PROJECT: IRIS

(Productverslag)

Projectcode:	IRIS
Datum voltooid:	07-06-2009
Auteur:	Tim Baas
Versie:	1.0
Bestandsnaam:	ProductVerslag.doc

Documenthistorie

Revisies

Versie	Status	Datum	Wijzigingen
1.0	Final		Geen wijzigingen

Inhoudsopgave

1 INTRODUCTIE 5

1.1 SCOPE VAN HET DOCUMENT. 5

2 NODE SOFTWARE 6

2.1 INTRODUCTIE 6
2.2 NANOBSD 6
2.2.1. Introductie. 6
2.2.2. Ontwerp 6
2.2.3. Systeemomgeving. 7
2.2.4. Configuratie. 9
2.2.5. Installatie. 12
2.2.6. Ports & packages. 13
2.2.7. Diskless-configuratie 14

3 NODE HARDWARE 16

3.1 INTRODUCTIE 16
3.1.1. Testmethode 16
3.2.1. Testomgeving 16
3.2 SOEKRIS NET4801 16
3.2.1. Systeemspecificaties 17
3.2.2. Prestaties over ethernet, bridged 18
3.2.3. Prestatiesr over ethernet, RIP 19
3.2.4. Prestaties over 802.11a/b/g, bridged 20
3.2.5. Prestaties over 802.11a/b/g, RIP 21
3.3 PCENGINES ALIX2D3 23
3.3.1. Systeemspecificaties 23
3.3.2. Prestaties over ethernet, bridged 24
3.3.3. Prestaties over ethernet, RIP 25
3.3.4. Prestaties over 802.11a/b/g, bridged 26
3.3.5. Prestaties over 802.11a/b/g, RIP 28
3.4 UBIQUITI NANOSTATION 5 29
3.4.1. Systeemspecificaties 29
3.4.2 Prestaties over 802.11a / 802.11Ta 30
3.4.2. NanoStation 5 vs Wandy 5XL 31
3.5 IRIS-OPSTELLINGEN 31
3.5.1. NanoStations met Alix2D3 31
3.5.2. NanoStation met Net4801 33

4 NODEFABRIEK 36

4.1 INTRODUCTIE 36
4.1.1. Nodefabriek directory-overzicht 37
4.2 INSTALLATIE NODEFABRIEK 38
4.2.1. Installatie van het hostsysteem. 38
4.2.2. Configuratie van het hostsysteem. 39
4.2.3. Installatie van de jailomgeving. 40
4.3 CONFIGURATIE NODEFABRIEK 42
4.3.1. Configuratie van de ports-jailomgeving 42
4.3.2. Configuratie NFS-root. 44
4.3.3. Configuratie, NFS-server. 45
4.3.4. Configuratie DHCP-server. 46
4.3.5. Configuratie van de TFTP-server. 46
4.3.6. Configuratie werkomgeving 46
4.3.7. Configuratie-overzicht. 49
4.4 GEBRUIK NODEFABRIEK 52
4.4.1. NanoBSD configuratie aanmaken. 52

4.4.2. NanoBSD configuratie-template. ... 52
4.4.3. NanoBSD ports & packages toevoegen. ... 52
4.4.4. NanoBSD-configuratiebestanden toevoegen. ... 54
4.4.5. NanoBSD-installatie aanmaken. ... 54
4.4.6. Systeemspecifieke installatie. ... 57

5 NODE-INSTALLATIE ... 59

5.1 INTRODUCTIE ... 59
5.2 NODE-INSTALLATIE, UITVOERING ... 59
5.2.1. Node-installatie, stroomschema. ... 59
5.2.2. Nieuwe installatie ... 61
5.2.3. Bestaande image. ... 62
5.2.4. PXE installatie. ... 63
5.3 NODE-INSTALLATIE, BESCHRIJVING ... 64
5.3.1. Boot configuratie. ... 64
5.3.2. rc.conf configuratie. ... 65
5.3.3. Apache22, webserver. ... 65
5.3.4. Named, Domain Name Server. ... 66
5.3.5. Lvrouted, dynamische routing. ... 67
5.3.6. Pen, loadbalancing. ... 67
5.3.7. NTPD, tijdsynchronisatie. ... 68
5.3.8. NET-SNMP, systeem-monitoring. ... 69
5.3.9. Syslogd, systeemlogging. ... 71
5.4 NODE-INSTALLATIE, BEHEER ... 72
5.4.1. Remote updates. ... 72

APPENDIX A , NANOBSD CONFIGURATIE-TEMPLATE ... 74

APPENDIX B , PORT-COMPILATIE-SCRIPT ... 79

APPENDIX C , NANOBSD-CONFIGURATIEBESTANDEN ... 81

APPENDIX D , UPDATE-IMAGE-SCRIPT ... 83

APPENDIX E , LVROUTED-RC-SCRIPT ... 85

APPENDIX F , PEN-SORTPROXIES-SCRIPT ... 87

APPENDIX G , MANAGE-INTERFACES-SCRIPT ... 89

1 Introductie

1.1 Scope van het document.

Dit document beschrijft de producten die zijn opgeleverd tijdens de aftudeeperiode van Tim Baas, student informatica aan de Hogeschool van Leiden. Het bevat de resultaten die zijn verkregen bij het uitvoeren het IRIS-project.

2 Node software

2.1 Introductie

In dit hoofdstuk worden de softwareoplossingen besproken die zijn gebruikt binnen het IRIS project.

2.2 NanoBSD

2.2.1. Introductie.

NanoBSD is een toepassing binnen FreeBSD, ontwikkeld door Poul Henning-Kamp, die gebruikt wordt voor het creëren van FreeBSD-systeeminstallaties. Deze installaties zijn bij uitstek geschikt voor een embedded systeemomgeving.
Met de NanoBSD toepassing kunnen systeemimages geproduceerd worden die geheel naar wens kunnen worden aangepast. Het ontwerp heeft als doel om systeeminstallaties gemakkelijk te kunnen uit rollen en beheren. Omdat het als toepassing voornamelijk gericht is op “computer appliances”, kan software gemakkelijk gebundeld worden en werkt het bij installatie zoveel mogelijk “out of the box”

Een NanoBSD-installatie wordt gecompileerd vanuit de FreeBSD-broncode. Ports and Packages die binnen FreeBSD kunnen worden geïnstalleerd zijn op dezelfde manier te gebruiken binnen een NanoBSD installatie. Operationeel draait het systeem “read-only” en er hoeven geen bestandsysteemcontroles te worden uitgevoerd bij een stroomuitval. Een NanoBSD systeeminstallatie heeft alle functionaliteiten van een standaard FreeBSD installatie tenzij bij de configuratie iets expliciet wordt verwijderd.

Het installeren en configureren van NanoBSD is gemakkelijk uit te voeren via een shell script en een bijbehorend configuratiebestand.

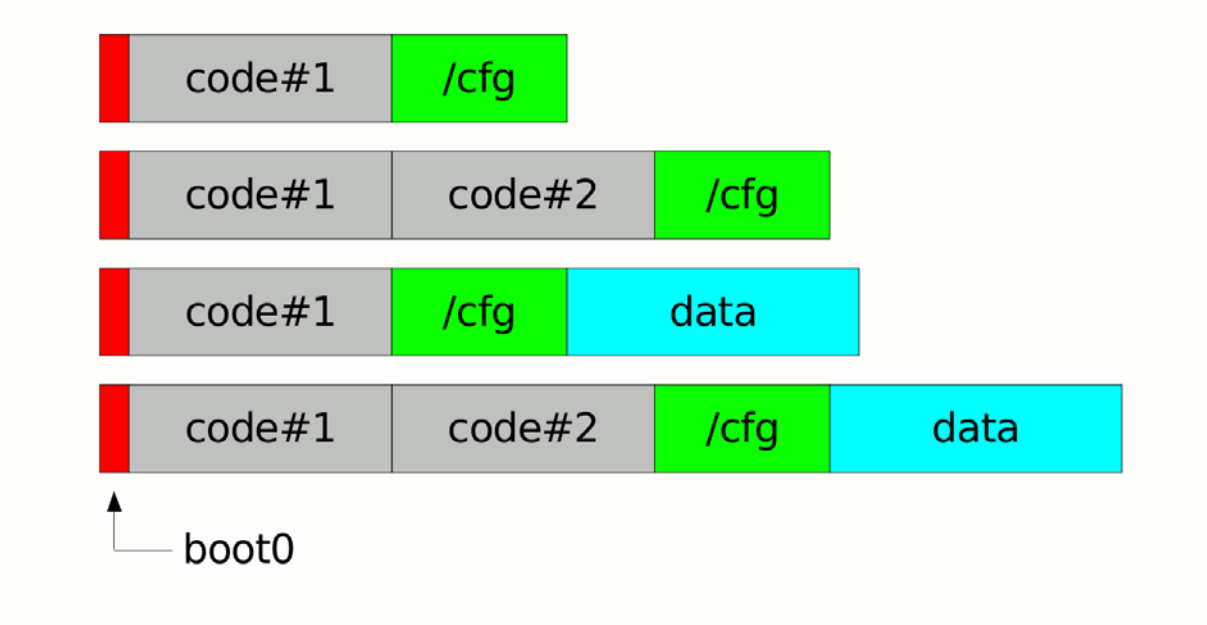
2.2.2. Ontwerp.

Bij de installatie van een NanoBSD-systeemimage wordt het beschikbare flashgeheugen in het geheel onderverdeeld in drie slices. Een standaardconfiguratie heeft de volgende onderdelen.

- Twee systeemslices code#1 en code#2,
- En een configuratie slice /cfg.

De twee systeemslices bevatten een identieke systeeminstallatie die naar keuze kan worden ingeladen bij het opstarten van het systeem.
Deze configuratieopstelling heeft als voordeel dat een van de twee systeemslices gebruikt kan worden als “live” omgeving en de andere, wanneer nodig, als testomgeving. Ook bij het updaten van het systeem geeft dit de mogelijkheid om naar een van de “slices” een compleet nieuwe installatie te schrijven zonder de operationele installatie te beïnvloeden.
De configuratieslice dient om systeeminstellingen en bestanden uit de /etc directory op te slaan, waarbij dus één configuratie voor beide systeeminstallaties wordt gebruikt.
Omdat de systeeminstallatie “read-only” draait, zijn de directories /etc en /var gekoppeld als ramdisk. Bestandswijzigingen binnen deze directories hoeven op deze manier niet direct naar het flashgeheugen geschreven te worden, zodat het aantal schrijfacties hierop beperkt blijft.
Omdat het geheugen van een ramdisk bij spanningsverlies, of een bewuste herstart van het systeem verloren gaat, dient gewijzigde data binnen de /etc directory naar de configuratieslice gekopieerd te worden om deze permanent op te slaan.
Bij het opnieuw opstarten van een systeem wordt de configuratie slice kort “read-only” gekoppeld om opgeslagen instellingen terug naar de /etc-ramdisk te kopiëren.

Bij operationeel gebruik dient de configuratieslice alleen gekoppeld te worden wanneer nieuwe of gewijzigde data uit de /etc-directory opgeslagen moet worden.
Het is raadzaam de configuratie slice na gebruik weer te ontkoppelen zodat onnodige schrijfacties naar het flashgeheugen worden voorkomen.



Figuur 1, NanoBSD disk lay-out

Naast de standaard configuratie-indeling is het binnen NanoBSD mogelijk om - indien gewenst - een andere disk lay-out te gebruiken. Wanneer voor een configuratie gekozen wordt met een enkele systeemslice, wordt hiermee het volume van de totale installatie gehalveerd. Dit kan wenselijk zijn wanneer besloten wordt dat geen testomgeving nodig is, of dat een beperkte hoeveelheid flashgeheugen op het doelsysteem beschikbaar is. Een extra slice kan worden aangemaakt voor de opslag van additionele data.

2.2.3. Systeemomgeving.

Om gebruikt te kunnen maken van NanoBSD-functionaliteiten is als eerste een “host systeem” nodig waarop een installatie van FreeBSD 6 of 7 is geïnstalleerd.
Op het hostsysteem dient in ieder geval de FreeBSD broncode beschikbaar te zijn.
Om te controleren of deze is geïnstalleerd kan het volgende uitgevoerd worden:

```
# ls /usr/src
```

Wanneer de output van het list commando ongeveer overeenkomt met het onderstaande directory overzicht is de broncode geïnstalleerd.

```
COPYRIGHT           compat              release
LOCKS               contrib             rescue
MAINTAINERS         crypto              sbin
Makefile            etc                 secure
Makefile.inc1       games               share
ObsoleteFiles.inc   gnu                 sys
README              include             tools
UPDATING            kerberos5           usr.bin
bin                 lib                 usr.sbin
cddl                libexec
```

Gebruik csup om de broncode te downloaden wanneer deze niet beschikbaar is, of een update nodig heeft. De onderstaande template download de volledige source tree van de FreeBSD CVS repository.

```
# vi /etc/csup.conf

      *default host=cvsup5.freebsd.org
      *default base=/var/db
      *default prefix=/usr
      *default release=cvs tag=RELENG_7_1
      *default delete use-rel-suffix
      *default compress
      src-all

# csup -g -L2 /etc/csup.conf
```

Als de FreeBSD broncode op het hostsysteem beschikbaar is, kan gestart worden met het bouwen van een NanoBSD-image vanuit de NanoBSD-configuratiedirectory.

```
# cd /usr/src/tools/tools/nanobsd
# ls

      Files
      FlashDevice.sub
      nanobsd.sh
```

De NanoBSD-configuratiedirectory is standaard uit de bovenstaande onderdelen opgebouwd:

In de "`Files`" directory kunnen bestanden worden ondergebracht die naar de root van het bestandsysteem worden gekopieerd bij het creëren van een NanoBSD-image. Hier kunnen bijvoorbeeld vooraf configuratiebestanden voor de `/etc` directory worden aangemaakt.

Het bestand "`FlashDevice.sub`" bevat geometrie-informatie van een aantal verschillende compactflash-kaarten. Deze informatie wordt door NanoBSD gebruikt bij het aanmaken van een systeemimage maar hoeft normaal gesproken niet exact overeen te komen met de specificaties van de compactflashkaart die voor het doelsysteem gebruikt wordt. Een uitzondering hierop is het ontbreken van LBA, Logical Block Adressing, ondersteuning in de bios van het doelsysteem. In dit geval is het noodzakelijk dat de geometrie-informatie van de gebruikte compactflashkaart expliciet overeenkomt met de configuratie gedefinieerd binnen het NanoBSD-configuratiebestand.

Indien het gewenste type compactflash niet binnen het FlashDevice-bestand staat omschreven, kan de informatie handmatig worden opgezocht en worden toegevoegd aan het bestand.

Het shell script "`nanobsd.sh`" wordt gebruikt om een NanoBSD-installatie daadwerkelijk te compileren. Het bevat alle installatie- en configuratieroutines die standaard worden uitgevoerd indien geen expliciete configuratie-instellingen zijn gedefinieerd.

2.2.4. Configuratie.

Aan een NanoBSD-installatie kunnen op verschillende manieren configuratiewijzigingen worden doorgevoerd, maar de belangrijkste instellingen worden geconfigureerd binnen een “template” dat kan worden aangevoerd bij het uitvoeren van het nanobsd.sh-script. In de template kunnen configuratie-instellingen worden aangemaakt die bij de verschillende stappen van het installatieproces worden gebruikt, om de standaard-instellingen in het nanobsd.sh-script te overschrijven.

Naast het overschrijven van standaardinstellingen is het ook mogelijk zelf shell-scripts te schrijven die gedurende het installatieproces worden uitgevoerd als “custom functions”. De NanoBSD configuratie template kan zelf worden aangemaakt in NanoBSD configuratie directory. Onderstaande lay-out is een voorbeeld en kan op verschillende manieren uitgebreid en of aangepast worden.

```
# Basisopties

NANO_NAME=wleiden
NANO_SRC=/usr/src
NANO_IMAGES=1
NANO_KERNEL=kernel.net4801

# Opties gerelateerd aan de schijfindeling

NANO_CONFSIZE=8192
NANO_DATASIZE=0
#NANO_CODESIZE=
#NANO_RAM_ETCSIZE=
#NANO_RAM_TMPVARSIZE=

# Opties toegepast gedurende build-world stage

CONF_BUILD='
NO_KLDLOAD=YES
NO_NETGRAPH=YES
NO_PAM=YES
'
# Opties toegepast gedurende install-world stage

CONF_INSTALL='
NO_ACPI=YES
NO_BLUETOOTH=YES
NO_CVS=YES
NO_SENDMAIL=YES
NO_CALENDAR=YES
NO_SHARE=YES
'
# Opties toegepast gedurende build & install-world stages

CONF_WORLD='
NO_BIND=YES
NO_MODULES=YES
NO_KERBEROS=YES
NO_GAMES=YES
NO_RESCUE=YES
NO_INFO=YES
'
# Definitie van flashmedia in FlashDevice.sub

FlashDevice kingston 256
```

```
NANO_NAME=wleiden
NANO_SRC=/usr/src
NANO_IMAGES=1
```

Deze bovenste drie regels van de configuratie zijn in ieder geval noodzakelijk om een eigen NanoBSD image te kunnen creëren.

NANO_NAME: dient om de naam van de installatie te definiëren en wordt ook gebruikt om de $NANO_OBJ-variabele te vullen waarmee de NanoBSD-objectdirectory wordt aangemaakt.

NANO_SRC: geeft de locatie van de FreeBSD-broncode aan.

NANO_IMAGES: bepaalt het aantal systeeminstallaties dat wordt aangemaakt en indirect ook de schijfindeling van de totale installatie. In het voorbeeld wordt dus slechts een systeemslice (code#1) aangemaakt.

NANO_KERNEL: geeft de naam van het kernel-configuratiebestand aan dat tijdens de "build kernel" stap van het installatieproces wordt gebruikt. De kernel-configuratiebestanden dienen opgeslagen te worden in "/usr/src/sys/i386/conf". Indien de NANO_KERNEL optie niet is geconfigureerd wordt automatisch een generieke kernel gebruikt.

```
NANO_CONFSIZE=
NANO_DATASIZE=
NANO_CODESIZE=
NANO_RAM_ETCSIZE=
NANO_RAM_TMPVARSIZE=
```

De bovenstaande opties bepalen het volume dat bij de configuratie van de verschillende schijven wordt gebruikt. De opties zijn optioneel en indien ze niet worden opgenomen in het NanoBSD-configuratiebestand, wordt automatisch de standaardindeling gebruikt. Waardes zijn genoteerd volgens de Unix standaard, 512 bytes/Block.

NANO_CONFSIZE: bepaalt het volume van de configuratieslice, de standaardwaarde is 2048.

NANO_DATASIZE: bepaalt het volume van de dataslice. Indien de waarde is ingesteld op "0" wordt geen dataslice aangemaakt.

NANO_CODESIZE: bepaalt het volume van de systeemomgeving. Indien de variabele niet is gebruikt, wordt automatisch de maximaal beschikbare ruimte op het flashgeheugen toegewezen.

NANO_RAM_ETCSIZE: bepaalt het volume van de /etc ramdisk, de standaardwaarde is 10240.

NANO_RAM_TMPVARSIZE: bepaalt het volume van de /var ramdisk, de standaardwaarde is 10240.

De overige opties binnen de configuratie-template kunnen worden onderverdeeld in verschillende installatiestappen. De installatie wordt achtereenvolgens met de volgende stappen uitgevoerd: "BuildWorld", "BuildKernel", InstallWorld, InstallKernel en als laatste worden de "custom functions" uitgevoerd, images gecreëerd en eventuele "last_order" scripts.

Alle opties onder CONF_BUILD=' worden doorgevoerd gedurende het BuildWorld-proces van de installatie.

Alle opties onder CONF_INSTALL=' worden doorgevoerd gedurende het InstallWorld-proces van de installatie

Alle opties onder CONF_WORLD=' worden doorgevoerd bij zowel BuildWorld als InstallWorld.

Deze manier van configureren heeft als voordeel dat bij het compileren van de NanoBSD “world”-directory een installatie geproduceerd wordt, die uiteindelijk bij de InstallWorld stap gestript en of verder aangepast kan worden.
Het is mogelijk om zowel de BuildWorld als de BuildKernel-stap over te slaan als deze zijn uitgevoerd op basis van dezelfde object-directory. Op deze manier kunnen verschillende installaties geproduceerd worden zonder deze tijdrovende “build stages” opnieuw te hoeven uitvoeren.

`FlashDevice:` geeft aan over welk type compactflash-kaart het doelsysteem beschikt. De opgegeven naam moet bekend zijn binnen het bestand “`FlashDevice.sub`”, waarbinnen de geometrie van verschillende typen compactflash-kaarten staan beschreven. Indien de deze geometrie niet bekend is kan dit handmatig worden opgezocht.
Als gebruik gemaakt wordt gemaakt van een compactflash-naar-IDE adapter, kan de informatie worden uitgelezen met “diskinfo -v /dev/adX”, waar X staat voor het nummer van de aangesloten schijf.
Wanneer dit niet mogelijk is, kan een compactflash-kaart ook gestart worden in een soekris hostsysteem. De bios post zou dan de volgende regel moeten weergeven:

```
Pri Mas     Hitachi XX.V.3.4.0.0          LBA 695-15-48     250 Mbyte
```

In het voorbeeld is een Hitachi compactflash-kaart beschikbaar van 256MB. De geometrie gegevens bereken je als volgt:

MEDIASIZE = 15 HEADS x 48 SECTORS x 695 = 500400 x 512
MEDIASIZE = 256204800 bytes

Als laatste stap binnen de installatie worden de “custom functions” of macro’s uitgevoerd. In het NanoBSD-configuratiebestand is het mogelijk eigen functies en shell-scripts te definiëren die aangeroepen worden via “`customize_cmd`”. De macro’s worden hierbij opeenvolgend van boven naar beneden uitgevoerd.

```
cust_nobeastie () (

    touch ${NANO_WORLDDIR}/boot/loader.conf
    echo "beastie_disable=\"YES\"" >> ${NANO_WORLDDIR}/boot/loader.conf
)
```

De bovenstaande functie in het voorbeeld zorgt ervoor dat het FreeBSD-beastie-menu bij het opstarten wordt uitgeschakeld.

```
cust_ttys () (

    sed -i "" -e /ttyd0/s/off/on/ ${NANO_WORLDDIR}/etc/ttys
    sed -i "" -e /ttyd0/s/std.9600/std.115200/ ${NANO_WORLDDIR}/etc/ttys
    sed -i "" -e /ttyd0/s/dialup/vt100/ ${NANO_WORLDDIR}/etc/ttys
    sed -i "" -e '/^ttyv[0-8]/s/    on/     off/' ${NANO_WORLDDIR}/etc/ttys
)
```

De bovenstaande functie zorgt ervoor dat een seriële consoleverbinding kan worden opgezet met het doel systeem, en dat een valide terminal type wordt ingesteld voor de `TERM` variabele. Dit is nodig om binnen het systeem een terminal editor zoals `vi` te kunnen gebruiken bij het aanmaken en bewerken van bestanden.

De macro’s / eigen functie worden als volgt in het NanoBSD-configuratiebestand aangeroepen:

customize_cmd cust_nobeastie
customize_cmd cust_ttys

Naast de uitgeschreven macro's in de bovenstaande voorbeelden, zijn binnen NanoBSD ook een aantal functies die standaard via "customize_cmd" kunnen worden aangeroepen. Voor deze overige configuratie opties kan het beste gekeken worden naar het nanobsd.sh shell-script waarin alle default configuratie-opties staan.

2.2.5. Installatie.

Als de gewenste configuratie-instellingen binnen de NanoBSD-configuratie-directory zijn doorgevoerd, kan gestart worden met het compileren van de systeeminstallatie.
Vanuit de promt kan de installatie worden gestart met de volgende opties:

```
# sh nanobsd.sh

[-b/-k/-w] [-c config_bestand]
-k     rem BuildKernel stage
-w     rem BuildWorld stage
-b     rem beide
-c     specificeer configuratiebestand
```

Wanneer geen opties worden gespecificeerd wordt een volledige NanoBSD-configuratie aangemaakt in de nanobsd.full object-directory.
Als gebruik wordt gemaakt van een configuratiebestand wordt een object-directory aangemaakt met de naam nanobsd.${NANO_OBJ}, waar de variabele staat voor de naam die binnen het configuratiebestand is ingevuld bij NANO_NAME.

Voor de verschillende stappen die het installatieproces doorloopt worden aparte log-bestanden aangemaakt. Het lokaliseren en debuggen van fouten kan op deze manier gemakkelijker worden uitgevoerd.

```
# cd /usr/obj/nanobsd.NET4801/
# ls

_.bk                          _.cust.install_rrdtool        _.etc
_.bw                          _.cust.soekris_comconsole     _.fdisk
_.cust.cust_addpkg            _.cust.soekris_ssh            _.ik
_.cust.cust_allow_ssh_root    _.di                          _.iw
_.cust.cust_comconsole        _.disk.full                   _.mnt
_.cust.cust_install_files     _.disk.image                  _.mtree
_.cust.cust_nobeastie         _.dl                          _.w
_.cust.install_apache         _.du                          make.conf
_.cust.install_netsnmp        _.env                         usr
```

Hierboven staat een voorbeeld van een object-directory waarbinnen een succesvolle NanoBSD-installatie is uitgevoerd. Op de images _.disk.full, _.disk.image en de NanoBSD-world-directory (_.w) na zijn de overige bestanden die beginnen met "_." De logs van de verschillende installatiestappen.
Bij het afronden van het installatieproces worden twee imagebestanden aangemaakt, het bestand _.disk.full en _disk.image. Het bestand _disk.full bevat de totale systeeminstallatie en moet gebruik worden om naar een flashdisk te schrijven.
Het bestand _disk.image bevat een enkele slice met daarop een systeeminstallatie, zodat het mogelijk is de systeemslice van het doelsysteem onafhankelijk van de overige configuratie te updaten.

De images kunnen naar de gewenste flashkaart worden geschreven met de dd utility. Hierbij moet rekening gehouden worden met de manier waarop de flash kaart op het host systeem is aangesloten. Vanuit de NanoBSD object directory kan het volgende commando worden gegeven.

```
# dd if=_.disk.full of=/dev/da0 bs=64k
```

In het voorgaande voorbeeld wordt het image naar een compactflash-kaart geschreven die is aangesloten via een USB-kaartlezer. Wanneer gebruik wordt gemaakt van een compactflash-naar-IDE-adapter dient bij de “output file” adX als apparaat-ID worden geselecteerd, waarbij de X voor het nummer van de doelschijf staat.

Als het image succesvol naar de compactflash kaart is geschreven kan het doelsysteem voor de eerste keer opgestart worden.
Wijzigingen aan het operationele systeem kunnen worden doorgevoerd door het root file systeem tijdelijk in “read-write” modus te koppelen.

```
# mount –u –o rw / of: mount –uw –o noatime /n
```

Na het aanbrengen van de wijzigingen dient het comapctflash-geheugen weer “read-only” gekoppeld te worden:

```
# mount –u –o ro / of: mount -r
```

2.2.6. Ports & packages.

Zoals in de introductie staat beschreven kunnen ports en packages binnen NanoBSD op dezelfde manier worden gebruikt als bij FreeBSD
Het toevoegen van diverse ports aan de installatie van NanoBSD kan op verschillende manieren worden uitgevoerd. In het nanobsd.sh shell-script is een functie opgenomen die het installeren van voorgecompileerde packages vereenvoudigt.

```
cust_pkg () (

	# Copy packages into chroot
	mkdir -p ${NANO_WORLDDIR}/Pkg
	cp ${NANO_PACKAGE_DIR}/* ${NANO_WORLDDIR}/Pkg

	# Count & report how many we have to install
	todo=`ls ${NANO_WORLDDIR}/Pkg | wc -l`
	echo "=== TODO: $todo"
	ls ${NANO_WORLDDIR}/Pkg
	echo "==="
	while true
	do
		# Record how may we have now
		have=`ls ${NANO_WORLDDIR}/var/db/pkg | wc -l`

		# Attempt to install more packages
		# ...but no more than 200 at a time due to pkg_add's internal
		# limitations.
		chroot ${NANO_WORLDDIR} sh -c \
			'ls Pkg/*tbz | xargs -n 200 pkg_add -F' || true

		# See what that got us
		now=`ls ${NANO_WORLDDIR}/var/db/pkg | wc -l`
		echo "=== NOW $now"
		ls ${NANO_WORLDDIR}/var/db/pkg
		echo "==="


		if [ $now -eq $todo ] ; then
			echo "DONE $now packages"
			break
		elif [ $now -eq $have ] ; then
			echo "FAILED: Nothing happened on this pass"
			exit 2
		fi
	done
	rm -rf ${NANO_WORLDDIR}/Pkg
)
```

Voor de bovenstaande functie is een omgevingsvariabele opgenomen in het NanoBSD shell-script; `${NANO_PACKAGE_DIR} = "/usr/src/tools/tools/nanobsd/Pkg"`. Aan deze directory kunnen voorgecompileerde packages worden toegevoegd, die bij het aanmaken van een NanoBSD-image worden geïnstalleerd binnen de systeemomgeving.
Voordat een FreeBSD-ports aan de `/Pkg` directory kan worden toegevoegd, dient hiervan eerst een pakkagebestand gecreëerd te worden. Vanuit de gewenste port-directory kan het volgende commando worden gegeven:

```
# make install package-recursive
```

Kopieer vervolgens het bestand "`port_naam.tbz`" naar de NanoBSD-packagedirectory. Om de functie aan te roepen moet deze worden opgenomen in de lijst met macro's die bij een NanoBSD-installatie worden uitgevoerd.
Naast het pakkagebestand van de betreffende port zelf, kan het zijn dat deze nog afhankelijkheden heeft. Bij het uitvoeren van de "cust_pkg" –functie, wordt een lijst gemaakt van alle packages die zijn toegevoegd aan `/Pkg`. Het script doorloopt vervolgens deze lijst en wanneer een port niet geïnstalleerd kan worden ,vanwege een (nog)niet geïnstalleerd afhankelijk pakket, gaat het script verder met de volgende port.
Op deze manier wordt de gehele lijst doorlopen, op volgorde van ASCI-reeks, en wordt aan het einde van het proces de initiële lijst met packages vergeleken met de packages die daadwerkelijk zijn toegevoegd. Aan de hand van dit resultaat kan bepaald worden of alle benodigde ports succesvol zijn geïnstalleerd. Indien dit niet het geval is wordt het proces voor de missende ports herhaald tot dat het proces geen packages meer kan installeren. Wanneer dan alsnog een port niet is geïnstalleerd, faalt het script en kan in het betreffende log-bestand worden teruggelezen welke benodigde afhankelijkheden niet zijn toegevoegd aan de `/Pkg` –directory.

Het is ook mogelijk packages te laten compileren gedurende het NanoBSD installatieproces. De onderstaande functie zou hiervoor gebruikt kunnen worden:

```
install_netsnmp() (
      cd /usr/ports/net-mgmt/net-snmp
      FORCE_PKG_REGISTER=1 make install package-recursive
        cp net-snmp-5.3.1_7.tbz ${NANO_OBJ}/_.w
      chroot "$NANO_WORLDDIR" sh -c "pkg_add -vF net-snmp-5.3.1_7.tbz"
      chroot "$NANO_WORLDDIR" sh -c "rm -f net-snmp-5.3.1_7.tbz"
)
```

In het voorbeeld wordt de port van net-snmp gecompileerd en geïnstalleerd. Deze functie is echter wel versie afhankelijk en kan bij een mogelijk versie update van de desbetreffende port niet meer uitgevoerd worden.

2.2.7. Diskless-configuratie.

Een belangrijk punt van een NanoBSD-installatie dat nog niet is besproken, is de methode die wordt gebruikt om de systeemomgeving "diskless" te laten functioneren. Met diskless wordt normaal gesproken een systeem bedoeld dat niet beschikt over een fysieke hardeschijf. Daar NanoBSD gericht is op een embedded-systeemomgeving, zal over het algemeen gebruik worden gemaakt van flashgeheugen voor de data opslag. Door het beperkte aantal schrijfacties wat op dit type geheugen kan worden uitgevoerd, zoals eerder in dit document staat beschreven, wordt de root van een NanoBSD systeemomgeving in "read-only" modus gekoppeld. Daarnaast worden ramdisks aangemaakt voor de rootdirectories waarbinnen schrijfacties uitgevoerd moeten kunnen worden. De methode die bij NanoBSD toegepast worden voor het aanmaken en bevolken van de `/etc` en `/var` ramdisks maakt gebruikt van het `rc.initdiskless` –script.

Het script wordt bij het opstarten gebruikt om ramdisks aan te maken en deze te bevolken vanuit een basisconfiguratie. De configuratie van `rc.initdiskless` –script kan in de systeemomgeving gevonden worden onder de directory `/conf`.

De onderliggende structuur kan worden onderverdeel in “/base” en “/default” , beide gelden als template-directories waarbinnen bestanden en directories kunnen worden geplaatst, die gebruikt worden bij het aanmaken en vullen van de ramdisks.
De /base -directory wordt gezien als basis voor het aanmaken van de ramdisks en bevat een replica van de originele /etc en /var -directories. Aan de directory “/default” kunnen bestanden en directories worden toegevoegd die de standaardconfiguratie, aangemaakt op basis van /conf/base, overschrijven. Naast de normale bestanden en directories kunnen een aantal specifieke configuratiebestanden voor het rc.initdiskless script worden aangemaakt. Voor de NanoBSD-installatie is in de directory /default/etc een remount bestand aangemaakt, met de volgende inhoud:

- mount –o ro /dev/ad0s3

Dit bestand zorgt ervoor dat bij het opstarten de configuratieslice tijdelijk “read-only” wordt gekoppeld, waarna eventueel aanwezige bestanden naar de /etc directory worden gekopieerd. Het volume van de ramdisks wordt bepaald door het configuratiebestand md_size. Dit bestand dient aanwezig te zijn in de root van de aanwezige directories onder /conf/base/. Voor de NanoBSD-installatie zijn dit respectievelijk /conf/base/var/md_size en /conf/base/etc/md_size. Het lege bestand diskless in de directory /etc zorgt voor de initialisatie van het rc.initdiskless proces.

De configuratiebestanden van het rc.initdiskless –script worden automatisch aangemaakt bij het NanoBSD-installatieproces. Naast deze bestanden is de volgende additionele wijziging gemaakt in het bestand /etc/defaults/rc.conf

- root_rw_mount=NO

Dit zorgt ervoor dat het root bestandsysteem standaard in “read-only” modus wordt gekoppeld.

3 Node Hardware

3.1 Introductie

In dit hoofdstuk worden de verschillende onderdelen besproken die gerelateerd zijn aan de implementatie van hardware. Het beschrijft de specificaties en prestaties van de hardwarecomponenten die gedurende het project zijn gebruikt, bij het testen en ontwikkelen van het IRIS-prototype.

3.1.1. Testmethode

Bij het testen van de hardwarecomponenten zijn zowel de individuele prestaties als de prestaties binnen de opzet van de IRIS-architectuur gemeten. Daarnaast zijn de componenten onderling achteraf vergeleken. Om de prestaties te meten, zijn binnenshuis verschillende testopstellingen geconfigureerd. Hierbij zijn met Iperf metingen uitgevoerd, waar naast de beschikbare bandbreedte ook de systeembelasting is genoteerd.

http://iperf.sourceforge.net/

Iperf is een applicatie voor het analyseren van de maximaal beschikbare bandbreedte binnen een netwerk. Hierbij kan getest worden met het TCP en UDP protocol. Omdat Iperf bij de dataoverdracht geen gebruikt maakt van fysieke opslag, kan de doorvoersnelheid hierdoor niet worden gelimiteerd.

3.2.1. Testomgeving

Voor het meten van de hardware prestaties is een testomgeving ingericht. Hierbij zijn een viertal systemen geïnstalleerd met een Linux-distributie, die dienen voor het opzetten van onafhankelijke Iperf- cliënt / server connecties. De minimale hardware specificaties waarover deze systemen beschikken is een Intel PIV processor met 1GB intergeheugen. De volgende iperf versie is aan deze systeemconfiguraties toegevoegd, Iperf 2.0.3.

Bij het opzetten van de WiFi-verbindingen zijn de volgende kaarten gebruikt:

- 2 x Ubiquiti SR 802.11a/b/g Cardbus, Atheros 5004 chipset,
- 1 x Winstron DCMA-82 802.11a/b/g Mini-PCI, Atheros 5006 chipset,
- 1 x Toshiba 802.11b/g Mini-PCI, Atheros 5213 chipset,
- 1 x 3com 802.11a/b/g PCI, Atheros AR5212 chipset,
- 1 x Prism 2.5 802.11b/g Mini-PCI.

3.2 Soekris NET4801

De Soekris NET4801 is een embedded moederbord gebaseerd op een 586 chip van de AMD Geode lijn. Het moederbord is ontwikkeld door Soekris Engineering en is gericht op het gebruik als communicatie toepassing.

http://www.soekris.com/net4801.htm

De beschikbaarheid van meerdere ethernet poorten, een Mini-PCI slot en de mogelijkheid om compactflash te gebruiken voor opslag, maakt het systeem geschikt als basis-node binnen de IRIS architectuur. Daarbij worden moederborden van Soekris engineering al geruime tijd gebruikt voor bestaande nodes binnen het netwerk van Wireless Leiden.

De Soekris NET4801 heeft echter inmiddels bijna zijn “end of life” cyclus bereikt, maar is toch gekozen als een optionele testversie vanwege de algemene beschikbaarheid binnen de organisatie.
De productie stop van de AMD Geode SC1100 processor, die in de Soekris NET4801 wordt gebruikt, is de reden dat deze systemen binnenkort niet meer geleverd kunnen worden. Bij de fabrikant wordt de Soekris NET5501 geprofileerd als opvolger.

Soekris engineering, NET4801

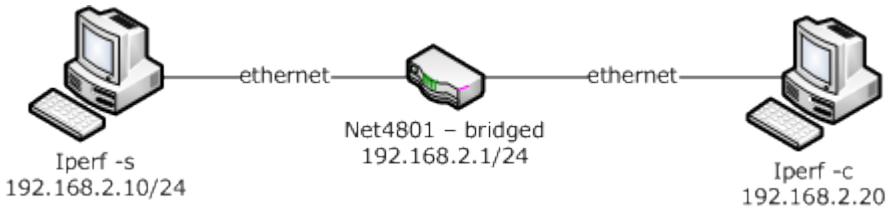
3.2.1. Systeemspecificaties

De onderstaande specificaties gelden voor het Soekris NET4801 moederbord dat is gebruikt gedurende het IRIS project. Het is mogelijk dat specificaties van soortgelijke types enigszins afwijken.

- 266 MHz AMD Geode NSC SC1100 single chip processor,
- 128 MB SDRAM, soldered on board,
- Compactflash Type I/II socket,
- UltraDMA-33 interface with 44 pins connector,
- 3 10/100 Mbit Ethernet ports, RJ-45, SiS
- 2 Serial ports,
- 1 Mini-PCI type III socket,
- 1 PCI slot,
- USB 1.1 interface,
- Board size 13.2 x 14.5cm.

3.2.2. Prestaties over ethernet, bridged

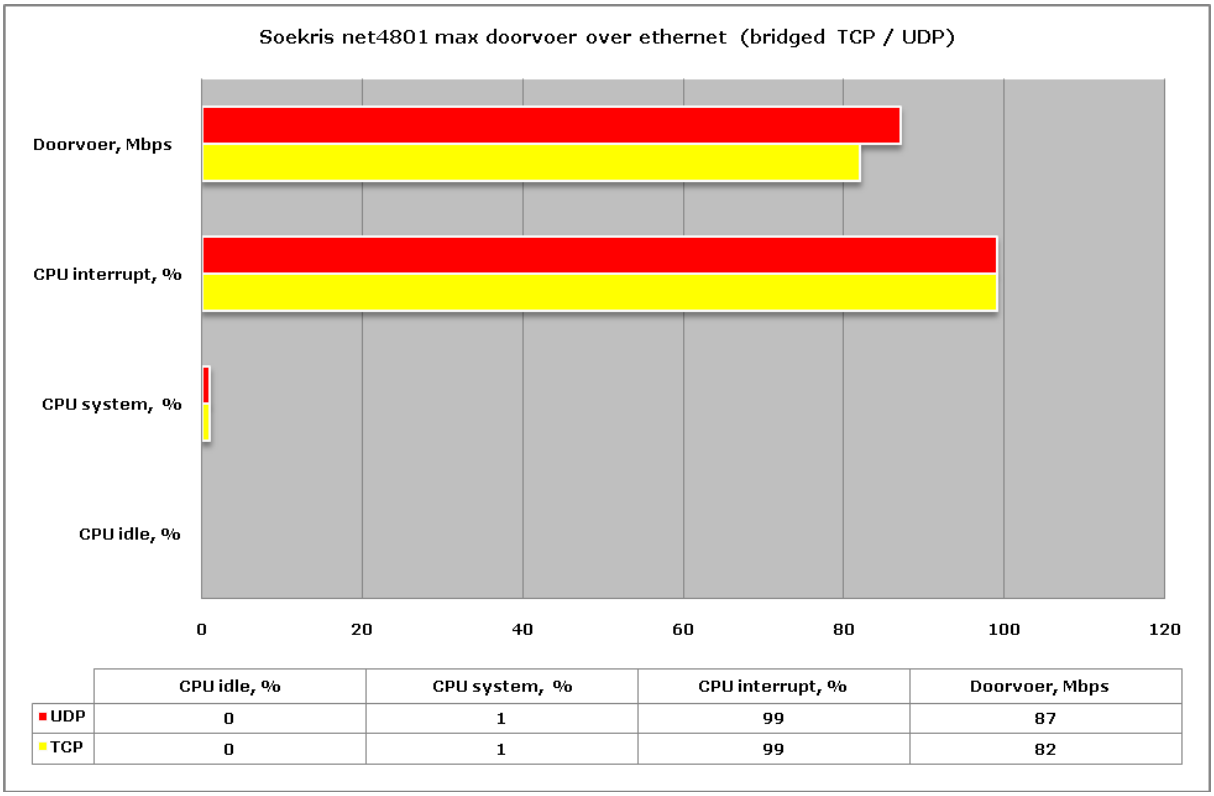
Omdat het Soekris net4801 systeembord beoogd is als mogelijk optie voor de basis-node binnen het IRIS-concept, is het belangrijk in een basisopstelling de prestaties van het systeem bij de maximale doorvoersnelheid te meten. Functioneel gezien zal het systeembord data van de aangesloten interlinks gaan routeren over ethernet. Hoewel deze prestatie er het meest toe doen, is eerst in een “bridged” -opstelling getest.



Soekris net4801, bridged -opstelling

De prestaties zijn gemeten in de opstelling hierboven. Het net4801 systeembord is geconfigureerd als bridge, waarbij de ethernet poorten sis0 t/m sis2 aan de bridge zijn toegevoegd. De iperf cliënt en server zijn opgezet op losstaande systemen. De Iperf metingen zijn een aantal keer uitgevoerd met de volgende opties om zowel met het TCP- als UDP-protocol de bandbreedte en systeembelasting te meten.

```
TCP # iperf –c IP-adres –t 30  /  iperf –s
UDP # iperf –c IP-adres –u –b 100mbit –t 30 / iperf –s -u
```



	CPU idle, %	CPU system, %	CPU interrupt, %	Doorvoer, Mbps
UDP	0	1	99	87
TCP	0	1	99	82

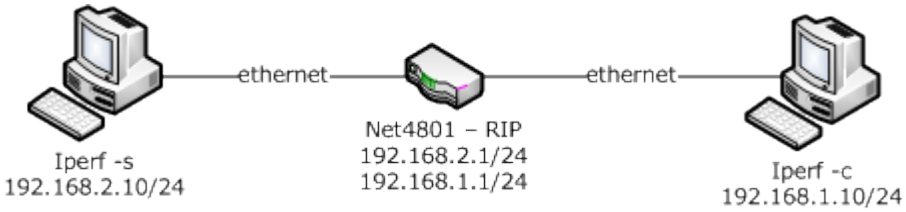
Soekris net4801, bridged -resultaten

De resultaten die in de grafiek hierboven staan weergegeven geven de maximale doorvoersnelheid weer, in Megabits per seconde, bij het gebruikt van UDP en TCP. Daarnaast wordt de systeembelasting getoond die is genoteerd gedurende deze iperf dataoverdrachten. Voor de resultaten is een gemiddelede genomen uit drie metingen. De systeembelasting is hierbij direct op het soekris net4801 systeembord gemeten met “top” . De systeembelasting is daarbij verdeeld in “system”, “interrupt” en “idle”

percentages. De maximale doorvoersnelheid ligt respecitievelijk op 87 en 82 Mbit per seconde. Naar verwachting ligt de doorvoersnelheid voor UDP bij een iets hoger vanwege de minimale overhead van dit protocol. Wat opvalt zijn de grote hoeveelheden interrupt-aanvragen, wat ervoor zorgt dat de processor van het systeembord volledig wordt belast.

3.2.3. Prestatiesr over ethernet, RIP

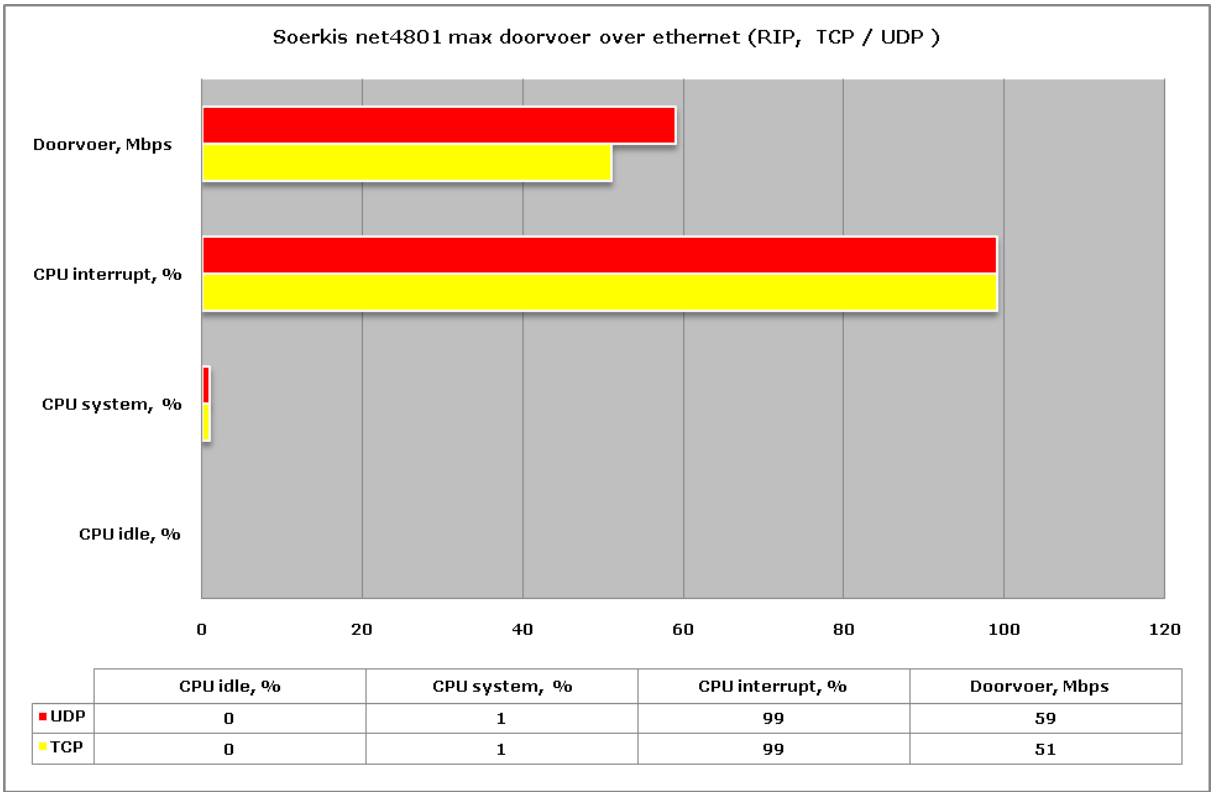
Naast het meten van de prestaties in een “bridge” -configuratie, is het belangrijkste de doorvoersnelheid en systeembelasting te meten in een testopstelling waarbij de data over ethernet gerouteerd wordt.



Soekris net4801, routed –opstelling

De prestaties zijn gemeten in de opstelling hierboven. Het net4801 systeembord is geconfigureerd als gateway, waarbij het standaard “Routing Information Protocol” wordt gebruikt om het dataverkeer te routeren. De ethernet poorten sis0 en sis1 zijn geconfigureerd binnen een eigen subnet. Op basis van deze ip-configuratie zijn de iperf cliënt en server connecties opgezet op losstaande systemen. De Iperf metingen zijn een aantal keer uitgevoerd met de volgende opties, om zowel met het TCP- als UDP-protocol de bandbreedte en systeembelasting te meten.

```
TCP # iperf –c IP-adres –t 30  /  iperf –s
UDP # iperf –c IP-adres –u –b 100mbit –t 30 / iperf –s -u
```



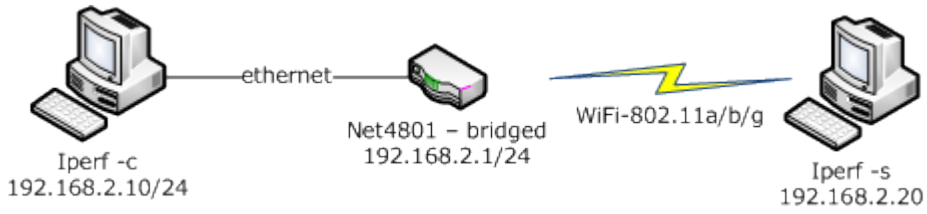
	CPU idle, %	CPU system, %	CPU interrupt, %	Doorvoer, Mbps
UDP	0	1	99	59
TCP	0	1	99	51

Soekris net4801, RIP -resultaten

De resultaten voor de “routed” -opzet zijn op dezelfde manier weegegeven als de “bridged” –opstelling, waarbij eveneens een onderverdeling is gemaakt in doorvoersnelheid voor de UDP- en TCP-dataoverdracht. De maximale doorvoersnelheid die is gemeten ligt respectievelijk op 59 en 51 Mbit/s. De resultaten liggen naar verwachting lager dan bij de voorgaande opstelling, daar de belasting voor het net4801 systeembord bij de configuratie als bridge rond 82 mbit/s maximaal is. Wanneer routing wordt toegapst vraagt dit logischerwijs meer van het systeem, waardoor de uiteindelijke doorvoersnelheid lager komt te liggen.

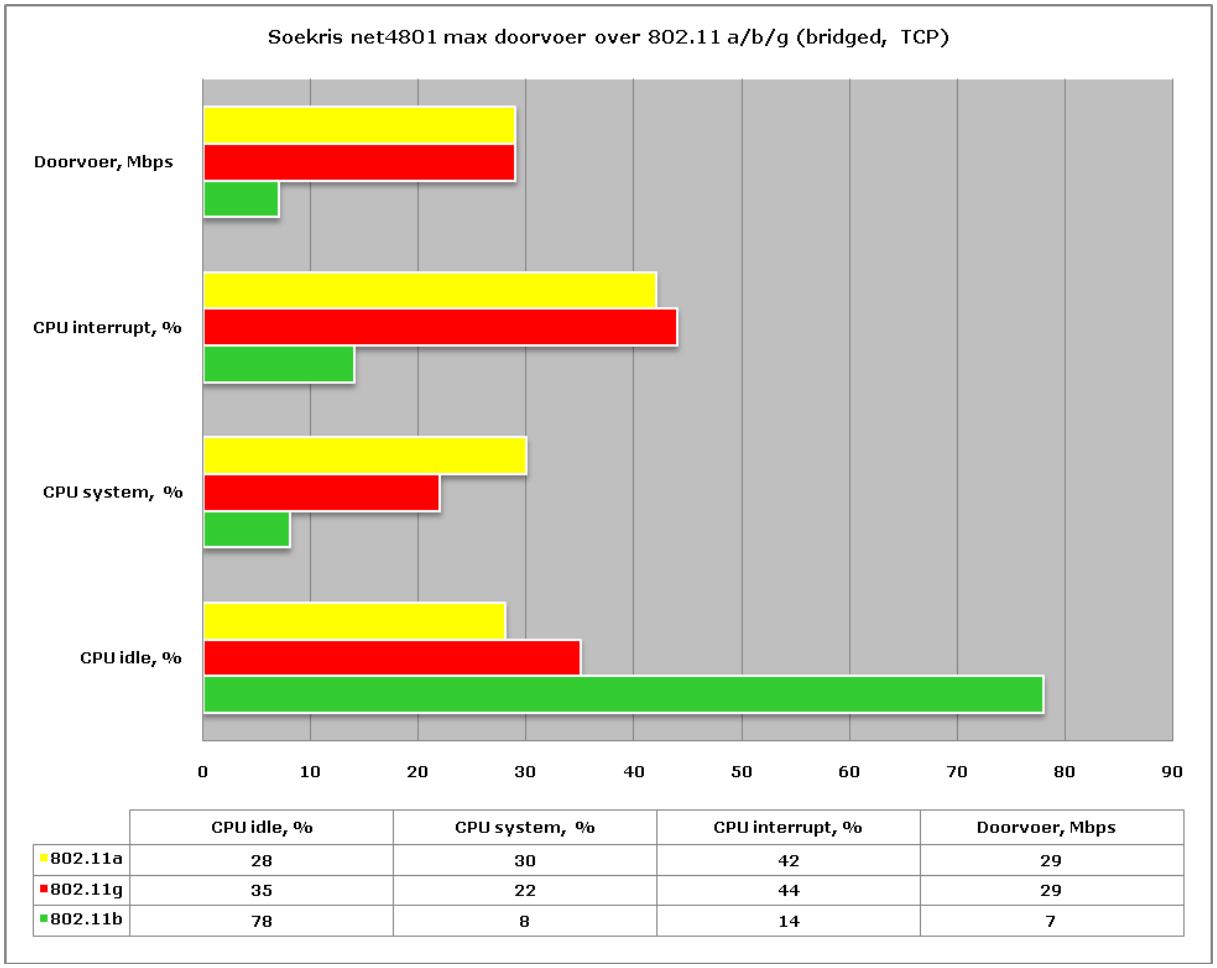
3.2.4. Prestaties over 802.11a/b/g, bridged

Hoewel binnen de IRIS-architectuur dataverkeer voornamelijk gerouteerd gaat worden over ethernet, kan het zijn dat een onmindirectioneel toegangspunt direct op het systeembord wordt geconfigureerd, in plaats van hiervoor een onafhankelijke interface aan te sluiten. In de volgende opstelling zijn de prestaties gemeten voor de verschillende 802.11-WiFi standaarden



Soekris net4801, 802.11a/b/g, bridged -opstelling

Het net4801 systeembord is evenals bij de voorgaande metingen eerste geconfigureerd als bridge, waarbij sis0 en ath0 aan de bridgeconfiguratie zijn toegevoegd.
Ath0 is geconfigureerd als toegangspunt voor het opzetten van de verschillende WiFi-verbindingen. Hiervoor is een Winstron DCMA-82 Mini-PCI kaart gebruikt met een antenne die geschikt is voor zowel de 2.4GHz- als de 5GHz-frequentieband.
Een van de systemen die gebruikt worden voor het opzetten van de Iperf cliënt / server verbindingen is draadloos op het op het toegangspunt van de soekris aangesloten, het andere systeem is direct via ethernet aangesloten. Voor de WiFi-verbinding is een Ubiquiti 802.11a/b/g PcCard, via een Cardbus-to-PCI-adapter geïnstalleerd.
De metingen zijn in dezelfde opstelling uitgevoerd voor de 802.11a, b, en g WiFi-standaard.



	CPU idle, %	CPU system, %	CPU interrupt, %	Doorvoer, Mbps
802.11a	28	30	42	29
802.11g	35	22	44	29
802.11b	78	8	14	7

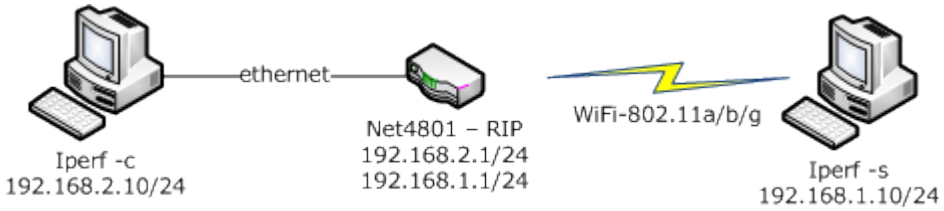
Soekris net4801, 802.11a/b/g, bridged –resultaten

De resultaten die in de grafiek hierboven staan weergegeven tonen de maximale doorvoersnelheid, in Megabits per seconde, die is behaald met 802.11a/b/g WiFi-standaard in een bridge-configuratie. Daarnaast wordt de bijbehorende systeembelasting weergegeven die is genoteerd tijdens de verschillende Iperf dataoverdrachten.
De resultaten laten een maximale doorvoersnelheid zien van respectievelijk 29, 29 en 7 Mbit/s. De resultaten van de 802.11a- en g-standaard zijn naar verwachting vergelijkbaar, daar deze naast dezelfde verbindingsnelheid ook gebruik maken van dezelfde modulatie-techniek. De 802.11b- standaard laat een duidelijk lagere belasting van het systeem zien, wat kan worden verklaard door de beperkte doorvoersnelheid.

Het gebruik van WiFi, met name de 802.11a/g –standaard, laat in vergelijking met ethernet een duidelijk hoger percentage zien aan CPU-cycles dat is toebedeeld aan “system”. Dit past bij de eerdere ervaringen binnen WireLess Leiden, waar werd verondersteld dat voor de 802.11a -standaard zwaardere systeemspecificaties aan de node-hardware gesteld moesten worden.

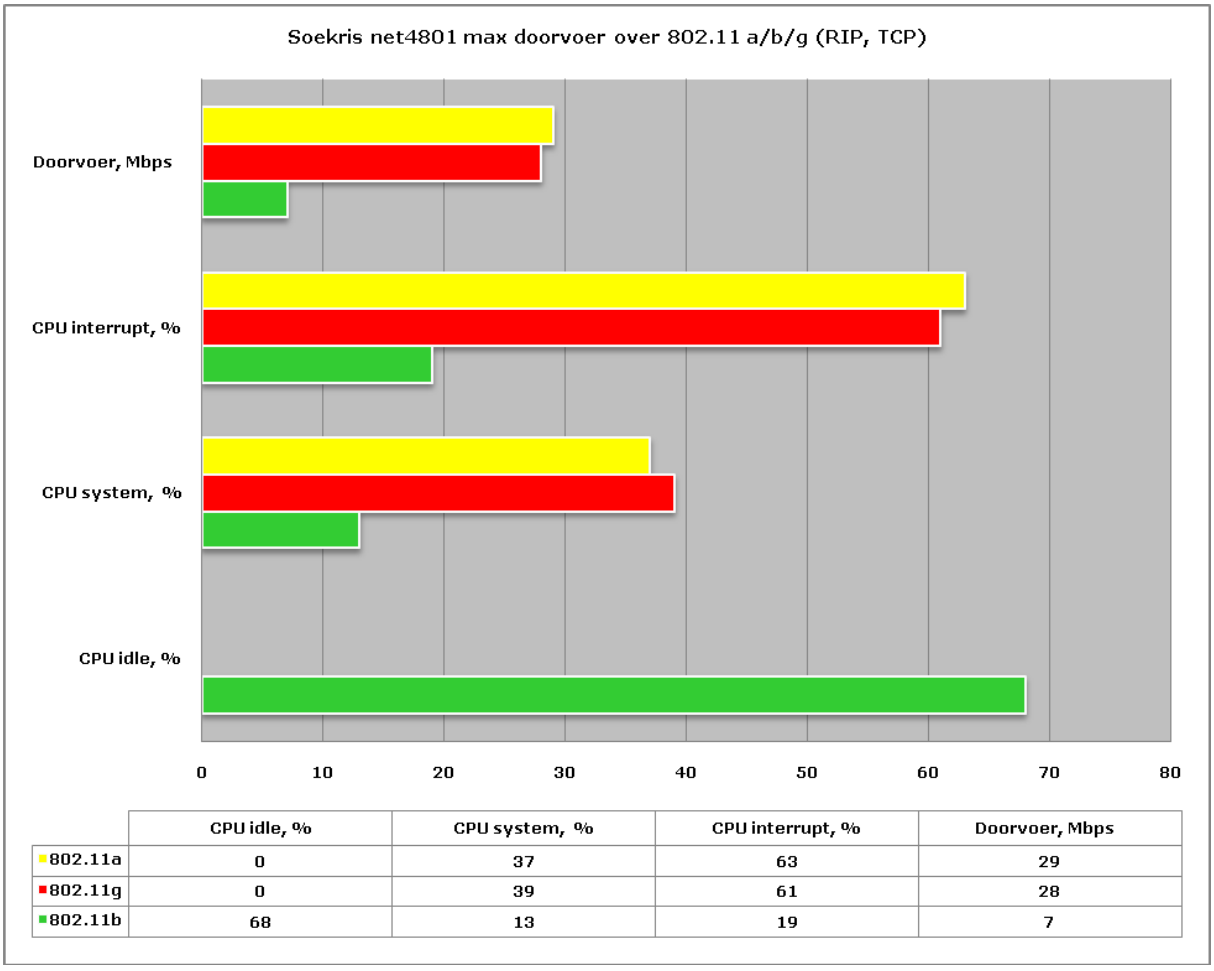
3.2.5. Prestaties over 802.11a/b/g, RIP

Naast de bridgeconfiguratie, zijn de prestaties van het net4801-systeembord ook gemeten bij het routeren van het dataverkeer over WiFi. Hierbij zijn in dezelfde testopstelling nogmaals de doorvoersnelheid en systeembelasting bij het gebruik van de 802.11a/b/g standaarden gemeten.



Soekris net4801, 802.11a/b/g, routed –opstelling

De prestaties zijn gemeten in de opstelling hierboven. Het net4801 systeembord is geconfigureerd als gateway, waarbij het standaard “Routing Information Protocol” wordt gebruikt om het dataverkeer te routeren. Sis0 en Ath0 zijn geconfigureerd binnen een eigen subnet. Op basis van deze ip-configuratie zijn de iperf cliënt en server connecties opgezet op losstaande systemen



	CPU idle, %	CPU system, %	CPU interrupt, %	Doorvoer, Mbps
802.11a	0	37	63	29
802.11g	0	39	61	28
802.11b	68	13	19	7

Soekris net4801, 802.11a/b/g, RIP –resultaten

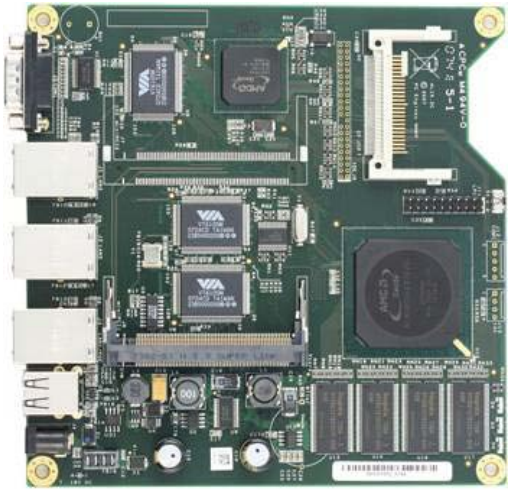
De resultaten in de grafiek hierboven laten zien dat bij het routeren van het dataverkeer over WiFi, de systeembelasting eveneens toeneemt ten opzichte van de bridgeconfiguratie. Dit veroorzaakt echter nog geen daling in de maximale doorvoersnelheid en de resultaten hiervan komen overeen met de behaalde snelheden binnen de bridge-opstelling. Omdat de systeembelasting bij de 802.11a- en g-standaard maximaal is, kan gesteld worden dat het routeren van dit type dataverkeer op volledige snelheid tegen de grens van de systeemcapaciteit aanloopt.

3.3 PcEngines Alix2D3

Het Pc Engines Alix2D3 moederbord is gebaseerd op de nieuwste processor van de AMD Geode lijn. De 500MHz AMD Geode Lx800, deze chipt dient samen met de lx700 als opvolger van de Geode SC1100 chip die onder andere gebruikt wordt op het systeembord van de Soekris net4801. Het Alix2D3 moederbord is ontwikkeld door de fabrikant Pc Engines en is gericht op het gebruik als “embedded” -communicatietoepassing.

http://www.pcengines.ch/alix2d3.htm

Het systeembord beschikt evenals de Soekris net4801 over meerdere ethernet poorten, een Mini-PCI slot en de mogelijkheid om compactflash te gebruiken voor dataopslag. Dit maakt dat het systeem geschikt is als toepassing voor de basis-node binnen de IRIS architectuur. De prestaties van het systeem zijn in dezelfde testopstellingen onderzocht als het Soekris net4801 systeembord, zodat deze onderling vergeleken kunnen worden.



Pc Engines, Alix2D3.

3.3.1. Systeemspecificaties

De onderstaande specificaties gelden voor het Pc Engines Alix2D3 moederbord dat is gebruikt gedurende het IRIS project. Het is mogelijk dat specificaties van soortgelijke types enigszins afwijken.

- 500 MHz AMD Geode Lx800 single chip processor,
- 256 MB SDRAM, soldered on board,
- Compactflash Type I/II socket,
- UltraDMA-33 interface with 44 pins connector,
- 3 10/100 Mbit Ethernet ports, RJ-45, Via
- 1 Serial port,
- 1 Mini-PCI type III socket,
- Power over Ethernet, via vr0,
- 2 USB 1.1 interfaces,
- Board size 152.4 x 152.4 mm.

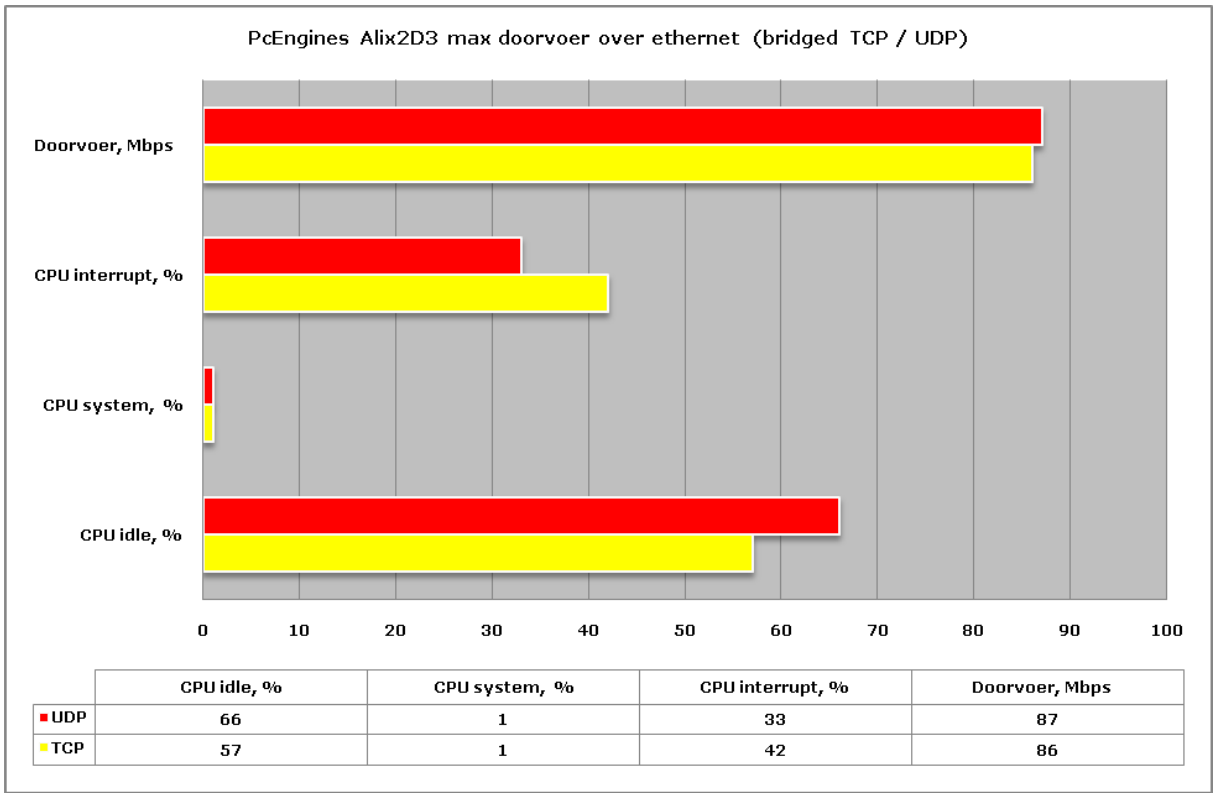
3.3.2. Prestaties over ethernet, bridged

In deze basisopstelling wordt als eerste de maximale doorvoersnelheid en bijbehorende systeembelasting bij een dataoverdracht over ethernet gemeten. De iperf testmetingen worden hiervoor uitgevoerd met zowel het UDP als TCP-protocol.

Alix2D3, bridged -opstelling

De prestaties zijn gemeten in de opstelling hierboven. Het Alix2D3-systeembord is geconfigureerd als bridge, waarbij de ethernet poorten vr0 t/m vr2 aan de bridge zijn toegevoegd. De iperf cliënt en server connecties zijn opgezet op losstaande systemen. De Iperf metingen zijn een aantal keer uitgevoerd met de volgende opties om zowel met het TCP- als UDP-protocol de bandbreedte en systeembelasting te meten.

```
TCP # iperf –c IP-adres –t 30  /  iperf –s
UDP # iperf –c IP-adres –u –b 100mbit –t 30 / iperf –s -u
```



	CPU idle, %	CPU system, %	CPU interrupt, %	Doorvoer, Mbps
UDP	66	1	33	87
TCP	57	1	42	86

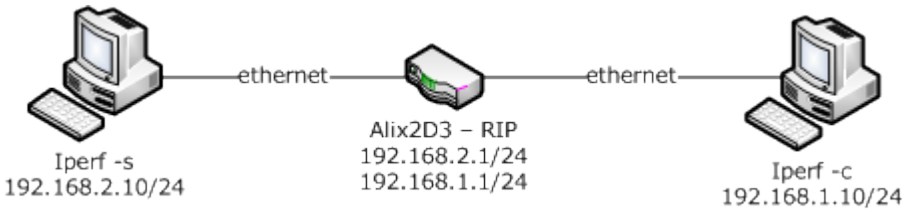
Alix2D3, bridged -resultaten

De resultaten die in de grafiek hierboven staan weergegeven geven de maximale doorvoersnelheid weer, in Megabits per seconde, bij het gebruikt van UDP en TCP. Daarnaast wordt de systeembelasting getoond die is genoteerd gedurende deze iperf dataoverdrachten. Voor de resultaten is een gemiddelede genomen uit drie metingen. De systeembelasting is hierbij direct op het soekris net4801 systeembord gemeten met “top” . De systeembelasting is daarbij verdeeld in “system”, “interrupt” en “idle” percentages.

De maximale doorvoersnelheden die zijn behaald, liggen met respectievelijk 86 en 87 megabits per seconde erg dicht bij elkaar. Het UDP protocol laat naar verwachting wel een lagere systeembelasting zien.

3.3.3. Prestaties over ethernet, RIP

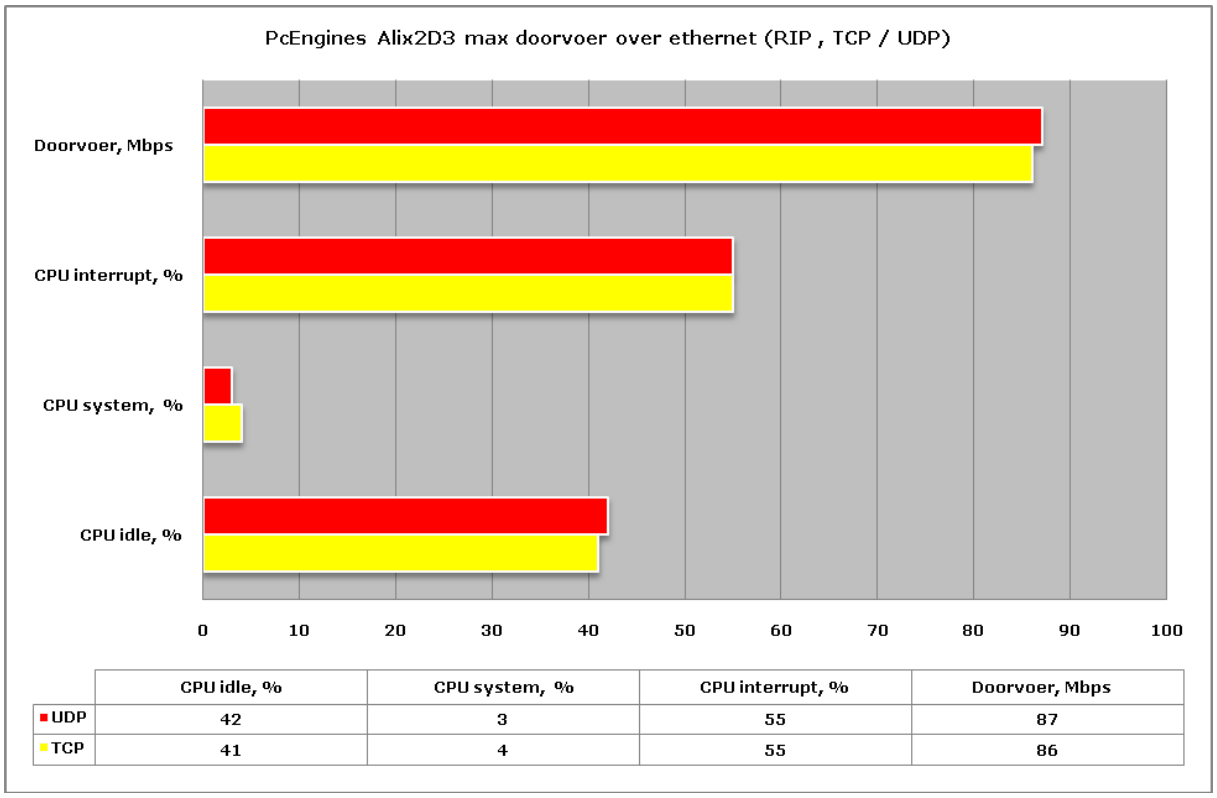
Naast het meten van de prestaties in een “bridge” -configuratie, is het belangrijkste de doorvoersnelheid en systeembelasting te meten in een testopstelling waarbij de data over ethernet gerouteerd wordt.



Alix2D3, routed –opstelling

De prestaties zijn gemeten in de opstelling hierboven. Het net4801 systeembord is geconfigureerd als gateway, waarbij het standaard “Routing Information Protocol” wordt gebruikt om het dataverkeer te routeren. De ethernet poorten sis0 en sis1 zijn geconfigureerd binnen een eigen subnet. Op basis van deze ip-configuratie zijn de iperf cliënt en server connecties opgezet op losstaande systemen. De Iperf metingen zijn een aantal keer uitgevoerd met de volgende opties, om zowel met het TCP- als UDP-protocol de bandbreedte en systeembelasting te meten.

```
TCP # iperf –c IP-adres –t 30  /  iperf –s
UDP # iperf –c IP-adres –u –b 100mbit –t 30 / iperf –s -u
```



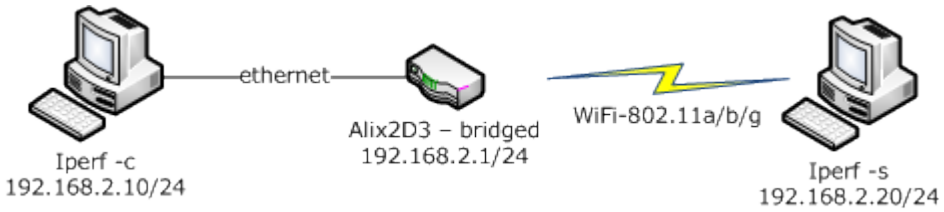
	CPU idle, %	CPU system, %	CPU interrupt, %	Doorvoer, Mbps
UDP	42	3	55	87
TCP	41	4	55	86

Alix2D3, routed –resultaten

De resultaten voor de “routed” -opzet zijn op dezelfde manier weegegeven als de “bridged” –opstelling, waarbij eveneens een onderverdeling is gemaakt in doorvoersnelheid voor de UDP- en TCP-dataoverdracht. De maximale doorvoersnelheid die is gemeten ligt respectievelijk op 87 en 86 Mbit/s, waarmee de resultaten gelijk zijn aan die van de dataoverdracht binnen bridgeconfiguratie. De totale systeembelasting ligt ongeveer 20% hoger wanneer het dataverkeer over ethernet gerouteerd wordt.

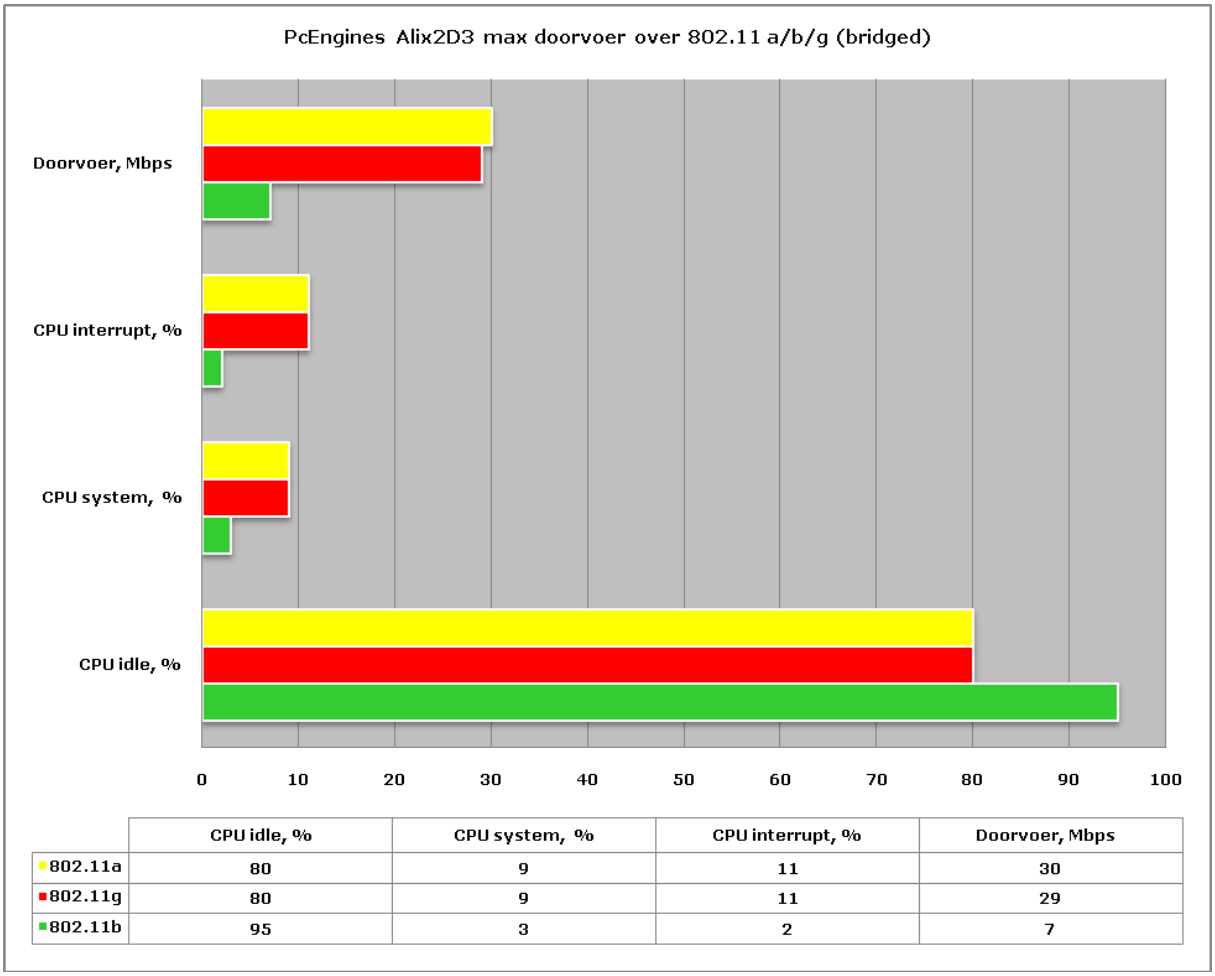
3.3.4. Prestaties over 802.11a/b/g, bridged

Evenals bij de Soekris net4801 is voor het Alix2D3-systeembord ook de dataoverdracht over WiFi, in combinatie met de bijbehorende systeembelasting, gemeten . In de volgende opstelling zijn de prestaties gemeten voor de verschillende 802.11-WiFi standaarden.



Alix2D3, 802.11a/b/g, bridged –opstelling

Het net4801 systeembord is evenals bij de voorgaande metingen eerste geconfigureerd als bridge, waarbij sis0 en ath0 aan de bridgeconfiguratie zijn toegevoegd.
Ath0 is geconfigureerd als toegangspunt voor het opzetten van de verschillende WiFi-verbindingen. Hiervoor is een Winstron DCMA-82 Mini-PCI kaart gebruikt met een antenne die geschikt is voor zowel de 2.4GHz- als de 5GHz-frequentieband.
Een van de systemen die gebruikt worden voor het opzetten van de Iperf cliënt / server verbindingen is draadloos op het op het toegangspunt van de soekris aangesloten, het andere systeem is direct via ethernet aangesloten. Voor de WiFi-verbinding is een Ubiquiti 802.11a/b/g PcCard, via een Cardbus-to-PCI-adapter geïnstalleerd.
De metingen zijn in dezelfde opstelling uitgevoerd voor de 802.11a, b, en g WiFi-standaard.



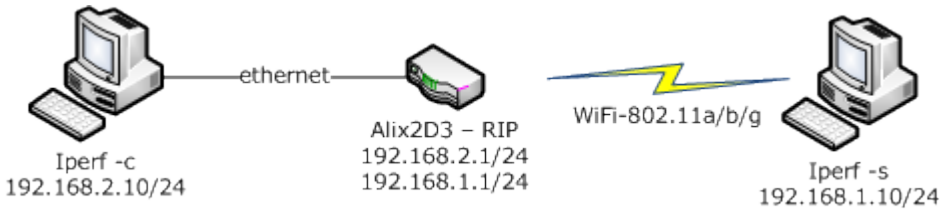
	CPU idle, %	CPU system, %	CPU interrupt, %	Doorvoer, Mbps
802.11a	80	9	11	30
802.11g	80	9	11	29
802.11b	95	3	2	7

Alix2D3, 802.11a/b/g, bridged –resultaten

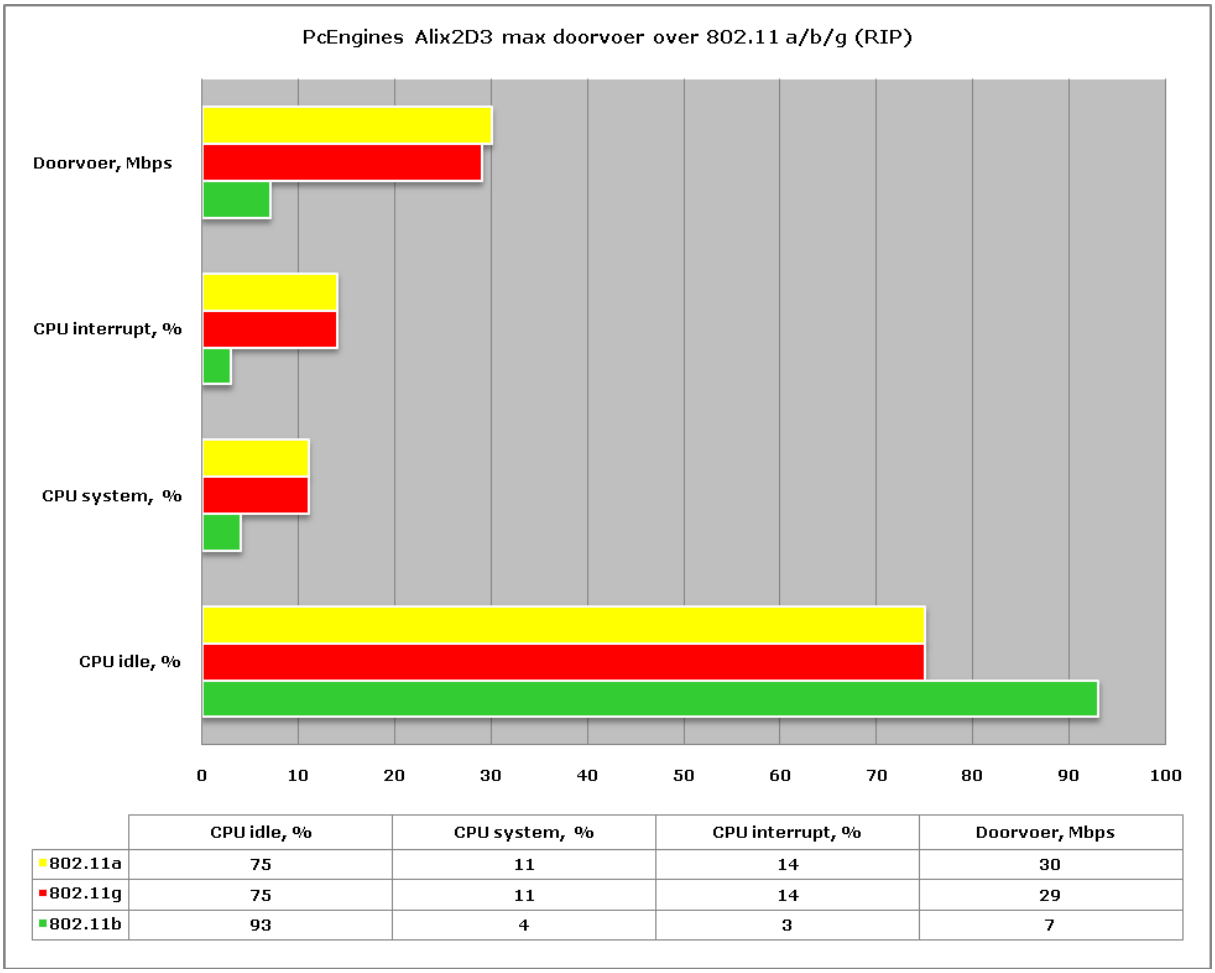
De resultaten die in de grafiek hierboven staan weergegeven tonen de maximale doorvoersnelheid, in Megabits per seconde, die is behaald met 802.11a/b/g WiFi-standaard in een bridge-configuratie. Daarnaast wordt de bijbehorende systeembelasting weergegeven die is genoteerd tijdens de verschillende Iperf dataoverdrachten.
De resultaten laten een maximale doorvoersnelheid zien van respectievelijk 29, 29 en 7 Mbit/s. De totale systeembelasting voor de 802.11a- en g-standaard ligt op ongeveeer 20%. Het gebruik van WiFi,met name de 802.11a/g –standaard, laat op het Alix2D3-systeembord eveneens een hoger percentage zien aan CPU-cycles dat is toebedeeld aan “system”, in vergelijking met de datadoorvoer over ethernet alleen.

3.3.5. Prestaties over 802.11a/b/g, RIP

Naast de bridgeconfiguratie, zijn de prestaties van het Alix2D3-systeembord ook gemeten bij het routeren van het dataverkeer over WiFi. Hierbij zijn in dezelfde testopstelling nogmaals de doorvoersnelheid en systeembelasting bij het gebruik van de 802.11a/b/g standaarden gemeten.



Alix2D3, 802.11a/b/g, routed –opstelling



	CPU idle, %	CPU system, %	CPU interrupt, %	Doorvoer, Mbps
802.11a	75	11	14	30
802.11g	75	11	14	29
802.11b	93	4	3	7

Alix2D3, 802.11a/b/g, RIP –resultaten

De resultaten in de grafiek hierboven laten zien dat bij het routeren van het dataverkeer over WiFi, de systeembelasting enigszins toeneemt ten opzichte van de bridgeconfiguratie. De doorvoersnelheid is hierbij gelijk aan de behaalde snelheden binnen de bridge-opstelling.

3.4 Ubiquiti NanoStation 5

De NanoStation is een “out-of-the-box” commerciële WiFi-toepassing die is gericht op het opzetten van outdoor point-to-point en point-to-multi-point verbindingen. De oplossing is ontwikkeld door de fabrikant Ubiquiti Networks.

http://www.ubnt.com/products/nano.php

Functioneel gezien komt het ontwerp overeen met het type Wandy-node, waarbij de radio, antenne en ethernet geïntegreerd zijn binnen een oplossing. Het systeem wordt daarbij geleverd met een besturingssysteem van de fabrikant zelf is ontwikkeld en gebaseerd is op een open source Linux distributie van Ubuntu. Over deze software staat meer beschreven in hoofdstuk 2.3, Ubiquiti AirOS.

Ubiquiti Networks, NanoStation 5

De NanoStation 5 bevat een implementatie van de 5GHz 802.11a WiFi-standaard en kan binnen de IRIS-architectuur toegepast worden als interface voor het opzetten van op 802.11a gebaseerde interlinks.

3.4.1. Systeemspecificaties

De onderstaande specificaties gelden voor het de NanoStation 5 zoals deze zijn gebruikt gedurende het IRIS project. Het is mogelijk dat specificaties van soortgelijke types enigszins afwijken.

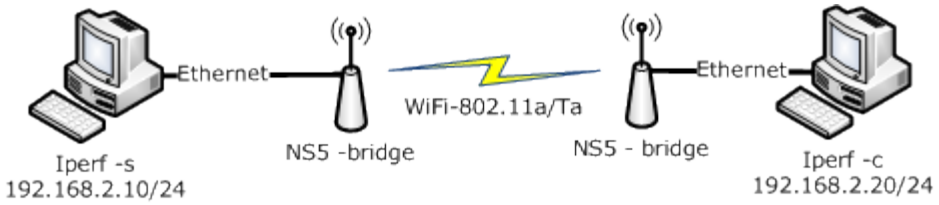
- Atheros AR2315 SOC, MIPS 4KC, 180MHzprocessor,
- 16 MB SDRAM, soldered on board,
- 4 MB flash, solderd on board
- 1 10/100 BASE-TX Ethernet ports, RJ-45,
- 1 Serial port DB9 internal,
- 12V, 1A (12 Watts) power supply (24V max),
- Passive Power over Ethernet (Non standard 802.3af),
- 14 dBi Integrated antenna array (5GHz),
- External RP-SMA.

De volledige datasheet van de NanoStation 5 kan worden gevonden op de website van Ubiquiti Networks:

http://www.ubnt.com/downloads/ns5_datasheet.pdf

3.4.2 Prestaties over 802.11a / 802.11Ta

Voor de NanoStation is het van belang de maximale doorvoersnelheid te meten bij het gebruikt van de 802.11a en 802.11Ta (turbo) standaard. De prestaties hiervan zijn getest in de volgend opstelling.



Ubiquiti NanoStation 5, testopstelling

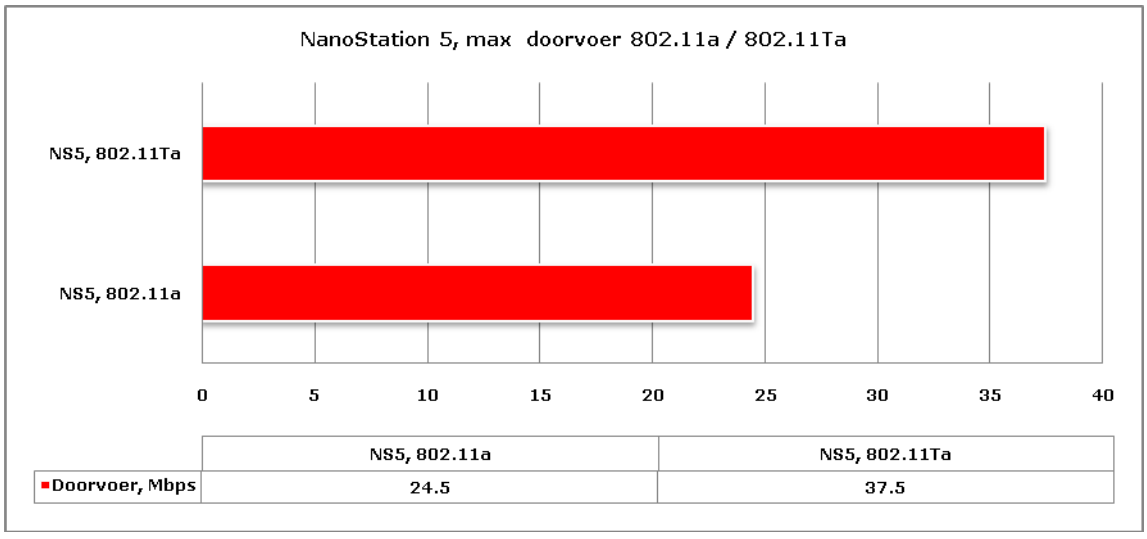
De NanoStations zijn in de testopstelling hierboven geconfigureerd als bridge. Deze configuratie zal ook toegepast worden binnen de opzet van het IRIS-prototype.
Beide NanoStations zijn via ethernet verbonden met de systemen waarop de iperf cliënt server verbindingen worden opgezet. De NanoStations zelf zijn onderling verbonden met WiFi, in een hostap / cliënt configuratie. In dezelfde opstelling zijn achtereenvolgens de doorvoerensnelheden gemeten van de 802.11a en 802.11Turbo-a standaard. Hierbij is zijn de volgende kanalen gebruikt (Amerikaanse indeling)

802.11a , channel 165

802.11Ta, channel 160

De Iperf metingen zijn uitgevoerd met de volgende opties:

`TCP # iperf –c IP-adres –t 30  /  iperf –s`

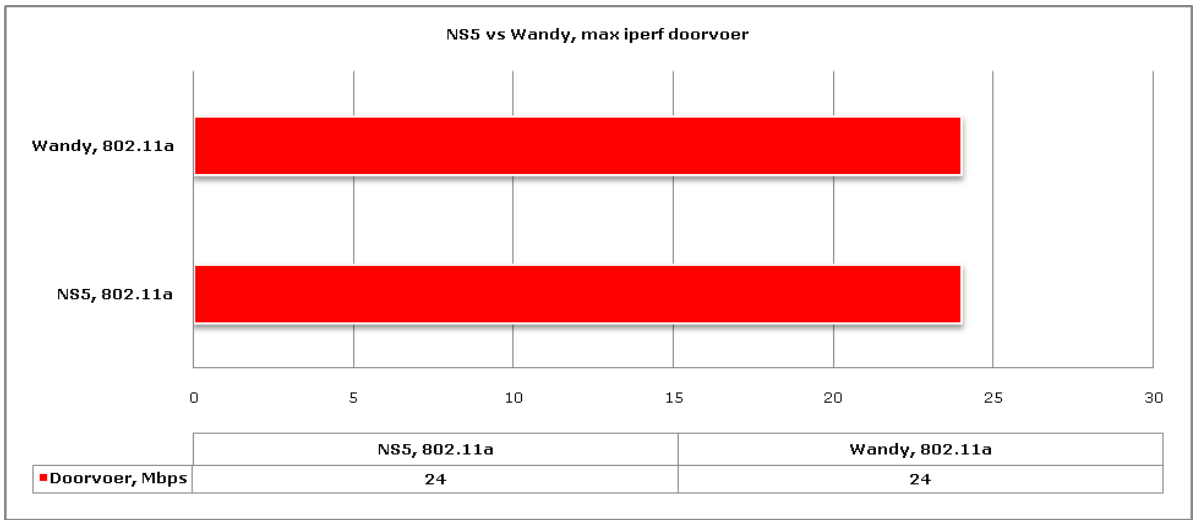


Ubiquiti NanoStation 5, testresultaten

De resultaten die in de grafiek hierboven staan weergegeven tonen de maximale doorvoersnelheid, in Megabits per seconde, die is behaald met 802.11a en 802.11Ta – standaard. De snelheden die zijn behaald liggen respectievelijk rond 24 en 37 megabits per seconde.

3.4.2. NanoStation 5 vs Wandy 5XL

In de volgende grafiek zijn de doorvoerresultaten van de van NanoStation 5 vergeleken met die van de het Wandy prototype, waar destijds in het 802.11a –project mee getest. De testopstellingen waarbinnen de prestaties zijn gemeten komen met elkaar overeen.



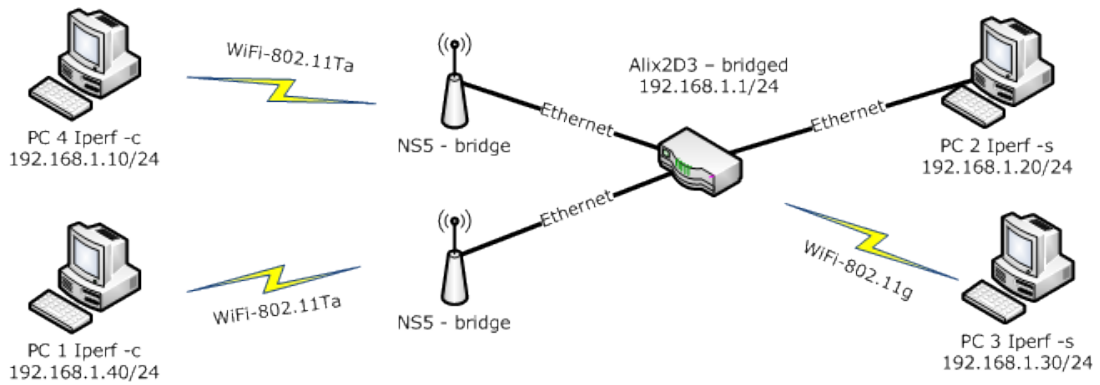
Ubiquiti NanoStation 5, testresultaten

De resultaten die in de grafiek hierboven staan weergegeven tonen de maximale doorvoersnelheid, in Megabits per seconde, die is behaald met 802.11a en standaard. De snelheden die zijn behaald liggen beide oplossingen rond 24 megabits per seconde, daarmee kan worden gesteld dat de maximale doorvoersnelheid voor de NanoStation 5 en de Wandy gelijk liggen.

3.5 IRIS-opstellingen

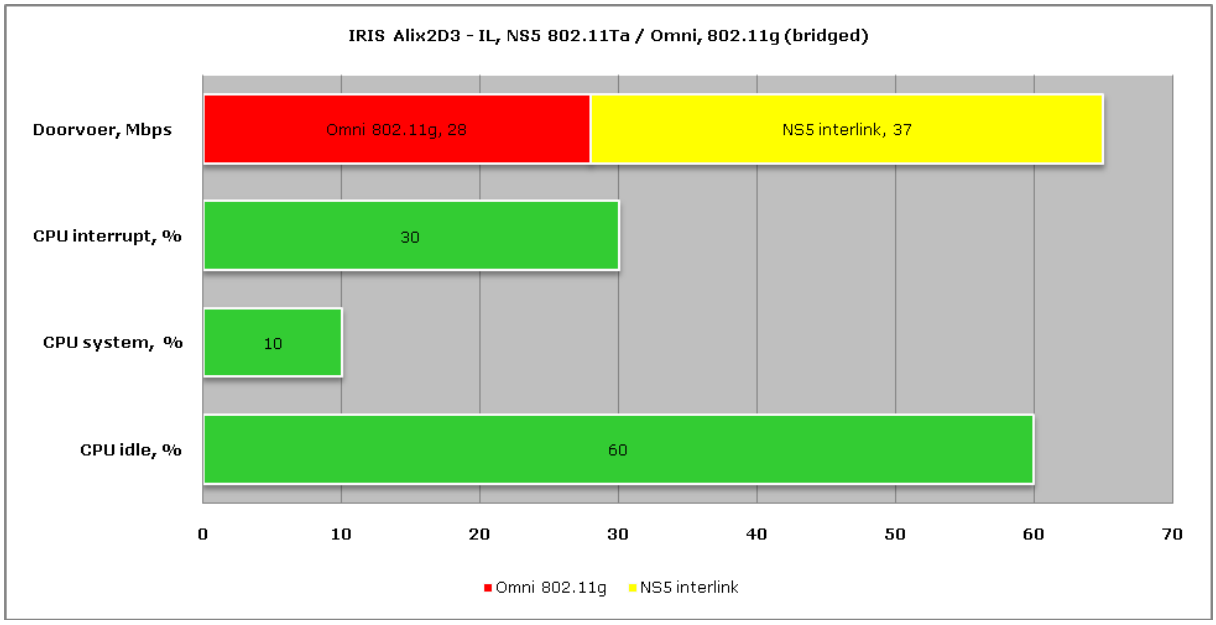
Bij het uitvoeren van de voorgaande metingen is steeds uitgegaan van een enkele Iperf datastroom per meetmoment. Omdat in de praktijk normaal gesproken meerdere dataoverdrachten simultaan worden uitgevoerd, is de onderstaande opstelling gebouwd op dit te testen. De node-hardware is daarbij geconfigureerd binnen een initiële IRIS-opzet. In deze opzet zijn voor zowel de Soekris net4801 als het Alix2D3-systeembord de prestaties gemeten.

3.5.1. NanoStations met Alix2D3



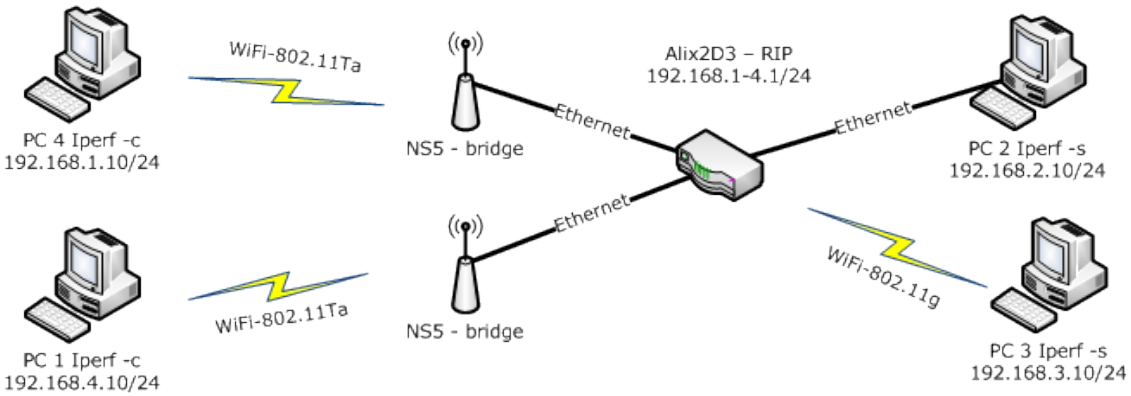
Testopstelling, initiële IRIS-opzet. Alix2D3 basis-node. Bridged

In deze opstelling zijn voor het Alix2D3-systeembord de prestaties als basis-node binnen de IRIS-opzet gemeten in een “bridged” configuratie. In de testopstelling zijn de twee NanoStations via ethernet op de basis-node aangesloten.
Deze zijn beiden geconfigureerd als toegangspunt, waarbij gebruik is gemaakt van de 802.11Ta-standaard. Verder is een Iperf hostsysteem direct aan basis-node gekoppeld via ethernet, en is een tweede hostsysteem via WiFi-aangesloten. Om een dubbele iperf datastroom op te zetten is vanaf de twee Iperf cliëntsystemen, die beiden via WiFi zijn aangesloten op een van de NanoStation toegangspunten, gelijktijdig een verbinding opgezet naar de Iperf serversystemen.



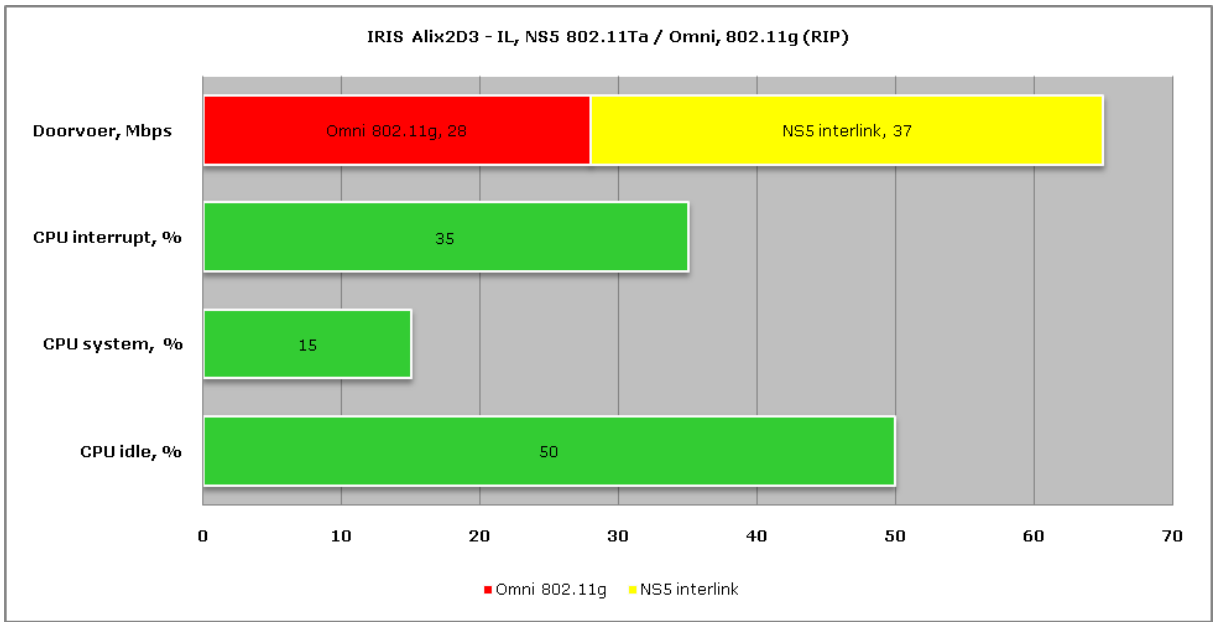
Alix2D3, initiële IRIS-opzet, bridged –resultaten

Vooraf aan de test kon worden bepaald dat de basis-node via de interlinks, een datastroom van ongeveer 37 Mbit/s en een tweede datastroom van ongeveer 30 Mbit/s moest gaan verwerken. Bij deze aanname is uitgegaan van de resultaten die zijn behaald bij het meten van de individuele prestaties.
Het Alix2D3-systeembord bleek in staat beide datastromen, met 28 en 37 megabit per seconde, op volledige snelheid door te kunnen sturen bij een totale systeembelasting van ongeveer 40%.



Testopstelling, initiële IRIS-opzet. Alix2D3 basis-node, RIP

In deze opstelling zijn voor het Alix2D3-systeembord de prestaties als basis-node binnen de IRIS-opzet gemeten in een routed opstelling. In de testopstelling zijn de twee NanoStations via ethernet op de basis-node aangesloten.
Deze zijn beiden geconfigureerd als toegangspunt, waarbij gebruik is gemaakt van de 802.11Ta-standaard. De aangesloten Iperf-systemen zijn allen geconfigureerd binnen een eigen IP-subnet.
Verder is een Iperf hostsysteem direct aan basis-node gekoppeld via ethernet, en is een tweede hostsysteem via WiFi-aangesloten. Om een dubbele iperf datastroom op te zetten is vanaf de twee Iperf cliëntsystemen, die beiden via WiFi zijn aangesloten op een van de NanoStation toegangspunten, gelijktijdig een verbinding opgezet naar de Iperf serversystemen.

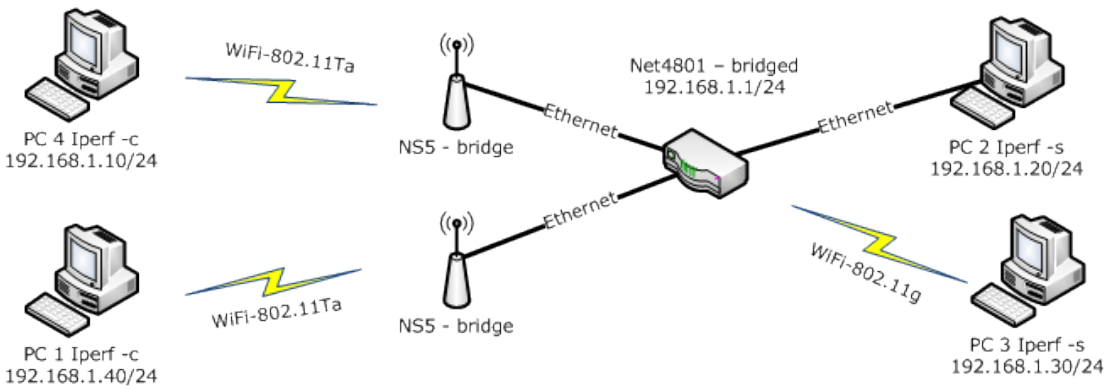


Alix2D3, initiële IRIS-opzet, RIP –resultaten

In de routed-opstelling zijn de resultaten voor het Alix2D3-systeembord vergelijkbaar met de behaalde resultaten in de bridgeconfiguratie. De doorvoersnelheid bleef onveranderd met een gecombineerde doorvoer van 65 megabit per seconde bij een totale systeembelasting van ongeveer 50%.

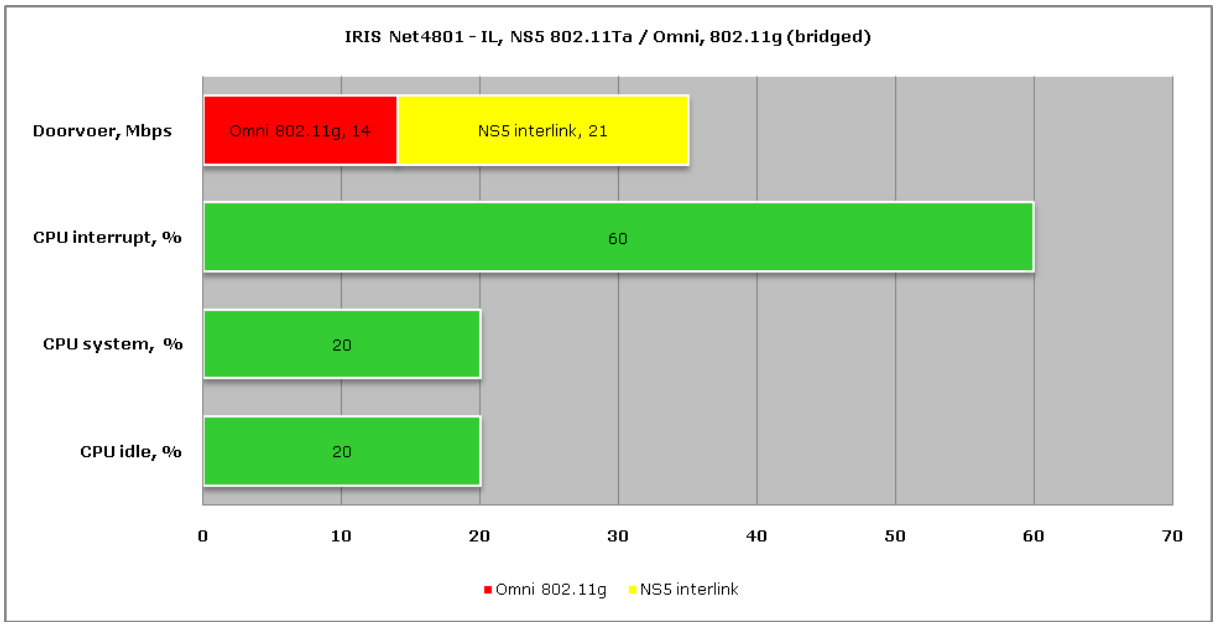
3.5.2. NanoStation met Net4801

IRIS-opstelling met net4801



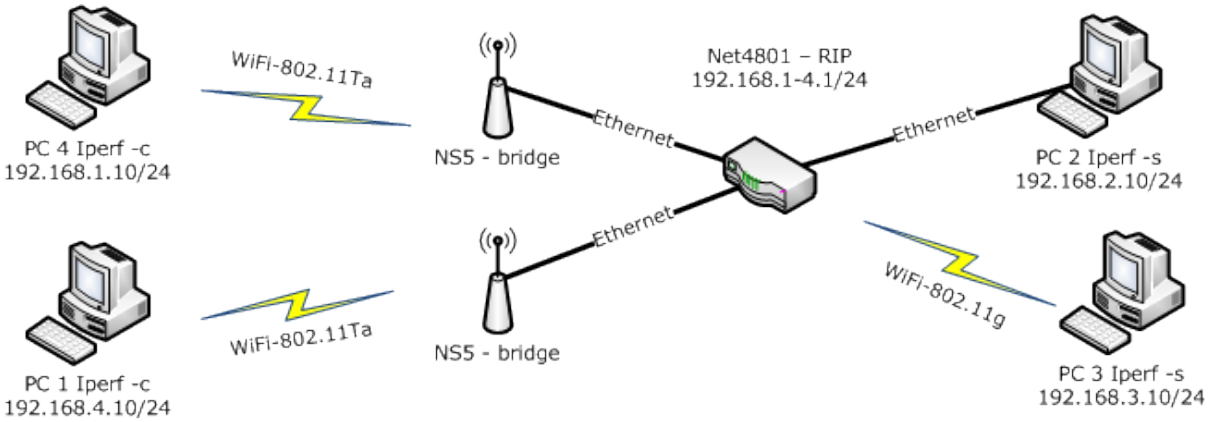
Testopstelling, initiële IRIS-opzet. Net4801 basis-node, bridged

In deze opstelling zijn voor het soekris net4801-systeembord de prestaties als basis-node binnen de IRIS-opzet gemeten in een “bridged” configuratie. In de testopstelling zijn de twee NanoStations via ethernet op de basis-node aangesloten.
Deze zijn beiden geconfigureerd als toegangspunt, waarbij gebruik is gemaakt van de 802.11Ta-standaard. Verder is een Iperf hostsysteem direct aan basis-node gekoppeld via ethernet, en is een tweede hostsysteem via WiFi-aangesloten. Om een dubbele iperf datastroom op te zetten is vanaf de twee Iperf cliëntsystemen, die beiden via WiFi zijn aangesloten op een van de NanoStation toegangspunten, gelijktijdig een verbinding opgezet naar de Iperf serversystemen.



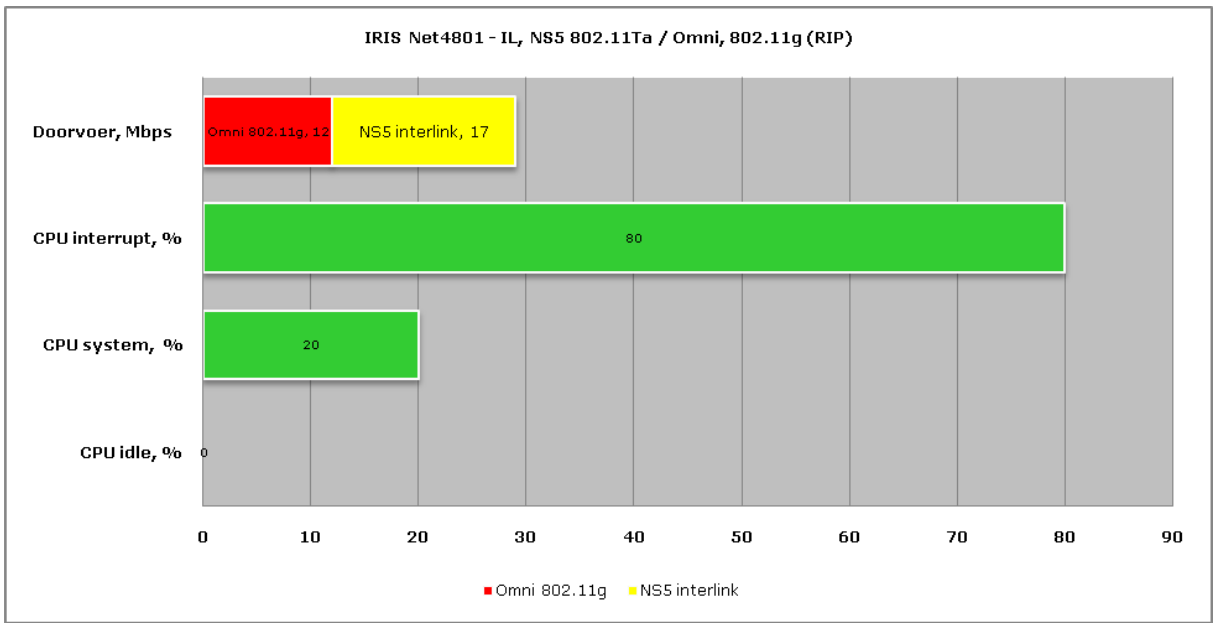
Alix2D3, initiële IRIS-opzet, bridged –resultaten

Vooraf aan de test kon eveneens worden bepaald dat de basis-node via de interlinks, een datastroom van ongeveer 37 Mbit/s en een tweede datastroom van ongeveer 30 Mbit/s moest gaan verwerken. Bij deze aanname is uitgegaan van de resultaten die zijn behaald bij het meten van de individuele prestaties.
de doorvoersnelheid die met beide datastromen werd behaald bij ligt op ongeveer 14 en 21 megabit per seconde. De totale doorvoersnelheid ligt hier met 35 Mbit/s een stuk lager dan de 65 Mbit/s die is behaald met het Alix2D3-systeembord.
Wat opviel is dat de systeembelasting was bij de dataoverdrachten niet maximaal was. Bij het monitoren van de belasting leken de twee datastromen onderling voor oponthoud in het dataverkeer te veroorzaken. De “interrupt” aanvragen versus de “system” en “idle” -percentages waren bij het monitoren dan ook niet consistent en varieerde onderling sterk.



Testopstelling, initiële IRIS-opzet. Net4801 basis-node, RIP

In deze opstelling zijn voor het Net4801-systeembord de prestaties als basis-node binnen de IRIS-opzet gemeten in een routed opstelling. In de testopstelling zijn de twee NanoStations via ethernet op de basis-node aangesloten.
Deze zijn beiden geconfigureerd als toegangspunt, waarbij gebruik is gemaakt van de 802.11Ta-standaard. De aangesloten Iperf-systemen zijn allen geconfigureerd binnen een eigen IP-subnet.
Verder is een Iperf hostsysteem direct aan basis-node gekoppeld via ethernet, en is een tweede hostsysteem via WiFi-aangesloten. Om een dubbele iperf datastroom op te zetten is vanaf de twee Iperf cliëntsystemen, die beiden via WiFi zijn aangesloten op een van de NanoStation toegangspunten, gelijktijdig een verbinding opgezet naar de Iperf serversystemen.



Alix2D3, initiële IRIS-opzet, RIP –resultaten

De doorvoersnelheid die met beide datastromen werd behaald bij ligt op ongeveer 12 en 17 megabit per seconde. Waarbij totale doorvoersnelheid lager ligt dan in de bridged – configuratie. Bij het routeren van het dataverkeer was systeembelasting echter wel maximaal, maar kon bij het monitoren van de belasting hetzelfde gedrag waargenomen worden; de twee datastromen lijken onderling voor oponthoud in het dataverkeer te veroorzaken. De “interrupt” aanvragen versus de “system” en “idle” -percentages waren bij het monitoren dan ook niet consistent en varieerde onderling sterk.

4 Nodefabriek

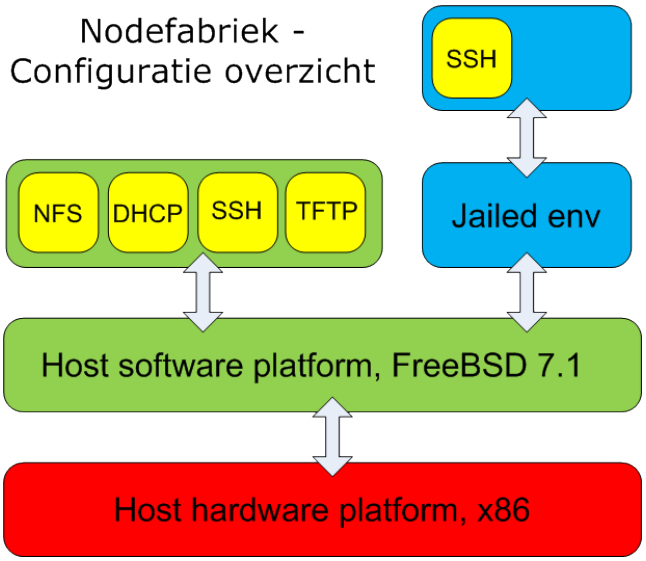
4.1 Introductie

Binnen dit hoofdstuk wordt de installatie, configuratie en het beheer van de nieuwe Wireless Leiden nodefabriek beschreven. Het concept voor een nodefabriek is binnen de organisatie al geruime tijd toegepast en dient daarbij primair voor het automatiseren en standaardiseren van nodeinstallaties. De voorgaande fabrieken zijn op verschillende manieren gebruikt voor het produceren van FreeBSD-systeeminstallaties, die bij uitstek geschikt zijn voor een ”embedded” software / hardwareconfiguratie. De nieuwe nodefabriek heeft als doel een platform te zijn voor het ontwikkelen en beheren van FreeBSD 7.x nodeinstallaties, voor zowel de huidige als toekomstige configuraties.

Het hoofdstuk start met het beschrijven van de basis-installatiestappen van een hostsysteem. Het bevat vereisten en richtlijnen voor het gebruik van de te installeren hardware en software. De configuratie van het hostsysteem wordt daarna verder uitgebreid met de installatie van een jail-omgeving voor het uitvoeren van geïsoleerde systeemtaken. Deze omgeving kan onder andere gebruikt worden bij het compileren van packages uit de FreeBSD ports tree; mogelijke conflicten met afhankelijkheden op het hostsysteem kunnen zo worden voorkomen. Daarnaast zou de omgeving bijvoorbeeld gebruikt kunnen worden bij het compileren van een kernel uit een afwijkende FreeBSD-broncode

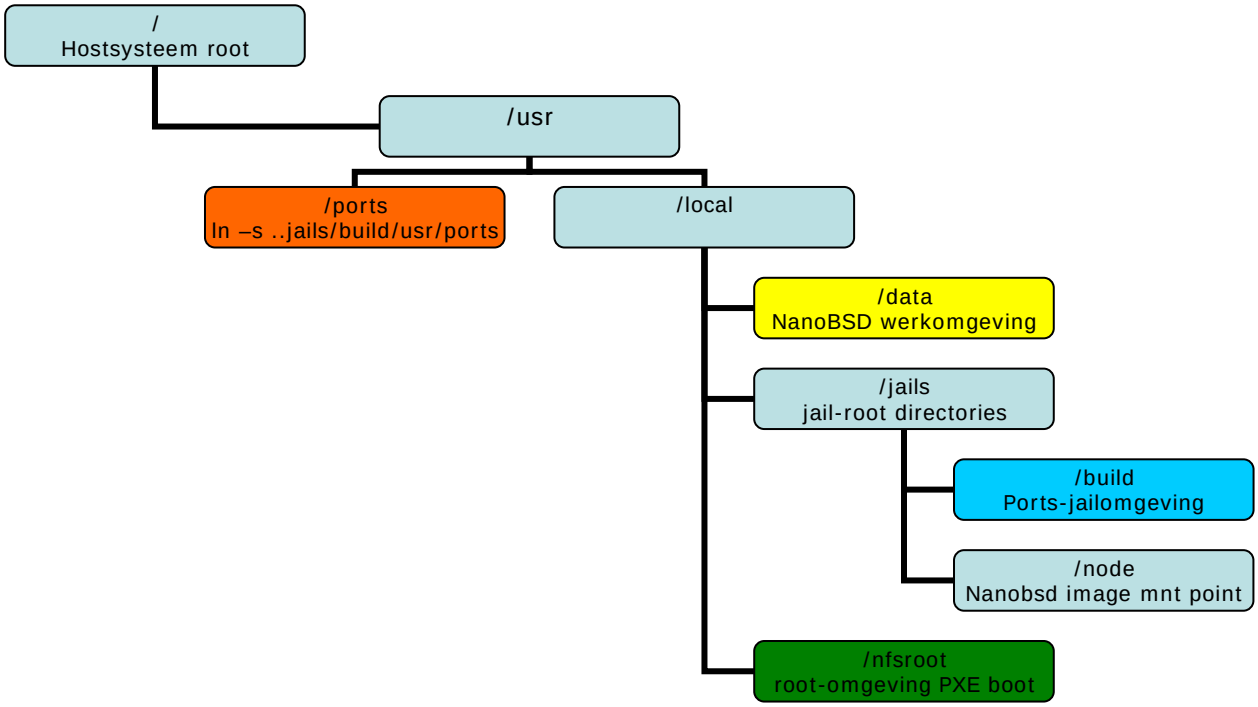
De configuratie de van nodefabriek kan verder worden onderverdeel in componenten die het uitrollen van node installatie-images faciliteren, en componenten die dienen ter ontwikkeling en beheer van deze installaties. Voor het ontwikkelen en produceren van nodeinstallaties wordt gebruik gemaakt van NanoBSD. Dit hoofdstuk gaat in op de uiteindelijke configuratie en het beheer van de NanoBSD werkomgeving. Een meer gedetailleerde beschrijving over de werking en principes van NanoBSD kan worden gevonden in Hoofdstuk twee van dit document. De overige paragrafen gaan verder in op de configuratie en het gebruik van de verschillende systeemonderdelen die nodig zijn ter ondersteuning van het installatie en of beheersproces.

Het installeren, configureren en updaten van nodeinstallaties kan worden uitgevoerd over het netwerk of direct op de nodefabriek zelf. Hieronder staat een configuratieoverzicht van de nodefabriek waar de verschillende systeemcomponenten zijn weergegeven.



4.1.1. Nodefabriek directory-overzicht.

Het onderstaande overzicht is aan de introductie van dit hoofdstuk toegevoegd om vooraf meer inzicht te geven in de structuur van de verschillende componenten die voor de nodefabriek geconfigureerd moeten worden.



Directory-overzicht hostsysteem.

Het bovenstaande overzicht toont een weergave van de omgeving waarbinnen de primaire componenten van de nodefabriek geconfigureerd worden. Het overzicht is onderdeel van het totale directory-overzicht in paragraaf 4.3.7 waar ook de onderliggende schema's in zijn opgenomen.

De componenten die in het directory-overzicht worden weergegeven zijn de NanoBSD werkomgeving, NFS-root omgeving en Ports-jail omgeving. De configuratie van de verschillende onderdelen wordt behandeld de volgende paragrafen:

- NanoBSD-werkomgeving, paragraaf 4.3.6
- NFS/PXE-omgeving, paragraaf 4.3.2 t/m 4.3.5
- Ports-jail-omgeving, paragraaf 4.2.3 en 4.3.1

4.2 Installatie nodefabriek

4.2.1. Installatie van het hostsysteem.

Voordat gestart kan worden met het configureren van de nodefabriek dient eerst een hostsysteem geïnstalleerd te worden. Voor de installatie is een FreeBSD 7.1 i386 release cd-rom gebruikt. De volgende minimale systeemeisen worden binnen het FreeBSD handboek gesteld aan de hardware van een hostsysteem:

http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/install-hardware.html

Omdat de nodefabriek niet alleen gebruikt wordt voor het uitrollen van nodeinstallaties, maar ook als ontwikkelomgeving, is het wenselijk zwaardere eisen de stellen aan de processor snelheid en de capaciteit van het werkgeheugen. Als aanbeveling zou het beste gewerkt kunnen worden met onderstaande specificaties als minima

- 1GHz+ x86 processor,
- 512MB RAM,
- 40GB hardeschijf

Indien een 64bits processorarchitectuur wordt gebruikt, is het belangrijk te kiezen voor een 32bits / i386 installatiebron. Dit voorkomt mogelijke problemen bij het kruis compileren naar een ander platform, daar de huidige node-installaties zijn ontwikkeld zijn doelsystemen met een 32 bits/x86 processor.
Ter ondersteuning van node-installaties over het netwerk kan het handig zijn een tweetal netwerkkaarten in het hostsysteem te installeren, dit in verband met het aansluiten van de nodefabriek op een bestaand lokaal netwerk.
De installatie van het hostsysteem kan verder naar eigen voorkeur worden uitgevoerd, maar het is van belang rekening te houden met enkele punten:

Fdisk: Bij het indelen van de hardeschijf (aanmaken van de slices) dient rekening gehouden te worden met het volume van de aangemaakte slices. Ter ondersteuning van de ontwikkeling van NanoBSD images, is voldoende ruimte nodig voor de opslag van de bijbehorende installatieomgeving. Een volledige NanoBSD installatieomgeving neemt ongeveer twee gigabyte aan ruimte in beslag.

- Kies voor een express FreeBSD installatie,
- Kies voor een developer distributie zonder X,

Het installeren van de onderstaande FreeBSD-ports is optioneel, maar het kan handig zijn bij het gebruik van de nodefabriek over deze pakketten te beschikken.

- Mc filemanager in /usr/ports/misc/mc
- BASH shell-omgeving in /usr/ports/shells/bash

4.2.2. Configuratie van het hostsysteem.

Hoewel de configuratie achteraf in principe naar wens kan worden aangepast, moeten een aantal aanpassingen worden gemaakt om het hostsysteem op het gebruik van een jail-omgeving voor te bereiden.
Netwerk-deamons die zijn geïnstalleerd op het hostsysteem dienen zo geconfigureerd te worden dat ze alleen naar service verzoeken luisteren van een specifiek IP-adres.

- Maak een basis-rc-configuratie aan voor het host systeem. Hieronder staat een voorbeeld dat kan worden aangepast aan de gewenste situatie

```
# vi /etc/rc.conf

hostname="host.example.org"
ifconfig_bge0="inet 192.168.2.1  netmask 255.255.255.0"
defaultrouter="192.168.2.254"
gateway_enable="YES"
sshd_enable="YES"
inetd_enable="YES"
inetd_flags="-wW -a 192.168.2.1"
```

Hoewel het voorbeeld naar wens aangepast kan worden, moet de regel `inetd_flags="-wW -a 192.168.2.1"` worden toegevoegd indien een of meerdere netwerkdiensten worden opgestart via de “inet” daemon. Dit zorgt er automatisch voor dat deze services alleen luisteren naar het IP-adres van het hostsysteem.

- Indien de SSH daemon niet met inetd wordt opgestart, dient de configuratie hiervan ook aangepast te worden.

```
# vi /etc/ssh/sshd_config

......
ListenAddress 192.168.2.1
......
```

- Het is belangrijk dat een internet verbinding op het hostsysteem beschikbaar is, en een bijbehorende nameserver is aangemaakt in /etc/resolv.conf

```
# echo nameserver 192.168.2.254 > /etc/resolv.conf
```

- Voeg de volgende beveiligingsoptie aan de kernel-configuratie van het hostsysteem toe, zodat vanuit een jailomgeving ping toegelaten wordt.

```
# echo 'security.jail.allow_raw_sockets=1' >> /etc/sysctl.conf
```

- Herstart het hostsysteem en controleer of de ingestelde jail-beveiligingsopties in de kernel overeenkomen met onderstaande opties.

```
# sysctl -a | grep ‘jail’

security.jail.jailed: 0
security.jail.mount_allowed: 0
security.jail.chflags_allowed: 0
security.jail.allow_raw_sockets: 1
security.jail.enforce_statfs: 2
security.jail.sysvipc_allowed: 0
security.jail.socket_unixiproute_only: 1
security.jail.set_hostname_allowed: 1
```

4.2.3. Installatie van de jailomgeving.

Zoals in de introductie staat beschreven wordt op het hostsysteem een jailomgeving geïnstalleerd. De jailomgeving kan gebruikt worden voor het uitvoeren van verschillende systeemtaken, die bij voorkeur geïsoleerd van het hostsysteem uitgevoerd moeten worden. Een voorbeeld zou kunnen zijn het compileren van een kernel vanuit een afwijkende FreeBSD source tree. Binnen de nodefabriek zal de jail voornamelijk gebruikt worden als omgeving voor het compileren van ports / packages. In deze paragraaf wordt de installatie van de jailomgeving beschreven. Deze installatiemethode is niet specifiek voor het uiteindelijke doel van de omgeving, en kan ook worden toegepast voor toekomstige, andere jail-installaties. De configuratie en toepassing worden verder besproken in paragraaf 4.3

Het installeren van een FreeBSD-jail kan op diverse manieren worden uitgevoerd, maar in deze installatieprocedure wordt gebruikt gemaakt van sysinstall. Indien geen installatiebron voor sysinstall beschikbaar is kan een jail ook gecompileerd worden vanuit de FreeBSD-broncode. Voordat een sysinstall wordt gestart, is het handig eerst een directorystructuur op het hostsysteem aan te maken waarbinnen de jailomgeving wordt ondergebracht.

```
# mkdir /usr/local/jails /usr/local/jails/build
```

- Maak vervolgens het onderstaande script aan in de directory `/usr/local/jails`

```
# vi /usr/local/MakeJail.sh

#!/bin/sh

JAILDIR="/usr/local/jails"
JAILS=`ls ${JAILDIR}`
for jail in ${JAILS}

do
   echo $jail
   D=${JAILDIR}/${jail}
   sysinstall nonInteractive=yes \
   mediaSetCDROM distSetDeveloper installRoot=$D \
   releaseName=7.1-RELEASE installCommit
done
```

Het script installeert via sysinstall automatisch een FreeBSD-distributie voor de jail-directory aangemaakt in `/usr/local/jails`. Het is mogelijk meerdere Jails met het script te laten aanmaken door additionele Jail root directories in `/usr/local/jails` te creëren. Als bron is een FreeBSD-7.1-installatie CD-ROM gebruikt, en daar de Jail gebruikt gaat worden als “build” omgeving, is een developers-distributie geselecteerd voor de installatie. De jailomgeving beschikt op deze manier na installatie automatisch over een eigen source en ports tree. Nadat de installatie met sysinstall is afgerond kan de jail worden gedefinieerd binnen `rc.conf`

- Maak de volgende algemene jail-variabelen aan binnen `/etc/rc.conf` van het host systeem:

```
# vi /etc/rc.conf

# Algemene Jail configuratie

jail_enable="YES"
jail_set_hostname_allow="YES"
jail_list="build"
jail_devfs_enable="YES"
jail_procfs_enable="YES"
jail_devfs_ruleset="devfsrules_jail"
```

- Maak de volgende systeemspecifieke jail-variabelen aan binnen `/etc/rc.conf` van het hostsysteem. De ip-adressering kan desgewenst aangepast worden.

```
# Jail configuratie voor node build omgeving

jail_build_rootdir=/usr/local/jails/build/
jail_build_hostname="nodebuild.example.org"
jail_build_ip="192.168.2.10"
jail_build_exec_start="/bin/sh /etc/rc"
jail_build_exec_stop="/bin/sh /etc/rc.shutdown"
```

- Maak binnen `rc.conf` als laatste de IP-alias-configuratie aan voor de jailomgeving. Gebruik hiervoor de netwerkadapter die is geconfigureerd voor de fysieke toegang tot het hostsysteem. Let op dat wanneer een IP alias geconfigureerd wordt binnen hetzelfde subnet als zijn host, een netmask van 255.255.255.255 wordt gebruikt.

```
ifconfig_bge0_alias0="inet 192.168.2.10 netmask 255.255.255.255"
```

- Sla de aangepaste rc configuratie op en herstart het netwerk, of start het host systeem opnieuw op.

```
# /etc/rc.d/netif stop
# /etc/rc.d/netif start
```

- De aangemaakte jail kan vanaf nu gestart en gestopt worden vanaf de command-line van het hostsysteem.

```
# /etc/rc.d/jail start

Configuring jails:.
Starting jails: nodebuild.example.org
```

Nu de jailomgeving is geconfigureerd, moeten een aantal algemene configuratiestappen worden doorgevoerd binnen de jail-installatie. Deze stappen kunnen handmatig worden uitgevoerd, of automatisch op basis van het script dat is gebruikt bij de initiële installatie van de Jail.

- Vraag het Jail-ID-nummer van de operationele Jail omgeving op:

```
# jls

JID  IP Address      Hostname                 Path
  1  192.168.2.10    nodebuild.example.org    /usr/local/jails/build
```

- Voer het volgende commando uit om een wachtwoord voor het root-account binnen de jailomgeving in te stellen, waar het getal één staat voor het betreffende jail-ID.

```
# jexec 1 passwd root
```

- Configureer de SSH-daemon in de jailomgeving zodat deze alleen naar service verzoeken luistert van het IP-adres dat is ingesteld als alias binnen de rc.conf van het hostsysteem, en geef root de mogelijkheid om direct in te loggen via SSH.

```
# vi /usr/local/jails/build/etc/ssh/sshd_config
......
ListenAddress 192.168.2.10
......
PermitRootLogin yes
```

- Maak een start-up variabele voor de SSH daemon aan in de rc.conf van beide Jail omgevingen.

```
# echo 'sshd enable="YES"' >> /usr/jails/build/etc/rc.conf
```

- Kopieer /etc/resolv.conf vanaf het hostsysteem naar de jailomgeving.

```
# cp /etc/resolv.conf /usr/local/jails/build/etc/
```

Na het afronden van de algemene installatie en configuratiestappen kan de jailomgeving worden herstart, en kan vanaf dat moment met ssh direct een shell naar de jailomgeving worden geopend, of via het hostsysteem met het commando jexec 1 sh

4.3 Configuratie nodefabriek

Bij de configuratie van de nodefabriek worden een aantal componenten op het hostsysteem geconfigureerd ter ondersteuning van ontwikkeling, installatie en het beheer van node-installaties. De volgende componenten worden achtereenvolgens geconfigureerd.

- Jailomgeving voor compilatie van FreeBSD ports.
- NFS-root/server ter ondersteuning van node-installaties over netwerk.
- DHCP-server ter ondersteuning van node-installaties over netwerk.
- TFTP-server ter ondersteuning van node-installaties over netwerk.
- NanoBSD-werkomgeving

De configuratiestappen van de verschillende componenten gaan uit van een reeds geïnstalleerd hostsysteem met daarop een jailomgeving, zoals in de voorgaande paragrafen van hoofdstuk vier staat beschreven. De paragrafen onder hoofdstuk 4.4 gaan verder detail in op het gebruik van de geconfigureerde onderdelen

4.3.1. Configuratie van de ports-jailomgeving.

De configuratie van de ports-jail bestaat uit het inrichten van een omgeving die gebruikt kan worden voor de compilatie van pakketten uit de FreeBSD portscollectie. Deze methode heeft als grootste voordeel dat benodigde ports op een schone manier gecompileerd kunnen worden, zonder eventuele conflicten en of wijzigingen op het hostsysteem te veroorzaken.

Het is bij het gebruik van FreeBSD ports belangrijk te werken met een actuele versie van de ports-tree. Het doorvoeren van updates aan de FreeBSD ports-tree kan op verschillende manieren worden uitgevoerd. Evenals de source-tree kan de ports-tree aangemaakt of vernieuwd worden aan de hand van de FreeBSD-CVS-repositories, in combinatie met het programma csup. Zie voor mee informatie over deze methode de beschrijving in hoofdstuk 2 paragraaf 2.2 van dit document.

Naast csup kan ook het portsnap commando worden gebruikt, dit is een gemakkelijke methode om een actuele ports-collectie te verkrijgen. Voor beide methoden is echter wel een verbinding met het internet vereist.

- Open een shell naar de jailomgeving en controleer de verbinding.

```
# jexec 1 ping www.google.nl
```

Controleer, indien de ping verzoeken geen antwoord geven, of het hostsysteem verbonden is met het internet en de geconfigureerde name-servers in /etc/resolv.conf overeenkomen.

- Voer de volgende portsnap commando's achtereenvolgens uit om de nieuwste ports-tree te downloaden en te installeren.

```
# portsnap fetch && portsnap extract
```

- Maak na afloop van het updateproces de volgende directory aan binnen de FreeBSD ports-tree.

```
# mkdir /usr/ports/packages /usr/ports/packages/All
```

Bij toekomstige compilaties van pakketten uit de portscollectie, worden de gecreëerde packages en bijbehorende afhankelijke packages in de directory /usr/ports/packages/All geplaatst. Na het aanmaken van de portstructuur kan de shell naar de jailomgeving worden gesloten. Gecompileerde packages kunnen vervolgens binnen het hostsysteem beschikbaar worden gemaakt, door de portstructuur van de jailomgeving via een symlink aan het hostsysteem te verbinden.

- Verwijder eerst de bestaande ports-tree op het hostsysteem.

```
# rm –rf /usr/ports
```

- Maak op het hostsysteem vervolgens een symlink aan naar de portstructuur van de jailomgeving.

```
# ln –s /usr/local/jails/build/usr/ports /usr/ports
```

Let hierbij op dat een symlink met een jail alleen op deze manier kan worden aangemaakt. Dat wil zeggen, locaties vanuit de jailomgeving kunnen via een symlink beschikbaar worden gemaakt op het hostsysteem, maar dit is andersom niet mogelijk. Bij het verbinden van locaties tussen jails, of vanuit het hostsysteem naar een jail, moet gebruikt gemaakt worden van een nullfs koppeling.

4.3.2. Configuratie NFS-root.

Op de het hostsysteem wordt een NFS server geïnstalleerd die voor meerdere doeleinden gebruikt kan worden:

- NFS root voor netwerk installatie via PXE boot,
- NFS export van de FreeBSD ports-tree.

Voordat de NFS-server geïnstalleerd wordt, dient voor de NFS-rootdirectory op het hostsysteem als eerste een minimaal systeem geïnstalleerd te worden.
Deze configuratie wordt gebruikt bij het starten van een netwerkinstallatie via PXE.
Deze installatie moet worden uitgevoerd met sysinstall vanaf het hostsysteem.

- Open een shell naar het hostsysteem en start sysinstall

```
# /usr/sbin/sysinstall
```

- Kies in het menu voor “Custom” > “Options” > stel de optie “Install Root” in met het volgende directory pad:

```
/usr/local/nfsroot
```

- Ga terug naar het menu met “Q” > Kies in het menu voor “Distributions” > selecteer “Minimal” > kies vervolgens voor “Custom” > “Src” > selecteer “lib” en “sys” .
- Ga terug naar het hoofdmenu, selecteer de gewenste mediabron en kies voor “Commit”

Na het afronden van de minimale systeeminstallatie, kan gestart worden met de overige configuratiestappen van de NFS-root

- Kopieer een kernel vanuit het hostsysteem naar de NFS root:

```
# cp /boot/kernel/kernel /usr/local/nfsroot/boot/kernel/kernel
```

- Stel een wachtwoord in voor het root-account binnen de installatie van de NFS root:

```
# chroot /usr/local/nfsroot /bin/csh
# passwd
# exit
```

- Maak een “fstab” aan binnen de NFS root met de volgende indeling, waarbij het ip adres moet overeenkomen met wat is geconfigureerd voor de hostomgeving.

```
# vi /usr/local/nfsroot/etc/fstab
192.168.2.1:/usr/local/nfsroot      /     nfs         rw    0     0
proc                                /proc procfs      rw    0     0
```

- Pas het “ttys” bestand binnen de NFS-root aan om een console-login op de seriële poort mogelijk te maken. Overige syscon-devices kunnen worden uitgeschakeld.

```
# vi /usr/nfsroot/etc/ttys
# ttyd0     “/usr/libexec/getty std.9600”  vt100   on   secure
...
# ttyv(0-8) “/usr/libexec/getty Pc”        cons25  off  secure
```

- Kopieer de volgende bestanden naar de /etc directory van de NFS root:

```
# cp /etc/resolv.conf /usr/local/nfsroot/etc
# cp /etc/localtime /usr/local/nfsroot/etc
```

- Wijzig tenslotte het bestand “loader.rc” om het FreeBSD-menu tijdens het opstarten uit te schakelen. Markeer alle bestaande opties als commentaar “ \ ” en voeg de volgende optie aan het bestand toe, zodat het systeem zonder pauze doorstart:

```
# vi /usr/local/nfsroot/boot/loader.rc
...
autoboot 0
```

4.3.3. Configuratie, NFS-server.

Na de configuratie van de NFS-root kan deze als export worden toegevoegd aan de NFS server. Voor de NFS-server moeten de volgende onderdelen worden geconfigureerd binnen de installatie omgeving.

- Maak het bestand “exports” aan in de /etc directory van de hostomgeving, en voeg de aangemaakte NFS root als export aan het bestand toe.

```
# vi /etc/exports

/usr/local/nfsroot -maproot=0 192.168.2.21 192.168.2.22
192.168.2.23 192.168.2.24 192.168.2.25
```

De export is in dit voorbeeld toegankelijk gemaakt voor een reeks van vijf IP-adressen. Dit zijn adressen die door de DHCP-server uitgedeeld gaan worden aan cliëntsystemen die via PXE een netwerkinstallatie initialiseren. Het is in dit voorbeeld dus mogelijk om met maximaal met vijf verschillende systemen gelijktijdig een installatie te starten.

- Maak voor de NFS server de volgende toevoegingen aan het rc.conf bestand van de installatie-omgeving.

```
# vi /etc/rc.conf

nfs_server_enable=″YES″
nfs_server_flags=″-u -t -n 4″
```

- De NFS-server kan vervolgens gestart / herstart worden met de volgende commando reeks.

```
# /etc/rc.d/nfsd stop
# /etc/rc.d/mountd forcestop
# /etc/rc.d/rpcbind forcestop
# /etc/rc.d/nfsd start
```

- Ter controle kan met showmount gecontroleerd worden of de NFS root als share is geëxporteerd.

```
# showmount -e

Exports list on localhost:
/usr/local/nfsroot     192.168.2.21 192.168.2.22 192.168.2.23
192.168.2.24 192.168.2.25
```

4.3.4. Configuratie DHCP-server.

Een DHCP-server wordt op het hostsysteem geïnstalleerd zodat PXE opstart verzoeken doorgestuurd kunnen worden naar een TFTP-server. Dit maakt het mogelijk voor aangesloten cliënt systemen via het netwerk de NFS-root-installatie te starten.

- Installeer binnen de installatieomgeving de ISC-DHCP-server distributie vanuit de FreeBSD-portstree. Gebruik de standaard make-configuratie.

```
# cd /usr/ports/net/isc-dhcp3-server && make install
```

- Maak het volgende “dhcpd.conf” configuratiebestand aan in de directory /usr/local/etc

```
# vi /usr/local/etc/dhcpd.conf

ddns-update-style none;

subnet 192.168.2.0 netmask 255.255.255.0 {
  option domain-name-servers 192.168.2.254 ;
  option routers 192.168.2.254 ;
  range 192.168.2.21 192.168.2.25 ;
  next-server 192.168.2.1 ;
  option root-path "192.168.2.1:/usr/local/nfsroot" ;
  filename "/boot/pxeboot";
}
```

Deze DHCP-serverconfiguratie deelt IP-leases uit in de range 192.168.2.21-25. Dit zijn de IP-adressen die binnen de NFS-serverconfiguratie toegang hebben tot de export van de NFS root. Hoewel de configuratie aan de gewenste situatie aangepast kan worden, moet voor de “next-server” optie wel het IP adres van de TFTP server worden ingesteld. Dit dient dus overeen te komen met het IP-adres van het hostsysteem

4.3.5. Configuratie van de TFTP-server.

De TFTP-server wordt binnen de nodefabriek alleen gebruikt om PXE-verzoeken, die zijn doorgestuurd vanuit de DHCP-server, te verwerken. Door de TFTP-opstartdirectory te wijzigen naar de locatie van de NFS-root omgeving, kan via NFS een installatie een installatie worden ingeladen.

- Maak een TFTP opstartregel aan binnen inetd.conf met de volgende configuratie

```
# vi /etc/inetd.conf

tftp  dgram udp   wait  root  /usr/libexec/tftpd      tftpd -l -s
/usr/local/nfsroot
```

- Maak een opstart variabele aan voor inetd binnen rc.conf indien deze nog niet is aangemaakt voor het hostsysteem

```
# vi /etc/rc.conf

inetd_enable=″YES″
```

- Start of herstart de inet daemon om de TFTP server te “starten”.

```
# /etc/rc.d/inetd forcerestart
```

4.3.6. Configuratie werkomgeving .

De configuratie van de werkomgeving bestaat uit het inrichten van een directorystructuur waarbinnen de verschillende configuratiebestanden, die de installatie en of ontwikkeling van node-images ondersteunen, ondergebracht kunnen worden. Op deze manier wordt een uniforme bestands/directory-indeling opgesteld, wat het uitvoeren van de

uiteindelijke beheertaken eenvoudiger maakt. Het inrichten van de werkomgeving wordt in dit document uitgevoerd aan de hand van een archief bestand en bijbehorend configuratiescript. Het archief bevat de bestandstructuur inclusief de meeste-recente configuratie- en installatiebestanden.

De configuratie omvat ook een doelbeschrijving van de aangemaakte structuur en inhoud, zodat het proces indien nodig ook handmatig uitgevoerd kan worden. Vanuit de webfolder van Wireless Leiden kan de meest recente versie van het NanoBSD_ENV archiefbestand worden gedownload. Let op dat de locaties van bestanden in de webfolder kunnen wijzigen. Met het onderstaande `fetch` -commando kan de huidige versie worden opgehaald.

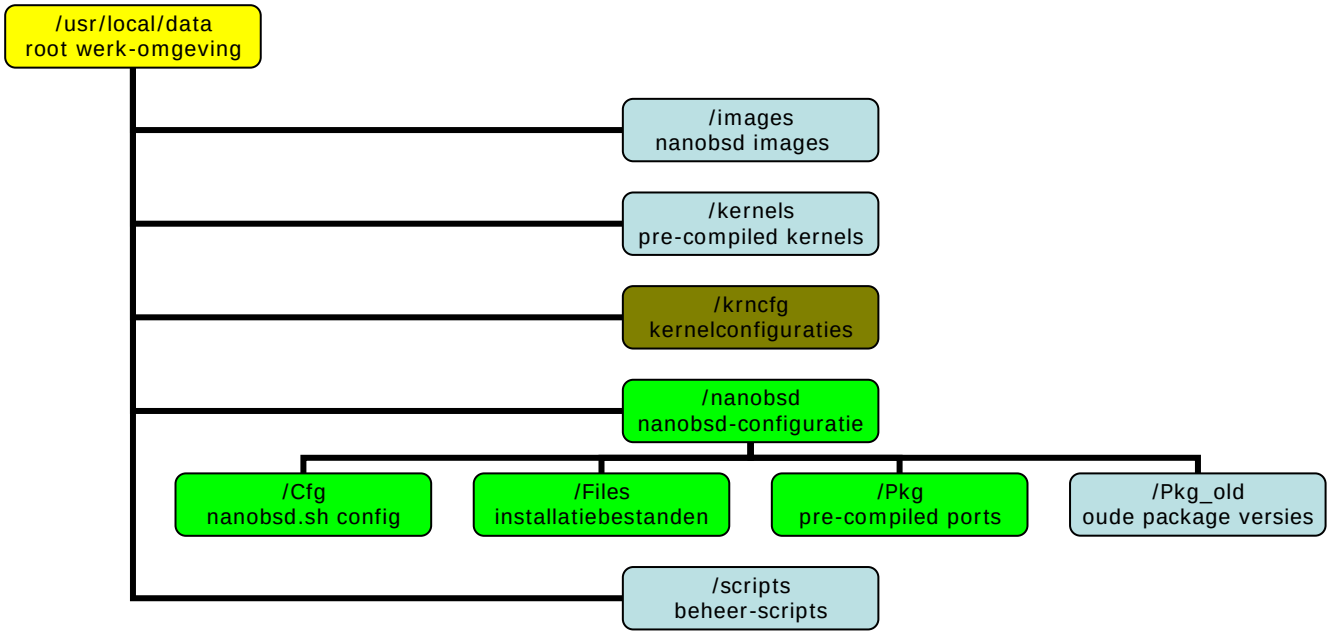
- Voer het onderstaande commando uit vanaf hostsysteem en download het laatste NanoBSD_ENV archief vanuit de Wireless Leiden WebFolder. Let op dat het huidige archief bestand ongeveer 340MB groot is

```
# cd /
# fetch http://webfolder.wirelessleiden.nl/projects/iris/NanoBSD_ENV_v05.tar.gz
```

- Pak het archiefbestand vervolgens uit in de root van het hostsysteem.

```
# tar xfz NanoBSD_ENV_v05.tar.gz
```

As de extractie van het archiefbestand zonder fouten is verlopen, kan de NanoBSD werkstructuur gevonden worden onder de directory `/usr/local/data`. Hierbinnen zijn naast de bestandsstructuren ook de huidige configuratie- en installatiebestanden uitgepakt. Hier onder staat een overzicht van de hoofd-bestandstructuur die is aangemaakt.



Directory-overzicht nodefabriek werkomgeving

De directory `/usr/local/data` kan worden aangeduid als “root” van de werkomgeving. Hierbinnen wordt `..data/images` gebruikt voor de opslag van voorgecompileerde node-images. Voorgecompileerde kernels voor de verschillende nodehardware worden opgeslagen binnen `..data/kernels`. De bijbehorende kernel-configuratiebestanden, die bij het uitvoeren van een NanoBSD-installatie ingesteld kunnen worden, staan opgeslagen in `../data/krncfg`. De directories die in het overzicht groen zijn weergegeven bevatten Alle bestanden die direct of indirect nodig zijn bij het uitvoeren van een NanoBSD-installatie. Aan de sub-directory `..nanobsd/Cfg` worden de configuratiebestanden voor het `nanobsd.sh` shell script toegevoegd. `..nanobsd/Files`

bevat de additionele installatie- en configuratiebestanden die worden toegevoegd aan de root van een NanoBSD installatie. De sub-directory ../nanobsd/Pkg bevat alle voorgecompileerde FreeBSD ports die gedurende een NanoBSD installatie geïnstalleerd worden. De scripts directory bevat verschillende ondersteunende scripts die gebruikt kunnen worden bij het beheer.

De verschillende sub-directories en configuratiebestanden zijn niet in het voorgaande directory-overzicht weergegeven, maar worden in verder detail behandeld in de hoofdstuk vier paragraaf 4.4.1 t/m 4.4.5
Hoewel de configuratiebestanden worden opgeslagen in de bovenstaande structuur, is het noodzakelijk voor een aantal onderdelen symlinks te creëren naar hun oorspronkelijke locatie, zodat ze vandaar uit gebruikt kunnen worden. Dit geldt voor de sub-directories en bestanden onder ../data/nanobsd en voor de kernel-configuratiebestanden in ../data/krncfg. Dit kan uitgevoerd worden aan de hand van het onderstaande script. Het script is bij het uitpakken van het archiefbestand toegevoegd aan de ../data/scripts directory en kan van daaruit ook gestart worden.

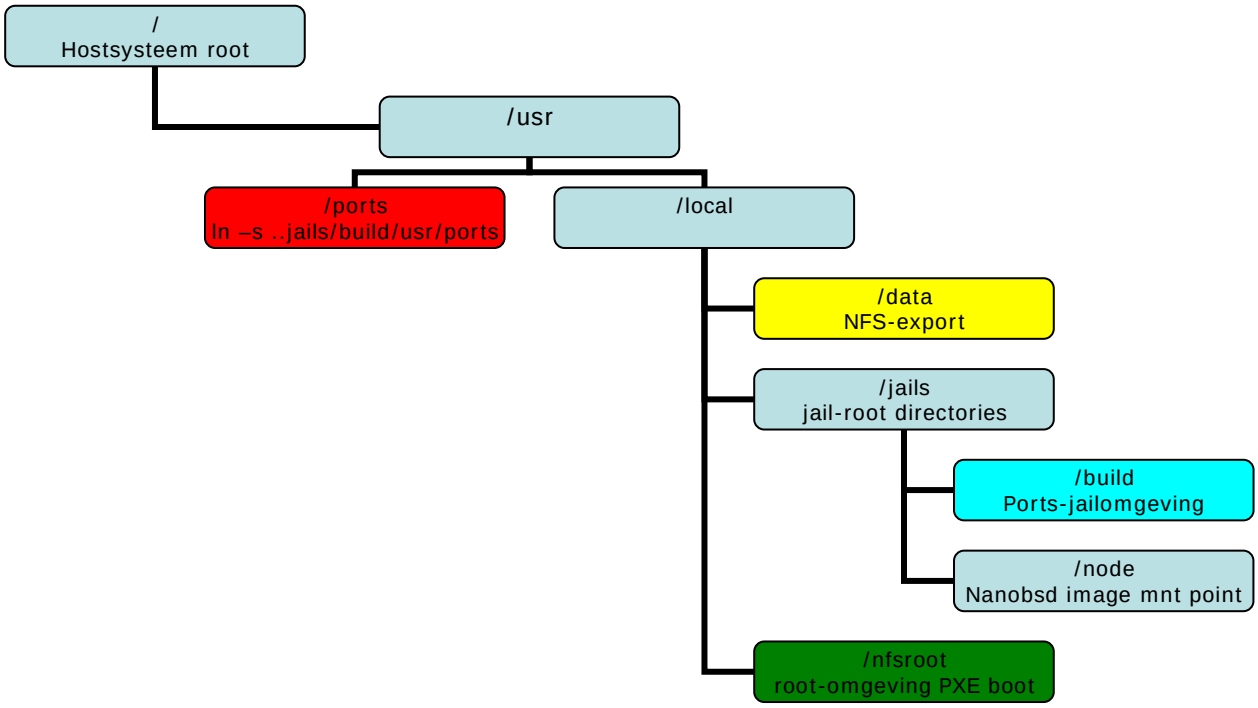
```
#!/bin/sh
# Create symlinks for the NanoBSD work environment

# Set NanoBSD work environment src/dest directory
NDIR="/usr/local/data/nanobsd"
NPATH="/usr/src/tools/tools/nanobsd"

# Set Kernel configuration src/dest directory
KDIR="/usr/local/data/krncfg"
KPATH="/usr/src/sys/i386/conf"

# Create symlinks for NanoBSD work directory
echo " "
echo "Creating symlinks for NanoBSD work directory..."
echo " "
LNDIRS=`ls ${NDIR}`
for cnt in ${LNDIRS}

do
   SRC=${NDIR}/${cnt}
   DST=${NPATH}/${cnt}
   rm -r ${DST}
   ln -s ${SRC} ${DST}
   echo "ln -s" ${SRC} ${DST}
done

# Create symlinks for kernel configuration files
echo " "
echo "Creating symlinks for kernel configuration files..."
echo " "
LNKRN=`ls ${KDIR}`
for krncfg in ${LNKRN}

do
   SRC=${KDIR}/${krncfg}
   DST=${KPATH}/${krncfg}
   ln -s ${SRC} ${DST}
   echo "ln -s" ${SRC} ${DST}
done

echo " "
echo Symlinks created succesfully.

exit 0
```

- Start het `create_symlinks.sh` script vanaf het hostsysteem.
  `# sh /usr/local/data/scripts/create_symlinks.sh`

Als het script succesvol is afgerond zijn voor de sub-directories onder `../data/nanobsd` respectievelijk symlinks gecreëerd in de standaard NanoBSD-werkomgeving in `/usr/src/tools/tools/nanobsd`. Deze manier van werken behoud de oorspronkelijke structuur waarbinnen de NanoBSD-installaties worden geproduceerd en zorgt ervoor dat in het geval de FreeBSD source-tree vernieuwd moet worden, de verschillende configuratie en installatiebestanden altijd in tact blijven.
Voor de bestaande kernel –configuratiebestanden binnen de directory `../data/krncfg`, zijn symlinks aangemaakt naar de oorspronkelijke locatie in `/usr/src/sys/i386/conf`. Hier geldt echter wel dat dit op bestandniveau is uitgevoerd. Voor toekomstige toevoegingen aan de ../data/krncfg directory dienen daarom alsnog symlinks gecreëerd te worden.

4.3.7. Configuratie-overzicht.

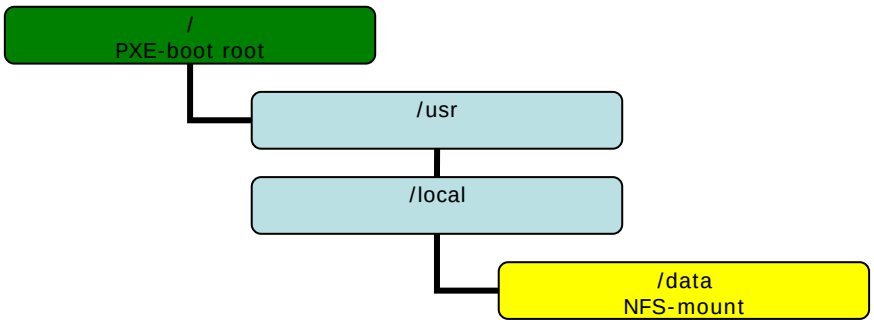
Binnen de onderstaande overzichten wordt een weergave gegeven van de verschillende geconfigureerde directories / componenten binnen de nodefabriek. De overeenkomende kleuren binnen het directory-overzicht laten de relaties tussen de verschillende componenten zien.



Directory-overzicht hostsysteem.

De componenten die in het bovenstaande directory-overzicht worden weergegeven zijn de NanoBSD werkomgeving, NFS-root omgeving en Ports-jail omgeving:

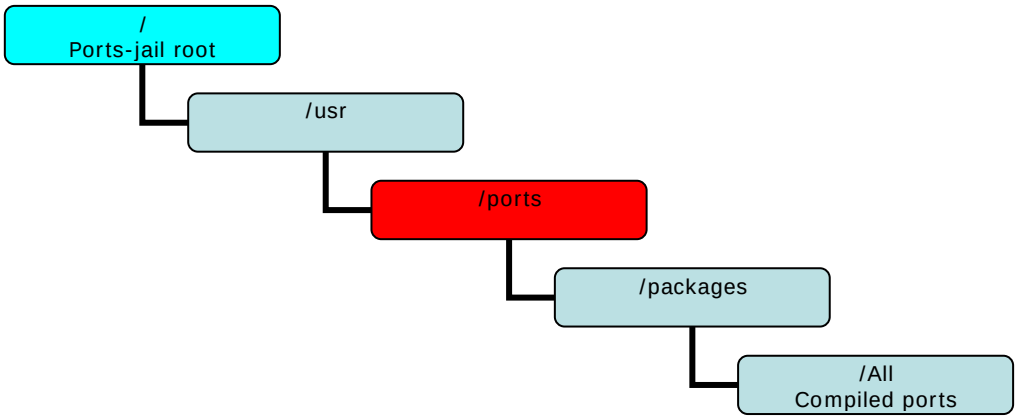
- NanoBSD-werkomgeving, Alle bestanden gerelateerd aan de node- en NanoBSD-installaties. Directory wordt geëxporteerd als NFS-share.
- NFS/PXE-omgeving, NFS-root, gebruikt als PXE-opstartomgeving .
- Ports-jail-omgeving, Ports-jail, jailomgeving die gebruikt kan worden voor het compileren van FreeBSD-ports.
- FreeBSD ports-tree, symlink verwijst naar de ports-tree in de jailomgeving.



Directory overzicht PXE-boot omgeving

Het bovenstaande overzicht geeft de omgeving weer die ingeladen wordt bij het initialiseren van een PXE-installatie.

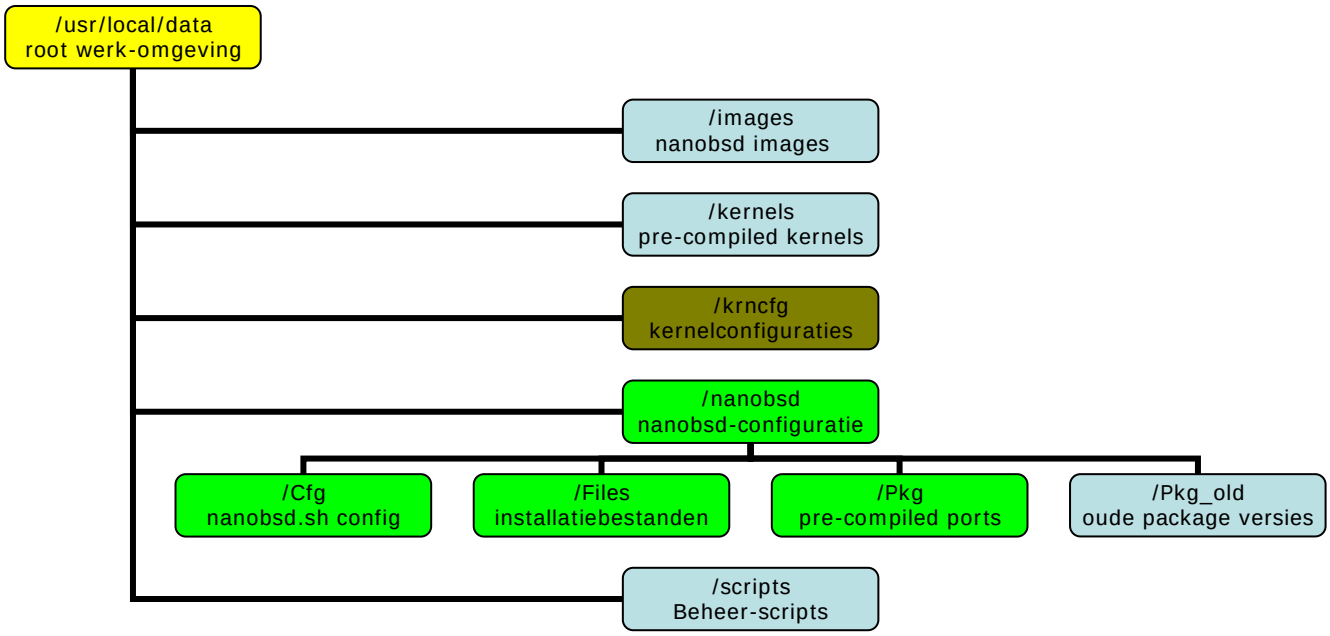
- PXE-boot, root van de PXE-installatie-omgeving. Gestart vanaf `/usr/local/nfsroot` op het hostsysteem.
- /data, NFS-mount van de `/usr/local/data` directory op het hostsysteem. Bevat alle benodigde installatie- en configuratiebestanden



Directory-overzicht ports-jail.

Het bovenstaande overzicht geeft de structuur van de ports-jail-omgeving weer.
De ports-tree op het hostsysteem is verwijderd en in plaats daarvan is een symlink aangemaakt naar de ports-tree van de jailomgeving.

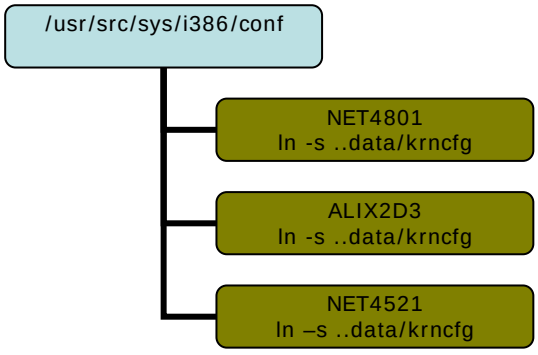
- Ports-jail-root, root van de jailomgeving. Geplaatst in de directory `/usr/local/jails/build` op het hostsysteem.
- Ports-tree, gecompileerde packages worden toegevoegd aan `/usr/ports/packages/All`



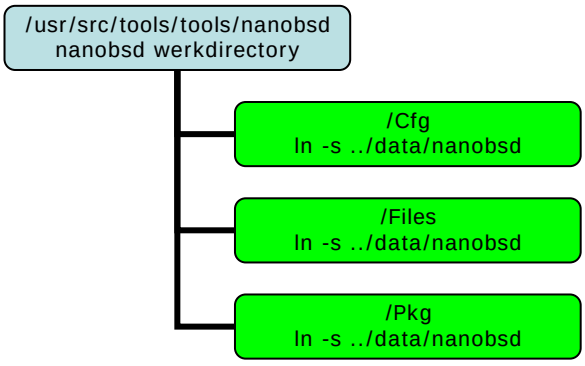
Directory-overzicht werkomgeving.

Het bovenstaande overzicht geeft de structuur van de NanoBSD-werkomgeving weer. Hierbinnen zijn alle bestanden ondergebracht die zijn gerelateerd aan de node-installatie.

- Werkomgeving /, de root van de werkstructuur waarbinnen de node-installatie- en –configuratiebestand zijn ondergebracht.
- /krncfg, bevat de kernel-configuratiebestanden voor de verschillende node-hardware.
- /nanobsd, bevat alle bestanden die gebruikt worden bij het uitvoeren van een NanoBSD-installatie .



Kernerl-configuratiedirectory.



Nanobsd werkomgeving.

- Symlinks voor de kernel-configuraties in `/usr/local/data/krncfg`

- Symlinks voor de NanoBSD installatie en configuratiedirectories

4.4 Gebruik nodefabriek

Dit hoofdstuk gaat in op het gebruik van de verschillende componenten die zijn geconfigureerd binnen de nodefabriek. Het gaat daarbij hoofdzakelijk in op het aanmaken en configureren van een NanoBSD-installatie. Het uitvoeren van node-installaties en een beschrijving van de daadwerkelijke installatie zelf staat verder uitgewerkt in hoofdstuk vijf van dit document. Deze handleiding gaat er vanuit dat de configuratie van de nodefabriek, zoals beschreven in paragraaf 4.2 en 4.3, reeds is aangemaakt.

4.4.1. NanoBSD configuratie aanmaken.

Voordat gestart kan worden met het produceren van een NanoBSD installatie moeten een aantal configuratiebestanden worden aangemaakt. Deze handleiding behandeld de verschillende stappen die zijn uitgevoerd om een NanoBSD-installatie creëren die uiteindelijk gebruikt kan worden binnen het netwerk van Wireless Leiden.
De handleiding beschrijft alleen de stappen om tot een installatie te komen. Voor een algemene beschrijving van NanoBSD en de functionaliteiten kunnen de betreffende paragrafen onder hoofdstuk 2 worden doorgelezen.

- Open een shell naar het hostsysteem en ga naar de ..nanobsd/Cfg directory in de werkomgeving

```
# cd /usr/local/data/nanobsd/Cfg
```

Binnen de ../nanobsd/Cfg directory worden de primaire NanoBSD configuratiebestanden ondergebracht. Dit zijn de configuratiebestanden die kunnen worden opgegeven bij het aanmaken van een NanoBSD-installatie. De opbouw van een NanoBSD configuratiebestand is slechts voor een gedeelte systeemspecifiek. Om het aanmaken van een NanoBSD configuratiebestand te vereenvoudigen is een template gecreëerd dat geschikt is voor de node-omgeving van Wireless Leiden. Het bestand is bij de configuratie van de werkomgeving toegevoegd aan de `../nanobsd/Cfg` directory. Indien dit niet het geval is, kan een template op basis van de onderstaande stappen ook handmatig aangemaakt worden.

4.4.2. NanoBSD configuratie-template.

De stappen voor het aanmaken van een template voor het NanoBSD configuratiebestand zijn opgenomen als bijlage in Appendix A, NanoBSD configuratie-template.

4.4.3. NanoBSD ports & packages toevoegen.

Het toevoegen van ports en packages gedurende een NanoBSD installatie kan op meerdere manieren worden uitgevoerd. Dit staat verder uitgelegd in de betreffende paragraaf van hoofdstuk 3, NanoBSD ports & packages.

Hoewel het dus mogelijk is om eigen functies aan het configuratiebestand van NanoBSD toe te voegen, en zo additionele packages te installeren, wordt bij een standaard installatieprocedure gebruik gemaakt van de `cust_pkg` functie. Deze functie is geïntegreerd binnen het `nanobsd.sh` shell script en installeert voor-gecompileerde packages die zijn toegevoegd aan de NanoBSD werkdirectory in `/usr/local/data/nanobsd/Pkg`. De werking van deze functie staat in verder detail beschreven in hoofdstuk 2 paragraaf 2.3.6. Er is een symlink aangemaakt van `/usr/src/tools/tools/nanobsd/Pkg` naar deze directory. De onderstaande lijst bevat alle FreeBSD-ports die standaard worden toegevoegd bij het uitvoeren van een NanoBSD-installatie. Alle packages moeten eenmalig recursief worden gecompileerd binnen de ports-jail en daarna worden toegevoegd aan de directory `/usr/local/data/nanobsd/Pkg`.

Het compileren van de ports en toevoegen van de packages kan handmatig worden uitgevoerd, of via het `create_packages.sh` script aan de hand van de lijst met FreeBSD-ports in het bestand `ports_list_full`, zie de bijlage in Appendix B
Het script en de lijst met ports zijn bij de configuratie van de werkomgeving toegevoegd aan de `..data/scripts` directory. Indien dit niet het geval is, kunnen de bestanden handmatig aangemaakt worden volgens de stappen die staan beschreven in de bijlage

Het bestand `ports_list_full` bevat een lijst met alle FreeBSD ports die standaard geïnstalleerd worden bij het uitvoeren van een NanoBSD-installatie. De verschillende ports zijn aan de lijst toegevoegd aan de hand van de directory waarbinnen de betreffende port zich bevind. Daarnaast kunnen in de lijst per port argumenten worden opgenomen die worden doorgevoerd aan de make-configuratie. Een argument kan achter de directory-naam worden geplaatst, gescheiden door middel van een spatie, met een maximum van drie argumenten. Zie de ports `dns/bind96` en `lang/php5` in de bijlage als voorbeeld.

Met het create_packages.sh script kunnen de ports die binnen het `port_list_full` bestand zijn gedefinieerd worden gecompileerd, en als packages toegevoegd worden aan de `..nanobsd/Pkg` directory in de werkomgeving. Als het script nog niet in de ..data/script directory aanwezig is kan het handmatig aangemaakt worden, zie de bijlage in Appendix B

Het script compileert de port aan de hand van de ingestelde ports-directory en kopieert het aangemaakte `.tbz` package bestand naar `/usr/local/data/nanobsd/Pkg`. Het script is daarbij niet versie afhankelijk zolang de betreffende ports-directories niet wijzigen. De volgende opties worden bij het compileren standaard aan het `make` commando meegegeven:

```
make package-recursive -DBATCH FORCE_PKG_REGISTER=yes
```

De optie `package-recursive` zorgt ervoor dat van een port en bijbehorende afhankelijkheden packages worden gecreëerd. Aan de hand van deze packages wordt een port geïnstalleerd binnen een NanoBSD-installatie. De `–DBATCH` optie zorgt ervoor dat een default `Makefile` bestand wordt gebruikt bij het compileren van een port en dus geen additionele configuratieschermen worden getoond.
De optie `FORCE_PKG_REGISTER=yes` wordt gebruikt om mogelijke conflicten met pakketten die binnen de ports-jail geïnstalleerd zijn te vermijden.

Bij de start van het script kan een keuze gemaakt worden of de volledig lijst met ports gecompileerd moet worden, of alleen een lijst met updates. Het aanmaken van een lijst met updates voor bestaande ports/packages wordt verder in hoofdstuk 5 behandeld, bij het updaten van geïnstalleerde ports binnen een NanoBSD-installatie.
Alle bestaande packages die binnen `..nanobsd/Pkg` zijn opgenomen en bij het compileren van nieuwe ports worden overschreven, kunnen na afloop van het script gevonden worden in de directory `/usr/local/data/nanobsd/Pkg_old`.
Van de lijst met ports in het bestand `ports_list_full`, zijn de benodigde packages bij het configureren van de werkomgeving reeds toegevoegd. Indien dit niet het geval is, of de packages opnieuw gecompileerd moeten worden, kan dit uitgevoerd worden volgens onderstaande stappen.

- Start vanaf het hostsysteem het script binnen de ports-jail. De jailomgeving en de nullfs-koppeling van de werkomgeving dienen hiervoor beschikbaar te zijn.

  ```
  # jexec 1 sh /usr/local/data/scripts/create_packages.sh
  ```

- Selecteer 1 voor het uitvoeren van een compilatie van de volledige lijst met ports. Afhankelijk van de snelheid van het hossysteem kan dit geruime tijd duren.

Aan het script is momenteel nog geen afhandeling voor foutmeldingen toegevoegd. Controleer daarom na afloop handmatig of alle benodigde ports aan de /usr/local/data/nanobsd/Pkg directory zijn toegevoegd.

4.4.4. NanoBSD-configuratiebestanden toevoegen.

Zoals in het hoofdstuk over NanoBSD staat beschreven, kan het doorvoeren van wijzigingen, of het aanmaken van een nieuw configuratiebestand gedurende een NanoBSD-installatie op diverse manieren worden uitgevoerd. Het heeft bij een klein aantal wijzigingen binnen een bestaand bestand de voorkeur hiervoor een macro aan te maken. Zie hiervoor de voorbeelden binnen de configuratie-template in paragraaf 4.4.2 Bij het doorvoeren van grote wijzigingen en of het toevoegen van een compleet nieuw configuratiebestand, heeft het de voorkeur de betreffende bestanden vooraf aan de NanoBSD-configuratiedirectories toe te voegen. Aangemaakte configuratiebestanden die op deze manier aan de systeeminstallatie worden toegevoegd zijn ondergebracht in de directory in /usr/local/data/nanobsd/Files. Hierbinnen kan een bestandstructuur worden aangemakt die bij het installatieproces naar de root van NanoBSD installatie-omgeving wordt gekopieerd. Dit geldt voor algemene node-configuratiebestanden. Node-specifieke configuraties kunnen na of gedurende een installatie gegenereerd worden via de configuratiedatabase Genesis / Exodus.

De huidige bestanden, die bij het configureren van de werkomgeving aan de ../nanobsd/Files directory zijn toegevoegd, worden in Appendix C, NanoBSD-configuratiebestanden kort inhoudelijk besproken. De functionaliteit van deze configuraties wordt verder besproken in hoofdstuk 5.
Binnen het overzicht in de bijlage wordt de bestandstructuur weergegeven van de directory /usr/local/data/nanobsd/Files.
Deze directorystructuur wordt recursief gekopieerd naar de root van een NanoBSD-systeemomgeving. Een directorystructuur wordt alleen gekopieerd als minimaal een bestand is toegevoegd aan een sub-directory. Een lege structuur wordt overgeslagen.

4.4.5. NanoBSD-installatie aanmaken.

Na het toevoegen van de verschillende installatie en configuratiebestanden, kan gestart worden met het aanmaken van een NanoBSD-installatie.

Bij het aanmaken van een NanoBSD-installatie kan onderscheid gemaakt worden tussen systeemspecifieke en generieke installatie-images. Bij systeemspecifieke installaties kunnen aanpassingen en optimalisaties zijn doorgevoerd die alleen geschikt zijn voor het beoogde doel systeem. Hoewel het handig kan zijn meerdere van deze NanoBSD images op maat te hebben, is het beheertechnisch lastiger om de installaties consistent en uniform te houden. Bij een algemene configuratiewijziging moeten de verschillende installaties individueel worden aangepast. Het heeft daarom voorkeur zoveel mogelijk gebruik te maken van een generieke-installatie voor de verschillende node types.

In de volgende stappen wordt een generieke NanoBSD-installatie gecreëerd. Voor een generieke NanoBSD-installatie wordt in het configuratiebestand geen NANO_KERNEL variabele gedefinieerd zodat automatisch een generieke-kernel gecompileerd wordt. Daarnaast worden geen verdere aanpassingen gemaakt aan het NanoBSD-configuratiebestand en worden geen node-specifieke configuratiebestanden geïnstalleerd.

De stappen in de handleiding gaan er vanuit dat de algemene configuratie voor de NanoBSD werkomgeving, beschreven in de voorgaande paragrafen, beschikbaar is.

- Open een shell het hostsysteem en ga vervolgens naar de NanoBSD werk-directory.

  ```
  # cd /usr/src/tools/tools/nanobsd
  ```

Vanuit de NanoBSD werk-directory kan direct gestart worden met het creëren van een installatie-image. Indien nog geen eerdere NanoBSD-installaties zijn aangemaakt, dient eerst een configuratiebestand te worden gecreëerd voor de beoogde installatie. Een NanoBSD configuratiebestand kan aangemaakt worden op basis van de NanoBSD configuratie-template in../nanobsd/Cfg.

- Maak een nieuw configuratiebestand aan op basis van de template.

  ```
  # cp Cfg/nanobsd.template Cfg/nanobsd.wleiden.v01
  ```

- Open het aangemaakte configuratiebestand in de vi editor

  ```
  # vi Cfg/nanobsd.wleiden.v01
  ```

- Binnen het configuratiebestand moeten de volgende “systeemspecifieke” variabelen ingesteld worden. Schakel de optie NANO_KERNEL uit.

  ```
  NANO_NAME=wleiden        # object naam in /usr/obj/nanobsd.{obj}
  NANO_SRC=/usr/src        # freebsd source tree
  #NANO_KERNEL=            # naam van het kernel configuratiebestand
  NANO_IMAGES=1            # aantal nanobsd code slices/installs
  ```

- Configureer de FlashDevice optie. Zie het bestand /usr/local/data/nanobsd/FlashDevice.sub voor mogelijke opties. In het voorbeeld wordt een Sandisk 512mb compactflash kaart geconfigureerd.

  ```
  FlashDevice sandisk 512mb
  ```

- Wijzig de naam van de last_orders() functie in no_last_orders(), zodat de node-specifieke configuratie niet naar het imagebestand wordt geschreven

Wanneer een standaardinstallatie wordt uitgevoerd hoeven de overige variabelen niet gewijzigd te worden. Sla de gemaakte wijzigingen op en sluit het configuratiebestand af.

- Start vanuit de NanoBSD werk-directory een installatie op met het volgende commando.

  ```
  # sh nanobsd.sh –c Conf/nanobsd.wleiden.v01
  ```

Het installatie- en compilatieproces kan geruime tijd duren; deels afhankelijk van de specificaties van het hostsysteem kan het gehele proces enkele uren in beslag nemen.
Na een succesvol afgeronde installatie kan de output van het proces gevonden worden in de object-directory /usr/obj/nanobsd.wleiden
Debug eventuele foutmeldingen aan de hand van de desbetreffende log bestanden.
Meer hierover staat beschreven in het software hoofdstuk over NanoBSD**.**

```
/usr/obj/nanobsd.wleiden/_.mnt: write failed, filesystem is full
```

Indien deze melding zich voordoet, betekend dit dat het bestandsysteem van het NanoBSD-image vol zit. De reden hiervoor is dat de het type flashkaart, gedefinieerd binnen het NanoBSD configuratiebestand, te klein is om de gehele NanoBSD-installatie te huisvesten. Voor de NanoBSD-installatie die voor Wireless Leiden gebruikt wordt dient rekening gehouden te worden met het volgende overzicht.

Type installatie	In gebruik	Vrije ruimte	Capaciteit %
1	289784	150575	66%
2	205443	243016	46%
3	180444	268015	40%
4	86853	361606	19%

1 = systeemomgeving met generieke kernel, met modules en met packages

2 = Systeemomgeving met net4801 kernel, met modules en met packages

3 = Systeemomgeving met net4801 kernel, geen modules en met packages

4 = Systeemomgeving met net4801 kernel, geen modules en geen packages

Het overzicht toont de omvang van de systeemomgeving in relatie tot een aantal onderdelen die in het NanoBSD-configuratiebestand ingesteld kunnen worden.
De beschikbare ruimte / capaciteit is ten opzichte van een 512 MB flashkaart. Indien een volledige installatie gecompileerd zou worden, met een generieke kernel, modules en de verschillende packages, heeft de systeemomgeving een volume van 282MB. Dit betekend dat het niet mogelijk is voor deze configuratie twee systeem slices te creëren in combinatie met 512MB flashgeheugen. Als een NanoBSD-installatie toch met deze configuratie wordt uitgevoerd, veroorzaakt dit de melding:

```
/usr/obj/nanobsd.wleiden/_.mnt: write failed, filesystem is full
```

Een optie zou zijn om een systeemspecifieke kernelconfiguratie te gebruiken, of in het NanoBSD-configuratiebestand het type flashkaart te veranderen naar 1024MB.

Binnen de object directory die voor de uitgevoerde installatie is aangemaakt zijn ook de benodigde imagebestanden weggeschreven. Hoewel de bestanden op deze locatie opgeslagen kunnen blijven, is het verstandig deze ook naar de `../data/images` directory in de werkomgeving te kopiëren. Dit zorgt ervoor dat de imagebestanden bewaard blijven, indien deze worden overschreven bij het uitvoeren van een toekomstige NanoBSD-installatie binnen dezelfde object-directory. Kopieer de installatie-images naar de `../data/images` directory indien ze bewaard moeten worden voor toekomstige node-installaties.

- Maak een sub-directory voor de imagebestanden aan in `../data/images`

  ```
  # mkdir /usr/local/data/images/generic.wleiden
  ```

- Kopieer de imagebestanden vanuit de object directory naar de aangemaakte directory in `../data/images`

  ```
  # cd /usr/obj/nanobsd.wleiden
  # cp _.disk.full _.disk.image /usr/local/data/images/generic.wleiden
  ```

- Maak voor de toegevoegde images het volgende readme-bestand aan.

  ```
  # vi /usr/local/data/images/generic.wleiden/readme

  De NanoBSD images binnen deze directory zijn
  Wireless Leiden specifiek.

  NanoBSD-installatie gecompileerd met:

  Configuratiebestand: nanobsd.wleiden.v01
  NanoBSD image-info : 512MB, packet-mode
  Kernel-configuratie: FreeBSD 7.1 generiek
  FreeBSD source-tree: 7.1 Release
  Node-specifieke cfg: nee
  ```

4.4.6. Systeemspecifieke installatie.

De NanoBSD installatie aangemaakt in de voorgaande paragraaf is een voorbeeld van een generieke installatie. Naast generieke-installatie kan het wenselijk zijn een NanoBSD-image op maat aan te maken, het is daarom ook mogelijk om een systeemspecifieke NanoBSD-image te compileren. Hiervoor kan een geheel nieuwe installatie worden uitgevoerd, of een voorgaande installatie voor worden gebruikt die binnen de `usr/obj` directory beschikbaar is. Indien de gewenste wijzigingen kunnen worden doorgevoerd op basis van een bestaande NanoBSD “world”-omgeving, die reeds is gecompileerd bij een eerdere NanoBSD installatie, kan het “buildworld”-gedeelte van de installatie worden overgeslagen.

In de volgende stappen wordt een systeemspecifiek NanoBSD image aangemaakt voor een PCEngines Alix2d3 systeembord. Daarnaast wordt in dit voorbeeld bij het uitvoeren van de installatie een specifieke node-configuratie vanuit de configuratiedatabase Genesis / Exodus opgehaald. De installatie wordt uitgevoerd op basis van de generieke NanoBSD-installatie die in de voorgaande stappen is gecreëerd. Voor de installatie wordt een specifiek kernel-configuratiebestand gebruikt. Controleer vooraf of de benodigde kernel-configuratie beschikbaar is in de `/usr/local/data/krncfg` directory, en ook een symlink is aangemaakt vanuit de `/usr/src/sys/i386/conf` –directory naar dit bestand. Indien geen geschikte kernel-configuratie beschikbaar is, kan deze gedownload worden vanuit svn en volgens onderstaande stappen toegevoegd worden.

- Plaats eerst het kernel configuratiebestand in de `/usr/local/data/krncfg` directory indien deze nog niet is toegevoegd.

  ```
  # cp ../alix2d3 /data/krncfg
  ```

- Symlink het kernel-configuratiebestand in de kernel conf directory naar dit bestand

  ```
  # ln –s /data/krncfg/Alix2d3 /usr/src/sys/i386/conf/alix2d3
  ```

- Maak nieuw NanoBSD-configuratiebestand aan op basis van de configuratie-template

  ```
  # cd /usr/src/tools/tools/nanobsd
  # cp Cfg/nanobsd.template Cfg/nanobsd.alix2d3.v01
  ```

- Wijzig de volgende variabelen binnen het aangemaakte configuratiebestand

  ```
  NANO_NAME=wleiden       # object naam in /usr/obj/nanobsd.{obj}
  NANO_SRC=/usr/src       # freebsd source tree
  NANO_KERNEL=alix2d3     # naam van het kernel configuratiebestand
  NANO_IMAGES=2           # aantal nanobsd code slices/installs
  ```

- Verwijder het commentaar “`#`” symbool voor de volgende optie.

  ```
  NANO_BOOT0CFG="-o nopacket -s 1 -m 3"
  ```

Let op, deze optie is specifiek voor het Alix systeembord en niet nodig voor de Soekris systemen. De Tinybios van het Alix bord lijkt een probleem te hebben met het opstarten van de flashdisk wanneer de boot0 configuratie in “packet mode” is gecompileerd. Hoewel in de bios ondersteuning voor LBA aangezet kan worden, biedt dit voor het probleem geen oplossing. Wanneer de `-o nopacket` optie wordt meegegeven aan de boot0 configuratie, dient wel de exacte geometrie data van de gewenste compactflash-kaart gedefinieerd te worden. Dit betekent dat het NanoBSD image dat bij de installatie wordt gecompileerd specifiek is voor de opgegeven geometrie, en alleen gestart kan worden vanaf een flashdisk met dezelfde specificaties.

Controleer of de geometrie informatie van de beoogde compactflash kaart beschikbaar is binnen het bestand FlashDevice.sub. Indien dit niet het geval is kan een toevoeging aan het bestand worden gemaakt, zie hiervoor de methode beschreven in paragraaf 2.2.4.

- Stel het compactflash-model in voor de FlashDevice configuratie, bijvoorbeeld:

```
FlashDevice kingston 512mb
```

De “custom” functies / macro’s en de opties die onder CONF_INSTALL zijn geconfigureerd kunnen desgewenst aangepast worden. Omdat in het voorbeeld als basis een bestaande NanoBSD-installatie wordt gebruikt en het compileren van de buildworld-omgeving kan worden overgeslagen, kunnen de opties onder CONF_WORLD genegeerd worden.
Voor het Alix2d3 systeembord zijn verder geen specifieke configuratie-instellingen vereist.

Met deze configuratie wordt aan het einde van de NanoBSD-installatie de last_orders -functie aangeroepen. Via deze macro kan een specifieke node-configuratie uit de Genesis / Exodus configuratiedatabase worden opgehaald. De functie start het config_update.sh script waarna de gewenste “CNodeNaam” ingevuld kan worden en vervolgens de node-specifieke bestanden naar de configuratieslice worden geschreven. Controleer voor de installatie te starten of een internetverbinding beschikbaar is.

- Start een NanoBSD-installatie zonder de “buildworld” stappen.

```
# sh nanobsd.sh –w –c nanobsd.alix2d3.v01
```

Het installatieproces slaat het genereren van de NanoBSD buildworld-omgeving over en start direct met de compilatie van de alix2d3 systeemkernel. De bestaande installworld-omgeving wordt hierbij verwijderd.

- Kopieer de aangemaakte imagebestanden naar de data/images directory

```
# mkdir /usr/local/data/images/alix2d3.wleiden
# cd /usr/obj/nanobsd.wleiden
# cp _.disk.full _.disk.image /usr/local/data/images/alix2d3.wleiden
```

- Maak voor de toegevoegde images het volgende readme-bestand aan.

```
# vi /usr/local/data/images/ali2d3.wleiden/readme

De NanoBSD images binnen deze directory zijn
Wireless Leiden specifiek.

NanoBSD-installatie gecompileerd met:

Configuratiebestand: nanobsd.alix2d3.v01
NanoBSD image-info : 512MB, no-packet-mode
Kernel-configuratie: FreeBSD 7.1 Alix2d3
FreeBSD source-tree: 7.1 Release
Node-specifieke cfg: ja
```

In de bovenstaande voorbeelden zijn bij het aanmaken van de NanoBSD-installaties geen wijzigingen doorgevoerd binnen de install- en build-world-opties van het NanoBSD configuratiebestand. De custom-macro’s en de bestaande opties onder CONF_INSTALL kunnen bij het configureren van een installatie vrij worden aangepast. Bij het wijzigen van de opties onder CONF_WORLD of het toevoegen van nieuwe variabelen, moet echter rekening gehouden worden met mogelijke conflicten die op kunnen treden bij het compileren en installeren van de NanoBSD build-world-omgeving. Hierover staat meet beschreven binnen het NanoBSD-hoofdstuk van dit document.

5 Node-installatie

5.1 Introductie

In dit hoofdstuk wordt de uiteindelijk node-installatie besproken zoals deze gebruikt kan worden voor het netwerk van Wireless Leiden.

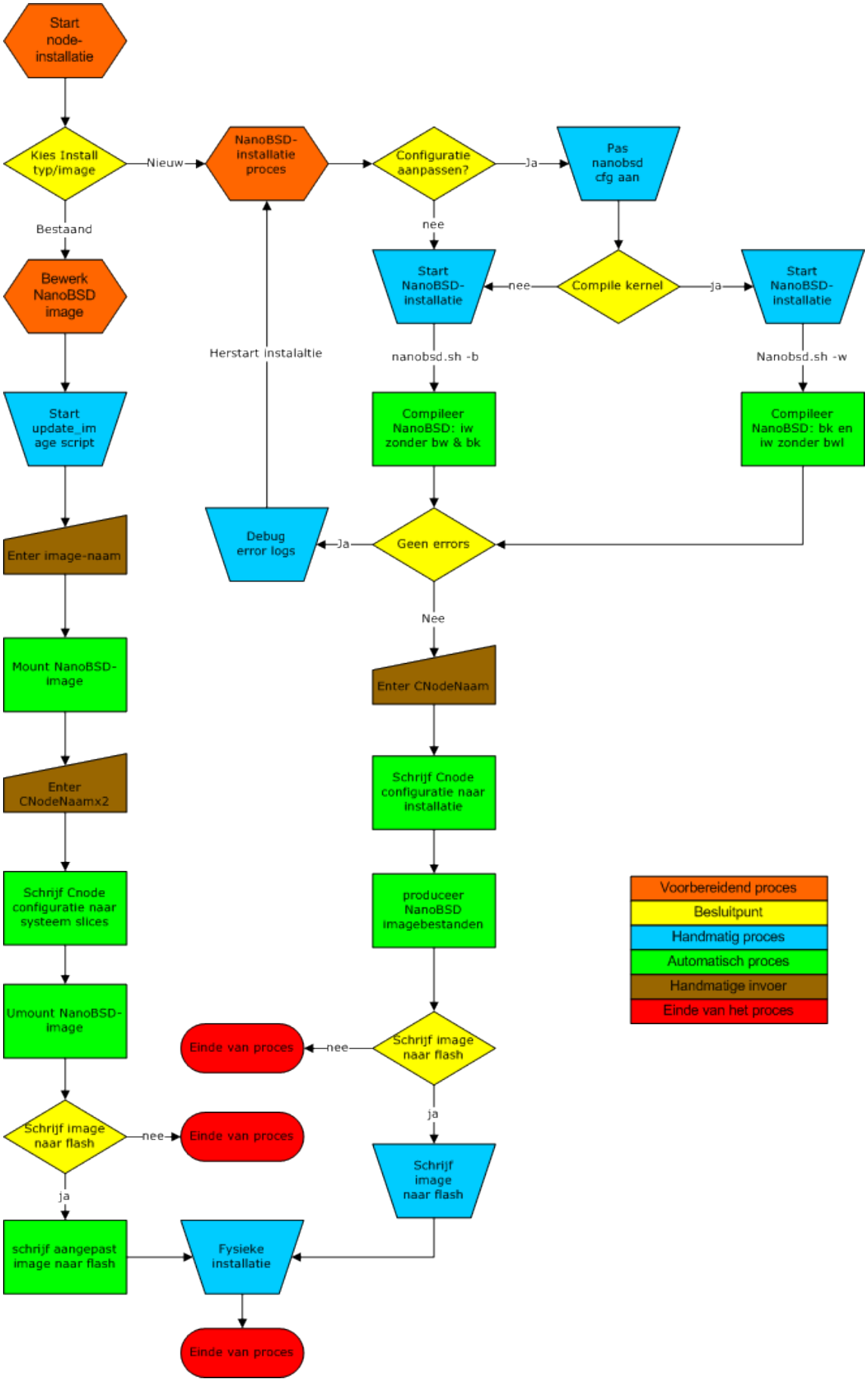
Het hoofdstuk start met een beschrijving van het node-installatieproces en de verschillende methoden die daarvoor gebruikt kunnen worden. De installatie wordt daarna inhoudelijk besproken aan de hand van de functionaliteit en eigenschappen van de geconfigureerde componenten. Als laatste gaat dit hoofdstuk in op een aantal aspecten van het operationeel beheer van de node-installatie. waaronder het doorvoeren van operationele wijzigingen, controleren van updates, en het uitvoeren van installaties op afstand.

5.2 Node-installatie, uitvoering

Binnen deze paragraaf worden de verschillende methoden beschreven voor het uitvoeren van een node installatie. De stappen in de procedures gaan er vanuit dat de NanoBSD-installaties, besproken in het voorgaande hoofdstuk, reeds zijn aangemaakt. Deze configuraties worden gebruikt als voorbeeld bij het uitvoeren van de daadwerkelijk node-installatie. Een node kan worden geïnstalleerd op basis van een bestaand NanoBSD-image, of door een nieuw image te produceren, waarbij een installatie compleet of gedeeltelijk opnieuw uitgevoerd wordt.

5.2.1. Node-installatie, stroomschema.

Het stroomschema beschrijft de processtappen voor het uitvoeren een node-installatie op basis een nieuwe NanoBSD-installatie en aan de hand van een bestaand NanoBSD-image.



Stroomschema node-installatie procedures

5.2.2. Nieuwe installatie

Hoewel de installatie van een node volgens verschillende methoden kan worden uitgevoerd, heeft het de voorkeur om een node te installeren op basis van een nieuwe NanoBSD-installatie. Bij het aanmaken van een nieuw NanoBSD-image wordt de systeemomgeving, samen met eventuele updates en of wijzigingen, opnieuw samengesteld. De consistentie en uniformiteit van een installatie kan op deze manier beter gegarandeerd worden dan wanneer de updates en of wijzigingen worden doorgevoerd aan een bestaand image. De systeemomgeving van een bestaand image kan bij een update vervuild raken met de configuratie van “oude” directories en of bestanden die eigenlijk niet meer nodig zijn. Doordat de systeemomgeving bij het uitvoeren van een NanoBSD-installatie opnieuw wordt samengesteld, kan een installatie op een schonere manier op maat gemaakt worden voor een beoogde node-configuratie. Doordat een NanoBSD-installatie in fasen kan worden uitgevoerd, neemt het produceren van een nieuw image een minimale hoeveelheid aan tijd in beslag.

In het volgende voorbeeld wordt een node-installatie uitgevoerd op basis van de NanoBSD-installatie die in het voorgaande hoofdstuk is aangemaakt. Bepaal voor de uitvoer van de node-installatie de invulling van de onderstaande punten:

- Het beoogde hardwareplatform: bijvoorbeeld Soekris net4801
- CNodeNaam van de betreffende node: bijvoorbeeld CNodeHuub
- Het volume en eventueel de geometrie informatie van de flashdisk: 1GB Kingston

De NanoBSD-installatie wordt gedeeltelijk opnieuw uitgevoerd op basis van de object-directory in `/usr/local/nanobsd.wleiden`. Binnen deze installatie-omgeving zijn twee eerdere installaties gecompileerd. Een generieke en een specifieke NanoBSD-installatie. Beide images zijn gecreëerd op basis van dezelfde buildworld-omgeving, maar hebben wel verschillende kernelconfiguraties gebruikt. Dit betekent dat voor de nieuwe installatie een van deze twee kernels geïnstalleerd kan worden, zonder het “buildkernel” -proces opnieuw te hoeven uitvoeren.

- Open een shell naar de nodefabriek en ga naar de NanoBSD werk-directory.

```
# cd /usr/src/tools/tools/nanobsd
```

- Sluit de compactflash kaart met een USB kaartlezer op hostsysteem aan zodat het NanoBSD-image na afloop van de installatie op het flashkaartje kan worden weggeschreven.

- Controleer of de verbinding met het internet beschikbaar is zodat de node-configuratie tijdens de installatie opgehaald kan worden uit de configuratiedatabase.

```
# ping ams-ix.net
PING ams-ix.net (91.200.16.49): 56 data bytes
64 bytes from 91.200.16.49: icmp_seq=0 ttl=59 time=18.028 ms
```

- Open het nanobsd.wleiden configuratiebestand.

```
# vi Cfg/nanobsd.wleiden
```

- Controleer de instellingen en pas deze naar wens aan. In het voorbeeld worden de instellingen verder niet gewijzigd, ervan uit gaande dat deze overeenkomen met de instellingen geconfigureerd in hoofdstuk 4

- Start de NanoBSD-installatie zonder de “buildworld” en “buildkernel” stappen.

```
# sh nanobsd.sh –b –c Cfg/nanobsd.wleiden
```

- Voer als laatste CNodeNaam in bij de onderstaande stap zodat de betreffende node-configuratie uit Genesis / Exodus opgehaald kan worden.

```
THIS script adds the config from GENESIS to this operating system
make sure you know what you are doing, if not press control-C
ENTER NODE NAME  ......(and press enter)
```

- Ga vervolgens naar de object-directory en schrijf het _.disk.full bestand naar naar de compactflash-kaart in /dev/da0. Let op dat het schijf-id kan variëren, afhankelijk van het aantal met USB aangesloten opslagsystemen op het hostsysteem beschikbaar zijn. Controleer zorgvuldig naar welk “device” geschreven mot worden.

```
# cd /usr/obj/nanobsd.wleiden
# dd if=_.disk.full of=/dev/da0 bs=64k
```

Als het installatieproces verder zonder fouten is verlopen, kan de compactflash-kaart fysiek op een node geïnstalleerd worden.

5.2.3. Bestaande image.

Naast de procedures beschreven in de voorgaande paragraaf, kan een node-installatie ook worden uitgevoerd op basis van een bestaand NanoBSD-image. Hierbij dient echter wel rekening gehouden te worden dat de consistentie van een image versus de huidige NanoBSD-installatie lastiger te garanderen is. Hoewel updates ook aan een bestaand image doorgevoerd kunnen worden, veroorzaakt deze methode vaak sneller vervuiling binnen de systeeminstallatie.

In een aantal gevallen, met name bij het updaten van operationele nodes, kan het echter handig zijn een node-installatie uit te voeren op basis van een bestaand NanoBSD-image. Voor de configuratie van een bestaand image is het gebruik van de nodefabriek in combinatie met NanoBSD niet noodzakelijk, maar kan het ook op een willekeurig FreeBSD systeem uitgevoerd worden. Dit maakt het bijvoorbeeld eenvoudiger een generiek image te publiceren op internet, waarna via een script de node-specifieke configuratie naar het image geschreven kan worden.

In het volgende voorbeeld wordt een node-installatie uitgevoerd op basis van het generieke NanoBSD-image dat in het voorgaande hoofdstuk is aangemaakt.
Het update_image.sh script dat hiervoor wordt gebruikt haalt de CNode-configuratie op uit genesis en voert deze door naar de configuratieslice binnen het image.

Het script wordt in het voorbeeld gestart vanaf de nodefabriek, maar kan ook uitgevoerd worden vanaf een willekeurig FreeBSD hostsysteem. Met volgende punten dient rekening gehouden te worden:

- Een bestaand NanoBSD-image moet op locatie beschikbaar zijn.
- De gebruikte compactflash-kaart moet groot genoeg zijn voor het NanoBSD-image. De geometrie van het imagebestand kan met het script niet worden aangepast.
- De kernelconfiguratie binnen het imagebestand moet geschikt zijn voor het beoogde doelsysteem.

Het script haalt de node-configuratie op uit de configuratiedatabase Genesis / Exodus met het bestaande config_update.sh script. Dit script moet binnen het imagebestand aanwezig zijn in de directory /tools. Het script wordt normaliter bij het uitvoeren van een NanoBSD-installatie aan het systeem toegevoegd.

- Indien het update_image.sh script nog niet op het hostsysteem aanwezig is, kan het aangemaakt worden volgens de weergegeven configuratie.
- Sluit een USB-kaartlezer met compactflash aan en controleer of de compactflash-kaart op schijf-id da0 is aangemeld.

```
# dmesg | grep da0
da0 at umass-sim0 bus 0 target 0 lun 0
da0: <USB2.0 CardReader CF RW 0.0>> Removable Direct Access SCSI-0
device
da0: 40.000MB/s transfers
da0: 3871MB (7928928 512 byte sectors: 255H 63S/T 493C)
```

- Start het script op de nodefabriek vanaf de volgende locatie

```
# sh /usr/local/data/scripts/update_image.sh
```

- Voer de naam en het volledige pad van het imagebestand in.

```
Enter NanoBSD image name...
/usr/local/data/images/generic.wleiden/_.disk.full
```

- Voer de gewenste CNodeNaam in bij de onderstaande stap

```
THIS script adds the config from GENESIS to this operating system
make sure you know what you are doing, if not press control-C
ENTER NODE NAME  ......(and press enter)
```

- Enter “y” om het aangepast imagebestand naar flash te schrijven.

```
Please wait, writing image to flash...
7819+1 records in
7819+1 records out
512483328 bytes transferred in 202.915879 secs (2525595 bytes/sec)
```

Als het installatieproces verder zonder fouten is verlopen, kan de compactflash-kaart fysiek op een node geïnstalleerd worden.

5.2.4. PXE installatie.

In de voorgaande paragrafen is bij het uitvoeren van de node-installatie het NanoBSD-image direct naar de compactflash-kaart geschreven. Wanneer geen mogelijkheid beschikbaar is om een compactflash-kaart op een hostsysteem aan te sluiten, of het beoogde systeembord van een node niet beschikt over een verwijderbaar opslag systeem, kan een node-installatie ook via het netwerk uitgevoerd worden.

In het volgende voorbeeld worden de stappen beschreven voor het uitvoeren van een node-installatie via de PXE en NFS-server die zijn geconfigureerd binnen de nodefabriek. De methode om een node op te starten via het netwerk dient alleen

Controleer voor het opstarten van een node eerst of de verschillende benodigde componenten binnen de nodefabriek actief zijn.

- Controleer de status van de DHCP-server.
- Controleer de status van de NFS-server en actieve exports.

- Controleer de status van de TFTP-server

-

Indien alle daemons naar behoren functioneren kan het systeembord via het netwerk worden aangesloten op de nodefabriek. Sluit het systeem via een UTP cat5 cross kabel aan op de vrije aansluiting van de nodefabriek.

5.3 Node-installatie, beschrijving

Binnen deze paragraaf wordt de uiteindelijke node-installatie besproken aan de hand van de eigenschappen en functionaliteit van de verschillende onderdelen die op het systeem zijn geconfigureerd. Het beschrijft de systeeminstallatie zoals deze in de voorgaande paragrafen is geconfigureerd, uitgevoerd en geïnstalleerd

5.3.1. Boot configuratie.

Aan de node-installatie zijn standaard de onderstaande instellingen aan de boot0 configuratie doorgevoerd.

```
-o packet -s 1 -m 3
```

De optie `–o packet` zorgt ervoor dat de disk packet interface wordt gebruikt in plaats van de Cylinder Head Sector, CHS interface. De bios van het systeem dient hierbij wel over ondersteuning voor Logical Block Adressing, LBA te beschikken. Bij het gebruikt van de disk packet interface is het niet nodig de exacte schijfgeometrie te definiëren.

De optie `–s 1` regelt welke van de beschikbare slices binnen de systeeminstallatie standaard wordt gebruikt voor het opstarten. In de bovenstaand configuratie wordt altijd van de eerste systeemslice opgestart, `ad0s1a`.

De optie `–m 3` activeert de status van de drie aanwezige slices binnen de installatie.

- `ad0s1a, ad0s2a, ad0s3`

De volgende instellingen worden bij het opstarten doorgevoerd met loader.conf:

```
beastie_disable="YES"
autoboot_delay="1"
kern.maxfiles="5000"
accf_http_load="YES"
```

De opties `beastie_disable` en `autoboot_delay` zijn doorgevoerd om respectievelijk het FreeBSD opstartmenu uit te schakelen en de vertraging in het laden van de kernel te minimaliseren.

De optie `kern.maxfiles` verhoogt het maximale aantal bestanden dat simultaan open kan staan naar 5000. Deze optie is toegevoegd naar aanleiding van de bind/named waarschuwingsmelding bij het starten van de daemon bij gebruik van de standaardconfiguratie

```
max open files (3520) is smaller than max sockets (4096)
```

De kernelmodule `accf_http_load` moet geladen worden voor de apache22 webserver. Deze module wordt gebruikt om een inkomende verbinding te bufferen totdat een compleet http verzoek ontvangen is. Zonder deze module geeft het starten van de apache daemon de volgende foutmelding

```
No such file or directory:Failed to enable the ‘httpready’ Accept Filter
```

5.3.2. rc.conf configuratie.

Het onderstaand `rc.conf` configuratiebestand geldt als standaard voor de nieuwe node-installatie. De configuratie is aangemaakt op basis van het bestaande `rc.conf` bestand, wat gebruikt wordt op de huidige node-installaties binnen het netwerk van Wireless Leiden. Configuratieopties voor de verschillende daemons zijn daarbij zoveel mogelijk ondergebracht in `rc.conf`. De configuratieopties die hiervoor zijn toegevoegd en of gewijzigd, worden inhoudelijk verder besproken bij de desbetreffende onderdelen.

```
apache22_enable="YES"
dhcpd_enable="YES"
dumpdev="NO"
gateway_enable="YES"
kern_securelevel_enable="NO"
lvrouted_enable="YES"
lvrouted_flags="-u -s s00p3rs3kr3t"
named_enable="YES"
nrpe_enable="YES"
ntpd_enable="YES"
ntpd_flags="-p /var/run/ntpd.pid -f /var/db/ntp.drift -g"
pen_enable="YES"
pen_flags="-b 30 -r -p /var/run/pen.pid -o prio 172.31.255.1:3128 `echo $proxylist`"
portmap_enable="NO"
sendmail_enable="NONE"
snmpd_enable="YES"
snmpd_flags="-a -LF w /var/log/snmpd.log"
sshd_enable="YES"
sshd_flags="-u0"
syslogd_flags="-s -A -c –b localhost"
sort_proxies="YES"
sort_proxies_list="proxy1:3128 proxy2:3128 proxy3:3128 proxy4:3128 proxy5:3128"
update_motd="NO"

if [ -f /etc/rc.node.local ] ; then
        . /etc/rc.node.local
fi
```

De nieuw toegevoegde optie `dumpdev="NO"` is niet gerelateerd aan een van de geïnstalleerde daemons, en is ingesteld zodat de volgende waarschuwingsmelding bij het opstarten wordt verholpen.

```
/etc/rc: WARNING: Dump device does not exist. Savecore not run.
```

De melding wordt veroorzaakt door het ontbreken van een swap partitie binnen de diskless-systeemconfiguratie.

5.3.3. Apache22, webserver.

Wireless Leiden heeft als functionele wens op de nodes een web-interface te gebruiken voor het uitvoeren van routinematige beheerstaken. De configuratie hiervan is momenteel nog steeds in ontwikkeling. Op de huidige node-installaties wordt Apache2 als standaard voor de installatie van een web-server. Binnen de nieuwe node-installatie is Apache versie 2.2.11. geïnstalleerd.

De configuratie van de Apache webserver is op verschillende punten aangepast.
De locatie van de wwwroot is verplaatst naar `/usr/local/etc/www`, zodat deze lees / schrijfbaar is en wijzigingen en of toevoegingen opgeslagen kunnen worden op de `/cfg`

slice. Vanuit de originele locatie /usr/local/www/apache22 is een symlink aangemaakt naar deze directory.

```
# ln –s /usr/local/etc/www /usr/local/www/apache22
```

De volgende site-configuratie is toegevoegd aan de documentroot van de Aapache webserver. De configuratie kan worden gebruikt voor web gebaseerd beheer van een aantal sleutelprocessen binnen de node-installatie. De documentroot bevat de meest recente configuratie die hiervoor binnen Wireless Leiden wordt gebruikt.

```
# ls /usr/local/www/apache22/data

action.php  exec.php    index.php
common.php  file.php    pen.php
```

Omdat voor de wwwroot-omgeving een symlink is aangemaakt vanaf de originele locatie, zijn de betreffende instellingen voor de locaties binnen de www-root niet gewijzigd in het httpd configuratiebestand. De volgende wijzigingen zijn aan het httpd.conf bestand in /usr/local/etc/apache22 doorgevoerd.

```
# Apache php5 module
LoadModule php5_module        libexec/apache22/libphp5.so

# Server-pool management (MPM specific)
Include etc/apache22/extra/httpd-mpm.conf
```

De Apache php5 module is toegevoegd om ondersteuning te bieden aan de op php gebaseerde web-interface. De regel hoeft niet handmatig toegevoegd te worden, de FreeBSD php5 port die bij het uitvoeren van de NanoBSD-installatie wordt geïnstalleerd is gecompileerd met de Apache-module.

De regel Include etc/apache22/extra/http-mpm.conf laad bij het starten van httpd een extra configuratiebestand in. Aan het http-mpm.conf bestand zijn wijzigingen voor de standaard Multi-Processing Module, MPM, van Apache doorgevoerd.

```
<IfModule mpm_prefork_module>
    StartServers          1
    MinSpareServers       1
    MaxSpareServers       5
    MaxClients           50
    MaxRequestsPerChild   0
</IfModule>
```

De configuratiewijziging is aan de MPM-module doorgevoerd zodat Apache opstart met een minimaal aantal wachtende servers, waardoor het standaard geheugen gebruik beperkt wordt.

5.3.4. Named, Domain Name Server.

Named wordt binnen Wireless Leiden op de node-installatie gebruikt als locale name-server. Hoewel deze configuratie in principe standaard is gebleven, is wel een “wijziging” doorgevoerd ten opzichte van de voorgaande node-installatie.
Aan het opstartscript van van de named daemon was destijds een hack toegevoegd met betrekking op het “read-only” functioneren van de /etc –configuratiedirectory. Omdat de /etc –directory in NanoBSD-opzet aangemaakt wordt als ramdisk, is deze wijziging in het opstartscript overbodig geraakt. In de nieuwe node-installatie wordt daarom gebruik gemaakt van het standaard named rc-script.

5.3.5. Lvrouted, dynamische routing.

Lvrouted is een routing daemon die binnen het netwerk van Wireless Leiden de dynamische routering verzorgt. De programmatuur is ontwikkeld door Wireless Leiden vrijwilliger Lodewijk Voege, zie de volgende Wiki-pagina voor meer informatie:

http://wiki.wirelessleiden.nl/DynamicRouting

Voor de nieuwe node-installatie zijn aanpassingen gemaakt aan de configuratie en opstart methode van de lvroute daemon. Zoals in de paragraaf over `rc.conf` staat beschreven, geldt als standaard geïnstalleerde daemons zoveel mogelijk binnen het FreeBSD RC framewerk te integreren. Het bestaande opstartscript van lvrouted is aangepast zodat de daemon in `rc.conf` kan worden opgenomen. De additionele configuratieopties kunnen ingesteld worden in `lvrouted_flags`. Het lvrouted configuratiebestand in `/usr/local/etc/` ,dat in de huidige node-installaties wordt gebruikt, is daarmee komen te vervallen. De volgende opties zijn ingesteld als vlagopties voor lvrouted binnen rc.conf: `lvrouted_flags="-u -s s00p3rs3kr3t"`

5.3.6. Pen, loadbalancing.

Pen is een “load balancing” -applicatie die toegepast kan worden voor een aantal tcp gebaseerde protocollen, zoals http of smtp. Het geeft de mogelijkheid meerde servers voor de gebruiker te laten verschijnen als een enkel systeem. Functioneel controleert Pen automatisch welke servers down zijn, en verdeelt de belasting van aangesloten cliëntsystemen over de beschikbare servers.

Bij Wireless Leiden wordt Pen gebruikt voor het verdelen van de http-belasting over de verschillende internet proxy-servers die binnen het netwerk van de organisatie beschikbaar zijn, zie de volgende Wireless Leiden Wiki-pagina voor meer informatie over de gebruikshistorie van Pen:

http://wiki.wirelessleiden.nl/LoadBalancing

Op de huidige nodes wordt Pen primair gebruikt om een redundant systeem te realiseren voor de verschillende toegangspunten naar het internet. Naast dit “failover” -principe is het echter belangrijk dat de beschikbare proxy-servers gesorteerd worden op de actuele beschikbare bandbreedte.

Het toewijzen van een proxy-volgorde op basis van prioriteit is in de voorgaande Pen configuraties gerealiseerd door een patch. Binnen het netwerk draait een gedeelte van de node-installatie met deze patch, waarbij de Pen-configuratie met optie –O wordt opgestart. In de laatste versie van Pen 0.18 is deze functionaliteit standaard ingebouwd en is de patch overbodig. De prioriteit kan in deze versie bepaald worden door de optie `-o prio` aan de configuratie mee te geven en per proxy een prioriteit toe te wijzen.

Omdat de bandbreedte vanaf en naar de verschillende proxy-servers per moment kan variëren, is het belangrijk dat de prioriteitsvolgorde niet statisch wordt toegevoegd, maar periodiek opnieuw berekend wordt de hand van de actuele capaciteit.

5.3.7. NTPD, tijdsynchronisatie.

Het synchroniseren van de systeemtijd wordt binnen de node-installatie uitgevoerd met ntpd. Ntpd is een daemon die binnen de FreeBSD-basis wordt gebruikt voor het synchroniseren van de tijd aan de hand van het “Network Time Protocol” Binnen de systeemomgeving van de NanoBSD-node-installatie is ntpd versie 4.5 geïnstalleerd. De volgende wijzigingen zijn doorgevoerd ten opzichte van de voorgaande node-installaties:

- De volgende vlagopties zijn voor ntpd opgenomen in rc.conf

  ```
  ntpd_flags="-p /var/run/ntpd.pid -f /var/db/ntp.drift -g"
  ```

Deze configuratie stond in de voorgaande node-installaties gedefinieerd in `/etc/ntp.conf`, daar het de voorkeur heeft de configuratie van de daemons zoveel mogelijk binnen rc.conf te integreren zijn de opties `–p`, locatie van het proces-id en `–f`, locatie van het `ntp.drift` bestand toegevoegd als vlagopties voor ntpd. De `–g` opties is toegevoegd om een eventueel gat tussen de tussen de lokale systeemklok en de serverklok te overbruggen en de serverklok te forceren als systeemtijd.

De verschillende beschikbare tijd-servers zijn gedefinieerd in het /etc/ntp.conf configuratiebestanden. De volgende servers zijn hierin opgenomen:

```
server proxy1.wleiden.net
server proxy2.wleiden.net
server proxy3.wleiden.net
server proxy4.wleiden.net
server proxy5.wleiden.net
```

Wanneer de node-hardware over een onnauwkeurige systeemklok beschikt, is het handig dat de tijdsaanpassingen, die worden weggeschreven in het `ntp.drift` –bestand, bewaard blijven bij het herstarten van een node. Omdat het ntp.drift bestand schrijfbaar dient te zijn, is het in de node-installatie opgeslagen op een ramdisk. Normaal gesproken zou het bestand daarom verloren gaan bij het herstarten of uitvallen van de systeemhardware. De volgende aanpassingen zijn gemaakt om de wijzigingen aan het ntp.drift bestand op het flashgeheugen op te slaan.

- Een leeg ntp.drift bestand is aangemaakt in de directory /usr/local/etc
- Een symlink naar dit bestand is aangemaakt in de originele ntp.drift directory

  ```
  ln –s /usr/local/etc/ntp.drift /var/db/ntp.drift
  ```

Omdat de locatie `/usr/local/etc` binnen de NanoBSD-installatie ge-symlinked is naar de directory `/etc/local` , kan door deze aanpassing het ntp.drift bestand op de configuratieslice worden opgeslagen. De bestandswijzigingen worden periodiek met een taak binnen crontab naar flashgeheugen geschreven. De volgende regel is opgenomen in de systeem-crontab:

```
0     12    *     *     *     root    /usr/local/bin/write_ntpdrift
```

Aan de hand van deze configuratie wordt zeven dagen per week eenmaal daags het write_ntpdrift –script opgestart, dat de wijzigingen naar de configuratieslice schrijft.

```
#!/bin/sh
# Back-up ntp.drift file changes to flash.

SRC="/var/db/ntp.drift"
DST="/cfg/local/ntp.drift"

mount /cfg
if [ -s $DST ]; then
  diff -i -b -B -q ${SRC} ${DST}
  if [ ! $? -eq 0 ]; then
    diff -i -b -B ${SRC} ${DST} | grep "<" | cut -d" " -f2 > ${DST}
  fi
else
  mkdir /cfg/local
  cp ${SRC} ${DST}
fi

umount /cfg

exit 0
```

5.3.8. NET-SNMP, systeem-monitoring.

Het monitoren van de systeemomgeving van de verschillende nodes wordt binnen het netwerk van Wireless Leiden primair uitgevoerd op basis van SNMP. In FreeBSD kan Net-SNMP worden gezien als de standaard voor een implementatie van de SNMP-architectuur. Op de huidige node-installaties wordt ook gebruik gemaakt van Net-SNMP. Aan de nieuwe node-installatie is de volgende FreeBSD-port versie van Net-SNMP toegevoegd: net-snmp-5.4.2.1_1.

De configuratie van de SNMP-daemon, snmpd.conf, is grotendeels gelijk gebleven ten opzichte van de voorgaande node-installatie. De optie `–LF w /var/log/snmpd.log` is toegevoegd aan de vlagopties zodat alleen waarschuwingmelding worden gelogd. Wanneer het log-niveau niet wordt aangepast loopt het volume van het log-bestand te hoog op waardoor de vrije ruimte op de `/var` –ramdisk volloopt.

Aan de Net-SNMP configuratie is de volgende extensie toegevoegd voor het monitoren van IP-leases binnen de beschikbare adrespools van de DHCP-server:

http://www.net-track.ch/opensource/dhcpd-snmp/

Hiervoor zijn de volgende bestanden toegevoegd aan de systeemomgeving van de node-installatie.

- `/usr/local/sbin/dhcpd-snmp`,
- `/usr/local/etc/dhcpd-snmp.conf`.

Om het dhcpd-snmp script in /usr/local/sbin te kunnen gebruiken, zijn de volgende opties opgenomen in het dhcp-snmp.conf configuratiebestand:

```
leases: /var/db/dhcpd.leases
pool: 1, pool1, 172.17.16.70-172.17.16.94
pool: 2, pool2, 172.17.16.10-172.17.16.60
```

leases: definieert de locatie van het dhcpd.leases bestand waarin de actuele dhcpd-leases worden weggeschreven. Dit bestand moet toegankelijk zijn voor het script.

pool: definieert de beschikbare adrespools die in de DHCP-server gebruikt worden. De regel kan als volgt gelezen worden, pool: index, beschrijving, ip1-ip2, ip3-ip4 waarbij index staat voor een uniek numeriek indexnummer.

Als laatste aanpassing is de volgende regel opgenomen in het dhcpd configuratiebestand

```
# vi /usr/local/share/snmp/snmpd.conf
…
pass_persist .1.3.6.1.4.1.21695.1.2 /usr/local/sbin/dhcpd-snmp
/usr/local/etc/dhcpd-snmp.conf
```

Deze configuratieregel maakt het mogelijk een SNMP-query uit te voeren naar het volgende OID:

```
CNodeDirkLos2# snmpwalk -v1 -c public localhost .1.3.6.1.4.1.21695.1.2

SNMPv2-SMI::enterprises.21695.1.2.1 = INTEGER: 2 ,aantal geconfigureerd pools
SNMPv2-SMI::enterprises.21695.1.2.2.1.1 = INTEGER: 1 , pool indexnummer
SNMPv2-SMI::enterprises.21695.1.2.2.1.2 = INTEGER: 2 , pool indexnummer
SNMPv2-SMI::enterprises.21695.1.2.2.2.1 = STRING: "pool1" , pool omschrijving
SNMPv2-SMI::enterprises.21695.1.2.2.2.2 = STRING: "pool2" , pool omschrijving
SNMPv2-SMI::enterprises.21695.1.2.2.3.1 = INTEGER: 51 , capaciteit pool p1
SNMPv2-SMI::enterprises.21695.1.2.2.3.2 = INTEGER: 25 , capaciteit pool p2
SNMPv2-SMI::enterprises.21695.1.2.2.4.1 = INTEGER: 0 , actieve leases p1
SNMPv2-SMI::enterprises.21695.1.2.2.4.2 = INTEGER: 1 , actieve leases p2
SNMPv2-SMI::enterprises.21695.1.2.2.5.1 = INTEGER: 0 , verlopen leases p1
SNMPv2-SMI::enterprises.21695.1.2.2.5.2 = INTEGER: 0 , verlopen leases p2
SNMPv2-SMI::enterprises.21695.1.2.2.6.1 = INTEGER: 51 , beschikbare leases p1
SNMPv2-SMI::enterprises.21695.1.2.2.6.2 = INTEGER: 24 , beschikbare leases p2
```

Daar de configuratie van het dhcpd-snmp.conf -bestand node-specifiek is, dient het bestand opgeslagen te worden op de configuratieslice. Omdat de configuratie nog nieuw is en getest moet worden is het bestand nog niet opgenomen in de configuratiedatabase Genesis / Exodus. Gedurende de NanoBSD-installatie wordt daarom een tijdelijk script gestart, warmee de configuratie van de dhcp-adrespools naar het dhcpd-snmp.conf wordt geschreven.

```
# vi /tools/dhcpd_snmp.sh

#!/bin/sh
# Write dhcpd ranges to dhcpd-snmp.conf

DHCPD="/cfg/local/dhcpd.conf"
DHSNMP="/cfg/local/dhcpd-snmp.conf"
INDEX="0"

DHPOOL=`cat ${DHCPD} | grep range | awk '{print $2"-"$3}' | cut -d";" -f1`
for range in ${DHPOOL}
do
   INDEXNR="$((INDEX=INDEX+1))"
   echo "`echo "pool:" $INDEX", pool"$INDEX", "$range | \
   sed 's/^.*(//'`" >> $DHSNMP
done
```

Het script wordt uitgevoerd tijdens de last_orders() functie in het NanoBSD-configuratiebestand, zie de bijlagen in Appendix A voor mee informatie.

5.3.9. Syslogd, systeemlogging.

Het omgaan met systeemlogs is een belangrijk aspect van zowel beveiligings- als systeembeheer. Bij Wireless Leiden wordt de syslog-daemon gebruikt voor het bijhouden van de diverse systeemlogs. Syslogd is opgenomen in het FreeBSD basis systeem. Naast het lokaal loggen van de systeemmeldingen, worden de log-bestanden verzameld op een centrale log-host binnen het Netwerk van Wireless Leiden. Naast dat het monitoren van de verschillende nodes hierdoor beheertechnisch eenvoudiger is, verzorgt het ook redundantie en in de opslag van de log-bestanden. Daarbij fungeert de centrale log-server ook als back-up wanneer de lokale log-bestanden op een node verloren gaan door een herstart of uitval van het systeem.

De volgende regel is hiervoor gewijzigd binnen het /etc/syslogd.conf configuratiebestand:

```
# uncomment this to enable logging to a remote loghost named loghost
*.*                                             @loghost.wleiden.net
```

Daarnaast zijn de volgende configuratievlaggen zijn voor syslogd opgenomen in rc.conf van de node-installatie:

```
syslogd_flags="-s -A -c -b CNodeDirkLos2.wLeiden.NET"
```

De extra optie die is toegevoegd ten opzicht van de voorgaande node-installatie is `-b CNodeDirkLos2.wLeiden.NET .` Deze configuratie zorgt ervoor dat het hostadres van een node aan gekoppeld wordt aan syslogd. Dit is noodzakelijk voor het op afstand loggen van de systeemmeldingen.

Het volgende script is tijdelijk opgenomen in het NanoBSD-installatie proces om het hostadres naar de –b vlagoptie van syslogd in rc.conf te schrijven.

```
#!/bin/sh
# Set hostname for syslogd -b flag

HOST=`cat /cfg/rc.node.local | grep hostname | cut -d'"' -f2`
sed -i "" -e /syslogd_flags=/s/localhost/${HOST}/ /cfg/rc.conf
```

Het script wordt aangeroepen in de last_orders() functie in het NanoBSD-configuratiebestand. Zie Appendix A voor meer informatie.

5.4 Node-installatie, beheer

Deze paragraaf gaat in op een aantal beheersaspecten van de uiteindelijk node-installatie. Het gaat voornamelijk in op het uitvoeren van updates aan de systeeminstallatie en beschrijft de procedures voor het doorvoeren van operationele wijzigingen. Als laatste worden een aantal punten besproken die specifiek zijn voor het prototype iris-node.

5.4.1. Remote updates.

De verschillende nodes binnen het netwerk van Wireless Leiden bevinden zich vaak op locaties die lastig te bereiken zijn, en waarbij voor fysiek onderhoud vaak een afspraak gemaakt moet worden. Hoewel het doorvoeren van updates en of wijzigingen op afstand in de voorgaande node-installatie ook mogelijk was, diende bij het vervangen van de gehele node-installatie op locatie de node offline gebracht te worden en al daar de bestaande flashdisk fysiek te wisselen met de nieuwe installatie.
Door de opzet van de NanoBSD-installatie kan de bovenstaande procedure ook op afstand uitgevoerd worden. Het ontwerp van de installatie bestaat uit een dubbele systeemomgeving waarbij slecht een van de twee actief is. Dit maakt het mogelijk om de niet actieve omgeving op afstand te voorzien van een nieuwe installatie.

Bij het aanmaken van een NanoBSD-installatie wordt een tweetal images geproduceerd; een volledig image, _.disk.full, dat de gehele installatie omvat inclusief de configuratie slice, en een image met alleen de data van een enkele systeemomgeving, _.disk.image. Het tweede image kan gebruikt worden voor het updaten van de niet actieve systeeminstallatie.
Voor het updaten van een node-installatie op afstand zijn binnen NanoBSD twee kant en klare scripts opgenomen. De `updatep1 / updatep2` scripts zijn in systeemomgeving van de nodeinstallatie toegevoegd aan de `/tools` directory, en kunnen gebruikt worden om respectievelijk systeemslice 1 een systeemslice 2 van een nieuwe installatie te voorzien. Voor de overdracht van het imagebestand kan het update script kan gebruikt worden in combinatie met bijvoorbeeld FTP of SSH.

- ssh usr@host cat _.disk.image | sh updatep1
- ftp host – get _.disk.image “| sh updatep1”

Zoals uit de voorbeelden blijkt, kan het imagebestand op een willekeurige locatie staan, zolang de benodigde netwerkdienst en bijbehorende toegangsrechten beschikbaar zijn.

De volgende stappen kunnen gevolgd worden om een installatie op afstand uit te voeren. De installatiestappen in het voorbeeld gaan ervan uit dat reeds een nieuw NanoBSD-image is aangemaakt. Indien dit niet het geval is kan in paragraaf 5.2.2 meer informatie gevonden worden over het aanmaken van een NanoBSD-installatie.
Voor het updaten van de node-installatie binnen het netwerk van Wireless Leiden is het belangrijk dat het imagebestand zo klein mogelijk is, zodat de overdrachtstijd beperkt blijft en de update ook uitgevoerd kan worden naar locaties waar minder bandbreedte beschikbaar is.

- Maak een tar-archief-bestand aan voor het betreffende _.disk.image bestand dat naar de gewenste node geschreven moet worden.

```
# cd /usr/local/data/images/new && ls
_.disk.full       _.disk.image

# tar cfz _.disk.image.tar.gz _.disk.image
```

Door een archief bestand aan te maken, kan het volume van het imagebestand gereduceerd worden van 240MB naar 64MB.

- Open een SSH-shell naar de betreffende node.

  ```
  # ssh root@CNodeDirkLos2.wLeiden.NET
  ```

- In het voorbeeld wordt een nieuwe installatie naar systeemslice 2 geschreven, start het updatep2 -script met het volgende commando:

  ```
  # ssh root@CNodeFabriek.wleiden.net cat /usr/local/data/
  images/new/_.disk.image.tar.gz | zcat | sh /tools/updatep2
  ```

Afhankelijk van de beschikbare banbreedte kan dit proces enige tijd duren. Het script geeft geen output gedurende de bestandsoverdracht. Wanneer de update succesvol is afgerond, geeft het script de volgende output:

```
495873+0 records in
3874+1 records out
253886976 bytes transferred in 325.602495 secs (779745 bytes/sec)
** /dev/ad0s2a (NO WRITE)
** Last Mounted on /mnt
** Phase 1 - Check Blocks and Sizes
** Phase 2 - Check Pathnames
** Phase 3 - Check Connectivity
** Phase 4 - Check Reference Counts
** Phase 5 - Check Cyl groups
8066 files, 360891 used, 126293 free (10565 frags, 14466 blocks, 2.2%
fragmentation)
#   flag     start chs   type       end chs       offset         size
1   0x80       0:  1: 1   0xa5    491: 15:63           63       495873
2   0x00     492:  1: 1   0xa5    983: 15:63       495999       495873
3   0x00     984:  0: 1   0xa5    992: 15:63       991872         9072

version=1.0  drive=0x80  mask=0x3  ticks=182
options=packet,update,nosetdrv
default_selection=F2 (Slice 2)
```

De laatste regel in de output van het script toont dat de `boot0` configuratie is aangepast, en de standaard slice waarvan wordt opgestart is ingesteld op slice nummer 2. Dit betekend dat bij de eerste volgende herstart van het systeem opgestart wordt van de systeemomgeving op slice 2.
Daar de node-specifieke configuratiebestanden worden opgeslagen op de configuratie slice, heeft het updaten van de systeemomgeving verder geen invloed op deze instellingen. Het systeem zal dan ook weer met dezelfde node-specifieke configuratie opstarten.

Appendix A , NanoBSD configuratie-template

Deze bijlage beschrijft de stappen die nodig zijn om een NanoBSD configuratie-template aan te maken. Het template bestand is specifiek voor Wireless Leiden maar kan verder naar wens worden aangepast. De verschillende onderdelen van het configuratiebestand kunnen worden onderverdeeld in een aantal algemene configuratie-opties, de opties die de nanobsd systeemomgeving strippen, en een aantal eigen functies /macro's.

De NanoBSD-installatie wordt gestript van de opties die zijn geconfigureerd onder `CONF_WORLD='` en `CONF_INSTALL='`. De opties zijn ingesteld aan de hand van een referentiekader dat is opgezet door Poul Henning-Kamp: http://phk.freebsd.dk/misc/build_options/ . De tabel die op deze site gevonden kan worden bevat een overzicht van opties die aan de make-configuratie van NanoBSD meegegeven zouden kunnen worden, en laat daarbij mogelijke conflicten zien tussen de Buil- en Install-world configuraties.

De configuratie onder `CONF_WORLD='` en `CONF_INSTALL='` geld als basis en zou mogelijk verder uitgebreid kunnen worden. Alleen Opties waarvan zeker gesteld kon worden dat deze niet noodzakelijk zijn voor de systeemomgeving van een node-installatie zijn aan de configuratie toegevoegd.

- Maak een NanoBSD configuratie-template aan binnen de `../nanobsd/Cfg` directory.

  ```
  # vi nanobsd.template
  ```

- Maak als eerste de volgende algemene variabelen aan binnen het template bestand.

  ```
  ## Dit is een NanoBSD configuratie-template voor WirelessLeiden.
  ## Instellingen weergegeven binnen dit bestand gelden als
  ## standaard binnen de organisatie.


  NANO_NAME=              # object naam in /usr/obj/nanobsd.{obj}
  NANO_SRC=               # nanobsd source tree
  NANO_KERNEL=            # naam van het kernel configuratiebestand
  NANO_IMAGES=            # aantal nanobsd code slices/installs (1/2)

  NANO_CONFSIZE=          # volume van de config slice, default 2048 (512bs)
  NANO_DATASIZE=          # volume van de data slice, 0 = niet geconfigureerd
  #NANO_CODESIZE=         # volume van de code slice, default = max beschikbaar
  #NANO_RAM_ETCSIZE=      # volume van de /etc ramdisk, default 10240 (512bs)
  #NANO_RAM_TMPVARSIZE=   # volume van de /var ramdisk, default 10240 (512bs)
  ```

- De onderstaande variabelen zijn optioneel, waarbij de optie voor de `boot0` configuratie systeemafhankelijk is.

  ```
  #NANO_PMAKE="make -j 4"       # parallelle make threads, default 3.

  #NANO_BOOT0CFG="-o nopacket -s 1 -m 3"
  ```

- Maak vervolgens de onderstaande `make.conf` variabelen aan, die worden doorgevoerd bij de “build & install-world” stappen van het NanoBSD installatieproces.

```
CONF_WORLD='       # opties parsed gedurende build & install world

NO_ACPI=YES        # geen advanced configuration power interface
NO_KERBEROS=YES    # geen ondersteuning Kerberos authenticatie
NO_GAMES=YES       # geen games gecompileerd
NO_SYSCONS=YES     # geen syscon devices gecompileerd
NO_INFO=YES        # geen info bestanden, readable online docs
NO_INET6=YES       # geen ondersteuning inet versie 6 architectuur
NO_ATM=YES         # geen ondersteuning Asynchronous Transfer Mode
NO_RESCUE=YES      # geen rescue bestanden gecompileerd
NO_LOCALES=YES     # geen ondersteuning lokalisatie
NO_CALENDAR=YES    # geen calendar reminder service gecompileerd
NO_FORTRAN=YES     # geen ondersteuning fortran compilers
NO_MAN=YES         # geen handleidingen gecompileerd
NO_HTML=YES        # geen html help bestanden gecompileerd
NO_EXAMPLES=YES    # geen voorbeeld configuratiebestanden
NO_SHAREDOCS=YES   # geen share/docs directories
NO_LPR=YES         # geen ondersteuning print services
NO_DICT=YES        # geen dictionary ondersteuning
NO_NIS=YES         # geen ondersteuning network information system
NO_I4B=YES         # geen ondersteuning voor isdn
NO_GDB=YES         # geen gnu debugger gecompileerd
NO_GCOV=YES        # geen gcov test coverage program
NO_GPIB=YES        # geen ondersteuning gpib kaarten
NO_RCMDS=YES       # geen ondersteuning rcmds,
NO_IPX=YES         # geen ondersteuning ipx protocols
'
```

- Maak de variabelen aan die alleen doorgevoerd moeten worden bij het “installworld” proces.

```
CONF_INSTALL='     # opties parsed gedurende install world

NO_AUDIT=YES       # geen event auditing / audit trails
NO_AUTHPF=YES      # geen authenticating gateway user shell
NO_BLUETOOTH=YES   # geen ondersteuning Bluetooth modules
NO_USB=YES         # geen ondersteuning usb modules
NO_TOOLCHAIN=YES   # geen freebsd toolchain
NO_MAILWRAPPER=YES # geen mailwrapper bij gebruik sendmail
NO_CVS=YES         # geen cvs tools geïnstalleerd
#NO_PF=YES         # geen packet filtering geïnstalleerd
#NO_IPFILTER=YES   # geen ip filtering geïnstalleerd
#NO_SHARE=YES      # geen share sub directory
#NO_SENDMAIL=YES   # geen sendmail geinstalleerd
#NO_BIND=YES       # geen bind tools, dns/named geïnstalleerd
#NO_MISC=YES       # geen misc sub directory
#NO_MODULES=YES    # geen modules
'
```

De verschillende variabelen die tot nu zijn aangemaakt zorgen ervoor dat de NanoBSD installatie uiteindelijk gestript wordt van overbodige functionaliteiten / opties.
De instellingen die gestript worden onder “CONF_WORLD” zijn niet mee gecompileerd bij het “build-world”-proces en kunnen dus niet meer worden geïnstalleerd bij het “install-world”-proces. Voor de opties die zijn geconfigureerd onder “CONF_INSTALL” geldt dat nog een keuze gemaakt kan worden wat betreft de uiteindelijke installatie.

- Voeg de volgende regel toe aan het template-bestand. Deze optie wordt gebruikt om de naam en het type van de gebruikte flashkaart te definiëren. Het model dient bekend te zijn binnen het bestand FlashDevice.sub

```
FlashDevice fabrikant 512
```

De “Custom macro’s” zijn de laatste onderdelen die aan de NanoBSD configuratie-template toegevoegd worden.

- Maak de onderstaande macro aan voor het wijzigen van de instellingen binnen het bestand /boot/loader.conf.

```
# Wijzigingen voor loader.conf

cust_loader() (
    touch ${NANO_WORLDDIR}/boot/loader.conf
    echo "beastie_disable=\"YES\"" >> ${NANO_WORLDDIR}/boot/loader.conf
    echo "accf_http_load=\"YES\"" >> ${NANO_WORLDDIR}/boot/loader.conf
    echo "autoboot_delay=\"1\"" >> ${NANO_WORLDDIR}/boot/loader.conf
    echo "kern.maxfiles=\"5000\"" >> ${NANO_WORLDDIR}/boot/loader.conf
)
```

- Maak de volgende macro aan voor het compileren en installeren van de lvroute daemon. Lvrouted wordt hierbij gedownload en gecompileerd vanuit de huidige bron-bestanden in subversion.

```
# Compile & install lvroute daemon vanuit source in svn

cust_install_lvrouted() (
   cd /tmp
   svn co http://svn.wirelessleiden.nl/svn/node-config/other/lvrouted/trunk/
   cd /tmp/trunk && autoconf && autoheader && ./configure && make || true
   cp src/lvrouted.opt ${NANO_WORLDDIR}/usr/local/sbin
)
```

- Voeg de volgende macro toe voor het aanpassen van de Apache web-server.

```
# Customize apache

cust_apache() (
    rm -rf ${NANO_WORLDDIR}/usr/local/www/apache22/
    chroot ${NANO_WORLDDIR} sh -c "ln -s /usr/local/etc/www \
    /usr/local/www/apache22"
)
```

- Maak de onderstaande macro aan voor het wijzigen van de fysieke locatie van het ntp.drift bestand, zodat deze kan worden opgeslagen op de /cfg slice

```
# Customize ntpd

cust_ntpd() (
    chroot ${NANO_WORLDDIR} sh -c "ln -s /usr/local/etc/ntp.drift \
    /var/db/ntp.drift"
)
```

- Maak de volgende macro aan voor het wijzigen van de comconsole instellingen.

```
# Customize comconsole
# Macro overschrijft default in nanobsd.sh

cust_comconsole() (
   sed -i "" -e /ttyd0/s/off/on/ ${NANO_WORLDDIR}/etc/ttys
   sed -i "" -e /ttyd0/s/dialup/ansi/ ${NANO_WORLDDIR}/etc/ttys
   sed -i "" -e '/^ttyv[0-8]/s/     on/   off/' ${NANO_WORLDDIR}/etc/ttys
   echo " -h" > ${NANO_WORLDDIR}/boot.config
)
```

- Voeg als laatste de onderstaande functie toe voor het doorvoeren van node-specifieke configuratie-instellingen.

```
# Functie om node-specifieke configuratiebestanden aan de /cfg
# configuratieslice toe te voegen. Functie wordt uitgevoerd
# als “last_order”, na het afronden van de NanoBSD-installatie,
# en dient niet toegevoegd te worden aan de customize_cmd –lijst.

last_orders () (

      # Mount config slice in ${NANO_WORLDDIR}
      MD=`mdconfig -a -t vnode -f ${MAKEOBJDIRPREFIX}/_.disk.full`
      mount -t ufs /dev/${MD}s3 ${NANO_WORLDDIR}/cfg

      # Create etc directories in /cfg
      cd ${NANO_WORLDDIR}/cfg/ && mkdir local namedb
      cd ${NANO_WORLDDIR} && cp etc/rc.conf cfg/
      cd ${NANO_WORLDDIR} && cp usr/local/etc/dhcpd-snmp.conf cfg/local/

      # Copy resolv.conf for internet access
      cp /etc/resolv.conf ${NANO_WORLDDIR}/etc

      # Run config_update.sh script to fetch node config to /cfg
      chroot ${NANO_WORLDDIR} sh -c "/tools/config_update.sh"
      # Temporary scripts
      chroot ${NANO_WORLDDIR} sh -c "/tools/dhcpd_snmp.sh"
      chroot ${NANO_WORLDDIR} sh -c "/tools/syslogd_flag.sh"

      # umount /cfg and delete vnode
      umount ${NANO_WORLDDIR}/cfg && mdconfig -d -u ${MD}
)
```

- De verschillende macro's die aan de configratie-template zijn toegevoegd, dienen aangeroepen te worden met de `cutomize_cmd` functie. Voeg hiervoor de volgende regels aan de template toe.

```
# Cust macro`s gestart in onderstaande volgorde

customize_cmd cust_pkg
customize_cmd cust_install_files
customize_cmd cust_apache
customize_cmd cust_ntpd
# customize_cmd cust_install_lvrouted
customize_cmd cust_allow_ssh_root
customize_cmd cust_comconsole
customize_cmd cust_loader
```

Wanneer de macro's niet binnen het NanoBSD configuratiebestand zijn uitgeschreven, en wel in de lijst van `customize_cmd` commando's staan, zijn het functies die in het `nanobsd.sh` shell script zijn geïntegreerd. Let op dat de functie `cust_install_lvrouted` in het template bestand standaard dus niet wordt uitgevoerd. De binary van lvrouted wordt normaal gesproken voor-gecompileerd aan de installatie toegevoegd.

- Sla de toevoegingen aan de configuratie-template op en sluit het bestand af.

Appendix B , port-compilatie-script

Binnen deze bijlage is het port-compilatie-script opgenomen. Daarnaast zijn de stappen beschreven voor het aanmaken van zowel het script als de bijbehorende lijst met FreeBSD ports die aan de NanoBSD-installatie worden toegevoegd.

- Maak het volgende bestand aan in `..data/scripts` directory van de werkomgeving.

```
# vi /usr/local/data/scripts/ports_list_full

shells/bash
dns/bind96 WITH_REPLACE_BASE=yes
textproc/expat2
net/fping
print/freetype2
devel/gettext
benchmarks/iperf
net/isc-dhcp3-server
converters/libiconv
textproc/libxml2
devel/m4
net-mgmt/nagios-plugins
net-mgmt/net-snmp
net-mgmt/nrpe2
security/p5-Crypt-CBC
security/p5-Crypt-DES
security/p5-Digest-HMAC
security/p5-Digest-SHA1
net-mgmt/p5-Net-SNMP
devel/p5-Locale-gettext
devel/pcre
net/pen
lang/perl5.8
lang/php5 WITH_APACHE=yes
devel/pkg-config
net/radiusclient
security/sudo
```

Het bestand `ports_list_full` bevat een lijst met alle FreeBSD ports die standaard geïnstalleerd worden bij het uitvoeren van een NanoBSD-installatie. De verschillende ports zijn aan de lijst toegevoegd aan de hand van de directory waarbinnen de betreffende port zich bevind. Daarnaast kunnen in de lijst per port argumenten worden opgenomen die worden doorgevoerd aan de make configuratie. Een argument kan achter de directory-naam worden geplaatst, gescheiden door middel van een spatie, met een maximum van drie argumenten. Zie de ports `dns/bind96` en `lang/php5` als voorbeeld.

Met het onderstaande script kunnen de ports die binnen het port_list_full bestand zijn gedefinieerd worden gecompileerd, en als packages toegevoegd worden aan de ..nanobsd/Pkg directory in de werkomgeving. Als het script nog niet in de ..data/script directory aanwezig is kan het eveneens handmatig aangemaakt worden.

- Maak het onderstaande script aan in de directory /usr/local/data/scripts

```
# vi /usr/local/data/scripts/create_packages.sh

#!/bin/sh
# Script to compile ports/packages from ports_list.

# Directory to place packages
PKGDIR="/usr/local/data/nanobsd/"

# FreeBSD directory for compiled ports
PRTDIR="/usr/ports/packages/All"

# Directory containing the ports_list
PLDIR="/usr/local/data/scripts"

# Make a choice to compile all ports or updates only
echo "This script compiles packages for ports listed"
echo "in either ports_list_full or ports_list_update."
echo "      "
echo "Please enter 1 for full, or 2 for update:"

read answer
case $answer in
    1)    PLIST=ports_list_full;;
    2)    PLIST=ports_list_update;;
    *)    echo You did not choose 1 or 2 exit 0;;
esac
clear
echo Compiling packages from $PLIST...

# Make command & options
MKOPT="make package-recursive -DBATCH FORCE_PKG_REGISTER=yes"

# Compile ports & move packages to /Pkg
cat "$PLDIR/$PLIST" $1|while read LINE; do
        PDIR=`echo $LINE|awk '{print $1}'`
        MKARG=`echo $LINE|awk '{print $2 $3 $4}'`
        PNAME=`echo $LINE|cut -d"/" -f2|cut -d" " -f1`
        cd /usr/ports/$PDIR && $MKOPT $MKARG
        mv $PKGDIR/Pkg/$PNAME*.* $PKGDIR/Pkg_old
        cp $PRTDIR/$PNAME*.* $PKGDIR/Pkg
        make clean
done

# Clean up the ports environment afterwards
echo Finished compiling ports, cleaning up ports environment...
portsclean --workclean --distclean

exit 0
```

Appendix C , NanoBSD-configuratiebestanden

Deze bijlage bevat een overzicht van alle additionele bestanden en directories die aan de systeemomgeving van de NanoBSD-installatie worden toegevoegd. De betreffende configuratiebestanden worden inhoudelijk besproken bij de het beschrijven van de node-installatie in paragraaf 5.3

De onderstaande directorystructuur wordt recursief gekopieerd naar de root van een NanoBSD-installatie. Een directorystructuur wordt alleen gekopieerd als minimaal een bestand is toegevoegd aan een sub-directory. Een lege structuur wordt overgeslagen.

```
+ data/nanobsd/Files:
      + etc       + root      + tools     + usr
```

De onderstaande wijzigingen en toevoegingen worden gemaakt binnen de /etc configuratie van een node:

```
+ data/nanobsd/Files/etc:
      - crontab   - master.passwd    - passwd   - syslog.conf     - rc.conf
      - group     - ntp.conf         - pwd.db   - spwd.db         + ssh

+ data/nanobsd/Files/etc/ssh:
      - ssh_host_dsa_key      - ssh_host_key          - ssh_host_rsa_key
      - ssh_host_dsa_key.pub  - ssh_host_key.pub      - ssh_host_rsa_key.pub
```

De volgende wijzigingen en toevoegingen worden gemaakt binnen de /usr/local configuratie van een node.

```
+ data/nanobsd/Files/usr:
      + local
+ data/nanobsd/Files/usr/local:
      + bin       + etc       + lib       + sbin      + share

+ data/nanobsd/Files/usr/local/bin:
      - ssh-copy-id     - write_ntpdrift


+ data/nanobsd/Files/usr/local/etc:
      + apache22              - dhcpd.conf      - ntp.drift       + www
      - dhcpd-snmp.conf       - nrpe.cfg        + rc.d

 + data/nanobsd/Files/usr/local/etc/apache22:
       + extra           - httpd.conf
 + data/nanobsd/Files/usr/local/etc/apache22/extra:
       - httpd-mpm.conf

 + data/nanobsd/Files/usr/local/etc/www:
       + cgi-bin   + data      + error     +icons

 + data/nanobsd/Files/usr/local/etc/www/cgi-bin:
       - printenv        - test-cgi

 + data/nanobsd/Files/usr/local/etc/www/data:
       - action.php      - exec.php        - index.php
       - common.php      - file.php        - pen.php


+ data/nanobsd/Files/usr/local/etc/rc.d:
      - lvrouted.sh      - pen       - sortproxies
```

Toevoegingen aan /usr/local/lib. Extra lib-bestand voor de net-snmp configuratie, toegevoegd in verband met een verwijzing naar een oudere lib versie na de upgrade van de Perl port versie 5.8.8 naar 5.8.9

```
+ data/nanobsd/Files/usr/local/lib:
      - libperl.so

+ data/nanobsd/Files/usr/local/sbin:
      - dhcpd-snmp
```

Toevoegingen aan /usr/local/share

```
+ data/nanobsd/Files/usr/local/share:
      + snmp

+ data/nanobsd/Files/usr/local/share/snmp:
      + mibs            - snmp.conf       - snmpd.conf

+ data/nanobsd/Files/usr/local/share/snmp/mibs:
      - IEEE802dot11-MIB.txt
```

Appendix D , update-image-script

Deze bijlage bevat het update_image.sh script, dat gebruikt kan worden om een bestaande NanoBSD-image te voorzien van node-specifieke configuratiebestanden. De configuratie wordt opgehaald uit de Genesis / Exodus configuratiedatabase aan de hand van het config_update.sh script.

```
#!/bin/sh
# Mount / write NanoBSD image script, to update node-configuration.
# Script assumes /tools/config_update.sh is available in image

# Image mount point
MNT=/mnt
# Flashdevice
FD=/dev/da0

# Set image name/path
echo Enter NanoBSD image name...
read IMG

# MD device id
MD=`mdconfig -a -t vnode -f ${IMG}`

# Mount system environment in ${MNT}
mount -t ufs ${MD}s1a ${MNT}
# Mount configuration slice in ${MNT}/cfg
mount –t ufs ${MD}s3 ${MNT}/cfg

# Create directory structure for /etc
cd ${MNT}/cfg/ && mkdir local namedb
# Copy temporary files to configuration slice
cd ${MNT} && cp etc/rc.conf cfg/
cd ${MNT} && cp usr/local/etc/dhcpd-snmp.conf cfg/local/

# Copy /etc/resolv.conf from host, for internet access
cp /etc/resolv.conf ${MNT}/etc
# Fetch node-specific files from configuration db
chroot ${MNT} sh -c "/tools/config_update.sh"

# temporary scripts, should be obsolete later on.
# scripts to set node specific conf in rc.conf & dhcpd_snmp.conf
chroot ${MNT} sh -c "/tools/dhcpd_snmp.sh"
chroot ${MNT} sh -c "/tools/syslogd_flag.sh"

# Del resolv.conf from system environment
rm ${MNT}/etc/resolv.conf

# Umount ${MNT} en del vnode
umount $MNT
mdconfig -d -u $MD
```

```
# Delete vnode
mdconfig -d -u $MD
clear

# Choice, write image to flash or quit script.
while true
do
  echo -n "Write modified image to flash (da0) ? (y or n) :"
  read CHOICE
  case $CHOICE in
    y|Y|YES|yes|Yes) break ;;
    n|N|no|NO|No)
      echo Aborting, you entered $CHOICE
      exit
      ;;
    *) echo Please enter only y or n
  esac
done
clear

# Write image to flash device $FD
echo Please wait, writing image to flash...
dd if=${IMG} of=${FD} bs=64k

exit 0
```

Appendix E , lvrouted-rc-script

Deze bijlage bevat het nieuwe rc-script voor de lvrouted daemon. Het script is aangemaakt op basis van het originele opstartscript dat is opgenomen in binnen de lvrouted broncode in SVN. De configuratie is opgenomen in het FreeBSD rc-raamwerk en dient nu binnen `rc.conf` opgenomen te worden aan de hand van `lvrouted_flags`

```
#!/bin/sh
#
# PROVIDE: lvrouted
# REQUIRE: DAEMON
# BEFORE:  LOGIN

. /etc/rc.subr

lvrouted_enable=${lvrouted_enable:-"NO"}
lvrouted_flags=${lvrouted_flags:-}

name="lvrouted"
rcvar=`set_rcvar`
lvrouted="/usr/local/sbin/lvrouted.opt"

pidfile="/var/run/lvrouted.pid"
pid=`ps ax | grep lvrouted.opt | grep -v grep | sed "s/^ *//" | sed "s/ .*//g"`

start_cmd="lvrouted_start"
stop_cmd="lvrouted_stop"
restart_cmd="lvrouted_restart"

extra_commands="reload continue"

reload_cmd="lvrouted_reload"
continue_cmd="lvrouted_continue"

lvrouted_start(){
      if test "$pid" != ""; then
            echo "Already running" >&2
      else
            /sbin/sysctl net.inet.ip.forwarding=1
            $lvrouted $lvrouted_flags
      fi
}

lvrouted_continue(){
      if test "$pid" != ""; then
            kill -USR2 $pid
            kill $pid
            $lvrouted -r $lvrouted_flags
      else
            echo "Not running, will start now (ct)"
            $lvrouted $lvrouted_flags
      fi
}
```

```
lvrouted_restart(){
      if test "$pid" != ""; then
            kill -9 $pid
            route flush; route flush; route flush; route flush; route
flush
            $lvrouted $lvrouted_flags
      else
            echo "Not running, will start now (rs)"
            $lvrouted $lvrouted_flags
      fi
}

lvrouted_stop(){
      if test "$pid" != ""; then
            kill $pid
            echo "stopped"
      else
            echo "Not running, nothing to kill"
      fi
}


load_rc_config ${name}
run rc command "$1"
```

Appendix F , pen-sortproxies-script

Deze bijlage bevat het sortproxies-script dat wordt gebruikt voor de periodieke sortering van beschikbare proxy-servers op basis van de actuele bandbreedte.

```
#!/bin/sh

# PROVIDE: sort_proxies
# REQUIRE: sshd cron apache22 snmpd

. /etc/rc.subr

sort_proxies=${sort_proxies:-"NO"}
sort_proxies_list=${sort_proxies_list:-}

name="sort_proxies"
rcvar="sort_proxies"
start_cmd="sort_proxies_start"
stop_cmd=":"

url="http://www.ams-ix.net/"
pen="/usr/local/etc/rc.d/pen"
cfg="0:0:1:1"
prinr="0"

test_proxy()
{
      http_proxy="$1:$2"
      export http_proxy
      test=`fetch -T 10 -o /dev/null $3 2>&1`
      if [ $? = 0 ]; then
            echo "`echo "$test" |grep kBps | \
            sed 's/^.*(//' | awk '{print $4}'` $1" >> /tmp/proxylist
      fi
}
```

Met de bovenstaande functie “test_proxy” wordt een lijst met beschikbare proxies geproduceerd worden die zijn gesorteerd op de actuele banbreedte.
Aan de hand deze lijst wordt de actuele configuratie opgemaakt die periodiek wordt doorgevoerd aan Pen

```
sort_proxies_start()
{
      echo "Waiting 60 seconds to sort proxies and load PEN"
      sleep 60
      echo "" > /tmp/proxylist
      echo "" > /var/db/proxylist
      for host in ${sort_proxies_list}
            do
            ip=`echo $host | cut -d ":" -f1`
            port=`echo $host | cut -d ":" -f2`
            test_proxy $ip $port $url
            done

      proxylist=`sort -nr /tmp/proxylist | awk '{print $2}'`
      for proxy in $proxylist
            do
            prio="$((prinr=prinr+1))"
            echo "`echo $proxy":"$port":"$cfg":"$prio | \
            sed 's/^.*(//'`" >> /var/db/proxylist
            done

      $pen restart
}


load_rc_config ${name}
run_rc_command "$1"
```

De bovenstaande functie maak op basis van de gesorteerde lijst met proxy-server de actuele configuratie van Pen aan, en voert deze door aan de configuratie van de daemon.

Appendix G , manage-interfaces-script

Deze bijlage bevat het “manage_interfaces.sh” -script dat kan worden gebruikt ter ondersteuning van het beheer van de Ubiquiti NanoStations. Het script is geschreven om gebruikt te worden binnen het IRIS-concept.
Om het script te kunnen gebruiken dient op de NanoStations “key-based” SSH authenticatie beschikbaar te zijn voor de basis-node. Daarnaast gaat het script ervan uit dat de configuratiebestanden van de aangesloten NanoStations lokaal zijn opgeslagen binnen de /usr/local/etc/interfaces –structuur op de installatie van de basis-node. Indien dit niet het geval is, kan dit volgens de onderstaande stappen worden uitgevoerd.

- Maak op de NanoStations een schrijfbare home-directory aan voor de user “root” door de volgende regel toe te voegen aan het configuratiebestand.

```
# ssh root@nanostation1
# vi /tmp/system.cfg
....
users.1.homedir=/etc/persistent
....
```

- Voer deze wijzigingen door voor alle NanoStation en sla de configuratie met het volgende commando op

```
# cfgmtd -w -p /etc
# reboot
```

- Maak vervolgens een SSH public-key pair aan op de installatie basis-node

```
# mount –u –o rw /
# ssh-keygen
```

- Kopieer de aangemaakte public rsa-key vervolgens naar de NanoStation met het volgende commando

```
# ssh-copy-id –i /root/.ssh/id_rsa.pub root@nanostation1
```

- Open nogmaals een shell naar de aangesloten NanoStations en voer nogmaals de volgende commando’s uit om de toegevoegde SSH-key permanent op te slaan

```
# cfgmtd -w -p /etc
# reboot
```

- Als laatste dienen de configuratiebestanden van de NanoStation opgeslagen te worden op de installatie van basis-node te worden in de volgende structuur

```
# mount /cfg && mkdir /cfg/local/interfaces
# cd /cfg/local/interface && mkdir 01 && mkdir 02 && mkdir 03
# scp root@nanostation1 /tmp/system.cfg /cfg/local/interfaces/01/
# scp root@nanostation2 /tmp/system.cfg /cfg/local/interfaces/02/
# scp root@nanostation3 /tmp/system.cfg /cfg/local/interfaces/03/
# cd / && umount /cfg && shutdown –r now
```

Vervolgens kan het manage_interfaces.sh script worden gebruikt.

```
#!/bin/sh

# SSH/SCP user
USR="root"
# SSH path
SSH="/usr/bin/ssh"
# SCP path
SCP="/usr/bin/scp"
# SNMP path/cmd
SNMP="/usr/local/bin/snmpget -v 1 -c public -Oqv"
# Interface to query
IF="ath0"
# Interlink/interface configuration directory
ILDIR="/usr/local/etc/interfaces"


get_iface_config()
{
      echo "Fetching current configuration..."
      $SCP $USR@$HOST:/tmp/system.cfg $ILDIR/$ILINK
}

save_iface_config()
{
      echo "Storing modified configuration..."
      $SCP $ILDIR/$ILINK $USR@$HOST:/tmp
      $SSH $USR@$HOST cfgmtd -w -p /etc
}

restart_iface()
{
      echo "Restarting interface..."
      $SSH $USR@$HOST reboot
}
```

De bovenstaande functies kunnen respectievelijk worden gebruikt om de configuratie van de NanoStation op te halen, weg te schrijven en daarnaast het systeem te herstarten.
vervolg van het script op de volgende pagina.

```
get_iface_stats()
{
      # SSH / SNMP connect cmd
      _CMD_SSH="$SSH $USR@$HOST"
      _CMD_SNMP="$SNMP $HOST"

      echo "Processing selected interface..."

      # Data to query interface for
      SSID="$($_CMD_SNMP 1.2.840.10036.1.1.1.9.6)"
      CHAN="$($_CMD_SNMP 1.2.840.10036.4.5.1.1.6)"
      FREQ="$($_CMD_SSH iwconfig $IF|grep Freq|cut -d":" -f3|cut -d" " -f1)"
      SIGN="$($_CMD_SSH iwconfig $IF|grep Signal|cut -d"=" -f3|cut -d" " -f1)"
      RATE="$($_CMD_SSH iwconfig $IF|grep Rate|cut -d"M" -f1|cut -b20-22)"
      TxByte="$($_CMD_SNMP 1.3.6.1.2.1.2.2.1.16.6)"
      RxByte="$($_CMD_SNMP 1.3.6.1.2.1.2.2.1.10.6)"
      # Note, RSSI value by Atheros standard.
      RSSI="$((test=SIGN+96))"

      # Print data on screen
      echo ""
      echo "      # $IF link statistics for $SSID"
      echo ""
      echo "      Current channel:  $CHAN"
      echo "      Channel frequency:      $FREQ MHz"
      echo "      Signal strength:  $SIGN dBm"
      echo "      Signal RSSI:            $RSSI"
      echo "      TxRate:           $RATE Mbits/s"
      echo ""
      echo "      Total data troughput Tx/Rx"
      echo ""
      echo "      Bytes Transmitted:      $TxByte Bytes"
      echo "      Bytes Received:   $RxByte Bytes"
      echo ""
}
```

Aan de hand van de bovenstaande functie kan een totaal of individueel overzicht worden opgevraagd van de aangesloten NanoStation interfaces.
Vervolg van het script op de volgende pagina.

```
# List available interfaces by ssid.
nr="0"
echo ""
ILINKS=`ls ${ILDIR}`
for dir in ${ILINKS}
do
  ilnr="$((nr=nr+1))"
  SSID=`cat $ILDIR/$dir/system.cfg|grep wireless.1.ssid|cut -d"=" -
f2`
  echo "$ilnr $SSID"
done

# Select interface to perfom task
echo ""
echo "Select desired interface/interlink: 1/2/3"
echo ""
read answer
case $answer in
    1)    ILINK=/01/system.cfg;;
    2)    ILINK=/02/system.cfg;;
    3)    ILINK=/03/system.cfg;;
    *)    echo You did not choose 1, 2 or 3 exit 0;;
esac
clear

# Set selected SSID
SSID=`cat $ILDIR/$ILINK|grep wireless.1.ssid|cut -d"=" -f2`

# Select desired task.
echo "Select task for interface $SSID:"
echo ""
echo "1. Get link statistics"
echo "2. Get interface configuration"
echo "3. Save interface configuration"
echo "4. Restart"
echo ""
read answer
case $answer in
    1)    TASK=get_iface_stats;;
    2)    TASK=get_iface_config;;
    3)    TASK=save_iface_config;;
    4)    TASK=restart_iface;;
    *)    echo You did not choose 1, 2 or 3 exit 0;;
esac
clear

# Set host ip
HOST=`cat $ILDIR/$ILINK|grep netconf.3.ip|cut -d"=" -f2`

# Run task
$TASK

exit 0
```