Ssh & sshd

- Kolme perustoimintoa:
  - Pääteyhteys
  - Tiedonsiirto (scp, sftp)
  - TCP-porttien edelleenohjaus (forwarding)
- Kryptografialla kolme funktiota:
  - Käyttäjän autentikointi (salasana tai käyttäjäkohtaiset avaimet lähtökoneessa ja kohdekoneessa authorized_keys)
  - Koneen autentikointi (lähtökoneessa known_hosts, kohdekoneessa konekohtaiset avaimet (etc/ssh))
  - Yhteyden salaus (kertakäyttöiset avaimet, generoidaan joka sessiolle)

Ssh: koneen autentikointi

- ”The authenticity of host [...] can't be established. RSA key fingerprint is [...]

  Are you sure you want to continue [...]”
- Kohdekoneen avainta ei tunneta ennestään: sitä ei löydy ~/.ssh/known_hosts -tiedostosta eikä /etc/ssh/ssh_known_hosts -tiedostosta
- Vastattaessa yes avain talletetaan käyttäjäkohtaiseen known_hosts -tiedostoon automaattisesti

Ssh: koneen autentikointi 2

- ”WARNING: REMOTE HOST IDENTIFICATION HAS CHANGED!”
- Kohdekoneen avain on erilainen kuin known_hosts -tiedostossa oleva
- Joko kohdekone (tai sen ssh-server) on asennettu uudestaan tai koko kone on vaihtunut **TAI** yhteys on kaapattu! Syy varmistettava ennen jatkotoimenpiteitä.
- Korjattavissa joko palauttamalla kohdekoneen avain (sen /etc/ssh -hakemistossa) tai poistamalla sen (vanhentunut) avain known_hosts -tiedostosta: virheilmoitus kertoo rivinumeron, poisto onnistuu käsin editoimallakin tai esim.

  sed -i 108d ~/.ssh/known_hosts

  ssh-keygen -f ~/.ssh/known_hosts -R kone

Ssh: koneen autentikointi 3

- ”Warning: the ECDSA host key for [...] differs from the key for the IP address [...]”
- Kohdekoneen avain on usein known_hosts -tiedostossa kahdesti, sekä koneen nimellä että IP-osoitteella, tuo varoitus tulee yleensä nimeen liitetyn avaimen poiston jälkeen
- Varoituksesta pääsee taas eroon poistamalla vanhentuneen avaimen known_hosts -tiedostosta (kuten edellä mutta nimen asemesta IP-osoite)
- Jos samassa IP:ssä on monta konetta (sshd eri porteissa) tai samassa IP:ssä on useita koneita (palomuurissa round robin –ohjaus tms), known_hosts -tiedostoon pitää laittaa (käsin) kaikkia tarvittavia yhdistelmiä vastaavat rivit

Ssh: koneen autentikointi 4

- /etc/ssh/ssh_known_hosts: vaikuttaa kaikkiin ko. konetta käyttäviin (sieltä ssh:lla ulos meneviin)
- Ylläpitäjä luo ja ylläpitää ettei käyttäjien tarvitse ihmetellä varoituksia avaimista
- Helpoin tapa luoda: ssh-keyscan
- Käyttäjäkohtainen known_hosts voidaan poistaa käytöstä (tai käyttäjältä poistaa oikeudet siihen) niin, ettei ssh:lla ylimalkaan pääse muihin kuin nimenomaisesti sallittuihin (ssh_known_hosts -tiedostossa lueteltuihin) koneisiin (konfiguraatiotiedostossa /etc/ssh/ssh_config asetukset UserKnownHostsFile ja StrictHostKeyChecking)

Ssh: käyttäjän autentikointi

- Login/password
  - Tehtävissä usealla eri tavalla (PasswordAuthentication, ChallengeResponse,), salasana voi olla lokaali tai tulla Kerberoksesta ym
- S/key (kertakäyttösalasanat): vanhentunut, ei suositeltava
- GSSApi: natiivi Kerberos-autentikointi
  - Kerberos-tikettipohjainen, ei kysele salasanaa
- Julkisen avaimen käyttö (public key)
  - Ei tarvitse salasanaa (mutta voi käyttää omaa salasanaa/fraasia avaimen suojaamiseen)
- Sallittujen/kiellettyjen koneiden listat: /etc/hosts.{allow,deny}

Ssh: julkisen avaimen käyttö

- PubkeyAuthentication
  - Julkinen avain (id_rsa.pub tms) ”lukko”, asennetaan koneeseen, johon halutaan päästä (authorized_keys); käytettävä avain valittavissa ssh:n -i -optiolla (ssh -i ~/.ssh/id_extra user@kone...)
  - Luodaan ssh-keygen [-t algoritmi] -komennolla
  - Salainen avain (id_rsa tms) suojattava salasanalla ellei tarkoitus ole käyttää ei-interaktiivisesti (erityisesti rootin avainta ei yleensä voi suojata salasanalla)
  - Algoritmit: rsa (oletus), dsa, ecdsa: rsa tällä hetkellä suositeltavin (dsa:ssa heikkouksia, ecdsa uusi ja huonommin tuettu)
  - Salainen avain voi olla myös smartcard'issa (turvallisempi, edellyttää lukijaa)

Ssh: yleisiä ongelmia

- Yleistä:
  - Testausta: ssh -v, katso loki (/var/log/{authlog,syslog}) kohdekoneessa
  - Pääseekö koneeseen edes siitä itsestään (ssh -v localhost)? Onko sshd käynnissä?
  - Onko /etc/hosts.{allow,deny} olemassa ja oikein? Palomuuri liian kireällä?
  - ~/.ssh/* ja /etc/ssh/ssh_config omassa koneessa, /etc/ssh/sshd_config kohdekoneessa
- Avainautentikointi ei toimi:
  - Ovatko tiedostojen ja hakemistojen omistajat ja suojaukset oikein?
  - Onko authorized_keys kunnossa? Onko siellä mahdollisesti ylimääräisiä kontrollimerkkejä (kokeile cat -vet:llä)? Puuttuuko lopusta rivinvaihto?
- Salasana-autentikointi ei toimi:
  - Onko Kerberos-konfiguraatio oikein (/etc/krb5.conf, dpkg-reconfigure krb5-libpam)?
  - Onko kello ajassa (ntpd/chrony asennettuna)?

Ssh: X forwarding

- ”X Window System”
- Etäkone pääsee käsiksi lokaaliin näyttöön
- Optiot -X, -Y, -x
  - -Y rajoittamaton lokaalin näytön etäkäyttö; vaarallinen, etäkone voi tehdä ”mitä vain”, kaapata näppäimistön, sotkea lokaaleja (X-)ohjelmia jne (aikojen alussa tämä oli ainoa vaihtoehto...)
  - -X rajoitettu ja turvallisempi: etäkoneen ohjelmat eivät pääse käsiksi lokaaleihin ja niiden käynnistymisaika on rajoitettu xauth-tokenilla. Silti myös -X on vaarallinen, etäkone pääsee helposti mm. seuraamaan näppäimistöä
  - -x estää X-forwardoinnin jos se on oletuksena päällä
  - tilapäisesti kumottavissa (esim. aliprosessille) tyhjentämällä ympäristömuuttuja $DISPLAY (DISPLAY='' ./proggis)

Port forwarding

-L [bind_address:]port:host:hostport

Ohjataan lokaalin koneen portti etäkoneen porttiin: lokaalin koneen porttia voidaan käyttää kuin se olisi etäkoneessa. Huom. *host* ei yleensä ole sama kone johon ssh-yhteys otetaan, vaan jokin kone sen verkossa. Jos se on palomuurin takana, nimipalvelu ei yleensä toimi vaan tarvitaan IP.

-R [bind_address:]port:host:hostport

Ohjataan etäkoneen portti lokaalin koneen porttiin: etäkone voi käyttää lokaalin koneen porttia kuin omaansa. Tässä *host* on jokin kone lokaalin koneen kanssa samassa verkossa (ja nimipalvelukin toimii).

-f Jätetään ssh taustalle

-N Ei suoriteta mitään komentoa (tehdään vain porttiohjaus)

-g Salli etäyhteydet lokaaleihin forwardoituihin portteihin

Local port forwarding

- Esimerkki: kotiverkossa on kone johon pääsee ssh:lla (”koti1”, sisäinen IP 192.168.1.5) ja toinen kone jossa pyörii ”koti-intranet” (”kotiwww”, 192.168.1.6). Komento (ulkopuolisesta koneesta):

  ssh koti1 -L 8080:192.168.1.6:80 -N -f

  ja selaimessa http://localhost:8080

  näkee koti-intranetin, mutta vain koneessa, jossa ssh pyörii.
- Jos portti halutaan jakaa muillekin, asetettava bind_address, esim.

  ssh koti1 -L '*:8080:192.168.1.6:80' -g -N -f

  tai se voi olla määrätty ssh_config- tai .ssh/config tiedostossa: GatewayPorts=yes
- Etäkoneelle pitää yleensä käyttää IP:tä jos se on palomuurin takana (nimipalvelukutsu tehdään lokaalissa koneessa, joka ei tiedä palomuurin takaisia nimiä)
- -L -optioita voi antaa useita (eri porteilla ja koneilla)

Remote port forwarding

- Esimerkki 1: Sama tilanne kuin edellä, mutta mennäänkin sisältä ulos avaamaan portti (kotikoneesta):

  ssh -R 8080:kotiwww:80 ulkokone

  ja jälleen (ulkokoneen!) portissa 8080 näkyy koti-intranet.
- Jälleen jako muille koneille bind_address -asetuksella:

  ssh -R '*:'8080:kotiwww:80 ulkokone

  mutta nyt tämä edellyttää, että se on sallittu ulkokoneen sshd_config -tiedostossa: GatewayPorts=yes (huom. eri merkitys kuin ssh_config'issa)
- Tässä nimipalvelu toimii kuten sisäverkossa yleensäkin, koti-intranetille voi käyttää nimeä (”kotiwww”)
- -R optioitakin voi antaa useita eri koneille ja porteille

Remote port forwarding 2

- Esimerkki 2: palvelin mokkulan päässä, dynaaminen IP, NAT, puhelinyhtiö ei salli yhteyksiä sisäänpäin. Avataan käänteinen ssh-yhteys etäkoneesta:

  ssh -R 52022:localhost:22 julkikone

  ja julkikoneesta pääsee takaisin:

  ssh -p 52022 localhost

- Usein kätevä **autossh**-ohjelman kanssa käynnistettynä /etc/rc.local -tiedostosta tyyliin

  autossh -M 50023 -f -N -g -R 50022:localhost:22 user@ulkokone

  (-M on autossh:n oma optio, jolla valitaan portti jota se käyttää lähinnä seuraavan (tässä 50024) kanssa yhteyden päälläpysymisen testaamiseen, muut se välittää ssh:lle. Vrt. TCPKeepAlive, ClientAliveCountMax ja ClientAliveInterval, ServerAliveCountMax, ServerAliveInterval)

ssh-agent

- Pitää (dekryptattuja) autentikointiavaimia muistissa session ajan – turvallinen tapa välttää salasanojen jatkuvaa syöttämistä
- Kaksi käyttötapaa:
  - Käynnistetään tietty komento tyyliin

    ssh-agent bash
  - Välitetään agentin tiedot ympäristömuuttujissa shellille:

    eval $(ssh-agent -s)
- Salasanan tallennus: ssh-addkey (voi olla automatisoitu)
- GUIt usein hoitavat automaattisesti session käynnistyessä
- GnuPG:n gpg-agent toimii myös ssh-agentin roolissa

sshd_config vs. ssh_config

- /etc/ssh/sshd_config:
  - Palvelinpään (sshd:n, ssh-demonin) asetukset
  - Globaali, ylläpitäjän hallinnassa, mutta käyttäjäkohtaisiakin asetuksia voi tehdä
  - Määrää mitä on _sallittu_, tärkeimmät tietoturva-asetukset täällä
- /etc/ssh/ssh_config, ~/.ssh/config
  - Asiakaspään (ssh, scp, sftp) asetukset
  - Sekä globaali (/etc/ssh/ssh_config) että käyttäjäkohtainen (~/.ssh/config) tiedosto plus komentorivioptio (-o asetus=arvo)
  - Määrää mitä _halutaan_ tehdä, ts. lähinnä käyttäjän mukavuussäätöjä, ei niinkään tietoturvaa
- Molemmissa sama syntaksi: ”asetus arvo” tai ”asetus=arvo”, kommenttimerkki #, jokerit * ? [...], luettelot pilkulla erotettuna, huutomerkki negaatio, esim. from=”!*.example.com,??.example.net”

sshd_config: ympäristömuuttujat

- AcceptEnv=muuttujanimi[,muuttujanimi...]
  - Mitkä ympäristömuuttujat asiakkaalta otetaan vastaan ja kopioidaan; jokerit sallittuja.
  - Yleensä sallitaan locale-asetukset (LANG, LC_*)
  - Hieman vaarallinen, etenkin jos ympäristöä yritetään muuten rajoittaa
- PermitUserEnvironment={no|yes}
  - Luetaanko ~/.ssh/environment ja environment=... -asetukset ~/.ssh/authorized_keys -tiedostosta. Mahdollistaa joidenkin rajoitusten kiertämisen.
- ForceCommand=komento
  - Suoritetaan aina annettu komento, jyrää clientin antaman komennon (ja mahdollisen ~/.ssh/rc:n); alkuperäinen komento löytyy $SSH_ORIGINAL_COMMAND -muuttujasta

sshd_config: autentikointi

- ChallengeResponseAuthentication={yes|no}
  - Yleinen interaktiviinen autentikointi server-päässä

    KbdInteractiveAuthentication={yes|no}

    KbdInteractiveDevices={bsdauth|pam|skey}
    - Linuxissa käytännössä aina pam, toiminta riippuu sitten pam-konfiguraatiosta, yleensä login/password joko lokaalista passwd:stä tai Kerberokselta tai LDAPista tai...
- PasswordAuthentication={yes|no}
  - Login/password -autentikointi client-päässä (kysyy tunnuksen ja salasanan ja välittää ne palvelimelle)
  - KerberosAuthentication={no|yes}
    - Yleensä helpompaa käyttää Kerberosta PAMin kautta

sshd_config: autentikointi

- UseLogin={no|yes}
  - Käytetäänkö järjestelmän login(1) komentoa interaktiiviseen autentikointiin (nykyisin harvoin)
- UsePAM={no|yes}
  - Sallitaan PAM (Pluggable Authentication Modules) -autentikointijärjestelmä (käytettäväksi joko ChallengeResponseAuthentication tai PasswordAuthentication kanssa)
- GSSAPIAuthentication={yes|no}
  - Natiivi (tikettipohjainen) Kerberos-autentikointi
  - Iso joukko GSSAPI-alkuisia asetuksia

sshd_config: autentikointi

- PubkeyAuthentication={yes|no}
  - AuthorizedKeysFile
    - %h = kotihakemisto, %u = username, %% = %
  - AuthorizedKeysCommand
  - AuthorizedKeysCommandRunAs
    - Jos sallittu avain halutaan hakea tietokannasta tms
- Joukko muitakin, harvemmin tarvittuja autentikointimenetelmiä, rajoituksia ja vaihtoehtoisia asetuksia
- RequiredAuthentications2
  - Jos halutaan vaatia useampaa autentikointia, esim.

    RequiredAuthentications2=pubkey,password

sshd_config: autentikointi

- MaxAuthTries
  - Montako autentikointiyritystä per yhteydenotto
- MaxStartups [=*start:rate:full*]
  - Montako yhtaikaista autentikoimatonta (salasanavastausta odottavaa tms) yhteyttä sallitaan (ennen kuin aikaisemmat ekspiroituvat, vrt. LoginGraceTime); argumenteilla voidaan hylätä yhteydet *start*'in jälkeen lineaarisesti kasvavalla todennäköisyydellä *rate*-100% kunnes *full* saavutetaan
- LoginGraceTime
  - Kauanko odotetaan autentikoinnin onnistumista
- MaxSessions
  - Multiplex-sessiorajoitus, harvoin käytetty (*ei* rajoita käyttäjäkohtaisten sessioiden määrää)

sshd_config: autentikointi

- DenyUsers, AllowUsers, DenyGroups, AllowGroups
  - Jos halutaan rajata pääsy tietyille käyttäjille tai ryhmille; käyttäjä- tai ryhmänimillä (ei numeroilla), jokerit sallittuja, käsittelyjärjestys ylläoleva
- PermitRootLogin={yes|no|without-password|forced-commands-only}

  without-password:
  - Vain public key autentikointi sallittu rootille

  forced-commands-only:
  - Vain public key autentikointi sallittu ja vain pakotetulle komennolle (ForceCommand); esim. voidaan sallia varmuuskopioinnin tarvitsemat komennot muttei muuta
  - Usein rajoitettu myös konekohtaisesti (ks. Match)
- StrictModes={yes|no}
  - Tarkistetaanko kriittisten tiedostojen ja hakemistojen oikeudet (onko kotihakemisto maailman kirjoitettavissa tms)

sshd_config: port forwarding

AllowAgentForwarding={yes|no}

- Sallitaanko ssh-agent autentikoinnin ketjuttaminen

AllowTCPForwarding={yes|no}

- Onko porttien forwardointi ylimalkaan sallittu

GatewayPorts={no|yes|clientspecific}

- Sallitaanko forwardoitujen etäporttien jakaminen muille

PermitOpen {any|*host:port}*

- Jos halutaan rajoittaa forwardointi kone- tai porttikohtaisesti

X11Forwarding={no|yes}

- Sallitaanko X forwarding (-X, -Y)

PermitTunnel={no|yes|point-to-point}

- Sallitaanko tun(4) laitteiden forwardointi

sshd_config: subsystem, chroot

- Subsystem

  ulkoinen järjestelmä tiedonsiirtoon tms, esim. sftp-server (josta myös sisäinen versio internal-sftp)

- ChrootDirectory

  Hakemisto chroot-kutsua varten (autentikoinnin jälkeen). Hakemistonimessä %h = käyttäjän kotihakemisto, %u = username, %% = %. Hakemiston pitää sisältää tarvittavat binäärit ja laitetiedostot kuten chroot yleensäkin. Esimerkiksi haluttaessa rajoittaa käyttö pelkkään tiedonsiirtoon sftp:llä: internal-sftp toimii ilman mitään ylimääräisiä tiedostoja chroot-hakemistossa (toisin kuin chroot yleensä).

sshd_config: ...alive

Näitä voi säätää jos yhteydet pätkivät välissä olevan palomuurin timeouttien tai huonojen yhteyksien tai hyökkäysten takia.

- ClientAliveInterval=*seconds*
  - Jos dataa ei liiku, kuinka pian lähetetään testipaketti. Oletus nolla (ei käytössä).
- ClientAliveCountMax
  - Kuinka monen vastauksettoman paketin jälkeen päätetään yhteys kuolleeksi (vrt. TcpKeepAlive).
- TCPKeepAlive={yes|no}
  - Lähetetäänkö TCP keepalive -paketteja. Oletuksena päällä, saattaa katkaista yhteyden tilapäisen katkon takia. Altis hyökkäyksille.
- ClientAliveCountMax, ClientAliveInterval
  - Montako pakettia saa hukkua ennen yhteyden katkaisua ja viive ennen lähetystä ellei mitään kuulu. Salattu, turvallinen.

sshd_config: sekalaista

- PrintMotd={yes|no}
  - Tulostetaanko /etc/motd (onnistuneen sisäänkirjautumisen jälkeen)
- Banner={none|*file*}
  - Tiedosto joka tulostetaan ennen sisäänkirjautumista
- PrintLastLog={yes|no}
  - Tulostetaanko edellisen kirjautumisen aika
- LogLevel={QUIET|FATAL|...|DEBUG3}
  - Kuinka paljon sshd loggaa tietoja. Oletus on INFO, tarvittaessa voi käyttää VERBOSEa, DEBUG* -asetuksia vain tilapäisesti

sshd_config: Match

Haluttaessa tehdä erilaisia asetuksia eri käyttäjille tai koneille:

Match {User|Group|Host|Address} *malli*

missä *malli* voi sisältää jokereita tavanomaiseen tapaan, tai se voi olla osoitepeite (192.168.0.0/16).

Match-lausetta seuraavat määreet vaikuttavat vain sen ehdon täyttäviin yhteyksiin, joko seuraavaan Match-lauseeseen tai tiedoston loppuun.

Yleinen määre Match-lauseen kanssa on *ForceCommand.*

Samassa Match-lauseessa voi olla useita ehtoja.

sshd_config: Match esimerkki

```
Match User root
  X11Forwarding no
Match Address 192.168.2.0/24
  PermitRootLogin yes
  PasswordAuthentication yes
Match Host omakone.example.net
  TCPForwarding yes
```

ssh_config

- Tuntee useimmat samat autentikointimekanismimääritykset jne kuin sshd_config, mutta täällä ne olennaisesti vain kertovat, mitä halutaan tai ei haluta tehdä – tällä ei voi estää käyttäjää tekemästä mitä haluaa.
- Käsittelyjärjestys (tärkein ensin):
  - Komentorivioptiot (osalle oma optio, aina myös -o ”asetus=arvo”)
  - Käyttäjän oma ~/.ssh/config
  - Globaali /etc/ssh/config
- Kunkin asetuksen ensimmäinen esiintymä voittaa, joten aina spesifimmät (konekohtaiset jne) ensin, yleisemmät myöhemmin

ssh_config: Host, User, Port

- Host: seuraavat määritykset koskevat vain nimettyä konetta (jokerit sallittuja), seuraavaan host-lauseeseen saakka
- Hostname: kohdekoneen oikea nimi
  - Usein kätevä vain lyhyemmän nimen antamiseksi jollekin koneelle
- User: käyttäjätunnus (kohdekoneessa); yleinen yhdessä host-määreen kanssa
- Port: tcp-portti jota käytetään (oletushan on 22)

Esimerkki:

```
host tt1
  hostname tt1.student.it.jyu.fi
  user tt0
  port 52022
```

Nyt **ssh tt1** toimisi kuten **ssh -p 52022 tt0@tt1.student.it.jyu.fi**

ssh_config

- CheckHostIP
  - valitetaanko jos IP ei täsmää known_hosts'in kanssa
- StrictHostKeyChecking={ask|yes|no}
  - Mitä tehdään jos kohdetta ei löydy known_hosts -tiedostosta:

    ask (oletus): kysytään lisätäänkö

    yes: yhteydenotto keskeytetään, ei lisätä

    no: lisätään kone kysymättä
- ConnectionAttempts
  - Montako kertaa yhteyttä yritetään
- ConnectTimeout
  - Kuinka kauan odotetaan yhteyden muodostumista

ssh_config

- ForwardAgent={no|yes}
  - Välitetäänkö ssh-agent -yhteys
- ForwardX11={no|yes}
  - Välitetäänkö X-näyttö (vrt optio -X)
- ForwardX11Trusted={no|yes}
  - Välitetäänkö X-näyttö luotettuna (vrt -Y)
- GatewayPorts
  - Sallitaanko etäyhteydet lokaaleihin forwardoituihin portteihin (-g)
- IdentityFile
  - Käytettävä salainen avain (-i), usein Host'in yhteydessä

ssh_config

- SendEnv var[,var...]
  - Mitkä ympäristömuuttujat välitetään eteenpäin
  - Voidaan käyttää monta kertaa (vaikutus kumuloituu eikä kumoa edellisiä)
  - Toiminta edellyttää, että kohdekoneen sshd_config sallii samat muuttujat
- ServerAliveCountMax
  - Kuinka monta kertaa kohdekoneelle lähetetyt paketit saavat kadota ilman vastausta ennen kuin yhteys päätellään kuolleeksi
- ServerAliveInterval
  - Kuinka pitkän ajan kuluttua lähetetään testipaketti ellei mitään dataa muuten liiku (vrt. ClientAliveCountMax ja ClientAliveInterval sshd_config'issa)
- TCPKeepAlive
  - Lähetelläänko tcp keepalive -paketteja

sshfs: ssh file system

- Asiakaskoneessa: apt-get install sshfs
- Säädä ssh-avaimet niin, että asiakaskoneesta pääsee palvelimeen ssh:lla (ei välttämätöntä)
- Käyttö: sshfs user@kone:remotedir localdir [options]
  - ssh:n optioiden lisäksi paljon omia sekä mount- ja fuse-optioita (”filesystem in user space”), ks. man sshfs, man fuse

    Usein hyödyllisiä: -o sshfs_sync, -o disable_hardlink,

    -o transform_symlinks, -o follow_symlinks, -o allow_other, -o allow_root
- Käytöstä poisto: fusermount -u localdir
  - Ei yleensä kannata jättää päälle pitemmäksi ajaksi kuin on tarpeen
- Ei edellytä root-oikeuksia (eikä yleensä syytä tehdä roottina)

VM:n valvonta

- virsh domstatus *kone* *# vrt. virsh list* |*grep kone*

  case $(virsh domstatus $KONE) in *running*) echo ”$KONE päällä”;; esac
- virsh event kone *event* [--all] [--loop] [--timeout *seconds*] [--list]
  - odottaa kunnes haluttu *event* tapahtuu
  - **virsh event --list** näyttää käytettävissä olevat *event*it
  - **virsh event *kone* --all --loop** näyttää kaikki eventit niiden tapahtuessa
- Sammutetaan VM, odotetaan että se sammuu

  virsh shutdown kone1

  virsh event kone1 lifecycle --timeout 60

  if virsh domstate kone1 | grep -q running ;then

  echo ”kone1 shutdown failed” >&2

  fi
- virsh help monitor