# inetd

- inetd (”Internet super server daemon”)  on ohjelma (demoni), joka kuuntelee useita tcp- ja udp-portteja ja liikennettä havaitessaan käynnistää ko. portille määritellyn ohjelman (demonin).
- Säästää muistia jos ko. palvelua ei tarvita koko aikaa ja yksinkertaistaa niiden koodaamista, kun niiden ei itse tarvitse osata verkkoprotokollia vaan inetd välittää niille yhteyden stdin'in ja stdout'in kautta ja tarjoaa tcp wrapper -toiminnallisuuden myös.
- Useita toteutuksia, jotka tarjoavat  useita saman perustoiminnallisuuden plus vaihtelevasti muuta, Linux-ympäristöissä ehkä yleisin xinetd. (Alkuperäinen ”inetd” on lähes kadonnut, sanaa käytetään yleisterminä sen toiminnallisuudelle.)
- Monet yleiset demonit voi konfiguroida toimimaan joko itsenäisinä tai inetd'n kautta. Useat inetd-toteutukset tarjoavat joitakin palveluita sisäisestikin (tyypillisesti ainakin echo ja discard).

# xinetd

- Konfiguraatiotiedosto /etc/xinetd.conf: ei yleensä tarvitse muuttaa muuten kuin lokituksen säätöä varten.
- Palvelukohtaiset konfiguraatiot hakemistossa /etc/xinetd.d, esim. /etc/xinetd.d/time (lyhennetty):

```
service discard
{
        disable         = yes
        type            = INTERNAL
        id              = discard-stream
        socket_type     = stream
        protocol        = tcp
        user            = root
        wait            = no
}
```

- Paljon muitakin asetuksia erilaisiin tarpeisiin.
- Porttinumero tulee oletuksena /etc/services -tiedostosta (ellei, sen voi antaa **port**-määreellä konfiguraatiotiedostossa).

# systemd

- Uusi ”superdemoni”, korvaa boottiskriptit, inetd:n, syslogd:n ja kaikenlaista muutakin.
- yleinen ohjauskomento systemctl, esim.

  systemctl start sshd.service

  systemctl enable sshd.service

  systemctl list-unit-files --type=service

  - paljon muitakin, paitsi .service tyyppejä on mm. .target, .device, .mount, .swap ...
- Lokitiedostot binäärisiä, lukemiseen tarvitaan komento journalctl
- ”unit files” = tapahtumien määrittelytiedostot, paljon mahdollisuuksia koska niitä käytetään monenlaisiin tarkoituksiin mutta eivät sinänsä erityisen monimutkaisia
- ks. /etc/systemd/...

# DNS

- Nimipalvelu (Domain Name Service) yhdistää nimet IP-osoitteisiin (ja päinvastoin).
- Globaali hajautettu tietokanta, jota hallinnoi ICANN.
- Palomuurin sisäpuolella usein on yksityinen DNS-palvelin, joka tuntee sen privaattiosoitteet. Siten saman koneen IP voi näyttää erilaiselta sen mukaan kysytäänkö sitä palomuurin sisä- vai ulkopuolella (”split DNS”).
- Yksittäisessä koneessa voi olla vain sen tiedossa olevia nimi-osoitepareja tiedostossa /etc/hosts. (Kaikki palvelut eivät käytä /etc/hosts'ia vaikka sellainen olisikin, erityisesti sähköpostipalvelimet yleensä eivät.)
- Nimipalvelimia on kaksi perustyyppiä:
  - *authoritative nameserver* tuntee ja hallitsee itse jonkin (sille delegoidun) domainin nimet; sisäisesti voi olla *master* tai *slave*
  - *recursive nameserver* ei tunne itse nimiä, mutta osaa hakea niitä maailmalta; *caching nameserver* pitää lisäksi vanhoja vastauksia tallessa ja palauttaa uudelleen kysyttäessä ne saman tien hakematta niitä uudestaan (elleivät liian vanhoja ts. TTL ei ylittynyt)
  - sama palvelin voi olla sekä rekursiivinen että autoritatiivinen (ei yleensä suositeltavaa)
- Nimipalvelinohjelmia on useita, mm. referenssitoteutus **bind** (named), **nsd** ja **dnsmasq**.

# DNS records

- DNS-tietokanta muodostuu tietueista (records), joita on useita tyyppejä, tärkeimmät:

  A koneen IPv4 -osoite

  AAAA koneen IPv6 -osoite

  CNAME alias, viittaa toiseen nimeen (ei osoitteeseen)

  MX koneen (tai domainin) sähköpostia välittävän koneen nimi

  kohdenimen pitää olla A- tai AAAA-record, ei CNAME

  TXT vapaa tekstikenttä

  PTR pointer, yhdistää osoitteen nimeen

  NS domainin nimipalvelimen nimi (osoite)

  SOA Start Of Authority, domainin perustiedot

- Kaikilla tietueilla on oma voimassaoloaikansa (TTL, Time To Live), tyypillisesti muutamasta tunnista muutamaan vuorokauteen (kannattaa lyhentää jos odotettavissa on muutostarpeita)

# DNS: kyselytyökaluja

- DNS-tietojen tutkimiseen on useita työkaluja:

  host [*options*] [*name*] [*server*]

  - Tärkeimmät optiot:

    -t *type* minkätyyppisiä tietueita haetaan (ANY = kaikki)

    -v verböösimpi tulostus

    -a sama kuin ”-v -t ANY”
  - *server* on halutun nimipalvelimen osoite (nimi)
  - *name* voi olla myös osoite

  dig [*@server*] [*options*] [[-q] *name*] [[-t] *type*] [*class*] [*queryoption...*]

  - kyselyoptioita on paljon, mm.

    +[no]tcp käytetäänkö tcp:tä udp:n asemesta

    +[no]trace näytetäänkö rekursiivinen hakuketju

    +[no]showsearch näytetäänkö välituloksia

    +[no]recurse tehdäänkö rekursiivinen haku

    +[no]short lyhyt tulostusmuoto

    +[no]stats näytetäänkö statistiikkaa (aikoja, kokoja jne) hausta

# DNS zone files

- Tietyn (delegoidun) domainin DNS-tiedot sisältävä tiedosto (voi olla oikea tiedosto tai tietokantakin). Esimerkki (klassinen muotoilu, jota sekä bind että nds käyttävät):

```
; tkvk.org.zone
$TTL        86400
@           IN    SOA   ns1.tkvk.org. hostmaster.tarvainen.info. (1 43200 3600 604800 86400)
            IN    NS    ns1.tkvk.org.
            IN    NS    ns2.tkvk.org.
            IN    MX    10    hauki.tapanitarvainen.fi.
            IN    MX    10    leuka.tarvainen.info.
            IN    MX    90    kannel.tarvainen.info.
            IN    A     80.66.162.88
ns1         IN    A     80.68.90.32
ns2         IN    A     64.79.206.244
www         IN    A     80.66.162.88
www2        IN  CNAME   www.tktk.org.
```

- $TTL määrää oletus-TTL:n, @ viittaa domainiin itseensä (”tkvk.org”), domain-kentän puuttuessa se tulkitaan samaksi kuin edellinen.

# DNS: PTR

- PTR-tietue (*reverse* DNS) kertoo tiettyä IP-osoitetta vastaavan nimen. Se on toteutettu erityisellä pseudo-TLD:llä ”**in-addr.arpa”:**

  $ host 130.234.208.16

  16.208.234.130.in-addr.arpa domain name pointer lonka6.it.jyu.fi.

  - IPv6:lle on vastaavasti ip6.arpa:

    $ host 2001:41c8:1:5292::10

    0.1.0.0.0.0.0.0.0.0.0.0.0.0.0.0.2.9.2.5.1.0.0.0.8.c.1.4.1.0.0.2.ip6.arpa domain name pointer hauki.tapanitarvainen.fi.

- PTR-tietuetta ei määritellä samassa zone-tiedostossa eikä useinkaan edes samalla palvelimella kuin muut, nimiin liittyvät tietueet, vaan ao. ip-osoiteavaruuden osan palvelimella. Käytännössä PTR-tietueen muutokset pitää yleensä pyytää erikseen palveluntarjoajalta, vaikka muuten ylläpitäisi omaa nimipalveluaan.
- Erityisesti sähköpostipalvelimilla pitäisi A-tietueen ja PTR:n vastata toisiaan, muuten postikulussa voi tulla ongelmia.

# Dynaaminen DNS

- Joissakin ympäristöissä (erityisesti kotiliittymät ja mobiililiittymät) koneiden IP-osoitteet voivat muuttua automaattisesti. Dynaaminen DNS tarjoaa mahdollisuuden päivittää niiden nimitieto (lähinnä A record) automaattisesti osoitteen muuttuessa.
- DNS-palvelimen ominaisuus, riippumaton osoitteen määräävästä ISP:stä
- Yksinkertainen mekanismi: asiakaskone ottaa yhteyden nimipalvelimeen, autentikoi itsensä jollakin tavalla ja ilmoittaa palvelimelle uuden osoitteensa
- Sisäänrakennettuna useissa palomuurilaitteissa (WLAN-tukiasemissa, ADSL-modeemeissa jne)

# WHOIS

- Globaali tietokanta, jossa on domainien haltijoiden yhteystiedot, ja protokolla niiden hakemiseen.
- Nimipalvelurekisterien ylläpitämä, luotettavuus vaihtelee, usein julkinen tieto on proxy.
- Komentoriviclient ”whois”, esim:

```
$ whois jyu.fi
domain:   jyu.fi
descr:    Jyväskylän yliopisto
…
```

- Aina whois ei automaattisesti löydä oikeaa palvelinta. Tällöin sitä voi joutua kutsumaan useaan kertaan: kunkin top-level domainin whois-palvelimen pitäisi aina löytyä IANA:n palvelimelta, ja sieltä edelleen ao. TLD:n domainit:

```
$ whois -h whois.iana.org fi
…
whois: whois.fi
$ whois -h whois.fi jyu.fi
```

- Whois-järjestelmän tulevaisuus on epävarma, muuttunee tai ehkä korvataan kokonaan eri järjestelmällä.

# dnsmasq

- Yhdistetty DHCP- ja DNS-palvelin erityisesti pienten sisäverkkojen tarpeisiin
- DNS-toiminnallisuus rajoitettu, ei sovellu julkiseksi (eikä varsinkaan autoritatiiviseksi) palvelimeksi (ensisijaisesti vain välityspalvelin, *caching dns-forwarder*)
- Tarjoaa myös TFTP-toiminnallisuuden verkkoasennuksia varten
- Hyvin yleinen erilaisissa Linux- ja *BSD-pohjaisissa palomuuripaketeissa
- Myös (Ubuntun/Debianin) libvirtd:n sisäisen (default) verkon dns/dhcp/tftp -ratkaisu
- Konfiguraatio /etc/dnsmasq.conf, lukee (yleensä) myös /etc/hosts ja /etc/resolv.conf -tiedostoja

# SSL/TLS

- SSL = Secure Sockets Layer, TLS = Transport Layer Security. Periaatteessa TLS on uusi versio SSL:stä, käytännössä termiä ”SSL” käytetään geneerisenä molemmista.
- Salakirjoitusprotokolla, perustuu ns. julkisen avaimen salakirjoitukseen (salaukseen käytetään myös symmetristä ts. jaetun salaisen avaimen salakirjoitusta).
- Kaksi funktiota: autentikointi ja salaus.
- Samaa tekniikkaa voidaan käyttää myös toisinpäin, käyttäjän autentikoimiseen palvelimelle.
- Itse allekirjoitettua sertifikaattia voi käyttää omien koneidensa välillä, ja muutenkin pelkkään salaukseen (ilman autentikointia), mutta nykyisin selaimet valittavat aggressiivisesti sertifikaateista, joita eivät tunnista.
- Kaupalliset sertifikaatit maksavat n. 10€/vuosi tai enemmän, ei-kaupallisiin tarkoituksiin saattaa saada ilmaiseksikin (esim. startssl.com, edellyttää omaa domainia); ks. myös https://letsencrypt.org
- Sertifikaatti voi olla yhdelle tai useammalle host-nimelle tai wildcard koko domainille (*.jyu.fi).

# Sertifikaatit

- Sertifikaatti sisältää
  - tunnistetietoja, erityisesti DN (Distinguished Name, tässä yhteydessä domain-nimi kuten ”kone.example.org”), erilaisia osoitetietoja mukaanlukien yhteyssähköpostiosoite allekirjoitettuna sertifikaatin haltijan salaisella avaimella
  - haltijansa *julkisen avaimen*, jota vastaavalla salaisella avaimella haltija voi todistaa sertifikaatin omakseen
  - sertifikaatin myöntäjän salaisella avaimella tehdyn allekirjoituksen, jonka voi tarkistaa vastaavalla julkisella avaimella
- Ottaessaan yhteyden suojattuun palvelimeen asiakas (selain tms) ensin tarkistaa sertifikaatin aitouden hallussaan olevalla myöntäjän julkisella avaimella ja sitten varmistaa palvelimen aitouden sertifikaatissa olevalla haltijan julkisella avaimella.
- Autentikointi siis edellyttää, että käyttäjä (selain, sähköpostiohjelma tms) tuntee entuudestaan julkisen avaimen, jolla palvelimen sertifikaatti (avain) on allekirjoitettu (tai avaimen, jolla on allekirjoitettu avain, jolla sen on allekirjoitettu, joskus ketju voi olla pitkäkin); sertifikaattien myöntäjien pitää siis saada oma julkinen avaimensa kaikkiin (yleisiin) selaimiin, mikä yleensä maksaa.

# Sertifikaatin muodostus

- Sertifikaatti luodaan seuraavasti:

  (1) Hakija luo itselleen avainparin (julkisen ja salaisen avaimen) ja

  (2) sertifikaattipyynnön (Certificate Signing Request), joka sisältää hakijan tunnistetiedot sekä julkisen avaimen allekirjoitettuna salaisella avaimella, ja

  (3) sertifikaatin myöntäjä (tarkistettuaan enemmän tai vähemmän luotettavasti hakijan tiedot) allekirjoittaa CSR:n omalla salaisella avaimellaan ja luo siten siitä sertifikaatin (CRT) ja toimittaa sen hakijalle.
- Sertifikaatin käyttö siis edellyttää vastaavan salaisen avaimen hallussapitoa. Jos salainen avain kaapataan, kaappari voi esiintyä sen haltijana kunnes sertifikaatti *revokoidaan* eli tehdään mitättömäksi: myöntäjä ylläpitää listaa revokoiduista avaimista, jota käyttäjien (ohjelmien) pitäisi seurata. Käytännössä kaapattua avainta voi usein käyttää kunnes se vanhenee (hyvä syy sertifikaattien määräaikaisuudelle).
- Palvelimellakin tarvitaan myöntäjän julkinen avain, nämä tulevat yleensä valmiina (Ubuntussa /etc/ssl/certs/*CA.pem) ja mahdollisesti (jos myöntäjä on jälleenmyyjä) ketjutiedosto (olennaisesti jälleenmyyjän avain, jonka on allekirjoittanut ”tukkukauppias”; ketju voi olla pitempikin).

# CSR:n luonti

- Avainten ja sertifikaattien hallintaan käytetään komentorivityökalua **openssl**
- Oma avainpari luodaan tähän tapaan (genpkey'n asemesta käytetään usein yhä vanhempaa genrsa -komentoa, silloin optiot menevät hieman toisin):

  openssl genpkey -algorithm RSA -out oma.pem -aes-256-cbc -pkeyopt rsa_keygen_bits:4096
- Syntyneen avainparin salaisen avaimen (molemmat avaimet ovat samassa tiedostossa) käyttö edellyttää aina sen passphrasen syöttämistä. Jos sitä ei haluta syöttää aina kun www-palvelin käynnistyy, luodaan siitä suojaamaton versio:

  openssl rsa -in oma.pem -out oma.key
- Syntynyt tiedosto  sisältää sekä salaisen että julkisen avaimen. Sillä voi nyt luoda CSR:n tähän tapaan:

  openssl req -new -key oma.key -out oma.csr
- Syntynyt CSR lähetetään sertifikaatin myöntäjälle (certificate authority), joka palauttaa sitä vastaavan sertifikaatin (esim. oma.crt).
- Huom. tiedostojen nimikonventiot ja muukin terminologia vaihtelevat, erityisesti *.pem ja *.key voivat sisältää milloin mitäkin.
- Sertifikaattien myöntäjillä on yleensä myös web-työkaluja sertifikaattien käsittelyyn.

# Käärmeöljyä

- Testaamiseen ja omien koneiden välisiin yhteyksiin voi käyttää itse allekirjoitettua sertifikaattia. Se tapahtuu yksinkertaisesti allekirjoittamalla luotu CSR omalla avaimella:

  openssl x509 -req -days 365 -in oma.csr -signkey oma.key -out oma.crt

- Allekirjoitusta varten voisi myös luoda eri avaimen (esim. yrityksen oman, jota vastaava julkinen avain sitten talletettaisiin työntekijöiden koneisiin)
- Kummassakin tapauksessa avain ja sertifikaatti talletetaan (paikka periaatteessa vapaa, tässä Ubuntun oletushakemistot):

  cp oma.crt /etc/ssl/certs

  cp oma.key /etc/ssl/private/

  chown root:ssl-cert /etc/ssl/private/oma.key

  chmod u=rw,g=r,o= /etc/ssl/private/oma.key

- Ubuntun openssl-paketin asennus luo samalla itseallekirjoitetun ”snakeoil” -sertifikaatin tiedostoihin /etc/ssl/private/ssl-cert-snakeoil.key ja /etc/ssl/certs/ssl-cert-snakeoil.pem, joita voi myös käyttää testaukseen.

# https

- https = http SSL:n yli
- käyttää (oletuksena) porttia 443
- normaalisti käyttää vain IP:tä ts. virtual hostit eivät toimi; sitä varten on kehitetty laajennus SNI (Server Name Indication), mutta se ei toimi ihan kaikilla selaimilla
- asennus erilainen eri palvelinohjelmille, mutta perusaskeleet samat:
  - hankitaan sertifikaatti edelläkuvattuun tapaan ja asennetaan se ja vastaava salainen avain sopiviin paikkoihin
    - lisäksi voidaan tarvita allekirjoitusketjutiedosto
    - ohjelmasta riippuen em. tiedostoja voi joutua yhdistelemään eri tavoin
  - konfiguroidaan palvelinohjelma käyttämään https:ää (usein saman tien pakko-ohjataan http-yhteydenotot https:ään)
  - avataan asianomaisiin palomuureihin tarvittavat reiät
  - testataan että kaikki toimii (mielellään usealla eri selaimella)
  - laitetaan kalenteriin muistutus sertifikaatin vanhenemispäivästä (voimassaoloaika vaihtelee, useimmiten yksi vuosi), että muistetaan hakea uusi ajoissa!

# lighttpd & https

- Lighttpd haluaa avaimen ja sertifikaatin samassa tiedostossa.
- Ubuntun mukana tulleella käärmeöljysertifikaatilla:

  ln -s ../conf-available/10-ssl.conf /etc/lighttpd/conf-enabled

  cat /etc/ssl/private/ssl-cert-snakeoil.key /etc/ssl/certs/ssl-cert-snakeoil.pem >/etc/lighttpd/server.pem

  service lighttpd restart
- Edellä itsetehdyllä sertifikaatilla vastaavasti:

  cat /etc/ssl/private/oma.key /etc/ssl/certs/oma.crt >/etc/lighttpd/server.pem
- Haluttaessa https usealle virtual hostille (/etc/lighttpd/conf-available/... -tiedostoon):

```
$SERVER["socket"] == ":443" {
        ssl.pemfile = "/etc/lighttpd/server.pem"   # oletus
        $HTTP["host"] == ”tt1.student.it.jyu.fi" {
           ssl.pemfile = "/etc/lighttpd/ssl/tt1.student.it.jyu.fi.pem"
        }
        $HTTP["host"] == "s019.vm.it.jyu.fi" {
            ssl.pemfile = "/etc/lighttpd/s019.vm.it.jyu.fi.pem"
        }
    }
```

# nginx & https

- Nginx:n https-konfiguraatiossa tarvitaan minimissään seuraavat rivit server{} -blokkiin:

```
listen 443 ssl;
...
ssl on;
server_name example.org;
ssl_certificate /etc/nginx/ssl/example.crt;
ssl_certificate_key /etc/nginx/ssl/example.key;
```

- Nyt siis sertifikaatti ja salainen avain pidetään eri tiedostoissa.
- Nginx:n https-konfiguraatiossa on enemmän säätövaraa, mm.

```
ssl_session_timeout 5m;
ssl_protocols SSLv3 TLSv1 TLSv1.1 TLSv1.2;
ssl_ciphers ... ;
ssl_prefer_server_ciphers on;
```

- Erityisesti tuo ssl_protocols -rivi on suositeltava (poistaa vanhat, turvattomat protokollat)

# DHCP

- Dynamic Host Configuration Protocol: jakaa IP-osoitteita
- Pyynnöt broadcast-liikennettä, eivät kohdistettuja
- Samassa aliverkossa voi yleensä olla vain yksi dhcp-palvelin (joskus kaksi jotka synkronoivat toimintansa), ettei samaa IP:tä annettaisi kahdelle asiakaskoneelle (”villi” dhcp-palvelin voi aiheuttaa paljon pahaa)
- Osoitteita voidaan jakaa dynaamisesti vapaiden osoitteiden poolista tai staattisesti asiakaskoneen hardware-osoitteen tai nimen perusteella (molempia voidaan tehdä samassa palvelimessa niin, että tunnetuille koneille annetaan kiinteät osoitteet ja muille dynaamiset)
- Palvelinohjelmistoja useita, mm. referenssitoteutus **ISC DHCPD** ja pienissä ympäristöissä (erityisesti myös virtuaalikoneiden kanssa) yleinen **dnsmasq**, joka toimii myös nimipalvelimena sekä tftp-palvelimena.

# TFTP

- Trivial File Transfer Protocol
- Hyvin yksinkertainen, vain autentikoimaton (anonyymi) download
- Käytetään erityisesti verkkoboottijärjestelyissä sekä erilaisten laitteiden firmware-päivityksissä
- Useita palvelintoteutuksia, mm. atftpd ja tftpd-hpa (klassinen tftpd ei toimi pxe-asennusten kanssa); esim. atftpd:n konfiguraatio kokonaisuudessaan /etc/default/atftpd, tftpd-hpa:n samoin /etc/default/tftpd-hpa; usein myös yhdistetty dhcp-palvelimeen (esim. dnsmasq)
- Konfigurointi ei yleensä vaadi kuin hakemiston, mutta useimmat toteutukset tarjoavat säätövaraa mm. aikarajoituksissa jne
- Myös komentoriviclientteja, lähinnä testaukseen:

```
$ atftp mytftp.example.org
tftp> get pxelinux.0
Received 27116 bytes in 0.1 seconds
```

# PXE

- Preboot eXecution Environment
- Koneen BIOSiin tai verkkokortin firmwareen tms rakennettu verkkoboottausmekanismi
- Olennaisesti DHCP- ja TFTP-clientit ja niiden päälle rakennettu boottausmekanismi: hakee ensin itselleen IP-osoitteen ja tarvittavat parametrit (kuten tftp-palvelimen osoitteen ja ladattavan tiedoston nimen) dhcp:llä ja sitten käyttöjärjestelmän (tai sen latausohjelman) tftp:llä
- Edellyttää dhcp-palvelimelta ominaisuuksia, joita kaikissa toteutuksissa ei ole; tarvittaessa ne voidaan järjestää erillisellä proxyDHCP-palvelimella, joka välittää tarvittavat lisäparametrit vain pxe-clienteiksi tunnistetuille koneille

# sudo

- sudo -komennolla voi suorittaa komentoja jonkun toisen käyttäjän (yleensä mutta ei aina rootin) oikeuksilla.
- Oikeusmäärittelyt tehdään /etc/sudoers -tiedostossa, nykyisin siihen usein sisällytetään kaikki /etc/sudoers.d -hakemistossa olevat tiedostot (lokaalit muutokset on helppo tehdä sinne).
- /etc/sudoers -tiedoston editointiin on oma komentonsa, visudo (kutsuu EDITOR-muuttujan määrittelemää editoria).
- sudoers-tiedoston perussyntaksi on:

  *kuka missä = (kenenä) mitä*

  - *kuka* = käyttäjä tai %ryhmä
  - *missä* = kone (usein ALL)
  - *kenenä* = minkä käyttäjän oikeuksilla komento suoritetaan (id:gid, usein ALL tai ALL:ALL)
  - *mitä* = komento jonka suoritus on sallittu (usein ALL)
  - lisämääreellä NOPASSWD: voidaan salasanan kysely jättää pois
- Esim.

  %sudo ALL=(ALL:ALL) ALL

  %libvirtd ALL=(root) /usr/bin/vmbuilder

  tt ALL=(ALL) NOPASSWD: ALL
- Komentojen rajaus vaikea tehdä hyvin, monista komennoista pääsee suorittamaan jopa shellin; usein turvallisempi vaihtoehto on ns. suid wrapper (pieni suid-ohjelma, joka suorittaa halutun komennon eikä muuta).