Quo vadis FreeBSD?

FUUG ry.
Jukka Ukkonen

Mitä on tarjolla nyt?
Mihin ollaan menossa?

FreeBSD:n versiot

- 4.x – 4.11 stabiili ei aktiivisesti kehitettävä linja
- 5.0 – 5.2 testi- ja kehitysversioita
- 5.3 ensimmäinen stabiili 5.x-sarjassa
- 5.4 erittäin stabiili, mutta edelleen joitakin aiempien 5.x-versioiden aikana syntyneitä puutteita on havaittavissa.
- 6.0 (julkaistu 2005-11-06)
  - 802.11-tuki paranee,
  - hienojakoinen lukitus VFS:ssä,
  - hienojakoinen SMP (lähes 100% giant-lock pois)
  - tavallaan performance-julkaisu,
  - paikkaa vähitellen 5-sarjaan jääneitä puutteita.

Pitkä tähtäin
eli missio

- Avoimuus
- Standardit
- Yhteensopivuus
- Turvallisuus
- Tehokkuus
- Stabiilius
- Käytettävyys

Keskipitkä tähtäin

- Edelleen kehitetty SMP
  - Hienojakoiset lukitukset kaikkialle koodiin
  - Scheduler vaihdetaan ULE:en
  - KSE (vrt. SA)
  - kqueue() & kevent()
  - Vanha libc_r (N-to-1) thread-kirjasto vaihdettu uuteen libthr:ään (N-to-M)
- Radikaalisti laajennettu 802.11
  - Liki kaikki, mitä kukaan on WLAN:iin ehdottanut.
- UFS2 (ACL, MAC, gen.attr.)
- Yleisesti ottaen merkittävästi lisää “vääntöä” ja toiminnallisuutta.

Käyttäjien toiveet

- Karkeasti yleistäen: “Kaikki mulle heti nyt!”
- Yhtä sekalaiset kuin käyttäjä- ja kehittäjä-kunta itsekin.
- 802.11
- N-to-M -threading
- Turvallisuus
- Suorituskyky
- Laajennettu XPG4-yhteensopivuus.
- Käytön helppous
- Enemmän juuri FreeBSD:lle käännettyä sof-taa ilman sovituskerroksia.

Mitä siis tehdään?
Lyhyt tähtäin ja nykyisyys (1)

- ULE on liki käyttökunnossa.
  - Parempi adaptiivisuus kuormitustilanteisiin timeshare-prosessien kesken.
  - RT- ja IDLE-scheduler eivät yritä adaptoitua.
  - RT-scheduler: 2 metodia (FIFO & time slice).
- UFS2 levyformaattina aktiivikäytössä 5.x:ssä
  - MacOS:n tapaan tiedostolla on kaksi haaraa: sisältö ja attribuutit (sis. mac, acl, gen.attr.)
  - Oletuksena aina SoftDep!
    => Ei koskaan rikki.
    => Ei fsck:ta boot:issa.
  - Background fsck ja dump käyttävät eräänlaista snapshot-metodia.

Mitä siis tehdään? Lyhyt tähtäin ja nykyisyys (2)

- KSE ja sen varaan rakennettu N-to-M thread-kirjasto ovat oletuksena 5.4:ssä.
- SMP lähes täysin irti giant-lock-virityksistä.
  - Hienojakoinen kriittisten alueiden lukitus lähes kaikessa. => N prosessoria tuottaa N kertaa yhden prosessorin tehon.
  - Vain jotkut hyvin vanhat ja vähän käytetyt driver:it syyllistyvät giant-lock-syntiin.
- 802.11-tuki on 5.4:ssä jo melko hyvä, mutta 6.0:ssa entistäkin parempi.
- Project EVIL = kuorrutus MS-drivereille
- Laajennettu XPG4-tuki

Mitä siis tehdään? Lyhyt tähtäin ja nykyisyys (3)

- 5.x:n oletuspalomuuri on aina ollut IPFW2.
  - helpompi konfiguroitava kuin vanha IPFW1.
  - IPFW2:ssa mukana ns. stateful inspection eli se voi ylläpitää omaa kopiota yhteyden tilasta.
  - IPFW2:n saa käyttöön myös 4.11:n kanssa.
  - IPFW2:een on kehitteillä fuusio toistaiseksi erillisen IP6FW:n toiminnallisuuden kanssa.
- 5.4:ssä /dev on aina devfs-mount, minkä alle kernel luo itse tarvittavat laitetiedostot.
- 6.0:ssä driver-puolelle paljon parannuksia: esim. laajennettu ehci ja monet MPSAFE.

Mitä siis tehdään? Lyhyt tähtäin ja nykyisyys (4)

- USB-tuki on jo 5.4:ssä parempi, ja 4.11:n usbd on vaihdettu devd:hen.
  - Enemmän tuettuja laitteita
  - Parempi tuki erikoisille ja omituisille tavoille käyttää USB:tä.
- 5.4:n myötä XFree86 on täysin hyllytetty ja tilalle on otettu Xorg-6.8.2.
  - Suuri joukko X:n kehittäjiä on siirtynyt Xorg:n riveihin, ja XFree86:n tuki on heikentynyt.
  - Xorg:lla on parempi tuki eri grafikkakorteille.
  - Xorg:lla on parempi DRI-/DRM-tuki.
  - Xorg:n saa myös 4.11:n kylkeen.

Mitä siis tehdään? Lyhyt tähtäin ja nykyisyys (5)

- IPSEC on saanut kylkeensä uuden IKE-demonin nimeltä racoon.
  - Käytön helpottuminen
  - Joustavuus
- 5.4:stä alkaen käytettävissä CARP
  - Common Address Redundancy Protocol
  - Toipuminen laitteistohäiriöistä siirtämällä “yhteisiä” osoitteita paikasta toiseen.
  - HA-cluster-tekniikka
- Kaikki levyt ovat 5.4:ssä geom-laitteita
  - Levyn ja file-systeemin väliin voi pinota erilaisia moduleja gmirror, gconcat, gstripe, gbde, jne.

Mitä siis tehdään? Lyhyt tähtäin ja nykyisyys (6)

- Järjestelmän yleisen laadun parantamiseksi käyttöön on otettu erillinen “kernel test bed”.
  - 5.x -sarjan alkuvaiheissa suuri määrä samaan aikaan muuttuvia asioita aiheutti kohtuuttomasti epästabiiliutta, mitä BSD-maailmassa ei ole totuttu sietämään.
  - Tilanteen korjaamiseksi tarvittiin uusi työkalu, joka systematisoi testausta ja nostaa esiin asioita, mitä manuaalisessa tarkastelussa jäisi muuten helposti huomaamatta.
  - Varsinkin 6.0:sta alkaen uuden testausmallin vaikutukset pitäisi näkyä.

Panokset, tulokset ja roadkill

- 5.x-versioiden myötä FreeBSD on saanut huomattavan määrän uusia piirteitä ja monia vanhoja piirteitä on paranneltu merkittävästi.
- Voimakas muutos tuottaa usein väistämättä ei-toivottuja lieveilmiöitä.
- Enimmäkseen kaikki muutokset kuitenkin ovat tuottaneet tulosta.
- Todella häiritseviä vikoja on ilmestynyt hämmästyttävän vähän.

Mitä on mennyt kiville?

- Vanha vinum ei enää käynnisty bootissa.
- Uusi geom-pohjainen vinum ei osaa vielä kaikkia samoja asioita, mitä vanha.
  - Kaikki vinum-pohjaiset peililevyt ja stripe-volumet on pakko rakentaa alusta asti uudestaan erillisten geom-modulien varaan. (gmirror, etc.)
- Vanha unionfs on seonnut jossain vaiheessa.
  - Kerrostetun mount:in tekeminen ei onnistu.
  - Nyt ei siis saa esim. turvallista read-only-kerrosta ja päälle vain muutokset keräilevää läpinäkyvää kerrosta.

Mitä on voitettu?
Mitä on heti kulman takana?

- Käytännössä kaikilla FreeBSD:n mission rintamilla on edetty merkittävästi, joskin vaikeuksien kautta.
  - Stabiilius kärsi kehitysversioissa (5.0-5.2).
  - Nyt 5.4:stä alkaen stabiiliuskin on taas totutulla BSD-tasolla, eli ei heilu, ei horju.
- Kun 6.0:ssa 802.11-tuki laajenee (WPA, etc.), voisi kuvitella FreeBSD:n mielenkiintoiseksi esim. PDA-laitteisiin.
- 6.0:n release notes kannattaa selata läpi.

Mihin FreeBSD on oikeasti menossa?

- Mission mukaan edetään.
- FreeBSD:n on tarkoitus olla sopiva laajaan spektriin erilaisia laitteita:
  - suurista rinnakkaisjärjestelmistä
  - pieniin sulautettuihin järjestelmiin ja
  - desktop- ja laptop-käyttöön siinä välissä.
  - SMP erittäin keskeisessä roolissa.
- Kaikkein vaikeimmat rastit ovat:
  - pienikokoiset sulautetut järjestelmät ja
  - tavallisten toimisto-softan käyttäjien helpokäyttöiset desktop-ympäristöt.
- Esim. FreeSBIE:lta hyvää tukea desktop:iin.

FreeBSD 4:10

Thou shalt not commit bikesheds

Kysymyksiä 0 (a)

- Miten päivitetään FreeBSD release:sta toiseen seuraamatta mitään stable-haaraa tms?
  - Oma tapani on suoraan source:sta...

```
cvsup -g sys-cvsup
cd /usr/src
make buildworld
make KERNCONF=Mjolnir buildkernel
make KERNCONF=Mjolnir installkernel
shutdown -r now
...
root-login -rituaalit...
```

  - *jatkuu...*

Kysymyksiä 0 (b)

```
cd /usr/src
mount -u /tmp          (jos /tmp on noexec)
mergemaster -p
make installworld
mergemaster
shutdown -r now
...
```

- Tarpeen mukaan voi make-muuttujia tunkea tiedostoon `/etc/make.conf`.
- Jos paikallisia virityksiä ei ole tehty, sysinstall:in tarjoama binääri-päivityskin voinee toimia. Itse en ole koskaan edes yrittänyt, koska minulla ei ole koskaan aivan “vanilla”-asennus. ;-)

Kysymyksiä 0 (c)

- Kernelin ja perus-user-space:n päivityksestä seuraa usein tarve erilaisten lisäohjelmistojen päivitykseen.
  - portupgrade -NRPC softa-xyz
  - Lisää optio -r, jos haluat päivittää riippuvuudet myös ylöspäin.
  - Jos cvsup ja/tai portupgrade ei ole asennettu

    cd /usr/ports/net/cvsup ; make install (jos tarpeen)
    cvsup -g ports-cvsup
    cd /usr/ports/sysutils/portupgrade
    make install
    portupgrade -NRPC cvsup (ei vara venettä ...)
- gnome2 kannattaa tarvittaessa hävittää ja asentaa uudestaan.

Kysymyksiä 0 (d)

- Edellä esimerkeissä mainittu sys-cvsup on suunnilleen:

```
######
#       cvsup -g -L 2 /path/to/this/file
######
*default host=cvsup.dk.freebsd.org
*default base=/usr
*default prefix=/usr
*default umask=022
*default release=cvs tag=RELENG_5
*default delete use-rel-suffix
src-all
```

Kysymyksiä 0 (e)

- Edellä esimerkeissä mainittu ports-cvsup on suunnilleen:

```
######
#       cvsup -g -L 2 /path/to/this/file
######
*default host=cvsup.dk.freebsd.org
*default base=/usr
*default prefix=/usr
*default umask=022
*default release=cvs tag=.
*default delete use-rel-suffix
ports-all
```

Kysymyksiä 1

- Miten FreeBSD omasta mielestään profiloituu käyttismarkkinoilla?
  - Itse FreeBSD pyrkii olemaan monikasvoinen.
    - Freesbie, DragonFly ja pcbsd painottavat erityisesti tavallista loppukäyttäjän laptop- tai desktop-ympäristöä.
- Missä ja kenen kannattaa käyttä FreeBSD:tä ja milloin taas ei?
  - Ks. ed.
- Onko rintamalinjoissa tapahtumassa muutosta?
  - Noviisit ja desktop on koko ajan listalla.
    - Mm. sysinstall vaihtunee FreeBSD:ssäkin bsdinstall:iin.

Kysymyksiä 2

- Onko ports:in tilalle tulossa jokin modernimpi (binääri-) paketointimuoto?
  - Itse ports ei ole varsinainen ongelma.
  - FreeBSD:n pkg-formaatti ei sisällä käsitteitä sovellusluokka ja/tai versiojoukko, vain sovellus ja versio.
  - Oikea työkalu päivityksiin on portupgrade
    - `portupgrade -NRCP xyzzy` päivittää xyzzy:n binääripaketista, jos löytyy, muuten source:sta.
  - `man portupgrade` auttaa.
  - xfree86-dri:n päivitys Xorg:n dri:hin

```
pkgdb -Fu
portupgrade -fo graphics/dri graphics/xfree86-dri
```

Kysymyksiä 3

- Akuutti aihe “*trusted computing*” ja/tai “*fare-share computing*”:
- salattu tiedostojärjestelmä?
  - gdbe : kryptattu block device, ihan yksi lysti, mihin sitä käytetään, UFS2, swap, NTFS, ...
- role based access control?
  - ACL, capabilities & MAC sekä vanhaan tapaan roolikohtaiset ryhmät. (MAC-label ~= lupa poikkeukselliseen kohteluun)
- Miten root:in oikeuksia voi rajata?
  - Split capabilities!
  - `sysctl security.bsd.suser_enabled=0`

Kysymyksiä 4 (a)

- Miten rajoittaa prosessin ajoympäristöä (chroot, jail)?
  - chroot() ei paljon rajoita, tarkoitettu testeihin
  - jail() ei ihan yhtä geneerinen kuin Solaris-10:n container, mutta sama perusidea
  - MAC-label: perus-policy-tarjonnasta löytyy mm. lomac, biba, mls. Lisäksi: portacl, bsdextended (fs firewall) + ugidfw, partition (prosessit eivät näe partition ulkopuolisia prosesseja), ifoff (sallii tai estää liikenteen tietyn verkkotöpselin yli)
  - *Jatkuu...*

Kysymyksiä 4 (b)

- Koko sysctl:n security-haara :

```
security.jail.set_hostname_allowed: 1
security.jail.socket_unixiproute_only: 1
security.jail.sysvipc_allowed: 0
security.jail.getfsstatroot_only: 1
security.jail.allow_raw_sockets: 0
security.jail.chflags_allowed: 0
security.jail.jailed: 0
security.bsd.suser_enabled: 1
security.bsd.see_other_uids: 0
security.bsd.see_other_gids: 0
security.bsd.conservative_signals: 1
security.bsd.unprivileged_proc_debug: 1
security.bsd.unprivileged_read_msgbuf: 1
security.bsd.hardlink_check_uid: 0
security.bsd.hardlink_check_gid: 0
security.bsd.unprivileged_get_quota: 0
```

Kysymyksiä 4 (c)

- Ihan ilman MAC-partition -viritystäkin käyttäjän voi rajata näkemään vain omat prosessinsa :

```
security.bsd.see_other_uids: 0
security.bsd.see_other_gids: 0

> ps -ax
  PID  TT  STAT      TIME COMMAND
33483  ??  S      0:00.01 sshd: jau@ttyp0 (sshd)
33484  p0  Ss     0:00.03 -tcsh (tcsh)
33491  p0  R+     0:00.00 ps -ax
```

Kysymyksiä 5

- Miten rajoittaa prosessin resurssikulutusta (max 10% cpu:sta, max 10 s cpu / 60 s, i/o-kuristus)
  - Verkko-I/O:ssa dummynet (IPFW:n jatke) tarjoaa kuristusta, viiveitä, yms.
  - Vanhastaankin on ollut eri scheduler-policy:t RT (PE/NPE), TS ja IDLE sekä nice-arvot time-share-prosessien säätämiseen. ULE parantaa TS:n automaatiota ja reiluutta.
  - Varsinaisia abs. limiittejä vain resource-limit-arvot.
  - MAC-partition voisi ehkä laajentua fare-share-rajoituksiin.