NG Trace

ADMINISTRATOR’S MANUAL

December 2009

Table of Contents

Table of Figures

1. Introduction

1.1 Overview

This Document describes NG Trace's general contents. This explains about **NG Trace** information, installation method, environment, architecture and how to use.

1.2 References

- libpcap (http://www.tcpdump.org)
- Ruby On Rails
- Apache
- Apache + Passenger
- Acl_system2 (RBAC access control for Ruby On Rails)
- Libgsoap
- Soap4r
- Smsd
- Git SCM
- Doxygen
- RDOC
- RSPEC
- ICMP – http://en.wikipedia.org/wiki/Internet_Control_Message_Protocol

2. System overview

2.1 About the system

NG Trace is a corporate security system which is capable of monitoring the network traffic and taking action on the occurrence of suspicious or potentially dangerous events.

NG Trace is a modern security system with flexible, multi-layered and easily configurable architecture and software design.

It has intuitive user-friendly interface and lots of functionalities.

It can apply both set of predefined rules following suspicious users' behavior and it can accept new targets of interest defined by newly inserted rule sets.

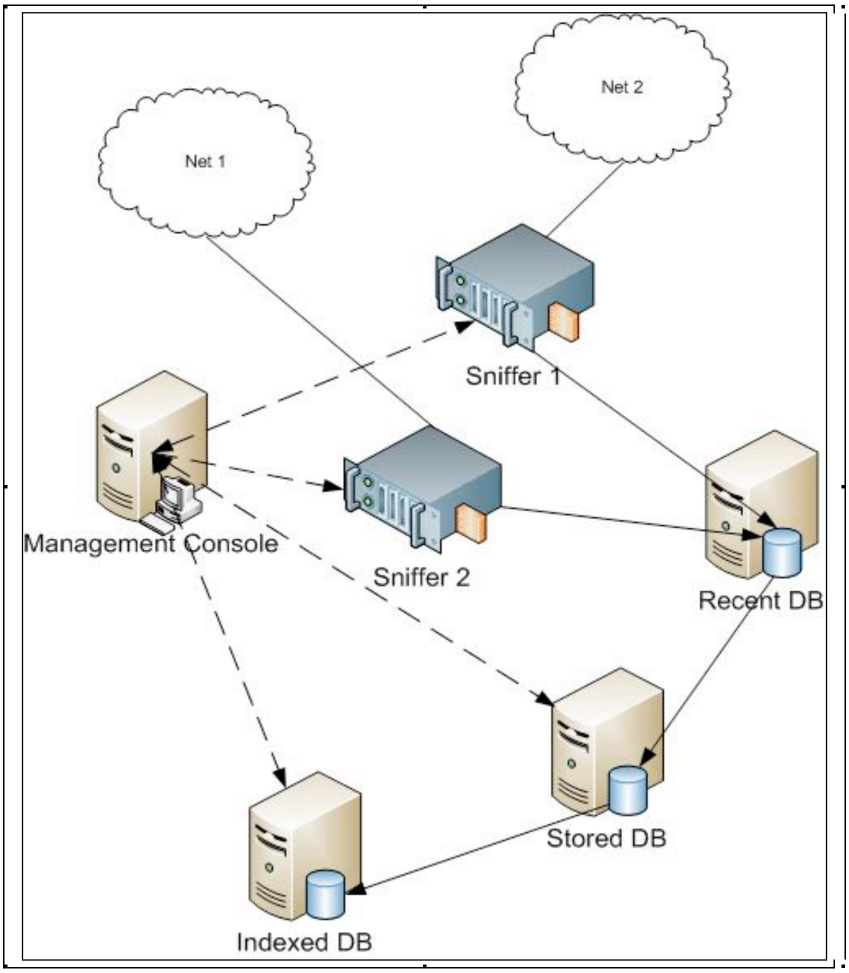


Figure 1. System Overview

2.2 Feature

- ✓ Capturing network traffic, transferring it to readable look and connecting of communication sessions.
- ✓ Saving the decoded traffic into database.
- ✓ Indexing of the decoded traffic into database.
- ✓ Exporting the data of database.
- ✓ Archiving of the database on hardware device.
- ✓ Sending e-mails in case of the emerging of different event.

2.3 Environment

2.3.1 OS Environment

All system components are supported on POSIX compliant UNIX systems like GNU / Linux 2.6.x, FreeBSD and OpenSolaris 10.x where GNU / Linux is a requirement and FreeBSD and OpenSolaris are optional.

The system is able to be executed on most modern GNU / Linux distributions.

Cent OS 5.3 is the default GNU / Linux distribution used.

2.3.2 Hardware Environment

The system's components run on Intel based, GNU / Linux compatible server machines, equipped with at least one network card, a CD / DVD drive, enough hard-disk space and RAM.

If all the system's components are deployed on a single server, it should be equipped with:

- Dual Core 2.4GHz Pentium CPU,
- 4GB RAM system memory,
- 80GB available disk space or more
- 100Mbit/s Network card or more

2.3.3 Environment of System's components

Sniffer

Sniffer implements the capturing, processing, storing the network traffic of interest, applying the filter and other rules in the configuration.

- **Software environment**

As the system's core – is implemented as a highly optimized, multi-threaded, C/C++, libpcap based, standalone user daemon process.

- **Hardware environment**

Processor: 32-bit Pentium (400MHz or greater) or 64-bit processor
Memory: 128MB RAM system memory (recommended: 256MB or more)
Disk Space: 80GB available disk space or more
Network card: 100Mbit/s or more

Recent DB

Working database kept for a short period of time.

- **Software environment.**

PostgreSQL

- **Hardware environment**

Processor: 32-bit Pentium (400MHz or greater) or 64-bit processor
Memory: 128MB RAM system memory (recommended: 256MB or more)
Disk Space: 80GB available disk space or more
Network card: 100Mbit/s or more

Stored DB

Data kept for a long period of time.

- **Software environment**

PostgreSQL

- **Hardware environment**

Processor: 32-bit Pentium (400MHz or greater) or 64-bit processor
Memory: 128MB RAM system memory (recommended: 256MB or more)
Disk Space: 160GB available disk space or more
Network card: 100Mbit/s or more

Index DB

Indexing of stored DB.

- **Software environment**

PostgreSQL

- **Hardware environment**

Processor: 32-bit Pentium (400MHz or greater) or 64-bit processor
Memory: 128MB RAM system memory (recommended: 256MB or more)
Disk Space: 80GB available disk space or more
Network card: 100Mbit/s or more

3. Working Architecture

3.1 Physical Architecture

In order to perform its basic functions, the system should be able to capture the needed network traffic.
In order to do so, the server should have at least one network card (Ethernet) and the user executing the Sniffer should have privileges to put the interface(s) in promiscuous mode.
In modern switched network environments the Sniffer should be deployed in such a way that it could capture the network traffic as appropriate.
The NG Trace system architecture can be divided in two main variants as below.

- One way to achieve this is to deploy the server running the Sniffer as a gateway, in front of the network.

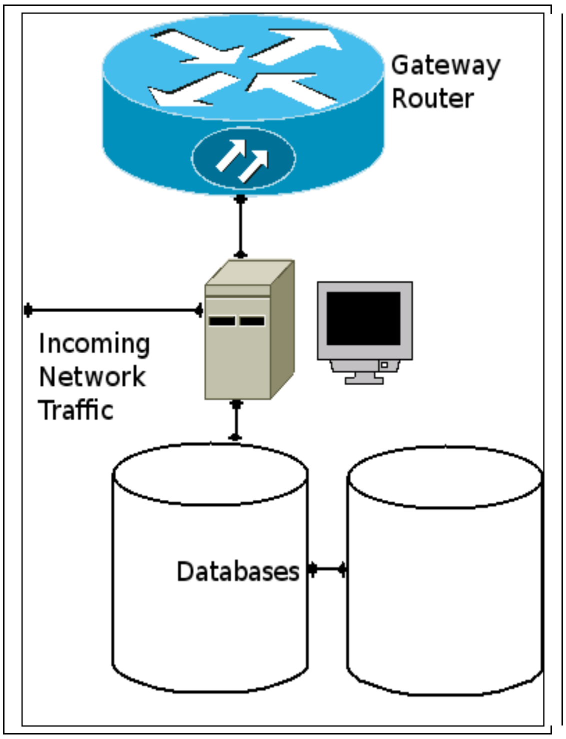


Figure 2. Interceptive mode

Figure 2. The analyzing server is set into an interceptive mode or manner – it simply stay on the way of the network traffic flow. Data is received and examined there and transparently passed on to the gateway router.

- The other – and most commonly used – method for making the network traffic available for capture is by configuring SPAN (also known as port mirroring) on the switches.
  When using SPAN, one of the switch's ports is configured as a monitoring port, thus making all traffic, which passes via the mirrored on the monitoring port.
  The server running the Sniffer should be connected to the monitoring port so that it can successfully capture the network traffic.

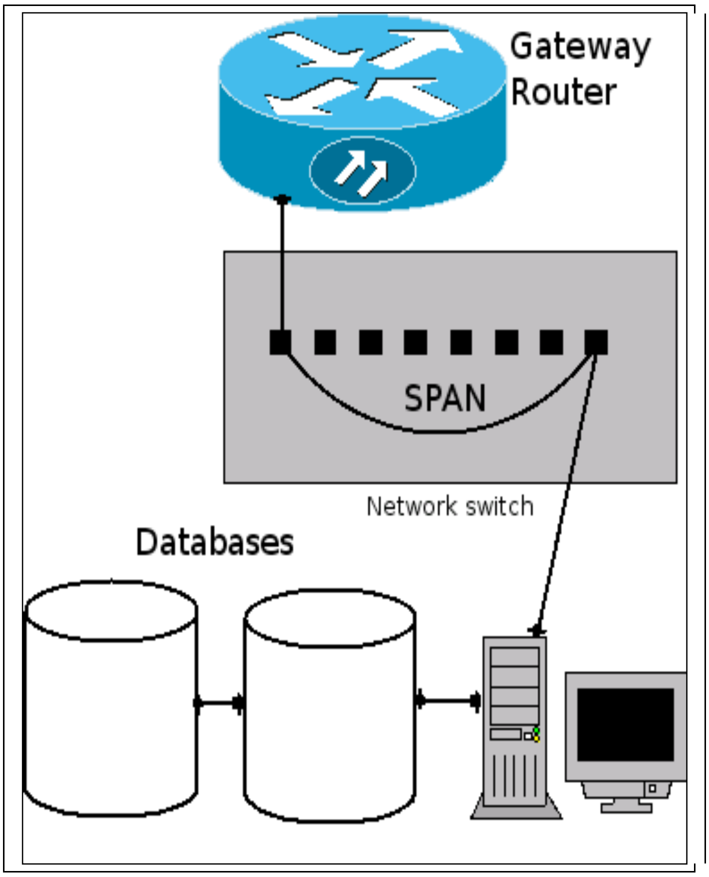


Figure 3. span mode

Figure 3. The analyzing server is connected to network switch with "port spanning" enabled, which means that it will receive network traffic in that manner – while passing through in and out each and every packet of data will pass also through it and will be afterwards saved onto the databases.

3.2 Logical Architecture

The System is built from the following main components.

- Sniffer – A C/C++ libpcap based system core, which handles capturing, processing, storing etc the network traffic of interest, applying the filter and other rules in the configuration.

- Datastores

For centralizing of the data and because of their large size, it's necessary to use one central database (stored Data) and one or more working database (Recent Data).

- ✓ Recent DB – A database (RDBMS and file-system based) in which the recent network activities are stored. Some of the functionalities of the system such as auditing the HTTP history of the users require a file-system based cache system to be developed for storing static contents (like CSS, images, video etc). Thus the disk usage can be optimized since the same static files will not be downloaded again and again.
- ✓ Stored DB – keeps audit etc data for long periods of time.
- ✓ Indexed DB – Indexing of Stored DB.

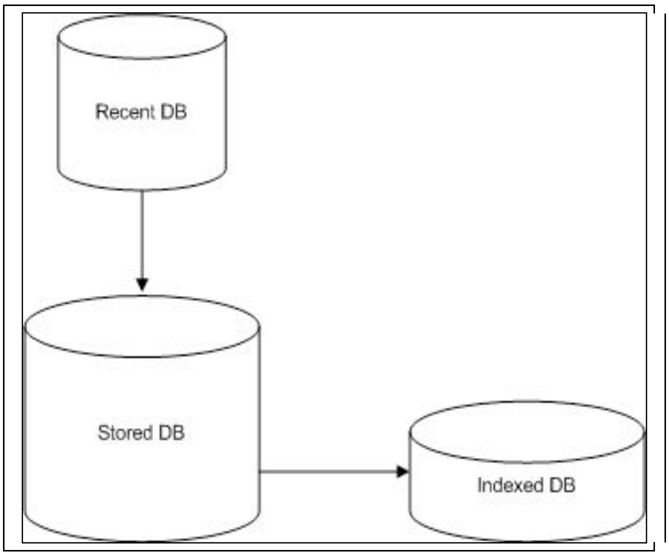


Figure 4. Database Architecture

- ➢ Management console (web interface) – The system is equipped with a functional and user friendly web interface, which make possible both remote system administration and configuration and viewing the various audit etc report generated by the system.
- ➢ Extra / Supporting programs (daemons, scripts etc)
  - ✓ Report generator – script(s) used to generate and send (via e-mail)

    Reports scheduled reports.
  - ✓ Notifier – A REST / SOAP web service, which listens (on a configured network port of one of the servers the systems components are deployed on) for notifications sent from the Sniffer on the event of critical

security violations.

Upon receiving a notification the Notifier processes the request and finally sends an e-mail (or SMS) notification with the details to the administrator.

- ✓ Exporter - Export of the data from working to the stored database.
- ✓ Indexer – Index of the content of the stored data after exporting.
- ✓ Archivator – Archiving of the stored database on tape device or DVD.
- ✓ FS Web Service – Ruby On Rails web service which allows secure browsing of web contents stored in Recent DB and Stored DB.

4. Installation Overview

4.1 Software Architecture

The following figure shows arrangement of system configuration.

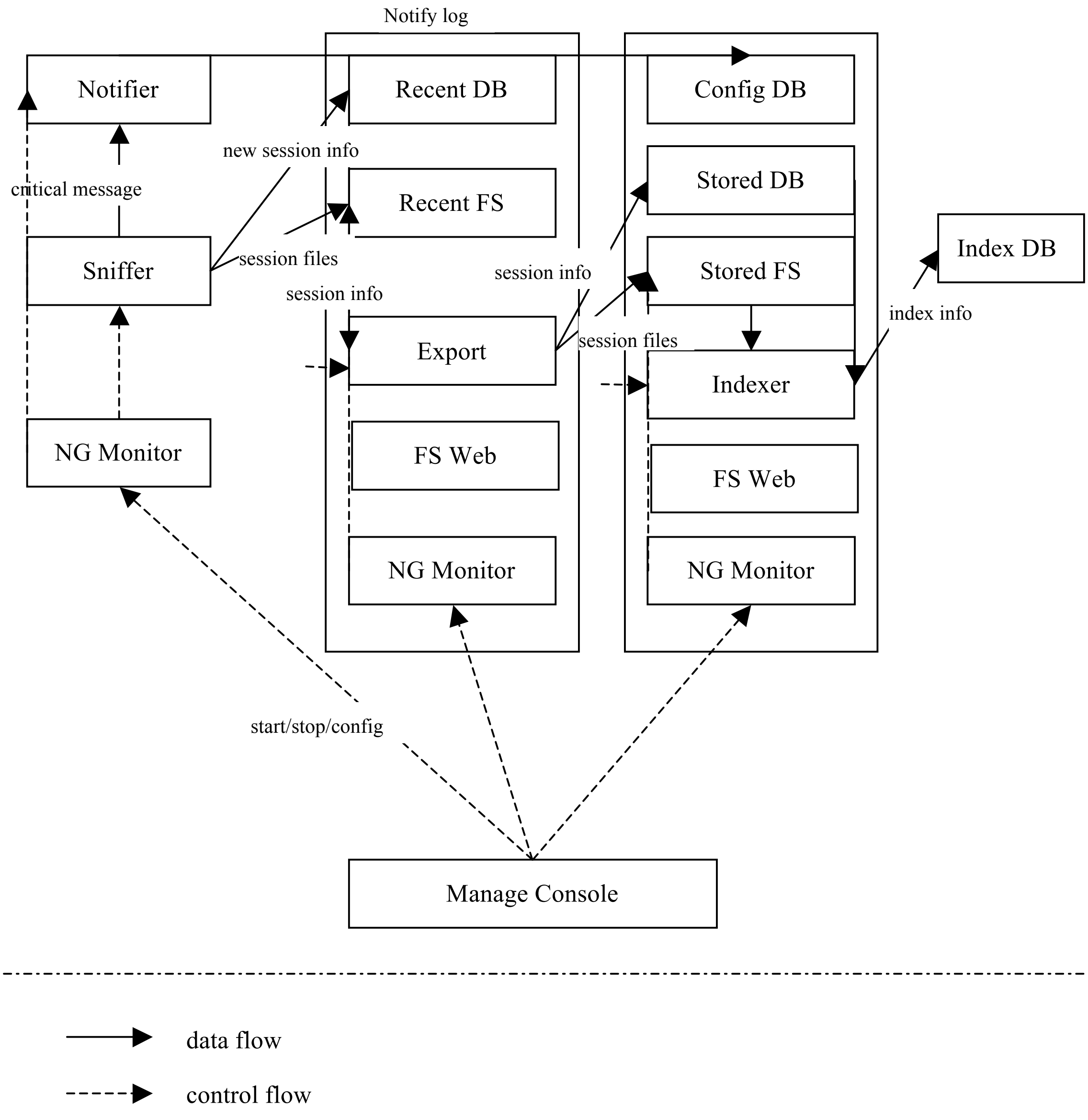


Figure 5. Software Configuration

4.2 Installation and Configuration Procedure

About installation of software components, refer to "5. Installations".
About configuration of them, refer to "6. Configurations"

Configure the system as following procedure:

- Recent DB Configuration

  Setup the PostgreSQL's authentication info so that components (Sniffer, Notifier, Exporter, Manage Console, etc…) connect to Recent DB.
- Recent FS Configuration

  Setup the port number for file uploading service (rfs.conf).
- Stored DB and Config DB Configuration

  Setup the PostgreSQL's authentication info so that components (Exporter, Indexer, Manage Console, etc…) connect to Stored DB and Config DB.
- Stored FS Configuration

  Setup the port number for file uploading service (sfs.conf).
- Manage Console Configuration

  Setup the DB connection information with the Config DB. Make configuration settings for Apache Web Service. (ngtrace_web.conf)
- Notifier Configuration

  Setup the DB connection information with the Config DB.(notifier.conf)
- Sniffer

  Setup the connection information with other components such as Notifier, Recent DB and Recent_fs etc(sniffer.conf).
- Exporter

  Setup the connection information with other components such as Recent DB, Stored DB and Stored_fs etc(exporter.conf).
- Indexer

  Setup the connection information with other components such as Stored DB(Local Path) and Index DB etc(indexer.conf).
- NG Monitor

Setup the port number for the soap service and the path for logging.

Besides, NG Monitor requires configuration files that specify paths for executable file and config file of each component to manage all components such as sniffer, notifier, recent_fs, stored_fs, exporter, and indexer. (m_sniffer.conf, m_exporter.conf, m_indexer.conf, m_notifyd.conf, m_rfs.conf, m_sfs.conf)

And configuration file for Apache web service is required.(ngtrace_web.conf)

- FS Web Service

  Configuration file for Apache web service is required..(ngtrace_web.conf)

5. Installations

5.1 Sniffer

Install the package:

```
ngtrace-sniffer-0.1.5-1.0.i386.rpm
```

When the installation is finished, the following files will be installed:

```
/usr/bin/pdump
/usr/bin/sniffer
/usr/lib/libngtcapt
/usr/lib/libngtds
/usr/lib/libngtmisc
/etc/ngtrace/sniffer.conf
/etc/ngtrace/mconf/m_sniffer.conf
/usr/bin/dissector_plugins/libaim
/etc/ngtrace/sniffer_plugin/aim.conf
/usr/bin/dissector_plugins/libbittorent
/etc/ngtrace/sniffer_plugin/bittorent.conf
/usr/bin/dissector_plugins/libftp
/etc/ngtrace/sniffer_plugin/ftp.conf
/usr/bin/dissector_plugins/libhttp
/etc/ngtrace/sniffer_plugin/http.conf
/usr/bin/dissector_plugins/libhttps
/etc/ngtrace/sniffer_plugin/https.conf
/usr/bin/dissector_plugins/libicmp
/etc/ngtrace/sniffer_plugin/icmp.conf
/usr/bin/dissector_plugins/libimap
/etc/ngtrace/sniffer_plugin/imap.conf
/usr/bin/dissector_plugins/libirc
/etc/ngtrace/sniffer_plugin/irc.conf
/usr/bin/dissector_plugins/libjabber
/etc/ngtrace/sniffer_plugin/jabber.conf
/usr/bin/dissector_plugins/libnfs
```

/etc/ngtrace/sniffer_plugin/nfs.conf
/usr/bin/dissector_plugins/libpop
/etc/ngtrace/sniffer_plugin/pop.conf
/usr/bin/dissector_plugins/libsmb
/etc/ngtrace/sniffer_plugin/smb.conf
/usr/bin/dissector_plugins/libsmtp
/etc/ngtrace/sniffer_plugin/smtp.conf

Dependencies (must be installed beforehand):

libpcap-0.9.4-14.el5.i386.rpm
libyaml-0.1.2-3.el5.i386.rpm
postgresql-libs-8.1.11-1.el5_1.1.i386.rpm
glib2-2.12.3-2.fc6.i386.rpm

5.2 Recent DB

Recent DB will be installed on PostgreSQL 8.1.11.

Create the database with the file "recentdb.sql".

```
[#root@hostname]su postgress
$bash psql –d template1 –a –f ./recentdb.sql
$bash exit
[#root@hostname]service postgress start|restart
```

Enable that system components can connect to the Database. To do this, change the config file of the PostgreSQL as the following.

```
Open the /var/lib/pgsql/data/pg_hba.conf.
# IPv4 local connections:
host   all   all  127.0.0.1/32  ident sameuser

Register IP Address of DB client that must access the recent
database server and change the trust or ident sameuser into
password.

host   all   all  192.168.1.127/32   password
host   all   all  192.168.1.130/32   password

Then open the /var/lib/pgsql/data/postgresql.conf and change the
value of listen_addresses as follows:
listen_addresses = '*'
```

5.3 Recent File Service

Install the package:

ngtrace-recentfs-0.1.5-1.0.i386.rpm

When the installation is finished, the following files will be installed:

/usr/bin/ngtrace_rfs
/etc/ngtrace/rfs.conf
/etc/ngtrace/mconf/m_rfs.conf
/etc/httpd/conf.d/ngtrace_web.conf
/etc/httpd/modules/mod_xsendfile.so
/var/www/fs
/var/www/ngconsole/public/fs

Dependencies (must be installed beforehand):

libyaml-0.1.2-3.el5.i386.rpm
glib2-2.12.3-2.fc6.i386.rpm
ruby-1.8.7-p72.tar.gz
rubygems-1.3.5.tgz

The following gems must be installed.

- rspec-1.2.8.gem
- actionmailer-2.3.2.gem
- actionpack-2.3.2.gem
- activerecord-2.3.2.gem
- activeresource-2.3.2.gem
- activesupport-2.3.2.gem
- dbi-0.4.2.gem
- deprecated-2.0.1.gem
- fastthread-1.0.7.gem

- passenger-2.2.5.gem
- pg-0.8.0.gem
- postgres-0.7.9.2008.01.28.gem
- rack-1.0.0.gem
- rails-2.3.2.gem
- rake-0.8.7.gem

5.4 Exporter

Install the package:

ngtrace-exporter-0.1.5-1.0.i386.rpm

When the installation is finished, the following files will be installed:

/usr/bin/ngtrace_exporter.rb
/etc/ngtrace/exporter.conf
/etc/ngtrace/mconf/m_exporter.conf

Dependencies (must be installed beforehand):

ruby-1.8.7-p72.tar.gz
rubygems-1.3.5.tgz

The following gems must be installed.

- columnize-0.3.1.gem
- dbd-pg-0.3.8.gem
- dbi-0.4.2.gem
- deprecated-2.0.1.gem
- hpricot-0.8.1.gem
- linecache-0.43.gem
- pg-0.8.0.gem
- soap4r-1.5.8.gem
- httpclient-2.1.5.2.gem

To install these gems, please use the file “ngtrace-gem-install-exporter-0.1.5.i386.tar.gz”. Decompress this file, and execute the file “install.sh” in it.

For management scheduling of Exporter, open the /etc/crontab and insert two lines as the following;

```
    30 02 * * * root exporter.rb -interval
    30  *   * * * root exporter.rb -capacity
```

The first line means that exporting work will be invoked at 02:30:00 every day.

The second line means that disk space checking will be invoked at 30:00 every hour.

5.5 Stored DB

Stored DB will be installed on PostgreSQL 8.1.11.

Create the database with the file “storedb.sql” and “configdb.sql”.

```
[#root@hostname]su postgress
$bash psql -d template1 -a -f ./storeddb.sql
$bash psql -d template1 -a -f ./configdb.sql
$bash exit
[#root@hostname]service postgress start|restart
```

Enable that system components can connect to Database. To do this, change the config file of the PostgreSQL as the following;

```
Open the /var/lib/pgsql/data/pg_hba.conf.
# IPv4 local connections:
host   all   all  127.0.0.1/32  ident sameuser

Register IP Address of DB client that must access the recent
database server and change the trust or ident sameuser into
password.

Ex)
host   all   all  192.168.1.127/32   password
host   all   all  192.168.1.130/32   password

Then open the /var/lib/pgsql/data/postgresql.conf and change the
```

```
value of listen_addresses as follows:
listen_addresses = '*'
```

5.6 Stored File Service

Install the package:

ngtrace-storedfs-0.1.5-1.0.i386.rpm

When the installation is finished, the following files will be installed:

/usr/bin/ngtrace_sfs.rb
/etc/ngtrace/sfs.conf
/etc/httpd/conf.d/ngtrace_web.conf
/etc/ngtrace/mconf/m_storedfs.conf
/etc/httpd/modules/mod_xsendfile.so
/var/www/fs
/var/www/ngconsole/public/fs

Dependencies (must be installed beforehand):

ruby-1.8.7-p72.tar.gz
rubygems-1.3.5.tgz

The following gems must be installed. (ngtrace-gem-install-sfs-0.1.5.i386.tar.gz)

- soap4r-1.5.8.gem
- httpclient-2.1.5.2.gem
- rspec-1.2.8.gem
- actionmailer-2.3.2.gem
- actionpack-2.3.2.gem
- activerecord-2.3.2.gem
- activeresource-2.3.2.gem
- activesupport-2.3.2.gem
- dbi-0.4.2.gem
- deprecated-2.0.1.gem
- fastthread-1.0.7.gem
- passenger-2.2.5.gem
- pg-0.8.0.gem

- postgres-0.7.9.2008.01.28.gem
- rack-1.0.0.gem
- rails-2.3.2.gem
- rake-0.8.7.gem

To install these gems, please use the file “ngtrace-gem-install-sfs-0.1.5.i386.tar.gz”. Decompress this file, and execute the file “install.sh” in it.

5.7 Indexer

Install the package:

ngtrace-indexer-0.1.5-1.0.i386.rpm

When the installation is finished, the following files will be installed:

/usr/bin/ngtrace_indexer
/etc/ngtrace/indexer.conf
/etc/ngtrace/mconf/m_indexer.conf

Dependencies (must be installed beforehand):

libyaml-0.1.2-3.el5.i386.rpm
postgresql-libs-8.1.11-1.el5_1.1.i386.rpm
glib2-2.12.3-2.fc6.i386.rpm

For management scheduling of indexer, open the /etc/crontab and insert the following line;

```
    10  *  * * * root ngtrace_indexer
```

The line means that indexing will be invoked at 10:00 every hour.

5.8 Index DB

Index DB will be installed on PostgreSQL 8.1.11.

Create the database using the file “indexdb.sql”.

```
 [#root@hostname]su postgress
$bash psql -d template1 -a -f ./indexdb.sql
$bash exit
[#root@hostname]service postgress start|restart
```

Enable that system components can connect to Database. To do this, change the config file of the PostgreSQL as the following.

```
Open the /var/lib/pgsql/data/pg_hba.conf.
# IPv4 local connections:
host   all   all  127.0.0.1/32  ident sameuser

Register IP Address of DB client that must access the recent
database server and change the trust or ident sameuser into
password.

Ex)
host   all   all  192.168.1.127/32   password
host   all   all  192.168.1.130/32   password

Then open the /var/lib/pgsql/data/postgresql.conf and change the
value of listen_addresses as follows:
listen_addresses = '*'
```

5.9 Notifier

Install the package:

ngtrace-notifier-0.1.5-1.0.i386.rpm

When the installation is finished, the following files will be installed:

/usr/bin/ngtrace_notifyd
/etc/ngtrace/notifier.conf
/etc/ngtrace/mconf/ m_notifyd.conf

Dependencies (must be installed beforehand):

libyaml-0.1.2-3.el5.i386.rpm
postgresql-libs-8.1.11-1.el5_1.1.i386.rpm
glib2-2.12.3-2.fc6.i386.rpm
sendmail-8.13.8-2.el5
smstools-2.2.20

5.10 NG Monitor

Install the package:

ngtrace-monitor-0.1.5-1.0.i386.rpm

When the installation is finished, the following files will be installed:

/usr/bin/ngtrace_monitor.rb
/etc/ngtrace/monitor.conf

Dependencies (must be installed beforehand):

ruby-1.8.7-p72.tar.gz
rubygems-1.3.5.tgz

Decompress the rubygems-1.3.5.tgz, and install as follows :

```
rubygem
# ruby setup.rb
# gem install ruby-postgres
```

The following gems must be installed.

- dbd-pg-0.3.8.gem
- dbi-0.4.2.gem
- deprecated-2.0.1.gem
- httpclient-2.1.5.2.gem
- postgres-pr-0.6.1.gem
- soap4r-1.5.8.gem

To install these gems, please use the file “ngtrace-gem-install-monitor-0.1.5.i386.tar.gz”. Decompress this file, and execute the file “install.sh” in it.

5.11 NG Console (Web)

Install the package:

ngtrace-console-0.1.5-1.0.i386.rpm

When the installation is finished, the following files will be installed:

/var/www/ngconsole/

/etc/httpd/conf.d/ngtrace_web.conf

Dependencies (must be installed beforehand):

ruby-1.8.7-p72.tar.gz

rubygems-1.3.5.tgz

Decompress the rubygems-1.3.5.tgz, and install as follows :

```
Rubygem
# ruby setup.rb
# gem install ruby-postgres
```

The following gems must be installed.

- rspec-1.2.8.gem
- actionmailer-2.3.2.gem
- actionpack-2.3.2.gem
- activerecord-2.3.2.gem
- activeresource-2.3.2.gem
- activesupport-2.3.2.gem
- dbi-0.4.2.gem
- deprecated-2.0.1.gem
- fastthread-1.0.7.gem
- passenger-2.2.5.gem
- pg-0.8.0.gem
- postgres-0.7.9.2008.01.28.gem
- rack-1.0.0.gem
- rails-2.3.2.gem
- rake-0.8.7.gem
- soap4r-1.5.8.gem

To install these gems, please use the file “ngtrace-gem-install-web-0.1.5.i386.tar.gz”. Decompress this file, and execute the file “install.sh” in it.

5.12 System Configuration

After installation was completed successfully, system software components must be configured properly for consistent operation, as the following procedures.

- Management Console's DB connection settings. (for more detail, refer to "6.7 Management Consol")
- Register monitors. (for more details, refer to "8.2.1 COMPONENTS - Monitors")
- Component settings. (for more details, refer to "8.2.1 COMPONENTS - System Components")
- Register and setup databases (recent DB, stored DB) connection (for more details, refer to "8.2.1 COMPONENTS - DBMS Components")
- Setup low-level / high-level filters for sniffing. (for more detail, refer to "8.2.3 FILTER RULES")
- Register Notifier Receiver's informations (e-mails and mobile number).(for more detail, refer to "8.2.4 NOTIFIERS")

6. Configurations

6.1 Sniffer

Upon Sniffer startup or a request for reloading the configuration, the component reads the configuration file, which has to be readable, prepared and edited by the administrator according to his preferences.

The Sniffer's configuration file is located in

/etc/ngtrace/**sniffer.conf**.

The following options are kept in configuration file.

- IP, DSN (username, password and DB name) of the DBs.
- Location of the PID file.
- Location of the log file and debug level.
- Path to the directory used for file-system based cache.
- Path to the directory used for file-system based data-store.
- Notifier service – URL for the exported by the Notifier web service, IP address, port, user name and password and/or certificates for accessing the Notifier web service.

Configuration file for each protocol plugged in Sniffer is stored as "/etc/ngtrace/sniffer_plugin/*protocol_name*.conf ". This configuration file specifies whether or not to process transactions of specific protocol.

Example 1. sniffer.conf

```
# Connection information of config database
config_database:
  adapter: postgresql <- name of DBD
  database: ngtraceconfig_dev <- name of database
  username: mypost <- login user name of config_database
  password: mypost <- loigin user password of config_database
  host: 192.168.1.130 <- IP Address of config_database
  port: 5432 <- To be connect port number

recent_database:
  adapter: postgresql <- name of DBD
  database: ngtracerecent_dev <- name of database
  username: mypost <- login user name of recent_database
  password: mypost <- login user password of recent_database
  host: 192.168.1.130 <- IP Address of recent_database
  port: 5432 <- connect port number

process_fs:
  capture_adapter_index: eth0 <- adapter to be captured
  data_directory: /tmp/.sniffer/ <- path of sniffer relative data
  pid_filename: /var/run/sniffer.pid <- name of sniffer's pid file.
  log_filename: /var/log/sniffer.log <- name of sniffer's log file.
  debug_level: 1 <- debug level
  cache_directory: /tmp/.sniffer/sessions/ <- path to directory used for file system
base cache
  pdump_directory: /tmp/.sniffer/pdump/ <- path of dump file by pdump

notifier:
  url: http://ngtracer.org/notifier <- url for the exported by the Notifier web service
  ip: 192.168.1.211 <- IP address
  port: 5378 <- port number
fs_service:
  ip: 192.168.1.104 <- IP Address of file system service
  port: 50 <- Port number of  file system service
```

Example 2. aim.conf

```
---
aim: 1  <- libaim module is loaded and filtering is applied to AIM Protocol if this
value is set to 1
```

NOTE: In this example red part is explanation of every instance.

6.2 Notifier

Upon Notifier startup or a request for reloading the configuration, the component reads the configuration file, which has to be readable, prepared and edited by the administrator according to his preferences.

The Notifier's configuration file is located in /etc/ngtrace/notifier.conf.

The following options are kept in configuration file.

- IP, DSN (username, password and DB name) of the DBs.
- Notifier web service port.
- Location of the PID file.
- Location of the log file.
- Delivery type (e-mail or SMS)

Example 3. notify.conf

```
# 03.09.2009
# Connection information of config database


config_database:
  adapter: postgresql <- name of DBD
  database: ngtraceconfig_dev <-name of config_database
  username: mypost <- login user name of config_database
  password: mypost <-login user password of config_database
  host: 192.168.1.130 <- IP Address of config_database
  port: 5432 <- to be connect port number


#
# Listen: Allows you to bind Notifyd to specific
# ports, in addition to the default.
#
#listen 9999
web_service:
   #port number for listening
   listen: 9999 <- port number for listening

process_fs:
  pid_filename: /var/run/notifyd.pid <- name of pid file
  log_filename: /var/log/notifyd.log <- name of pid file
  debug_level: 2 <- debug level

delivery_type:
  e-mail: yes <- delivery type
  SMS: yes <- delivery type
```

NOTE: In this example red part is explanation of every instance.

6.3 Exporter

Upon Exporter startup or a request for reloading the configuration, the component reads the configuration file, which has to be readable, prepared and edited by the administrator according to his preferences.

The Exporter's configuration file is located in

/etc/ngtrace/exporter.conf.

The following options are kept in configuration file.

- IP, DSN (username, password and DB name) of the DBs.
- Path to directory used for file – system based cache (for static files) in Recent DB.
- Exporting interval.
- Disk used percent for export.

Example 3. exporter.conf

```
# This configuration file use for exporter, fs_server

# use for gsoap server
fs_service:
  recent_base_path: /ngtfs
  port: 3721

# DB information
recent_database:
  adapter: postgresql <- name of DBD
  database: ngtraceconfig_dev <- name of config_database
  username: mypost <- login user name of config_database
  password: mypost <- Login user password of config_database
  host: 192.168.1.130 <- IP Address of config_database
  port: 5432 <- connected port number

stored_database:
  adapter: postgresql <- name of DBD
  database: ngtracerecent_dev <- name of recent_database
  username: mypost <- login user name of recent_database
  password: mypost <- login user password of recent_database
  host: 192.168.1.130 <- IP Address of recent_database
  port: 5432 <- connected port number

# moving rules
sending:
  disk_used_percent: 80 <- percent number of disk capacity

process_fs:
  pid_filename: sniffer.pid <- name of pid file
  log_filename: sniffer.log <- name of log file
  debug_level: 1 <- debug level
```

NOTE: In this example red part is explanation of every instance.

6.4 Indexer

Upon Indexer startup or a request for reloading the configuration, the component reads the configuration file, which has to be readable, prepared and edited by the administrator according to his preferences.

The Indexer's configuration file is located in
/etc/ngtrace/indexer.conf.

The following options are kept in configuration file.

- IP, DSN (username, password and DB name) of the DBs.
- Path to directory used for file – system based cache (for static files) in Stored DB.

- Indexing interval.

Example 4. indexer.conf

```
# Config Informations for Indexer daemon


stored_database:

  adapter: postgresql <- name of DBD

  database: ngtraceconfig_dev <- name of stored DB

  username: mypost <- login user name of stored DB

  password: mypost <- login user password of stored DB

  host: 192.168.1.130 <- IP Address of stored DB

  port: 5432 <- connected port number


index_database:

  adapter: postgresql <- name of DBD

  database: ngtraceconfig_dev <- name of index DB

  username: mypost <- login user name

  password: mypost <- login password

  host: 192.168.1.130 <- IP Address of index DB

  port: 5432 <- connected port number
```

NOTE: In this example red part is explanation of every instance.

6.5 Recent FS

This receives the files from Sniffer, and stores them in local Storage.
User can browse or download stored file in Management Console through FS Web Service which requires certification.

Example 5. rfs.conf

```
# Config information for the Recent File-system Storage Service
fs_service:

  port: 3720 <- port number for soap communication

  recent_base_path: /ngtfs <- path for storing file

process_fs:

  pid_filename: /var/run/notifyd.pid <- name of Pid file

  log_filename: /var/log/notifyd.log <- name of log file
  debug_level: 2 <- Debug level
```

6.6 Stored FS

This receives the files from Sniffer, and stores them in local Storage.
User can browse or download stored file in Management Console through FS Web Service which requires certification.

Example 6. Stored_fs.conf

```
# Config information for the Stored File-system Storage Service
fs_service:
  port: 3720 <- port number for soap communication
  recent_base_path: /ngtfs <- path for storing file
process_fs:
  pid_filename: /var/run/notifyd.pid <- name of pid file
  log_filename: /var/log/notifyd.log <- name of log file
  debug_level: 2 <- Debug level
```

6.7 Management Console

When management console is starting, information for connect configdb is fetched.
The database's configuration file is located in
/{consol root folder} /config/database.yml.

Example 7. database.yml

```
# MySQL.  Versions 4.1 and 5.0 are recommended.
#
# Install the MySQL driver:
#   gem install mysql
# On Mac OS X:
#   sudo gem install mysql -- --with-mysql-dir=/usr/local/mysql
# On Mac OS X Leopard:
#   sudo env ARCHFLAGS="-arch i386" gem install mysql -- --with-mysql-
config=/usr/local/mysql/bin/mysql_config
#       This sets the ARCHFLAGS environment variable to your native architecture
# On Windows:
#   gem install mysql
#       Choose the win32 build.
#       Install MySQL and put its /bin directory on your path.
#
# And be sure to use new-style password hashing:
#   http://dev.mysql.com/doc/refman/5.0/en/old-client.html
development:
  adapter: postgresql <- Type of DBD.
  encoding: utf8 <- Type of encoding.
  reconnect: false <- True; reconnect, False;do not reconnect.
  database: ngtraceconfig_dev <- Name of database.
  pool: 5 <-
  username: mypost <- user name of databae.
  password: mypost <- user password of database.
  host: localhost <- IP Address of host or local host.
production:
  adapter: postgresql <-  Type of DBD.
  encoding: utf8 <- Type of encoding.
  reconnect: false <- True; reconnect, False;do not reconnect.
  database: ngtraceconfig_dev <- Name of database.
  pool: 5 <-
  username: mypost <- user name of databae.
  password: mypost <- user password of database.
  host: localhost <- IP Address of host or local host.
```

6.8 Configuration DB

After (re)reading the basic options from the file, the components can continue with loading the rest of the configuration from the RDBMS and go on with its normal operation since then.

The following configuration directives are stored in database tables:

- Traffic filter rules – filter_high_rules and filter_low_rules:
  The Sniffer handles the network traffic using the configured rule set.
- Host management – host group and host managers and hosts:
  Web consol handles the hosts using the necessary authorization mechanisms – RBAC and group based access controls.

- Notifier management – notify_receiver_contacts and notify logs: Notifier registration information.
- File_cache: When transmit the cached file from Sniffer to Recent DB, the using hash data.
- Monitors: Daemon registration information for remote control the software components on web consol.
- User management – permissions , roles and users: Web interface handles the user using the registered user information.

7. Filter Rule-sets and Dissectors

There are – at each time – many rules configured in the system, each with a given name, description, priority, an expression and a target which determines what action and processing should be done to the network packets matching the given rule.

Each rule also has an attribute which relates it to the type of processing which should be done to packets matching the rule i.e. to the protocol dissector which should be used and the application level protocol specific filter attributes.

Drop (ignore), Notify, Log and Audit targets are supported.

Having filtered the undesired sessions and split the traffic in different types using the configured low level rules, the Sniffer delegates the application level protocol specific processing to the appropriate protocol dissector to apply the high level rules and take the appropriate action and do the needed processing.

Filter rule-set have two the following low - level filter rule and High - level filter rule which specific app.

7.1 Low–level Filter Rule-sets

Low – level filter include the filter expression by support libpcap, and data- time expression.

Data – time expression shall consist of date, week of data, and hour etc.

Whenever new network packets arrive on the monitored interfaces, the Sniffer consults the low level filter rule set to determine how to handle specific session.

The rules are applied in order of their priority.

The low – level filter rule shall consist the following rule items.

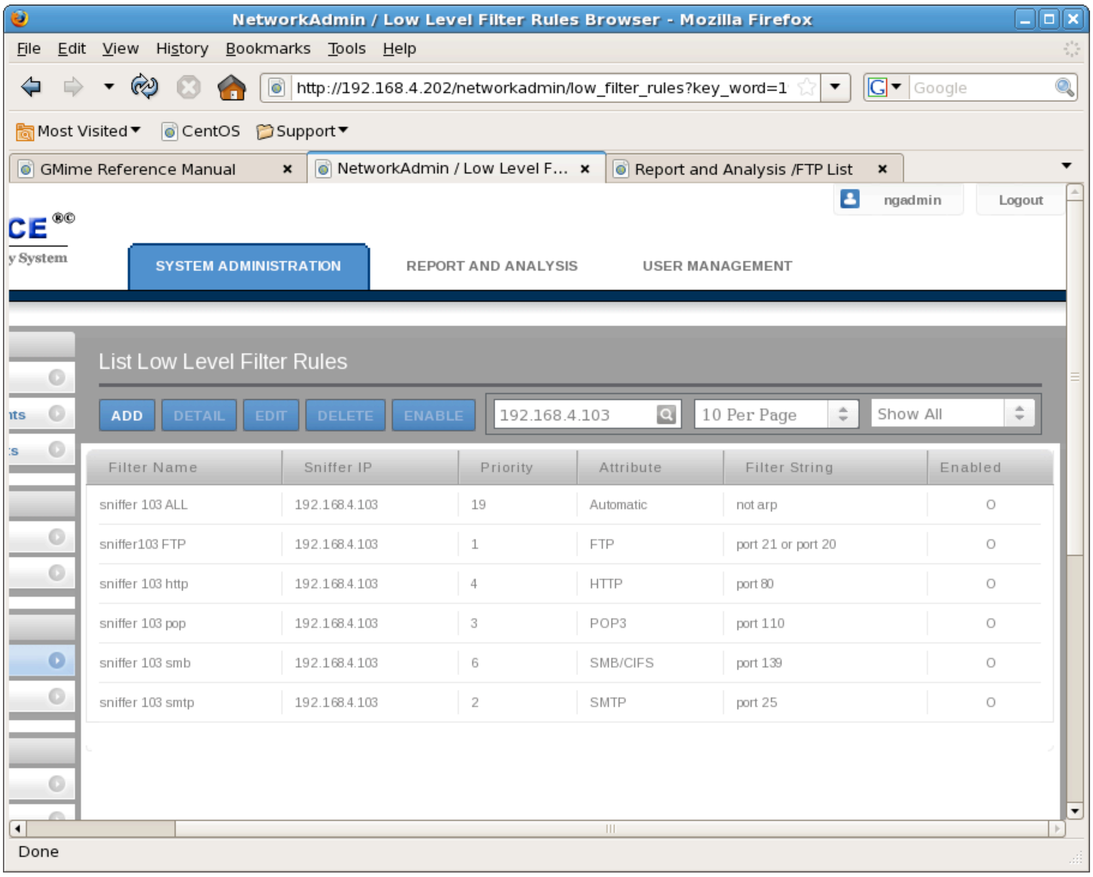
- Id : indentification value

- Filter_name : name of low - level filter rule
- Description : description
- Priority : priority of process (The smaller the value is , the higher the priority is.)
- Expression : expression of filter

You must restart sniffer components after change low-level filters.

In case one sniffer's IP has multiple low-level filter rules, total filter is set as OR sum of each valid filter. And if captured packet meets filter rule with high priority, attributes of that filter are applied and filter rules with lower priority are ignored. As a result, captured packet passes through filter rules in the order of their priorities.

Show following example:


NetworkAdmin / Low Level Filter Rules Browser - Mozilla Firefox
File Edit View History Bookmarks Tools Help
http://192.168.4.202/networkadmin/low_filter_rules?key_word=1
Google
Most Visited
CentOS
Support
GMime Reference Manual
NetworkAdmin / Low Level F...
Report and Analysis /FTP List
ngadmin
Logout
SYSTEM ADMINISTRATION
REPORT AND ANALYSIS
USER MANAGEMENT
ADD
DETAIL
EDIT
DELETE
ENABLE
10 Per Page
Show All
Done

List Low Level Filter Rules

192.168.4.103

Filter Name	Sniffer IP	Priority	Attribute	Filter String	Enabled
sniffer 103 ALL	192.168.4.103	19	Automatic	not arp	O
sniffer103 FTP	192.168.4.103	1	FTP	port 21 or port 20	O
sniffer 103 http	192.168.4.103	4	HTTP	port 80	O
sniffer 103 pop	192.168.4.103	3	POP3	port 110	O
sniffer 103 smb	192.168.4.103	6	SMB/CIFS	port 139	O
sniffer 103 smtp	192.168.4.103	2	SMTP	port 25	O

libpcap filters can be inserted using the pcap filter statement.
Such filters allow filtering of packetsprior to being passed into the *ipacc* engine for processing.
By default, the filter expression is set to ip, i.e. all IP packets will be processed.
Note that whatever filter expression is used, only IP packets will be processed.

The expression consists of one or more primitives.Primitives usually consist of an id (name or number) preceded by one or more qualifiers.
There are three different kinds of qualifier:

type qualifiers say what kind of thing the id name or number

refers to. Possible types are host, net and port. E.g., host foo, net 128.3, port 20. If there is no type qualifier, host is assumed.

dir qualifiers specify a particular transfer direction to

and/or from id. Possible directions are src, dst, src or dst and src and dst. E.g., src foo, dst net 128.3, src or dst port ftp-data. If there is no dir qualifier, src or dst is assumed. For "null" link layers (i.e. point to point protocols such as slip) the inbound and outbound qualifiers can be used to specify a desired direction.

proto qualifiers restrict the match to a particular protocol.

Possible protos are: ether, fddi, tr, ip, ip6, arp, rarp, decnet, lat, sca, moprc, mopdl, iso, esis, isis, icmp, icmp6, tcp and udp. E.g., ether src foo, arpnet 128.3, tcp port 21. If there is no proto qualiier, all protocols consistent with the type are assumed. E.g., src foo means (ip or arp or rarp) src
foo (except the latter is not legal syntax), net barmeans (ip or arp or rarp) net bar and port 53 means (tcp or udp) port 53.

(fddi is actually an alias for ether; the parser treats them identically as meaning "the data link levelused on the specified network interface." FDDI headers contain Ethernet-like source and destination addresses, and often contain Ethernet-like packet types, so you can filter on these FDDI fields just as with the analogous Ethernet fields. FDDI headers also contain other fields, but you cannot name them explicitly in a filter expression.
Similarly, tr is an alias for ether; the previous paragraph's statements about FDDI headers also apply to Token Ring headers.)

In addition to the above, there are some special primitive keywords that don't follow the pattern: gateway, broadcast, less, greater and arithmetic expressions.

All of these are described below.

More complex filter expressions are built up by using the words and, or and not to combine primitives. E.g., host foo and not port ftp and not port ftp-data. To save typing, identical qualifier lists can be omitted. E.g., tcp dst port ftp or ftp-data ordomain is exactly the same as tcp dst port ftp or tcp dst port ftp-data or tcp dst port domain.

Allowable primitives are:

dst host *host*	True if the IPv4/v6 destination field of the packet is host, which may be either an address or a name.
src host *host*	True if the IPv4/v6 source field of the packet is host.
host *host*	True if either the IPv4/v6 source or destination of the packet is host. Any of the above host expressions can be prepended with the keywords, ip, arp, rarp, or ip6 as in: ip host *host* which is equivalent to: ether proto \ip and host *host* If host is a name with multiple IP addresses, each address will be checked for a match.
ether dst *ehost*	True if the ethernet destination address is *ehost*. *Ehost* may be either a name from /etc/ethers or a number (see ethers (3N) for numeric format).
ether src *ehost*	True if the ethernet source address is *ehost*.
ether host *ehost*	True if either the ethernet source or destination address is *ehost*.
gateway *host*	True if the packet used *host* as a gateway. I.e., the ethernet source or destination address was host but neither the IP source nor the IP destination was *host*. *Host* must be a name and must be found both by the machine's host-name-to-IP-address resolution mechanisms (host name file, DNS, NIS, etc.) and by the machine's host-name-to-Ethernet-address resolution mechanism (/etc/ethers, etc.). (An equivalent expression is ether host *ehost* and not host *host* which can be used with either names or numbers for *host* / *ehost*.) This syntax does not work in IPv6-enabled

	configuration at this moment.
dst net *net*	True if the IPv4/v6 destination address of the packet has a network number of *net*. *Net* may be either a name from /etc/networks or a network number (see *networks* (4) for details).
src net *net*	True if the IPv4/v6 source address of the packet has a network number of *net*.
net *net*	True if either the IPv4/v6 source or destination address of the packet has a network number of *net*.
net *net* netmask*netmask*	True if the IP address matches *net* with the specific *netmask*. May be qualified with src or dst. Notethat this syntax is not valid for IPv6 *net*.
net *net* / *len*	True if the IPv4/v6 address matches net with a netmask len bits wide. May be qualified with src or dst.
dst port *port*	True if the packet is ip/tcp, ip/udp, ip6/tcp or ip6/udp and has a destination port value of *port*. The port can be a number or a name used in /etc/services (see *tcp*(4P) and *udp*(4P)). If a name is used, both the port number and protocol are checked. If a number or ambiguous name is used, only the port number is checked (e.g., dst port 513 will print both tcp/login traffic and udp/who traffic, and port domain will print both tcp/domain and udp/domain traffic).
src port *port*	True if the packet has a source port value of *port*.
port *port*	True if either the source or destination port of the packet is *port*. Any of the above port expressions can be pre pended with the keywords, tcp or udp, as in: tcp src port *port* which matches only tcp packets whose source port is *port*.
less *length*	True if the packet has a length less than or equal to *length*. This is equivalent to: len <= *length*.
greater *length*	True if the packet has a length greater than or equal to *length*. This is equivalent to: len >= *length*.
ip proto *protocol*	True if the packet is an IP packet (see *ip* (4P)) of protocol type *protocol*. *Protocol* can be a number or one of the names *icmp*, *icmp6*, *igmp*, *igrp*, *pim*, *ah*, *esp*, *vrrp*, *udp*, or *tcp*. Note that the identifiers *tcp*, *udp*, and *icmp* are also keywords and must be escaped via backslash (\).

	Note that this primitive does not chase the protocol header chain.
ip6 proto *protocol*	True if the packet is an IPv6 packet of protocol type *protocol*. Note that this primitive does not chase the protocol header chain.
ip6 protochain *protocol*	True if the packet is IPv6 packet, and contains protocol header with type *protocol* in its protocol header chain. For example, ip6 protochain 6 matches any IPv6 packet with TCP protocol header in the protocol header chain. The packet may contain, for example, authentication header, routing header, or hop-by-hop option header, between IPv6 header and TCP header. The BPF code emitted by this primitive is complex and cannot be optimized by BPF optimizer code in libpcap, so this can be somewhat slow.
ip protochain *protocol*	Equivalent to ip6 protochain *protocol*, but this is for IPv4.
ether broadcast	True if the packet is an ethernet broadcast packet.The ether keyword is optional.
ip broadcast	True if the packet is an IP broadcast packet. It checks for both the all-zeroes and all-ones broadcast conventions, and looks up the local subnet mask.
ether multicast	True if the packet is an ethernet multicast packet. The *ether* keyword is optional. This is shorthand for ether[0] & 1 != 0.
ip multicast	True if the packet is an IP multicast packet.
ip6 multicast	True if the packet is an IPv6 multicast packet.
ether proto *protocol*	True if the packet is of ether type *protocol*. *Protocol* can be a number or one of the names ip, ip6, arp, rarp, atalk, aarp, decnet, sca, lat, mopdl, moprc, iso, stp, ipx, or netbeui. Note these identifiers are also keywords and must be escaped via backslash (\). (In the case of FDDI (e.g., fddi protocol arp) and Token Ring (e.g., tr protocol arp), for most of those protocols, the protocol identification comes from the 802.2 Logical Link Control (LLC) header, which is usually layered on top of the FDDI or Token Ring header. When filtering for most protocol identifiers on FDDI or Token Ring, libpcap checks only the protocol ID field of an LLC header in so-called SNAP format

	with an Organizational Unit Identifier (OUI) of 0x000000, for encapsulated Ethernet; it doesn't check whether the packet is in SNAP format with an OUI of 0x000000. The exceptions are iso, for which it checks the DSAP (Destination Service Access Point) and SSAP (Source Service Access Point) fields of the LLC header, stp and netbeui, where it checks the DSAP of the LLC header, and atalk, where it checks for a SNAP-format packet with an OUI of 0x080007 and the Appletalk etype. In the case of Ethernet, libpcap checks the Ethernet type field for most of those protocols; the exceptions are *iso*, sap, and *netbeui*, for which it checks for an 802.3 frame and then checks the LLC header as it does for FDDI and Token Ring, *atalk*,where it checks both for the Appletalk etype in an Ethernet frame and for a SNAP-format packet as it does for FDDI and Token Ring, *aarp*, where it checks for the Appletalk ARP etype in either anEthernet frame or an 802.2 SNAP frame with an OUI of 0x000000, and *ipx*, where it checks for the IPX etype in an Ethernet frame, the IPX DSAP in the LLC header, the 802.3 with no LLC headerencapsulation of IPX, and the IPX etype in a SNAP frame.)
decnet src *host*	True if the DECNET source address is *host*, which may be an address of the form 10.123, or a DECNET host name. (DECNET host name support is only available on Ultrix systems that are configured to run DECNET.)
decnet dst *host*	True if the DECNET destination address is *host*.
decnet host *host*	True if either the DECNET source or destination address is *host*.
ip, ip6, arp, rarp, atalk, aarp, decnet, iso, stp, ipx, netbeui	Abbreviations for: ether proto *p* where *p* is one of the above *protocols*.
lat, moprc, mopdl	Abbreviations for: ether proto *p*

	where *p* is one of the above protocols. Note that libpcap does not currently know how to parse these protocols.
vlan [*vlan_id*]	True if the packet is an IEEE 802.1Q VLAN packet. If *vlan_id* is specified, only true is the packethas the specified *vlan_id*. Note that the first vlan keyword encountered in *expression* changes the decoding offsets for the remainder of *expression* on the assumption that the packet is a VLAN packet.
tcp, udp, icmp	Abbreviations for: ip proto p or ip6 proto *p* where *p* is one of the above protocols.
iso proto *protocol*	True if the packet is an OSI packet of protocol type *protocol*. *Protocol* can be a number or one of the names *clnp*, *esis*, or *isis*.
clnp, esis, isis	Abbreviations for: iso proto *p* where *p* is one of the above protocols. Note that libpcap does an incomplete job of parsing these protocols.
expr relop expr	True if the relation holds, where *relop* is one of >, <, >=, <=, =, !=, and *expr* is an arithmetic expression composed of integer constants (expressed in standard C syntax), the normal binary operators [+, -, *, /, &, |], a length operator, and special packet data *accessors*. To access data inside the packet, use the following syntax: proto [*expr* : *size*] Proto is one of ether, fddi, tr, ip, arp, rarp, tcp, udp, icmp or ip6, and indicates the protocol layer for the index operation. Note that *tcp*, *udp* and other upper-layer protocol types only apply to IPv4, not IPv6 (this will be fixed in the future). The byte offset, relative to the indicated protocol layer, is given by *expr*. *Size* is optional and indicates the number of bytes in the field of interest; it can be either one, two, or four, and defaults to one. The length operator, indicated by the keyword len, gives the length of the packet. For example, ether[0] & 1 != 0 catches all multicast traffic. The expression ip[0] & 0xf != 5 catches all IP packets with options. The expression ip[6:2] & 0x1fff = 0 catches only unfragmented datagrams and frag zero of fragmented datagrams. This check is implicitly applied to the tcp and udp

	index operations. For instance, tcp[0] always means the first byte of the TCP *header*, and never means the first byte of an intervening fragment. Some offsets and field values may be expressed as names rather than as numeric values. The following protocol header field offsets are available: icmptype (ICMP type field), icmpcode (ICMP code field), and tcpflags (TCP flags field). The following ICMP type field values are available: icmp-echoreply, icmp-unreach, icmp-sourcequench, icmp-redirect, icmp-echo, icmp-routeradvert, icmp-routersolicit, icmp-timxceed, icmp-paramprob, icmp-tstamp, icmp-tstampreply, icmp-ireq, icmp-ireqreply, icmp-maskreq, icmp-maskreply. The following TCP flags field values are available: tcp-fin, tcp-syn, tcp-rst, tcp-push, tcp-push, tcp-ack, tcp-urg.

Primitives may be combined using:

- ✓ A parenthesized group of primitives and operators.
- ✓ Negation (! or not).
- ✓ Concatenation (&& or and).
- ✓ Alternation (|| or or).

Negation has highest precedence. Alternation and concatenation have equal precedence and associate left to right. Note that explicit and tokens, not juxtaposition, are now required for concatenation.

If an identifier is given without a keyword, the most recent keyword
is assumed. For example,

    not host vs and ace

is short for

not host vs and host ace

which should not be confused with

not (host vs or ace)

- Attribute : This enforce the frame will be processed by the protocol dissector specified in this field.

0: automatic
1: ICMP
10: HTTP/ WebDAV
11: HTTPS
20: SMTP
21: POP3
22: IMAP
30: ICQ
31: IRC
32: Jabber
33: AIM
34: Skype
40: FTPs
41: SMB/CIFS
42: NFS
43: Reserved
44: Bittorrent

7.2 High–level Filter Rule-sets

Filter handled by the App - specific dissector, process the session data which the matching expression in Target set.
Also this time make notify according to value reflected to critical.
The high – level filter rule shall consist of the following rule items.

- Id : indentification value
- Filter_name : name of high - level filter rule
- Description : description
- Priority : priority of process (The more value is small, the more priority is high.)
- Expression : expression of filter
- Target : 1: Drop, 2: Log, 3:Audit

The Drop target means that the matching traffic should be dropped from future consideration. The packets that match a rule with a drop target are excluded from the network stream as if they were never in it.
The Log target denotes that a brief log of the matching sessions should that a given

interaction has taken place is usually enough.
The Audit target means that the matching traffic should be carefully monitored and stored in the data stores, so that it is available for later inspection, reconstruction and session replying.

- Critical : True: Critical, False: No Critical(If true, the notification request should be sent to the notifier.)
- Protocol :

  1: ICMP
  10: HTTP/ WebDAV
  11: HTTPS
  20: SMTP
  21: POP3
  22: IMAP
  30: ICQ
  31: IRC
  32: Jabber
  33: AIM
  34: Skype
  40: FTP
  41: SMB/CIFS
  42: NFS
  43: Reserved
  44: Bittorrent

High–Level Filter Expressions

Selects which packets will be dumped. If expression is “ALL”, all packets on the net will be dumped. Otherwise, only packets for which expression is ‘true’ will be dumped.

Expression is construction of one or more qualifiers. There are four different fields of qualifier:

id (identifier) is name of item of each and all protocols. Possible **id**s are ***srcip*** and ***destip*** for all protocol, ***Msgtype***, ***Msgcode*** and ***bodysize*** for ICMP, domain, ***server***, ***Client***, ***url***, ***method***, ***cntype***, ***string*** and ***bodysize*** for HTTP, ***server***, ***Client*** and ***method*** for FTP and ***server***, ***direction***, ***cntype***, ***string*** for IRC. E.g. ‘***srcip***

192.168.1', '***bodysize*** > 30', '***direction*** UP' and so on. In a qualifier **id** cannot NULL.

value is **id** value to distinguish packets. **value** are both string and number: '192.168.1', 'UP', 'text', 'file', 'www.google.com', 30, 100 and all integer and depend on given **id**.
If **id** was ***srcip***, value is a string and If **id** was ***bodysize***, value is a number. If value is a string, you may quote this string with ' ' and " " or not. E.g. string :'192.168.1', "UP", 202.69.64.68, file, 'text' and so on number: 30, 32, 68 and so on.

cp(compare operator) is a compare operator between **id** and **value**. Possible **cp** are '=', '==', '<', '<=', '=<', '>', '>=', '=>' and '!='. This symbols are used to c or c++ language. E.g. '***bodysize*** > 30', '***bodysize*** = 130' and so on.

lp(logical operator) is a logical operator to specify of two or more qualifiers. Possible **lp** are 'and' and 'or'. If expression was made of two or more qualifiers, always each qualifiers will specify with **lp** ('and' or 'or') and parenthesis ('(', ')'). E.g. ***srcip*** 192.168.1 and ***destip*** '192.168.1', (***cntype*** 'text' and ***string*** "ngtrace") or (***cntyp*** "file" and ***direction*** UP) and so on.

If you want all packets on the network be dumped, you type "All".

A qualifier format

id [cp] value
All

A expression format

qualifier1 [[lp1] qualifier2 [lp2] qualifier3 ...]
id [cp] value [[lp] **id** [cp] value [lp] **id** [cp] value ...]

Common filter expressions

Srcip 192.168.1.101 | 202.96.69 | 192.168
Destip 192.168.1.101 | 202.96.69 | 192.168

ICMP filter expressions

Msgtype > 0 and ***Msgtype*** < 41

Msgcode 0 | 1 | 2 | 3

bodysize > 3000

E.g : ***srcip*** ‘192.168.1’ and ***bodysize*** >= 30

Source ip address is 192.168.1.* and binary data to transfer is more than 30 bytes.

HTTP filter expressions

domain google.com

server 192.168.1.1 | 192.168.1 | 192.168

Client 192.168.1.1 | 192.168.1 | 192.168

url www.google.com/index.html

method HEAD|GET|POST|PUT|DELETE|TRACE|OPTIONS|CONNECT

cntype text/html

string ‘abcd efg’

bodysize > 3000

E.g : ***domain*** google.com and ***cntype*** html and ***string*** ‘ngtrace’

Domain name contains string ‘Google’, file format is html and html file contains string ‘ngtrace’.

FTP filter expressions

server 192.168.1.1

Client 192.168.1.1 | 192.168.1 | 192.168

method RETR|STOR|LIST

E.g : ***client*** 192.168.1 and (***method*** ‘RETR’ or ***method*** “STOR” or ***method*** DELE)

Client IP address is 192.168.1.* and, FTP request command is RETR(retrieve) or STOR (store) or DELETE(delete file).

IM filter expressions

server 192.168.1.1

direction UP|DOWN

cntype text/file

string ‘abcd efg’

E.g : (***cntype*** text and ***string*** ‘ngtrace’) or (***cntype*** file and ***direction*** UP)

Message type is conversation and message contains ‘ngtrace’ or, Message type is file transfer and direction is upload.

SMTP/POP filter expressions

from {address of mail sender} (ex: ***from*** aaa@yahoo.com)
to {address of mail receiver} (ex: ***to*** bbb@hotmail.com)
attachment {yes | no} (ex: ***attachment*** yes)
contents {string} (ex: ***contents*** 'abcd efg')

7.3 Dissectors

The Sniffer support as plugins / dynamically loadable object modules the following protocol dissector, grouped by their type.

- Simple protocol dissectors – The so called "simple protocol dissectors "are not typical protocol dissectors in the sense that they do none or very little processing of the data payloads of the packets.

  They are usually used for quick and brief auditing of network interactions by inspecting only the packet headers.

  - ✓ ICMP protocol dissector – The ICMP protocol dissector is fed, by the appropriate filter rules, all or some of the ICMP traffic in the network.

    The dissector stores brief audit information in the database about the exchanged ICMP packets.

    It examines the IP packet headers to extract the packet's source, destination etc.

    It further inspects the ICMP headers to determine the ICMP Type and Code.

    It then prepares, formats etc an audit entry from the gathered data and, depending on the matched rule target, forwards the information to be stored in the DBMS or to the Notifier.

- Complex protocol dissector

  - ✓ Protocol dissectors for handling e-mail messages - The protocol dissectors for handling e-mail communication permit the interception of e-mail messages and of any files attached to the e-mails.

    Several specific dissectors are implemented to deal with the different e-mail protocols of use.

They handle the data (conversation and attached files) extraction from the network flow and deal with protocol specifics and details like multi-part e-mail etc.

The e-mail protocol dissectors achieve their goal by gathering the data from the used protocol headers (IP, TCP, SMTP, POP, IMAP etc) and the actual information exchanged.

They consult the high level filter rules, specific to thee-mail protocol dissectors and then take action depending on the rules' targets e.g. store the information in the datastores for future reference and audit purposes (if a rule with Notify target has matched, the packed entry is handled to the Notifier).

· SMTP protocol dissector – Handles the processing of outgoing mail.

· POP 3 protocol dissector – Handles the processing of the e-mail messages being downloaded by the e-mail clients from the servers' mailboxes.

· IMAP protocol dissector

✓ Dissectors for handling IM protocols – This section describes the supported Instant Messaging protocols and the common functionalities supported by them.

All IM protocol dissectors support filtering based on the contents of the messages exchanged and protocol specific attributes.

They permit reconstruction of the conversations that took place and the files that were exchanged, using a given IM protocol.

· ICQ

· IRC

· Jabber

· AIM

· Skype – only notices on the event of detecting Skype conversations and/or file transfers are stored, since the Skype protocol is encrypted

✓ File transfer/exchange protocol dissectors – The file transfer protocol dissector handle different protocols for exchanging files.

The associated high level filter rules and the protocol dissectors themselves are intended to handle the files being up and downloaded and gather data for the transfers' source and destination

· FTP

· SMB/CIFS (Windows file sharing)

· NFS

· Bittorrent

✓ HTTP protocol dissectors

· The HTTP/WebDAV protocol dissector permits high level content filtering and the full reconstruction of HTTP/WebDAV sessions that took place in the past.

To do so it processes the HTTP traffic (including cookies etc), extracts and puts in the data-stores the needed information.

It uses a files systems based cache for storage of static files like images, CSS, JavaScript etc files, as well as whole static web pages.

The sessions are stored partly in the databases and partly inthe file-system data stores where the data in the RDBMS may reference static files from the file-system based data stores. Thus in some cases link (<a href>) etc tags may need to be properly rewritten, so that the web interaction that took place can be reconstructed and replayed the way it happened.In the event of HTTPS encrypted communications, only a brief audit notice is stored in the database, so that an evidence that the interaction has taken place is present.

✓ VPN protocol dissectors – The VPN protocol dissectors permit the processing of data communications that took place over the corporate VPN tunnels (IPSec etc) is the encryption keys used are available and properly configured.

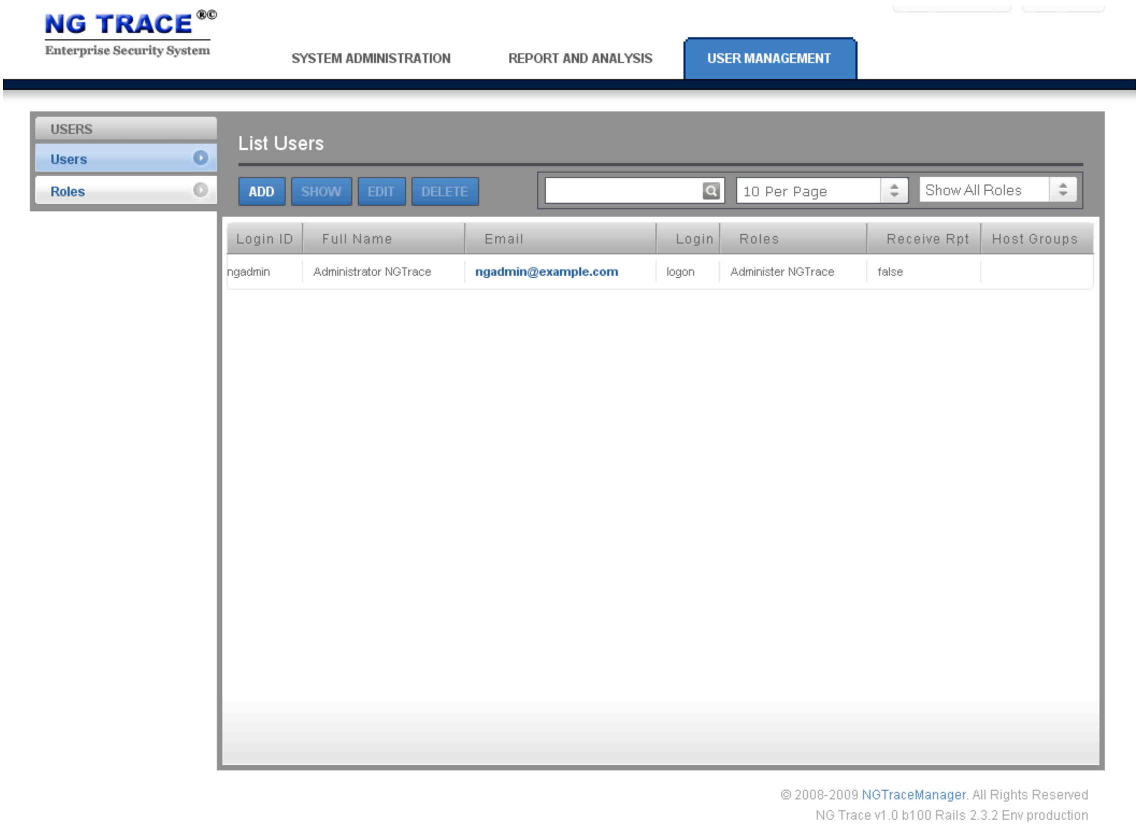
The VPN protocol dissector is used as a kind of preprocessor, which handles decryption and other specific tasks that should be taken care of before the processing is tunneled to the other higher level protocol dissectors like HTTP, IM, e-mail, File transfer etc.

8. System Administration

8.1 Users and Privilege

8.1.1 Users

User management section provides functionalities such as adding and deleting of user, displaying and modifying user’s information.



Login ID	Full Name	Email	Login	Roles	Receive Rpt	Host Groups
ngadmin	Administrator NGTrace	ngadmin@example.com	logon	Administer NGTrace	false	

© 2008-2009 NGTraceManager. All Rights Reserved
NG Trace v1.0 b100 Rails 2.3.2 Env production

Figure 6. User Management Tab.

User information contains user login ID, user name, e-mail address, phone number, registered date, modified date, option whether or not to receive notify information from notifier, login status, login date/time, user role, manageable host groups, description and etc.

- [ADD] :

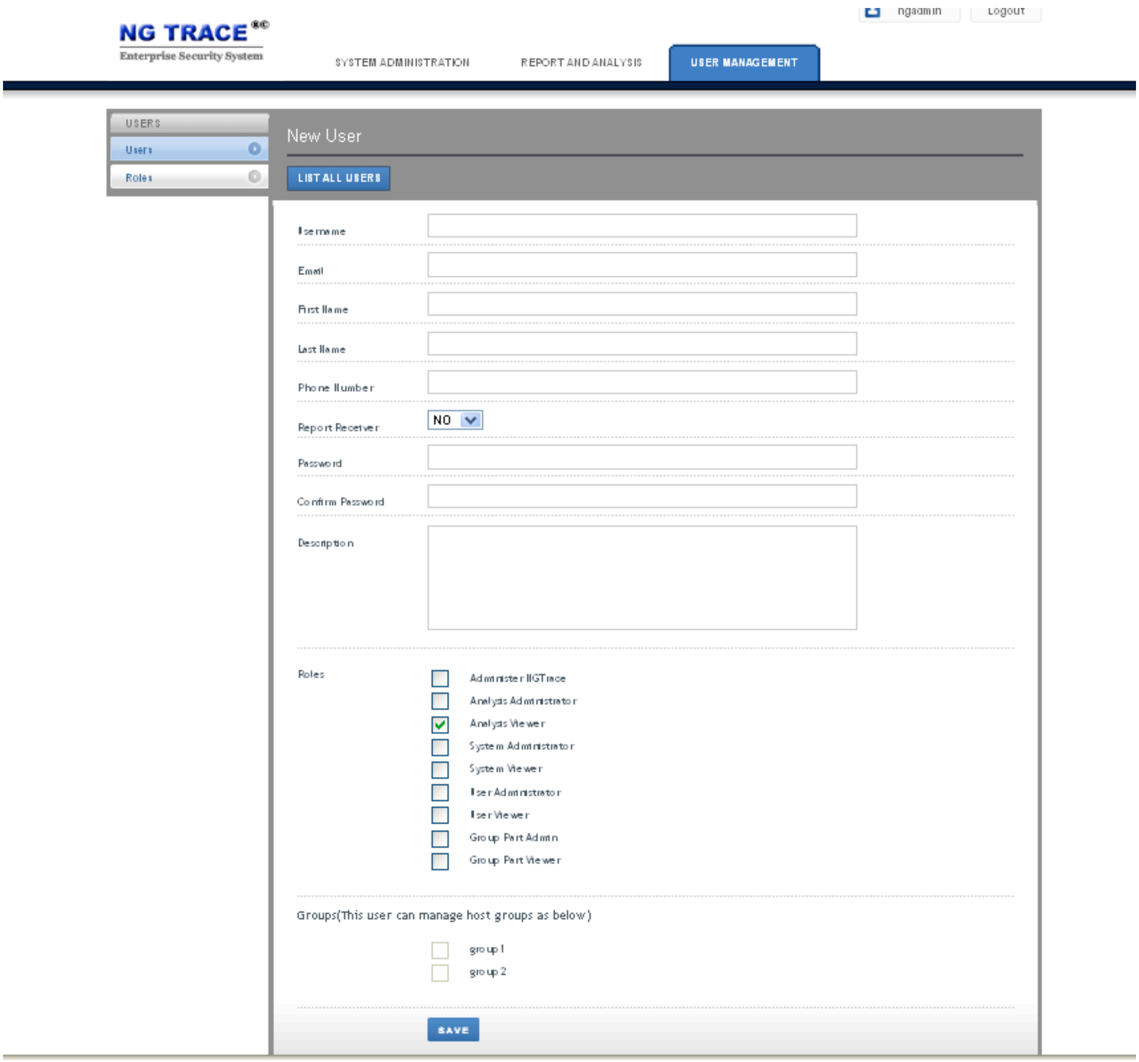
Register new user.

Enter user information of new user.

Login ID and e-mail address should not be same as the ones of already registered users.

Symbol “*” indicates mandatory fields. If mandatory field is left blank, user registration is not processed.

If user role “Administer NG Trace” is selected, other user role options become disabled. “Groups” item becomes activated only if user role “Group Part Admin” or “Group Part Viewer” is selected.



New User

Username

Email

First Name

Last Name

Phone Number

Report Receiver NO

Password

Confirm Password

Description

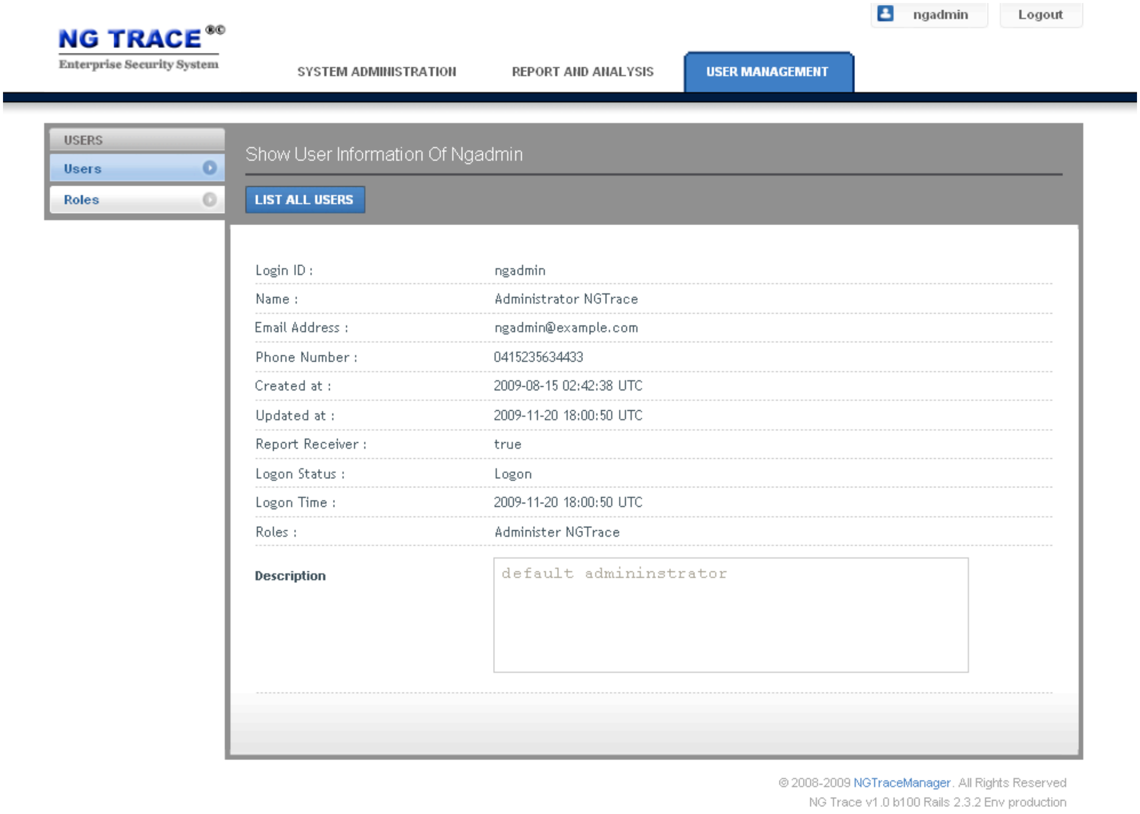
Roles
□ Administer NGTrace
□ Analysis Administrator
☑ Analysis Viewer
□ System Administrator
□ System Viewer
□ User Administrator
□ User Viewer
□ Group Part Admin
□ Group Part Viewer

Groups(This user can manage host groups as below)
□ group1
□ group2

Figure 7. Add User

- [SHOW]

It displays information of registered user.



Show User Information Of Ngadmin

Login ID :	ngadmin
Name :	Administrator NGTrace
Email Address :	ngadmin@example.com
Phone Number :	0415235634433
Created at :	2009-08-15 02:42:38 UTC
Updated at :	2009-11-20 18:00:50 UTC
Report Receiver :	true
Logon Status :	Logon
Logon Time :	2009-11-20 18:00:50 UTC
Roles :	Administer NGTrace
Description	default admininstrator

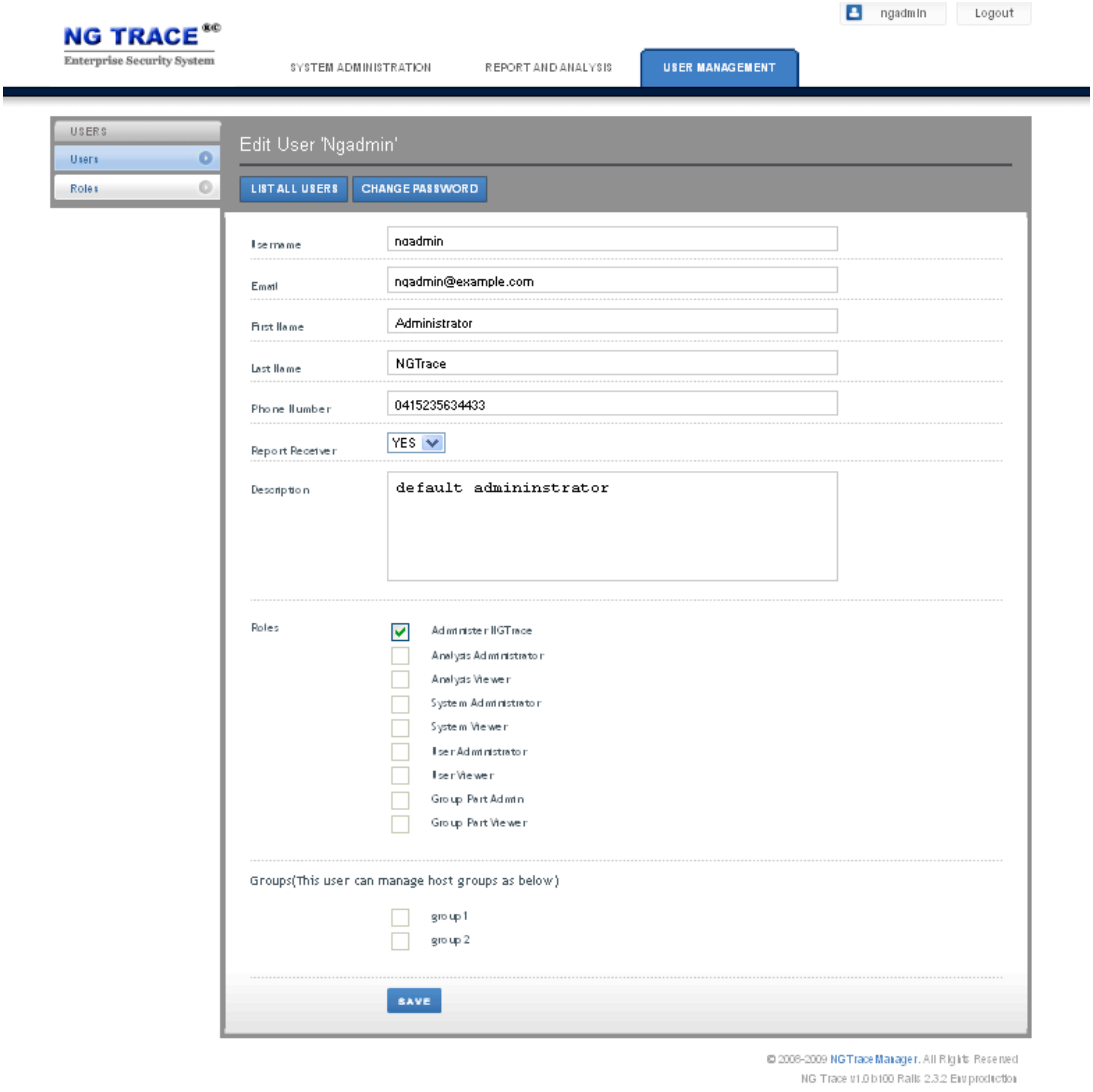
© 2008-2009 NGTraceManager. All Rights Reserved
NG Trace v1.0 b100 Rails 2.3.2 Env production

Figure 8. Show User Information

- [EDIT]

You can edit information and password of registered user here.

If user role “Administer NG Trace” is selected, other user role options become disabled. “Groups” item becomes activated only if user role “Group Part Admin” or “Group Part Viewer” is selected.



Edit User 'Ngadmin'

Username	ngadmin
Email	ngadmin@example.com
First Name	Administrator
Last Name	NGTrace
Phone Number	0415235634433
Report Receiver	YES
Description	default admininstrator

Roles

☑ Administer NGTrace
□ Analysis Administrator
□ Analysis Viewer
□ System Administrator
□ System Viewer
□ User Administrator
□ User Viewer
□ Group Part Admin
□ Group Part Viewer

Groups(This user can manage host groups as below)

□ group 1
□ group 2

© 2008-2009 NGTraceManager. All Rights Reserved
NG Trace v1.0 b100 Rails 2.3.2 Env production

Figure 9. Edit User

To change the Password, input new Password and Confirm Password, press [SAVE].

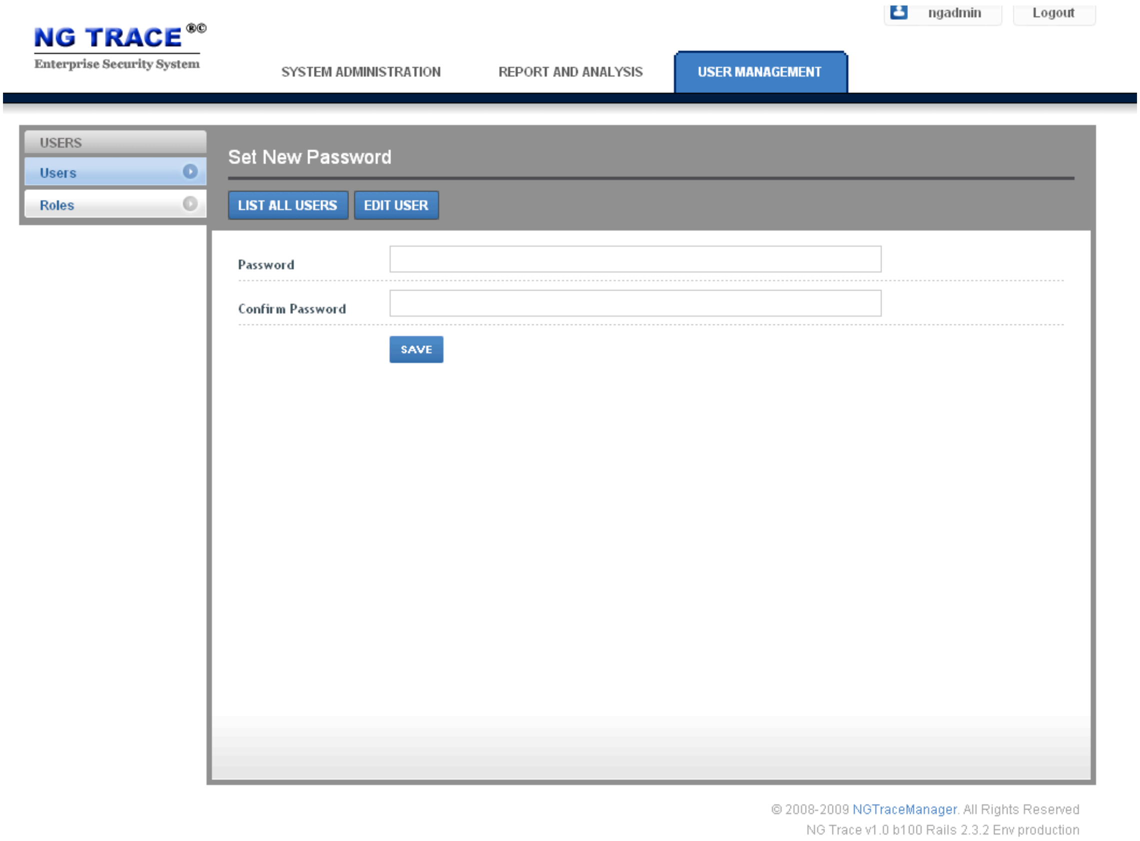


Figure 10. Set New Password

- [DELETE]

User selected in the list can be removed.

8.1.2 Privilege

NG Trace Management Console provides several user privileges such as Administer NGTrace, Analysis Administrator, Analysis Viewer, System Administrator, System, Viewer, User Administrator, User Viewer, Group Part Admin and Group Part Viewer.

- Administer NG Trace

  The user of an account with Administer NG Trace privilege has all privilege of NG Trace System.

  That is, can manage all information within SYSTEM ADMINISTRATION page and REPORT AND ANALYSIS page and USER MANAGEMENT.
- Analysis Administrator

  The user with this privilege has access to all functionalities of REPORT

AND ANALYSIS page.

- Analysis Viewer

  The user with this privilege can only view information of REPORT AND ANALYSIS page.

- System Administrator

  The user with this privilege has access to all functionalities of SYSTEM ADMINISTRATION page.

- System viewer

  The user with this privilege can only view information of SYSTEM ADMINISTRATION page.

- User Administrator

  The user with this privilege has access to all functionalities of USER MANAGEMENT page.

- User Viewer

  The user with this privilege can only view information of USER MANAGEMENT page.

- Group Part Admin

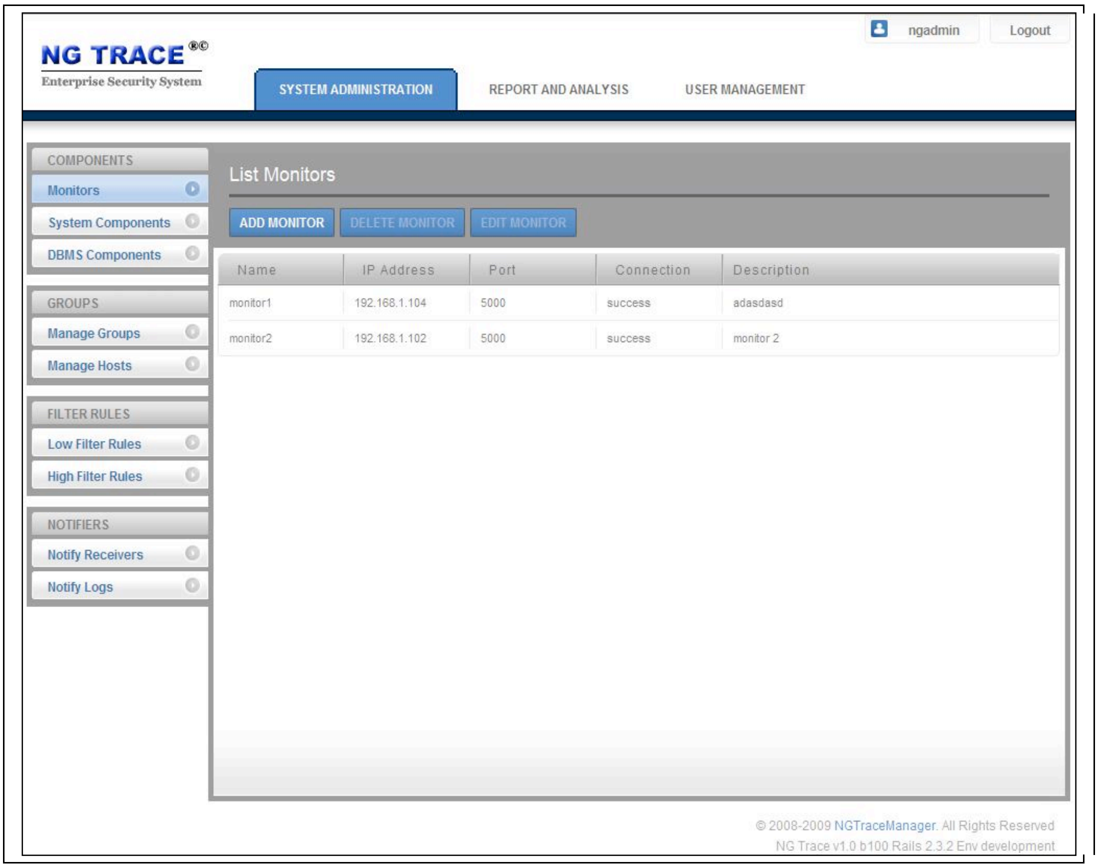
- Group Part Viewer

8.2 Software Component management

8.2.1 COMPONENTS

SYSTEM ADMINISTRATON page manage software components.

Software components include the monitors, executable components, DBMS components.



List Monitors

Name	IP Address	Port	Connection	Description
monitor1	192.168.1.104	5000	success	adasdasd
monitor2	192.168.1.102	5000	success	monitor 2

© 2008-2009 NGTraceManager. All Rights Reserved
NG Trace v1.0 b100 Rails 2.3.2 Env development

Figure 11. System Administration

- **Monitors**

  Daemon registration information for remote control the software components on web console.

  NG Monitor does administrations for various software components that installed in node.

  It has to installation unconditional to every Node, and management of components by administrator of start, stop state etc, is attained through service that offer in monitor.

  Figure 8 show installed monitor's list and it's information.

  - Name: Name of installed Monitor.
  - IP Address: IP Address of computer that monitor is installed.
  - Port: Communication port number
  - Connection: Communication status.

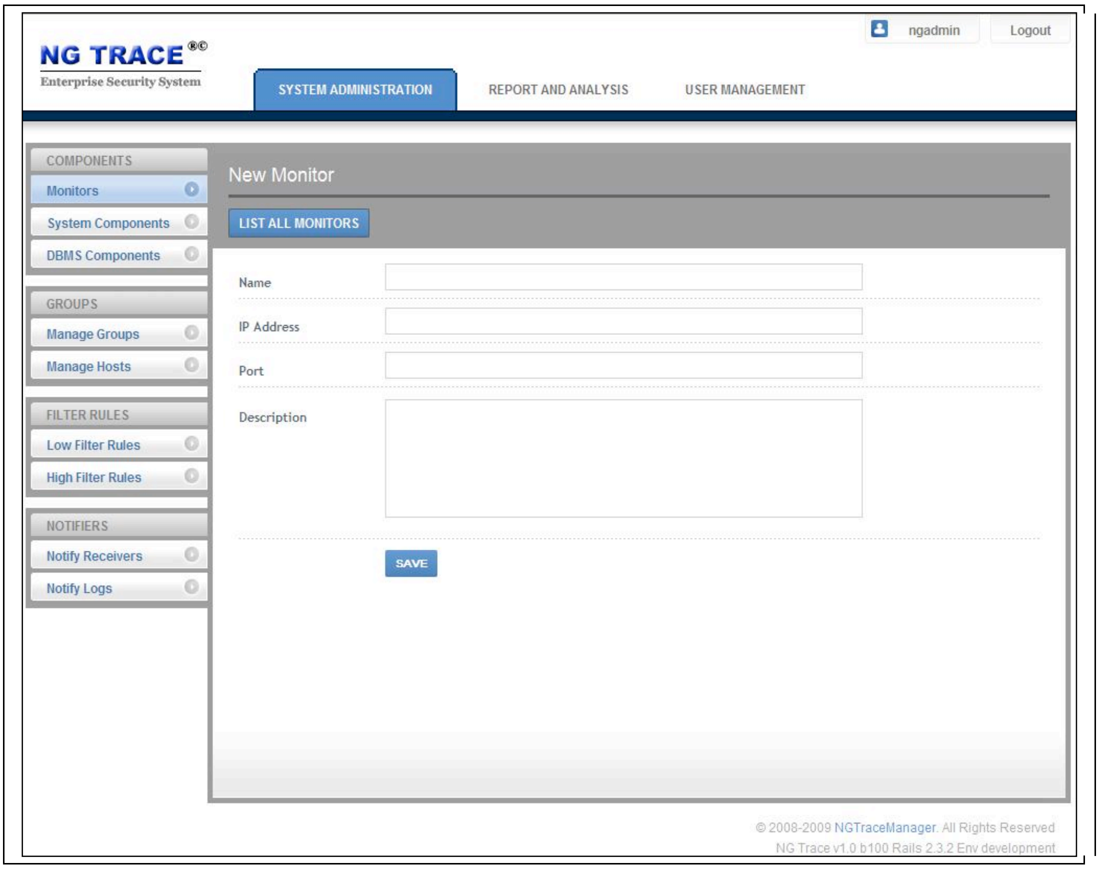
    true: In communication

    Failed: Connection Failure

- Description: Description of Monitor

- [ADD MONITOR]

  Add newly installed Monitor.



New Monitor

Name

IP Address

Port

Description

© 2008-2009 NGTraceManager. All Rights Reserved
NG Trace v1.0 b100 Rails 2.3.2 Env development

Figure 12. New Monitor

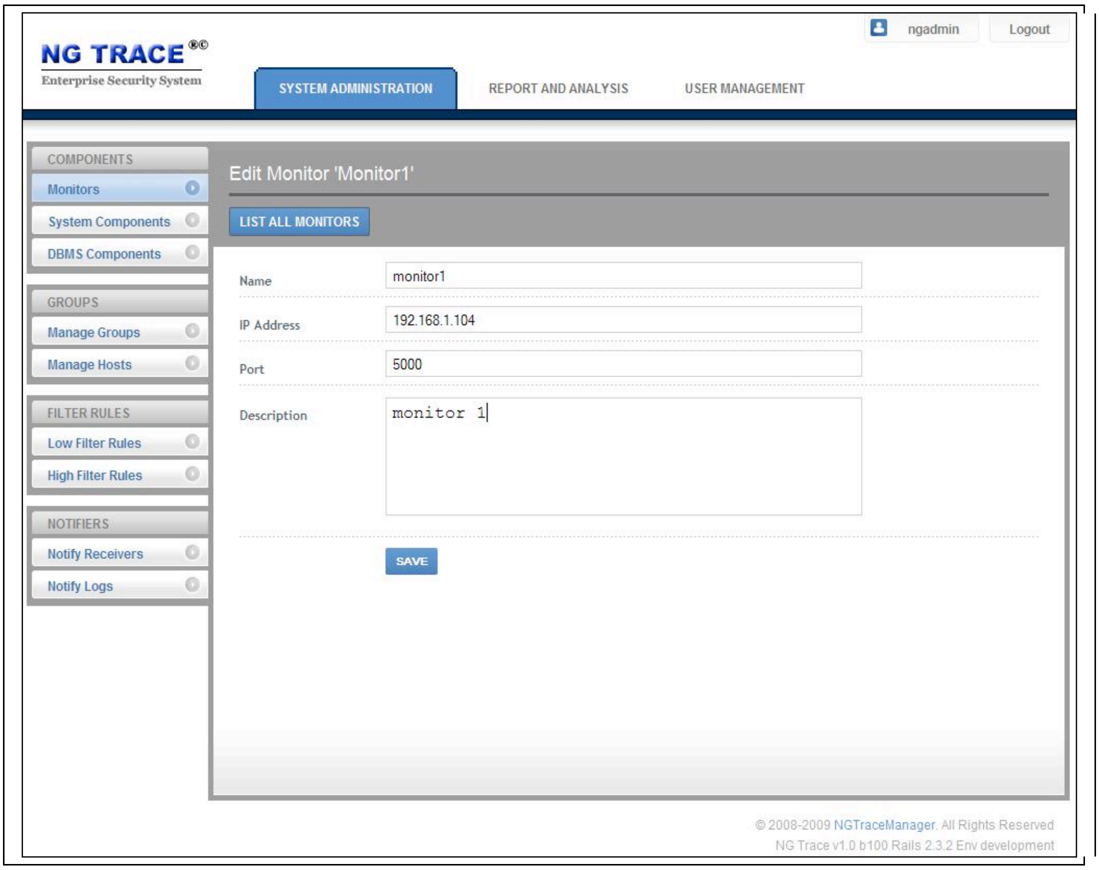
- Name; Monitor's name to add.
- IP Address: IP Address of computer that monitor is installed.
- Port; Communication port number.
- Description: Monitor's description to add.

- [DELETE MONITOR]

  Delete selected Monitor.

- [EDIT MONITOR]

  Information of selected Monitor, that is, can change Name, IP Address, Port, Description.



Edit Monitor 'Monitor1'

Name: monitor1

IP Address: 192.168.1.104

Port: 5000

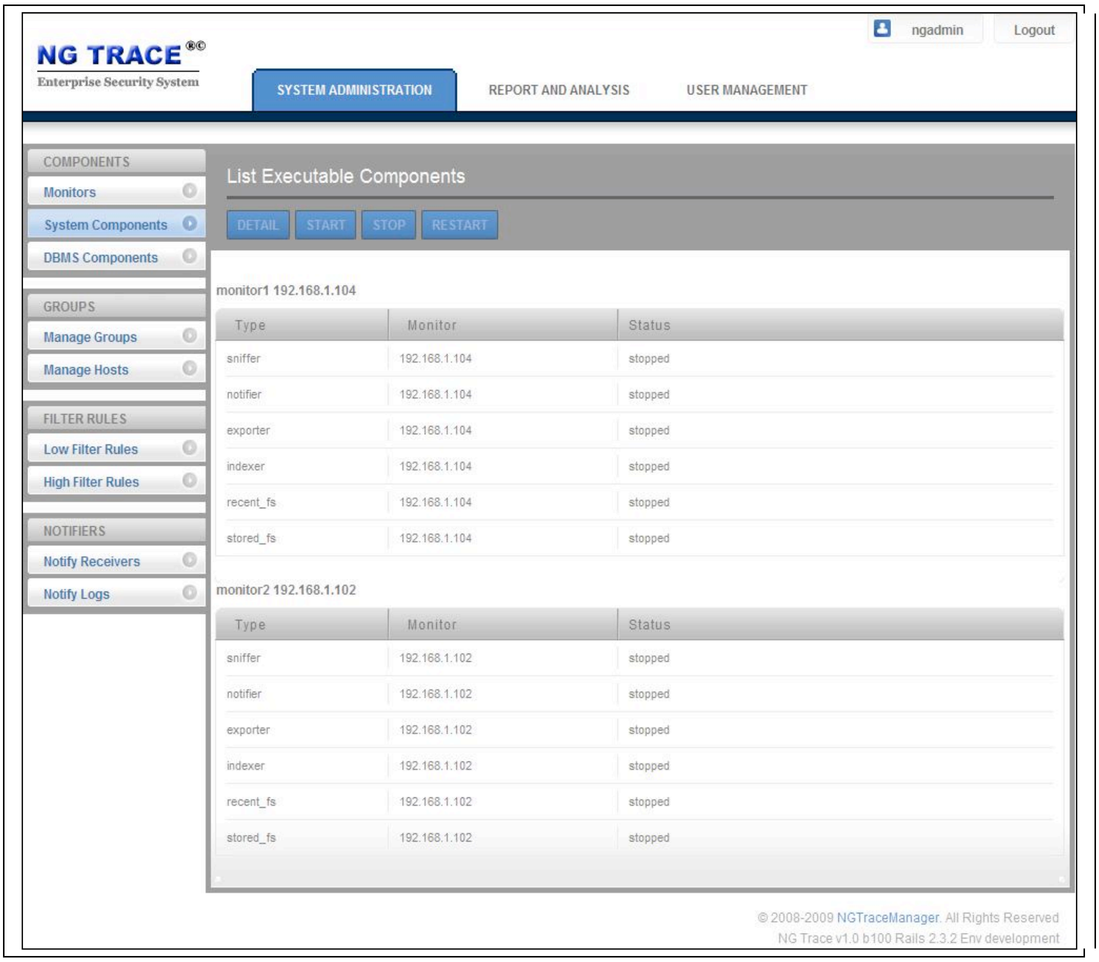
Description: monitor 1

Figure 13. Edit Monitor

- **System Component**

  It shows list of executive components that link to by monitor.

  Components that can connect to Monitor are Sniffer, Notifier, Exporter, Indexer, recent_fs, and Stored_fs.



List Executable Components

monitor1 192.168.1.104

Type	Monitor	Status
sniffer	192.168.1.104	stopped
notifier	192.168.1.104	stopped
exporter	192.168.1.104	stopped
indexer	192.168.1.104	stopped
recent_fs	192.168.1.104	stopped
stored_fs	192.168.1.104	stopped

monitor2 192.168.1.102

Type	Monitor	Status
sniffer	192.168.1.102	stopped
notifier	192.168.1.102	stopped
exporter	192.168.1.102	stopped
indexer	192.168.1.102	stopped
recent_fs	192.168.1.102	stopped
stored_fs	192.168.1.102	stopped

Figure 14. List Executable Component

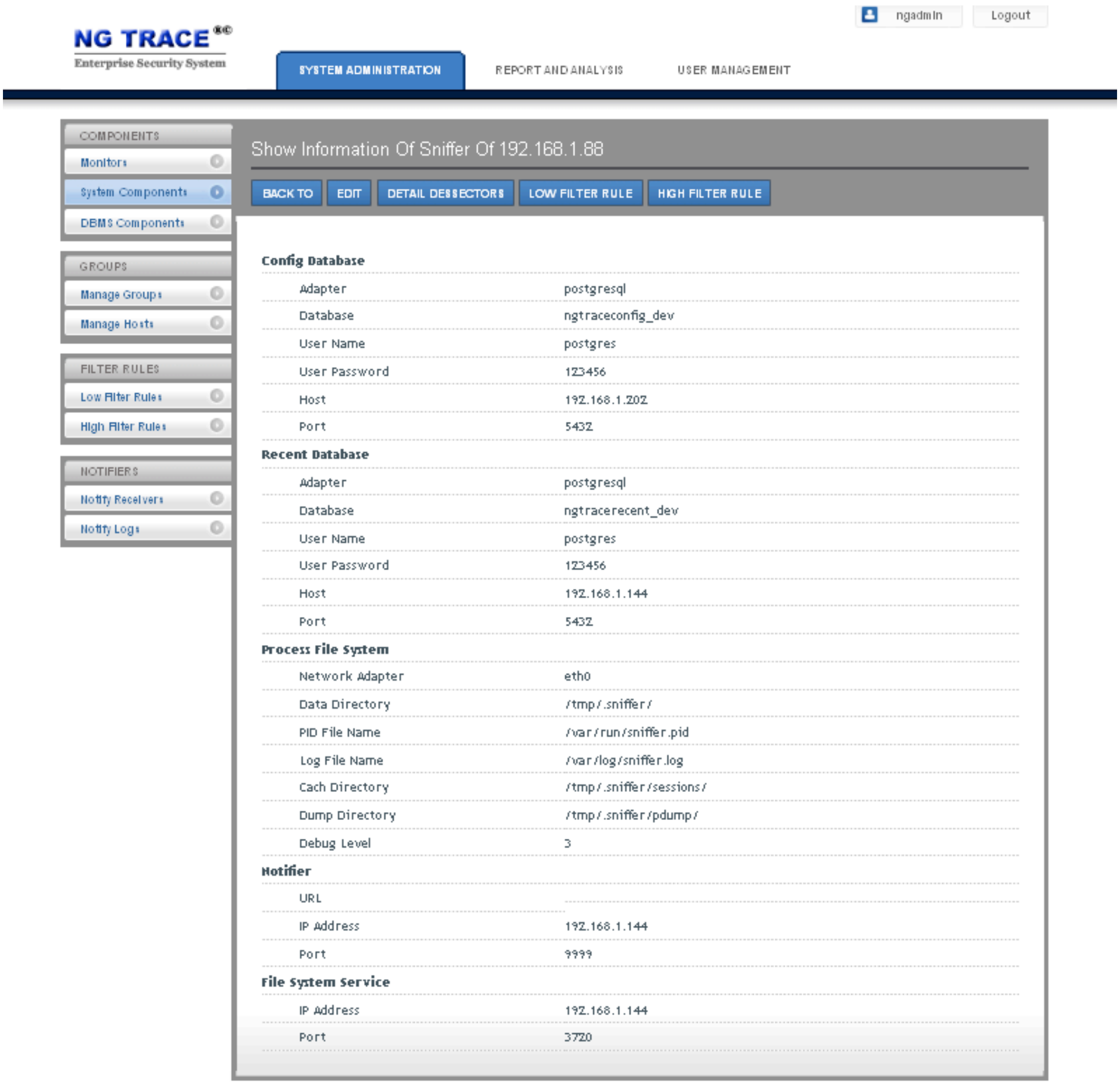
- Type: Type of connected component.
- Monitor: IP Address of connected Monitor.
- Status: Status of component. “stopped” if component has been stopped , “started with pid *pid*” if component is currently running.

- [Detail]: Shows information of component.

  ✓ **Sniffer**

  Shows information of Sniffer’s Configuration file .

  For more information on position, see **Chapter 5 Configuration - 5.1 Sniffer** in this document.



Show Information Of Sniffer Of 192.168.1.88

Config Database	
Adapter	postgresql
Database	ngtraceconfig_dev
User Name	postgres
User Password	123456
Host	192.168.1.202
Port	5432
Recent Database	
Adapter	postgresql
Database	ngtracerecent_dev
User Name	postgres
User Password	123456
Host	192.168.1.144
Port	5432
Process File System	
Network Adapter	eth0
Data Directory	/tmp/.sniffer/
PID File Name	/var/run/sniffer.pid
Log File Name	/var/log/sniffer.log
Cach Directory	/tmp/.sniffer/sessions/
Dump Directory	/tmp/.sniffer/pdump/
Debug Level	3
Notifier	
URL	
IP Address	192.168.1.144
Port	9999
File System Service	
IP Address	192.168.1.144
Port	3720

Figure 15. Show Information of Sniffer

- [EDIT]
  Change information of sniffer.

- [Low Filter Rule]
  Shows Low Level Filter Rule list of selected Sniffer..
  For more information on position, refer to "7.2.3 FILTER RULES".

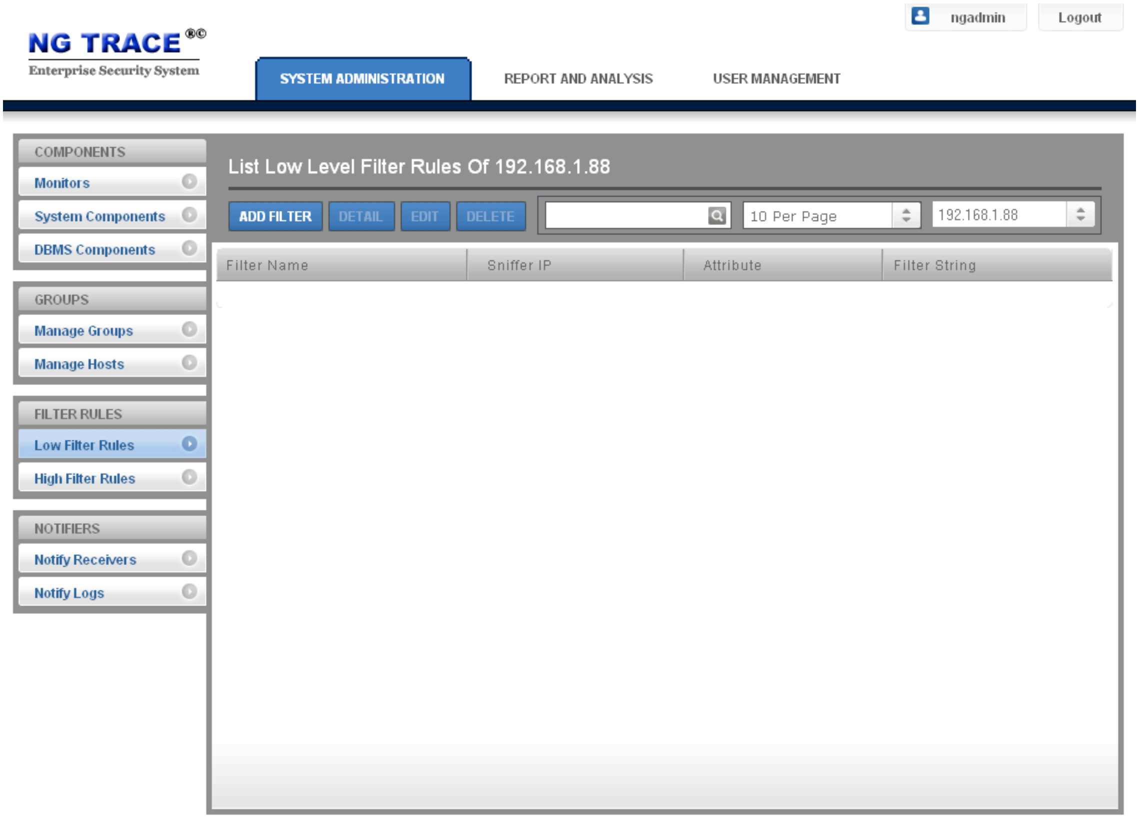
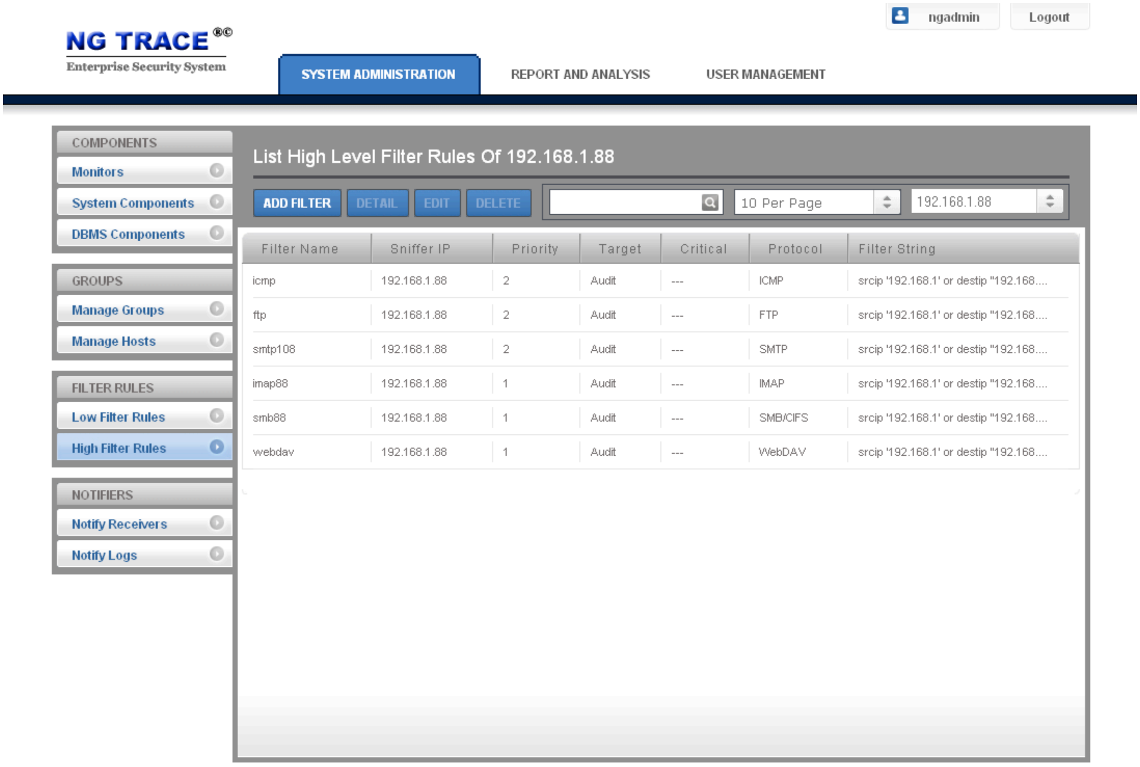


Figure 16. List Low – Level Filter Rules

- [High Filter Rule]
  Show High Level Filter Rule list of selected Sniffer.
  For more information on position, see "7.2.3 FILTER RULES".



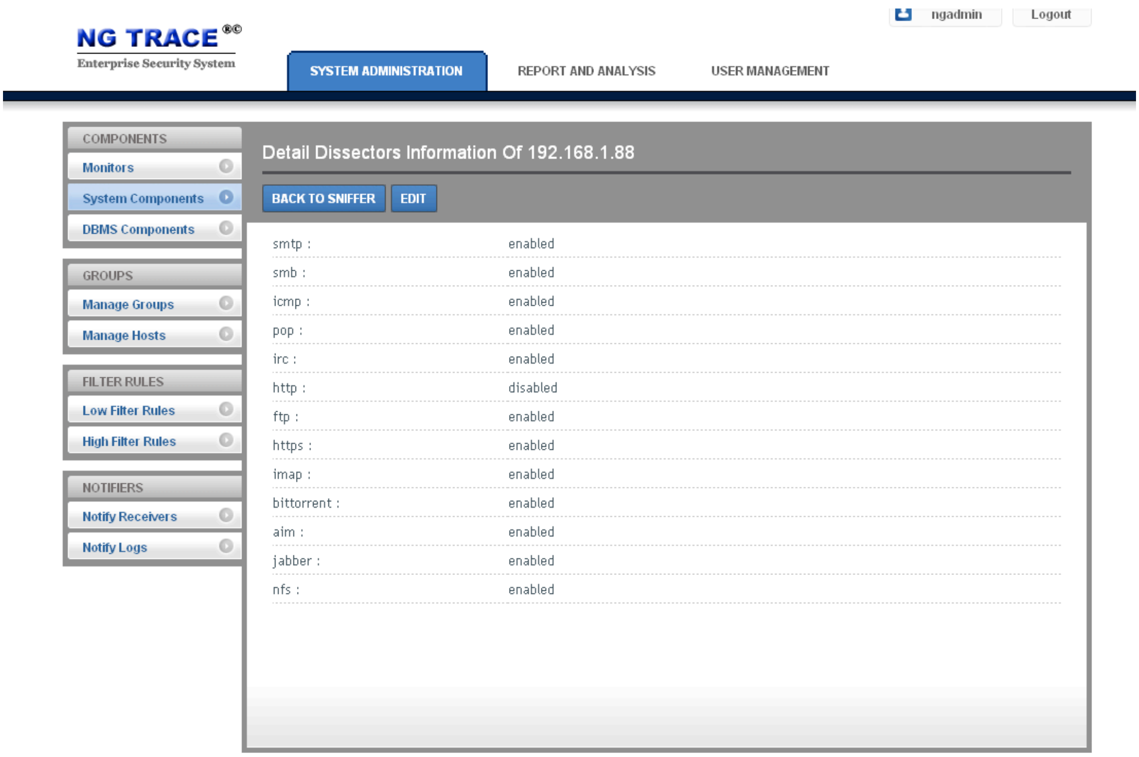
List High Level Filter Rules Of 192.168.1.88

Filter Name	Sniffer IP	Priority	Target	Critical	Protocol	Filter String
icmp	192.168.1.88	2	Audit	---	ICMP	srcip '192.168.1' or destip "192.168....
ftp	192.168.1.88	2	Audit	---	FTP	srcip '192.168.1' or destip "192.168....
smtp108	192.168.1.88	2	Audit	---	SMTP	srcip '192.168.1' or destip "192.168....
imap88	192.168.1.88	1	Audit	---	IMAP	srcip '192.168.1' or destip "192.168....
smb88	192.168.1.88	1	Audit	---	SMB/CIFS	srcip '192.168.1' or destip "192.168....
webdav	192.168.1.88	1	Audit	---	WebDAV	srcip '192.168.1' or destip "192.168....

Figure 17. List High Level Filter Rule

- [Detail Dissectors]

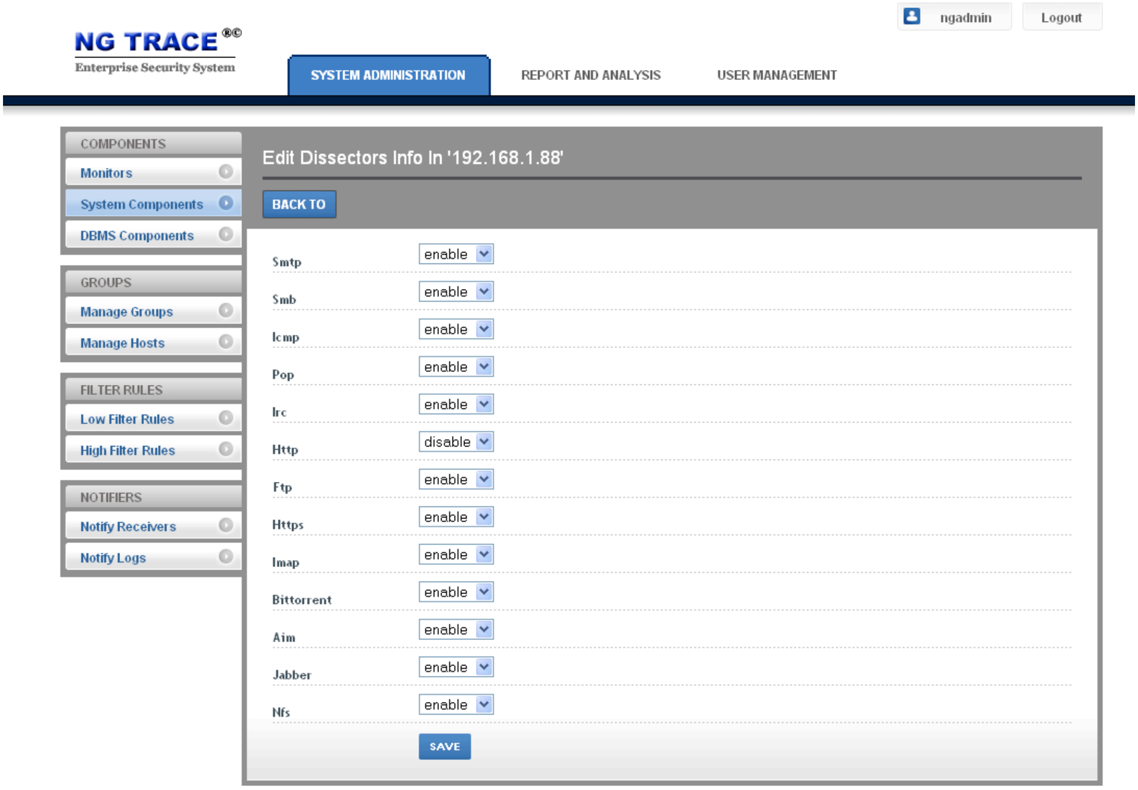
Load status of dissectors which can be loaded in Sniffer is displayed or changed.



Detail Dissectors Information Of 192.168.1.88

smtp :	enabled
smb :	enabled
icmp :	enabled
pop :	enabled
irc :	enabled
http :	disabled
ftp :	enabled
https :	enabled
imap :	enabled
bittorrent :	enabled
aim :	enabled
jabber :	enabled
nfs :	enabled

Figure 18. Status of dissectors



Edit Dissectors Info In '192.168.1.88'

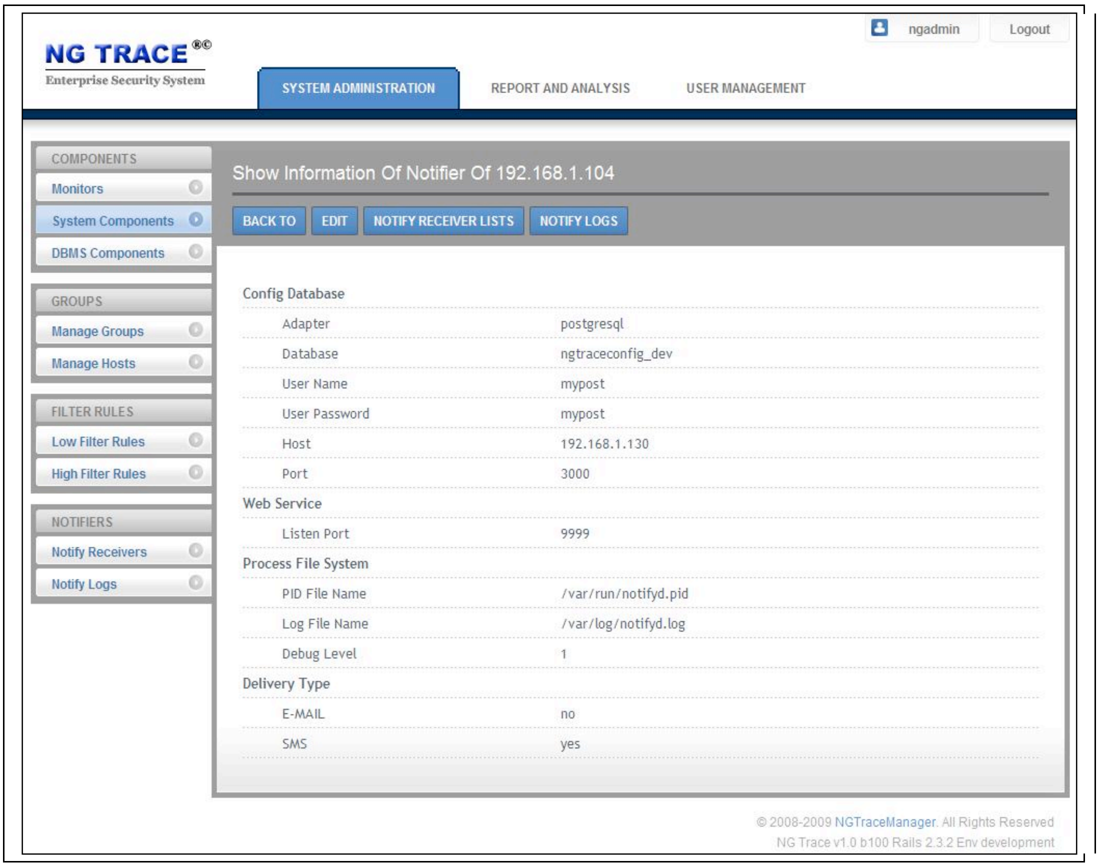
Smtp	enable
Smb	enable
Icmp	enable
Pop	enable
Irc	enable
Http	disable
Ftp	enable
Https	enable
Imap	enable
Bittorrent	enable
Aim	enable
Jabber	enable
Nfs	enable

Figure 19. Edit the status of dissectors

✓ **Notifier**

Show information of Notifier's Configuration file .

For more information on position, see **Chapter 5 Configuration - 5.2 Notifier** in this document.



Show Information Of Notifier Of 192.168.1.104

Config Database	
Adapter	postgresql
Database	ngtraceconfig_dev
User Name	mypost
User Password	mypost
Host	192.168.1.130
Port	3000
Web Service	
Listen Port	9999
Process File System	
PID File Name	/var/run/notifyd.pid
Log File Name	/var/log/notifyd.log
Debug Level	1
Delivery Type	
E-MAIL	no
SMS	yes

Figure 20. Show Information Of Notifier

- [EDIT]

  Change information of Notifier.

- [NOTIFY RECEIVER LISTS]

  For more information on position, see **Chapter 7 System Administration - 7.2 SOFTWARE COMPONENT MANAGEMENT – 7.2.4 NOTIFIERS** in this document.

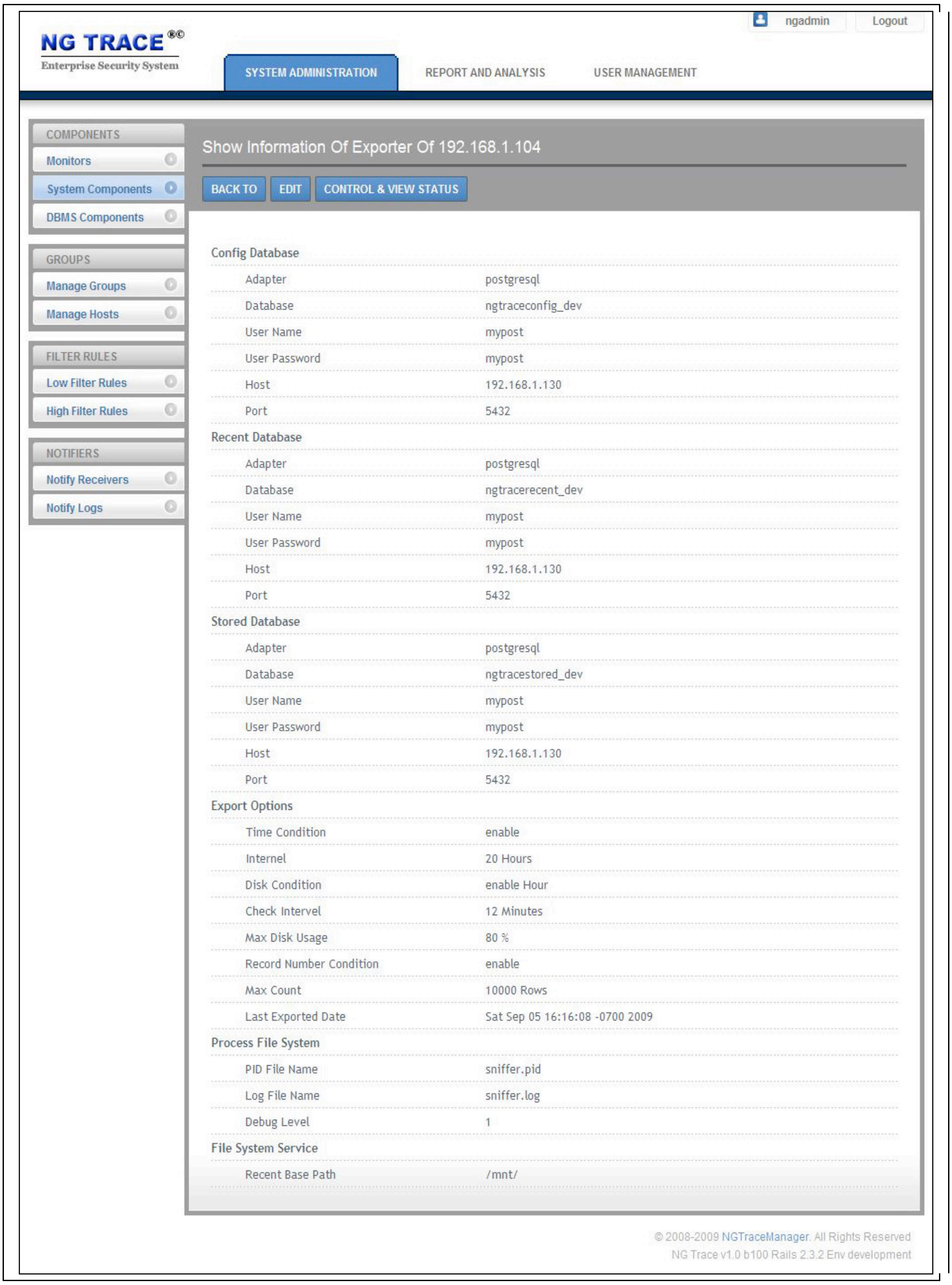
- [NOTIFY LOGS]

  For more information on position, see **Chapter 7 System Administration - 7.2 SOFTWARE COMPONENT MANAGEMENT – 7.2.4 NOTIFIERS** in this document.

✓ **Exporter**

Show information of Exporter’s Configuration file .

For more information on position, see **chapter 5 Configuration - 5.3 Exporter** in this document.



Show Information Of Exporter Of 192.168.1.104

Config Database	
Adapter	postgresql
Database	ngtraceconfig_dev
User Name	mypost
User Password	mypost
Host	192.168.1.130
Port	5432
Recent Database	
Adapter	postgresql
Database	ngtracerecent_dev
User Name	mypost
User Password	mypost
Host	192.168.1.130
Port	5432
Stored Database	
Adapter	postgresql
Database	ngtracestored_dev
User Name	mypost
User Password	mypost
Host	192.168.1.130
Port	5432
Export Options	
Time Condition	enable
Internel	20 Hours
Disk Condition	enable Hour
Check Intervel	12 Minutes
Max Disk Usage	80 %
Record Number Condition	enable
Max Count	10000 Rows
Last Exported Date	Sat Sep 05 16:16:08 -0700 2009
Process File System	
PID File Name	sniffer.pid
Log File Name	sniffer.log
Debug Level	1
File System Service	
Recent Base Path	/mnt/

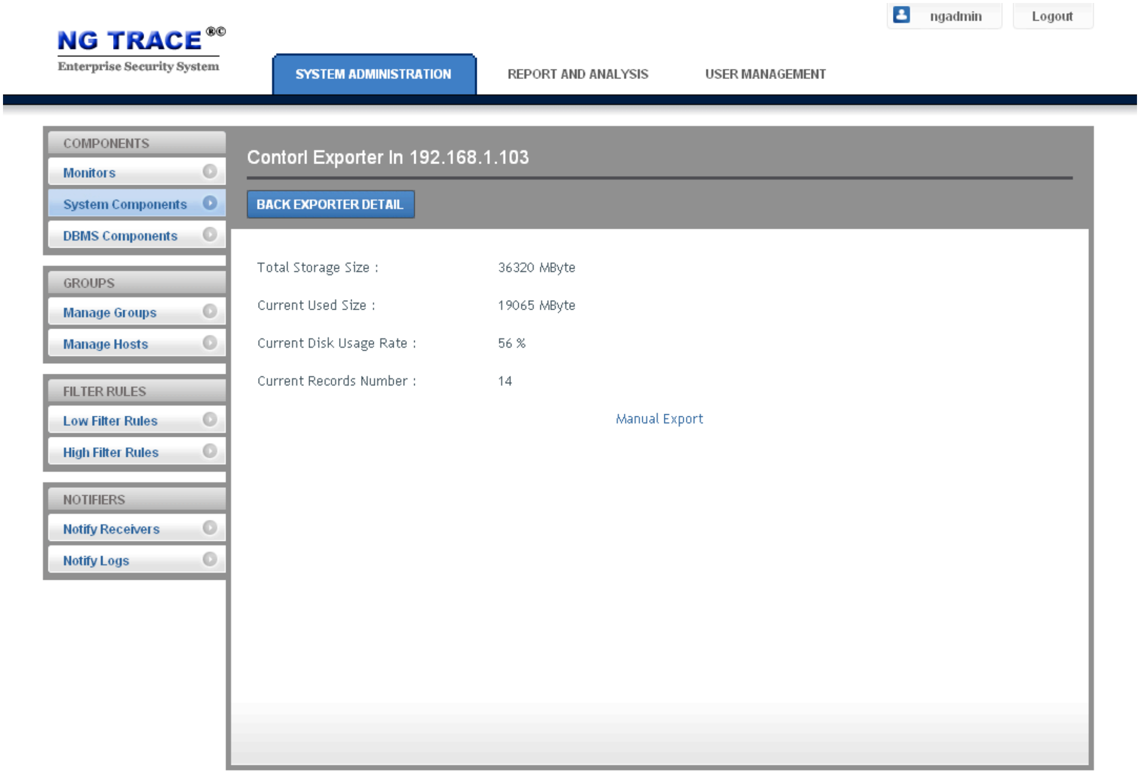
© 2008-2009 NGTraceManager. All Rights Reserved
NG Trace v1.0 b100 Rails 2.3.2 Env development

Figure 21. Show Information of Exporter

- [EDIT]
  Change information of Exporter.

- [CONTROL & VIEWSTATUS]

  Exporter see status of Recent DB or the data can do export by manual.



Contorl Exporter In 192.168.1.103

Total Storage Size :	36320 MByte
Current Used Size :	19065 MByte
Current Disk Usage Rate :	56 %
Current Records Number :	14

Figure 22. Control Exporter

Total Storage Size:
Current Used Size:
Current Disk Usage Rate:
Current Record Number: Current record number of Recent DB

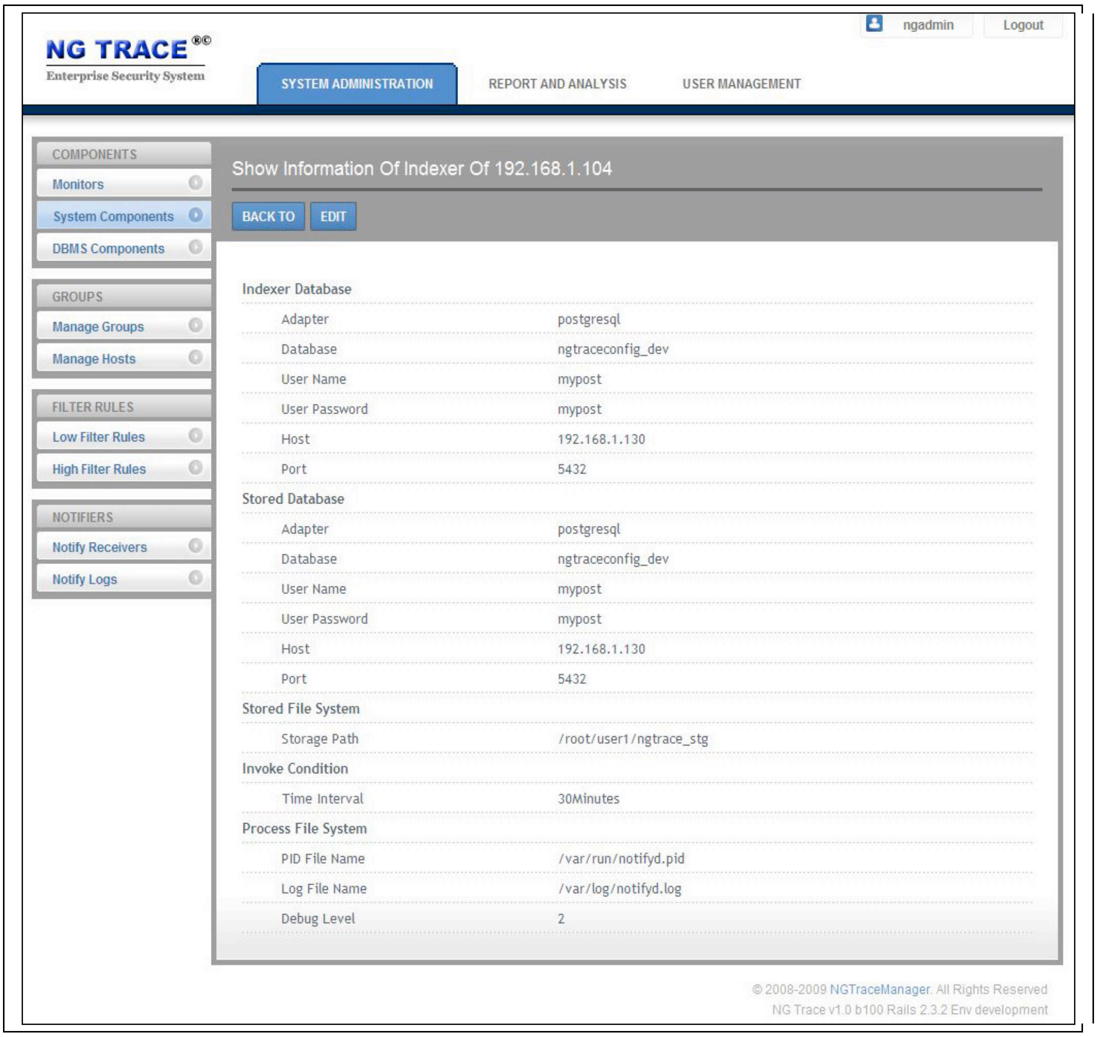
- [Manual Export]

  Export manually.

✓ **Indexer**

Show information of Indexer's Configuration file .

For more information on position, see **chapter 5 Configuration - 5.4 Indexer** in this document.



Show Information Of Indexer Of 192.168.1.104

Indexer Database	
Adapter	postgresql
Database	ngtraceconfig_dev
User Name	mypost
User Password	mypost
Host	192.168.1.130
Port	5432
Stored Database	
Adapter	postgresql
Database	ngtraceconfig_dev
User Name	mypost
User Password	mypost
Host	192.168.1.130
Port	5432
Stored File System	
Storage Path	/root/user1/ngtrace_stg
Invoke Condition	
Time Interval	30Minutes
Process File System	
PID File Name	/var/run/notifyd.pid
Log File Name	/var/log/notifyd.log
Debug Level	2

Figure 23. Show Information Of Indexer

- [EDIT]
  Change information of Indexer.

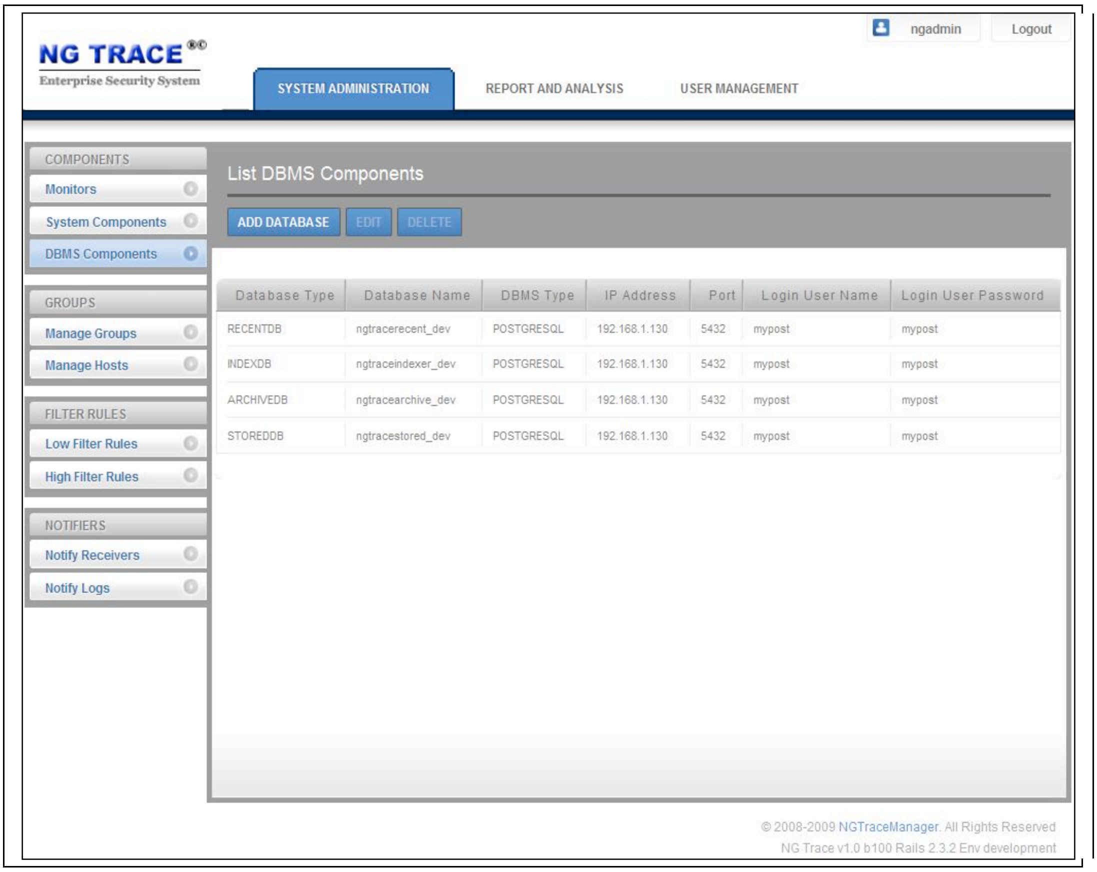
- [MANUAL INDEXING]
  Perform indexing by hand.

- ● [START]: Start component.
- ● [STOP]: Stop component.
- ● [RESTART]: Restart component.

- **DBMS Components**
  Show list of connected Database and Database Type, Database Name,

DBMS Type, IP Address, Port, Login User Name, Login User Password.

.



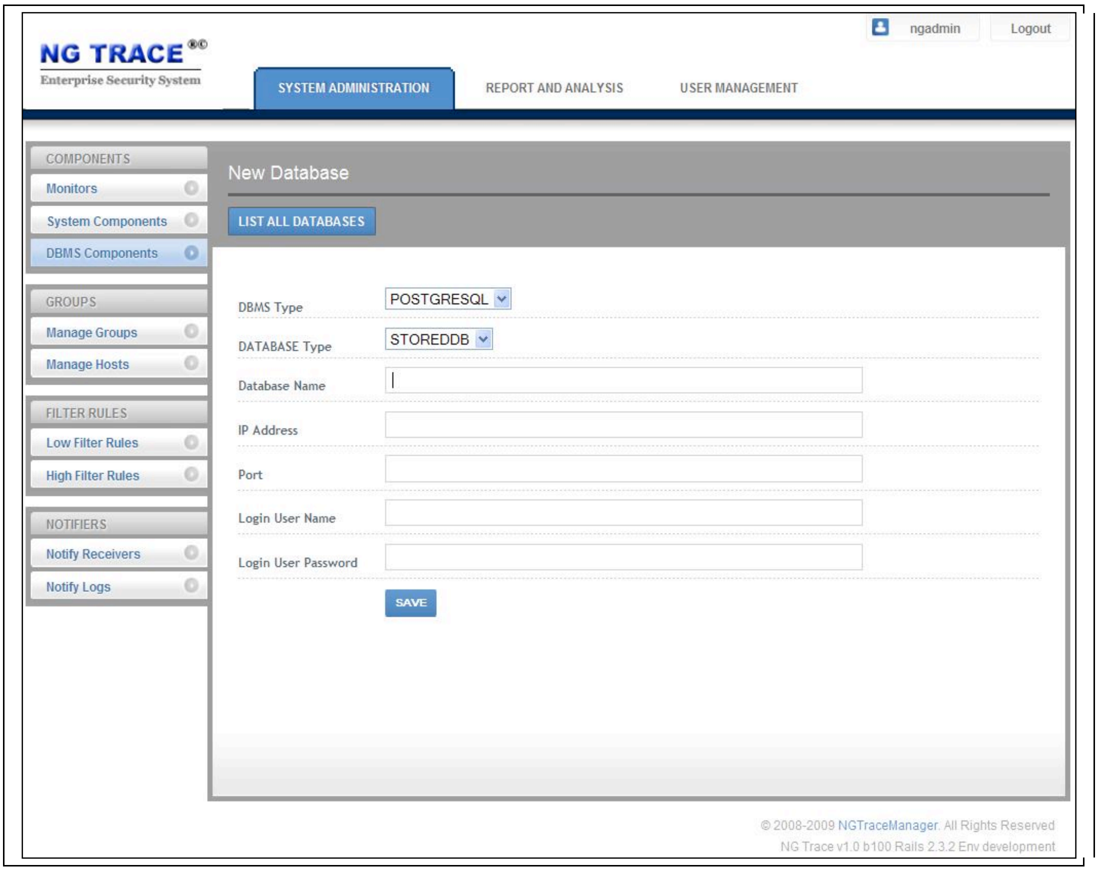
List DBMS Components

Database Type	Database Name	DBMS Type	IP Address	Port	Login User Name	Login User Password
RECENTDB	ngtracerecent_dev	POSTGRESQL	192.168.1.130	5432	mypost	mypost
INDEXDB	ngtraceindexer_dev	POSTGRESQL	192.168.1.130	5432	mypost	mypost
ARCHIVEDB	ngtracearchive_dev	POSTGRESQL	192.168.1.130	5432	mypost	mypost
STOREDDB	ngtracestored_dev	POSTGRESQL	192.168.1.130	5432	mypost	mypost

© 2008-2009 NGTraceManager. All Rights Reserved
NG Trace v1.0 b100 Rails 2.3.2 Env development

Figure 24. List DBMS Components

- [ADD DATABASE]
  Add new database.



DBMS Type	POSTGRESQL
DATABASE Type	STOREDDB
Database Name	
IP Address	
Port	
Login User Name	
Login User Password	

Figure 25. New Database

DBMS Type: Type of DBMS. (DBMS Type can be selected among POSTGRESQL, MYSQL, and ORACLE)

DATABASE Type: Type of Database. (DATABASE Type can be selected among STOREDDB, RECENTDB, INDEXDB, and ARCHIVEDB).

Database Name: Newly add Database' name.

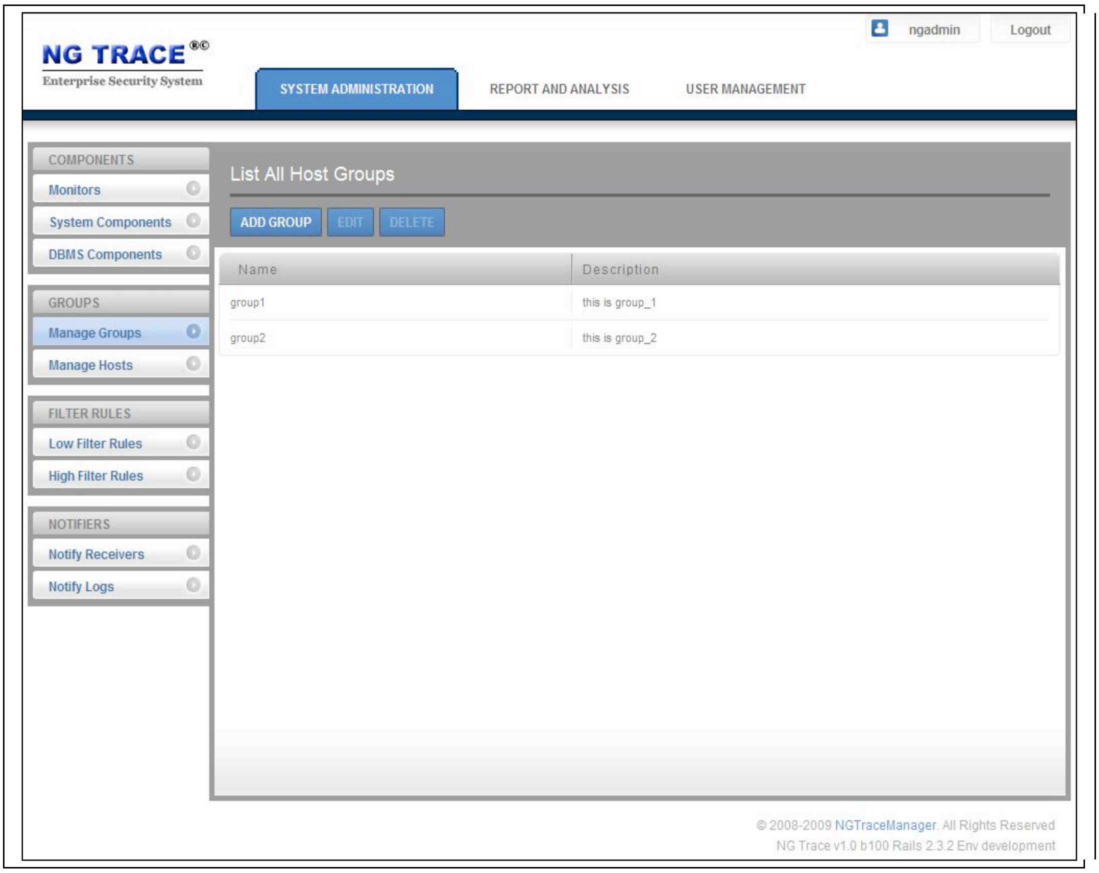
IP Address: IP Address of

- [EDIT]
  Change information of connected Database.
- [DELET]
  Delete selected Database.

8.2.2 GROUPS

- **Manage Group**

Shows list of added Group and add new Group, can edit or delete.



List All Host Groups

Name	Description
group1	this is group_1
group2	this is group_2

© 2008-2009 NGTraceManager. All Rights Reserved
NG Trace v1.0 b100 Rails 2.3.2 Env development

Figure 26. List All Host Groups

Group Name: Group Name to be adding.

Description: Description of Group

- [ADD GROUP]
  Add new Group.

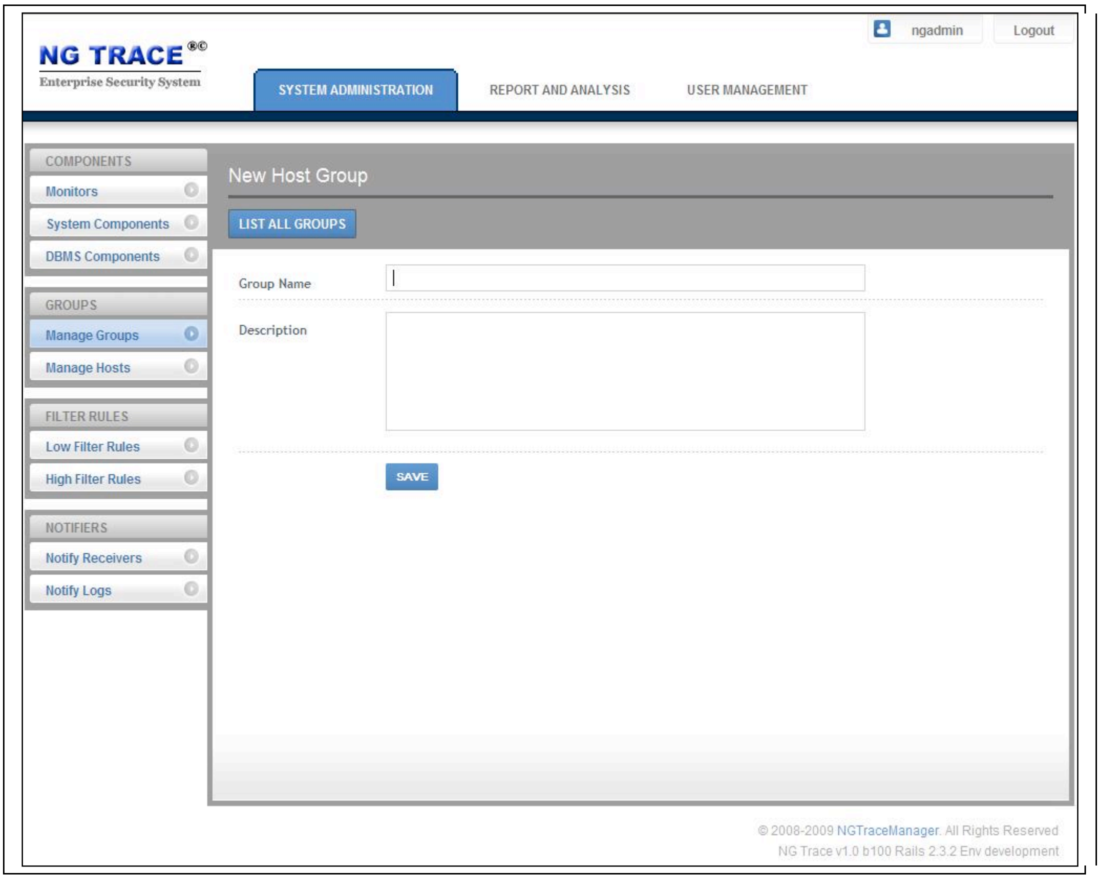


Figure 27. New Host Group

Group Name: Group Name to be adding.

Description: Description of Group

- [EDIT]

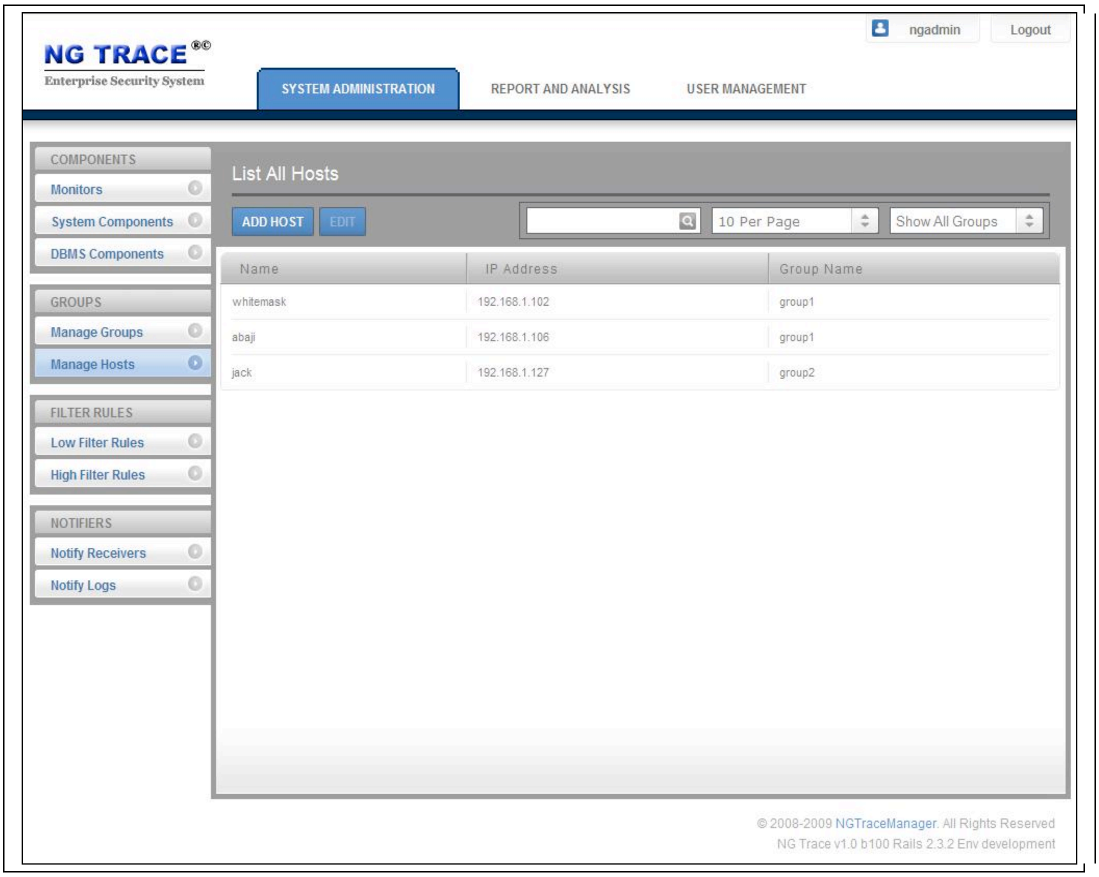
  Change the selected Group's information.

- [DELETE]

  Delete selected Group.

- **Manage Host**

  Show Host List that connect to each Group and information.



Name	IP Address	Group Name
whitemask	192.168.1.102	group1
abaji	192.168.1.106	group1
jack	192.168.1.127	group2

© 2008-2009 NGTraceManager. All Rights Reserved
NG Trace v1.0 b100 Rails 2.3.2 Env development

Figure 28. List All Host

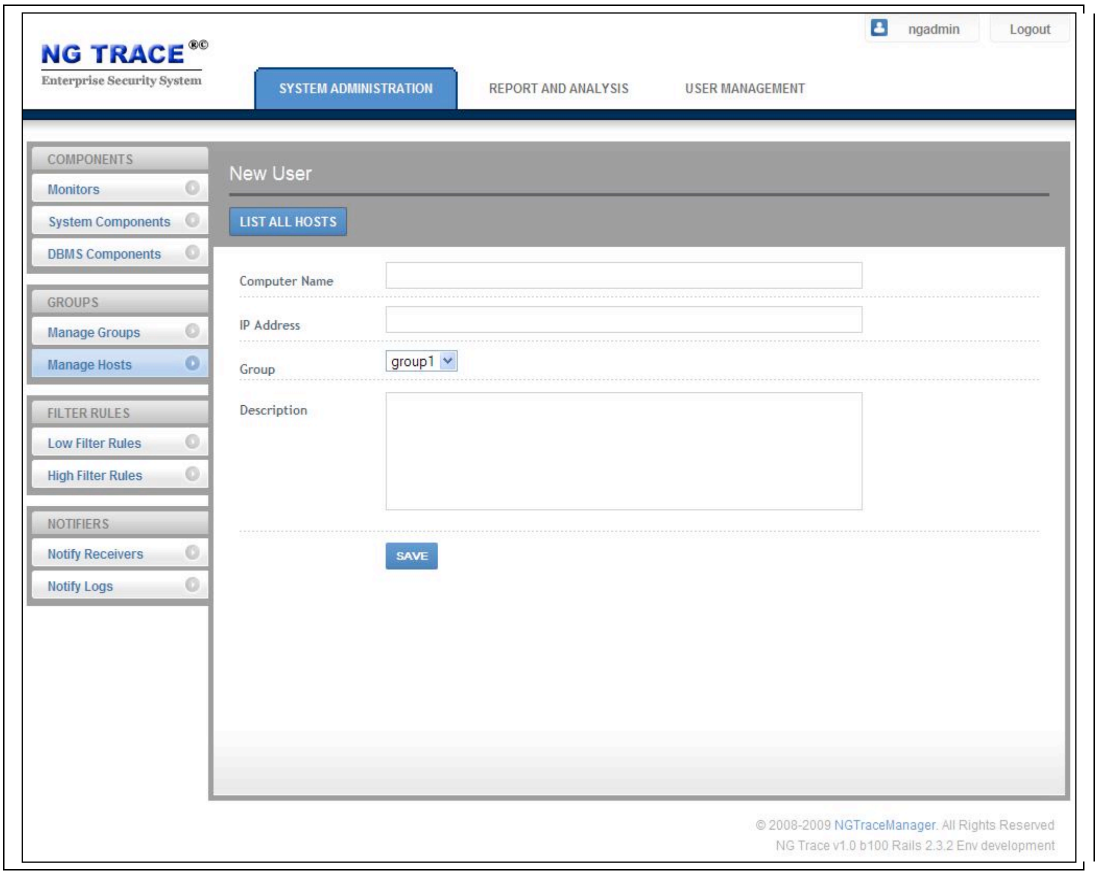
Name: Host name

IP Address: Host IP Address

Group Name: Group's Name of including Host

[ADD HOST]

Add new Host.



Computer Name
IP Address
Group: group1
Description

Figure 29. Add New Host

Computer Name: Computer Name to be adding.

IP Address: Computer IP Address to be adding.

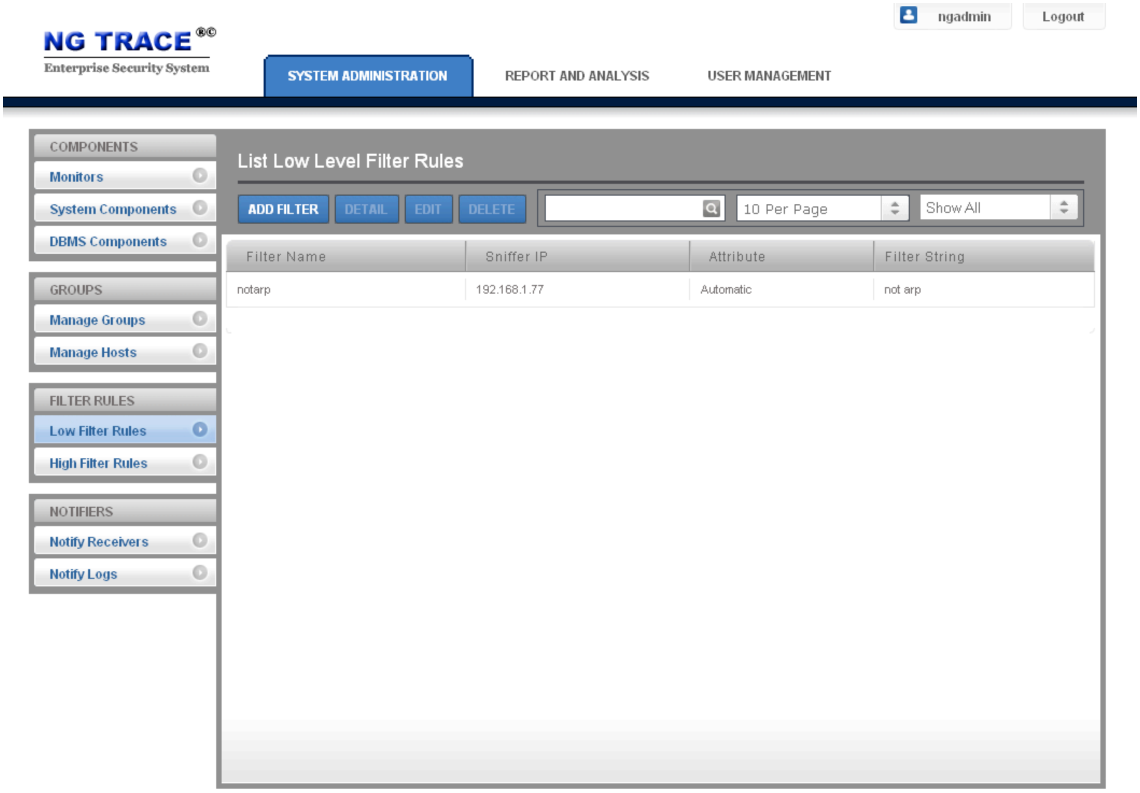
Group: Group to be including.

Description: Host Description.

- [EDIT]
  Modify selected Host’s information like Computer name, IP Address, Group, and Description.

8.2.3 FILTER RULES

- **Low Filter Rules**
  Display all list of Low Filter Rules that connect to Sniffer.



List Low Level Filter Rules

Filter Name	Sniffer IP	Attribute	Filter String
notarp	192.168.1.77	Automatic	not arp

Figure 30. List Low Level Filter Rules

Filter Name: Name of Filter.

Sniffer IP Address: IP Address of Sniffer.

Attribute: Attribute of protocol.

Filter String: Filter Expression.

- [ADD FILTER]

  Add new low filter rule.

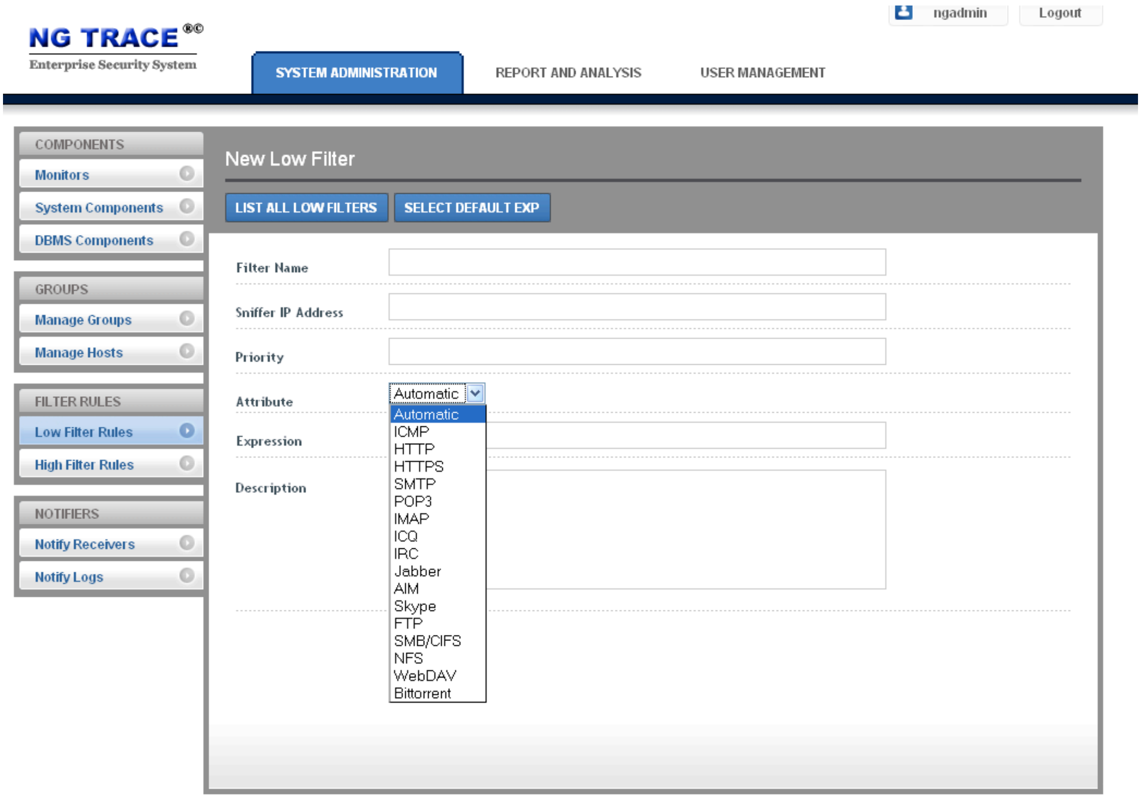


Figure 31. New Log Filter

Filter Name: Name of low level filter to add.

Sniffer IP Address: IP Address of Sniffer.

Priority: Priority of low level filter to add.

Attribute: Attribute of protocol.

Expression: Expression of filter rule.

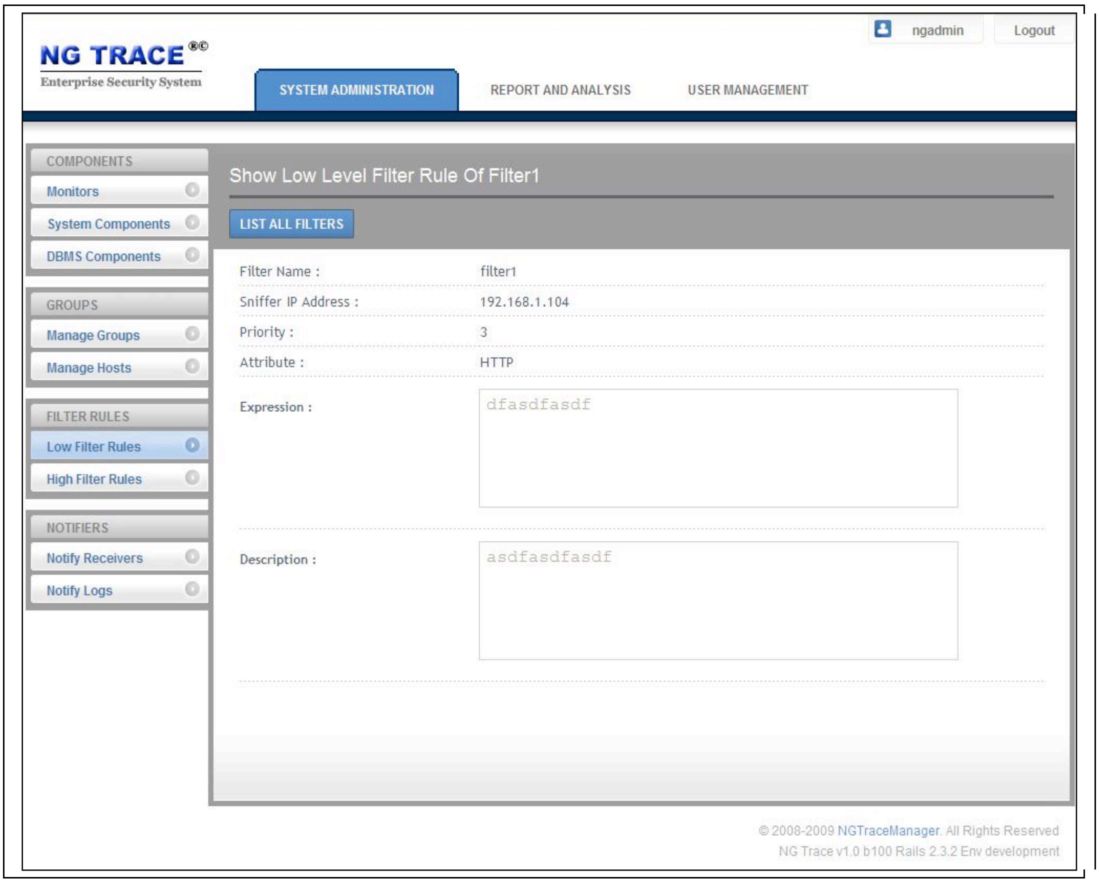
Description: Description of filter rule.

[SELECT DEFAULT EXP]

Frequently used filter expression can be added, edited, deleted and associated with Low Filter Expression.

- [DETAIL]

Show information of selected filter rule.



Show Low Level Filter Rule Of Filter1

Filter Name : filter1

Sniffer IP Address : 192.168.1.104

Priority : 3

Attribute : HTTP

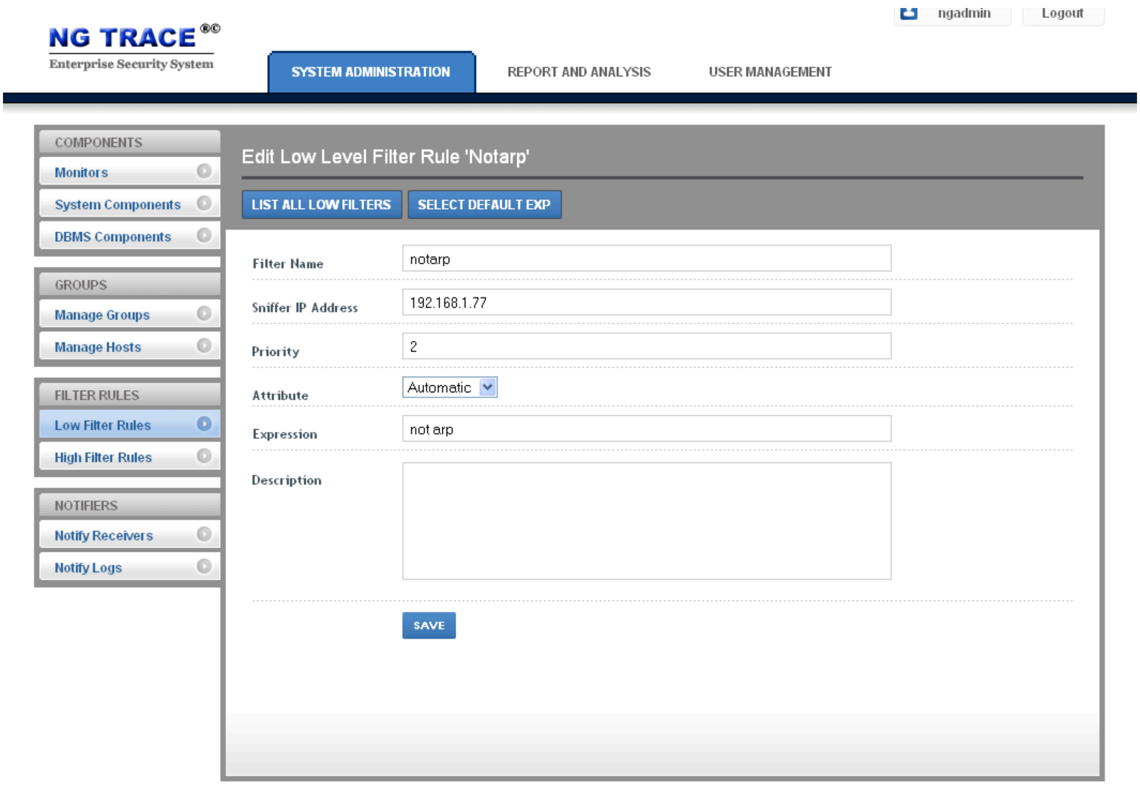
Expression : dfasdfasdf

Description : asdfasdfasdf

© 2008-2009 NGTraceManager. All Rights Reserved
NG Trace v1.0 b100 Rails 2.3.2 Env development

Figure 32. Shows Low Level Filter Rule

- [EDIT]
  Change established information of selected filter rule.



Edit Low Level Filter Rule 'Notarp'

Filter Name	notarp
Sniffer IP Address	192.168.1.77
Priority	2
Attribute	Automatic
Expression	not arp
Description	

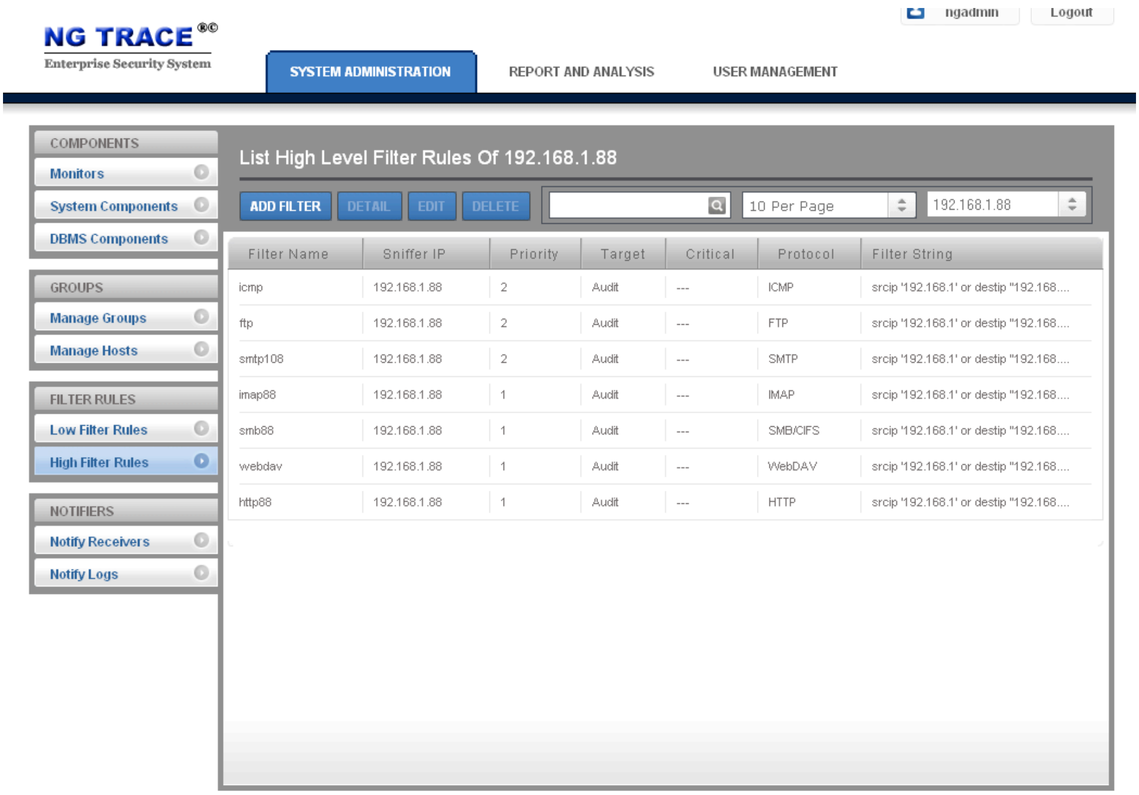
Figure 33. Edit Low Level Filter Rule

[SELECT DEFAULT EXP]

Frequently used filter expression can be added, edited, deleted and associated with Low Filter Expression.

- [DELETE]
  Delete selected Low Level Filter Rule.

- **High Filter Rules**
  Show all list of High Filter Rules that connect to Sniffer.



List High Level Filter Rules Of 192.168.1.88

Filter Name	Sniffer IP	Priority	Target	Critical	Protocol	Filter String
icmp	192.168.1.88	2	Audit	---	ICMP	srcip '192.168.1' or destip "192.168....
ftp	192.168.1.88	2	Audit	---	FTP	srcip '192.168.1' or destip "192.168....
smtp108	192.168.1.88	2	Audit	---	SMTP	srcip '192.168.1' or destip "192.168....
imap88	192.168.1.88	1	Audit	---	IMAP	srcip '192.168.1' or destip "192.168....
smb88	192.168.1.88	1	Audit	---	SMB/CIFS	srcip '192.168.1' or destip "192.168....
webdav	192.168.1.88	1	Audit	---	WebDAV	srcip '192.168.1' or destip "192.168....
http88	192.168.1.88	1	Audit	---	HTTP	srcip '192.168.1' or destip "192.168....

Figure 34. List High Level Filter Rules

Filter Name: Name of filter.

Sniffer IP Address: IP Address of Sniffer.

Priority: Priority of filter.

Target: Established filter Target.

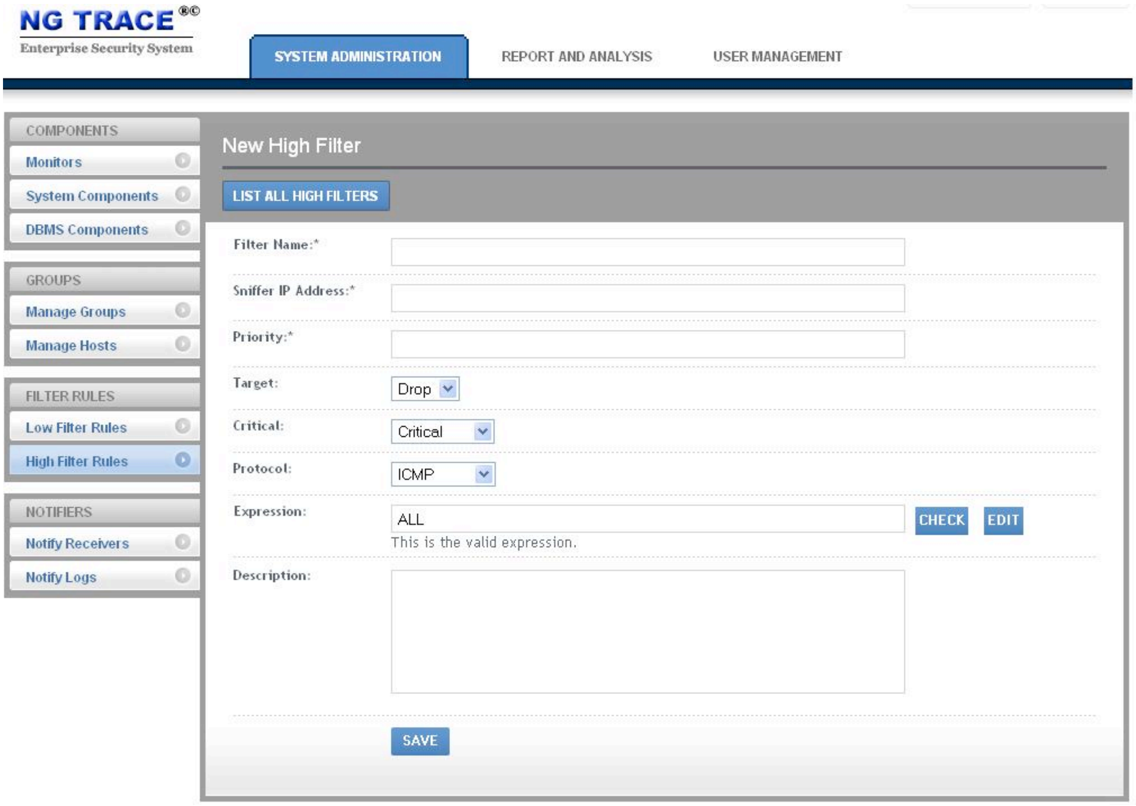
Critical: Established filter Critical.

Protocol: Established protocol.

Filter String: Established filter expression.

- [ADD FILTER]

  Add new high level filter rule.



Filter Name:*

Sniffer IP Address:*

Priority:*

Target: Drop

Critical: Critical

Protocol: ICMP

Expression: ALL

This is the valid expression.

Description:

Figure 35. New High Filter

Filter Name: Name of filter to add.

Sniffer IP Address: IP Address of Sniffer.

Priority: Priority of filter to add.

Target: Target of filter to add.

Critical: Critical of filter to add.

Protocol: Protocol of filter to add.

Expression: Expression of filter to add.

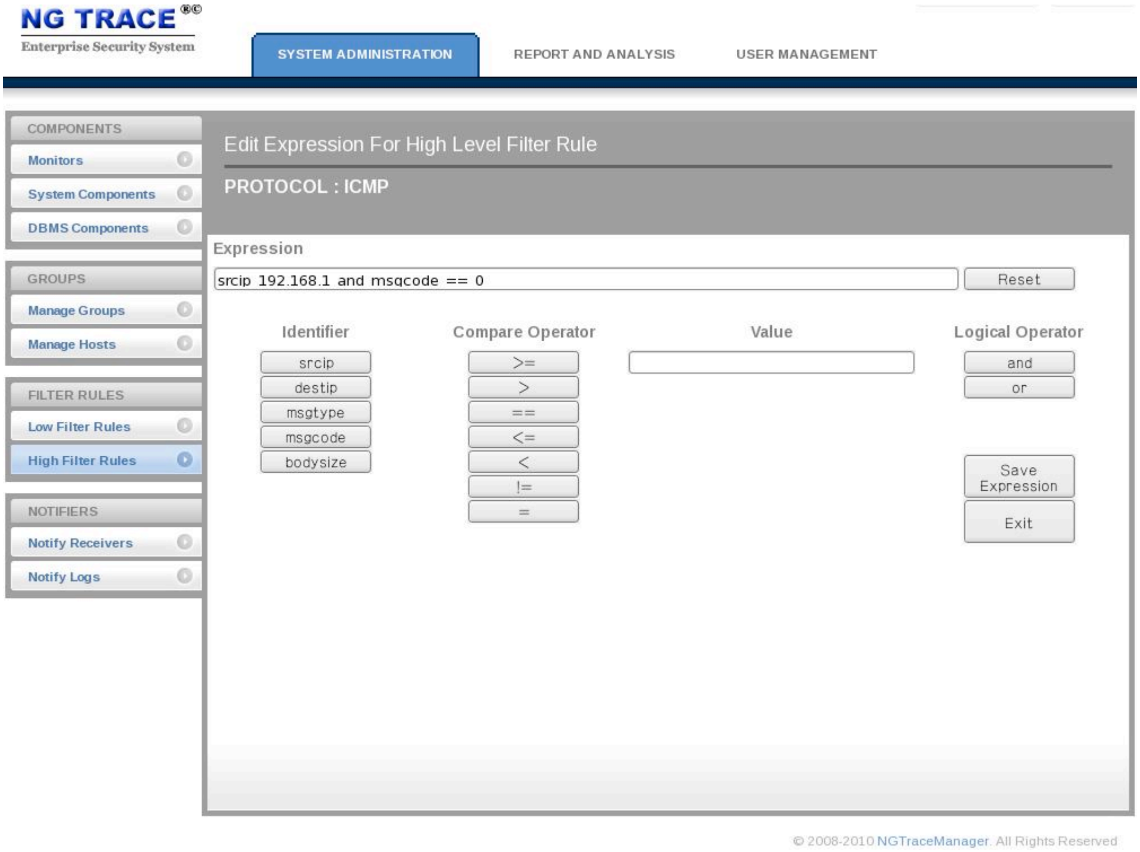
Description: Description of filter to add.

“CHECK” button: Check whether expression is valid or not.

“EDIT” button: Open expression edit window.

“SAVE” button: Save filter. If expression is evaluated to be valid by clicking “CHECK” button, it is displayed in the page.

- [EDIT EXPRESSION]
  Edit new expression.



Edit Expression For High Level Filter Rule

PROTOCOL : ICMP

Expression: srcip 192.168.1 and msqcode == 0

Identifier: name of item of each protocol

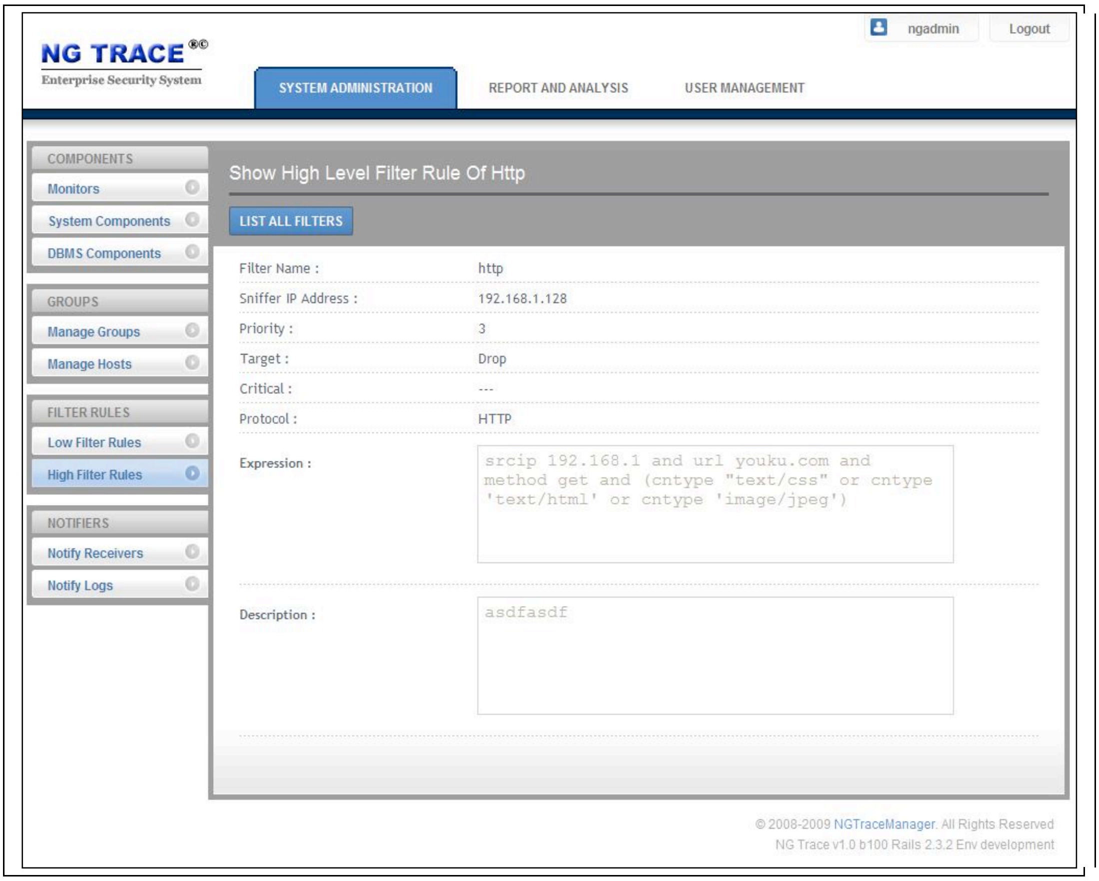
Compare Operator: compare operator between **Identifier** and **value**

Value: String or number value of identifier

Logical Operator: Logical operator combining two expressions

- [DETAIL]

  Show information of selected filter rule.

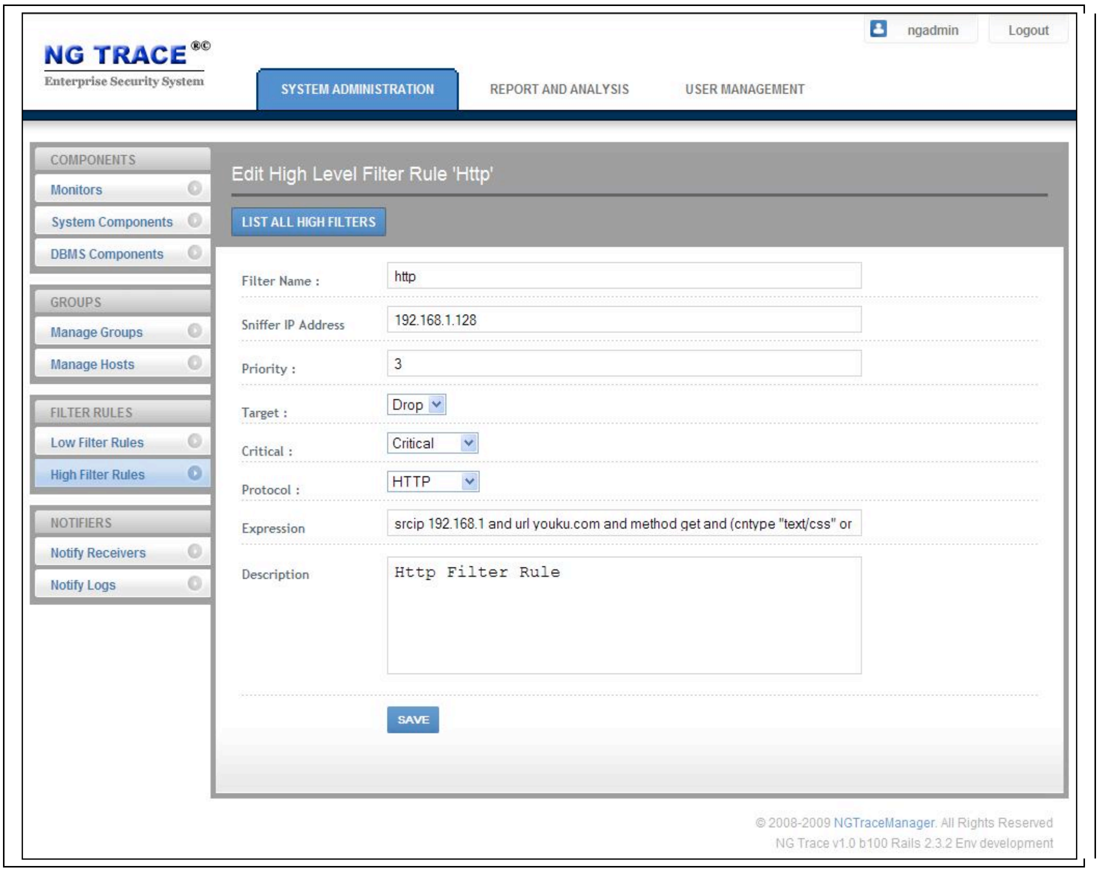


Show High Level Filter Rule Of Http

Filter Name :	http
Sniffer IP Address :	192.168.1.128
Priority :	3
Target :	Drop
Critical :	---
Protocol :	HTTP
Expression :	srcip 192.168.1 and url youku.com and method get and (cntype "text/css" or cntype 'text/html' or cntype 'image/jpeg')
Description :	asdfasdf

Figure 36. Show High Level Filter Rule

- [EDIT]
  Change established information of selected filter rule.



Edit High Level Filter Rule 'Http'

Filter Name : http

Sniffer IP Address 192.168.1.128

Priority : 3

Target : Drop

Critical : Critical

Protocol : HTTP

Expression srcip 192.168.1 and url youku.com and method get and (cntype "text/css" or

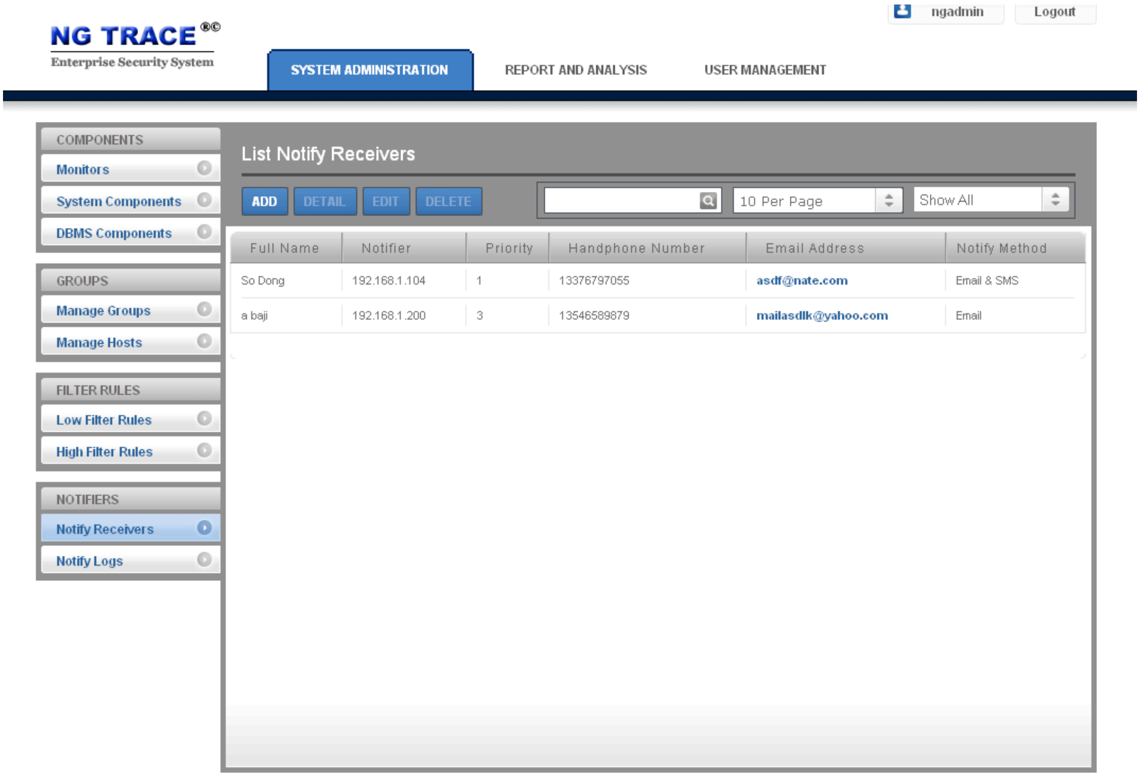
Description Http Filter Rule

Figure 37. Edit High level Filter Rule

- [DELETE]
  Delete selected High Level Filter Rule.

8.2.4 NOTIFIERS

- **Notify Receivers**
  We can see list of notify target and the information about the event of critical security violation.

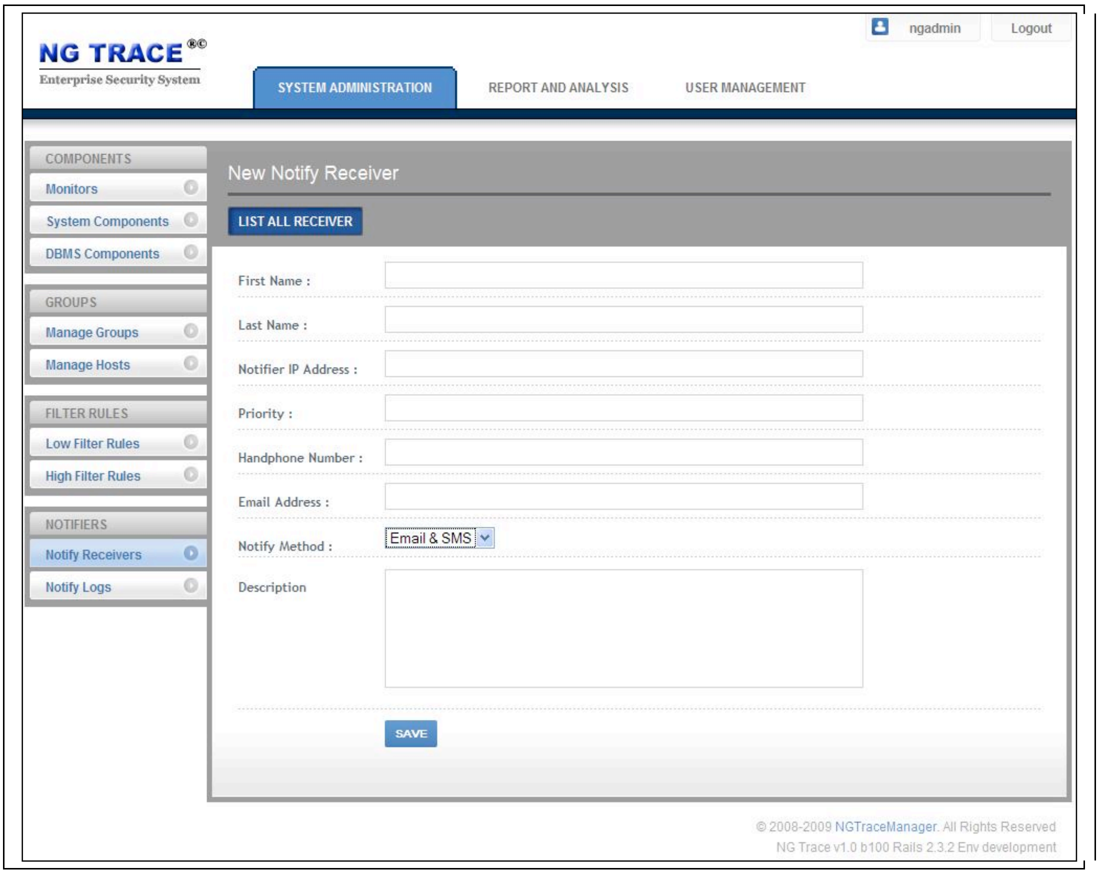


Full Name	Notifier	Priority	Handphone Number	Email Address	Notify Method
So Dong	192.168.1.104	1	13376797055	asdf@nate.com	Email & SMS
a baji	192.168.1.200	3	13546589879	mailasdlk@yahoo.com	Email

Figure 38. List Notify Receiver

Full Name: Name of Receiver.
Notifier: IP Address of Notifier.
Priority: Priority of Receiver.
Hand phone Number: Hand phone number of Receiver.
Email Address: Email address of Receiver.
Notify Method: Notify method.

- [ADD]
  Add new Receiver.

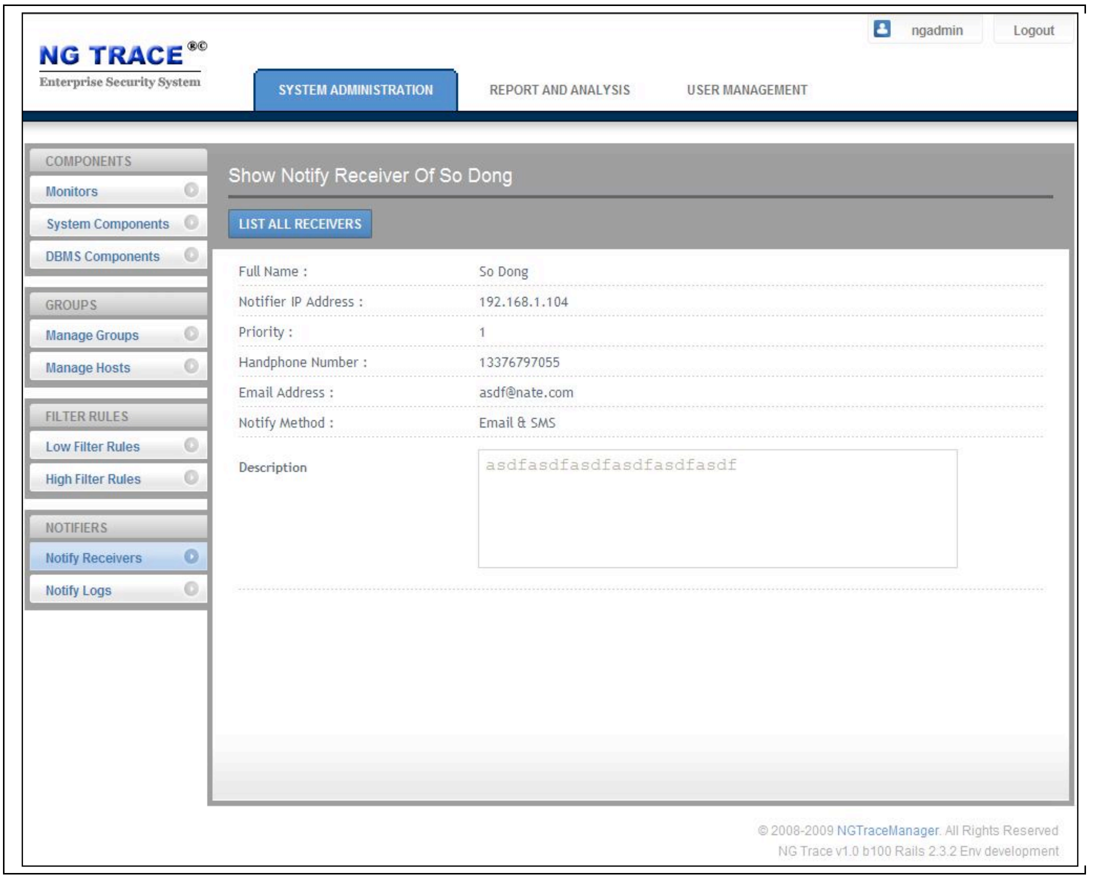


Notify Method : Email & SMS

Figure 39. New Notify Receiver

First Name: Receiver’s first name to add.
Last Name: Receiver’s Last name to add.
Notifier IP Address: IP Address of Notifier to add.
Priority: Receiver’s priority to add.
Handphone Number: Receiver’s Handphone number to add.
Email Address: Receiver’s email address to add.
Notify method: Notify method to be sending.
Description: Receiver’s description to add.

- [DETAIL]
  View selected Receiver’s information such as Full Name, Notifier IP Address, Priority, Handphone number, Email Address, Notify method, and Description.

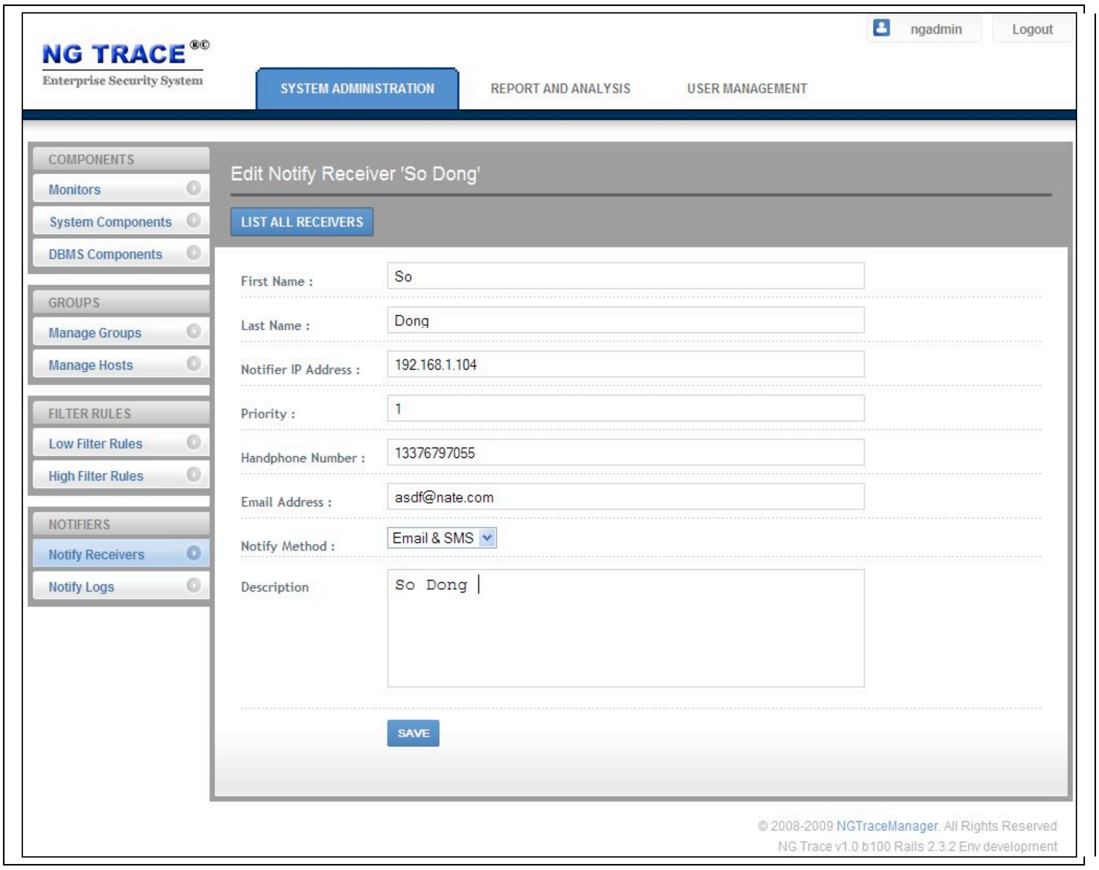


Show Notify Receiver Of So Dong

Full Name :	So Dong
Notifier IP Address :	192.168.1.104
Priority :	1
Handphone Number :	13376797055
Email Address :	asdf@nate.com
Notify Method :	Email & SMS
Description	asdfasdfasdfasdfasdfasdf

Figure 40. Show Notify Receiver

- [EDIT]
  Change selected Receiver’s information like Full Name, Notifier IP Address, Last Name, Priority, Handphone Number, and Email Address, Notify Method, Description.



Edit Notify Receiver 'So Dong'

First Name : So

Last Name : Dong

Notifier IP Address : 192.168.1.104

Priority : 1

Handphone Number : 13376797055

Email Address : asdf@nate.com

Notify Method : Email & SMS

Description So Dong

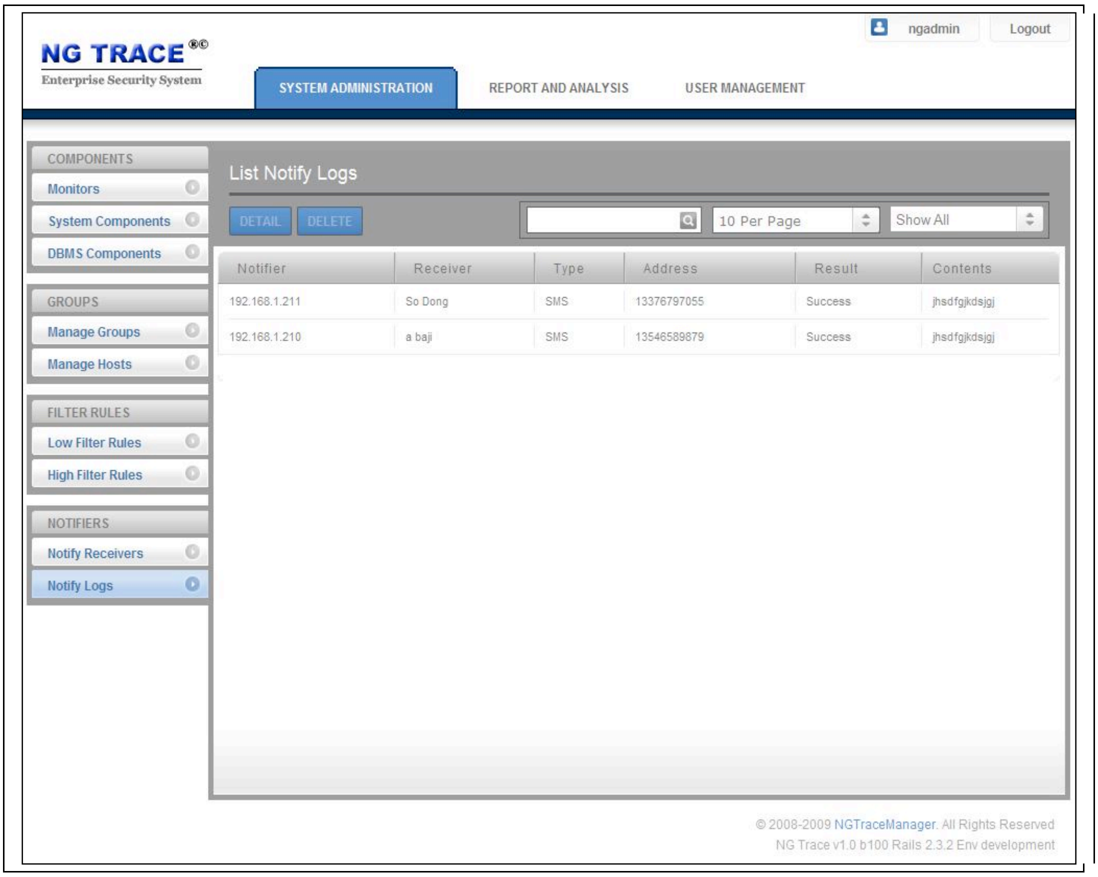
© 2008-2009 NGTraceManager. All Rights Reserved
NG Trace v1.0 b100 Rails 2.3.2 Env development

Figure 41. Edit Notify Receiver

- [DELETE]
  Delete selected Receiver.

- **Notify Logs**

  We can see the log list of notifier activities sending E-mail or SMS message to Notify target.



Notifier	Receiver	Type	Address	Result	Contents
192.168.1.211	So Dong	SMS	13376797055	Success	jhsdfgjkdsjgj
192.168.1.210	a baji	SMS	13546589879	Success	jhsdfgjkdsjgj

Figure 42. List Notify Logs

Notifier: IP Address of Notifier.

Receiver: Name of Receiver.

Type: Delivery type.

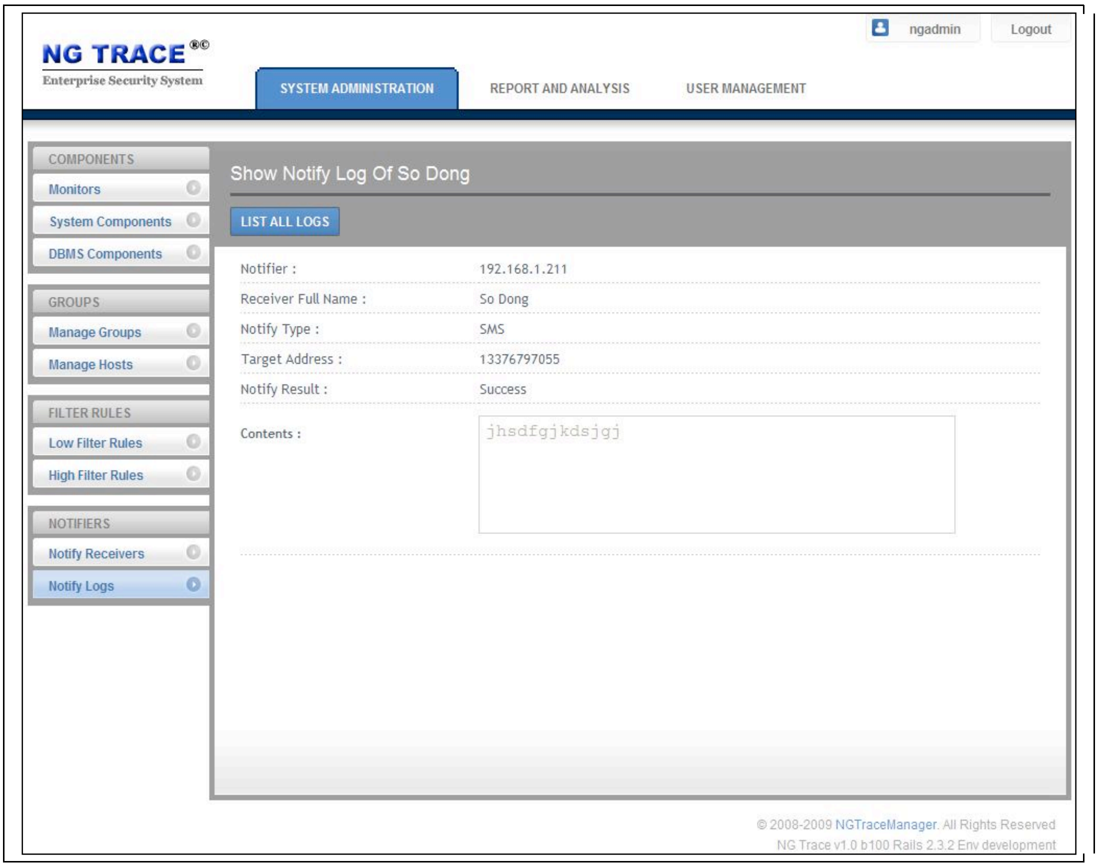
Address: Address of Receiver.(If Type is SMS, Address is Handphone number and if Type is E-mail, Address is E-mail address.)

Result: Send result.

Contents: Static contents to send.

- [DETAIL]

  View selected log’s information like Notifier, Receiver Full Name, Target Address, Notify Result, and Contents.



Show Notify Log Of So Dong

Notifier :	192.168.1.211
Receiver Full Name :	So Dong
Notify Type :	SMS
Target Address :	13376797055
Notify Result :	Success
Contents :	jhsdfgjkdsjgj

Figure 43. Show Notify Log

- [DELETE]
  Delete selected Log.