



Denne forelesningsøkten vil bli tatt opp og lagt ut i emnet i etterkant.

Hvis du ikke vil være med på opptaket:

	La være å delta med webkameraet ditt.
	La være å delta med mikrofonen din.
To: Marianne Sundby (Privately) Type message here...	Still spørsmål i Chat i stedet for som lyd. Hvis du ønsker kan spørsmålet også sendes privat til foreleser.

Påminnelse - Regler for Zoom forelesninger

Jeg fikk takk fra studenter for at jeg hadde tydelige regler, det ble påpekt at det i andre fag har vært mye støy i chat, så jeg minner om:

- Skru av mikrofon
- KUN faglige spørsmål i chat
- Ikke kommenter andres spørsmål i chat, vent til foreleser svarer (Marcus Dahl vil noen dager være med på zoom som hjelpe-lærer; han vil svare spørsmål i chat, han er ikke student :-)
- Ikke snakk med hverandre i chat (det forstyrrer andre)
- I øvingstimene prøver vi å få en bedre dialog, da anbefaler jeg at spørsmål stilles med mikrofon



Høyskolen
Kristiania

TK2100: Informasjonsikkerhet

Tredje forelesning:
OPERATIVSYSTEM

Pensum: Goodrich & Tamassia, kapittel 3

Pensum: Nätt & Heide, kapittel 5

- Operativsystemets (OS)
 - oppgaver, oppbygging, ansvar og begreper
- Booting, pålogging og sikkerhet
- Prosesser og sikkerhet
- Minne og Filsystemet
 - ACLer i NT og OSX
- Applikasjoner og exploits

Repetisjon: KRYPTERING

- Siste ukes emne kryptering er det vanskeligste emnet innen datasikkerhet
- Viktigste felter av kryptering er:
 - Kjenne substitution ciphers, feks ROT13
 - Sikkerheten i One Time Pads
 - Moderne block cipher; AES
 - Data i block ciphers må paddes (PKCS5)
 - Diffie-Hellman; assymetrisk nøkkel utveksling
 - RSA; assymetrisk kryptering
 - SHA-2; sikker hash algoritme
 - OpenSSL; open source krypto bibliotek

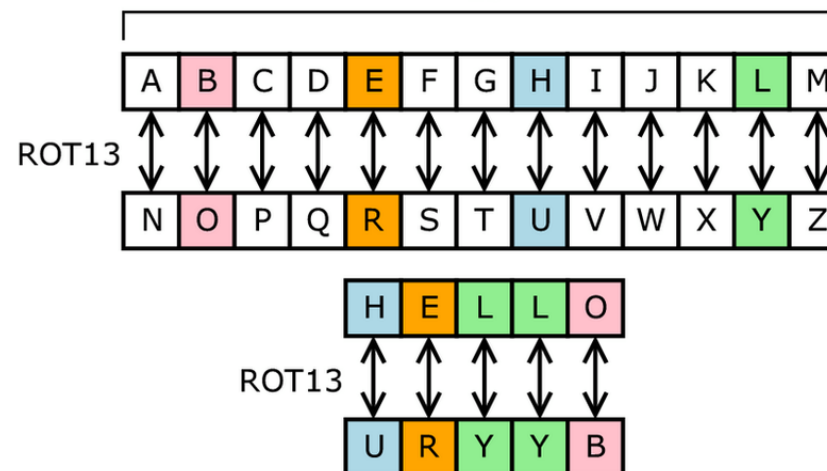
Substitusjons Chiffer

- Hver bokstav erstattes med en annen..
- På engelsk vil det være 26!
 $= 26 \times 25 \times 24 \times 23 \times \dots \times 3 \times 2 \times 1$
 mulige slike koder.
- Det blir 4.03×10^{26} ulike.

- En populær versjon på Internett pleide å være ROT13.

$$C = (P + 13) \% 26$$

$$P = (C + 13) \% 26$$

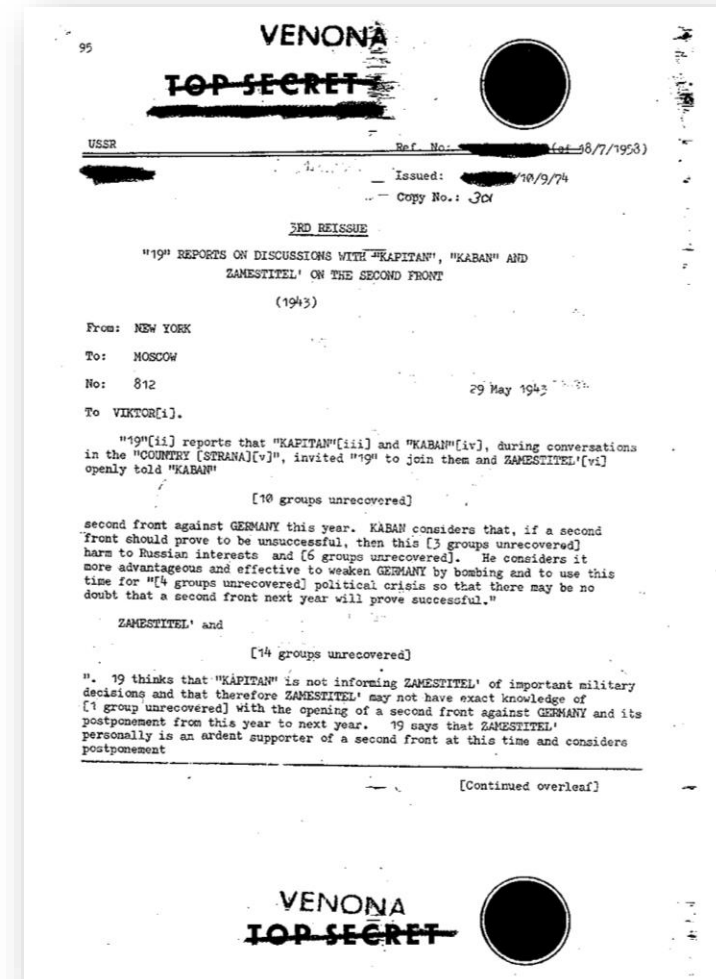


One-Time Pads (Engangs blokk)

- Fullstendig **umulig** å knekke (i prinsippet).
 - Oppfunnet i 1917 av Joseph Mauborgne og Gilbert Vernam
 - Ble funnet en eldre kilde høsten 2013... (1865)
 - Vi benytter en tabell med shift-nøkler, (k_1, k_2, \dots, k_n) , til å kryptere en rentext, M , med lengde n , der hver shift-nøkkel er valgt fullstendig tilfeldig (“uniformt tilfeldig”)
- Siden hvert shift er fullstendig tilfeldig så er ethvert kryptogram like sannsynlig for enhver klartext!

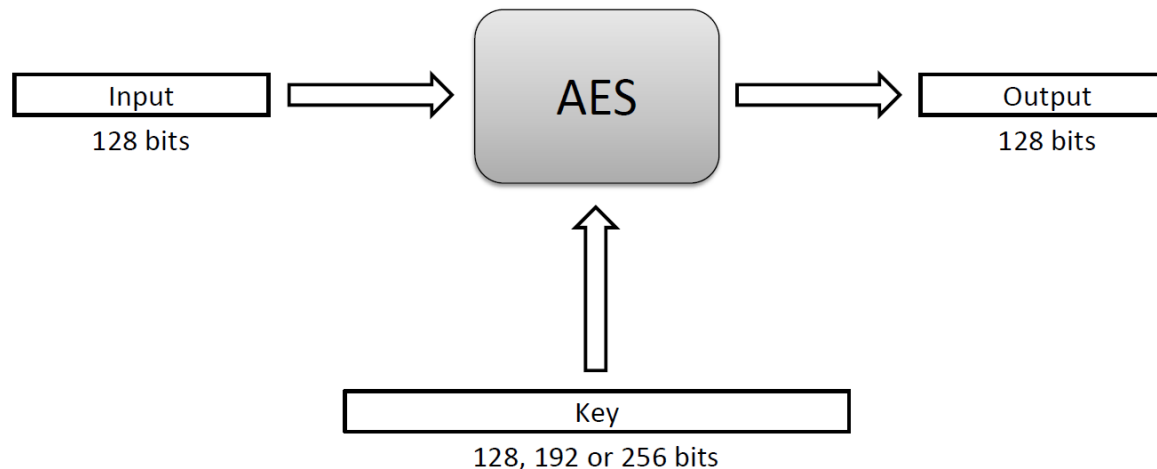
Svakheter ved engangs-blokk

- Nøkkelen må være like lang som klarteksten
- Nøkler må aldri gjenbrukes!!!
 - USA klarte å knekke noen Sovjetiske kryptogrammer statistisk pga gjenbruk av kodeblokk



Advanced Encryption Standard(AES)

- Opprinnelig kalt Rijndael
- Blokkchiffer
- 128 bit blokker
- Versjoner AES-128, AES-192 og AES-256 ut fra nøkkel-lengde
- Veldig kompleks, men designet for datamaskiner!



- Blokkchiffre krever at lengden **n** på klartext må være et mutippel av blokkstørrelsen **b**
- Padding i siste blokk kan ikke være flertydig (kan ikke bare legge til 0'er)
- Når blokk-størrelsen og klartexten er multipler av 8 så er en vanlig padding-metode **PKCS5**: der legges det inn en sekvens med idetiske bytes, som hver er lengden på paddingen
- Eksempel for $b = 128$ (16 bytes)
 - Klartext: "blislog" (7 bytes)
 - "Padda" klartext: "blislog999999999" (16 bytes), der 9 er tallet, ikke tegnet
- Vi vil alltid måtte padde siste blokk, som kan bestå nærmest utelukkende av padding

Diffie-Hellman

- Divisjon er ikke vanskelig nok
- Vi *tror* at diskrete logaritmer er tilstrekkelig vanskelig.
 - Å finne verdien til x når du vet at f.eks. $2^x \% 11 = 3$ krever masse testing, og blir vanskelig når tallene er store!
 - Og se gangetabellen for $\%11$ øverst til venstre...
- Starter med å velge en basis (f.eks. 2) og en modulus (f.eks. 11)
 - Modulus må være et primtall:

	1	2	3	4	5	6	7	8	9	10
1	1	2	3	4	5	6	7	8	9	10
2	2	4	6	8	10	1	3	5	7	9
3	3	6	9	1	4	7	10	2	5	8
4	4	8	1	5	9	2	6	10	3	7
5	5	10	4	9	3	8	2	7	1	6
6	6	1	7	2	8	3	9	4	10	5
7	7	3	10	6	2	9	5	1	8	4
8	8	5	2	10	7	4	1	9	6	3
9	9	7	5	3	1	10	8	6	4	2
10	10	9	8	7	6	5	4	3	2	1

n	2^n	3^n	6^n
1	2	3	6
2	4	9	3
3	8	5	7
4	5	4	9
5	10	1	10
6	9	3	5
7	7	9	8
8	3	5	4
9	6	4	2
10	1	1	1

Offentlige tall:

$\% = 11, b = 2$



Ditt offentlig*private tall: $(2^8)\%11 = 3$

Bobs offentlig*private tall: $(2^9)\%11 = 6$



Privat tall:

8



Privat tall:

9

$(6^8)\%11 = 4 = (3^9)\%11 = 4$

Delt hemmelig tall!!!

RSA: Kryptering, dekryptering

0. Gitt off. krypteringsnøkkel (n,e) og privat (n,d) dekrypteringsnøkkel

1. For å **kryptere** bit mønsteret, m , beregn

$$c = m^{e \bmod n}$$

2. For å **dekryptere** mottatt bit mønster, c , beregn

$$m = c^{d \bmod n}$$

Magi?!

$$m = \underbrace{(m^{e \bmod n})}_c^{d \bmod n}$$

RSA (leke-)eksempel:

- Gitt at dere har en public key hvor $n = 3233$ og $e = 17$, og deres privat key er $n = 3233$ og $d = 2753$. Dekrypter min melding til dere:
- 3000 28 2726 2726 1307 1992 641 2726 2790 2680 2680

$$M = C^d \bmod n = 3000^{2753} \bmod 3233 = 72 = \text{bokstaven 'H'}$$

72	69	76	76	79	32	67	76	65	83	83
H	E	L	L	O		C	L	A	S	S

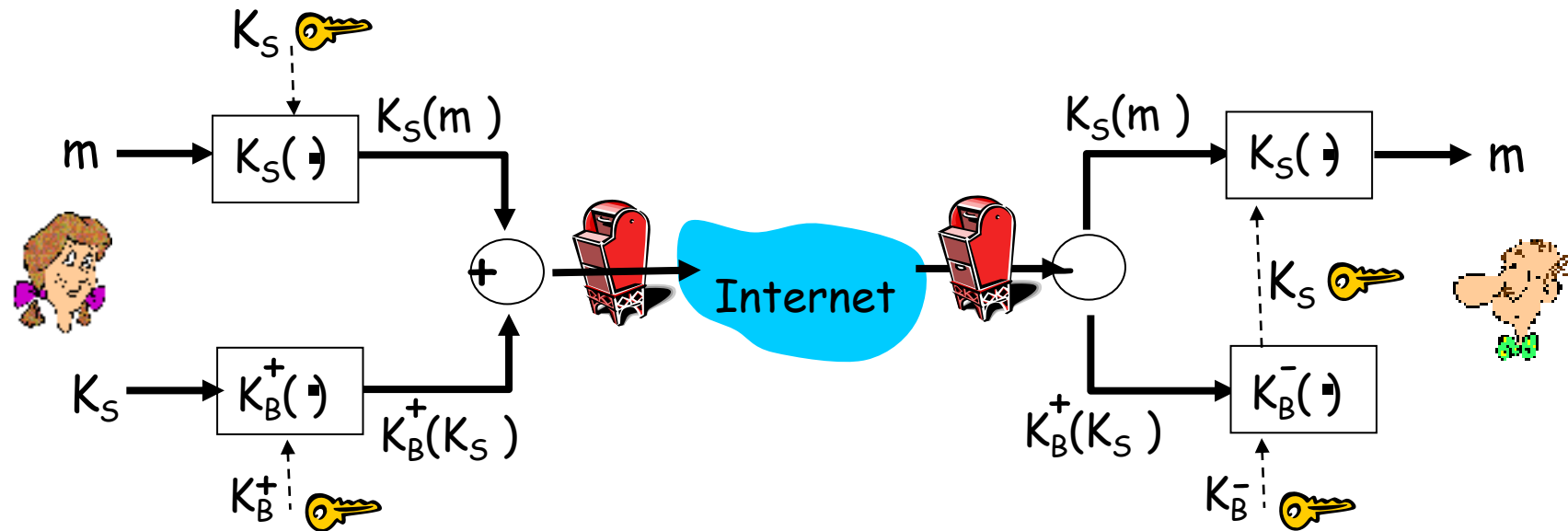
Secure Hash Algorithm (SHA)

- Utviklet ved NSA og godkjent av NIST
- SHA-0 og SHA-1 (1993)
 - 160-bit
 - Regnes som usikker
 - Fremdeles i bruk
 - Mindre sårbar enn MD5
- SHA-2 (2002)
 - 256 bits (SHA-256) eller 512 bits (SHA-512)
 - Finnes publiserte angrepsteknikker, men regnes fremdeles som sikre
- Offentlig konkurranse om SHA-3 startet i 2007
 - Keccak algoritmen ble valgt som SHA-3, men det er mange kontroverser rundt interne endringer som ble tvunget inn av NIST, som gjør at mange i dag ikke stoler på den
 - Brukes ikke, av frykt for at den er bevisst svekket (av NSA?)

- Ved implementasjon av TLS og/eller krypto systemer (i C/C++) anbefales det å bruke OpenSSL biblioteket
- Hvis du har kontroll på både server og klient kan du begrense cipher suites til de sikreste
- Eksempel på "cipher suite" format i TLS 1.2
 TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256
 - Key exchange: ECC (ECDHE)
 - (Server) authentication: RSA
 - Encryption: AES 128, GCM mode
 - Message Authentication Code: SHA256

Sikker epost: kryptering

- Alice sende hemmelig e-mail, m , til Bob.



Alice:

- generer random *symmetrisk* privat nøkkel, K_S .
- krypterer meldingen med K_S (effektivitet)
- Krypterer K_S med Bobs offentlige nøkkel
- sender både $K_S(m)$ og $K_B(K_S)$ til Bob.

Praktisk forsvar av private maskiner

Før vi går løs på en del teori – det viktigste første, hva må man gjøre for å holde en privat PC sikker «nok» som student?

- Grunnleggende beskyttelse alle datamaskiner MÅ ha er anti-virus programvare
- «Common sense» trenger ikke alltid være nok, sårbarheter i programvare kan gjøre deg sårbar ovenfor datavirus (malware) uansett hvor forsiktig du er!
- Det er flere komponenter enn bare anti-virus i en sikkerhetspakke, man bør installere alt som da omfatter brannmur, «threat prevention» og kanskje beskyttelse mot ransomware
- Mer om dette neste uke...

- Hvis «uhellet» er ute må man ha BACKUP
- Backup er eneste helt sikre metode for å ikke miste viktige data etter et dataangrep (eller datavirus infeksjon)
- Backup kan være til SKY eller til et ekstern lagring (feks USB disk)

- For å forhindre at en angriper får kontroll på online kontoer aktiver to-faktor autentisering
- Høyest sikkerhet: fysisk kodebrikke, eller «authenticator app»
- Akseptabel sikkerhet: engangskode til epost eller SMS

- Bevissthet om at det finnes mange trusler, og forvent å «bli angrepet»
 - ✓ Ikke trykk på linker i eposter
 - ✓ Ikke svar på eposter fra personer du ikke kjenner
 - ✓ Ikke sett USB enheter du «finner» inn i PCen din
 - ✓ Ikke oppgi passord eller koder til noen hverken via telefon eller epost
 - ✓ Ikke last ned piratkopierte filer – de inneholder ofte malware
- Beskytt personlig og sensitiv informasjon
- Vær spesielt oppmerksom på info om bankkonto
- Dekk til web kamera – mange blir presset for penger etter at private bilder har blitt stjålet eller lekket ut
- Det er mange på internett som ikke ønsker deg noe godt...

- En sikkerhetsleder / administrator kan ikke stole på at ansatte har «common sense», derfor må man ha sikkerhet til tross for dumme brukere
 - ✓ Brukere må ikke ha admin privilegier
 - ✓ Overvåke nettverket for å stoppe angrep
 - ✓ Opplæring av brukere
 - ✓ mm

Utenfor pensum å gå i dybden i dette faget, se Cyber Defense (CYB2100) i Cybersecurity spesialiseringen

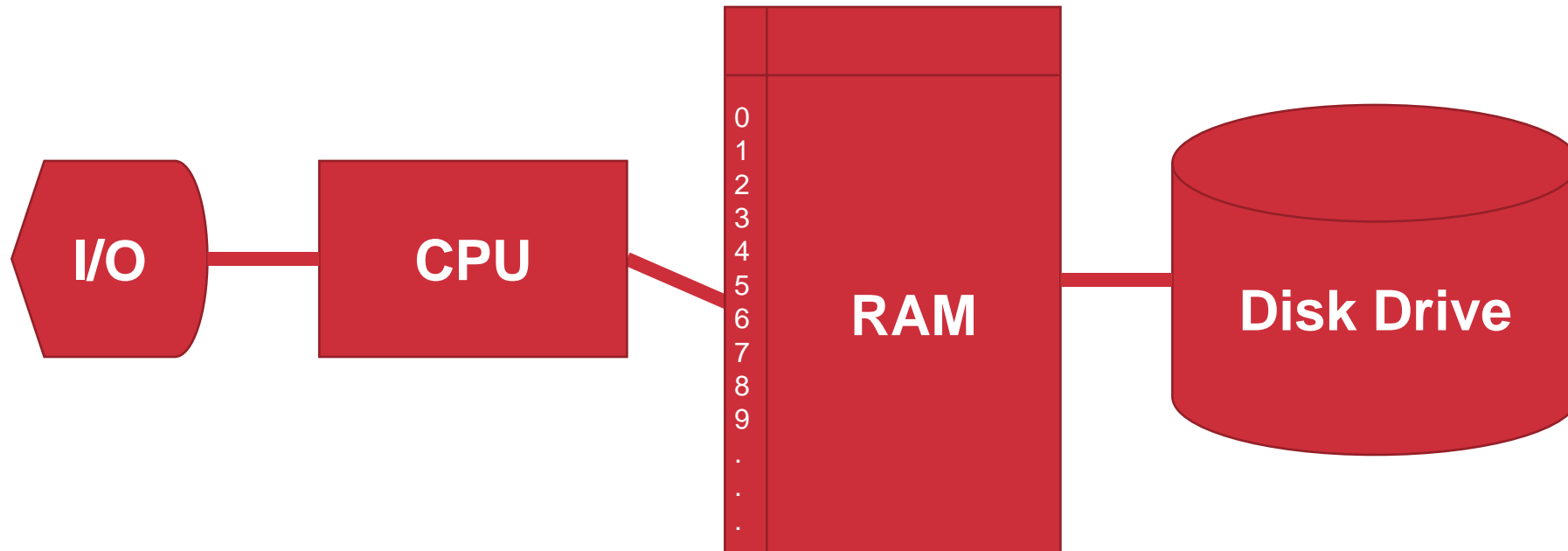
OS Teori og begreper

Hva er operativsystemets rolle i informasjonssikkerhet?

Von Neuman (igjen)



- Operativsystemet må forholde seg til (pålitelig og sikker) bruk av CPU, RAM, I/O-utstyr og persistente lagre (disk...)



- **Operativsystemet** tilbyr et grensesnitt mellom brukerne av en datamaskin og maskinvaren
 - Administrerer hvordan prosesser aksesserer ressurser i datamaskinen på (CPU, RAM, IO-utstyr, nettverkskort,..)
 - Administrerer mange brukere med ulike tilgangsrettigheter
 - Administrerer mange applikasjoner, programmer, prosesser og tjenester

User- vs Kernel-modus

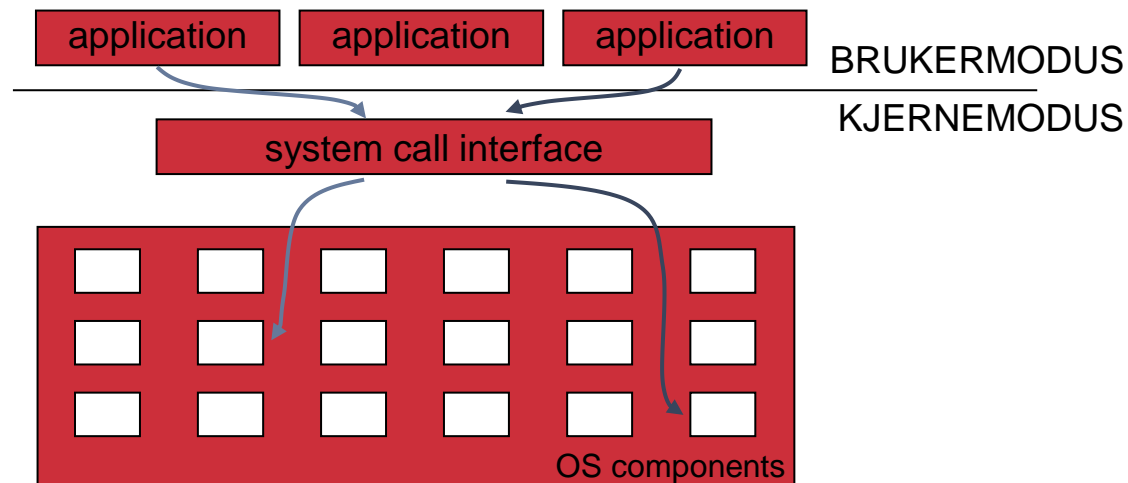
- For å oppnå sikkerhet og **beskyttelse** gir de fleste **CPU**er muligheten til å kjøre instruksjoner enten i **User**-eller **kjerne**-modus
 - Primært pga **mistillit til brukerapplikasjoner**
- Vanlige applikasjoner samt mange OS-tjenester og under-systemer kjører i “**user** mode” (ring 3)
 - real mode
 - Kan **ikke aksessere HW**, utstyrsdrivere direkte , må **bruke en API**
 - Kun adgang til **minnet som OSet har tildelt**
 - Begrenset instruksjonssett
- OS-kjernen kjører i **kjerne**modus (ring 0)
 - protected mode
 - Tilgang til **hele minnet**
 - **Alle instruksjoner** kan kjøres
 - Ingen sikring fra HW

- Applikasjoner får tilgang til operativsystemtjenester ved **systemkall**
 - Innebærer et kontekst-skifte (**context switch**) og (oftest) et skifte fra applikasjon- til kjerne-modus
- 5 klasser av systemkall:
 - Prosess-kontroll
 - Filmanipulering
 - Utstyrshåndtering og –bruk (Device manipulation)
 - Informasjonsvedlikehold (“OS-bokføring”)
 - Kommunikasjon (mellom prosesser) (IPC)

Systemkall



- Grensesnittet mellom OS og brukerne defineres med en mengde systemkall
- Å utføre et systemkall er tilsvarende et vanlig metodekall, men systemkallet kjører kjerne-kode:

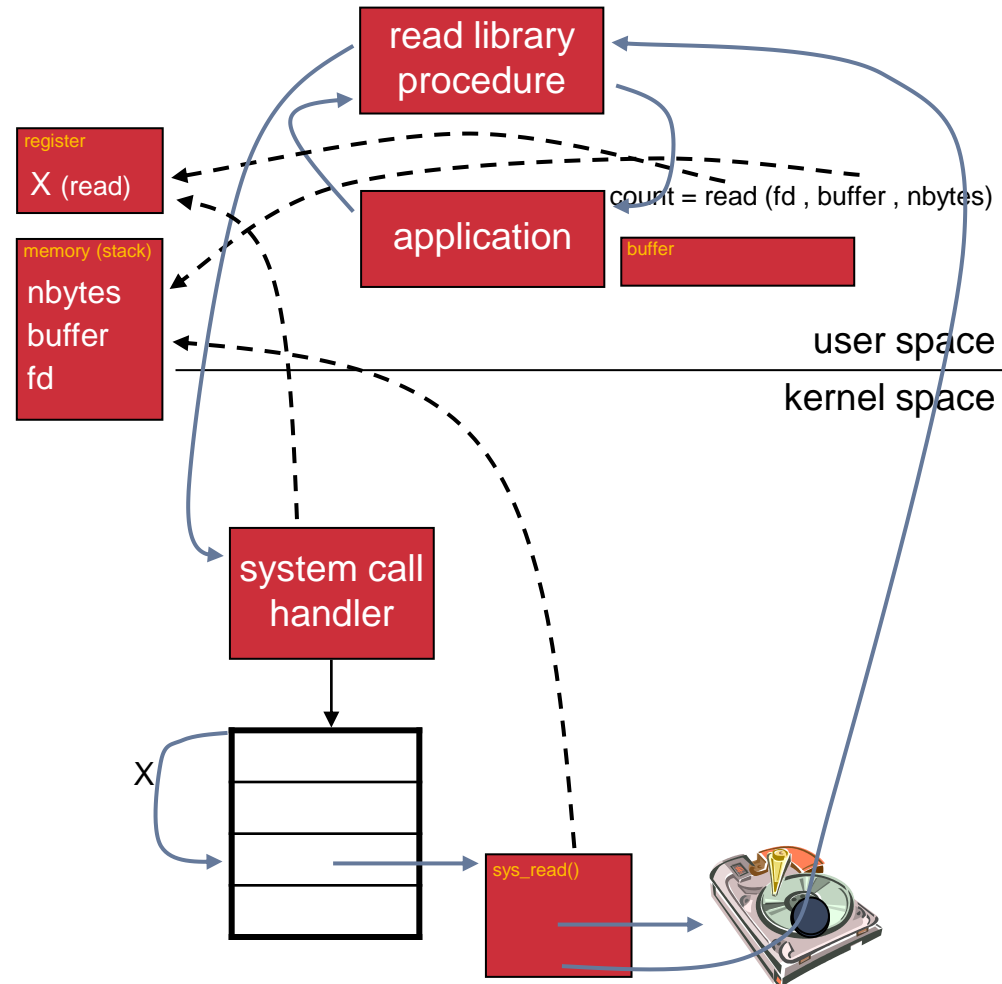


SystemKALL: read



- C example:
`count = read(fd,buffer,nbyte)`

1. push parameters on stack
2. call library code
3. put system call number in register
4. call kernel (TRAP)
 - ✓ kernel examines system call number
 - ✓ finds requested system call handler
 - ✓ execute requested operation
5. return to library and clean up
 - ✓ increase instruction pointer
 - ✓ remove parameters from stack
6. resume process



- Hvis en angriper, eller skadelig programvare, får tilgang til kjerne-modus (kernel mode):
 - Har angriperen mulighet til å utføre systemkall direkte
 - Har angriperen tilgang til minne og all hardware uten beskyttelse fra prosessoren (og operativsystemet)
 - Har angriperen FULL kontroll

Filsystemer



- Tre nivåer:

- Bruker (user)
- Gruppe (group)
- Andre (other)

- Tre mulige adgangsrettigheter for hvert nivå:

- Lese (read)
- Skrive (write)
- Utføre (execute)

- Adgangsrettigheter endres av eier ved bruk av kommandoen *chmod*

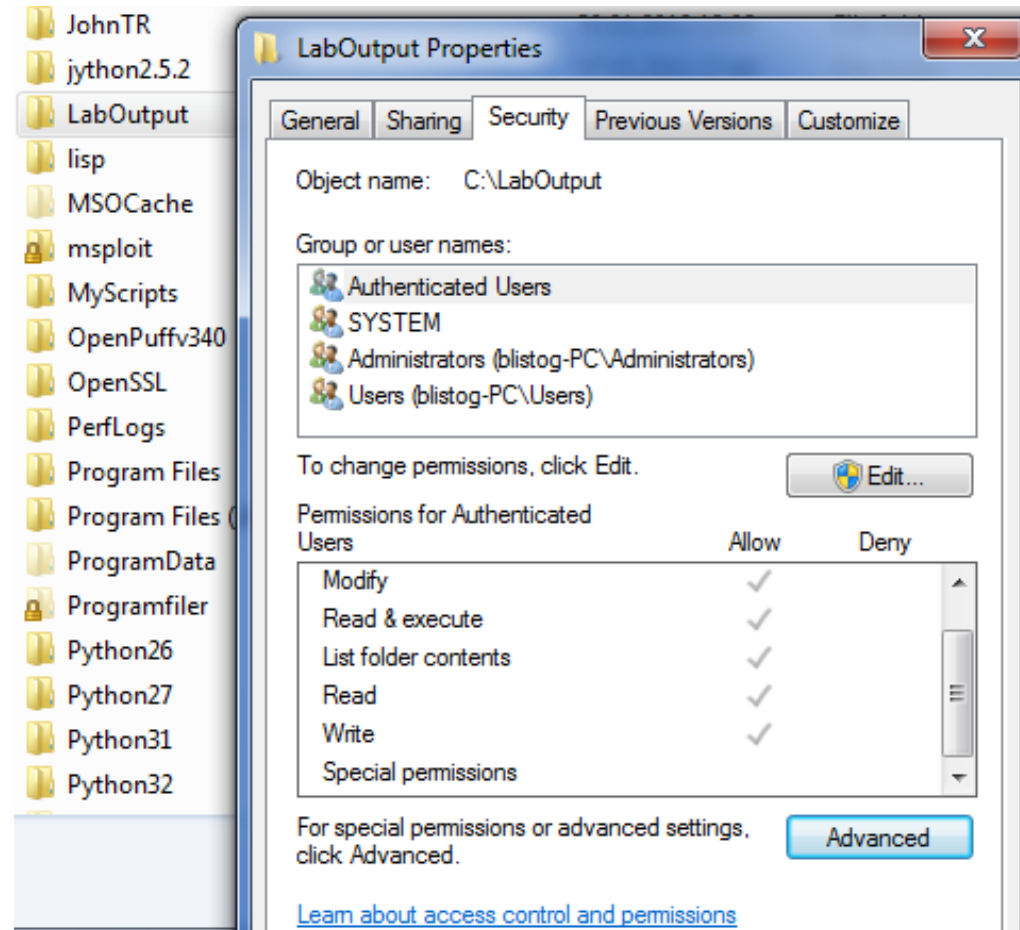
Eksempel: chmod o+r filen.min

For kataloger:

- r lese som en fil
- x åpne lokale filer
- w lage og slette filer



- Standard filsystem er **NTFS**, som understøtter en litt mer finmasket inndeling enn klassisk UNIX FS



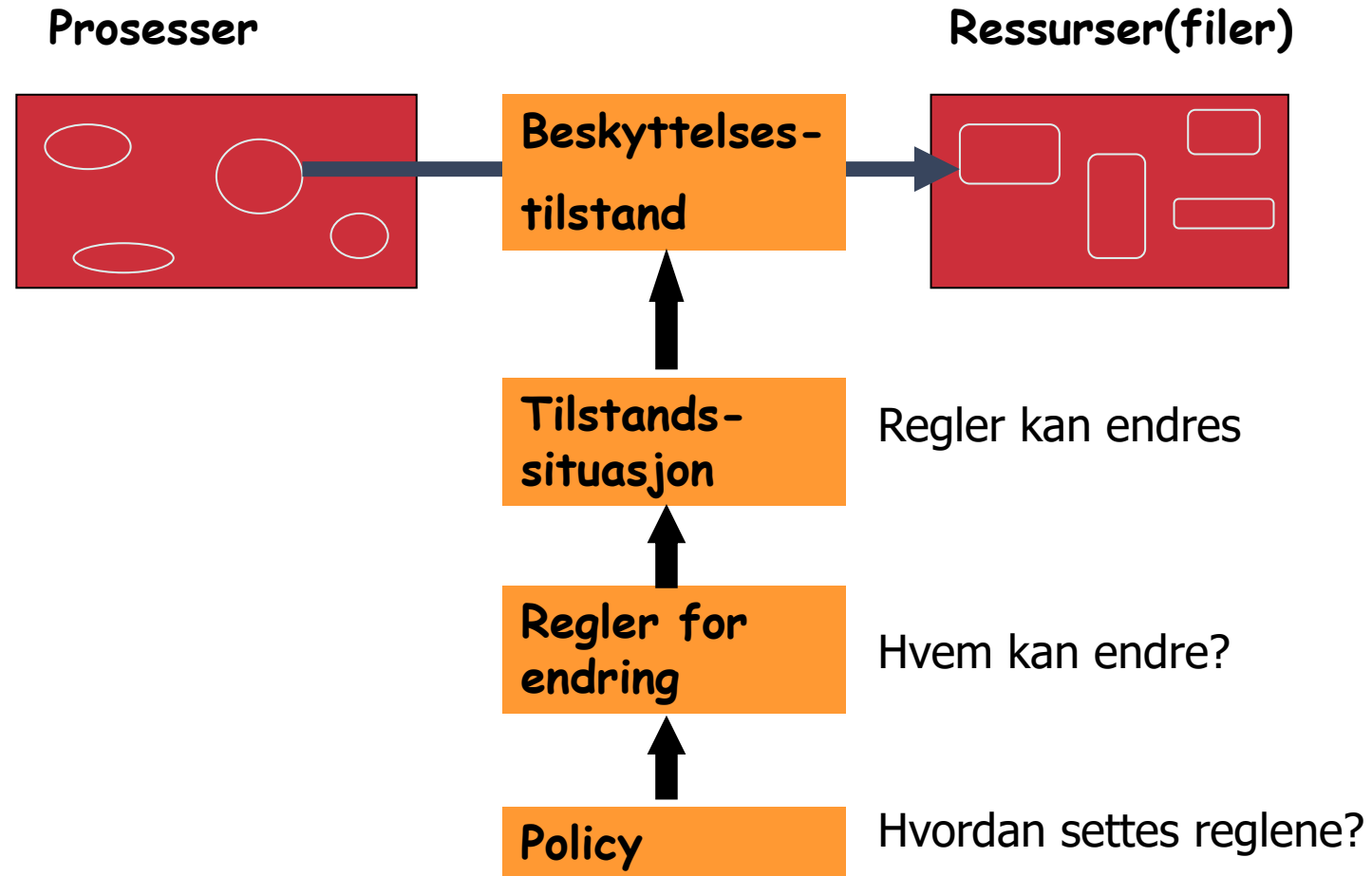
Minneadministrasjon

- Minnet er der filer og prosesser «lever»
- Å undergrave sikkerheten handler dermed i stor grad om å få tilgang til minneområder...

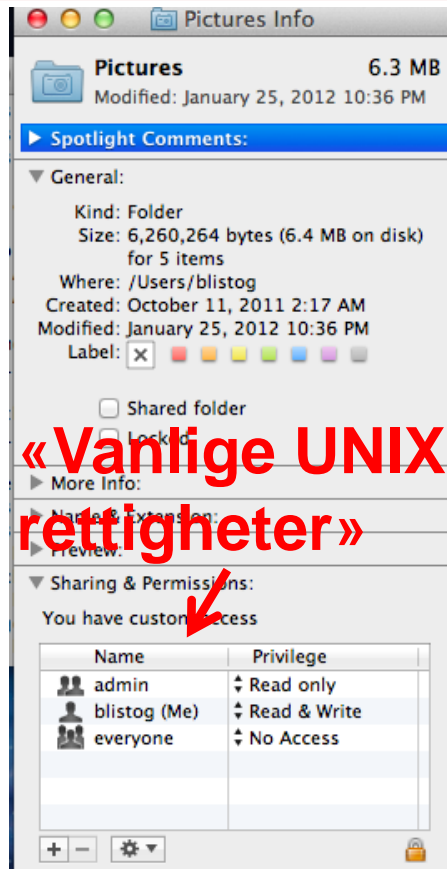
- OS og alle applikasjoners kjøretilstand befinner seg i minne eller i filsystemet
- Dermed handler det å beskytte en datamaskin om å beskytte minne og filsystem
- De viktigste teknikkene på OSX, Win7-10 og Linux er knyttet til:
 - Brukernavn og adgangstegn («tokens»)
 - Rettigheter for prosess med et adgangstegn i forhold til en fil

- Det er på alle moderne OS langt flere «brukere» enn de du selv har startet
 - Test selv feks med procexp64.exe (SysInternals verktøy fra TK1104)
- I tillegg er det alltid:
 - root/Administrator (system-adm)
 - Har absolutt alle tenkelige rettigheter, inkl å gi og ta fra rettigheter...
 - Ulike systembrukere
 - Prosesser og tjenester som er automagisk startet av systemet skal også ha begrensede rettigheter iht «Principle og least privilege»

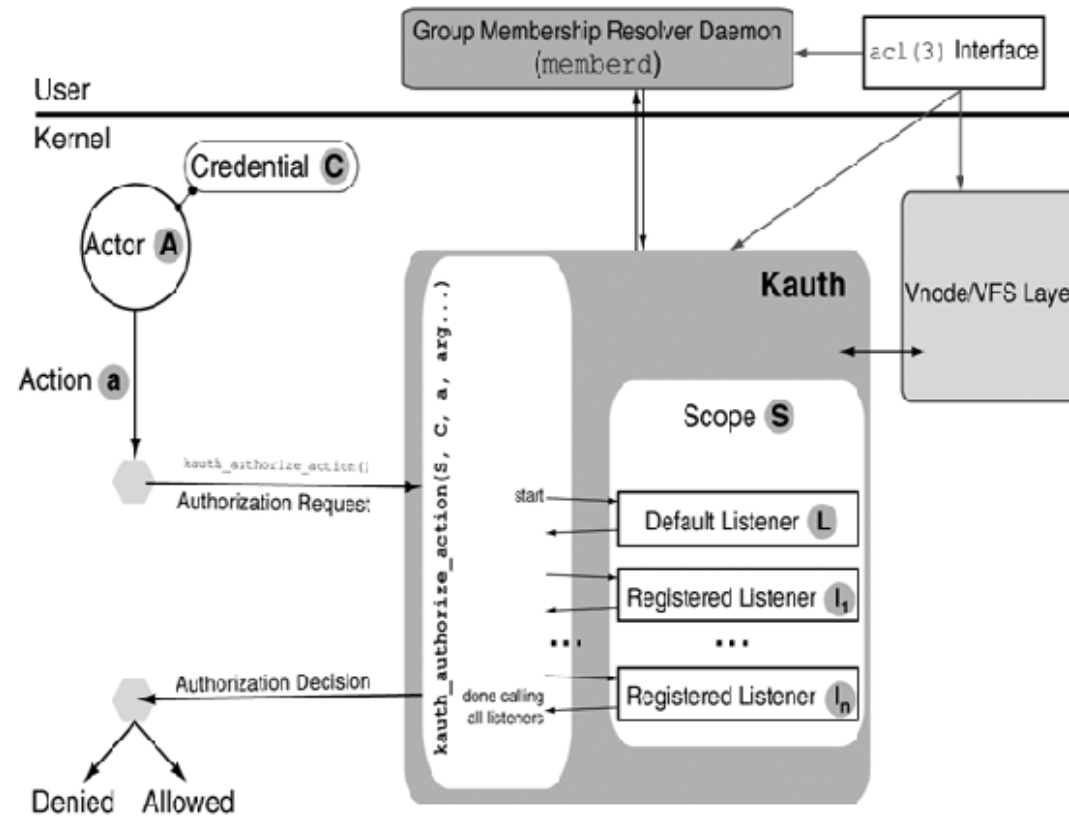
Eks: En prosess skal skrive til fil



OSX: Autorisering og ACL-bruk



«Vanlige UNIX-rettigheter»



```
McBOL:~ blistog$ chmod +a "blistog deny delete" ./aclTest.txt
McBOL:~ blistog$ ls -el aclTest.txt
-rw-r--r--+ 1 blistog  staff  0 Jan 30 22:03 aclTest.txt
  0: user:blistog deny delete,chown
McBOL:~ blistog$
```

ACL for filen

Endre ACL med (f.eks.) `chmod +a`

Liste ut ACL med (f.eks.) `ls -el`



- Local Security Authority (LSA) -
som lokal applikasjonsmodus prosess
 - Står for bruker autentisering på lokal maskin
- LSA – på domenekontroller i et LAN
 - Autentisering og autorisering i sentralt administrert LAN
- Security Reference Monitor
 - Står for beskyttelse av objekter (prosesser, ressurser, ...)

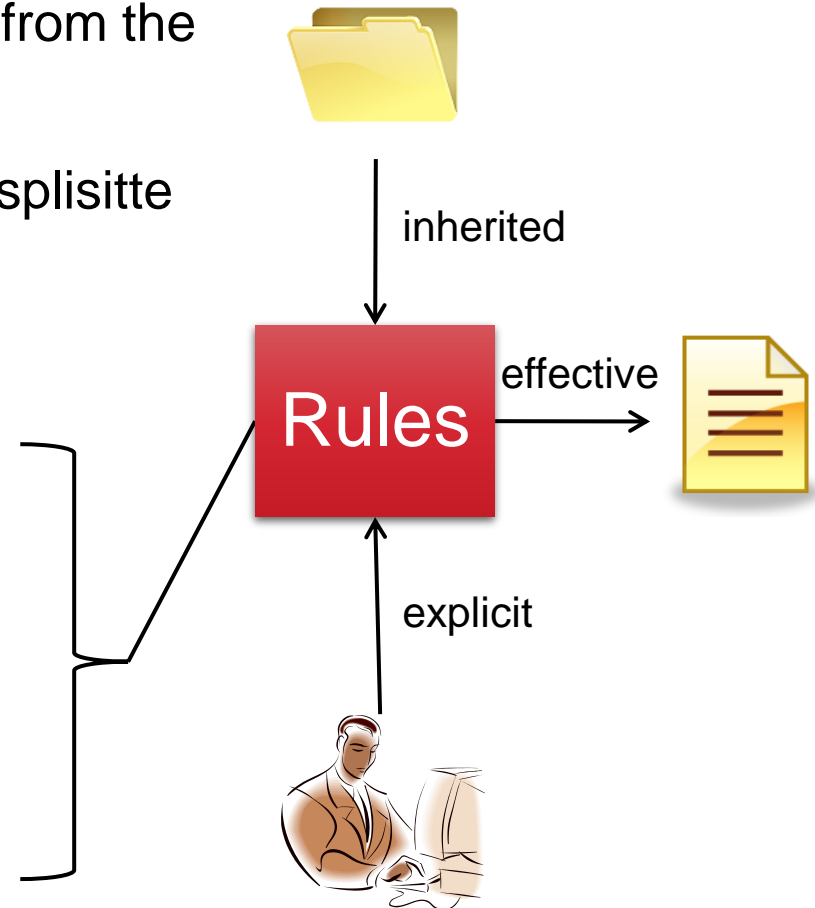
Basic NTFS Permissions

NTFS Permission	Folders	Files
Read	Open files and subfolders	Open files
List Folder Contents	List contents of folder, traverse folder to open subfolders	Not applicable
Read and Execute	Not applicable	Open files, execute programs
Write	Create subfolders and add files	Modify files
Modify	All the above + delete	All the above
Full Control	All the above + change permissions and take ownership, delete subfolders	All the above + change permissions and take ownership

- **Explicit:** settes av eieren (*owner*) for hver bruker/gruppe.
- **Arvet (Inherited):** dynamically inherited from the explicit permissions of ancestor folders.
- **Effective:** finnes ved å kombinere de eksplisitte og arvede tillatelsene

Å finne effektive filtillatelser:

- Pr default, så har user/group ingen tillatelser
- Eksplisitte tillatelser går foran arvede dersom de er motstridende.
- Denied-regler går foran Allowed dersom de er motstridende.





- Er komplisert og vanskelig å få oversikt over
- Leder til mye hodebry for administratorer
 - Som jo sjelden opplever problemet, bare brukere som sier at ting ikke virker...
- Både NT og OSX (varierer?) lagrer brukernavn/passord som kryptografiske hash og aldri som ren text.
 - OSX har benyttet saltet sha512 fra 10.7
 - Begge benytter det til å dele ut security-tokens til bruker/prosess.

Sikkerhet i operativsystemet

Med fokus på eksempler hentet fra Windows kernel

- Isolering av privilegert kode i Ring 0 (kernel mode)
- Hver usermode prosess har separate minneområder
- Nytt fra og med Vista (Windows 6.0):
 - Session 0 Isolation
 - Protected Mode
 - Address Space Layout Randomization
 - 2-way Windows Firewall
 - Innebygget anti-spyware; Windows Defender
 - BitLocker volume kryptering
 - Patch Guard (kun på 64 bit)

- Windows NT Logon Process (winlogon.exe) er ansvarlig for å logge inn brukere lokalt på maskinen
- Dette utføres gjennom det som kalles Graphical Identification and Authentication (gina.dll)
- 3. parts leverandører kan tilby alternative gina.dll implementasjoner
- Autentifiseringen utføres med passord, eller smartkort
- 3. parts leverandører tilbyr også biometriske login metoder, samt USB
- Hvis smartkort eller USB brukes kan EFS og BitLocker krypto nøkler lagres der for å tilby *single sign-on* funksjonalitet
- Når brukeren er logget inn lastes userinit.exe brukerens konto, og det blir utstedt et security token – alle prosesser som startes i brukerens kontekst arver dette tokenet

Problem:

- De fleste brukere er alltid logget inn som administrator

Microsofts løsning:

- Selv om du logger deg inn som administrator – så er du ikke det...
- Løst ved å splitte administrator kontoen i to deler
- For å få fulle administrator privilegier kreves *elevation*
- Elevation gjøres ved at brukeren må svare på et spørsmål fra mekanismen som kalles User Account Control
- UAC prompten blir vist på Secure Desktop for å forhindre at malware kan forfalske brukerens svar gjennom å sende meldinger til UAC

Svakheter:

- UAC foregår ikke ved å inspisere privilegiene som blir forsøkt brukt
- UAC ber kun brukeren om levering av prosessen, ikke av hendelsen prosessen forsøker å utføre
- Det betyr at en prosess som blir elevet kjører med fulle privilegier
- De fleste brukere har ingen forutsetninger til å forstå om levering er trygt eller ikke, og siden veldig mange prosesser trenger det blir det en vane for brukeren å trykke Allow på dette spørsmålet
- Slik har Microsoft fraskrevet seg ansvaret for alle sikkerhetsproblemer!
- Med UAC er det ikke lenger Windows sin "feil" at malware kan spre seg, men brukeren sin feil siden han har tillatt infeksjonen

Session 0 isolation (Win)

- Tidligere kjørte første innloggede bruker som Session 0
- Siden de fleste klienter kun kjører logget inn som en bruker var det derfor lett for malicious kode å hijacke servicer (som kjører som LocalSystem)
- I Vista er Session 0 isolert og første bruker blir logget inn på Session 1
- Servicer (på Session 0) kan ikke interagere med brukeren og er derfor ikke sårbare ovenfor angrep gjennom meldingssystemet
- På grunn av at LocalSystem kjører med et høyere integritetsnivå enn vanlige brukere er det veldig vanskelig å hijacke en service direkte gjennom DLL injection osv

Protected Mode (Win)

- Den mest sårbare applikasjonen på en maskin (ovenfor remote angrep) har vist seg å være Internet Explorer
- Sikkerhetsfeil i Internet Explorer selv, og i plugins, har gjort dette til et yndet angrepsmål
- På Vista er dette løst med at Internet Explorer kjører i såkalt Protected Mode (Internet Explorer er per i dag den eneste som gjør dette)
- En Protected Mode applikasjon kjører på laveste integritetsnivå hvilket betyr at den ikke kan skrive til ressurser lokalt (annen enn i en egen temp mappe), og den kan ikke åpne ressurser som andre applikasjoner har åpnet (shared minne, mutexer el)
- Dette resulterer i at hvis Internet Explorer blir angrepet så kan den ikke brukes som en plattform for å angripe maskinen videre fra

- Filtrere all innkommende trafikk (Windows XP SP2 og Vista)
- Mulighet til å også konfigurere regler for utgående trafikk (kun Vista)
- Støtter IPv6 og IPSec
- Regler kan begrense trafikk til visse porter og IPer
- Regler kan konfigureres for individuelle servicer

Svakheter:

- Har ikke håndtering av trojaner teknikker (Dll injection el)
- Regler kan legges til gjennom et COM interface – av alle...
- Regler ligger i plain text i registry

- Basert på GIANT AntiSpyware (men mye funksjonalitet ble fjernet av Microsoft, mest sannsynlig av frykt for at brukere skulle gjøre noe feil)
- Støtte for Windows XP, 2003 og Vista (Beta 1 og 2 også på 2000)
- Preinstallert fra og med Windows Vista
- Ikke en fullgod antivirus, brukere må ha dedikert antivirus i tillegg

- Encrypted File System
 - *Se forelesning om kryptering forrige uke*
- BitLocker Disk Encryption
 - Full volume kryptering
 - Bruker AES algoritmen (samme som EFS)
 - Krever boot passord for adgang til maskinen
 - Støtter også USB device som inneholder nøkkelen

- FileVault
 - Full volume kryptering
 - Bruker AES algoritmen
 - Krever boot passord for adgang til maskinen
 - Støttes av «Find My Mac», kan fjernslette maskinen
- Disk Utility
 - Kan kryptere USB disker
 - (BitLocker støttes også på OSX for dette formålet)

- Word, PDF, osv
 - Mange tekstbehandlere kan kryptere filer med passord
- ZIP, osv
 - Det tryggeste vil være å legge filer i et arkivformat som ZIP, og kryptere denne
 - Filnavnene inne i arkivet vil være synlig, men det kreves passord for å pakke ut filene – evt pakk arkivet ned i et nytt arkiv

Patch Guard (Win)

- På x86 versjoner av Windows er det mulig, fra en driver, å endre strukturer i Windows kernel direkte – dette kalles *kernel patching*
- Dette brukes primært av to grupper software
 - Rootkits; filtrerer kernel kall for å skjule seg selv
 - AntiVirus/Spyware; for å tilby økt sikkerhet
- På 64 bits Windows beskytter Patch Guard kritiske deler av operativ systemets kernel (SDT, IDT, GDT, kode i HAL og ntoskrnl) ved å kalkulere checksum av minnet – og krasje systemet hvis checksum'en ikke lenger stemmer
- Implementasjonen av Patch Guard endres periodisk for å forhindre at noen reverse engineerer koden og knekker mekanismene
- Dette bryr ikke nødvendigvis malware seg om, men sikkerhetssoftware kan ikke lenger benytte seg av kernel patching i sine produkter...

Angrepsvektorer – malicious code

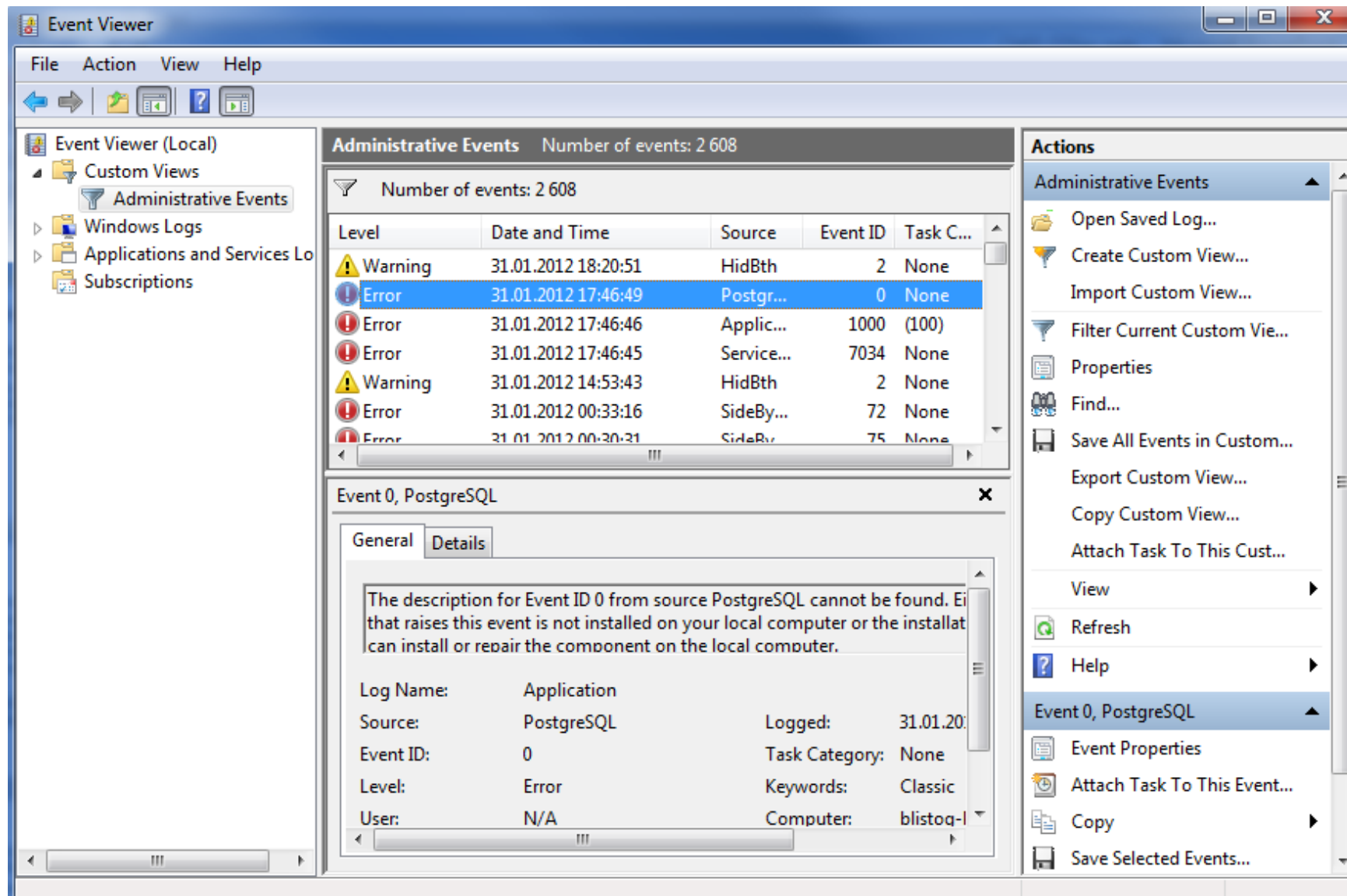
- Det finnes ingen KJENTE hack for å ta over en maskin lokalt uten bruker interaksjon (derfor vi kaller slike angrep «0-day»)
- At en bruker på Windows eksekverer malicious kode er ikke tilstrekkelig til å infisere maskinen, brukeren må i tillegg akseptere en UAC popup
- Enkleste måte å knekke sikkerheten i Windows eller OSX er å angripe det svakeste leddet; brukeren.
- Å lure brukeren til å utføre en operasjon han ikke burde gjøre kalles for *social engineering*, og er veldig vanlig

- Utnytte svakheter i (operativ system) servicer som lytter på nettverket
- Utnytte svakheter i 3. parts programvare som lytter på nettverket (for eksempel fildelingsprogramvare)
- Lure brukeren, gjennom social engineering, til å eksekvere kode han mottar på epost
- Internet Explorer har begrensede rettigheter da den kjører i Protected Mode; vanskeliggjør infeksjon (Chrome, Firefox bruker ikke dette)

- **PATCH!!!!**
- Alle OS svakheter som blir oppdaget og reparert blir så kjente for angripere.
 - Dersom du ikke har oppgradert/patchet er du da særlig sårbar!
 - Når Microsoft har «patch Tuesday» omtaler vi i bransjen neste dag som «exploit Wednesday»

- Jf. Prinsipp 10 (s 17): Compromise recording
- Å holde oversikt over hvilke prosesser som har blitt startet opp av hvem er også en del av sikkerhetsarbeidet
- Logger kan gi hint om mistenkelig adferd fra brukere, prosesser og drivere
- Kan gi hint om hvor(dan) sikkerheten har blitt kompromitert

NT Logging: eventvwr.msc



The screenshot shows the Windows Event Viewer application. The left pane displays the tree view with 'Administrative Events' selected. The main pane shows a list of events with the following columns: Level, Date and Time, Source, Event ID, and Task Category. The selected event is an Error from PostgreSQL at 17:46:49 on 31.01.2012.

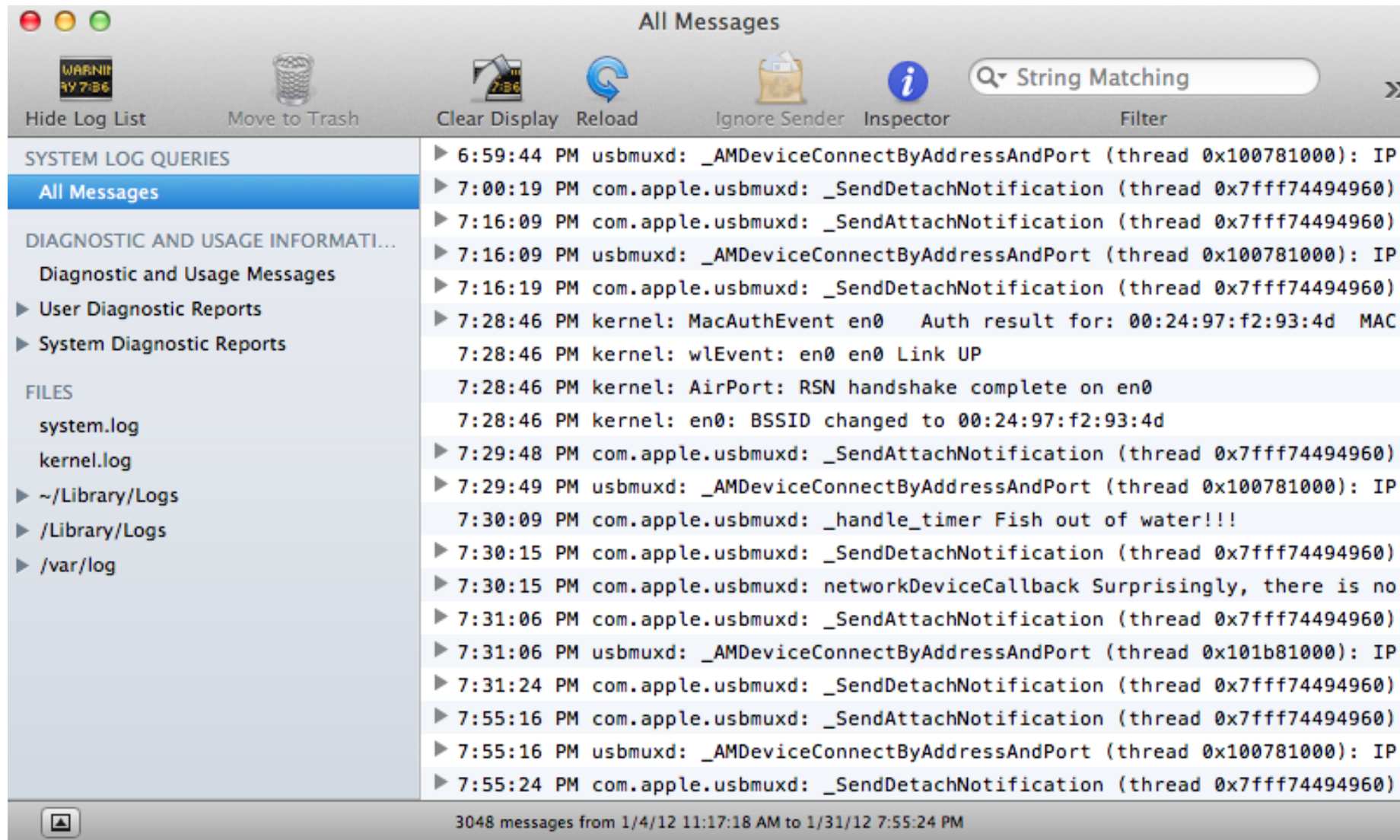
Level	Date and Time	Source	Event ID	Task Category
Warning	31.01.2012 18:20:51	HidBth	2	None
Error	31.01.2012 17:46:49	Postgr...	0	None
Error	31.01.2012 17:46:46	Applic...	1000	(100)
Error	31.01.2012 17:46:45	Service...	7034	None
Warning	31.01.2012 14:53:43	HidBth	2	None
Error	31.01.2012 00:33:16	SideBy...	72	None
Error	31.01.2012 00:30:31	SideBy...	75	None

The details pane for the selected event shows the following information:

The description for Event ID 0 from source PostgreSQL cannot be found. Either the component that raises this event is not installed on your local computer or the installation can install or repair the component on the local computer.

Log Name: Application
Source: PostgreSQL
Event ID: 0
Level: Error
User: N/A
Logged: 31.01.2012
Task Category: None
Keywords: Classic
Computer: blisteq-l

OSX: Utilities/Console

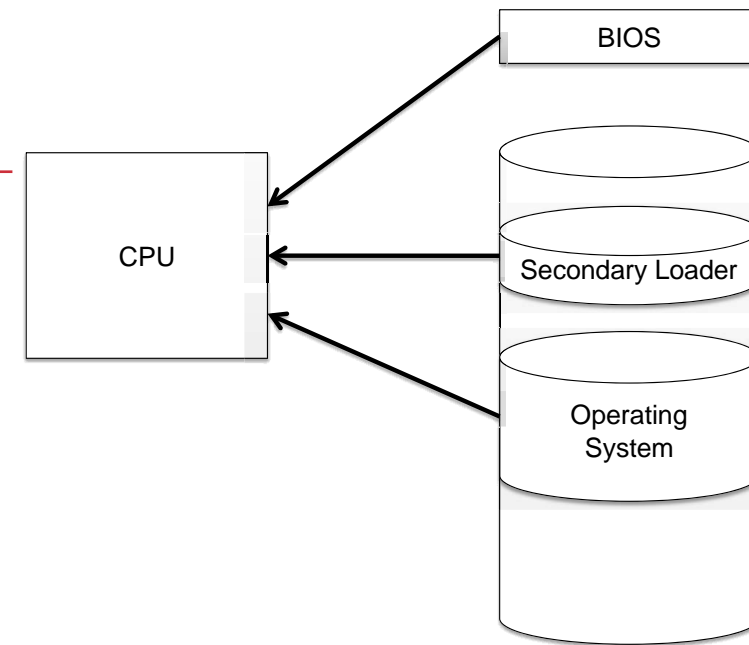


Noen andre trusler

- Hvordan kan bruker-programmer få rot-privilegier?

Boot sekvensen

- Lastingen av OS går via BIOS/UEFI
- som så laster en sekundære bootloader
- Som så laster OS og overlater kontrollen til OS påloggingsystem
- «Eve» kan få full kontroll på flere stadier her



- Passordbeskytt BIOS/EFI (Moderne BIOS/EFI tilbyr begge dette)?
- Passordbeskytt og krypter alt på maskin og i filsystemet (OSX, Win tilbyr)?

Dvale (hibernation)



1. Bruker lukker laptopen som går i dvale og lagrer fullstendig minnetilstand i en fil (hiberfil.sys på Win7)

2. Angriper kopierer hiberfil.sys og kan potensielt finne alle ukrypterte passord som befant seg i RAM da dvale inntraff



Bakdører (Thompson)

```
while (TRUE) {  
    printf("login: ");  
    get_string(name);  
    disable_echoing();  
    printf("password: ");  
    get_string(password);  
    enable_echoing();  
    v = check_validity(name, password);  
    if (v) break;  
}  
execute_shell(name);
```

(a)

a) Normalt program

```
while (TRUE) {  
    printf("login: ");  
    get_string(name);  
    disable_echoing();  
    printf("password: ");  
    get_string(password);  
    enable_echoing();  
    v = check_validity(name, password);  
    if (v || strcmp(name, "zzzzz") == 0) break;  
}  
execute_shell(name);
```

(b)

b) Kode med bakdør (se siste linje i while-løkken)

En av designerne av UNIX gjorde noe tilsvarende i den første C-kompilatoren!!!

Buffer Overflow

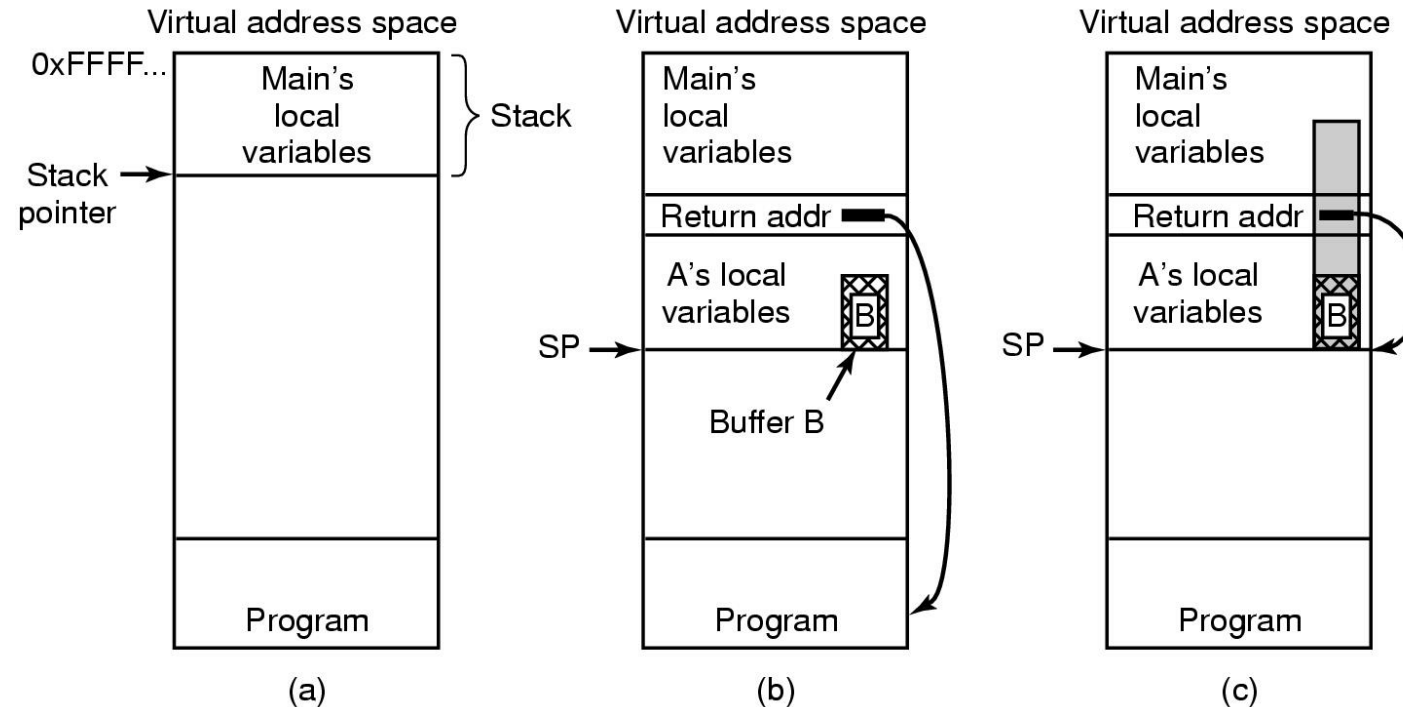
```
int i;  
char x[1024];  
i = 12000;  
c[i] = 0;
```

vil overskrive
hva som
måtte befinne
seg i minne
10976 plasser
over starten
av arrayen
c[].

- a) Hovedprogrammet kjører
- b) Kaller program/funksjon A
- c) Buffer Overflow (grått) lar A erstatte adressen i minnet hovedprogrammet skal kjøre fra etter retur fra A

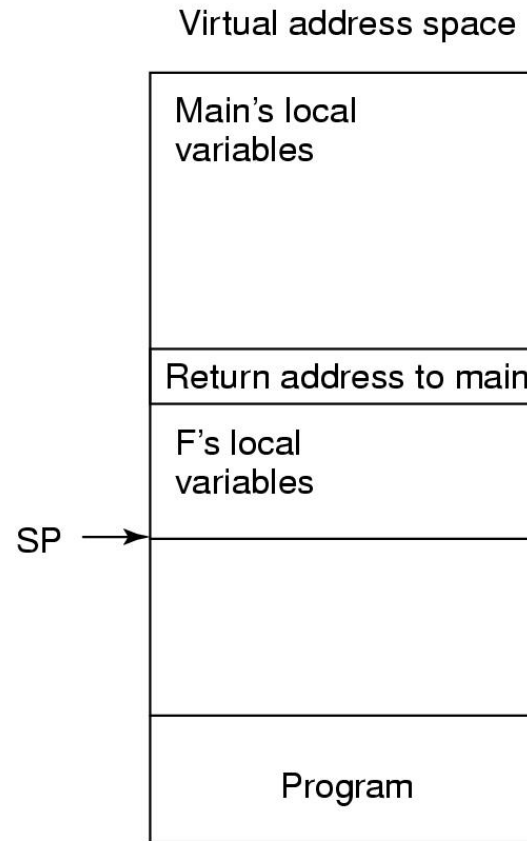
Årsaken er så enkel som at C-språk ikke sjekker grensene for arrays når de deklarereres/kompileres

OG: De leser stort sett User-input inn i arrays...

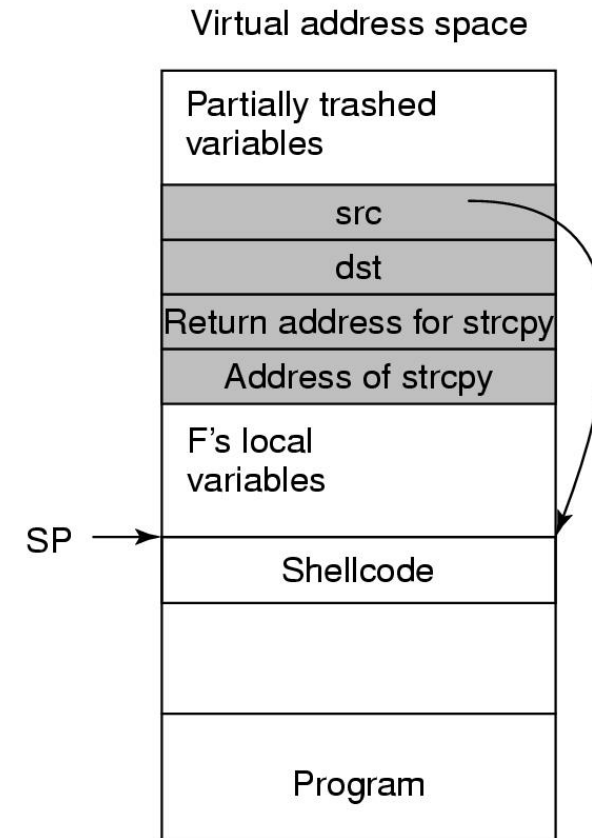


Kode injeksjon

- Typisk teknikk i forbindelse med rootkits



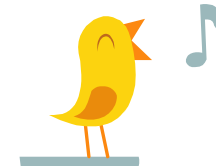
(a)



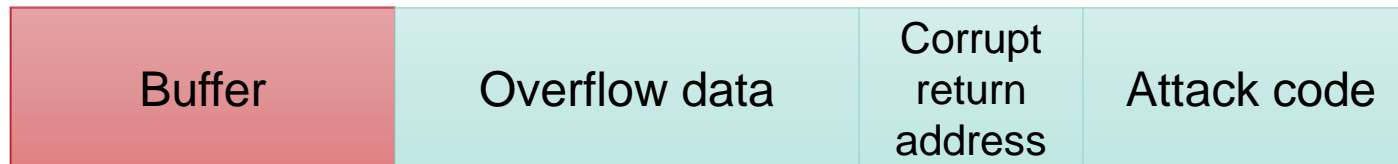
(b)

Mottiltak: Kanarifugl m.m.

Normal (safe) stack configuration:



Buffer overflow attack attempt:



- Kanari-verdiene legges inn i stacken før returadressen slik at alle forsøk på å skrive den over vil bli oppdaget av OS-monitor
- Andre teknikker
 - **Randomisering av hvilke pages som lastes hvor i Virtuetl Minne mellom hver gang (brukes nå både av Win og OSX)**
 - Kryptering

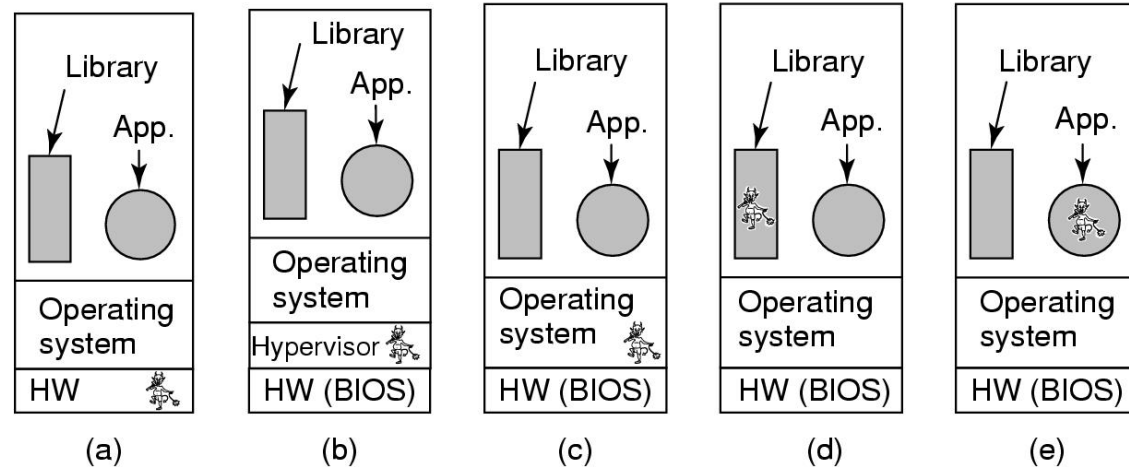
- **Address space layout randomization**
 - Støttes av alle moderne OS
 - Beskytter mot buffer overflow angrep
- Plasserer alle minne-blokker tilfeldig rundt i RAM
 - Eksekverbart område, stack, heap, biblioteker...
 - Umulig å vite hvilke adresser man må «overta» for å få tilgang til kjørbare kode og «root»

Rootkits

1. Firmware
2. Hypervisor
3. Kernel
4. Bibliotek
5. Applikasjon

Teknikker brukt av rootkits:

- Skjuling av prosesser/drivere
- Skjuling av registry oppføringer
- Skjuling av fysiske filer på disk

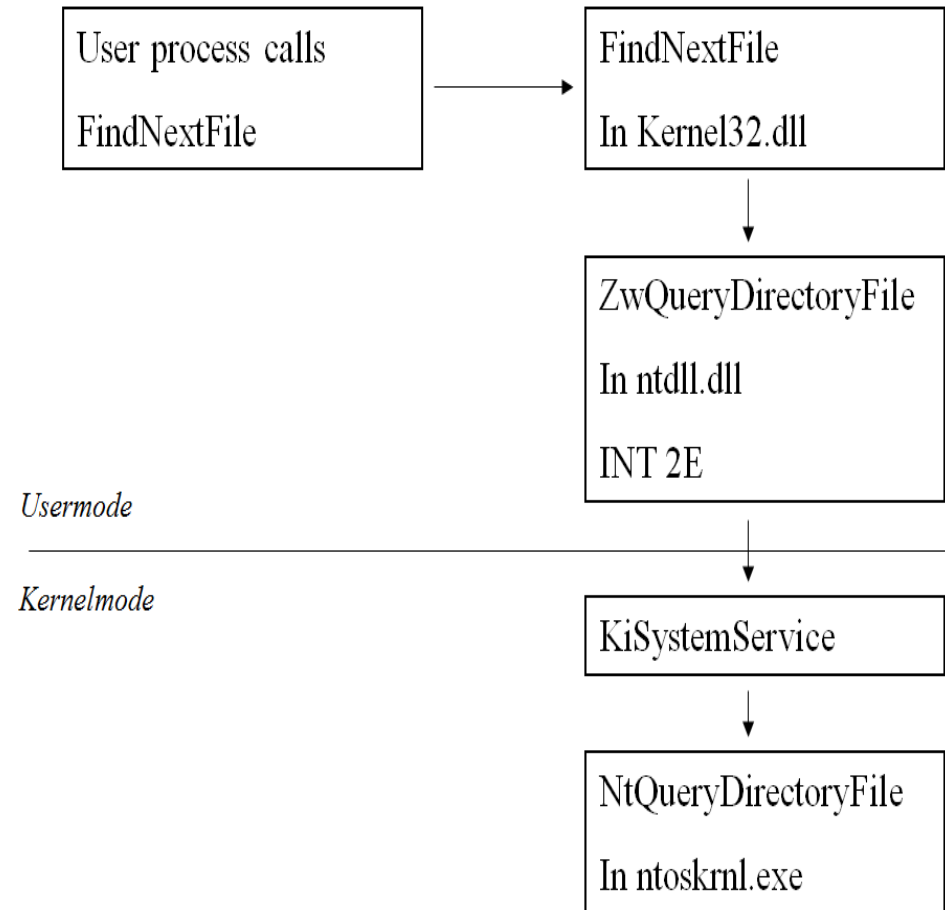


"Tidligere: Verktøy brukt av en hacker for å skaffe seg root access på en UNIX server. Inneholdt også ofte mulighet til å slette logger for å skjule spor etter hackingen."

"I dag: Egentlig ikke en egen type malware, men en teknikk brukt av malware for å skjule sine spor og gjøre seg selv usynlig på maskinen som er infisert."

NT (Win32): Rootkit-eksempel

- Ved å hooke NtQueryDirectoryFile kan man velge å fjerne oppføringer av enkelte filer (som man ønsker å skjule).
- Dette vil da medføre at FindNextFile ikke viser filen du ønsker å skjule.
- Anti-Virus applikasjoner som skanner filer ved å enumerere ut filer på harddisken vil da ikke finne filen du har skjult.
- På denne måten har "rootkittet" klart å skjule seg selv på systemet.



AVSLUTNING

Hva er operativsystemets rolle i informasjonssikkerhet?

- Siden OS (kjernen/rot-bruker) har alle rettigheter til alt så:
- MÅ noen ting gjøres for at OS ikke skal være lett å kompromitere
 - Patche automatisk og jevnlig!!!
 - Ingen brukere med høyere tilgangsnivå enn absolutt nødvendig
 - Lett å gå i surr med alle de mekanismene som er tilgjengelige
 - Windows UAC f.o.m. Vista...
 - Konservativ installasjonspolicy generelt
 - Kun signerte drivere?
 - Kun leverandører du stoler 100% på?
- SANDBOXING av alle applikasjoner er en mulighet, men lite brukt i praksis

- Kryptere arkiv filer
- Brukeradministrasjon og filrettigheter
- Innebygget brannmuren (vi skal erstatte den neste uke med en bedre)
- *Anti-virus beskyttelse (øvingsoppgave neste uke)*

TK2100_F02_øvingsoppgaver.pdf er litt repetisjon fra TK1104, og litt fra dagens forelesning om sikkerhet.

KRYPTERTE ARKIV FILER (Windows)

- På Windows anbefaler jeg en av følgende: WinZip, WinRar eller 7Zip

Oppgave 1: Installer en av de overnevnte, Windows har innebygget støtte for zip filer – men støtter ikke krypterte filer...

Oppgave 2: Opprett et arkiv med en eller flere filer, velg at filen skal krypteres med AES kryptering, og sett et passord

Oppgave 3: Pakk ut filene og se at du får ÅPNET arkivet, men ikke sett innholdet i filene – uten å oppgi passordet for å dekryptere. Bytt gjerne filer med en medstudent ved å sende zip filen på epost.

KRYPTERTE ARKIV FILER (OSX)

- 7zip støtter OSX, et alternativ er Archiver (<https://archiverapp.com/>), eller ZIP

Oppgave 1: Installer en av de overnevnte

<https://formulae.brew.sh/formula/zip>

Oppgave 2: Opprett et arkiv med en eller flere filer, velg at filen skal krypteres med AES kryptering, og sett et passord

<https://setapp.com/how-to/password-protect-zip>

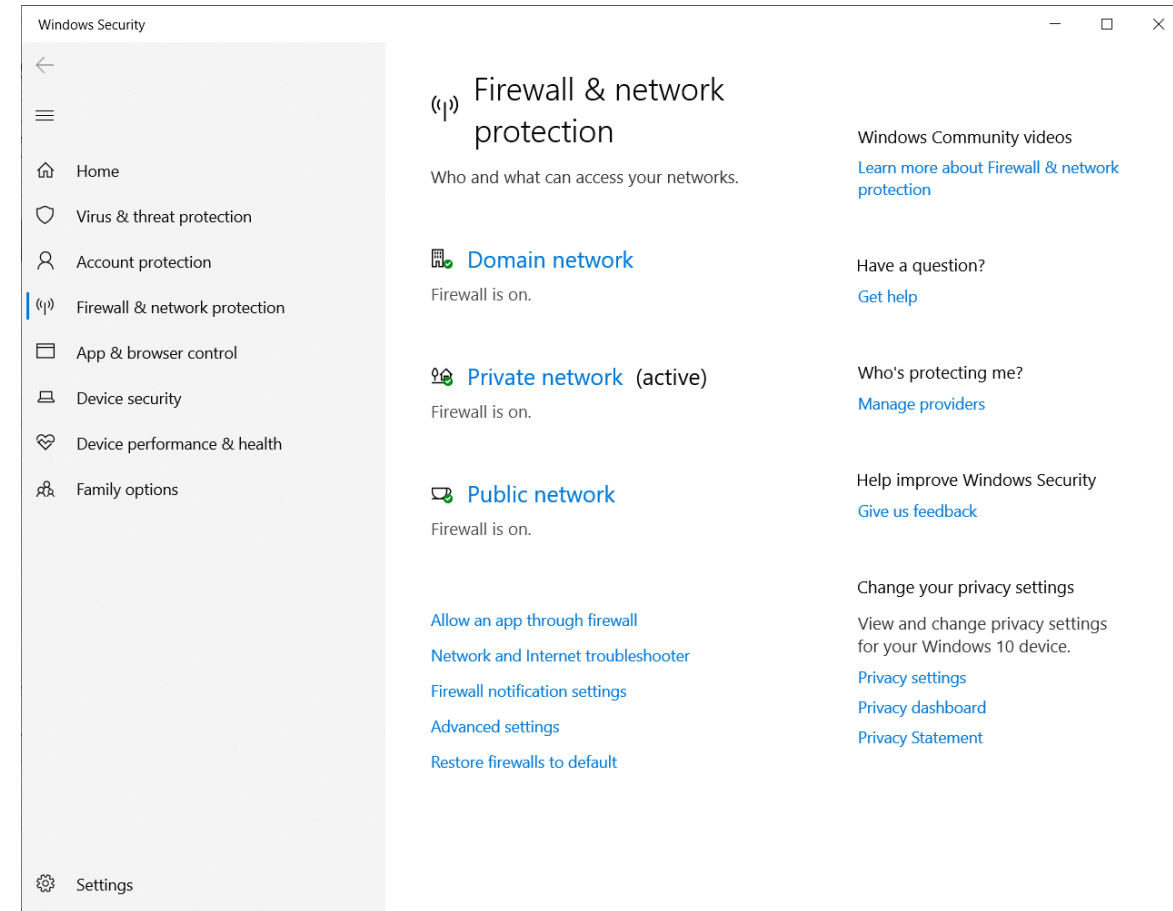
Oppgave 3: Pakk ut filene og se at du får ÅPNET arkivet, men ikke sett innholdet i filene – uten å oppgi passordet for å dekryptere. Bytt gjerne filer med en medstudent ved å sende zip filen på epost.

- Opprett en ny bruker som «vanlig bruker» (ikke admin rettigheter)
- Opprett en ny fil, gi kun leserettigheter til den nye brukeren
- Logg inn som ny bruker, prøv å endre filen
- Forsøk å gi deg selv rettigheter til filen (krever admin passordet ditt)
- Eksperimenter med «take ownership» for å få full kontroll på filen (ikke gjør dette med noe annet enn denne/disse testfilene, take ownership på en brukermappe for en annen bruker kan ødelegge brukeren 😊)

- Opprett en ny bruker som «vanlig bruker» (ikke admin rettigheter)
 - <https://www.techsolutions.support.com/how-to/how-to-add-a-new-user-account-on-a-mac-12488>
- Opprett en ny fil, gi kun leserettigheter til den nye brukeren
- Logg inn som ny bruker, prøv å endre filen
- Forsøk å gi deg selv rettigheter til filen
 - <https://www.macinstruct.com/tutorials/how-to-set-file-permissions-on-a-mac/>

INNEBYGGET BRANNMUR (Windows)

- Eksperimenter litt med den innebygde brannmuren i Windows
- *Denne brannmuren er «dårlig», vi skal bytte den ut med Bitdefender Total Security neste uke...*
- Eksempel: Installer en ny nettleser, feks Mozilla Firefox (hvis du ikke allerede bruker den), endre rettigheter i brannmuren for denne
- Bedre eksempel: Hvis dere brukte NC.exe / NCAT.exe i øvingstimene i høst kan dere også endre rettigheter for denne da den kan kjøre som «server»

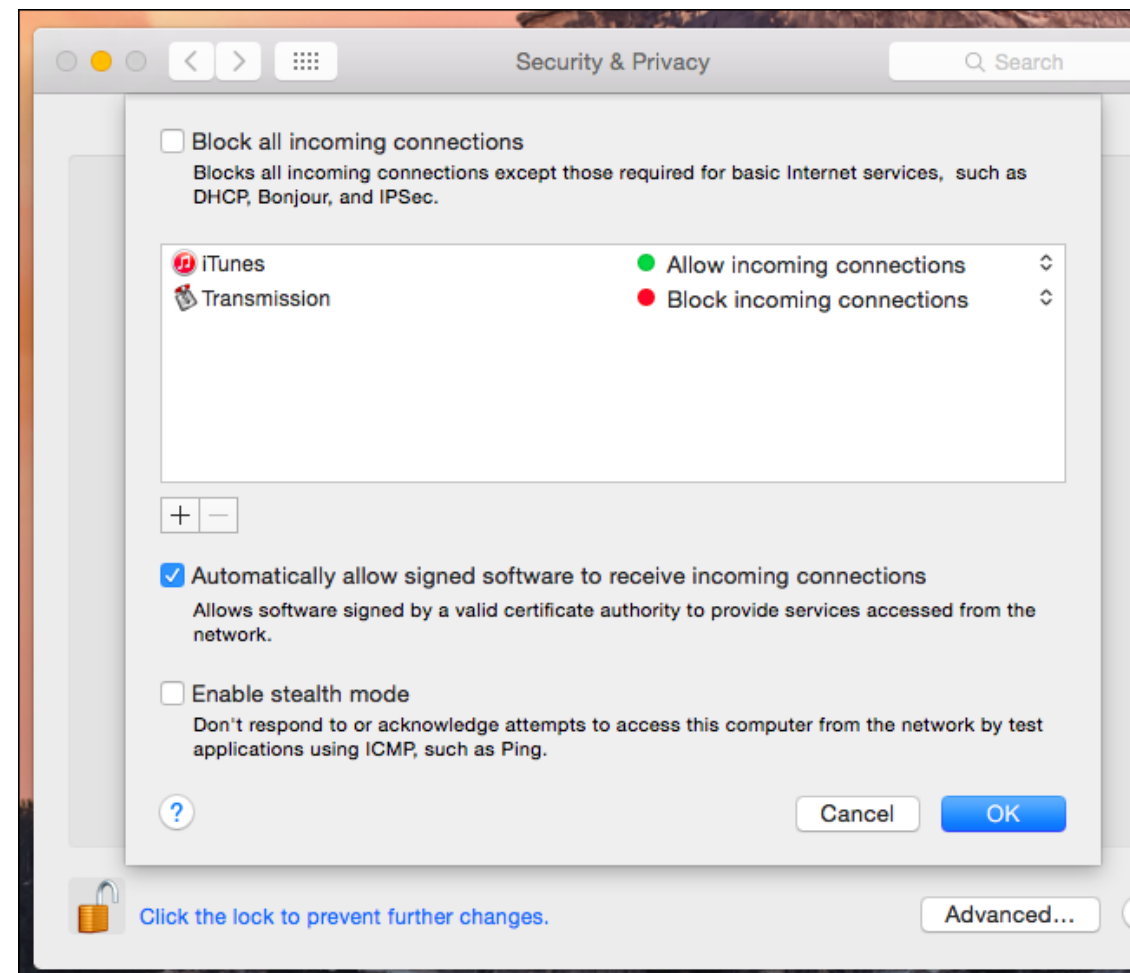


INNEBYGGET BRANNMUR (OSX)

- Eksperimenter litt med den innebygde brannmuren i Windows

<https://support.apple.com/en-us/HT201642>

- Eksempel: Installer en ny nettleser, feks Mozilla Firefox (hvis du ikke allerede bruker den), endre rettigheter i brannmuren for denne
- Bedre eksempel: Hvis dere brukte NC.exe / NCAT.exe i øvingstimene i høst kan dere også endre rettigheter for denne da den kan kjøre som «server»



Neste gang

MALWARE