BSD-sikkerhedsnoter v.0.1

Lars Sommer < *lasg@lasg.dk* >

11. april 2006

Indhold

0.1 Forord og introduktion 4

0.1.1 Forord 4

0.1.2 Introduktion 4

0.1.3 Tak til 4

1 Sikkerhed med Free- og OpenBSD 5

1.1 Generelt 5

1.1.1 Hvad er sikkerhed? 5

1.1.2 Hvilke risici er der? 5

1.1.3 Sikkerhedsproces 6

1.1.4 Sikkerhedsprincipper 6

1.1.5 Se disse RFC'er 6

2 Byggesten til sikkerhed 7

2.1 Filsystemet 7

2.1.1 UFS flag 7

2.1.2 POSIX Access Control Lists 7

2.2 Kernen 8

2.2.1 sysctl 8

2.2.2 kern.securelevel 8

2.2.3 Random pids 8

2.2.4 Core dumps 8

2.2.5 Netværkstweaking 9
2.3 Kontrol af brugerprocesser 9
2.3.1 Systemkaldet chroot 9
2.3.2 jail (kun på FreeBSD) 10
2.4 Medfødt beskyttelse 10
2.4.1 Bufferoverflow-beskyttelse 11
2.4.2 Kryptografi 11
2.4.3 Code Audit 11
2.5 Optimeringer 12
2.5.1 maxusers 12
2.5.2 maxfiles og somaxconn 12
2.5.3 Netværksbuffere 12

3 Installation **13**
3.1 Generelt 13
3.1.1 Overvej hvad du bygger 13
3.1.2 Medier og netværk 13
3.1.3 Præeksisterende sårbarheder 14
3.1.4 Opdeling af filsystemet 14
3.1.5 X 15
3.2 Opdater systemet 15
3.3 Første upgrade 15
3.4 OpenBSD installation 16
3.4.1 Opdater systemet 16
3.5 Hærdning efter installation og opdatering 16
3.5.1 Konfigurer brugere og grupper 16
3.5.2 Indstil mount options 17
3.5.3 Indstil sshd 17
3.5.4 Opsæt logning 17

3.5.5 Lav login bannere 17
3.5.6 Opsæt NTP 18

4 Administration **19**

4.1 Adgangskontrol 19
4.2 Hverdagsting 20
4.2.1 Datagendannelse 20
4.3 Opgradering 20
4.4 Sårbarhedsrespons 21
4.5 Netværksservices 22
4.5.1 inetd 22
4.5.2 Ting der skal undgåes 22
4.5.3 NTP 22
4.6 Hold øje med systemet 23

0.1 Forord og introduktion

0.1.1 Forord

BSD er godt. BSD kan gøres sikkert. Jeg interesserer mig for operativsystemer og sikkerhed. Her er en lille kombination.

0.1.2 Introduktion

Dette er ikke en bog! Dette er hverken en veldokumenteret vejledning eller opslagsværk! Dette er mine egne personlige noter til et par artikler og bøger. Det kan på ingen måde garanteres at du kan få noget somhelst brugbart ud af dette.

Hvis du kan bruge det til noget, skrive det bedre, eller blot har spørgsmål eller kommentarer, er du velkommen til at skrive til mig på lasg@lasg.dk

0.1.3 Tak til

- *Mastering FreeBSD and OpenBSD Security*, Yanek Korff, Paco Hope, Bruce Potter, O'Reilly 2005
- *OpenBSD's FAQ* på http://www.openbsd.org/faq/index.html
- *FreeBSD's handbook* på http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/
- VIM, LaTeX, vilkårlige folk der har forsøgt at omgå min sikkerhed, folk på dkIRC

Kapitel 1

Sikkerhed med Free- og OpenBSD

1.1 Generelt

1.1.1 Hvad er sikkerhed?

- Konfidentialitet: At være bevidst om niveauer af fortrolighed. At sikre fortrolige data bedre end ikkefortrolige data.
- Integritet: At sikre at ens data er som den skal være. At sikre ens data indeholder det den skal. At bruge signaturer og hash-tjeks f.eks.
- Tilgængelighed: Ens sikkerhedselementer og ens data skal være tilgængeligt de rigtige steder. Duer ikke hvis en bruger har brugt al diskpladsen eller båndbredden.

1.1.2 Hvilke risici er der?

- Angreb: Personer eller robotter laver tilfældige eller målrettede angreb.
- Programmelfejl: Programmer kan indeholde fejl, der gør systemet usikkert.
- Forkert brug: Dumme brugere eller uerfarne admins kan bruge korrekt programmel forkert.

1.1.3 Sikkerhedsproces

- Initierende installation: Her sikres sikkerheden godt i første omgang
- Løbende vedligeholdelse: Så snart et system er klar til brug, begynder det at ændre sig. Løbende vedligeholdelse kræver løbende sikkerhedstjek.
- Uheld og efterforskning: Er skaden sket, skal der gøres ekstra meget for at fejl bliver fundet og rettet, og det sikres at de ikke kommer igen.

1.1.4 Sikkerhedsprincipper

Forskellige måder at håndtere sikkerhed på:

- Tilføje sikkerhed hen ad vejen: Finde det svageste punkt, og sikre dette, lidt ad gangen, hen ad vejen.
- Sikkerhedslag: Tilføje flere lag af sikkerhed, sådan at man både skal igennem firewall, ids, login osv.
- Fejlsikret: Hvis der sker en fejl eller et uheld, så sørg for at det ikke kan gøre stor skade, fx med jails.
- Minimer privilegierne: Enhver bruger og service, skal have så absolut få rettigheder som muligt.
- Fordel servicene: Hvis een ting fejler, skal resten kunne fungere.
- Simplificer: Jo mere avanceret, jo sværere at fejlfinde på, og jo sværere at opdatere og sikre.
- Dokumenter security through obscurity grundigt
- Mistænk som standard: Hvis noget tyder den mindste smule på at noget er usikkert, så regn med det er det, og tjek op på det.
- Opdater og følg med: Hold systemet opdateret, og følg med på maillister osv.

1.1.5 Se disse RFC'er

2196, 2504, 2828, 3013, 3365, 3631

Kapitel 2

Byggesten til sikkerhed

2.1 Filsystemet

2.1.1 UFS flag

Husk nogle flag kan ikke unsettes i visse securelevels.

Kommandoen chflags bruges til at sætte og afsætte flag. Man-siden for chflags er kort og præcis, med lister over flag osv.

Kandidater for schg-flaget: alle binaries og libraries (søg på foldere med navne som bin, sbin, lib, libexec, libdata), /etc/ssh, admins .ssh/authorized_keys.

Append flag er usmart til logfilerne, da det ikke gir mulighed for rotation.

At søge efter filer med flags kan man med find. Den skal have parametren -flags. F.eks: `find / -flags +uunlnk`

2.1.2 POSIX Access Control Lists

Både ACL og UFS2 er kun til FreeBSD endnu. Det er en slags detaljeret udvidelse af normale unix permissions.

ACL slåes til på en partition via fstab eller ved at angive det i superblokken på den. I fstab angives det som en option, acls, ala:

```
/dev/ad0s2g /usr ufs rw,userquota,groupquota,acls 2 2
```

For at angive det i superblokken, afmonteres partitionen, og der køres

en `tunefs -a /usr`

For at sætte og læse ACL'er, bruges setfacl og getfacl. Deres man-sider indeholder gode forklaringer, samt eksempler på begge dele.

ACL på en fil, kan ikke ses med ls -l, men hvis der er sat ACL, vises et + efter de normale unix permissions.

ADVARSEL: Hverken UFS flag eller ACL er understøttet i NFS. Så hvis NFS bruges, kan der nemt opstå problemer hvis man prøver at håndtere flag eller ACL via dette.

2.2 Kernen

2.2.1 sysctl

En kørende kerne kan tunes med sysctl. Se `sysctl -a` for alle options. Se mansiderne for sysctl(3) og (8).

2.2.2 kern.securelevel

Se `man securelevel`. Der er forskel på hvad der er tilladt i de enkelte levels (fra -1 til 3) i Free- og OpenBSD.

Root kan altid sætte højere securelevel, men ikke lavere, mens systemet kører.

2.2.3 Random pids

Nogle exploits beregner PID'er. Tilfældige PID'er er godt. Det gør OpenBSD altid. I FreeBSD skal det slåes til ved at sætte sysctl-variablen `kern.randompid` til 1

2.2.4 Core dumps

Coredumps indeholder vigtig info om crashede programmer. Bl.a. stackpointere og memoryadresser. Det kan være nyttigt for crackere. Hvis man ikke udvikler programmer selv, og vil have debug-mulighed, er det måske smart at slå fra. I FreeBSD sættes `kern.coredump` til 0

2.2.5 Netværkstweaking

Det kan være smart at reducere sin synlighed i netværk. Som default svarer man på forespørgsler til porte der ikke er åbne. Det slåes fra ved at sætte `net.inet.tcp.blackhole` til 2, og `net.inet.udp.blackhole` til 1. Dette tager en del netværksload af ens egen maskine, og det gør at angriberens skanning tager meget længere tid.

Scannere der identificerer OS, bruger tit pakker med SYN og FIN flag sat. Det er ikke normalt i "normal"traffik, så dem kan man godt droppe. Det gør man ved at sætte `net.inet.tcp.drop_synfin`, men i FreeBSD kræver det at man inkluderer option `TCP_DROP_SYNFIN` i sin kerne.

2.3 Kontrol af brugerprocesser

2.3.1 Systemkaldet chroot

Skaber virtuelle root-foldere for processer. Kræver man opretter /etc,/dev,/lib,/bin osv inde i det virtuelle system.

Man kopierer sin binary ind i den virtuelle root, og ser om den kører.

Hvis ikke, kan man bruge ldd til at se hvilke libs den kræver. Hvis det stadig ikke kører, kan ktrace bruges til at se hvad den leder efter.

Til at lave /dev, bruges mknod. For at se parametre, ses på ls -l af den rigtige /dev. F.eks:

```
ls -l /dev/random
```

```
crw-rw-rw- 1 root wheel 45, 0 Aug 14 12:12 /dev/random
```

Så bruges:

```
sudo mknod /chrootenv/dev/random c 45 0
```

Et chroot'et miljø er ikke bare perfekt. Processer kan stadig se andre processer. I FreeBSD kan man sætte sysctl-variablene `security.bsd.see_other_uids` og `security.bsd.see_other_gids` til 0. Men hvis PID's kan beregnes, kan de stadig dræbes.

2.3.2 jail (kun på FreeBSD)

Udover UID og GID har processer også en Jail ID (JID). De kan kun tale med processer med samme JID. Bortset fra grundsystemet som har JID 0, og godt kan påvirke andre JID'er.

På netværk kan chroot normalt alt, mens jail kun kan det der er specifikt tilladt.

Jails kan ikke køre mknod, hvorfor alt til jailets /dev _skal_ laves på forhånd. Det gør ikke så meget hvis mem eller kmem er krævet, da et jail ikke kan læse eller skrive til områder i mem eller kmem, som ikke har samme JID. Evt. kan man lave en /etc/devfs.conf i sit jail, så der bliver lavet de devs der skal bruges.

Man kan lave tynde og tykke jails. Et tyndt jail er nogenlunde som en chroot, mens et tykt jail nærmest er et helt komplet virtuelt system / virtuel maskine. Mansiden for jail(8) giver eksempler på hvordan man laver dem.

sysctl's til kontrol af jails

`security.jail.socket_unixiproute_only` er sat på som default, og den sikrer at kun enkelte netværkshandlinger er tilladte gennem jails.

`security.jail.sysvipc_allowed` er også sat som default, og styrer om jailede ting kan bruge sysV-ipc'er. Der kan laves avancerede sysV-ipc'er, der kan omgå jails, hvorfor denne er fin at have sat til.

Kommando til kontrol af jails

jexec kan bruges til at eksekvere processer inde i jails, ude fra hovedsystemet (JID 0). Den kan også bruges til at dræbe et helt jail (`sudo jexec 3 kill -1`)

2.4 Medfødt beskyttelse

OpenBSD leger secure by default, FreeBSD er lidt mere fleksibel

2.4.1 Bufferoverflow-beskyttelse

W$\hat{\text{X}}$ (Write xor execute) RAM-beskyttelse

RAM-sider er enten skrivbare eller eksekverbare. Ikke begge dele. Det er en del af kernen og loaderen (ld.so). Hvis et program skriver eller eksekverer en eksekverbar eller en skrivbar RAM-side, crasher programmet. Kernen og loaderen prøver at sørge for at programmer har data og instruktioner på hver deres sider, men nogle programmer er bare dårligt skrevet, og crasher så.

ProPolice Stack-beskyttelse

Normalt loades programmer til RAM ens hver gang de køres. ProPolice indsætter en tilfældig "canary-værdi i hver eneste funktionsramme. Når programmet returnerer, bliver der tjekket om denne værdi er ændret. Hvis den er det, crasher programmet. Der er $2\hat{3}2$ mulige værdier.

2.4.2 Kryptografi

Kryptering er til bl.a. HTTPS, SSL, SSH, VPN og lign.

Udover fin softwarebaseret kryptering, indeholder både Free- og OpenBSD et krypto-API/-Framework for hardware-kryptokort. Se crypto(9) i FreeBSD og crypto(4) i OpenBSD. Der kan dog være problemer med nye kort med AES-algoritmer.

2.4.3 Code Audit

Både Free- og OpenBSD laver code audit før noget ryger ind i OS'et. Men OpenBSD laver også code audit af alt i ports, hvorfor der endnu ikke er helt så mange ports til OpenBSD som til FreeBSD. Ofte indeholder OpenBSD ports en del patches, for at rette op på usikkerheder, hvorfor den enkelt port så også er blevet specielt tilpasset OpenBSD.

2.5 Optimeringer

En del ting kan finjusteres i /etc/sysctl.conf og i FreeBSD i /boot/loader.conf

FreeBSD forsøger at skalere sig selv pænt, ved at se på CPU og RAM.

2.5.1 maxusers

I OpenBSD er det en kernevariabel, der kræver rekompilering af kernen.

I FreeBSD er den dynamisk, og udregnes ved boot, når den står til 0 (default)

2.5.2 maxfiles og somaxconn

Hvor mange filer må være åbne, og hvor mange indgående tcp-forbindelser må oprettes. Det sættes med kern.maxfiles, og med kern.ipc.somaxconn i FreeBSD og kern.somaxconn i OpenBSD.

2.5.3 Netværksbuffere

Hvis der skal overføres store filer, er det godt at sætte mængden af RAM til netværksbuffere højt. Det er net.inet.tcp.sendspace.

Hvis der skal modtages store filer, er det smart at sætte `net.inet.tcp.recvspace` højt.

Hvis man har for få mbufs (netstat -m, se limits), kan net.ipc.nmbclusters sættes om i FreeBSD. I OpenBSD er det en del af maxusers, men kan også sættes som en kerneoption NMBCLUSTERS.

Kapitel 3

Installation

3.1 Generelt

3.1.1 Overvej hvad du bygger

Eksempelvis workstation, hvor folk logger direkte ind, workgroupserver, hvor flere folk logger ind via fjernadgang, infrastrukturserver, som andre servere stoler på, og henter data fra. Jo længere henne, jo mere sikkerhed behøves måske.

Man kan også være ude for at lave multipurpose servere, som måske dækker over de to sidstnævnte i ovenstående, eller endnu mere.

3.1.2 Medier og netværk

Installeres fra originale cd'er, bør indholdet være korrekt. Hvis installationen foregår via netværk, kan der ske man-in-the-middle-angreb, så man får korrupt data ind. En umiddelbar sikkerhed er at installere fra eget kontrolleret repositorie. Kør altid md5-tjek på downloadede pakker, før de bruges. Optimalt hvis installationen foregår fra egne maskiner, helt uden internetforbindelse. Først når maskinen er godt sikret, gives internetforbindelse.

DHCP er uden autentifikation eller ægthedstjek, så undgå endelig dette, i alle sammenhænge!

3.1.3 Præeksisterende sårbarheder

Der kan være huller i systemer når de udkommer. FreeBSD skriver om dette på http://www.freebsd.org/security/, og OpenBSD på http://openbsd.org/errata.html Begge har jo også maillinglister for emnet, så husk at tjek der.

Hvis du installerer en ældre udgave end nyeste, så er dette endnu vigtigere!

Og husk self. at patche FØR du giver internetforbindelse til maskinen.

3.1.4 Opdeling af filsystemet

Man kan f.eks. opdele i /, swap, /tmp, /usr, /var, (og /home på OpenBSD)

Det er godt at dele filsystemet op i flere partitioner. Det er det fordi:

Integritet

Hvis strømmen tages fra et system, vil det være de partitioner der sker flest ændringer på, der tager størst skade. Det vil sige at /var og /home måske rammes, men ikke / og /usr

Tilgængelighed

Nogle partitioner må ikke løbe tør for plads, f.eks. /. Nogle gange kører en proces måske vild i /tmp, og med opdeling, bliver skaden begrænset.

Sikkerhed

Mount options kan øge sikkerheden. F.eks. nosuid, noguid, readonly og muligheden for at eksekvere binære filer kan ændres på enkelte partitioner.

Hastighed

Fragmentation kan ikke ske i nær så høj grad, når der er flere mindre partitioner.

3.1.5 X

X tager klient/server-modellen helt op på applikationsniveau. En X-server kan være farlig. Kan måske køre ud til netværket på en workgroup server, men ellers bør den være slået helt fra udadtil, hvis X overhoved skal eksistere.

3.2 Opdater systemet

Opdater dit system efter endt installation. Installer cvsup-without-gui.

Se efter en brugbar supfile. Kig i /usr/share/examples/cvsup

Linjen *default release=cvs tag=RELENG_5_3 siger at den opdaterer til STABLE. Hvis tag=., opdateres til CURRENT.

Og opdater: `cvsup -L 1 -h cvsupx.freebsd.org /path/to/supfile`

Hvor x er et mirror i din nærhed.

Du kan gemme update-konfigurationen i /etc/make.conf:

```
SUP_UPDATE=yes
SUP=/usr/local/bin/cvsup
SUPFLAGS= -g -L 2
SUPHOST=cvsup.dk.FreeBSD.org
SUPFILE=/usr/local/etc/standard-supfile
```

Og du kan så bruge make fra /usr/src til at opdatere med.

3.3 Første upgrade

Chapter 21 af FreeBSD handbook

Læs /usr/src/UPDATING

Lav en /etc/make.conf med gode indstilliner. Se man make.conf

Læs evt man mergemaster, hvis der er konfigurationer der skal merges.

3.4 OpenBSD installation

Husk:

Se hvilke flags mount automatisk sætter på de enkelte partitions. Måske vil man godt kunne lave devs i /var, til chrooted named.

Læs man afterboot

Lav en brugerkonto til dig selv, konfigurer sudo "username ALL = ALL", eller hellere kun til hvad du har behov for.

3.4.1 Opdater systemet

cvs-checkout for den patch branch du vil bruge. Se liste af cvs'er på openbsd.org/anoncvs.html

```
setenv CVSROOT anoncvs@some.server.tld:/cvs
cd /usr
cvs checkout -P -r OPENBSD\X\X src
```

Hvor X_X er den branch du bruger, f.eks. 3_8. Genbyg din kerne, og kør make world.

3.5 Hærdning efter installation og opdatering

3.5.1 Konfigurer brugere og grupper

Opret brugere f.eks. i /home og i /var/www/htdocs

Tillad dem at bruge ftp, hvis det er nødvendigt.

Skal de have mail-aliaser?

Opsæt quota af disk og andet for dem.

Medlemmer af gruppen wheel, kan læse devices, og bruge su til root.

Brugeren toor på FreeBSD er nærmest historisk. Slet den, hvis du ikke ved du skal bruge den.

3.5.2 Indstil mount options

Se efter at mount options er korrekte. Se i fstab og i man mount.

Se bl.a. noauto, nodev, noexec, nosuid, rdonly, suiddir

3.5.3 Indstil sshd

Brug keys!

Følg en guide i at opsætte sshd med nøgler i stedet for passwords, og husk passphrases til nøglerne. Husk at sætte PasswordAuthentication til NO

Læs man sshd

Lav en liste af tilladte users med AllowUsers eller AllowGroups. Lav f.eks. en gruppe sshers

Sæt PermitRootLogin til NO, så man ikke kan logge direkte ind med root.

Sæt Protocol til 2, da der er mulighed for mitm-angreb med 1.

Se StrictModes står til YES. Den tjekker at korrekte permissions er sat på de filer der har med ssh at gøre.

Sæt Port til noget andet end 22, for at forhindre en masse automatiske angreb. Gerne noget over 1024, eller højere, så default portskannere ikke ser den.

Se UsePrivilegeSeparation står til YES. Den spawner en børneproces når en bruger logger ind, og kører sessionen som denne bruger.

Slå al X11 fra, med mindre du ved det skal bruges.

3.5.4 Opsæt logning

Log hellere for meget end for lidt. Læs man syslog.conf og newsyslog.conf

3.5.5 Lav login bannere

Lav login bannere, så folk får velkomster, og f.eks. advarsler om logging.

/etc/motd, sshd_config's Banner, og tcpwrappers via inetd.

Kør inetd med -w, og sæt bannerdirektiver i /etc/hosts.allow

3.5.6 Opsæt NTP

Ved boot kan den sættes med f.eks. rdate eller ntpdate.

ntpdate kan konfigureres med sysinstall på FreeBSD

Eller sættes op selv i /etc/rc.conf med:

- ntpdate_enable="YES"
- ntpdate_flags=-b public.ntp.adr.tld"

rdate kan via /etc/rc.conf.local på OpenBSD

- rdate_flags=-n public.ntp.adr.tld"

Se www.eecis.udel.edu/ mills/ntp/servers.html for liste over servere

ntpd

Både rdate og ntpdate trækker kun tid fra een server. Hvis denne er forkert, eller forbindelsen er langsom, kan man få noget forkert ned. Der er ntpd bedre. Den tjekker på flere, og gør nogle beregninger.

ntpd sættes op med

`ntpd_enable="YES"` i FreeBSD, og

`ntpd="YES"` i OpenBSD

Det er vigtigt at lave en conf-fil, med minimum een server i. Læs man ntp.conf

Kapitel 4

Administration

4.1 Adgangskontrol

Hold systemets konti og brugeres konti skarpt adskildte. Systemkonti som named og _bitlbee skal ikke have shells.

Det er en god ide at lade alle brugere have en gruppe med samme navn som deres brugernavn, og derudover også lade alle rigtige brugere være med i en fælles gruppe "users"f.eks.

Sørg for at nye ting brugere laver, bliver lavet med deres egen gruppe.

Resursebegrænsninger, miljøvariable og lignende kan sættes i /etc/login.conf

Som default er der en umask på 022. Måske er en umask på 077 smartere. Det kan også sættes i login.conf, og den enkelte bruger kan selv ændre det hvis denne vil.

Undgå klartekst logins som telnet, rsh og rlogin, brug i stedet ssh.

Brug sudo i stedet for su, men ikke "sudo su". Mange programmer kan kalde shells (f.eks. vi og less), så pas på med hvad der tillades med sudo.

Vær specifik i sudoers-linjer. Man kan sagtens give de enkelte kommandoer de tilladte parametre.

4.2 Hverdagsting

Kan ske gennem ports eller packages. Ports tilbyder mere konfiguration og ofte nyere versioner.

Overvej at bruge CVS eller lign. til styring af conf-filer. Så er der altid styr på dem.

4.2.1 Datagendannelse

Kan typisk deles op i fire kategorier:

- Katastrofegendannelse, hvor et helt system er forsvundet. F.eks. ved diskkrasj, brand, indbrud eller lign.
- Datagendannelse, hvor en bruger eller program har fået slettet data der ikke skulle slettes.
- Efterforskning, eksempelvis efter man har haft fremmede i systemet.
- Retssager, hvor man f.eks. skal genfinde gamle mails og lign.

Ofte er programmerne dump eller amanda gode til backup.

Datakompletheden sikres ikke med tar. Men det gør den med dump. Derfor er dump god til at tage backup af et helt system.

Tænk over sikkerheden i backuppen. Måske er det smart at kryptere sin backup.

Hvis backuppen kører over netværk, så husk at kryptere trafikken. F.eks. med ssh eller vpn.

Hvis ssh, er det smart at bruge keys, måske uden passphrases. Så er det vigtigt at sætte en "From=¡hostname¿"i auth_keys, så kun de(n) korrekte maskine kan komme ind.

4.3 Opgradering

Det er bedre at opdatere programmerne ofte, end sjældent. Ellers kommer der lige pludselig mange opdateringer, og tingene går nemmere galt.

Husk at patche hurtigt, når sikkerhedspatches udkommer, ellers glemmes det nemt.

Man bør opdatere branch, når der kommer en ny. Både Open- og FreeBSD har god dokumentation om det.

4.4 Sårbarhedsrespons

Følg med på postlister. Både Open- og FreeBSD's, securityfocus, nogle applikationsspecifikke, og måske freshports til FreeBSD. Mange lister har announce-only, som er lavvolume-lister, hvor der kun kommer vigtige ting.

Husk på at automatiske angrebsscripts ikke tænker over om din server er stor eller ej. Derfor bør du altid regne med at blive angrebet.

Lav en reaktionsliste for hvordan forskellige sikkerhedshuller skal håndteres.

Husk der er forskellige typer sikkerhedshuller. De der kan køres af alle udefra, er farligere end de der kræver lokal brugerkonto. De der giver umiddelbar root-adgang, er værre end de der blot lammer dele af systemet.

Der er en række forskellige handlinger der kan udføres når der kommer oplysninger om et sikkerhedshul:

- Lappe hullet NU, hvis der er patch, det er vigtigt, og der er tid.
- Lappe hullet indenfor kort tid, hvis der er patch men det er mindre vigtigt.
- Lappe hullet ved næste alm. vedligeholdelsesomgang, hvis det ikke er specielt vigtigt.
- Lukke servicen der er hullet og senere lappe, hvis det er vigtigt, men der ikke er tid. Det kan f.eks. bare være at ændre firewallen.
- Gøre ingenting, hvis det er komplet irrelevant. Dog kan det det jo blive relevant senere hen.

4.5 Netværksservices

4.5.1 inetd

inetd tillader ofte en del gammeldags og usikre services som finger, telnet og rlogin. Slå dem fra!

Normalt tillader inetd 256 forbindelsesoprettelser pr sekund. Med mindre du har så mange, så sænk det tal, med parametren -R (rate)

Brug tcpwrappers. Det gør det nemt at styre forskellige regler for forskellige hosts.

4.5.2 Ting der skal undgåes

NFS kan ikke sættes sikkert op. Undgå det! Brug CFS, SFS eller lign., hvis der skal bruges et netfilsystem.

YP / NIS. Hvis det er noget centralt auth der skal bruges, er kerberos med LDAP smart.

Hvis man skal have nogle filer distribueret ofte, som ala NIS, kan man bruge scp. Alternativt CVS/SVN over ssh.

4.5.3 NTP

Hvis der ikke holdes korrekt orden i tiden på servere, kan der bl.a. ske problemer som:

- umuligt at lave efterforskningsspor
- CVS/SVN
- IRCd'er
- Stempling af mails ud

ntp er i base-systemet på både open- og freebsd.

Config-filen hedder:

- /etc/ntp.conf på freebsd

- /etc/ntpd.conf på openbsd

Tilladelser bør ske som med en firewall. Altså først nægt alt, og så tillad enkelte.

Forbindelser kan ske med auth gennem key kryptering. Der er en smart kommando ntpd-genkeys, og en god dokumentation på ntp.org

4.6 Hold øje med systemet

F.eks. med nagios. http.//nagios.org/

For at kunne lave lokale tjek på fjernmaskiner, skal NRPE (Nagios Remote Plugin Executor) installeres. HUSK at bygge det med SSL (make WITH_SSL=yes).