HOWTO - amavisd-new + sophie + Kolab

History

Version 1.0 Stephan Buys (s.buys@codefusion.co.za) 15 July 2003

Version 1.1 Stephan Buys (s.buys@codefusion.co.za) 21 July 2003

Add eval `/kolab/etc/rc --eval all env` to sophos_pattern_update.sh script

Version 1.2 Stephan Buys (s.buys@codefusion.co.za) 21 July 2003

Add $QUARANTINEDIR = undef; to amavisd.conf section

Fix user_prefs file section for spamassassin

Version 1.3 Stephan Buys (s.buys@codefusion.co.za) 09 September 2003

- Add tweaks and improvements, thanks to Patrick Ammann, this adds the bayesian filter to SpamAssassin.

Introduction

This document outlines how to integrate Anti-Spam and Anti-Virus measures into a Kolab server. The components used to accomplish this is amavisd-new which integrates well into postfix, using Spam-assasin and Sophos (through sophie).

Software

amavisd-new

http://www.ijs.si/software/amavisd/

sophie

http://www.vanja.com/tools/sophie/

opa

Opa is a script which eases the adminitration of any Openpkg machine. It sets the environment in such a manner that your user will access the “Kolab” tools. Installing software, modifying permissions, etc. it is easier (and sometime necesarry) to use the root user.

You can download opa somewhere at www.openpkg.com or use the code listing below:

```
#OpenPKG Add Environment
opa () {
root="$1"
if [ ! -d $root ]; then
  echo "opa:ERROR: OpenPKG root directory $root not found"
  return 1
fi
if [ ! -f "$root/etc/rc" ]; then
  echo "opa:ERROR: does not contain OpenPKG hierarchy"
  return 1
fi
eval `$root/etc/rc --eval all env`
}
```

Copy the above script into .bashrc or similar and call the function opa like this:

```
#opa /kolab
```

This will set up your environment to be in Kolab.

The output of the *which rpm* command should now look as follows:

```
# which rpm
/kolab/bin/rpm
```

Important. You need to assure that you are in the Kolab environment when you do the steps in this howto. I am going to assume that you are.

Alternative ways to get into the Kolab environment as root:

```
(from the root user)
#su – kolab
(will drop you into the Kolab user)
>su
(you will be prompted to give the root password)
```

Install amavisd-new

Recommended: Read the INSTALL and AAAREADME.first files that is included in the amavisd-new distribution. This will give you insight into why you are doing the steps I list.

Download and Install prerequisites

Download the necesarry tarballs and patches from the amavisd-new website and extract them. This example assumes you are in the same folder as the archive.

```
#tar -xzf amavisd-new-<version>.tar.gz
#cd amavisd-new-<version>
```

Patch the software if necesarry:

```
#patch < amavisd-new-<version>-1.patch
```

Check that you have the necesarry software installed:

File 3.41 or later is recommended for security reasons.

```
#file -v
file-4.02
magic file from /usr/share/misc/file/magic
```

Here is a little script file that will attempt to include all the perl modules necesarry for amavisd-new to work. If it reports any errors you need make sure that the correct software is installed.

`checkmodules.pl`

```
#!/kolab/bin/perl -w
use strict;

use Archive::Tar;
use Archive::Zip;
use Compress::Zlib;
use Convert::TNEF;
use Convert::UUlib;
use MIME::Base64;
use MIME::Parser;
use Mail::Internet;
use Net::Server;
use Net::SMTP;
use Digest::MD5;
use IO::Stringy;
use Time::HiRes;
use Unix::Syslog;
use Mail::SpamAssassin;
```

Note that I used the correct version of perl (the Kolab one) in the first line of the code-listing. On my test-server I received the error:

```
#perl checkmodules.pl
Can't locate Compress/Zlib.pm in @INC ...
```

This means that I need to install the Compress::Zlib library for Perl.

Important: It is essential that you install the libraries into the correct perl folders, ie. the Kolab perl folders.

First, confirm that you are using perl in the Kolab tree.

```
#which perl
/kolab/bin/perl
```

Important: If the above command does not tell you give same output as I have listed above you are not within the Kolab environment. Do not attempt to do any further steps, please read the section called **opa** above.

An easy and convenient way to install perl modules is through the use of the CPAN module. To activate the CPAN module type the following:

```
#perl -MCPAN -e shell
```

If this the first time that you run the CPAN module you will be prompted to configure your system. Just follow the prompts until you see the prompt:

```
cpan>
```

Now install the modules that you need for amavisd-new to work. In my case I need the Compress::Zlib module, so I execute:

```
cpan>install Compress::Zlib
...
```

To test that it is installed correctly verify the output of:

```
cpan>test Compress::Zlib
...
  /kolab/bin/make test – OK
cpan>quit
```

At this stage rerun the checkmodules.pl script to see if it loads correctly.

```
 #perl checkmodules.pl
Can't locate Mail/SpamAssassin.pm in @INC...
```

I now repeat the above procedure to install Maill::Spamassassin

```
#perl -MCPAN -e shell
cpan>install Mail::SpamAssassin
cpan>test Mail::SpamAssassin
#perl checkmodules.pl
#
```

If the checkmodules.pl script just returns it means that it successfully loaded all the required modules.

To be able to scan compressed attachments you need to install the following programs:

compress, gzip, bzip2, nomarch (or arc), lha, arj (or unarj), rar (unrar),

zoo, cpio, lzop, freeze (or unfreeze or melt).

As long as they are reachable by your script during runtime it does not matter if the program is provided by your distribution or by the Kolab OpenPKG packages.

Setup amavisd-new

We will use the kolab-r (restricted Kolab) user and group to run our daemons in.

First we create the amavisd-new home directory and set appropriate permissions:

```
#mkdir /kolab/var/amavis
#chown kolab:kolab-r /kolab/var/amavis
#chmod 770 /kolab/var/amavis
```

Now we copy the executable into the Kolab tree and also set correct permissions

```
(change directory to where you extracted amavis-new)
#cp amavisd /kolab/local/sbin/
#chown kolab:kolab-r /kolab/local/sbin/amavisd
#chmod 775  /kolab/local/sbin/amavisd
```

Fix the amavisd script to use the correct version of perl. With your text editor edit:

`/kolab/local/sbin/amavisd`

```
#!/kolab/bin/perl -T

#------------------------------------------------------------------
# This is amavisd-new.
....
```

Create a folder for the amavisd configuration

```
#mkdir /kolab/etc/amavisd
#chown kolab:kolab-r /kolab/etc/amavisd
#chmod 775 /kolab/etc/amavisd
```

Copy the amavisd.conf file to the configuration folder:

```
(change directory to where you extracted amavis-new)
#cp amavisd.conf /kolab/etc/amavisd/
#chown kolab:kolab-r /kolab/etc/amavisd/amavisd.conf
```

Optional. Use steps similar to the above to create a folder into which you want to quarantine virus mails. This did not fit our requirements, we just delete virus mails.

Configure the amavisd.conf file as per your requirements.

From the INSTALL quide, the important parts are:

```
edit file /etc/amavisd.conf and adjust variables $daemon_group
and $daemon_user to match the chosen group and user name,
adjust variables $MYHOME, $TEMPBASE and $QUARANTINEDIR to match
the directories just created, then check/adjust other variables,
especially those in 'Section I', including $mydomain.
```

Important. For our Kolab server we set the following:

`/kolab/etc/amavisd/amavisd.conf`

```
$MYHOME = '/kolab/var/amavis';
$daemon_user  = 'kolab-r';
$daemon_group = 'kolab-r';
```

```
$QUARANTINEDIR = undef;         #delete virus/spam mails

(Uncomment the sophie section, search for
### http://www.vanja.com/tools/sophie/)
['Sophie',
 \&ask_daemon, ["{}/\n", '/kolab/var/sophie/sophie'],
```

To be clear, make sure you set the above mentioned variables if you want to use this install guide, also uncomment the sophie section and set the ask_daemon variable. There are other variables you need to set, this I leave up to you own judgement.

Create the SpamAssassin user_prefs file:

```
#mkdir /kolab/var/amavis/.spamassassin
#chown kolab:kolab-r /kolab/var/amavis/.spamassassin
#chmod 770 /kolab/var/amavis/.spamassassin
#touch /kolab/var/amavis/.spamassassin/user_prefs
#chown kolab:kolab-r /kolab/var/amavis/.spamassassin/user_prefs
#chmod 664 /kolab/var/amavis/.spamassassin/user_prefs
```

Before we install sophie, we will test that amavisd initializes:

```
#su - kolab-r -c '/kolab/local/sbin/amavisd -c \
/kolab/etc/amavisd/amavisd.conf debug'
```

This will execute the daemon in debug/verbose mode using the kolab-r user. Check the output carefully for any errors. You can terminate the daemon with ctrl-c.

Install Sophos Anti-Virus

Extract the sophos distribution that is suited for your machine. I used the linux libc6 version of Sophos.

From the sav-install folder issue the following to install Sophos into Kolab:

```
(from sav-install folder)
#./install.sh -d /kolab/local/ -ni -nidc -so
Sophos Anti-Virus installation utility [Linux/Intel]
Copyright (c) 1998,2001 Sophos Plc, Oxford, England
```

The options meaning: -ni (no interscan) -nidc (no interscan diskless) -so (add a libsavi.so symlink in the /kolab/local/lib folder)

Install sophie

Extract the sophie distribution and from within source distribution issue:

```
(from the sophie distribution)
#./configure --with-savilib=/kolab/local/lib/ --enable-net
...
#make
```

This will create the sophie executable.

Copy the sophie executable into the Kolab tree:

```
(from the sophie distribution)
#cp sophie /kolab/local/sbin
```

Create a sophie configuration folder and set appropriate permissions:

```
(from the sophie distribution)
#mkdir /kolab/etc/sophie
#chown kolab:kolab /kolab/etc/sophie
```

Copy the sophie configuration file to the sophie folder and set permissions.

```
(from the sophie distribution)
#cp sophie.cfg /kolab/etc/sophie
#chown kolab-r:kolab-r /kolab/etc/sophie/sophie.cfg
```

Copy the sophie savi configuration file to the sophie folder and set permissions.

```
(from the sophie distribution)
#cp etc/sophie.savi /kolab/etc/sophie
#chown kolab-r:kolab-r /kolab/etc/sophie/sophie.savi
```

Important: This file determines the behaviour of the scanner. You should verify that it has settings that suite your needs.

Edit the sophie configuration file and assure of the following:

`/kolab/etc/sophie/sophie.cfg`

```
saviconfig: /kolab/etc/sophie/sophie.savi
pidfile: /kolab/var/sophie/sophie.pid
socketfile: /kolab/var/sophie/sophie
user: kolab-r
group: kolab-r
```

Create a sophie home folder and set the correct permissions:

```
#mkdir /kolab/var/sophie
#chown kolab:kolab-r /kolab/var/sophie/
#chmod 775 /kolab/var/sophie/
```

Test if sophie starts with the following command:

```
#su - kolab-r -c '/kolab/local/sbin/sophie -C \
/kolab/etc/sophie/sophie.cfg -d'
```

This will execute the daemon in debug/verbose mode. Check the output carefully for any errors. You can terminate the daemon with ctrl-c.

Configure SpamAssassin

To added Bayesian filter support to SpamAssassin add the following text to /kolab/var/amavis/.spamassassin/user_prefs.

`/kolab/var/amavis/.spamassassin/user_prefs`

```
# Path to Bayes
bayes_path /kolab/var/amavis/.spamassassin/bayes
```

Construct the Bayes files needed for the scanner:

```
#su kolab-r
#/kolab/bin/sa-learn -p
#/kolab/var/amavis/.spamassassin/user_prefs --rebuild
```

To set SpamAssassin to learning mode:

```
#/kolab/bin/sa-learn -p
#/kolab/var/amavis/.spamassassin/user_prefs --spam ....
```

Configure amavisd-new

So far we have installed all the components that we will be using, now we will begin with the final configuration for production use.

Start sophie.

```
#su - kolab-r -c '/kolab/local/sbin/sophie -C \
/kolab/etc/sophie/sophie.cfg -D'
```

Start the amvisd-new daemon.

```
#su - kolab-r -c '/kolab/local/sbin/amavisd -c \
/kolab/etc/amavisd/amavisd.conf'
```

This time we did not specify debug, so amavisd will load and fork into the background.

To test that the daemon is running as expected issue the following:

```
#telnet 127.0.0.1 10024
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is '^]'.
220 [127.0.0.1] ESMTP amavisd-new service ready
#quit
221 2.0.0 [127.0.0.1] (amavisd) closing transmission channel
Connection closed by foreign host.
```

Kolab Postfix master.cf Configuration

Kolab uses template files to manage changes that occur in LDAP. It is critical that you update the correct template files as any change in the LDAP directory will override the main configuration files with their templates.

Important: To be clear about it: Dont change the /kolab/etc/postfix/master.cf file. It will be overwritten by /kolab/etc/kolab/master.cf.template.

We add the following lines to master.cf.template.

`/kolab/etc/kolab/master.cf.template`

```
smtp-amavis unix -      -       n       -       2       smtp
    -o smtp_data_done_timeout=1200
    -o disable_dns_lookups=yes
127.0.0.1:10025 inet n  -       n     -       -  smtpd
    -o content_filter=
    -o local_recipient_maps=
    -o relay_recipient_maps=
    -o smtpd_restriction_classes=
    -o smtpd_client_restrictions=
    -o smtpd_helo_restrictions=
    -o smtpd_sender_restrictions=
    -o smtpd_recipient_restrictions=permit_mynetworks,reject
    -o mynetworks=127.0.0.0/8
    -o strict_rfc821_envelopes=yes
```

To propogate the changes to our postfix configuration files:

```
#/kolab/etc/kolab/kolab -v -o
```

This runs the kolab backend in verbose (-v) mode once (-o) – in other words it does not go into daemon mode. We most probably already have a kolab daemon running.

To verify that your changes were propogated, issue the following.

```
#tail -14 /kolab/etc/postfix/master.cf
```

It should show the lines above.

Reload the postfix daemon:

```
#postfix reload
```

This is one of those commands that depend on you being in the Kolab environment. Check the opa section if you dont know what I mean.

Verify that the daemon reloaded successfully:

```
#tail -20 /kolav/var/postfix/log/postfix.log
```

To check that postfix added a listener for us on the correct port we telnet to the new listener on the local host.

```
#telnet 127.0.0.1 10025
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is '^]'.
220 mail-router1.internal.adtza.co.za ESMTP Postfix
#quit
221 Bye
Connection closed by foreign host.
```

Testing amavis-new, sophie and postfix

First we will send a normal, untainted message via the amavis listener and check that it gets delivered to the end-user.

```
#telnet 127.0.0.1 10024
Trying 127.0.0.1...
Connected to 127.0.0.1.
Escape character is '^]'.
220 [127.0.0.1] ESMTP amavisd-new service ready
#MAIL FROM:<test@example.com>
250 2.1.0 Sender test@example.com OK
#RCPT TO:<postmaster>
250 2.1.5 Recipient postmaster OK
#DATA
354 End data with <CR><LF>.<CR><LF>
#Subject: test1
#
#test1
#.
250 2.6.0 Ok, id=26807-01, from MTA: 250 Ok: ...
```

This will send a message with the subject test1 to postmaster. If you inspect the contents of the file there should be an entry:

```
X-Virus-Scanned: by amavisd-new at mydomain.com
```

While still connected to the port, we will check sophie with the Eicar.org test pattern.

```
#MAIL FROM:<test@example.com>
250 2.1.0 Sender test@example.com OK
#RCPT TO:<postmaster>
250 2.1.5 Recipient postmaster OK
#DATA
354 End data with <CR><LF>.<CR><LF>
#Subject: test2
#
#X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-
FILE!$H+H*
#.
250 2.5.0 Ok, but 1 BOUNCE
```

As you can see I set amavisd-new to reject mail and bounce it. You might want to ensure that the MAIL FROM: command is from a valid user, this will assure that you can verify the amavisd-new message.

Kolab Postfix main.cf Configuration

Now that we have verified that the scanner is working according to expectations we will set postfix to always forward messages to the amavisd-new scanner.

Edit the main.cf.template file and add the following line:

`/kolab/etc/kolab/main.cf.template`

```
content_filter = smtp-amavis:[127.0.0.1]:10024
```

Ensure that the template file change is propogated to the postfix configuration files:

```
#/kolab/etc/kolab/kolab -v -o
```

Verify that the change was properly propogated:

```
#grep content_filter /kolab/etc/postfix/main.cf
content_filter = smtp-amavis:[127.0.0.1]:10024
```

Reload the postfix daemon's configuration with:

```
#postfix reload
```

Mail coming through your server should now be content scanned by amavisd-new and sophos (through) sophie. You can verify this by sending a message through the server and verifying that it contains the X-Virus-Scanned tag mentioned above.

Startup and shutdown

We need to assure that we can properly start and stop these services as well as reload them when needed. To this extent I have created the necesarry files for startup and shutdown of the services.

Aquire the files rc.amavisd, rc.sophie and rc.kolab.diff from either Apendix A or from a better download location.

Set up the rc.sophie script:

```
#cp rc.sophie /kolab/etc/rc.d/
#chown kolab:kolab /kolab/etc/rc.d/rc.sophie
#chmod 755 /kolab/etc/rc.d/rc.sophie
```

Set up the rc.amavisd script:

```
#cp rc.amavisd /kolab/etc/rc.d/
#chown kolab:kolab /kolab/etc/rc.d/rc.amavisd
#chmod 755 /kolab/etc/rc.d/rc.amavisd
```

Patch the rc.kolab script to call the new scripts

```
#cd /kolab/etc/rc.d/
#patch < /path/to/rc.kolab.diff
```

You can test the scripts by restaring Kolab:

```
#/kolab/etc/rc.d/rc.kolab stop
stopping kolab backend ...
stopping postfix ...
postfix/postfix-script: stopping the Postfix mail system
stopping amavisd...
stopping sophie...
stopping apache ...
```

```
/kolab/sbin/apachectl stop: httpd stopped
stopping cyrus imapd ...
stopping saslauthd ...
stopping openldap ...
stopping proftpd (if running) ...

#/kolab/etc/rc.d/rc.kolab start
starting openldap ...
starting sasl ...
starting cyrus imapd ...
starting apache ...
/kolab/sbin/apachectl start: httpd started
starting sophie
/kolab/local/sbin/sophie placed in the background
starting amavisd
starting postfix ...
postfix/postfix-script: starting the Postfix mail system
starting kolab backend (please wait)
```

Maintaining Amavisd-new

It is necessary to clean up amavis temporary files every now and then. To this end Patrick Ammann has contributed the following cleanup script:

`/kolab/local/bin/amavis_cleanup.sh`

```
#!/bin/sh

# delete not used/forgotten temporary data
find /kolab/var/amavis -type d -name 'amavis-20??????T*' \
    -prune -mtime +1 -exec rm -rf {} \;
```

See the section, “Scheduling Tasks” for instructions on how to automate the process.

Updating Sophos

There are two aspects to keeping Sophos up to date. Firstly you need to keep the Sophos engine up to date. For this we will be using the sophos_full_update.sh script which can be obtained from Appendix A.

Sophos scan engine

Install the sophos_full_update.sh script as follows:

```
#cp /path/to/sophos_full_update.sh /kolab/local/bin/
#chown kolab:kolab /kolab/local/bin/sophos_full_update.sh
#chmod 755 /kolab/local/bin/sophos_full_update.sh
```

You will need to edit the script and set the username and password fiels.

`/kolab/local/bin/sophos_full_update.sh`

```
username='yoursophosid'
password='yoursophospassword'
```

The script downloads the latest version of the engine, installs it under /kolab/local and restarts sophie.

To test the update script we will first check the version of the currently installed sophos engine.

```
#/kolab/local/bin/sweep -v
SWEEP virus detection utility
Copyright (c) 1989,2003 Sophos Plc, www.sophos.com

System time 12:35:33, System date 16 July 2003

Product version           : 3.69
Engine version            : 2.14
User interface version    : 2.05.034
Platform                  : Linux/Intel
Released                  : 05 May 2003
Total viruses (with IDEs) : 81434
```

As you can see on the “Product version” line this demo has version 3.69 of Sophos.

Now we execute the script and monitor the output of the logfile.

```
#/kolab/local/bin/sophos_full_update
#cat /var/log/sophos_update.log
```

After execution your syslog file should also report that Sophie has reloaded successfully with the updated engine.

```
/var/log/syslog
```

```
NOTICE           : Reloading virus patterns/engine
NOTICE           : Virus patterns/engine reloaded
NOTICE           : Setting configuration options - please
wait...
NOTICE           : Configuration options set
Sophos engine    : Sophos engine version 2.14
Sophie IDE       : Sophos IDE version 3.71 (detects 82916
viruses)
```

Sophos pattern (IDE) files

For the pattern file updates we will use a script called sophos_pattern_update.sh.

Install the sophos_pattern_update.sh script as follows:

```
#cp /path/to/sophos_pattern_update.sh /kolab/local/bin/
#chown kolab:kolab /kolab/local/bin/sophos_pattern_update.sh
#chmod 755 /kolab/local/bin/sophos_pattern_update.sh
```

To test if the script is performing as expected we execute:

```
#/kolab/local/bin/sophos_pattern_update.sh
#cat /var/log/sophos_pattern_update.log
```

After execution your syslog file should also report that Sophie has reloaded successfully with the updated engine.

```
/var/log/syslog
```

```
NOTICE           : Reloading virus patterns/engine
NOTICE           : Virus patterns/engine reloaded
NOTICE           : Setting configuration options - please
wait...
NOTICE           : Configuration options set
Sophos engine    : Sophos engine version 2.14
Sophie IDE       : Sophos IDE version 3.71 (detects 82916
viruses)
```

You can also verify that the IDE files were loaded by running:

```
#/kolab/local/bin/sweep -v
...
Information on additional data files:

Data file name            : /kolab/local/sav/coconuta.ide
Data file type            : IDE
Data file date            : 16 July 2003, 12:22:53
Data file status          : Loaded
...
```

Scheduling Tasks

It is critical that the Sophos pattern files keep up to date and that the engine stays current. To this extent it is highly recommended that you schedule a task to run the update scripts regularly, also Amavis temp files need to be cleaned up.

To edit cron:

```
#crontab -e
```

Add the entries for a hourly pattern file update and a monthly engine update:

crontab

```
1 * * * *    /kolab/local/bin/sophos_pattern_update.sh
0 0 1 * *     /kolab/local/bin/sophos_full_update.sh
0 23 * * *      /kolab/local/bin/amavis_cleanup.sh
```

Conclusion

At this point you should have a proper anti-virus and anti-spam solution integrated into your Kolab server.

Please report any comments, errors or improvements to kolab@codefusion.co.za

Enjoy!

Apendix A – Supporting files

/kolab/etc/rc.d/rc.amavisd

```
#!/kolab/lib/openpkg/bash /kolab/etc/rc
##
##  rc.amavisd
##
##  Copyright (c) 2003 Stephan Buys <s.buys@codefusion.co.za>
##  adopted from: rc.kolab
##  rc.kolab:
##  Copyright (c) 2002 Martin Konold <martin.konold@erfrakon.de>
##  Copyright (c) 2002 Tassilo Erlewein
<tassilo.erlewein@erfrakon.de>


%config
    amavisd_enable="yes"

%start -p 200 -u root
    opServiceEnabled amavisd || exit 0

    echo "starting amavisd"
    if [ -f /kolab/var/amavis/amavisd.pid ]; then
       PID=`cat /kolab/var/amavis/amavisd.pid | awk '{print
$1}'`
       AMAVISS=`ps -p $PID 2>/dev/null | grep -c amavisd | awk
'{print $1}'`
       if [ $AMAVISS -gt 0 ]; then
          echo "Warning: amavisd is already running under pid
$PID!"
       else
          /kolab/local/sbin/amavisd -c
/kolab/etc/amavisd/amavisd.conf
       fi
    else
       /kolab/local/sbin/amavisd -c
/kolab/etc/amavisd/amavisd.conf
    fi

%stop -p 200 -u root
    opServiceEnabled amavisd || exit 0

    echo "stopping amavisd..."
    if [ -f /kolab/var/amavis/amavisd.pid ]; then
       kill `cat /kolab/var/amavis/amavisd.pid` 2>/dev/null
    fi
    sleep 1
    killall -9 amavisd 2>/dev/null
    exit 0

%reload -u root
    opServiceEnabled amavisd || exit 0

    echo "reload amavisd (if running) ..."
    if [ -f /kolab/var/amavis/amavisd.pid ]; then
```

```
      kill -HUP `cat /kolab/var/amavis/amavisd.pid` 2>/dev/null
    fi



%restart -u root
    opServiceEnabled amavisd || exit 0
    /kolab/etc/rc.d/rc.amavisd stop
    sleep 1
    /kolab/etc/rc.d/rc.amavisd start
```

/kolab/etc/rc.d/rc.sophie

```
#!/kolab/lib/openpkg/bash /kolab/etc/rc
##
##  rc.sophie
##
##  Copyright (c) 2003 Stephan Buys <s.buys@codefusion.co.za>
##  adopted from: rc.kolab
##  rc.kolab:
##  Copyright (c) 2002 Martin Konold <martin.konold@erfrakon.de>
##  Copyright (c) 2002 Tassilo Erlewein
<tassilo.erlewein@erfrakon.de>



%config
    sophie_enable="yes"

%start -p 200 -u root
    opServiceEnabled sophie || exit 0

    echo "starting sophie"
    if [ -f /kolab/var/sophie/sophie.pid ]; then
       PID=`cat /kolab/var/sophie/sophie.pid | awk '{print $1}'`
       SOPHIES=`ps -p $PID 2>/dev/null | grep -c sophie | awk
'{print $1}'`
       if [ $SOPHIES -gt 0 ]; then
          echo "Warning: sophie is already running under pid
$PID!"
       else
          /kolab/local/sbin/sophie -C
/kolab/etc/sophie/sophie.cfg -D
       fi
    else
       /kolab/local/sbin/sophie -C /kolab/etc/sophie/sophie.cfg
-D
    fi

%stop -p 200 -u root
    opServiceEnabled sophie || exit 0

    echo "stopping sophie..."
    if [ -f /kolab/var/sophie/sophie.pid ]; then
       kill `cat /kolab/var/sophie/sophie.pid` 2>/dev/null
    fi
    sleep 1
    killall -9 sophie 2>/dev/null
    exit 0

%reload -u root
    opServiceEnabled sophie || exit 0

    echo "reload sophie (if running) ..."
    if [ -f /kolab/var/sophie/sophie.pid ]; then
      kill -HUP `cat /kolab/var/sophie/sophie.pid` 2>/dev/null
    fi


%restart -u root
    opServiceEnabled sophie || exit 0
    /kolab/etc/rc.d/rc.sophie stop
    sleep 1
    /kolab/etc/rc.d/rc.sophie start
```

rc.kolab.diff

```
--- rc.kolab    2003-07-04 14:31:26.000000000 +0200
+++ rc.kolab-new        2003-07-15 15:23:45.000000000 +0200
@@ -57,6 +57,10 @@
     echo "starting apache ..."
     /kolab/sbin/apachectl start

+    /kolab/etc/rc.d/rc.sophie start
+
+    /kolab/etc/rc.d/rc.amavisd start
+
     echo "starting postfix ..."
     /kolab/sbin/postfix start

@@ -87,6 +91,10 @@
     echo "stopping postfix ..."
     /kolab/sbin/postfix stop

+    /kolab/etc/rc.d/rc.amavisd stop
+
+    /kolab/etc/rc.d/rc.sophie stop
+
     echo "stopping apache ..."
     /kolab/sbin/apachectl stop

@@ -153,6 +161,10 @@
       kill -HUP `cat /kolab/var/imapd/imapd.pid` 2>/dev/null
     fi

+    /kolab/etc/rc.d/rc.sophie reload
+
+    /kolab/etc/rc.d/rc.amavisd reload
+
     echo "reload postfix ..."
     /kolab/sbin/postfix reload
```

sophos_full_update.sh

```
#!/bin/sh

username='yoursophosid'
password='yoursophospassword'
sav_installation_dir='/kolab/local'
tar_location='/bin/tar'
wget_location='/usr/bin/wget'
log_file='/var/log/sophos_update.log'
temp='/tmp'
sophie_rc='/kolab/etc/rc.d/rc.sophie'
download_location='http://www.sophos.com/sophos/products/full/'

downloadfile='linux.intel.libc6.tar.Z'


check_if_file_already_exists ()
        {
        echo "Checking for old files" > $log_file
        if [ -f $temp/$downloadfile ]; then
        echo "File already exists. Removing
$temp/$downloadfile" >> $log_file
        rm -rf $temp/$downloadfile
        fi
        }

download ()
        {
        echo "Downloading engine" >> $log_file
        $wget_location -P$temp --http-user=$username --http-
passwd=$password $download_location/$downloadfile >> $log_file
2>&1
        }

check_download ()
        {
        echo "Check for $temp/$downloadfile..."
        if [ -f $temp/$downloadfile ]; then
        echo "File collected."
        else
        echo "WARNING!! Download was not successful. Check your
connection to ww
w.sophos.com" >> $log_file
        error_exit
        fi
        }

untar_sav ()
        {
        echo "Extracting sav-install in $temp" >> $log_file
        $tar_location -C $temp -zxvf $temp/$downloadfile >>
$log_file || error_e
xit
        }
```

```
install_sav ()
        {
        echo "Installing from in $temp..." >> $log_file
        cd $temp/sav-install/
        ./install.sh -ni -nidc -so -rm -v -d
$sav_installation_dir >> $log_file
|| error_exit
        }

clean_up ()
        {
        echo "Cleaning up in $temp..." >> $log_file
        rm -rf $temp/$downloadfile
        rm -rf $temp/sav-install
        }

restart_sophie ()
        {
        $sophie_rc reload
        }

error_exit ()
        {
        echo "WARNING!! Sophos Anti virus was not updated!" >>
$log_file
        exit 2
        }
#
# Run the functions from here
check_if_file_already_exists
download
check_download
untar_sav
install_sav
clean_up
restart_sophie
```

sophos_pattern_update.sh

```
#!/bin/sh
eval `/kolab/etc/rc --eval all env`
sav_installation_dir='/kolab/local'
unzip_location='/usr/bin/unzip'
wget_location='/usr/bin/wget'
log_file='/var/log/sophos_pattern_update.log'
temp='/tmp'
download_location='http://www.sophos.com/downloads/ide/'
sophie_rc='/kolab/etc/rc.d/rc.sophie'
sophos_version=''
downloadfile=''

get_sophos_version ()
        {
        sophos_version=`$sav_installation_dir/bin/sweep -v |
grep "Product version" | awk 'BEGIN { FS = " +: +" } ; { print
$2 }' | sed "s/\.//"`
        downloadfile="${sophos_version}_ides.zip"
        }



check_if_file_already_exists ()
        {
        echo "Checking for old files" > $log_file
        if [ -f $temp/$downloadfile ]; then
        echo "File already exists. Removing
$temp/$downloadfile" >> $log_file
        rm -rf $temp/$downloadfile
        fi
        }

download ()
        {
        echo "Downloading patterns" >> $log_file
        $wget_location -P$temp --http-user=$username --http-
passwd=$password $download_location/$downloadfile >> $log_file
2>&1
        }

check_download ()
        {
        echo "Check for $temp/$downloadfile..." >> $log_file
        if [ -f $temp/$downloadfile ]; then
        echo "File collected." >> $log_file
        else
        echo "WARNING!! Download was not successful. Check your
connection to www.sophos.com" >> $log_file
        error_exit
        fi
        }

clear_ide ()
        {
        echo "Removing old ide files in
$sav_installation_dir/sav" >> $log_file
        rm -f $sav_installation_dir/sav/*.ide
        }

unzip_ide ()
        {
        echo "Extracting ides in $sav_installation_dir/sav" >>
$log_file
        cd $sav_installation_dir/sav
        $unzip_location $temp/$downloadfile >> $log_file
        }

clean_up ()
        {
        echo "Cleaning up in $temp..." >> $log_file
        rm -rf $temp/$downloadfile
        }

restart_sophie ()
        {
        $sophie_rc reload
        }

error_exit ()
        {
        echo "WARNING!! Sophos Anti virus was not updated!" >>
$log_file
        exit 2
        }
#
# Run the functions from here
get_sophos_version
check_if_file_already_exists
download
check_download
clear_ide
unzip_ide
clean_up
restart_sophie
```