nginx

- ”Engine X”, toiseksi yleisin (apachen jälkeen) www-palvelin Linux-ympäristössä, yleistyy nopeasti
- Ei ihan yhtä ”full-featured” kuin apache, mutta monissa tilanteissa kevyempi (etenkin muistin tarve pienempi, joskus vertailukelpoinen jopa lighttpd:n kanssa)
- Yleinen (tehokas) edustapalvelimena (proxy)

nginx

- Ubuntu tarjoaa useita asennuspaketteja:

  nginx-core

  nginx-light

  nginx-full

  nginx-extras

  - Virtuaalipaketti "nginx" asentaa kulloisenkin oletussetin (tällä hetkellä ~ nginx-core) mutta ei vaihda sitä päivityksissä

nginx konfiguraatiotiedostot

- Konfiguraatiohakemisto /etc/nginx
- Pääkonfiguraatiotiedosto /etc/nginx/nginx.conf
- Lisäkonfiguraatiotiedostoja /etc/nginx/conf.d/*.conf (automaattisia) ja /etc/nginx/snippets/*.conf (otettava käyttöön erikseen)

nginx konfiguraatiotiedostot

- Omia konfiguraatiotiedostoja voi sisällyttää muualtakin include-direktiivillä (vrt. grep include /etc/nginx/nginx.conf)
- Virtual hostit (www-) hakemistossa /etc/nginx/sites-available, käytössä olevat /etc/nginx/sites-enabled (linkkejä edelliseen)

nginx konfiguraatiotiedostot

- Oletus /etc/nginx/sites-available/default, sijainti:

  grep root /etc/nginx/sites-available/default

  - /var/www/html, /usr/share/nginx/html tms

  grep index /etc/nginx/sites-available/default

  - Jos useita vaihtoehtoja, ensimmäistä löytynyttä käytetään
  - Yleensä ainakin index.html toimii
- Jos default-konfiguraatiota ei käytetä:

  rm /etc/nginx/sites-enabled/default

nginx konfiguraatiotiedostot

- Oletus /etc/nginx/sites-available/default, sijainti:

  grep root /etc/nginx/sites-available/default

  - /var/www/html, /usr/share/nginx/html tms

  grep index /etc/nginx/sites-available/default

  - Jos useita vaihtoehtoja, ensimmäistä löytynyttä käytetään
  - Yleensä ainakin index.html toimii
- Jos default-konfiguraatiota ei käytetä:

  rm /etc/nginx/sites-enabled/default

nginx: virtual host, esimerkki

/etc/nginx/sites-available/tt2.student.it.jyu.fi:

```
server {
  listen 80; # Ipv4; voisi olla myös esim. 8080 tai 443 (https)
  listen [::]:80; # IPv6
  server_name tt2.student.it.jyu.fi;
  root /var/www/tt2.student.it.jyu.fi/html;
  index index.html;
  location / {
      try_files $uri $uri/ =404;
   }
}
```

nginx: virtual host, esimerkki

ln -s ../sites-available/tt2.student.it.jyu.fi /etc/nginx/sites-enabled

huom. ei erityistä komentoa (vrt. lighty-enable-mod)

mkdir -p /var/www/tt2.student.it.jyu.fi/html

nano /var/www/tt2.student.it.jyu.fi/html/index.html # jotain sopivaa

service nginx reload

tail /var/log/nginx/error.log # jos ei toimi...

nginx konfiguraatiodirektiivit

- Nginx'in konfiguraatiotiedostossa olevat määritykset (direktiivit) voivat olla globaaleja tai jossain aaltosuluilla {} rajatussa kontekstissa: http{} server{} location{} events{}
- Direktiivien loppuun tulee yleensä puolipiste (ei }:n jälkeen)
- Yleisiä globaaleja asetuksia: user (käyttäjätunnus jolla nginx pyörii), worker_processes (montako palvelinprosessia käynnistetään), pid (minne nginx-pääprosessin PID talletetaan); näitä ei yleensä tarvitse muuttaa
- Konfiguraatiotiedostoon voi lukea muita include-direktiivillä, jokerit toimivat

nginx konfiguraatiodirektiivit

- Kaikki http(s)-palvelimiin liittyvät direktiivit (erityisesti myös server{}) ovat http{}:n sisällä; kaikki .../sites-enabled/... tiedostot tulevat automaattisesti sen sisään (toteutettu include-direktiivillä html{}:n sisällä)
- Virtual host -määritykset tehdään server{} -direktiivillä, niiden sisällön ohjaukset sen sisällä location{}'illa
- events{} ohjaa yhteyksien muodostumista (esim. worker_connections: kuinka monta yhtaikaista yhteyttä per palvelinprosessi sallitaan)

nginx: server{}

listen [*osoite*][:*portti*] [*optiot*];

- jos :*portti* puuttuu, oletetaan 80
- jos *osoite* puuttuu, oletus = * = kaikki IPv4-osoitteet
- [::] = kaikki IPv6 -osoitteet tai myös IPv4 optiolla ipv6only=off

nginx: server{}

server_name *nimi* [*nimi...*];

- nimi tai nimet joihin vastataan; ensimmäinen ensisijainen
- jokerit ja regexp'it sallittuja: *.example.com ~^www\..*\.example\.net

root *polku*;

- sisällön juurihakemisto, johon suhteessa URL-polut tulkitaan

nginx: server{}

index *file* [*file...*];

- tiedostonimi jota käytetään URLin viitatessa hakemistoon
- jos useita, käytetään ensimmäistä joka löytyy

error_log *file* | stderr | syslog:server=*...*

- minne virheilmoitukset kirjoitetaan (paljon optioita)

location ...

nginx: location{}

location [= ~ ~* ^~] *URI* {...}

location @*nimi* {...}

- URIsta riippuvia asetuksia (server{}in sisällä); voi olla useita sisäkkäin
- @*nimi* määrittää nimetyn sijainnin käytettäväksi uudelleenohjausviittauksissa
- location{}in sisällä voi olla mm.

    root, index, error_log, proxy_pass ...

nginx: location{}

- URI voi olla

  polun *alku* (esim. /); jos edessä ^~, ohittaa regexpit

  - käsitellään pituusjärjestyksessä, pisin voittaa

  =koko polku (”= /kala/hauki.html”)

  - voittaa kaikki muut; voi olla esim. yhtaikaa sekä ”/” että ”= /”

  ~regexp (”~ \.php$”), ~* case-insensitive

  - regexp'it voittavat polunalkumääritykset ilman ^~ -prefiksiä
  - regexp'it tulkitaan järjestyksessä, ensimmäinen voittaa

nginx: try_files

try_files *file ... URI;*

- Vaihtoehtoja miten tiedosto tulkitaan, ensimmäinen löytynyt voittaa
  - $uri on alkuperäinen polku sellaisenaan
  - =*code* palauttaa virhekoodin

nginx: try_files

- Esim.

```
location / {
  try_files /huoltokatko.html $uri $uri/ $uri/index.html $uri.html =404;
}
location /pictures/ {
  try_files $uri /pictures/oletus.jpg;
}
```

nginx: try_files

```
location / {
  try_files $uri @varakone;
}
location @varakone {
  proxy_pass http://192.168.6.7:8080;
}
```

nginx: php

- sudo apt-get install php-common php-cli php-fpm
- Haluttuun konfiguraatiotiedostoon:

```
location ~ \.php$ {
  fastcgi_split_path_info ^(.+\.php)(/.+)$;
  try_files $fastcgi_script_name =404;
  set $path_info $fastcgi_path_info;
  fastcgi_param PATH_INFO $path_info;
  fastcgi_pass unix:/var/run/php-fpm.sock;
  fastcgi_index index.php;
  include fastcgi.conf;
}
```

- sudo service nginx restart

nginx: rewrite

rewrite *regexp korvaus* [*lippu*]

- Muuntaa URIn toiseksi regexp-säännöllä
- Jos useita, suoritetaan järjestyksessä
- Liput:

  break: lopeta rewrite-sääntöjen suoritus tähän

  last: lopeta tähän ja ala käsitellä muutettua URIa alusta

  - voi johtaa silmukkaan...

  redirect: palauta ”temporary redirect” -koodi

  permanent: palauta ”permanent redirect” -koodi

nginx: rewrite

- Esim.

```
location /download/ {
   rewrite ^(/download/.*)/media/(.*)\..*$ $1/mp3/$2.mp3 break;
   rewrite ^(/download/.*)/audio/(.*)\..*$ $1/mp3/$2.ra break;
   return  403;
}
```

nginx: proxy_pass

proxy_pass *URL*;

- Ohjaa (yleensä location{}in määräämän) URLin toiseen
- Esim.

```
location / {
  proxy_pass http://192.168.122.4:8080/;
}
location /images/ {
  proxy_pass http://192.168.122.5/pics/;
}
```

nginx: proxy_pass

- Yhdistettävissä esim. rewrite-direktiivin kanssa:

```
location /user/ {
    rewrite    /user/([^/]+) /users?name=$1 break;
    proxy_pass http://192.168.7.8/;
}
```

nginx: reverse proxy

- Esimerkki: halutaan erottaa virtual hosteja sisäisesti eri virtuaalikoneisiin.
- Ohjataan kaikki ulkoa tulevat pyynnöt edustapalvelimena toimivalle nginx'ille, jossa on kullekin virtual hostille tämäntapainen määritys:

```
server {
  listen :80;
  server_name: 1.example.org;
  location / { proxy_pass http://kone1:8080/; }
}
```

nginx: reverse proxy

- Virtual hosteja voidaan jakaa eri virtuaalikoneisiin kuten halutaan, jossain voi pyöriä nginx, jossain apache, jossain lighttpd... jopa useita samassa koneessa eri porteissa, edustakoneen huolehtiessa ohjauksesta oikeaan paikkaan.

Skriptien suorituksesta

- Skriptin ensimmäisellä rivillä voidaan määrätä käytettävä tulkkiohjelma (shell) mahdollisine optioineen:

  ```
  #! /bin/bash -e
  ```

  - optioita: -e = keskeytä virheeseen, -x = jäljitä suoritusta
- Jos skriptitiedosto on suoritettava (`chmod +x`), sen voi ajaa suoraan tyyliin `./skripti.sh`, muuten `bash skripti.sh` (tai `bash -ex skripti.sh` jne)
- Shellille voi antaa skriptin komentorivillä tyyliin `sh -c '…'`, hyödyllistä mm. sudo tai ssh:n kanssa rajaamaan mikä shell tulkitsee minkäkin osan komentorivistä

Tilapäiset tiedostot

- Kun skriptissä tarvitaan tilapäistä tiedostoa, sopiva paikka on usein /tmp tai /var/tmp - mutta tiedostonimeä ei yleensä kannata kiinnittää, useammastakin syystä:
  - Jos skriptiä voidaan ajaa monta kertaa samaan aikaan, saman tiedostonimen käyttö aiheuttaa ongelmia
  - Jos aiottu nimi on jo jonkin muun käytössä, sen käyttö ei yleensä onnistu
  - Kiinteä tiedostonimi saattaa muodostaa tietoturvaongelman (hyökkäysvektori mustahatuille)
- Yksi keino on prosessin id, ympäristömuuttujassa $$:

  MYFILE=/tmp/.jotain$$

mktemp

- Yleinen ratkaisu tilapäistiedostotarpeisiin on

  mktemp [-du] [-p DIR] *template*

  template on nimimalli, jossa pitää olla vähintään 3 X:ää (oletus tmp.XXXXXXXXXX)

  -p (--tmpdir) hakemisto jonka alle tiedosto luodaan

  -d (--directory) luo hakemisto tiedoston asemesta

  -u (--dry-run) älä luo tiedostoa, palauta vain nimi
- Vastaavat kirjastokutsut mktemp(3), mkstemp(3) ja mkdtemp(3)
- Luotu tiedosto pitää erikseen hävittää kun sitä ei enää tarvita.

trap

- Skriptissä voi varautua sen keskeytymiseen virheeseen:

  trap '*komento*' *signal[s]*

  Signaalina 0 tarkoittaa skriptin loppumista mistä tahansa syystä, ja se on hyödyllinen erityisesti "loppusiivoukseen".

  Esim.

  ```
  MYTEMPFILE=$(mktemp /tmp/.jotainXXX)
  trap "rm $MYTEMPFILE" 0
  ```

  (Huom. lainausmerkkien ' ' ja " " ero)

Palveluiden käynnistyksestä jne

- Historiallisista syistä palveluiden käynnistämiseen, pysäyttämiseen jne on useita erilaisia tapoja, ikäjärjestyksessä:

```
/etc/init.d/palvelu start        # sysV
service palvelu start            # upstart & sysV
systemctl start palvelu          # systemd
```

Monille palveluille toimivat kaikki vaihtoehdot.

Palveluiden käynnistyksestä jne

- Eri palveluille käytettävissä olevat toiminnot vaihtelevat, aina kuitenkin vähintään start ja stop, yleensä myös restart, reload ja status
- Kaikkien (...) palveluiden tilaa voi tarkastella komennoilla

```
service --status-all            # upstart & sysV
systemctl -t service status # systemd
```

Palveluiden käynnistyksestä jne

- Käynnistysskriptejä ja -määrityksiä voi tutkia:

  sysV: /etc/init.d/

  upstart: /etc/init/

  systemd: /etc/systemd/system/
- Vrt myös /etc/default/ - käynnistysskriptit lukevat sitä
- Jos palvelu ei käynnisty, voi olla hyödyllistä yrittää käynnistää se käsin katsomalla ensin mitä käynnistysskripti tekee (joskus sh -x … toimii)

Esimerkki: levytilaongelma

- Tilanne: /usr on täynnä, /home'ssa tilaa vaikka kuinka, ei LVM:ää tai ei vapaata levytilaa VG:ssä.
- Ongelma: /home'n pienentäminen on aina vaikeaa, ei juuri koskaan onnistu ilman boottia ja voi olla vaikeaa bootatenkin (esim. jos /home on xfs, jota ei voi pienentää) ja /usr:n laajentaminen on helppoa vain jos käytössä on LVM ja volume groupissa on tilaa.

Esimerkki: levytilaongelma

Ratkaisu: siirretään osa /usr:stä, esimerkiksi /usr/src, /home'n alle ja linkitetään se takaisin:

```
mkdir /home/usr
mv /usr/src /home/usr
ln -s /home/usr/src /usr
```

Helppoa ja nopeaa, mutta ylläpidon kannalta jatkossa hankalaa ja virhealtista - paras kohdella tilapäisratkaisuna ja siirtää /usr/src takaisin tai omaksi tiedostojärjestelmäkseen jos ja kun levyn lisäys myöhemmin sen sallii.

Esimerkki: levytilaongelma

Siirrettävän alihakemiston valinnassa syytä olla varovainen, erityisesti pitää varoa käytössä olevia ja bootissa tarpeellisia hakemistoja. Hyviä vaihtoehtoja /usr:n alla ovat /usr/src, /usr/local ja /usr/share/doc, mahdollisesti jopa koko /usr/share; /var:n alta voi bootin puolesta yleensä siirtää melkein mitä vain, mutta se yleensä edellyttää ao. alihakemistoa käyttävien palveluiden sammuttamista (/var/run ei pidä siirtää). Ennen siirtoa pitää tietysti aina katsoa paljonko tilaa siirrettävän hakemiston alla on (tyhjää hakemistoa ei kannata siirtää).

Esimerkki: levytilaa kateissa

- Jos levyn/osion/taltion mounttaa hakemistoon, joka ei ole tyhjä, alle jäävät tiedostot jäävät näkymättömiin mutta vievät yhä levytilaa (ja ilmestyvät taas näkyviin umountin jälkeen)
- Esim. tehdään /tmp:stä erillinen taltio (LV):

  lvcreate -n lvtmp -L500M tt1-vg

  mkfs -t ext2 /dev/tt1-vg/lvtmp

  mount /dev/tt1-vg/lvtmp /tmp # vanha sisältö jää piiloon

Esimerkki: Kerberos ei toimi

- Jos Kerberos-autentikointi (pam) ei toimi, yleisimmät syyt ovat
  - /etc/krb5.conf virheellinen (väärä realm tms)
  - palomuuri blokkaa portin 88
  - kello on pielessä (Onko ntpd/chronyd asennettuna? Blokkaako palomuuri sen?)
  - käyttäjätunnus puuttuu /etc/passwd:stä (tai Kerberoksesta tai LDAPista tai NISistä tms)

Esimerkki: Kerberos ei toimi

- Omia (tai muiden ylläpitäjien) tumpelointeja epäillessä voi tutkia mitä on tehty:

  zgrep sudo /var/log/authlog*

  history # komentohistoriaa voi selata myös nuolinäppäimillä

  cat /root/.bash_history /home/tunnus/.bash_history

Esimerkki: Kerberos ei toimi

- Autentikointiongelmista tulee yleensä virheilmoitus /var/log/auth.log'iin. Kerran sieltä löytyi tällainen viesti:

  sshd[1472]: pam_krb5(sshd:auth): (user tt)
  mkstemp("/tmp/krb5cc_pam_o2YOkN") failed: No such file or directory

  Jos mkstemp valittaa ”No such file or directory”, se tarkoittaa että hakemistoa, jonka alle se yrittää tilapäistä tiedostoa luoda, ei ole - ja tässä tapauksessa todellakin /tmp oli kadonnut. Syyksi osoittautui se yleisin eli ”operator error”, samasta lokitiedostosta löytyi myös tällainen:

  xx3 sudo: xx0 : TTY=pts/0 ; PWD=/ ; USER=root ; COMMAND=/bin/rm -r -f /tmp/

  (name changed to protect the guilty)

  ja korjaukseksi riitti:

  sudo mkdir /tmp; sudo chmod 1777 /tmp # tai chmod o=rwxt,ug=rwx /tmp