HOWTO INSTALL & CONFIGURE SNORT ON DEBIAN/UBUNTU

====================================
== REQUIREMENTS ==
====================================

mysql-common - MySQL database common files
mysql-client - MySQL database client (metapackage depending on the latest version)
mysql-server - MySQL database server (metapackage depending on the latest version)
php5-dev - Files for PHP5 module development
php5-gd - GD module for php5
php5-ldap - LDAP module for php5
php5-mysql - MySQL module for php5
php-pear - PEAR - PHP Extension and Application Repository
libpcap-dev - development library for libpcap (transitional package)
libpcap0.8 - system interface for user-level packet capture
libpcap0.8-dev - development library and header files for libpcap0.8
libpcre3 - Perl 5 Compatible Regular Expression Library - runtime files
libpcre3-dev - Perl 5 Compatible Regular Expression Library - development files
expect - A program that can automate interactive applications
gobjc - The GNU Objective-C compiler
libnet0 - library for the construction and handling of network packets (obsolete)
libnet0-dev - Development files for libnet0 (obsolete)
bison - A parser generator that is compatible with YACC
libmysql++-dev - MySQL C++ library bindings (development)
libapache2-mod-php5 - server-side, HTML-embedded scripting language (Apache 2 module)
php5-cgi - server-side, HTML-embedded scripting language (CGI binary)

Installing MySQL Server:

```
[root@snortbox ~]# apt-get install mysql-common mysql-client mysql-server
```

Installing requirements:

```
[root@snortbox ~]# apt-get install php5-dev php5-gd php5-ldap php5-mysql php-pear libnet1
libnet1-dev libpcap-dev libpcap0.8 libpcap0.8-dev libpcre3 expect gobjc libpcre3-dev flex libnet0
libnet0-dev bison libmysql++-dev libapache2-mod-php5 php5-cgi
```

== SECURING MYSQL ==

After we install the MySQL server, there is a builtin script that comes with it which we can use to secure our MySQL server. To do that do the following:

```
[root@snortbox ~]# mysql_secure_installation

NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MySQL
SERVERS IN PRODUCTION USE!  PLEASE READ EACH STEP CAREFULLY!

In order to log into MySQL to secure it, we'll need the current password for the root user.  If you've just
installed MySQL, and you haven't set the root password yet, the password will be blank, so you should
just press enter here.

Enter current password for root (enter for none):
OK, successfully used password, moving on...

Setting the root password ensures that nobody can log into the MySQL root user without the proper
authorisation.

You already have a root password set, so you can safely answer 'n'.
Change the root password? [Y/n] n
 ... skipping.

By default, a MySQL installation has an anonymous user, allowing anyone to log into MySQL without
having to have a user account created for them.  This is intended only for testing, and to make the
installation go a bit smoother.  You should remove them before moving into a production environment.

Remove anonymous users? [Y/n] y
 ... Success!

Normally, root should only be allowed to connect from 'localhost'.  This ensures that someone cannot
guess at the root password from the network.

Disallow root login remotely? [Y/n] y
 ... Success!

By default, MySQL comes with a database named 'test' that anyone can access.  This is also intended
only for testing, and should be removed before moving into a production environment.

Remove test database and access to it? [Y/n] y
 - Dropping test database...
ERROR 1008 (HY000) at line 1: Can't drop database 'test'; database doesn't exist
... Failed!  Not critical, keep moving...
 - Removing privileges on test database...
 ... Success!

Reloading the privilege tables will ensure that all changes made so far will take effect immediately.

Reload privilege tables now? [Y/n] y
 ... Success!

Cleaning up...

All done!  If you've completed all of the above steps, your MySQL installation should now be secure.

Thanks for using MySQL!
```

Lets make sure MySQL is now running:

```
[root@snortbox ~]# netstat -ntlp | grep mysql
tcp        0      0 127.0.0.1:3306          0.0.0.0:*                   LISTEN      16000/mysqld
```

== CONFIGURE & INSTALL SNORT ==

Now everything is ready, lets configure and install Snort. First you need to download the latest version of Snort and Snort-rules from Snort.org. Now extract them and enter the extracted snort directory to start the configuration process:

```
[root@snortbox snort-2.8.5]# ./configure --enable-sourcefire --enable-targetbased --enable-flexresp --with-mysql
```

After the configuration is complete we are now ready to Compile and install Snort:

```
[root@snortbox snort-2.8.5]# make
[root@snortbox snort-2.8.5]# make install
```

Now lets make some configurations for SNORT to work. First we create the configuration directory under /etc to be used for snorts configurations:

```
[root@snortbox snort-2.8.5]# mkdir /etc/snort
```

Now we create a directory for snort to log into:

```
[root@snortbox snort-2.8.5]# mkdir /var/log/snort
```

Now we extract and copy the Snort rules to our snort configuration directory we just created above:

```
[root@snortbox snort-2.8.5]# tar xvfz snortrules-snapshot-CURRENT.tar.gz -C /etc/snort
```

Now we start coping some of the configuration files needed by snort:

```
[root@snortbox snort-2.8.5]# cp -r preproc_rules /etc/snort
[root@snortbox snort-2.8.5]# cp etc/*.conf* /etc/snort
[root@snortbox snort-2.8.5]# cp etc/*.map /etc/snort
```

Lets create a symlink to

/usr/local/bin/snort and put it in /usr/sbin:

```
[root@snortbox snort-2.8.5]# ln -s /usr/local/bin/snort /usr/sbin/snort
```

We need a user and group for snort to operate with:

```
[root@snortbox snort-2.8.5]# groupadd snort
[root@snortbox snort-2.8.5]# useradd -g snort snort
```

Let us change the owner of the snort log directory:

```
[root@snortbox snort-2.8.5]# chown snort:snort /var/log/snort
```

Now edit snort's defualt configuration file:

```
[root@snortbox snort-2.8.5]# vim /etc/snort/snort.conf
```

Now search for the line “**RULE_PATH ../rules**” and replace it with:

```
var RULE_PATH /etc/snort/rules
```

Now search for “**PREPROC_RULE_PATH ../preproc_rules**”, and replace it with:
PREPROC_RULE_PATH /etc/snort/preproc_rules

Now search for “**output database: log, mysql, user=root password=test dbname=db host=localhost** ” and replace it with:
output database: log, mysql, user=USERNAME password=PASSWORD dbname=DATABASENAME host=HOST

Legend:
USERNAME = Your Snort's DB username (That shall be used later in configuration).
PASSWORD = Your Snort's DB username password (That shall be used later in configuration).
DATABASENAME = the name of the Snort database (That shall be used later in configuration).
HOST = the hostname of the machine running Snort (Usually this is localhost, specially if you shall only have one sensor).

Now goto the end of the file and search for:
# include $PREPROC_RULE_PATH/preprocessor.rules
# include $PREPROC_RULE_PATH/decoder.rules

And comment them out.

Now close and exit the **snort.conf** file using “**:x**”.

=======================================

==   SETUP THE SNORT DATABASE   ==

=======================================

Configuration is ready we need to prepare the MySQL database for Snort. To do that we need to create a database and user for Snort to use:

Legend:

DBNAME = The database name that you shall use for Snort.
SNORTDBUSER = The snort db user that you shall use for Snort.
YOURPASSWORD = The password that you shall use for the Snort db user.

```
[root@snortbox ~]# mysql -p
Enter password:
Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 40
Server version: 5.0.51a-24+lenny2 (Debian)

Type 'help;' or '\h' for help. Type '\c' to clear the buffer.

mysql> CREATE DATABASE DBNAME;
Query OK, 1 row affected (0.00 sec)

mysql> GRANT CREATE, INSERT, SELECT, DELETE, UPDATE ON DBNAME.* TO
SNORTDBUSER@LOCALHOST;
Query OK, 0 rows affected (0.00 sec)

mysql> SET PASSWORD FOR
SNORTDBUSER@LOCALHOST=PASSWORD('YOURPASSWORD');
Query OK, 0 rows affected (0.00 sec)

mysql> FLUSH PRIVILEGES;
Query OK, 0 rows affected (0.00 sec)

mysql> exit
Bye
```

Now lets setup the database schema for Snort:

```
[root@snortbox schemas]# cd schemas/
```

Now lets import the schema into the database we created for Snort:

```
[root@snortbox schemas]# mysql -p -u SNORTDBUSER DBNAME < create_mysql
Enter password:
```

You shall be asked to enter the password you set for the **SNORTDBUSER** that you setup in the previous lines above.

Now lets make sure that everything went well and our database is ready:

```
[root@snortbox schemas]# mysql -p
Enter password:

Welcome to the MySQL monitor.  Commands end with ; or \g.
Your MySQL connection id is 43
Server version: 5.0.51a-24+lenny2 (Debian)

mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| DBNAME             |
| mysql              |
+--------------------+
3 rows in set (0.00 sec)

mysql> use DBNAME;
Reading table information for completion of table and column names You can turn off this feature to
get a quicker startup with -A
Database changed

mysql> show tables;
+---------------------------+
| Tables_in_DBNAME          |
+---------------------------+
| data                      |
| detail                    |
| encoding                  |
| event                     |
| icmphdr                   |
| iphdr                     |
| opt                       |
| reference                 |
| reference_system          |
| schema                    |
| sensor                    |
| sig_class                 |
| sig_reference             |
| signature                 |
| tcphdr                    |
| udphdr                    |
+---------------------------+
16 rows in set (0.00 sec)

mysql>
exit
```

== TESTING SNORT ==

Now everything is done, lets make a test:

```
[root@snortbox schemas]# snort -c /etc/snort/snort.conf
```

Your output shall be something like this:

```
[ Port and Service Based Pattern Matching Memory ]
+-[AC-BNFA Search Info Summary]------------------------------
| Instances        : 284
| Patterns         : 23051
| Pattern Chars    : 156930
| Num States       : 90782
| Num Match States : 12254
| Memory           :   3.87Mbytes
|   Patterns       :   1.03M
|   Match Lists    :   1.43M
|   Transitions    :   1.30M
+-------------------------------------------------

        --== Initialization Complete ==--
   ,,_     -*> Snort! <*-
  o"  )~   Version 2.8.5 (Build 106)
   ""    By Martin Roesch & The Snort Team: http://www.snort.org/snort/snort-team
           Copyright (C) 1998-2009 Sourcefire, Inc., et al.
           Using PCRE version: 7.6 2008-01-28

           Rules Engine: SF_SNORT_DETECTION_ENGINE  Version 1.11  <Build 17>
           Preprocessor Object: SF_DCERPC  Version 1.1  <Build 5>
           Preprocessor Object: SF_FTPTELNET  Version 1.2  <Build 12>
           Preprocessor Object: SF_SSLPP  Version 1.1  <Build 3>
           Preprocessor Object: SF_SSH  Version 1.1  <Build 2>
           Preprocessor Object: SF_SMTP  Version 1.1  <Build 8>
           Preprocessor Object: SF_DCERPC2  Version 1.0  <Build 2>
           Preprocessor Object: SF_DNS  Version 1.1  <Build 3>
Not Using PCAP_FRAMES
```

Press ctrl+c to get back to the shell and stop snort.

As we shall be running snort using a startup script, we need to change the alert file created by the Snort process so that the snort has access to it, we need to change its permissions to be:

```
[root@snortbox schemas]# chown snort:snort /var/log/snort/alert
[root@snortbox schemas]# chmod 600 /var/log/snort/alert
```

==

== SETUP THE GRAPHICAL INTERFACE FOR SNORT ==

==

Now lets setup the graphical interface “**Basic Analysis and Security Engine (BASE)**” that we shall be using with Snort.

```
[root@snortbox /var/www]# cd /var/www
```

First we need to download **BASE** and **ADOdb** from **SourceForge**, then extract them in the web root directory which is **/var/www** on **debian** and then give the apache user access to the **BASE** directory files:

```
[root@snortbox /var/www]# chown www-data base-1.4.4
```

Now we need to tune the error reporting level of PHP, so lets edit the php.ini file:

```
[root@snortbox /var/www]# vim /etc/php5/cli/php.ini
```

Search for:

```
error_reporting = E_ALL & ~E_NOTICE
```

And uncomment it. Then Comment out the line below which can be found a several lines below:

```
error_reporting = E_ALL
```

Save your changes and exit “**:x**”.

Now restart Apache:

```
[root@snortbox /var/www]# /etc/init.d/apache2 restart
```

Before we continue to the configuration of BASE we need to make sure of the availability of the following PHP extensions:
Mail, Mail_Mime, Log, Image_Color, Image_Canvas, Numbers_Roman, Numbers_Words

You can make sure of that using the following command:

```
[root@snortbox /var/www]# find / -name "Mail.php"
```

If you don't get any answers this means that the Mail PHP Extension is not installed. So lets install it and install the others using **PEAR** “PHP Extension and Application Repository”. If PEAR is not installed then you can install it like this:

```
[root@snortbox /var/www]# apt-get install php-pear
```

If you already have PEAR, its a good idea to update and upgrade it:

```
[root@snortbox /var/www]# pear channel-update pear.php.net
[root@snortbox /var/www]# pear upgrade PEAR
```

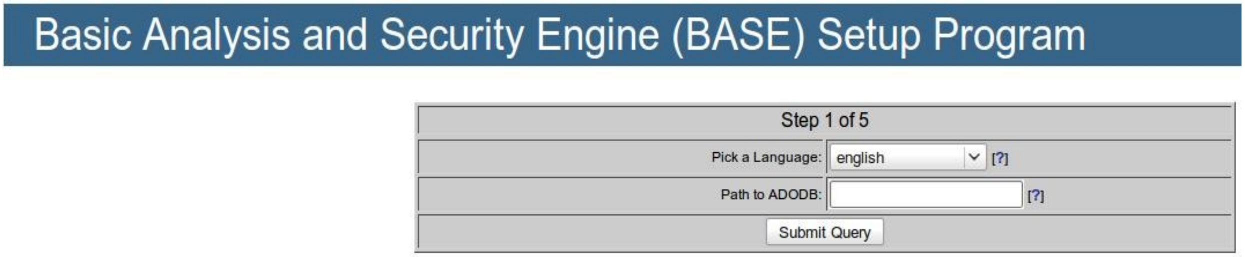
Now lets use PEAR to install the missing php extensions needed by BASE:

```
[root@snortbox /var/www]#  pear install Mail Mail_Mime Image_Color Log Numbers_Roman
Numbers_Words Image_Canvas
```

After the installation of the php extensions is finished lets move on to configure BASE. Goto your web browser and open the following link: http://localhost/base-1.4.4/

Note:
If you have changed the name of the BASE directory, then don't forget to browse that name and not the directory base-1.4.4.

Now you should have reached the BASE setup page, which looks like this:



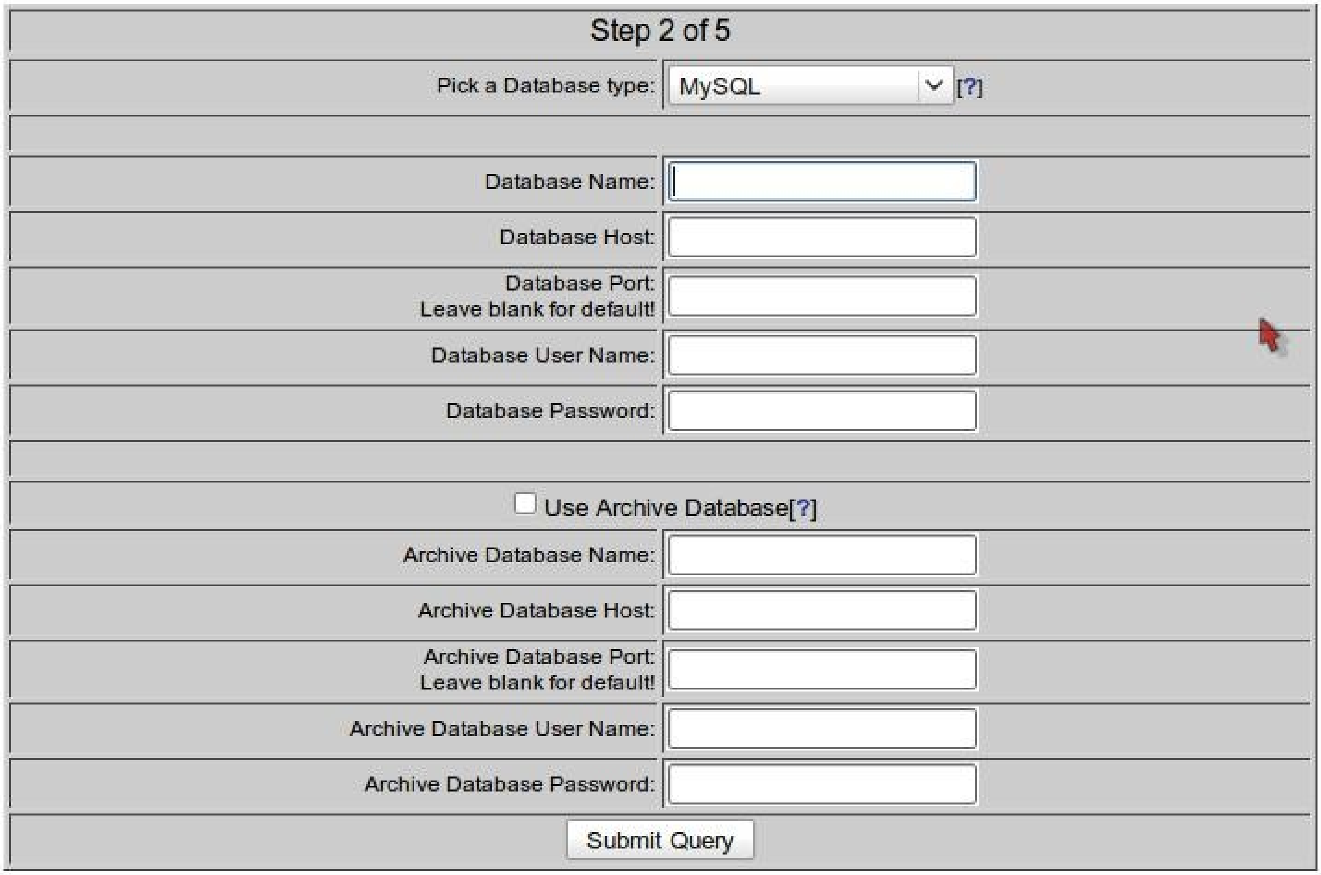
Basic Analysis and Security Engine (BASE) Setup Program

Step 1 of 5

Pick a Language: english

Path to ADODB:

Enter the path of the ADOdb, which is: ***/var/www/adodb5.*** And then press on the ***Submit Query*** button. You shall now be taken to the next screen of BASE setup, which is similar to:



Step 2 of 5

Pick a Database type: MySQL

Database Name:

Database Host:

Database Port:
Leave blank for default!

Database User Name:

Database Password:

□ Use Archive Database[?]

Archive Database Name:

Archive Database Host:

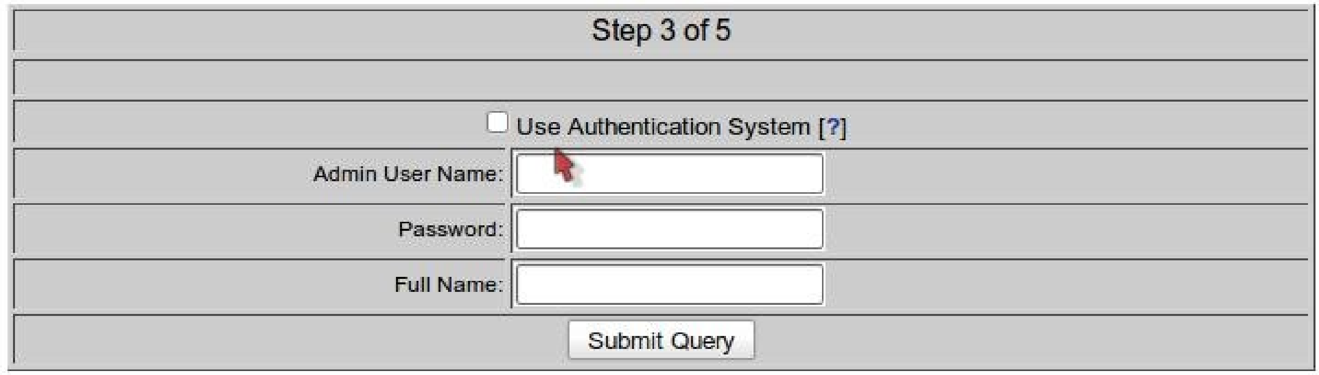
Archive Database Port:
Leave blank for default!

Archive Database User Name:

Archive Database Password:

Insert your database name, database hostname, database username, and the username's password that we created earlier for Snort to use. **Note:** you can leave the database port field empty if you didn't change the default port "**3306**". Finally press the ***Submit Query*** button.

Now you should have reached the third BASE setup page:



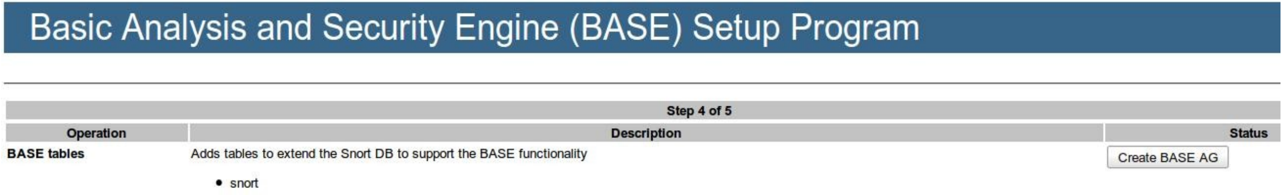
Step 3 of 5

□ Use Authentication System [?]

Admin User Name:	
Password:	
Full Name:	

Here we can define a username and password for it to be used with accessing BASE. Choose a username, password, and add the username's full name and press the ***Submit Query*** button.

Now you should have reached the fourth BASE setup page:

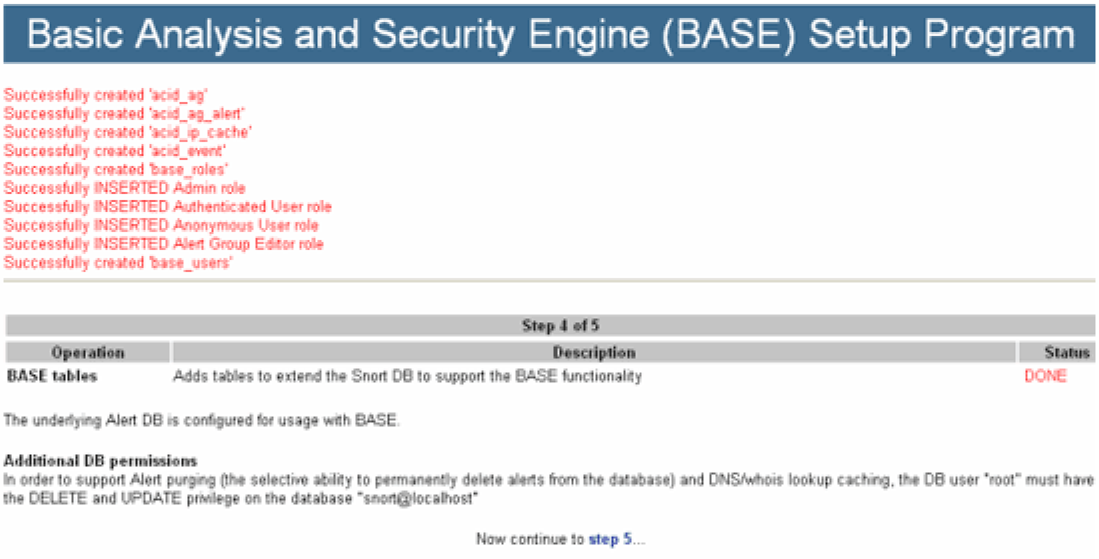


Basic Analysis and Security Engine (BASE) Setup Program

Step 4 of 5

Operation	Description	Status
BASE tables	Adds tables to extend the Snort DB to support the BASE functionality • snort	Create BASE AG

Go ahead and press the ***Create BASE AG*** button. After doing that we shall get the results page which looks like:



Basic Analysis and Security Engine (BASE) Setup Program

```
Successfully created 'acid_ag'
Successfully created 'acid_ag_alert'
Successfully created 'acid_ip_cache'
Successfully created 'acid_event'
Successfully created 'base_roles'
Successfully INSERTED Admin role
Successfully INSERTED Authenticated User role
Successfully INSERTED Anonymous User role
Successfully INSERTED Alert Group Editor role
Successfully created 'base_users'
```

Step 4 of 5

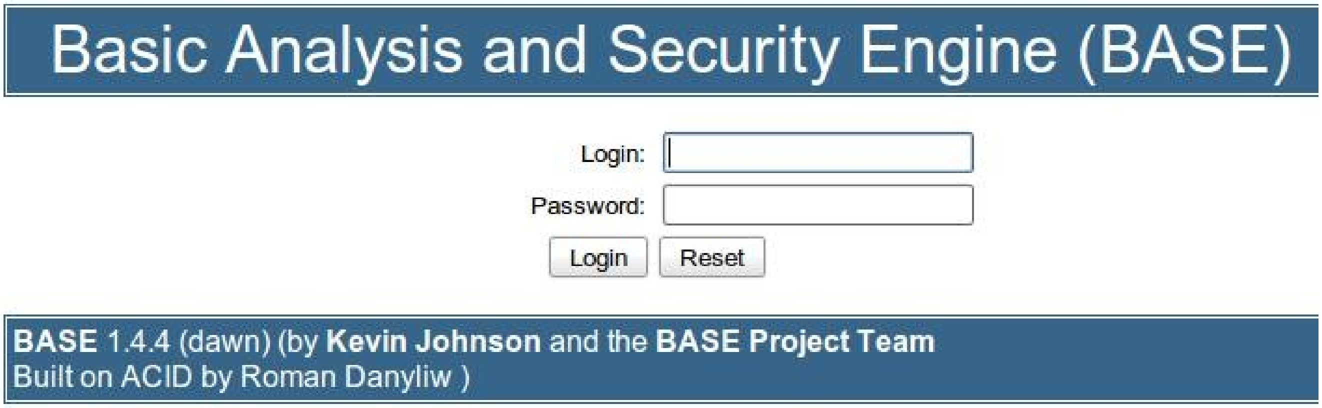
Operation	Description	Status
BASE tables	Adds tables to extend the Snort DB to support the BASE functionality	DONE

The underlying Alert DB is configured for usage with BASE.

Additional DB permissions
In order to support Alert purging (the selective ability to permanently delete alerts from the database) and DNS/whois lookup caching, the DB user "root" must have the DELETE and UPDATE privilege on the database "snort@localhost"

Now continue to step 5...

This page informs you of the completion of the BASE setup. You can now go ahead and press on the "***step 5***" link found at the bottom of the page. This shall lead you to the BASE login page:



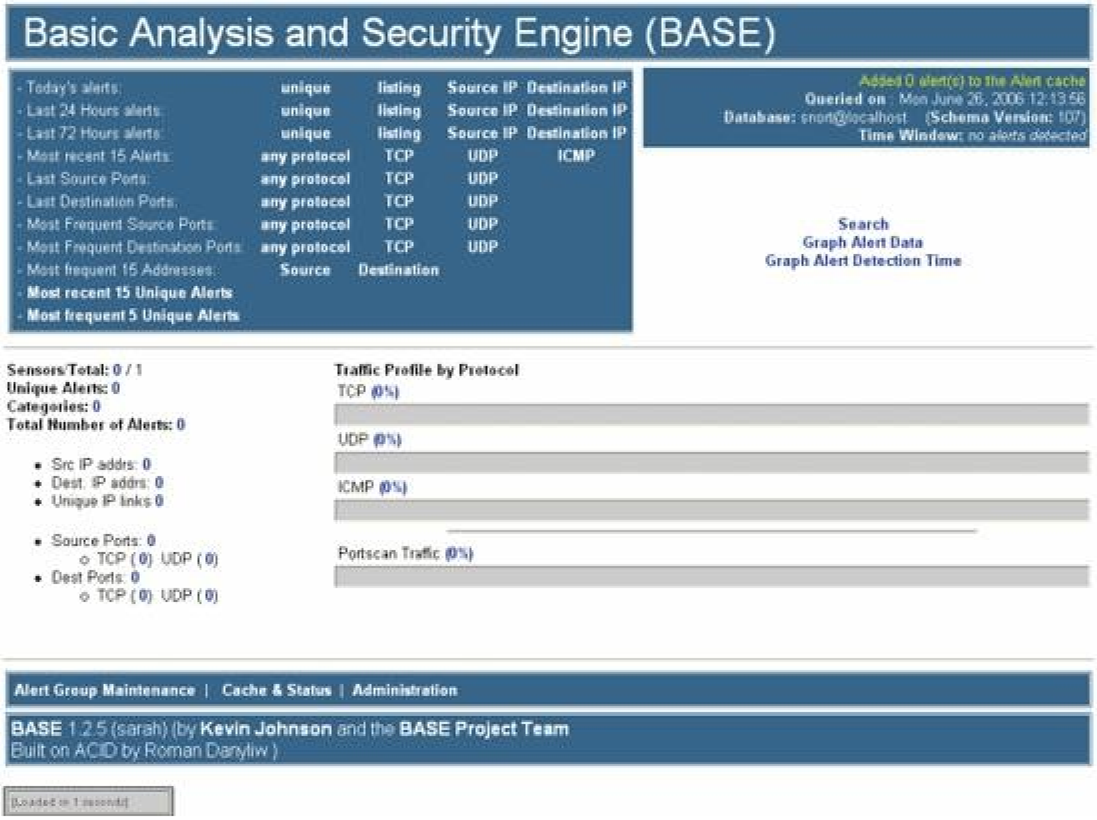
Basic Analysis and Security Engine (BASE)

Login:

Password:

BASE 1.4.4 (dawn) (by **Kevin Johnson** and the **BASE Project Team**
Built on ACID by Roman Danyliw)

Enter the username and password you chose previously, and you shall be inside BASE:



Basic Analysis and Security Engine (BASE)

- Today's alerts:	unique	listing	Source IP	Destination IP
- Last 24 Hours alerts:	unique	listing	Source IP	Destination IP
- Last 72 Hours alerts:	unique	listing	Source IP	Destination IP
- Most recent 15 Alerts:	any protocol	TCP	UDP	ICMP
- Last Source Ports:	any protocol	TCP	UDP	
- Last Destination Ports:	any protocol	TCP	UDP	
- Most Frequent Source Ports:	any protocol	TCP	UDP	
- Most Frequent Destination Ports:	any protocol	TCP	UDP	
- Most frequent 15 Addresses:	Source	Destination		
- **Most recent 15 Unique Alerts**				
- **Most frequent 5 Unique Alerts**				

Added 0 alert(s) to the Alert cache
Queried on : Mon June 26, 2006 12:13:56
Database: snort@localhost (**Schema Version:** 107)
Time Window: *no alerts detected*

Sensors/Total: 0 / 1
Unique Alerts: 0
Categories: 0
Total Number of Alerts: 0

- Src IP addrs: 0
- Dest. IP addrs: 0
- Unique IP links 0

- Source Ports: 0
  - TCP (0) UDP (0)
- Dest Ports: 0
  - TCP (0) UDP (0)

Traffic Profile by Protocol
TCP (0%)
UDP (0%)
ICMP (0%)
Portscan Traffic (0%)

BASE 1.2.5 (sarah) (by **Kevin Johnson** and the **BASE Project Team**
Built on ACID by Roman Danyliw)

[Loaded in 1 seconds]

As you can see it is clean now, there is no activity yet. To generate some noise and see if snort is doing its job? From another computer initiate a port scan using ***nmap*** on the BOX that is running Snort:
root@scanbox ~# ***nmap -sS ip-address-of-snort-box -p1-1024***

Did I mention that we havn't started Snort yet? Well yes actually we haven't and we need to do that first before our nmap noisy job gets discovered by Snort.

So download this startup script for snort from here. Copy it, and do the following:

```
[root@snortbox ~]# cp ubuntu.snort.init.txt /etc/init.d/snort
[root@snortbox ~]# chown root:root /etc/init.d/snort
[root@snortbox ~]# chmod 500 /etc/init.d/snort
```

Now if you are using eth0 for monitoring then you don't need to change the settings in the script, but if you are using for example eth1? Then edit the snort file and search for:

```
[root@snortbox ~]# vim /etc/init.d/snort
IFACE=”eth0”
```

And change it to:

```
IFACE=”eth1”
```

Save and exit “***Esc***” then “***:x***”.

Everything is ready now, lets start Snort:

```
[root@snortbox ~]# /etc/init.d/snort restart
```

Now go and run the nmap scan command again to generate some noise and watch BASE as it displays the information about that attack for you.

Congratulations now Snort is up and running.

Written by:

Ali Al-Shemery (aka: B!n@ry)
Linux Arab Community
binary [at] linuxac [dot] org

Special Thanks:

Snort Developers for the wonderful IDS/IPS.
bodhi.zazen for the startup script