HiSPARC Server Setup Documentation

Release 0.1

HiSPARC team

June 28, 2013

CONTENTS

Contents:

CHAPTER

ONE

INSTALLATION OF FROME

Granting davidf rights to manage software and services:

```
(root)$ visudo
```

and adding:

```
davidf  ALL = SOFTWARE, SERVICES
```

Preparing for source install:

```
(root)$ cd /usr/local/src/
(root)$ mkdir hisparc
(root)$ chown davidf.hisparc hisparc/
$ chmod g+w hisparc/
```

In /etc/ld.so.conf.d new file usrlocal.conf, to let ldconfig find libraries of locally installed software:

```
/usr/local/lib
```

1.1 Python

Python:

```
$ cd /usr/local/src/hisparc
$ wget http://www.python.org/ftp/python/2.6.4/Python-2.6.4.tgz
$ tar xvzf Python-2.6.4.tgz
$ cd Python-2.6.4
$ ./configure --enable-shared
$ make
(root)$ make install
```

Then, run:

```
(root)$ ldconfig
```

Now, the python libraries are registered.

1.2 Python Setuptools

From egg:

```
$ cd /usr/local/src/hisparc
$ wget http://pypi.python.org/packages/2.6/s/setuptools/setuptools-0.6c11-py2.6.egg#md5=
(root)$ sh setuptools-0.6c11-py2.6.egg
```

1.3 IPython, an interactive Python shell

Download and install IPython:

```
(root)$ easy_install ipython
```

1.4 Web server

Install apache development libraries:

```
$ sudo yum install httpd-devel
```

```
================================================================================
 Package              Arch         Version                  Repository        Size
================================================================================
Installing:
 httpd-devel          i386         2.2.3-31.sl5.2           sl-security      147 k
 httpd-devel          x86_64       2.2.3-31.sl5.2           sl-security      147 k
Installing for dependencies:
 apr                  x86_64       1.2.7-11.el5_3.1         sl-security      118 k
 apr-devel            x86_64       1.2.7-11.el5_3.1         sl-security      237 k
 apr-util             x86_64       1.2.7-7.el5_3.2          sl-security       74 k
 apr-util-devel       x86_64       1.2.7-7.el5_3.2          sl-security       53 k
 httpd                x86_64       2.2.3-31.sl5.2           sl-security      1.2 M
```

Change configuration in /etc/httpd/conf/httpd.conf. Patch:

```
--- httpd.conf.orig     2009-12-04 14:35:39.000000000 +0100
+++ httpd.conf  2009-12-04 14:35:50.000000000 +0100
@@ -228,8 +228,8 @@
 #  when the value of (unsigned)Group is above 60000;
 #  don't use Group #-1 on these systems!
 #
-User apache
-Group apache
+User www
+Group www

 ### Section 2: 'Main' server configuration
 #
```

Enabling httpd on startup:

```
$ sudo /sbin/chkconfig --add httpd
$ sudo /sbin/chkconfig --levels 35 httpd on
```

Starting httpd now:

```
$ sudo /sbin/service httpd start
```

For mod_wsgi:

```
$ cd /usr/local/src/hisparc
$ wget http://modwsgi.googlecode.com/files/mod_wsgi-3.1.tar.gz
$ tar xvzf mod_wsgi-3.1.tar.gz
$ cd mod_wsgi-3.1
$ ./configure
$ make
(root)$ make install
```

Change configuration in /etc/httpd/conf/httpd.conf. Patch:

```
--- httpd.conf.orig     2009-12-04 15:19:01.000000000 +0100
+++ httpd.conf  2009-12-04 15:34:30.000000000 +0100
@@ -197,6 +197,7 @@
 LoadModule mem_cache_module modules/mod_mem_cache.so
 LoadModule cgi_module modules/mod_cgi.so
 LoadModule version_module modules/mod_version.so
+LoadModule wsgi_module modules/mod_wsgi.so

 #
 # The following modules are not loaded by default:
```

Restarting apache:

```
$ sudo /sbin/service httpd restart
```

1.5 Version control

Install bazaar from source:

```
$ cd /usr/local/src/hisparc
$ wget http://launchpad.net/bzr/2.0/2.0.2/+download/bzr-2.0.2.tar.gz
$ tar xvzf bzr-2.0.2.tar.gz
$ cd bzr-2.0.2
(root)$ python setup.py install
```

1.5.1 Paramiko

Paramiko supports ssh2 for python, which is needed to do a checkout of our application's sources over sftp. Install using easy_install:

```
(root)$ easy_install paramiko
```

This will automatically download, compile and install dependencies (pycrypto).

1.6 Datastore web application

The datastore application is driving our central data storage solution. It is a pure python implementation under complete version control.

1.6.1 Prerequisites

The datastore application uses PyTables and the underlying HDF5 library to store binary data files. PyTables depends heavily on NumPy.:

```
(root)$ easy_install numpy
```

This gives an error:

```
/tmp/easy_install-JePGOA/numpy-1.4.0rc1/numpy/distutils/misc_util.py:248: RuntimeWarning
Error in atexit._run_exitfuncs:
Traceback (most recent call last):
  File "/usr/local/lib/python2.6/atexit.py", line 24, in _run_exitfuncs
    func(*targs, **kargs)
  File "/tmp/easy_install-JePGOA/numpy-1.4.0rc1/numpy/distutils/misc_util.py", line 248,
ImportError: No module named numpy.distutils
Error in sys.exitfunc:
Traceback (most recent call last):
  File "/usr/local/lib/python2.6/atexit.py", line 24, in _run_exitfuncs
    func(*targs, **kargs)
  File "/tmp/easy_install-JePGOA/numpy-1.4.0rc1/numpy/distutils/misc_util.py", line 248,
ImportError: No module named numpy.distutils
```

So, rerun the command, this time without errors:

```
(root)$ easy_install numpy
```

Now:

```
$ cd /usr/local/src/hisparc
$ wget http://www.hdfgroup.org/ftp/HDF5/prev-releases/hdf5-1.8.3/src/hdf5-1.8.3.tar.gz
$ tar xvzf hdf5-1.8.3.tar.gz
$ cd hdf5-1.8.3
$ ./configure --prefix=/usr/local
$ make
(root)$ make install
(root)$ ldconfig
```

And, finally, install PyTables itself:

```
(root)$ easy_install tables
```

1.6.2 Setting up datastore

In summary:

- Created a /var/www/wsgi-bin directory from which to run the wsgi applications
- Created a subdirectory owned by davidf.hisparc inside this wsgi-bin
- Did a checkout of the datastore sources inside the subdirectory
- Made a local copy of the application into the parent (wsgi-bin) and edited to set the correct local full path
- Added the wsgi application to the Apache configuration

Here we go:

```
(root)$ cd /var/www
(root)$ mkdir wsgi-bin
(root)$ cd wsgi-bin
(root)$ mkdir datastore
(root)$ chown davidf.hisparc datastore
(root)$ chmod g+w datastore
$ cd /var/www/wsgi-bin/datastore/
$ bzr co sftp://admhispa@login.nikhef.nl/project/hisparc/bzr/datastore/trunk .
```

Copy the application.wsgi and config.ini from the examples directory:

```
(root)$ cd /var/www/wsgi-bin
(root)$ cp datastore/examples/application.wsgi datastore.wsgi
(root)$ cp datastore/examples/config.ini datastore/
(root)$ chown davidf.hisparc datastore.wsgi datastore/config.ini
(root)$ chmod g+w datastore.wsgi datastore/config.ini
```

Edited /var/www/wsgi-bin/datastore.wsgi and set the correct paths:

```
sys.path.append('/var/www/wsgi-bin/datastore/wsgi')
configfile = ('/var/www/wsgi-bin/datastore/config.ini')
```

The config.ini now reads:

```
[General]
log=/var/log/hisparc/hisparc.log
loglevel=debug
station_list=/databases/frome/station_list.csv
data_dir=/databases/frome

[Writer]
sleep=1
```

I had to create the appropriate directory in /var/log and grant rights:

```
(root)$ cd /var/log
(root)$ mkdir hisparc
(root)$ chown www.hisparc hisparc
(root)$ chmod g+w hisparc
```

Then, added datastore to the Apache configuration:

```
(root)$ cd /etc/httpd/conf.d/
(root)$ touch hisparc.conf
(root)$ chown davidf.hisparc hisparc.conf
(root)$ chmod g+w hisparc.conf
```

And edited hisparc.conf to contain:

```
WSGIScriptAlias /hisparc/upload /var/www/wsgi-bin/datastore.wsgi
```

Reload Apache configuration:

```
$ sudo /sbin/service httpd reload
```

1.7 TODO

Writer app!

1.8 (Maybe) Not relevant

install: yum-utils easy_install dozer easy_install pil (requirement of dozer) easy_install mysql-python (for migration) install: gcc-gfortran easy_install virtualenvwrapper install: blas-devel lapack-devel (for scipy)

CHAPTER

TWO

INSTALLATION OF PIQUE

Granting davidf rights to manage software and services:

```
(root)$ visudo
```

and adding:

```
davidf  ALL = SOFTWARE, SERVICES
```

Preparing for source install:

```
(root)$ cd /localstore
(root)$ mkdir -p usr/local
(root)$ mv /usr/local/src usr/local
(root)$ cd /usr/local
(root)$ ln -s /localstore/usr/local/src .
(root)$ cd /usr/local/src/
(root)$ mkdir hisparc
(root)$ chown davidf.hisparc hisparc/
$ chmod g+w hisparc/
```

In /etc/ld.so.conf.d new file usrlocal.conf, to let ldconfig find libraries of locally installed software:

```
/usr/local/lib
```

2.1 Python

Python:

```
$ cd /usr/local/src/hisparc
$ wget http://www.python.org/ftp/python/2.6.4/Python-2.6.4.tgz
$ tar xvzf Python-2.6.4.tgz
$ cd Python-2.6.4
$ ./configure --enable-shared
$ make
(root)$ make install
```

Then, run:

```
(root)$ ldconfig
```

Now, the python libraries are registered.

2.2 Python Setuptools

From egg:

```
$ cd /usr/local/src/hisparc
$ wget http://pypi.python.org/packages/2.6/s/setuptools/setuptools-0.6c11-py2.6.egg#md5=
(root)$ sh setuptools-0.6c11-py2.6.egg
```

2.3 IPython, an interactive Python shell

Download and install IPython:

```
(root)$ easy_install ipython
```

2.4 Web server

Install apache development libraries:

```
$ sudo yum install httpd-devel
```

```
================================================================================
 Package              Arch        Version                 Repository        Size
================================================================================
Installing:
 httpd-devel          i386        2.2.3-31.sl5.2          sl-security      147 k
 httpd-devel          x86_64      2.2.3-31.sl5.2          sl-security      147 k
Installing for dependencies:
 apr                  x86_64      1.2.7-11.el5_3.1        sl-security      118 k
 apr-devel            x86_64      1.2.7-11.el5_3.1        sl-security      237 k
 apr-util             x86_64      1.2.7-7.el5_3.2         sl-security       74 k
 apr-util-devel       x86_64      1.2.7-7.el5_3.2         sl-security       53 k
 httpd                x86_64      2.2.3-31.sl5.2          sl-security      1.2 M
```

Change configuration in /etc/httpd/conf/httpd.conf. Patch:

```
--- httpd.conf.orig     2009-12-04 14:35:39.000000000 +0100
+++ httpd.conf  2009-12-04 14:35:50.000000000 +0100
@@ -228,8 +228,8 @@
 #  when the value of (unsigned)Group is above 60000;
 #  don’t use Group #-1 on these systems!
 #
-User apache
-Group apache
+User www
+Group www

 ### Section 2: ’Main’ server configuration
 #
```

Enabling httpd on startup:

```
$ sudo /sbin/chkconfig --add httpd
$ sudo /sbin/chkconfig --levels 35 httpd on
```

Starting httpd now:

```
$ sudo /sbin/service httpd start
```

For mod_wsgi:

```
$ cd /usr/local/src/hisparc
$ wget http://modwsgi.googlecode.com/files/mod_wsgi-3.1.tar.gz
$ tar xvzf mod_wsgi-3.1.tar.gz
$ cd mod_wsgi-3.1
$ ./configure
$ make
(root)$ make install
```

Change configuration in /etc/httpd/conf/httpd.conf. Patch:

```
--- httpd.conf.orig     2009-12-04 15:19:01.000000000 +0100
+++ httpd.conf  2009-12-04 15:34:30.000000000 +0100
@@ -197,6 +197,7 @@
 LoadModule mem_cache_module modules/mod_mem_cache.so
 LoadModule cgi_module modules/mod_cgi.so
 LoadModule version_module modules/mod_version.so
+LoadModule wsgi_module modules/mod_wsgi.so

 #
 # The following modules are not loaded by default:
```

Restarting apache:

```
$ sudo /sbin/service httpd restart
```

2.5 Version control

Install bazaar from source:

```
$ cd /usr/local/src/hisparc
$ wget http://launchpad.net/bzr/2.0/2.0.2/+download/bzr-2.0.2.tar.gz
$ tar xvzf bzr-2.0.2.tar.gz
$ cd bzr-2.0.2
(root)$ python setup.py install
```

2.5.1 Paramiko

Paramiko supports ssh2 for python, which is needed to do a checkout of our application's sources over sftp. Install using easy_install:

```
(root)$ easy_install paramiko
```

This will automatically download, compile and install dependencies (pycrypto).

2.6 Public database web application

The public database blablabla. It is a pure python implementation under complete version control.

2.6.1 Prerequisites

The public database application uses PyTables and the underlying HDF5 library to read binary data files. PyTables depends heavily on NumPy.:

```
(root)$ easy_install numpy
```

This gives an error:

```
/tmp/easy_install-JePGOA/numpy-1.4.0rc1/numpy/distutils/misc_util.py:248: RuntimeWarning
Error in atexit._run_exitfuncs:
Traceback (most recent call last):
  File "/usr/local/lib/python2.6/atexit.py", line 24, in _run_exitfuncs
    func(*targs, **kargs)
  File "/tmp/easy_install-JePGOA/numpy-1.4.0rc1/numpy/distutils/misc_util.py", line 248,
ImportError: No module named numpy.distutils
Error in sys.exitfunc:
Traceback (most recent call last):
  File "/usr/local/lib/python2.6/atexit.py", line 24, in _run_exitfuncs
    func(*targs, **kargs)
  File "/tmp/easy_install-JePGOA/numpy-1.4.0rc1/numpy/distutils/misc_util.py", line 248,
ImportError: No module named numpy.distutils
```

So, rerun the command, this time without errors:

```
(root)$ easy_install numpy
```

Now:

```
$ cd /usr/local/src/hisparc
$ wget http://www.hdfgroup.org/ftp/HDF5/prev-releases/hdf5-1.8.3/src/hdf5-1.8.3.tar.gz
$ tar xvzf hdf5-1.8.3.tar.gz
$ cd hdf5-1.8.3
$ ./configure --prefix=/usr/local
$ make
(root)$ make install
(root)$ ldconfig
```

And, finally, install PyTables itself:

```
(root)$ easy_install tables
```

The public databases graphing capabilities come from Enthought Chaco, a python plotting library. It needs swig to build. Install with:

```
$ wget http://prdownloads.sourceforge.net/swig/swig-1.3.40.tar.gz
$ tar xvzf swig-1.3.40.tar.gz
$ cd swig-1.3.40
$ ./configure
$ make
(root)$ make install
```

It also needs a GUI toolkit, like wxPython:

```
$ wget http://downloads.sourceforge.net/wxpython/wxPython-src-2.8.10.1.tar.bz2
$ tar xvjf wxPython-src-2.8.10.1.tar.bz2
$ cd wxPython-src-2.8.10.1
$ ./configure --enable-unicode --with-opengl
$ make && make -C contrib/src/gizmos && make -C contrib/src/stc
```

```
(root)$ make install && make -C contrib/src/gizmos install && make -C contrib/src/stc in
$ cd wxPython/src/gtk
$ patch < /usr/local/src/hisparc/gdi.patch
$ cd ../..
(root)$ python setup.py install
```

The contents of the aforementioned gdi.patch is:

```
--- wxPython/src/gtk/_gdi_wrap.cpp.orig 2009-08-08 16:26:48.000000000 +0200
+++ wxPython/src/gtk/_gdi_wrap.cpp      2009-08-08 16:32:50.000000000 +0200
@@ -4195,6 +4195,10 @@
     virtual wxGraphicsBrush CreateRadialGradientBrush(wxDouble , wxDouble , wxDouble ,
                                                        const wxColour &, const wxColour
     virtual wxGraphicsFont CreateFont( const wxFont & , const wxColour & ) { return wxN
+
+    // patch required as explained in
+    // http://groups.google.com/group/wxPython-users/browse_thread/thread/129ba27e2f868
+    wxGraphicsBitmap CreateBitmap( const wxBitmap &bitmap ) const { return wxNullGraphi
 };
```

We currently run Chaco straight out of the subversion repository. Once a new release has been finalized, we might go back to simply install from PyPI. Now, however, we have to issue:

```
(root)$ easy_install etsprojecttools
$ ets co Chaco
(root)$ ets install Chaco_3.2.1
```

2.6.2 Setting up the public database

In summary:

Here we go:

```
$ cd /usr/local/src/hisparc
$ bzr co sftp://admhispa@login.nikhef.nl/project/hisparc/bzr/publicdb/trunk publicdb
(root)$ cd /var/www
(root)$ mkdir django_publicdb
(root)$ chown davidf.hisparc django_publicdb/
$ ln -s /usr/local/src/hisparc/publicdb/django_publicdb/* .
$ cp --remove-destination /usr/local/src/hisparc/publicdb/django_publicdb/settings.py .
$ cp --remove-destination /usr/local/src/hisparc/publicdb/django_publicdb/manage.py .
$ cp /usr/local/src/hisparc/publicdb/examples/django.wsgi .
```

And edit django.wsgi so that it contains the right system path:

```
sys.path.append('/var/www')
```

Then, added the public database to the Apache configuration:

```
(root)$ cd /etc/httpd/conf.d/
(root)$ touch hisparc.conf
(root)$ chown davidf.hisparc hisparc.conf
(root)$ chmod g+w hisparc.conf
```

And edit hisparc.conf to contain:

```
RedirectMatch ^/$ http://data.hisparc.nl/django/

WSGIScriptAlias /django /var/www/django_publicdb/django.wsgi
WSGIPythonEggs /tmp

Alias /django/media /usr/local/lib/python2.6/site-packages/Django-1.1.1-py2.6.egg/djang
```

Reload Apache configuration:

```
$ sudo /sbin/service httpd reload
```

2.7 TODO

South.

```
mkdir /var/www/media
chown www.www media
ln -s /var/www/django_publicdb/static media

cd /usr/local/bin
cp /usr/local/src/hisparc/publicdb/examples/django-cron.py hisparc-update

# Run a daily check for new events, but it _must_ be a few hours after
# midnight, so don't place this script in cron.daily, just to be sure.
0 4 * * * root /usr/local/bin/hisparc-update

python PIL

django cron script on pique, changed a bit?
```

CHAPTER

THREE

INSTALLATION OF TIETAR

Note: New nagios.cfg config and postfix config, template.cfg, shorewall rules

This server was originally installed by Tristan in May 2008. Only recently did David start making changes to the system. Later changes are documented here. Hopefully, they will be expanded to include a description of the complete system.

Due to a partial disk crash February 18th, 2010, we reinstalled the system. Due to lack of time, a lot of the original configuration was retrieved from backups without analyzing the design.

3.1 Installation

Adding user davidf:

```
(root)$ adduser davidf
(root)$ passwd davidf
```

Granting davidf rights to manage software and services:

```
(root)$ visudo
```

and adding:

```
davidf  ALL = SOFTWARE, SERVICES
```

3.2 Adding the hisparc group

We've added the hisparc group to the system and made a few users part of it:

```
(root)$ groupadd hisparc
(root)$ usermod -G hisparc davidf
```

3.3 Preparing for source install

Issue:

```
(root)$ cd /usr/local/src/
(root)$ mkdir hisparc
(root)$ chown davidf.hisparc hisparc/
$ chmod g+sw hisparc/
```

In /etc/ld.so.conf.d new file usrlocal.conf, to let ldconfig find libraries of locally installed software:

```
/usr/local/lib
```

Then, install a compiler:

```
$ sudo yum install gcc
```

3.4 Setting up RPMForge

RPMForge provides extra packages for CentOS, including Nagios and more recent versions of the SSL libraries. To enable it:

```
$ cd /usr/local/src/hisparc
$ wget http://packages.sw.be/rpmforge-release/rpmforge-release-0.5.1-1.el5.rf.i386.rpm
$ sudo rpm --import http://dag.wieers.com/rpm/packages/RPM-GPG-KEY.dag.txt
$ rpm -K rpmforge-release-0.5.1-1.el5.rf.*.rpm
$ sudo rpm -i rpmforge-release-0.5.1-1.el5.rf.*.rpm
```

Check succesful installation with and update packages:

```
$ sudo yum check-update
$ sudo yum update
```

3.5 Python

Prerequisites for standard libraries:

```
$ sudo yum install zlib-devel
$ sudo yum install bzip2-devel
```

Python:

```
$ cd /usr/local/src/hisparc
$ wget http://www.python.org/ftp/python/2.6.4/Python-2.6.4.tgz
$ tar xvzf Python-2.6.4.tgz
$ cd Python-2.6.4
$ ./configure --enable-shared
$ make
(root)$ make install
```

Then, run:

```
(root)$ ldconfig
```

Now, the python libraries are registered.

3.6 Python Setuptools

From egg:

```
$ cd /usr/local/src/hisparc
$ wget http://pypi.python.org/packages/2.6/s/setuptools/setuptools-0.6c11-py2.6.egg#md5=
(root)$ sh setuptools-0.6c11-py2.6.egg
```

3.7 Web server

Install apache:

```
$ sudo yum install httpd
```

```
================================================================================
 Package                Arch       Version                      Repository     Size
================================================================================
Installing:
 httpd                  i386       2.2.3-31.el5.centos.2        updates       1.2 M
Installing for dependencies:
 apr                    i386       1.2.7-11.el5_3.1             base          123 k
 apr-util               i386       1.2.7-7.el5_3.2              base           76 k
 postgresql-libs        i386       8.1.18-2.el5_4.1             updates       196 k
```

Enabling httpd on startup:

```
$ sudo /sbin/chkconfig --levels 35 httpd on
```

Starting httpd now:

```
$ sudo /sbin/service httpd start
```

3.8 OpenVPN

Install OpenVPN from source, as we require version 2.1.1, which has no official RPM:

```
$ sudo yum install lzo2-devel
$ sudo yum install openssl-devel
$ wget http://openvpn.net/release/openvpn-2.1.1.tar.gz
$ tar xvzf openvpn-2.1.1.tar.gz
$ cd openvpn-2.1.1
$ ./configure
$ make
(root)$ make install
```

Blindly copy old configuration, but changed one directory name:

```
(root)$ cp -r /mnt/oldroot/etc/openvpn/* .
$ cd /etc/openvpn
(root)$ mv easy-rsa easy_rsa
```

To add OpenVPN as a service and start it:

```
$ cd /usr/local/src/hisparc/openvpn-2.1.1/sample-scripts/
(root)$ cp openvpn.init /etc/init.d/openvpn
$ sudo /sbin/chkconfig --add openvpn
$ sudo /sbin/service openvpn start
```

3.9 Dnsmasq

Dnsmasq handles our DNS requirements. On this system, it was already installed. Edited configuration, with the following resulting diff:

```
--- dnsmasq.conf.orig   2010-02-22 10:59:01.000000000 +0100
+++ dnsmasq.conf        2010-02-25 13:43:19.000000000 +0100
@@ -13,7 +13,7 @@
 # Never forward plain names (without a dot or domain part)
 #domain-needed
 # Never forward addresses in the non-routed address spaces.
-#bogus-priv
+bogus-priv


 # Uncomment this to filter useless windows-originated DNS requests
@@ -26,7 +26,7 @@

 # Change this line if you want dns to get its upstream servers from
 # somewhere other that /etc/resolv.conf
-#resolv-file=
+resolv-file=/etc/resolv.conf-nikhef

 # By  default,  dnsmasq  will  send queries to any of the upstream
 # servers it knows about and tries to favour servers to are  known
@@ -55,6 +55,7 @@
 # Add local-only domains here, queries in these domains are answered
 # from /etc/hosts or DHCP only.
 #local=/localnet/
+local=/his/

 # Add domains which you want to force to an IP address here.
 # The example below send any host in doubleclick.net to a local
@@ -85,6 +86,7 @@
 #interface=
 # Or you can specify which interface _not_ to listen on
 #except-interface=
+except-interface=eth0
 # Or which to listen on by address (remember to include 127.0.0.1 if
 # you use this.)
 #listen-address=
@@ -108,10 +110,11 @@
 # or if you want it to read another file, as well as /etc/hosts, use
 # this.
 #addn-hosts=/etc/banner_add_hosts
+addn-hosts=/etc/hosts-hisparc

 # Set this (and domain: see below) if you want to have a domain
 # automatically added to simple names in a hosts-file.
-#expand-hosts
```

```
+expand-hosts

 # Set the domain for dnsmasq. this is optional, but if it is set, it
 # does the following things.
@@ -121,6 +124,7 @@
 #    domain of all systems configured by DHCP
 # 3) Provides the domain part for "expand-hosts"
 #domain=thekelleys.org.uk
+domain=his

 # Set a different domain for a particular subnet
 #domain=wireless.thekelleys.org.uk,192.168.2.0/24
```

Copy /etc/resolv.conf to /etc/resolv.conf-nikhef and edit /etc/resolv.conf to contain:

```
search nikhef.nl his
nameserver 127.0.0.1
```

Enabling dnsmasq on startup and start it for the first time:

```
$ sudo /sbin/chkconfig --level 35 dnsmasq on
$ sudo /sbin/service dnsmasq start
```

3.10 Nagios

Install nagios from RPMForge:

```
$ sudo yum install nagios nagios-plugins nagios-plugins-nrpe nagios-nsca
$ sudo /sbin/chkconfig --level 35 nsca on
```

Edited several configuration files:

```
--- nagios.conf.orig    2010-02-22 13:50:14.000000000 +0100
+++ /etc/httpd/conf.d/nagios.conf 2010-02-22 13:50:31.000000000 +0100
@@ -17,10 +17,10 @@
 #  Order deny,allow
 #  Deny from all
 #  Allow from 127.0.0.1
-   AuthName "Nagios Access"
-   AuthType Basic
-   AuthUserFile /etc/nagios/htpasswd.users
-   Require valid-user
+#   AuthName "Nagios Access"
+#   AuthType Basic
+#   AuthUserFile /etc/nagios/htpasswd.users
+#   Require valid-user
 </Directory>

 Alias /nagios "/usr/share/nagios"
@@ -34,9 +34,9 @@
 #  Order deny,allow
 #  Deny from all
 #  Allow from 127.0.0.1
-   AuthName "Nagios Access"
-   AuthType Basic
-   AuthUserFile /etc/nagios/htpasswd.users
-   Require valid-user
+#   AuthName "Nagios Access"
+#   AuthType Basic
+#   AuthUserFile /etc/nagios/htpasswd.users
+#   Require valid-user
 </Directory>


--- cgi.cfg.orig        2010-02-22 13:41:05.000000000 +0100
+++ /etc/nagios/cgi.cfg 2010-02-26 11:44:01.000000000 +0100
@@ -105,6 +105,7 @@
 # server will inherit all rights you assign to this user!

 #default_user_name=guest
+default_user_name=nagiosadmin



@@ -272,7 +273,7 @@
 # This option allows you to specify the refresh rate in seconds
 # of various CGIs (status, statusmap, extinfo, and outages).

-refresh_rate=90
+refresh_rate=30


--- nagios.cfg.orig     2010-02-22 13:37:45.000000000 +0100
+++ /etc/nagios/nagios.cfg  2010-02-22 15:05:03.000000000 +0100
@@ -33,7 +33,7 @@
 cfg_file=/etc/nagios/objects/templates.cfg

 # Definitions for monitoring the local (Linux) host
-cfg_file=/etc/nagios/objects/localhost.cfg
+#cfg_file=/etc/nagios/objects/localhost.cfg

 # Definitions for monitoring a Windows machine
 #cfg_file=/etc/nagios/objects/windows.cfg
@@ -44,6 +44,9 @@
 # Definitions for monitoring a network printer
 #cfg_file=/etc/nagios/objects/printer.cfg

+# Definitions for HiSPARC
+cfg_file=/etc/nagios/objects/hisparc.cfg
+

 # You can also tell Nagios to process all config files (with a .cfg
 # extension) in a particular directory by using the cfg_dir


--- nsca.cfg.orig       2010-02-22 15:38:01.000000000 +0100
+++ /etc/nagios/nsca.cfg    2010-02-22 15:38:06.000000000 +0100
@@ -187,5 +187,5 @@
 #      26 = SAFER+
 #

-decryption_method=1
+decryption_method=0
```

```
--- commands.cfg.orig   2010-02-22 15:06:44.000000000 +0100
+++ /etc/nagios/objects/commands.cfg        2010-02-22 15:18:59.000000000 +0100
@@ -237,4 +237,19 @@
        command_line    /usr/bin/printf "%b" "$LASTSERVICECHECK$\t$HOSTNAME$\t$SERVICEDE
        }

+# NRPE!

+define command{
+        command_name check_nrpe
+        command_line $USER1$/check_nrpe -t 30 -H $HOSTADDRESS$ -c $ARG1$ -a $ARG2$ $ARG
+}
+
+define command{
+        command_name check_mysql
+        command_line $USER1$/check_mysql -H $HOSTADDRESS$ -u $ARG1$ -p $ARG2$
+}
+
+define command{
+        command_name check_dummy
+        command_line $USER1$/check_dummy $ARG1$ $ARG2$
+}
```

Reload apache configuration and start nagios:

```
$ sudo /sbin/service httpd reload
$ sudo /sbin/service nagios start
$ sudo /sbin/service nsca start
```

3.11 Version control

Install bazaar from source:

```
$ cd /usr/local/src/hisparc
$ wget http://launchpad.net/bzr/2.1/2.1.0/+download/bzr-2.1.0.tar.gz
$ tar xvzf bzr-2.1.0.tar.gz
$ cd bzr-2.1.0
(root)$ python setup.py install
```

3.11.1 Paramiko

Paramiko supports ssh2 for python, which is needed to do a checkout of our application's sources over sftp. Install using easy_install:

```
(root)$ easy_install paramiko
```

This will automatically download, compile and install dependencies (pycrypto).

3.12 Setting up the HiSPARC public database scripts

First, do a checkout of the public database sources:

```
$ cd /usr/local/src/hisparc
$ bzr co sftp://admhispa@login.nikhef.nl/project/hisparc/bzr/publicdb/trunk publicdb
```

Symlink the vpn server example scripts into /usr/local/bin:

```
(root)$ ln -s /usr/local/src/hisparc/publicdb/examples/create_admin_keys.sh .
(root)$ ln -s /usr/local/src/hisparc/publicdb/examples/create_keys.sh .
(root)$ ln -s /usr/local/src/hisparc/publicdb/examples/vpn-cron.py hisparc-nagios
(root)$  ln -s /usr/local/src/hisparc/publicdb/examples/vpn-xmlrpc-server.py hisparcvpnd
```

And set execute permissions:

```
$ cd /usr/local/src/hisparc/publicdb/examples
$ chmod +x vpn-cron.py
$ chmod +x vpn-xmlrpc-server.py
```

Change some paths and host information, resulting in the following diff:

```
=== modified file 'examples/vpn-cron.py'
--- examples/vpn-cron.py        2010-01-15 21:36:15 +0000
+++ examples/vpn-cron.py        2010-02-22 11:32:43 +0000
@@ -1,4 +1,4 @@
-#!/usr/bin/python
+#!/usr/local/bin/python
 """ Reload nagios if necessary

     This script checks for the existence of the nagios restart flag,

=== modified file 'examples/vpn-xmlrpc-server.py'
--- examples/vpn-xmlrpc-server.py       2010-01-15 14:31:24 +0000
+++ examples/vpn-xmlrpc-server.py       2010-02-22 11:35:27 +0000
@@ -1,4 +1,4 @@
-#!/usr/bin/python
+#!/usr/local/bin/python
 """ Simple XML-RPC Server to run on the VPN server

     This daemon should be run on HiSPARC's VPN server.  It will handle the
@@ -17,21 +17,22 @@
 import os
 import base64

-OPENVPN_DIR = '/home/david/tmp/openvpn'
-HOSTS_FILE = '/tmp/hosts-hisparc'
+OPENVPN_DIR = '/etc/openvpn'
+HOSTS_FILE = '/etc/hosts-hisparc'
 FLAG = '/tmp/flag_nagios_reload'

 def create_key(host, type, ip):
     """create keys for a host and set up openvpn"""

     if type == 'client':
-        subprocess.check_call(['./create_keys.sh', OPENVPN_DIR, host])
+        subprocess.check_call(['/usr/local/bin/create_keys.sh', OPENVPN_DIR,
+                               host])
         with open(os.path.join(OPENVPN_DIR, 'ccd', host), 'w') as file:
             file.write('ifconfig-push %s 255.255.254.0 194.171.82.1\n' %
                        ip)
     elif type == 'admin':
```

```
-         subprocess.check_call(['./create_admin_keys.sh', OPENVPN_DIR,
-                                 host])
+         subprocess.check_call(['/usr/local/bin/create_admin_keys.sh',
+                                OPENVPN_DIR, host])
     else:
         raise Exception('Unknown type %s' % type)

@@ -89,7 +90,7 @@
         rpc_paths = ('/RPC2',)

     # Create server
-    server = SimpleXMLRPCServer(("localhost", 8001),
+    server = SimpleXMLRPCServer(("tietar.nikhef.nl", 8001),
                                  requestHandler=RequestHandler)
     server.register_introspection_functions()
```

To set up the cron job for reloading nagios config, execute:

```
(root)$ crontab -e
```

and add:

```
# Run nagios reload check every minute
* * * * * /usr/local/bin/hisparc-nagios
```

3.13 Shoreline Firewall (Shorewall)

Get an RPM from:

```
$ wget http://slovakia.shorewall.net/pub/shorewall/4.4/shorewall-4.4.7/shorewall-4.4.7-5
$ sudo rpm -i shorewall-4.4.7-5.noarch.rpm
```

There is a lot of configuration to change. After thoroughly checking the existing configuration, I decided that it was not very clean. Some relevant options were missing and things were not documented very well.

For the new configuration, we start with our zones file:

```
--- zones.orig  2010-02-25 11:22:18.000000000 +0100
+++ zones       2010-02-25 11:23:52.000000000 +0100
@@ -10,3 +10,6 @@
 #ZONE   TYPE           OPTIONS          IN                       OUT
 #                                       OPTIONS                  OPTIONS
 fw      firewall
+net     ipv4
+det     ipv4
+adm     ipv4
```

with the matching interfaces file:

```
--- interfaces.orig     2010-02-25 11:51:46.000000000 +0100
+++ interfaces  2010-02-25 12:05:52.000000000 +0100
@@ -8,3 +8,6 @@
 #
 ###############################################################################
 #ZONE   INTERFACE      BROADCAST       OPTIONS
```

```
+net    eth0            detect          logmartians,nosmurfs,routefilter,tcpflags
+det    tun1            detect          logmartians,nosmurfs,routefilter,tcpflags
+adm    tun0            detect          logmartians,nosmurfs,routefilter,tcpflags
```

First, we'll define the policy:

```
--- policy.orig 2010-02-25 11:29:47.000000000 +0100
+++ policy      2010-02-25 11:46:41.000000000 +0100
@@ -9,3 +9,22 @@
 ###############################################################################
 #SOURCE        DEST    POLICY          LOG     LIMIT:          CONNLIMIT:
 #                               LEVEL   BURST           MASK
+
+# The firewall may connect to the internet
+$FW    net     ACCEPT
+
+# The internet should not be aware of any services running on the
+# firewall, except for a few exceptions (see rules)
+net    all     DROP            info
+
+# HiSPARC detector pc's should never route traffic over their VPN
+# interfaces, except for a few exceptions (see rules)
+det    net     DROP            err
+det    adm     DROP            err
+
+# HiSPARC admins should never route internet traffic over their VPN
+# interfaces
+adm    net     DROP            err
+
+# All other connections: reject
+all    all     REJECT          info
```

To easily enable the VPN traffic, without having to add various exception rules, we can define the VPN tunnels in the tunnels file:

```
--- tunnels.orig        2010-02-25 13:26:53.000000000 +0100
+++ tunnels     2010-02-25 13:29:56.000000000 +0100
@@ -9,3 +9,9 @@
 ###############################################################################
 #TYPE                  ZONE    GATEWAY         GATEWAY
 #                                              ZONE
+
+# Admin VPN
+openvpnserver          net     0.0.0.0/0
+
+# Detector VPN
+openvpnserver:tcp:443  net     0.0.0.0/0
```

The rest of the traffic has to be enabled by adding exceptions to the rules file:

```
--- rules.orig  2010-02-25 11:50:52.000000000 +0100
+++ rules       2010-02-25 14:06:13.000000000 +0100
@@ -12,3 +12,39 @@
 #SECTION ESTABLISHED
 #SECTION RELATED
 SECTION NEW
+
+# Always accept SSH to tietar
```

```
+SSH(ACCEPT)    all             $FW
+# Accept SSH from detector vpn to admin vpn
+SSH(ACCEPT)    det             adm
+
+# Accept ping to firewall and icmp from firewall
+Ping(ACCEPT)   all             $FW
+ACCEPT         $FW             all             icmp
+# Accept ping from admin vpn to detector vpn
+Ping(ACCEPT)   adm             det
+
+#
+# Services running on tietar
+#
+# DNS
+DNS(ACCEPT)    det             $FW
+DNS(ACCEPT)    adm             $FW
+# Web
+Web(ACCEPT)    net             $FW
+# vpn xml-rpc server (allowed from pique)
+ACCEPT         net:192.16.185.167      $FW             tcp     8001
+
+#
+# Nagios traffic
+#
+# NRPE, NSClient running on detector pc's
+ACCEPT         $FW             det     tcp     5666,12489
+# NSCA running on detector pc's
+ACCEPT         det             $FW     tcp     5667
+
+#
+# Admin access to detector pc's
+#
+# VNC
+ACCEPT         adm             det     tcp     5900
```

Our firewall is now set up. To keep the server accessible when the firewall is stopped, starting or stopping, we can edit the routestopped file:

```
--- routestopped.orig   2010-02-25 12:39:00.000000000 +0100
+++ routestopped        2010-02-25 12:39:59.000000000 +0100
@@ -12,3 +12,4 @@
 ###############################################################################
 #INTERFACE     HOST(S)                 OPTIONS         PROTO   DEST    SOURCE
 #                                                              PORT(S) PORT(S)
+eth0          -                       -               tcp     ssh
```

where we've only enabled SSH access. The only thing remaining is enabling our firewall:

```
--- shorewall.conf.orig 2010-02-25 12:33:32.000000000 +0100
+++ shorewall.conf      2010-02-25 14:33:41.000000000 +0100
@@ -18,7 +18,7 @@
 #                      S T A R T U P   E N A B L E D
 ###############################################################################

-STARTUP_ENABLED=No
+STARTUP_ENABLED=Yes

 ###############################################################################
```

```
#                                   V E R B O S I T Y
```

Starting our firewall is accomplished with:

```
$ sudo /sbin/service shorewall start
```

3.14 (Maybe) not relevant

Installed screen Installed ntp

CHAPTER

FOUR

INSTALLATION OF NECKAR

Granting davidf rights to manage software and services:

```
(root)$ visudo
```

and adding:

```
davidf  ALL = SOFTWARE, SERVICES
```

4.1 Web server

Change configuration in /etc/httpd/conf/httpd.conf. Patch:

```
--- httpd.conf.orig     2009-12-04 14:35:39.000000000 +0100
+++ httpd.conf  2009-12-04 14:35:50.000000000 +0100
@@ -228,8 +228,8 @@
 #  when the value of (unsigned)Group is above 60000;
 #  don't use Group #-1 on these systems!
 #
-User apache
-Group apache
+User www
+Group www

 ### Section 2: 'Main' server configuration
 #
```

Enabling httpd on startup:

```
$ sudo /sbin/chkconfig --add httpd
$ sudo /sbin/chkconfig --levels 35 httpd on
```

Starting httpd now:

```
$ sudo /sbin/service httpd start
```

4.2 MySQL Server

The mysql server was pre-installed on this system, but not configured. To configure mysql and create the TYPO3 database:

```
$ sudo /sbin/chkconfig --levels 35 mysqld on
$ sudo /sbin/service mysqld start
$ mysqladmin -u root password 'secret_password'
$ mysql -u root -p
mysql> create database hisparc_t3 default character set 'utf8';
mysql> grant all on hisparc_t3.* to 'hisparc'@'localhost' identified by 'secret_passwor
```

4.3 HiSPARC website

The HiSPARC website is a typical TYPO3 installation with some added modules. This installation is created and provided by OOiP. From the TYPO3 website:

> TYPO3 is a free Open Source content management system for enterprise purposes on the web and in intranets. It offers full flexibility and extendability while featuring an accomplished set of ready-made interfaces, functions and modules.

4.3.1 Prerequisites

TYPO3 has some prerequisites, some of which were already installed: PHP and ImageMagick. Unfortunately, MySQL support for PHP was not yet installed. Do this by issuing:

```
$ sudo yum install php-mysql
```

It turns out the permissions for the PHP session directory were incorrect. Correct them as follows:

```
(root)$ chown www.www /var/lib/php/session
```

To make sure TYPO3 uses loopback connections to itself, update the /etc/hosts file to contain:

```
127.0.0.1       localhost.localdomain localhost neckar.nikhef.nl neckar www.hisparc.nl
```

4.4 Website

To install the HiSPARC website, untar the OOiP-provided directory dump:

```
$ cd /usr/local
(root)$ mkdir www
(root)$ chown www.www www
$ cd www
(root)$ tar xvzf hisparc-web.tar.gz --strip-components=1
(root)$ chown -R www.www *
(root)$ chmod -R a-x *
(root)$ chmod -R a+X *
$ mysql -u hisparc -p hisparc_t3 < hisparc_t3.sql
```

Create the apache config by creating and editing /etc/httpd/conf.d/hisparc.conf to contain:

```
<VirtualHost *:80>
    ServerName www.hisparc.nl
    ServerAlias neckar.nikhef.nl

    DocumentRoot /usr/local/www/web
    <Directory /usr/local/www/web>
        AllowOverride All
        Allow from All

        Options +FollowSymLinks +ExecCGI

    </Directory>

</VirtualHost>
```

After that, reload the web server:

```
$ sudo /sbin/service httpd reload
```

Installation should now be complete.

CHAPTER FIVE

INDICES AND TABLES

- *genindex*
- *modindex*
- *search*