Welcome to

9. Confinement and isolation

KEA Kompetence Computer Systems Security

Henrik Kramselund he/him han/ham xhek@kea.dk @kramse

Slides are available as PDF, kramse@Github

9-isolation.tex in the repo security-courses

Plan for today

Subjects

- Confinement and isolation
- Virtual Machines and Sandboxes
- Covert Channels
- Firewall Flow Controls

Exercises

- Research VM escapes
- Try running a Docker container

Reading Summary

Bishop chapter 18: Confinement Problem, except 18.3.1.3 until 18.3.2.3, skip math

Capsicum: practical capabilities for UNIX

Removing ROP Gadgets from OpenBSD

Information Flow Policies

Define the way information moves throughout a system

Describes the authorized paths along which that information can flow

- Preserve confidentiality
- Preserve integrity

Can be done using labels representing security class, as in Bell-PaPadula Model

Goals for today: Confinement and isolation

Todays goals:

- Known the problem of keeping applications and data confined
- Discuss isolation - including sandboxes, virtual machines and capabilities
- Introduce network isolation, VLAN, firewalls

The confinement problem

Definition 18-1 The *confinement problem* is the problem of preventing a server from leaking information that the user of the service considers confidential.

A covert channel

Definition 18-2 A *covert channel* is a path of communication that was not designed to be used for communication.

The rule of transitive confinement

Definition 18-3 The *rule of transitive confinement* states that if a confined process invokes a second process, the second process must be as confined as the caller.

Isolation

> A controlled environment is an environment that contains process execution in such a way that it can only interact with other entities in a manner that preserves its isolation.

Quote from Matt Bishop, Computer Security 2019

Virtual Machines

Definition 18-4 A *virtual machine* is a program that simulates the hardware of a (possibly abstract) computer system.

Also called hypervisor

Common technologies include VMware, Virtualbox, HyperV, Qemu, KVM

Qubes OS uses the Xen Project `https://xenproject.org/`

Also similarities to sandboxes implemented in Java Virtual Machine (JVM) and other places

Sandbox definition

Definition 18-6 A *sandbox* is an environment in which the actions of a process are restricted according to a security policy

Mentions firewall, which is why we also discuss these later today

Chroot, Jails and

Der findes mange typer *jails* på Unix

Ideer fra Unix chroot som ikke er en egentlig sikkerhedsfeature

- Unix chroot - bruges stadig, ofte i daemoner som OpenSSH
- FreeBSD Jails
- SELinux
- Solaris Containers og Zones
- VMware virtuelle maskiner, er det et jail?

Hertil kommer et antal andre måder at adskille processer - sandkasser

Husk også de simple, database som `_postgresql`, Tomcat som `tomcat`, Postfix postsystem som `_postfix`, SSHD som `sshd` osv. - simple brugere, få rettigheder

JVM security policies

```
// ========== WEB APPLICATION PERMISSIONS ====================================
// These permissions are granted by default to all web applications
// In addition, a web application will be given a read FilePermission
// and JndiPermission for all files and directories in its document root.
grant {
    // Required for JNDI lookup of named JDBC DataSource's and
    // javamail named MimePart DataSource used to send mail
    permission java.util.PropertyPermission "java.home", "read";
    permission java.util.PropertyPermission "java.naming.*", "read";
    permission java.util.PropertyPermission "javax.sql.*", "read";
...
};
// The permission granted to your JDBC driver
// grant codeBase "jar:file:${catalina.home}/webapps/examples/WEB-INF/lib/driver.jar!/-" \{
//      permission java.net.SocketPermission "dbhost.mycompany.com:5432", "connect";
// \};
```

Eksempel fra apache-tomcat-6.0.18/conf/catalina.policy

Apple sandbox named generic rules

```
;; named - sandbox profile
;; Copyright (c) 2006-2007 Apple Inc.  All Rights reserved.
;;
;; WARNING: The sandbox rules in this file currently constitute
;; Apple System Private Interface and are subject to change at any time and
;; without notice. The contents of this file are also auto-generated and not
;; user editable; it may be overwritten at any time.
;;
(version 1)
(debug deny)

(import "bsd.sb")

(deny default)
(allow process*)
(deny signal)
(allow sysctl-read)
(allow network*)
```

Apple sandbox named specific rules

```
;; Allow named-specific files
(allow file-write* file-read-data file-read-metadata
  (regex "^(/private)?/var/run/named\\pid$"
         "^/Library/Logs/named\\.log$"))

(allow file-read-data file-read-metadata
  (regex "^(/private)?/etc/rndc\\.key$"
         "^(/private)?/etc/resolv\\.conf$"
         "^(/private)?/etc/named\\.conf$"
         "^(/private)?/var/named/"))
```

Eksempel fra `/usr/share/sandbox` på Mac OS X

Capability-based security

Capability-based security is a concept in the design of secure computing systems, one of the existing security models. A capability (known in some systems as a key) is a communicable, unforgeable token of authority. It refers to a value that references an object along with an associated set of access rights. A user program on a capability-based operating system must use a capability to access an object. Capability-based security refers to the principle of designing user programs such that they directly share capabilities with each other according to the principle of least privilege, and to the operating system infrastructure necessary to make such transactions efficient and secure. Capability-based security is to be contrasted with an approach that uses hierarchical protection domains.

`https://en.wikipedia.org/wiki/Capability-based_security`

Capsicum in FreeBSD

Implementation status

Capsicum for FreeBSD was implemented by Robert Watson and Jonathan Anderson. Capsicum first appeared in FreeBSD 9.0 as an experimental feature, compiled out of the kernel by default. As of FreeBSD 10.0, Capsicum capability mode, capabilities, and process descriptors are compiled into the kernel by default, and available for use by both base-system and third-party applications.

Significant KPI and API changes were made in FreeBSD 10.0 following several years' experience deploying Capsicum in experimental applications and the FreeBSD base system. In FreeBSD 10.0, a number of base-system applications use Capsicum "out of the box"including tcpdump, auditdistd, hastd, dhclient, kdump, rwhod, ctld, iscsid, and even uniq.

Quote from:
`https://www.cl.cam.ac.uk/research/security/capsicum/freebsd.html`

Covert Channels

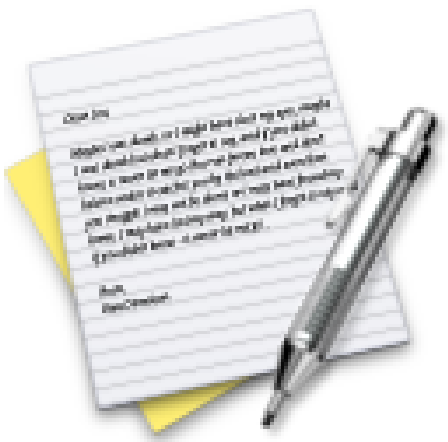
Definition 18-7 A *covert storage channel* uses an attribute of the shared resource. A *covert timing channel* uses a temporal or ordering relationship among access to a shared resource

Example the recent Intel CPU attacks meltdown etc.

Book describes ways to detect and mitigate these covert channels, but it is hard

Chapter 17 - which we haven't read discuss information flow, which can be analyzed

Exercise

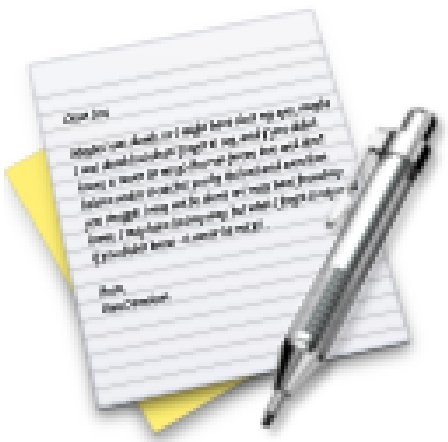


Now lets do the exercise

Research Virtual Machine Escapes 20min

which is number **27** in the exercise PDF.

Exercise



Now lets do the exercise

Try running a Docker container 20min

which is number **28** in the exercise PDF.

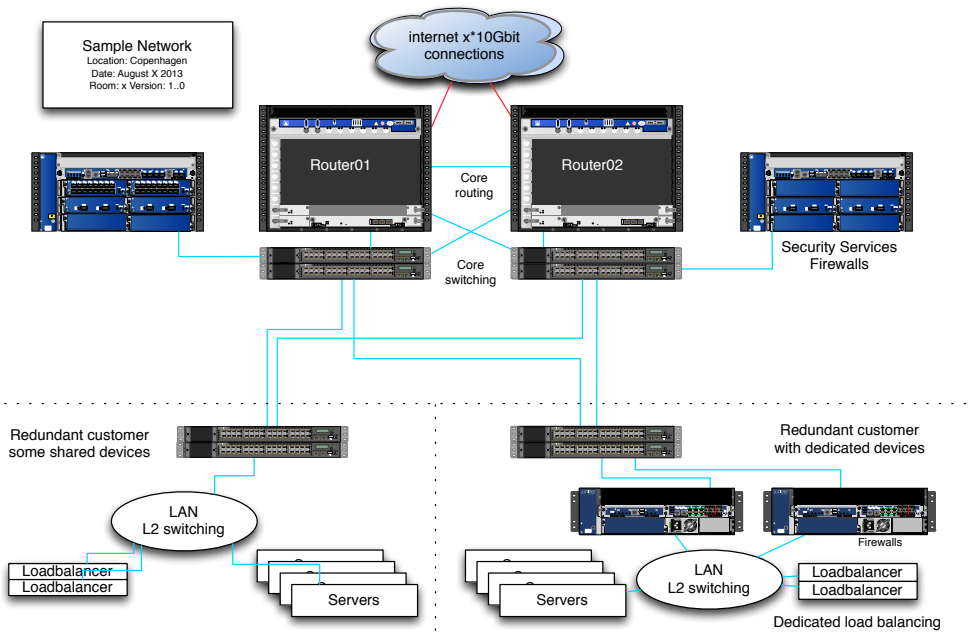
Design a robust network Isolation and segmentation

Hvad kan man gøre for at få bedre netværkssikkerhed?

- Bruge switche - der skal ARP spoofes og bedre performance
- Opdele med firewall til flere DMZ zoner for at holde udsatte servere adskilt fra hinanden, det interne netværk og Internet
- Overvåge, læse logs og reagere på hændelser

Husk du skal også kunne opdatere dine servere

Firewall Flow Controls – *the firewall infrastructure*



Conclusion: Do as much as possible with your existing devices

Tuning and using features like stateless router filters works wonders

Firewalls and related issues

> 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

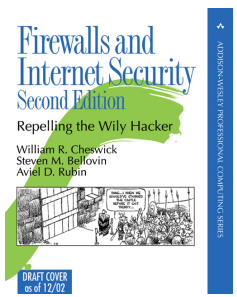
`https://en.wikipedia.org/wiki/Firewall_(computing)`

`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!

Firewall historik

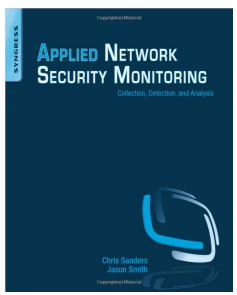


Firewalls har været kendt siden starten af 90'erne

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

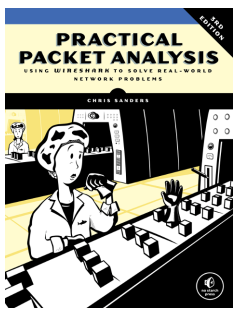
Book: Applied Network Security Monitoring (ANSM)



Applied Network Security Monitoring: Collection, Detection, and Analysis 1st Edition, Chris Sanders, Jason Smith
eBook ISBN: 9780124172166 Paperback ISBN: 9780124172081 496 pp., Syngress, December 2013

`https://www.elsevier.com/books/applied-network-security-monitoring/unknown/978-0-12-417208-1`

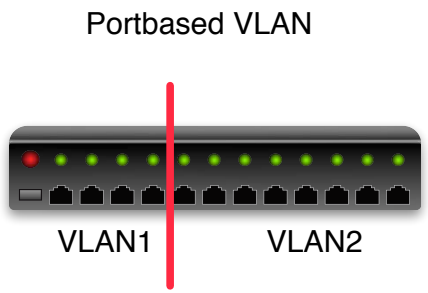
Book: Practical Packet Analysis (PPA)



Practical Packet Analysis, Using Wireshark to Solve Real-World Network Problems
by Chris Sanders, 3rd Edition April 2017, 368 pp. ISBN-13: 978-1-59327-802-1

`https://nostarch.com/packetanalysis3`

Together with Firewalls - VLAN Virtual LAN



Nogle switche tillader at man opdeler portene

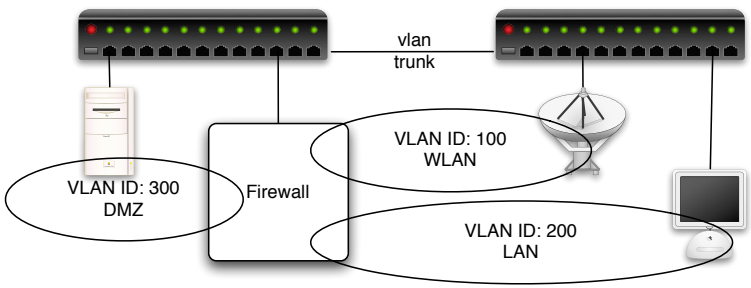
Denne opdeling kaldes VLAN og portbaseret er det mest simple

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

Basic Network Security Pattern Isolate in VLANs



Du bør opdele dit netværk i segmenter efter trafik, IEEE 802.1q VLANs er mulighed

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

Du bør altid holde interne og eksterne systemer adskilt!

Du bør isolere farlige services i jails og chroots

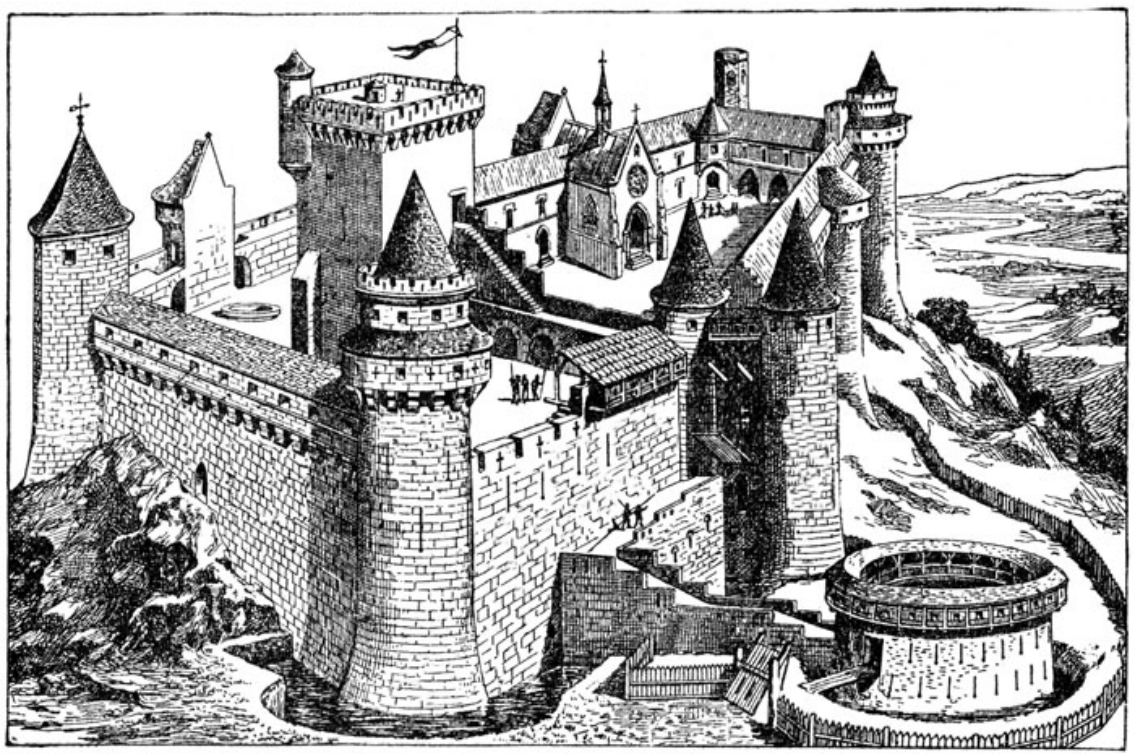
Brug port security til at sikre basale services DHCP, Spanning Tree osv.

Generic IP Firewalls

En firewall er noget som blokerer traffik på Internet

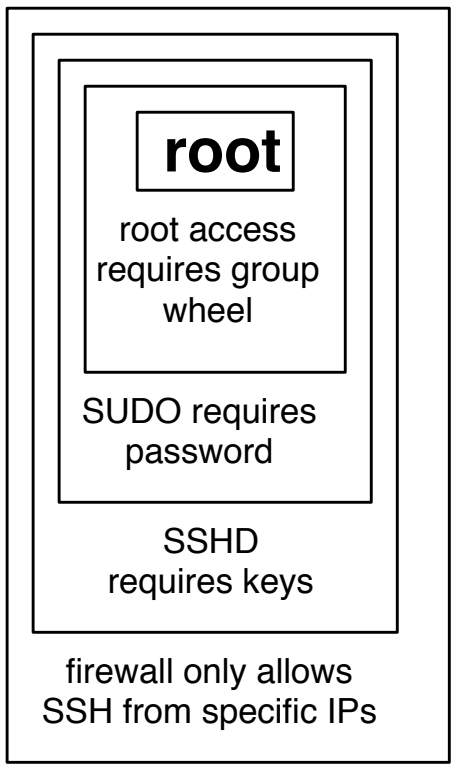
En firewall er noget som tillader traffik på Internet

Defense in depth



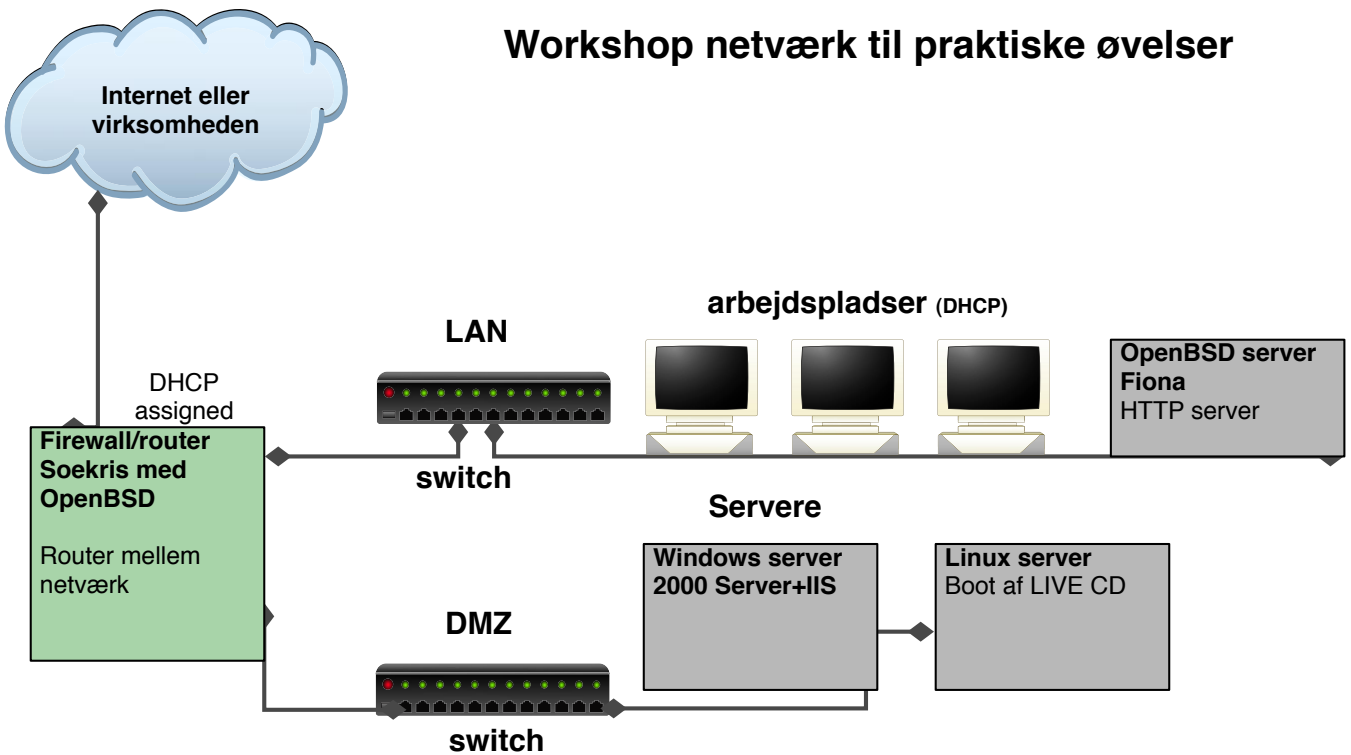
Picture originally from: `http://karenswhimsy.com/public-domain-images`

Defense in depth - layered security

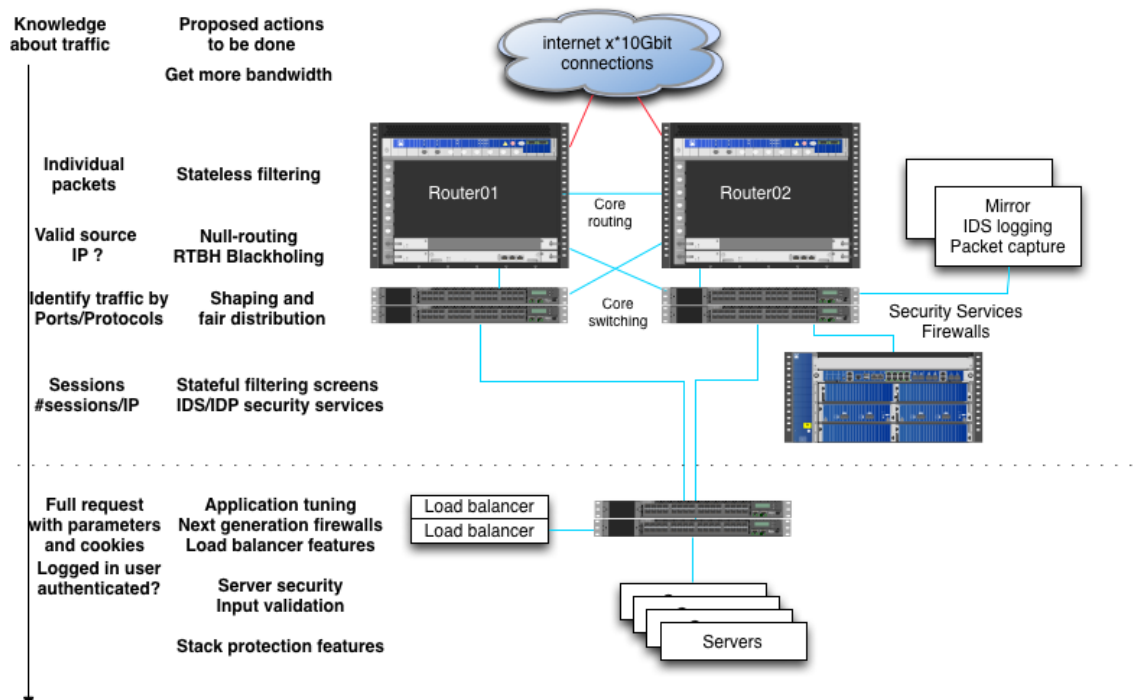


Multiple layers of security! Isolation!

Small networks

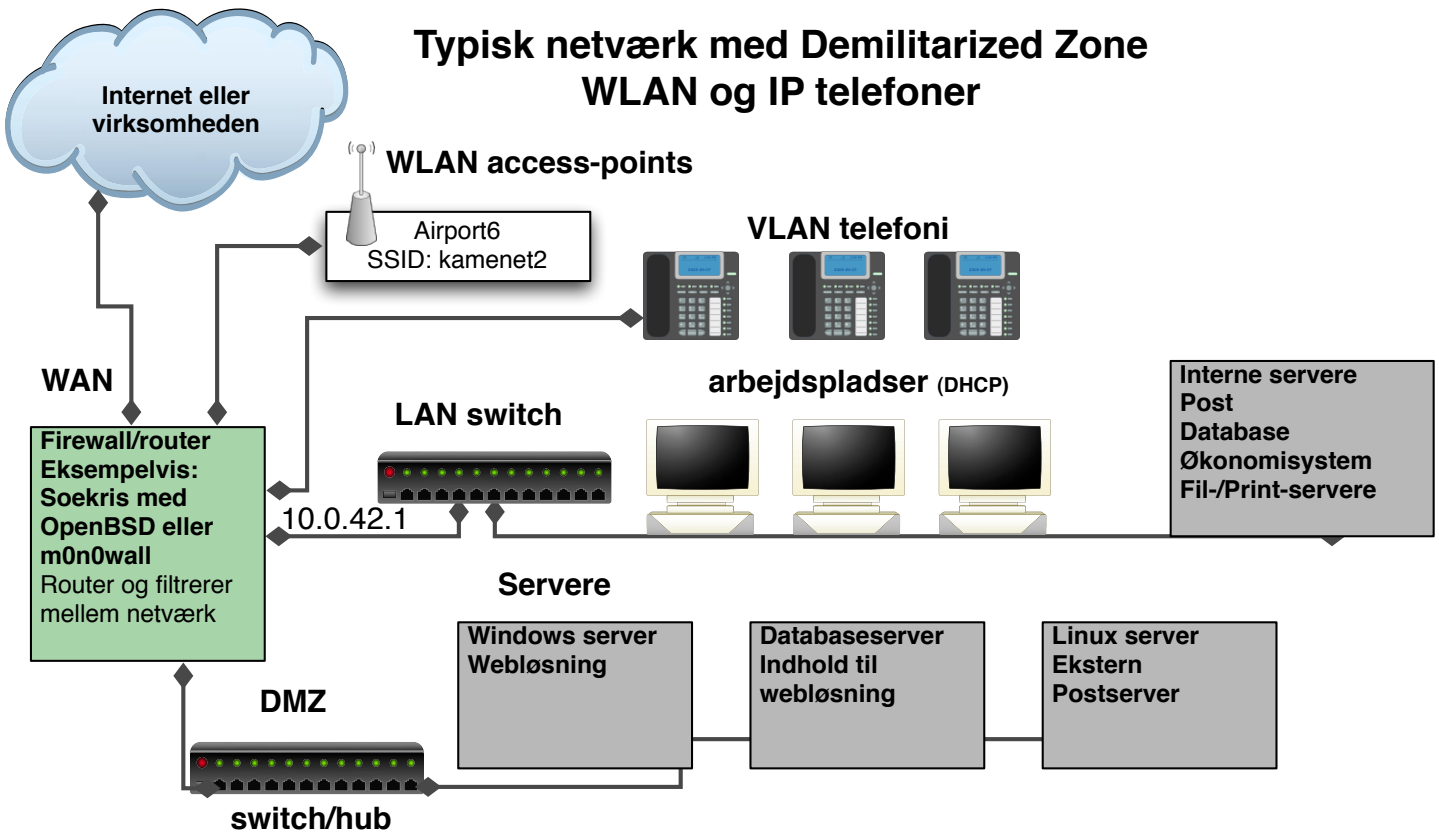


Firewall er ikke alene



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

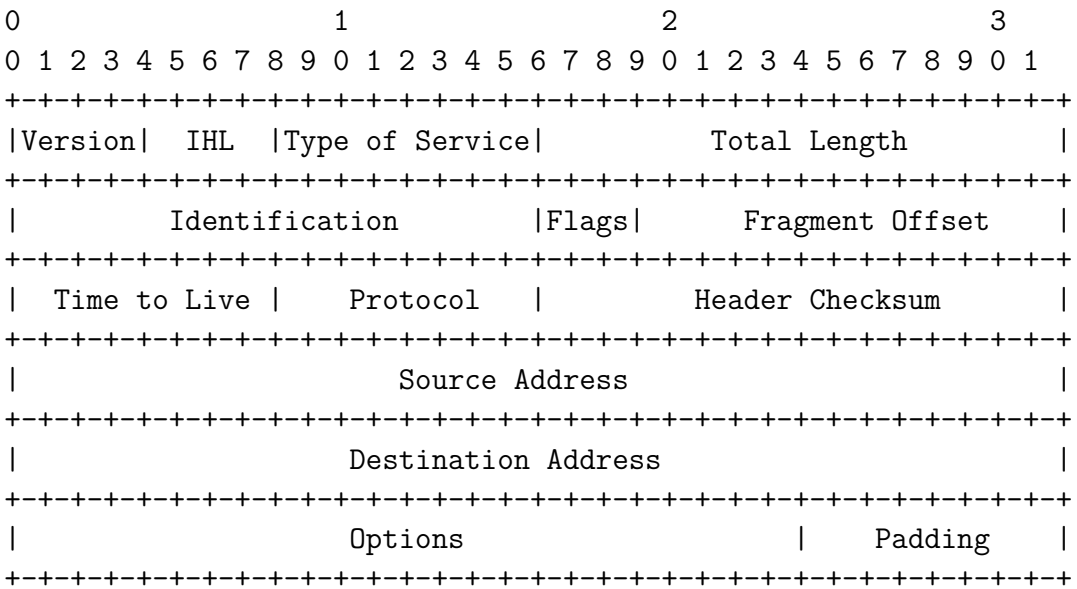
```
# hosts and networks
router="217.157.20.129"
webserver="217.157.20.131"
homenet=" 192.168.1.0/24, 1.2.3.4/24 "
wlan="10.0.42.0/24"
wireless=wi0
set skip lo0
# things not used
spoofed=" 127.0.0.0/8, 172.16.0.0/12, 10.0.0.0/16, 255.255.255.255/32 "

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
```

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

Der findes også mange gode open source baserede firewalls

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 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

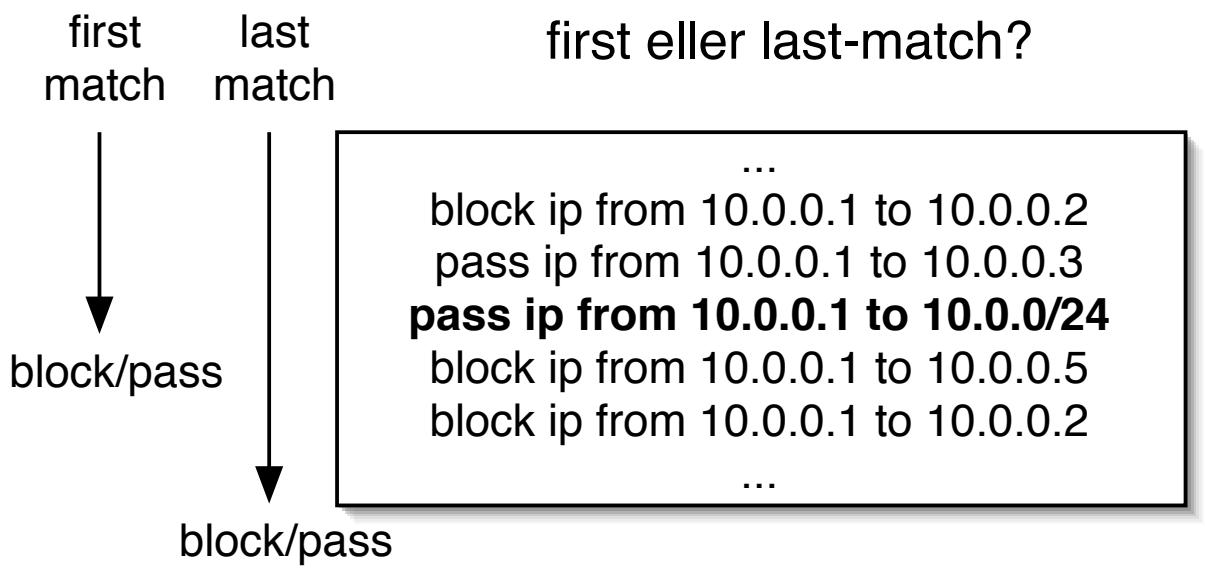
Det er ekstremt vigtigt at vide hvilken type firewall man bruger!

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

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

Der er helt tilsvarende nogle ICMPv6 beskeder som bør tillades i firewalls

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

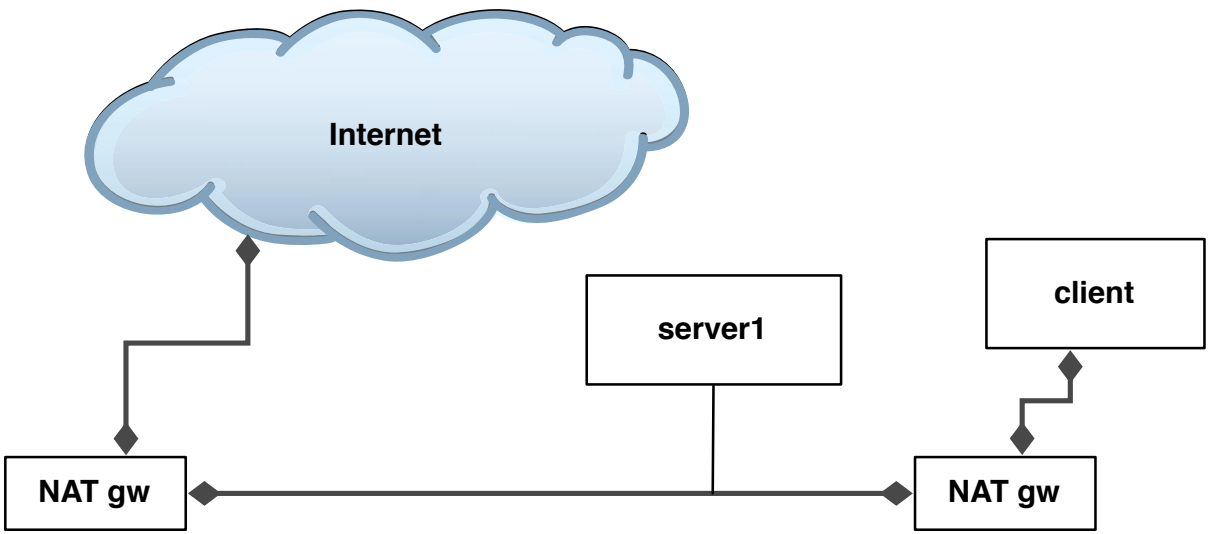
Der er porte og services som altid bør blokeres

Det kan være kendte sårbare services

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

Kendte problemer som efter min mening altid skal blokeres

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

Linux TCP SACK PANIC - CVE-2019-11477 et al

kea

Kernel vulnerabilities, CVE-2019-11477, CVE-2019-11478 and CVE-2019-11479

Executive Summary
Three related flaws were found in the Linux kernel's handling of TCP networking. The most severe vulnerability could allow a remote attacker to trigger a kernel panic in systems running the affected software and, as a result, impact the system's availability.

The issues have been assigned multiple CVEs: CVE-2019-11477 is considered an Important severity, whereas CVE-2019-11478 and CVE-2019-11479 are considered a Moderate severity.

The first two are related to the Selective Acknowledgement (SACK) packets combined with Maximum Segment Size (MSS), the third solely with the Maximum Segment Size (MSS).

These issues are corrected either through applying mitigations or kernel patches. Mitigation details and links to RHSA advsories can be found on the RESOLVE tab of this article.

Source: `https://access.redhat.com/security/vulnerabilities/tcpsack`

2019 ERNW White Paper 68 – Cisco ACI

kea

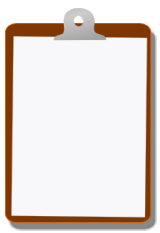
SECURITY ASSESSMENT OF CISCO ACI

Software-defined networking (SDN) along-side with micro-segmentation has been proposed as a new paradigm to deploy applications faster and, simultaneously, protect the individual workloads against lateral movement. One of the major solutions in this area is the Application Centric Infrastructure (ACI) by Cisco. As part of our research endeavors, we performed a security assessment of Cisco ACI. In this whitepaper, we will provide an introduction to Cisco ACI, present the results of our assessment, and give recommendations on how to operate Cisco ACI more securely.

Source: `https://ernw.de/en/whitepapers/issue-68.html`

- Remote Code Execution on Leaf Switches over IPv6 via Local SSH Server (CVE-2019-1836, CVE2019-1803, and CVE-2019-1804)
- Cisco Nexus 9000 Series Fabric Switches ACI Mode Fabric Infrastructure VLAN Unauthorized Access Vulnerability (CVE-2019-1890)
- Cisco Nexus 9000 Series Fabric Switches Application Centric Infrastructure Mode Link Layer Discovery Protocol Buffer Overflow Vulnerability (CVE-2019-1901)
- Cisco Application Policy Infrastructure Controller REST API Privilege Escalation Vulnerability (CVE2019-1889)

For Next Time



Think about the subjects from this time, write down questions

Check the plan for chapters to read in the books

Visit web sites and download papers if needed

Retry the exercises to get more confident using the tools