giver ikke mening at kalde en ASA 5501 en hardware firewall
og en APU2C4 med OpenBSD for en software firewall!

Man kan til gengæld godt argumentere for at en dedikeret firewall som en separat enhed kan give bedre sikkerhed

Det er også fint at tale om host-firewalls, altså at servere og laptops har firewall slået til

Firewall historik

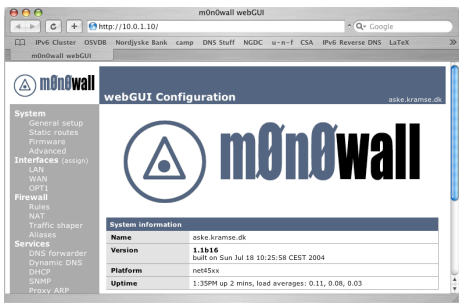


Firewalls har været kendt siden starten af 90'erne

Første bog *Firewalls and Internet Security* udkom i 1994 men kan stadig anbefales, læs den på `http://www.wilyhacker.com/`

2003 kom den i anden udgave *Firewalls and Internet Security* William R. Cheswick, Steven M. Bellovin, Aviel D. Rubin, Addison-Wesley, 2nd edition

m0n0wall



Tidlig firewall bygget på FreeBSD

Idag erstattet af pfSense `https://www.pfsense.org/`
og OPNsense `https://opnsense.org/`

De nye bruges i produktion i danske firmaer

firewall koncepter

Rækkefølgen af regler betyder noget!

- To typer af firewalls: First match - når en regel matcher, gør det som angives block/pass Last match - marker pakken hvis den matcher, til sidst afgøres block/pass

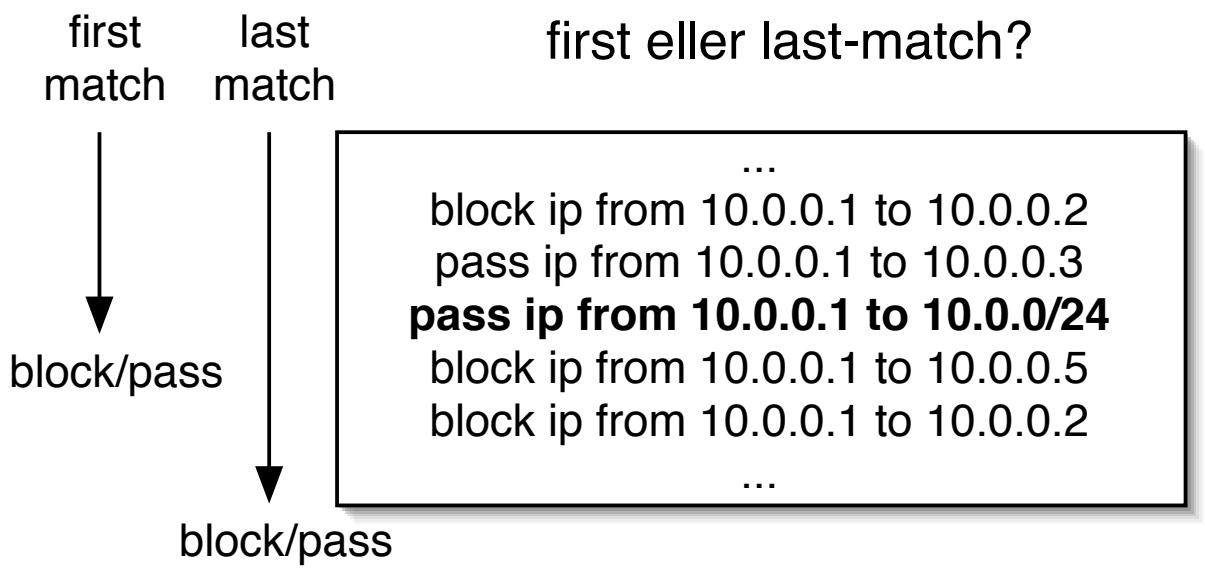
Det er ekstremt vigtigt at vide hvilken type firewall man bruger!

OpenBSD PF er last match

FreeBSD IPFW er first match

Linux iptables/netfilter er last match

First or Last match firewall?



Med dette regelsæt vil en first-match firewall blokere pakker fra 10.0.0.1 til 10.0.0.2 - men tillade alt andet fra 10.0.0.1 til 10.0.0/24

Med dette regelsæt vil en last-match firewall blokere pakker fra 10.0.0.1 til 10.0.0.2, **10.0.0.1 til 10.0.0.5, 10.0.0.1 til 10.0.0.2** - men ellers tillade alt andet fra 10.0.0.1 til 10.0.0/24

First match - IPFW

```
00100 16389  1551541 allow ip from any to any via lo0
00200     0        0 deny log ip from any to 127.0.0.0/8
00300     0        0 check-state
...

65435    36     5697 deny log ip from any to any
65535   865    54964 allow ip from any to any
```

Den sidste regel nås aldrig!

Last match - OpenBSD PF

```
ext_if="ext0"
int_if="int0"

block in
pass out keep state

pass quick on { lo $int_if }

# Tillad forbindelser ind på port 80=http og port 53=domain
# på IP-adressen for eksterne netkort ($ext_if) syntaksen
pass in on $ext_if proto tcp to ($ext_if) port http keep state
pass in on $ext_if proto { tcp, udp } to ($ext_if) port domain keep state
```

Pakkerne markeres med block eller pass indtil sidste regel

nøgleordet *quick* afslutter match - god til store regelsæt

Linux iptables/netfilter eksempel

```
# Firewall configuration written by system-config-securitylevel
# Manual customization of this file is not recommended.
*filter
:INPUT ACCEPT [0:0]
:FORWARD ACCEPT [0:0]
:OUTPUT ACCEPT [0:0]
:RH-Firewall-1-INPUT - [0:0]
-A INPUT -j RH-Firewall-1-INPUT
-A FORWARD -j RH-Firewall-1-INPUT
-A RH-Firewall-1-INPUT -i lo -j ACCEPT
-A RH-Firewall-1-INPUT -p icmp --icmp-type any -j ACCEPT
-A RH-Firewall-1-INPUT -p 50 -j ACCEPT
-A RH-Firewall-1-INPUT -p 51 -j ACCEPT
-A RH-Firewall-1-INPUT -p udp --dport 5353 -d 224.0.0.251 -j ACCEPT
-A RH-Firewall-1-INPUT -p udp -m udp --dport 631 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state ESTABLISHED,RELATED -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 443 -j ACCEPT
-A RH-Firewall-1-INPUT -m state --state NEW -m tcp -p tcp --dport 22 -j ACCEPT
-A RH-Firewall-1-INPUT -j REJECT --reject-with icmp-host-prohibited
COMMIT
```

NB: husk at aktivere IP forwarding

Firewalls og ICMP

```
ipfw add allow icmp from any to any icmptypes 3,4,11,12
```

Ovenstående er IPFW syntaks for at tillade de interessant ICMP beskeder igennem

Tillad ICMP types:

- 3 Destination Unreachable
- 4 Source Quench Message
- 11 Time Exceeded
- 12 Parameter Problem Message

Firewall konfiguration

Den bedste firewall konfiguration starter med:

- Papir og blyant
- En fornuftig adressestruktur

Brug dernæst en firewall med GUI første gang!

Husk dernæst:

- En firewall skal passes
- En firewall skal opdateres
- Systemerne bagved skal hærdes!

Bloker indefra og ud

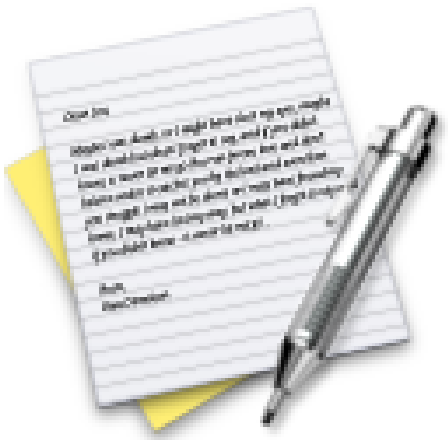
Der er porte og services som altid bør blokeres

Det kan være kendte sårbare services

- Windows SMB filesharing - ikke til brug på Internet!
- UNIX NFS - ikke til brug på Internet!

Kendte problemer som minimum

Exercise

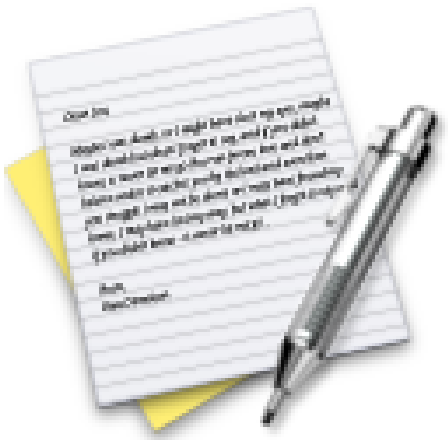


Now lets do the exercise

Perform nmap service scan 10 min

which is number **16** in the exercise PDF.

Exercise

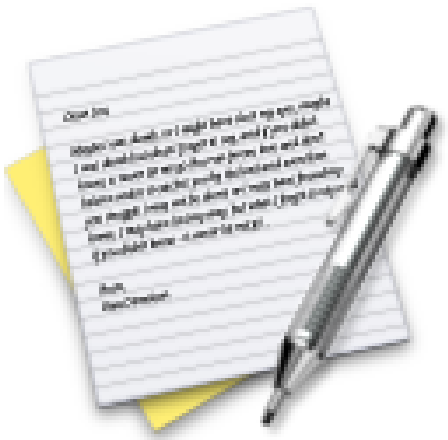


Now lets do the exercise

Nmap full scan - strategy 15 min

which is number **17** in the exercise PDF.

Exercise



Now lets do the exercise

Reporting HTML 15 min

which is number **18** in the exercise PDF.

Specielle features

- Network Address Translation - NAT
- IPv6 funktionalitet
- Båndbredde håndtering
- VLAN funktionalitet - mere udbredt i forbindelse med VoIP
- Redundante firewalls - pfsync og CARP
- IPsec og Andre VPN features
- inspection - diverse muligheder for at lave deep inspection i protokoller
- Eksemplvis DNS inspection

Proxy servers

Filtrering på højere niveauer i OSI modellen er muligt

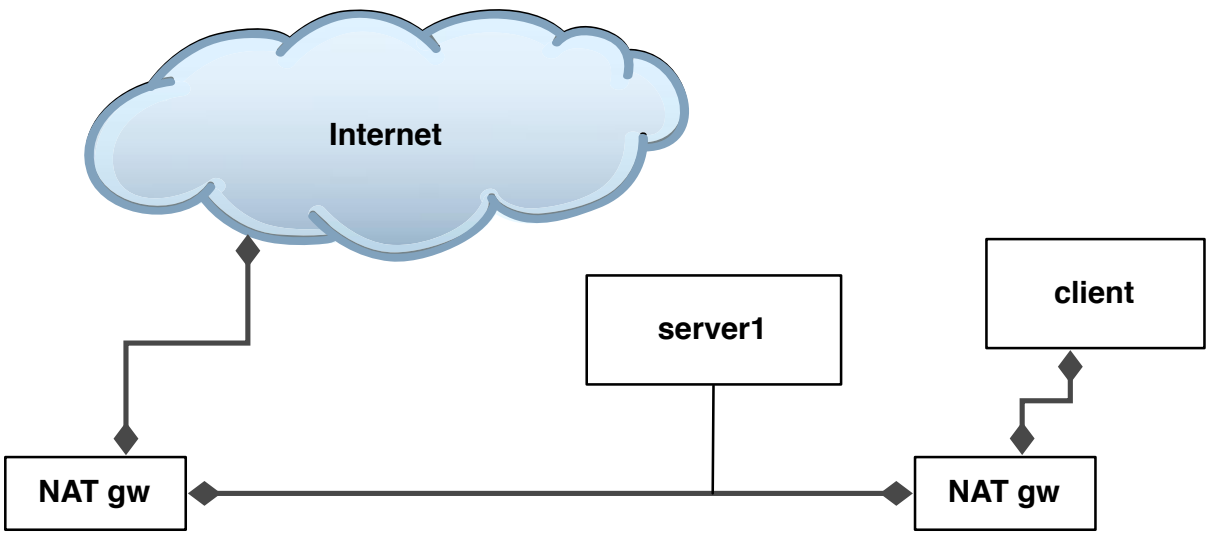
Idag findes proxy applikationer til de mest almindelige funktioner

Den typiske proxy er en caching webproxy der kan foretage HTTP request på vegne af arbejdsstationer og gemme resultatet

NB: nogle protokoller egner sig ikke til proxy servere

SSL forbindelser til *secure websites* kan per design ikke proxies

Anti-pattern dobbelt NAT i eget netværk



Det er nødvendigt med NAT for at oversætte traffik der sendes videre ud på internet.

Der er ingen som helst grund til at benytte NAT indenfor eget netværk!

IPsec og Andre VPN features

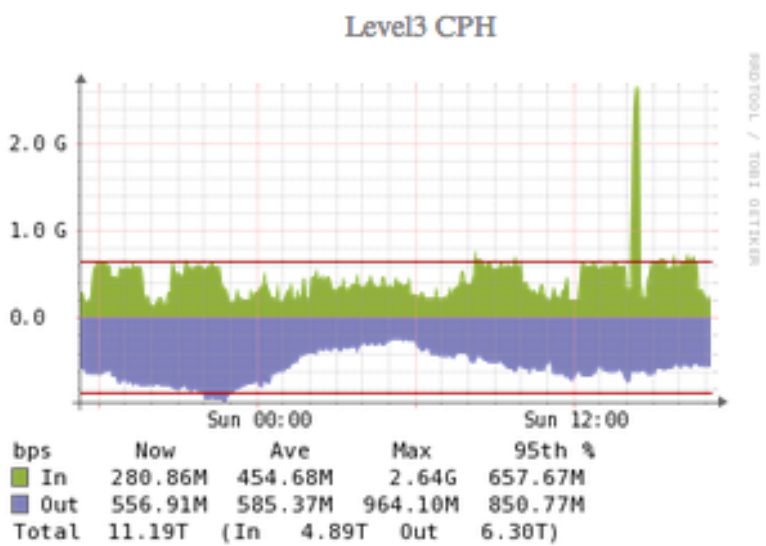
De fleste firewalls giver mulighed for at lave krypterede tunneler

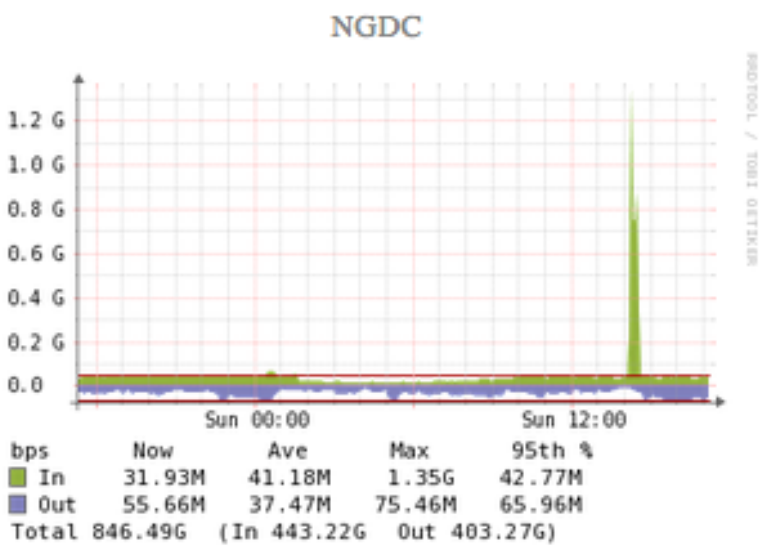
Nyttigt til fjernkontorer der skal have usikker traffik henover usikre netværk som Internet

Konceptet kaldes Virtual Private Network VPN

IPsec er de facto standarden for VPN og beskrevet i RFC'er

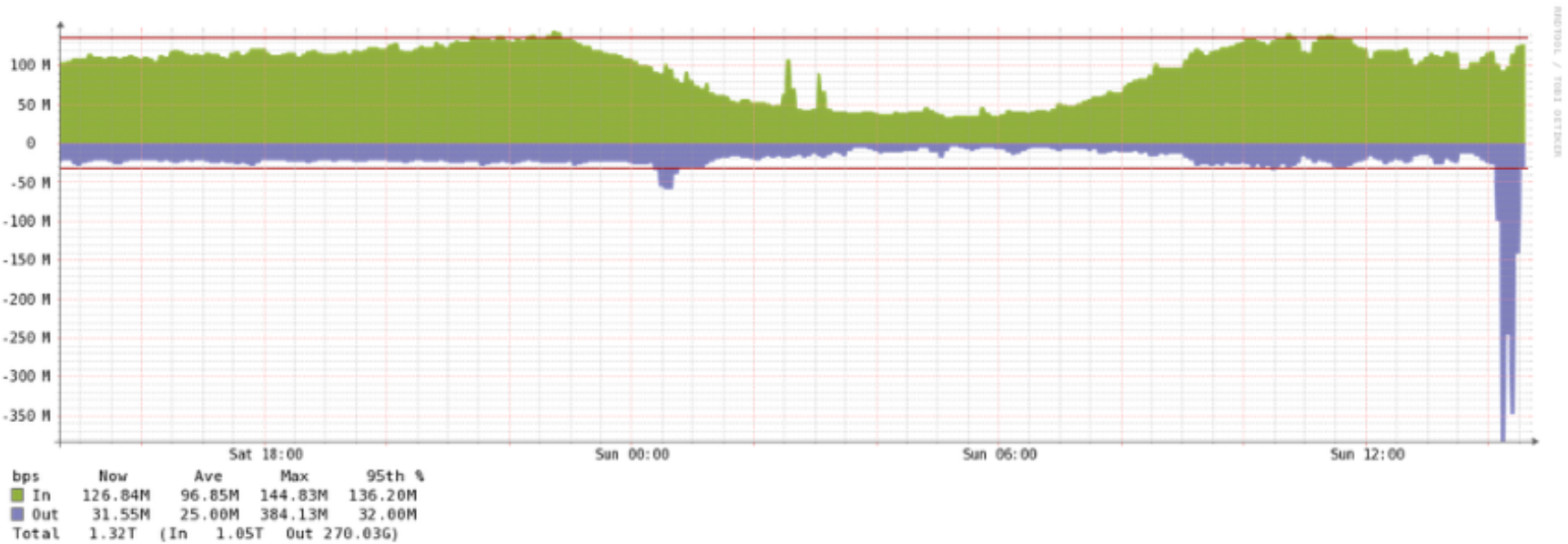
DDoS traffic before filtering





Only two links shown, at least 3Gbit incoming for this single IP

DDoS traffic after filtering



Link toward server (next level firewall actually) about 350Mbit outgoing

Better to filter stateless before traffic reaches firewall, less work!

Stateless firewall filter throw stuff away

```
hlk@MX-CPH-02> show configuration firewall filter all | no-more
/* This is a sample, better to use BGP flowspec and RTBH */
inactive: term edgeblocker {
    from {
        source-address {
            84.180.xxx.173/32;
...
            87.245.xxx.171/32;
        }
        destination-address {
            91.102.91.16/28;
        }
        protocol [ tcp udp icmp ];
    }
    then {
        count edge-block;
        discard;
    }
}
```

Hint: can also leave out protocol and then it will match all protocols

Stateless firewall filter limit protocols

```
term limit-icmp {
    from {
        protocol icmp;
    }
    then {
        policer ICMP-100M;
        accept;
    }
}
term limit-udp {
    from {
        protocol udp;
    }
    then {
        policer UDP-1000M;
        accept;
    }
}
```

Routers also have extensive Class-of-Service (CoS) tools today

Strict filtering for some servers, still stateless!

```
term some-server-allow {
    from {
        destination-address {
            109.238.xx.0/xx;
        }
        protocol tcp;
        destination-port [ 80 443 ];
    } then accept;
}
term some-server-block-unneeded {
    from {
        destination-address {
            109.238.xx.0/xx; }
        protocol-except icmp;  }
    then { count some-server-block; discard;
    }
}
```

Wut - no UDP, yes UDP service is not used on these servers

Firewalls - screens, IDS like features

When you know regular traffic you can decide:

```
hlk@srx-kas-05# show security screen ids-option untrust-screen
icmp {
    ping-death;
}
ip {
    source-route-option;
    tear-drop;
}
tcp {    Note: UDP flood setting also exist
    syn-flood {
        alarm-threshold 1024;
        attack-threshold 200;
        source-threshold 1024;
        destination-threshold 2048;
        timeout 20;
    }
    land;
} Always select your own settings YMMV
```

Routing RTBH Realtime Blackhole

What about a really big DDoS?

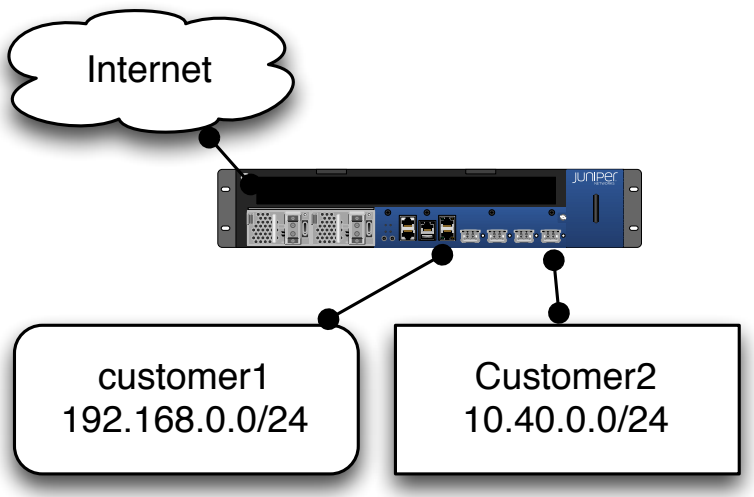
and routers can do more

uRPF unicast Reverse Path Forwarding

> Reverse path forwarding (RPF) is a technique used in modern routers for the purposes of ensuring loop-free forwarding of multicast packets in multicast routing and to help prevent IP address spoofing in unicast routing.

Source: `http://en.wikipedia.org/wiki/Reverse_path_forwarding`

Strict vs loose mode RPF




```
user@router# show interfaces
ge-0/0/0 {
        unit 2 {
                family inet {
                        rpf-check fail-filter rpf-special-case-dhcp;
                        address 192.168.0.254/24;
                }
        }
}
ge-0/0/1 {
        unit 2 {
                family inet {
                        rpf-check fail-filter rpf-special-case-dhcp;
                        address 10.40.0.254/24;
                }
        }
}
```

Configuring Unicast RPF Strict Mode

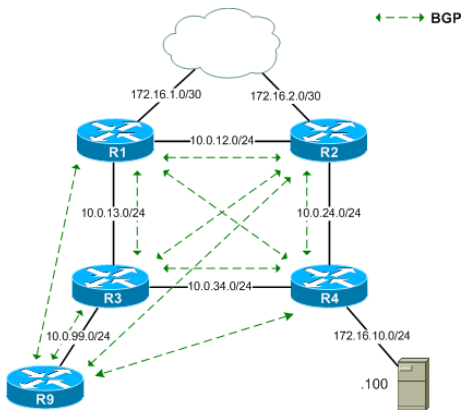
In strict mode, unicast RPF checks whether the incoming packet has a source address that matches a prefix in the routing table, **and whether the interface expects to receive a packet with this source address prefix.**

uRPF Junos config with loose mode

```
xe-5/1/1 {
    description "Transit: Blah (AS65512)";
    unit 0 {
        family inet {
            rpf-check {
                mode loose;
            }
            filter {
                input all;
                output all;
            }
            address xx.yy.xx.yy/30;
        }
        family inet6 {
            rpf-check {
                mode loose;
            }
            address 2001:xx:yy/126;
} } }
```

See also: `http://www.version2.dk/blog/den-danske-internettrafik-og-bgp-49401`

Remotely Triggered Black Hole Configurations



Picture from packetlife.net showing R9 as a standalone "management"router for route injection.

http://packetlife.net/blog/2009/jul/6/remotely-triggered-black-hole-rtbh-routing/

https://ripe65.ripe.net/presentations/285-inex-ripe-routingwg-amsterdam-2012-09-27.pdf https://www.inex.ie/rtbh

Remotely Triggered Black Hole at upstreams

```
6.  Black Hole Server (Optional)
   ###############################################################################
   #                            NOTE                                             #
   #  The Cogent Black Hole server will allow customers to announce a /32 route  #
   #  to Cogent and have all traffic to that network blocked at Cogents backbone.#
   #  All peers on the Cogent black hole server require a password and IP address#
   #  from your network for Cogent to peer with.                                 #
   ###############################################################################

       [   ]  Please set up a BGP peer on the Cogent Black Hole server
       Black Hole server password:
       Black Hole server peer IP:

       North American Black Hole Peer:  66.28.8.1
       European Black Hole Peer:  130.117.20.1
```

Source:

http://cogentco.com/files/docs/customer_service/guide/bgpq.sample.txt

Better drop single /32 host than whole network!

More information about DDoS testing

More DDoS and testing for DDoS can be found in this presentation:

Simulated DDoS Attacks, Breaking the Firewall Infrastructure Henrik Lund Kramshøj

DDoS Attacks have become a daily annoyance for many, and we need to create robust infrastructure. This tutorial will go through a proposed method for testing your own infrastructure using off-the-shelf tools like packet generators hping3 and t50 on Kali Linux.

The goal for the tutorial is to explain: * How to create DDoS attack simulations * My actual experience with doing this - testing banks, etc. * Evaluate how good is this, value proposition for you

Can be found at `https://ripe72.ripe.net/wp-content/uploads/presentations/32-simulated-ddos-ripe.pdf` or

`https://github.com/kramse/security-courses/tree/master/presentations/pentest/simulated-ddos-ripe`

Bonus exercise

Try running hping3 from the DDoS presentation, if we have time.

The Spamhaus Don't Route Or Peer Lists

The Spamhaus Don't Route Or Peer Lists

DROP (Don't Route Or Peer) and EDROP are advisory "drop all traffic"lists, consisting of stolen 'hijacked' netblocks and netblocks controlled entirely by criminals and professional spammers. DROP and EDROP are a tiny subset of the SBL designed for use by firewalls and routing equipment.

`http://www.spamhaus.org/drop/`

Firewalls og IPv6

Læg mærke til forskellen mellem ARP og ICMPv6

Hvis det er muligt lav een regel der tillader adgang til services uanset protokol

NB: husk at aktivere IP forwarding når I skal lave en firewall

IPv6 neighbor discovery protocol (NDP)

OSI	IPv4	IPv6
Network	IP / ICMP	IPv6 / ICMPv6
Link	ARP	
Physical	Physical	Physical

ARP er væk

NDP erstatter og udvider ARP, Sammenlign `arp -an` med `ndp -an`

Til dels erstatter ICMPv6 således DHCP i IPv6, DHCPv6 findes dog

NB: bemærk at dette har stor betydning for firewallregler!

ARP vs NDP

```
hlk@bigfoot:basic-ipv6-new$ arp -an
? (10.0.42.1) at 0:0:24:c8:b2:4c on en1 [ethernet]
? (10.0.42.2) at 0:c0:b7:6c:19:b on en1 [ethernet]
hlk@bigfoot:basic-ipv6-new$ ndp -an
Neighbor                        Linklayer Address  Netif Expire    St Flgs Prbs
::1                             (incomplete)         lo0 permanent R
2001:16d8:ffd2:cf0f:21c:b3ff:fec4:e1b6 0:1c:b3:c4:e1:b6 en1 permanent R
fe80::1%lo0                     (incomplete)         lo0 permanent R
fe80::200:24ff:fec8:b24c%en1  0:0:24:c8:b2:4c      en1 8h54m51s  S  R
fe80::21c:b3ff:fec4:e1b6%en1  0:1c:b3:c4:e1:b6     en1 permanent R
```

OpenBSD PF IPv6 NDP

```
# Macros: define common values, so they can be referenced and changed easily.
int_if=vr0
ext_if=vr2
tunnel_if=gif0
table <homenet6>  2001:16d8:ffd2:cf0f::/64
set skip on lo0
scrub in all
# Filtering: the implicit first two rules are
block in all

# allow ICMPv6 for NDP
# server with configured IP address and router advertisement daemon running
pass in inet6 proto ipv6-icmp all icmp6-type neighbradv keep state
pass out inet6 proto ipv6-icmp all icmp6-type routersol keep state

# client which uses autoconfiguration would use this instead
#pass in inet6 proto ipv6-icmp all icmp6-type routeradv keep state
#pass out inet6 proto ipv6-icmp all icmp6-type neighbrsol keep state

...  probably not working AS IS
```

IPv4 + IPv6 serviceeksempel, med tables

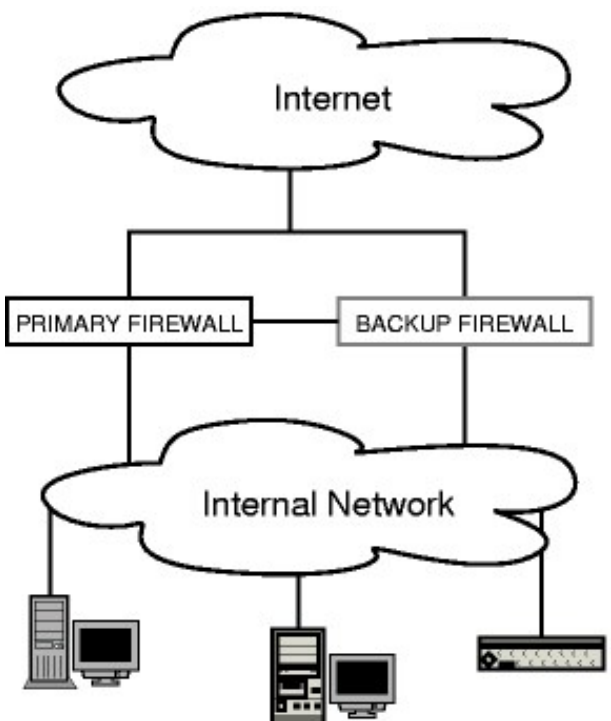
PF har tabeller og simpel regel til PF konfigurationsfilen kan give adgang til både IPv4 og IPv6:

```
table <webservers>  2001:7b8:3e4:72::20 10.0.72.20
...
pass in on $ext_if proto tcp to <webserver> port http
```

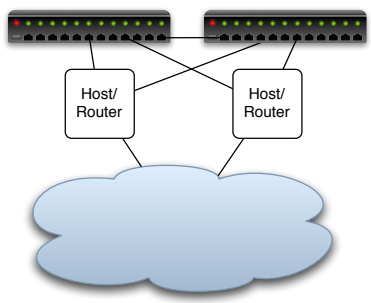
Redundante firewalls

.

- Mange producenter giver mulighed for redundante firewalls/routere
- Eksempler VRRP, CARP, HSRP Cisco, VARP Arista
- OpenBSD Common Address Redundancy Protocol CARP - både IPv4 og IPv6 overtagelse af adresse både IPv4 og IPv6
- pfsync - sender opdateringer om firewall states mellem de to systemer



Redundante forbindelser IP-niveau



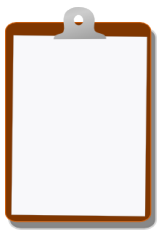
HSRP Hot Standby Router Protocol, Cisco protokol, RFC-2281

VRRP Virtual Router Redundancy Protocol, IETF RFC-3768, åben standard - ikke fri

CARP Common Address Redundancy Protocol, findes på OpenBSD og FreeBSD

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

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