

How to Install MRTG

OVERVIEW	2
INSTALL.....	2
BUILD INFRASTRUCTURE.....	2
<i>Supporting Code</i>	2
<i>MRTG</i>	2
<i>Define pseudo-user</i>	2
<i>Create Web Support</i>	3
CONFIGURE	4
<i>Modify build_octet_config</i>	4
<i>Modify build_octet_config.conf</i>	5
<i>Set ownership & permissions</i>	5
<i>Modify mrtg-rrd.cgi</i>	5
<i>Modify mrtg-html.cgi</i>	5
INIT SCRIPTS.....	6
AUTO-REBUILDING CONFIG FILES.....	8
<i>rebuild_mrtg_configs</i>	8
<i>stop-start-mrtg</i>	9
TROUBLE-SHOOTING.....	11
PID FILES	11
EXAMINE LOG FILES.....	11
APPENDIX	11
BUILD_OCTET_CONFIG.CONF	11

OVERVIEW

This document describes the installation and operation of MRTG.

See the following for additional documentation:

<http://www.mrtg.org>

<http://www.enterastream.com/whitepapers/mrtg/mrtg-manual.html>

<http://www.linuxhomenetworking.com/linux-hn/mrtg-advanced.htm>

INSTALL

Build Infrastructure

SUPPORTING CODE

Verify that perl, zlib, jpeg, freetype, libpng, gd, net-snmp, and rrdtool are installed. See [compile-suse.doc](#) for details.

MRTG

Compile and install MRTG. See [compile-suse.doc](#) for details.

DEFINE PSEUDO-USER

Create a pseudo-user which will run MRTG -- we use "netops". We need non-trivial disk space to hold the data (~40GB currently).

Create Directory Structure

```
mkdir /home/netops
mkdir /home/netops/bin
mkdir /home/netops/etc
mkdir /home/netops/var
mkdir /home/netops/var/log
mkdir /home/netops/var/mrtg
mkdir /home/netops/var/mrtg/abc_blg
mkdir /home/netops/var/mrtg/core
mkdir /home/netops/var/mrtg/d_blg
mkdir /home/netops/var/mrtg/distribution
mkdir /home/netops/var/mrtg/dmz
mkdir /home/netops/var/mrtg/g_blg
mkdir /home/netops/var/mrtg/hosts
mkdir /home/netops/var/mrtg/j_blg
mkdir /home/netops/var/mrtg/m_blg
mkdir /home/netops/var/mrtg/mmz
mkdir /home/netops/var/mrtg/mp_blg
mkdir /home/netops/var/mrtg/remote-access
mkdir /home/netops/var/mrtg/satellite
mkdir /home/netops/var/mrtg/ups
```

```
cd /home/netops
chown -R netops:local *
chmod -R 770 *
cd /home/netops/var/mrtg
```

```
chmod -R 777 *
```

Install in-house code

Populate /home/netops/bin with in-house code (build_xxxx_config).

```
chown netops:local /home/netops/bin/*
chmod 770 /home/netops/bin/*
```

CREATE WEB SUPPORT

mod_perl support

Compile apache with mod_perl support.

Install “startup.pl” in ~/apache/lib

```
use strict;
use warnings;

use lib "/opt/local/rrdtool/lib/perl";
use lib "/opt/local/apache/lib";

use Apache::Registry();
use Apache::Constants();

use Carp();
$SIG{__WARN__} = \&Carp::cluck;

use CGI();
CGI->compile(':all');

use Getopt::Std();
use Getopt::Long();
use HTML::Parser();
use IO::File;
use POSIX();
use RRDs();
use Time::Local();
use URI();

1;
```

Populate ~/apache/lib/perl/MRTG with mod_perlerized versions of mrtg-rrd.cgi. See Hosts.pm and ByteOMeter.pm as examples.

Modify HTTPD.CONF

Copy the old httpd.conf to the new location. Here are extracts of the key mods.

Add “ExecCGI” to the “Options” line under <Directory "/opt/local/apache/htdocs"> to allow CGI execution anywhere.¹

```
Options Indexes FollowSymLinks MultiViews ExecCGI
```

Uncomment the relevant AddHandler line:

¹ Would be cooler to keep CGI scripts in the cgi-bin directory, but I haven't figured out how to make this work..

```
AddHandler cgi-script .cgi .fcgi
```

Add index.cgi to the DirectoryIndex line

```
DirectoryIndex index.html index.cgi
```

In the mod_perl section, add handlers for each copy of mod_perlerized mrtg-rrd.cgi:

```
### Section 4: mod_perl
<LocationMatch /*ByteOMeter>
  SetHandler perl-script
  PerlHandler +MRTG::ByteOMeter
  PerlSendHeader On
</LocationMatch>

<LocationMatch /*Hosts>
  SetHandler perl-script
  PerlHandler +MRTG::Hosts
  PerlSendHeader On
</LocationMatch>
```

File & Directory Structure

Create the following file & directory structure:

```
cd ~/apache/htdocs
mkdir mrtg
cd mrtg
cp /opt/local/mrtg-2/share/mrtg2/icons/* .
chown netops:local *png *gif
chmod 444 *png *gif
ln -s /home/netops ./htdocs/mrtg/netops
```

Install CGI scripts

Populate the ..apache/htdocs/mrtg directory tree.

```
cd /opt/local/apache/htdocs
tar cf mrtg.tar mrtg
cd /newlocation
tar xvf /oldlocation/mrtg.tar
chown -R netstats:local *cgi
chmod -R 555 *cgi
```

Configure

MODIFY BUILD_OCTET_CONFIG

Edit build_octet_config and set the Configuration Variable section as follows:

```
#####
##### Begin user-configuration variables

# Directories
my $cfg_dir = "/home/netops/etc";
my $work_dir = "/home/netops/var/mrtg";

# File locations
```

```

my $cfg_file_default = "/home/netops/etc/build_octet_config.conf";
my $hosts_file_default = "/etc/hosts";
my $cfgmaker_default = "/opt/local/mrtg-2/bin/cfgmaker";

# Miscellaneous
my $version = "2.00";

# Group Options
my $community = "public";
my $sif_desc = "name";
my $sif_filter = '((($sif_is_ethernet or $sif_is_wan) and not $sif_is_ciscovlan)';
my $sif_ref = "name";

# Global Options
my $cfgmaker_options = "--global 'Refresh: 600' --global 'IconDir: /mrtg/icons'
--global 'RunAsDaemon: Yes' --global 'Interval: 5' --global 'Options[_]: growrig
ht' --snmp-options=::::2 --global 'LogFormat: rrdtool' --global 'PathAdd: /opt/
local/rrdtool/bin' --global 'LibAdd: /opt/local/rrdtool/lib/perl' --global 'With
Peak[_]: wmy' --global 'Unscaled[_]: dwmy' --no-down";

##### End user-configuration variables
#####

```

MODIFY BUILD_OCTET_CONFIG.CONF

build_octet_config uses a configuration file; see the Appendix for the current version.

SET OWNERSHIP & PERMISSIONS

```

cd /home/netops/etc
chown netops:local *cfg
chmod 664 *cfg

```

MODIFY MRTG-RRD.CGI

Find the “BEGIN { @config_files = qw” section and make it look like this.

```

# EDIT THIS to reflect all your MRTG config files
BEGIN { @config_files = qw(
    /home/netops/etc/abc_blg.cfg
    /home/netops/etc/core.cfg
    /home/netops/etc/cpu.cfg
    /home/netops/etc/d_blg.cfg
    /home/netops/etc/distribution.cfg
    /home/netops/etc/dmz.cfg
    /home/netops/etc/g_blg.cfg
    /home/netops/etc/j_blg.cfg
    /home/netops/etc/m.cfg
    /home/netops/etc/remote-access.cfg
    /home/netops/etc/routers.cfg
    /home/netops/etc/switches.cfg
    /home/netops/etc/ups.cfg
); }

```

MODIFY MRTG-HTML.CGI

```
#####
# Begin Configurable Section

# Default source.
my $default_source = "lan-man.htm";

# The web-accessible location of the mrtg-rrd.cgi.
# Note: Do not put a trailing "/" in the URL.
my $base_url = "/mrtg/mrtg-rrd.cgi";

# The MRTG config files used by $base_url (the contents of @config_files).
my @config_files = qw(
    /home/netops/etc/abc_blg.cfg
    /home/netops/etc/core.cfg
    /home/netops/etc/cpu.cfg
    /home/netops/etc/d_blg.cfg
    /home/netops/etc/distribution.cfg
    /home/netops/etc/dmz.cfg
    /home/netops/etc/g_blg.cfg
    /home/netops/etc/j_blg.cfg
    /home/netops/etc/m.cfg
    /home/netops/etc/mp_blg.cfg
    /home/netops/etc/remote-access.cfg
    /home/netops/etc/routers.cfg
    /home/netops/etc/switches.cfg
);

# Class B network prefix.
# Note: Do not put a trailing "." in the prefix.
my $ip_prefix = "140.107";

# Restrict the possible sources to the following regular expressions:
# Note that "/./" is not allowed as a sub-string implicitly and that
# all the regular expressions are wrapped in /^re$/.
my @allowed_sources = ("lan-man.htm");

# Restrict the possible mrtg configs to the following regular expressions:
# Note that "/./" is not allowed as a sub-string implicitly and that
# all the regular expressions are wrapped in /^re$/.
my @allowed_mrtg_configs = ("/home/netops/etc/.\.cfg");

# End Configurable Section
#####
```

Init Scripts

Here is a sample init.d script:

```
/etc/init.d/mrtg-dmz
#!/bin/bash
#
# Startup script for mrtg-dmz
#
# chkconfig: - 85 15
# description: MRTG is a data collection engine. It is used to gather
#              utilization statistics against network devices and end-stations
# processname: mrtg
# pidfile: /home/netops/etc/dmz.pid
# config: /home/netops/etc/dmz.cfg

# Source function library.
. /etc/init.d/functions

RETVAL=0
```

```

# See how we were called.

cfg="/home/netops/etc/dmz.cfg"
mrtg="/opt/local/mrtg-2/bin/mrtg"
logging="--logging /home/netops/var/log/dmz.log"
pidfile="/home/netops/etc/dmz.pid"
prog="dmz"
user="--user=netops --group=local"

start() {
    echo -n "Starting $prog: "
    daemon $mrtg $cfg $user $logging
    RETVAL=$?
    echo
    [ $RETVAL -eq 0 ]
    return $RETVAL
}

stop() {
    pid=`/bin/cat $pidfile`
    if [ ! -z "$pid" ]; then
        echo -n "Stopping $prog: "
        kill $pid
    fi
    RETVAL=$?
    echo
    [ $RETVAL -eq 0 ]
    return $RETVAL
}

rhstatus() {
    status $prog
}

restart() {
    stop
    start
}

reload() {
    echo -n "Reloading $prog daemon configuration: "
    killproc $prog -HUP
    retval=$?
    echo
    return $RETVAL
}

case "$1" in
    start)
        start
        ;;
    stop)
        stop
        ;;
    restart)
        restart
        ;;
    reload)
        reload
        ;;
    status)
        rhstatus
        ;;
    condrestart)
        restart || :
        ;;
    *)

```

```

        echo $"Usage: $0 {start|stop|status|reload|restart|condrestart}"
        exit 1
esac

exit $?

```

Populate /etc/init.d with the appropriate scripts. Then link these to versions in /etc/rc2.d:

e.g.

```
ln /etc/init.d/mrtg-dmz /etc/rc2.d/S85mrtg-dmz
```

Auto-Rebuilding Config Files

The 'netops' user runs a cronjob called "rebuild_mrtg_configs" every night to rebuild the config files.

REBUILD_MRTG_CONFIGS

```

[skendric@llua01 bin]$ pwd
/home/netops/bin
[skendric@llua01 bin]$ ls
build_host_config      build_temperature_config  rebuild_mrtg_configs
build_list             build_ups_config          stop-start-mrtg
build_octet_config     build_wap_client_config
build_router_config    build_wap_rf_int_config
build_switch_config    cfg_mrtg
[skendric@llua01 bin]$ cat rebuild_mrtg_configs
#!/opt/local/bin/perl

# This script automates the process of building the MRTG config files,
# stopping the MRTG processes, wiping their log files, and restarting
# them

# V      Who          When          What
# -----
# 1.02   skendric     04-28-2004   Add build_hosts_config
# 1.01   skendric     04-10-2004   Use strict/warnings
# 1.0    skendric     09-23-2003   First Version
#

# Load modules
use strict;
use warnings;

# Declare/define global variables
my $chmod = "/bin/chmod";
my $chown = "/bin/chown";
my $find = "/usr/bin/find";
my $grep = "/bin/grep";
my $home = "/home/netops";
my $mrtgDir = "$home/var/mrtg";
my $owners = "netops:local";
my $rm = "/bin/rm";
my $xargs = "/usr/bin/xargs";

# Rebuild config files
system ("$home/bin/build_host_config -f $home/etc/hosts.conf");
system ("$home/bin/build_octet_config");
system ("$home/bin/build_router_config all");

```

```

system ("$home/bin/build_switch_config all");
system ("$home/bin/build_temperature_config all");
system ("$home/bin/build_ups_config all");
system ("$home/bin/build_wap_client_config all");
system ("$home/bin/build_wap_rf_int_config all");

# Wipe backup configs
system ("rm $home/etc/*0");

# Stop MRTG processes, wipe logs, start them again
system ("$home/bin/stop-start-mrtg");

# Set ownership & permissions
system ("chown $owners $mrtgDir");
system ("chmod 777 $mrtgDir");
system ("find $mrtgDir | grep rrd | xargs chown $owners");
system ("find $mrtgDir | grep rrd | xargs chmod 664");

# Delete PNG files
`find $mrtgDir | grep png | xargs rm 2>&1`;

```

STOP-START-MRTG

```

[skendric@llua01 bin]$cat stop-start-mrtg
#!/opt/local/bin/perl

# This script automates the process of stopping the MRTG processes,
# wiping their log files, and restarting
# them

# V      Who      When      What
# -----
# 1.02  skendric   04-28-2004  Divide m_blg into two files
# 1.01  skendric   04-10-2004  Use strict/warnings
# 1.0   skendric   03-23-2004  First Version
#

# Load modules
use strict;
use warnings;

# Declare/define variables
my $cat = "/bin/cat";
my $chmod = "/bin/chmod";
my $chown = "/bin/chown";
my $find = "/usr/bin/find";
my $grep = "/bin/grep";
my $init = "/etc/init.d";
my $logDir = "/home/netops/var/log";
my $mrtgDir = "/home/netops/var/mrtg";
my $null = "/dev/null";
my $owners = "netops:local";
my $rm = "/bin/rm";
my $xargs = "/usr/bin/xargs";

# Stop MRTG processes
system ("$init/mrtg-abc_blg stop");
system ("$init/mrtg-bag stop");
system ("$init/mrtg-byte-o-meter stop");
system ("$init/mrtg-core stop");
system ("$init/mrtg-d_blg stop");
system ("$init/mrtg-distribution stop");
system ("$init/mrtg-dmz stop");

```

```

system ("${init}/mrtg-g_blg stop");
system ("${init}/mrtg-hosts stop");
system ("${init}/mrtg-j_blg stop");
system ("${init}/mrtg-m_idf_blg stop");
system ("${init}/mrtg-m_sr_blg stop");
system ("${init}/mrtg-remote-access stop");
system ("${init}/mrtg-routers stop");
system ("${init}/mrtg-switches stop");
system ("${init}/mrtg-temperature stop");
system ("${init}/mrtg-ups stop");
system ("${init}/mrtg-wap-clients stop");
system ("${init}/mrtg-wap-rf-int stop");

# Wipe logs
system ("${cat} $null > $logDir/abc_blg.log");
system ("${cat} $null > $logDir/bag.log");
system ("${cat} $null > $logDir/byte-o-meter.log");
system ("${cat} $null > $logDir/core.log");
system ("${cat} $null > $logDir/d_blg.log");
system ("${cat} $null > $logDir/distribution.log");
system ("${cat} $null > $logDir/dmz.log");
system ("${cat} $null > $logDir/g_blg.log");
system ("${cat} $null > $logDir/hosts.log");
system ("${cat} $null > $logDir/j_blg.log");
system ("${cat} $null > $logDir/m_idf_blg.log");
system ("${cat} $null > $logDir/m_sr_blg.log");
system ("${cat} $null > $logDir/remote-access.log");
system ("${cat} $null > $logDir/routers.log");
system ("${cat} $null > $logDir/switches.log");
system ("${cat} $null > $logDir/temperature.log");
system ("${cat} $null > $logDir/ups.log");
system ("${cat} $null > $logDir/wap_clients.log");
system ("${cat} $null > $logDir/wap_rf_int.log");

# Start MRTG processes
system ("${init}/mrtg-abc_blg start");
system ("${init}/mrtg-bag start");
system ("${init}/mrtg-byte-o-meter start");
system ("${init}/mrtg-core start");
system ("${init}/mrtg-d_blg start");
system ("${init}/mrtg-distribution start");
system ("${init}/mrtg-dmz start");
system ("${init}/mrtg-g_blg start");
system ("${init}/mrtg-hosts start");
system ("${init}/mrtg-j_blg start");
system ("${init}/mrtg-m_idf_blg start");
system ("${init}/mrtg-m_sr_blg start");
system ("${init}/mrtg-remote-access start");
system ("${init}/mrtg-routers start");
system ("${init}/mrtg-switches start");
system ("${init}/mrtg-temperature start");
system ("${init}/mrtg-ups start");
system ("${init}/mrtg-wap-clients start");
system ("${init}/mrtg-wap-rf-int start");

# Set ownership & permissions
system ("${chown} $owners $mrtgDir");
system ("${chmod} 777 $mrtgDir");
system ("${find} $mrtgDir | ${grep} rrd | ${xargs} ${chown} $owners");
system ("${find} $mrtgDir | ${grep} rrd | ${xargs} ${chmod} 664");

# Delete PNG files
`${find} $mrtgDir | ${grep} png | ${xargs} ${rm} 2>&1`;

```

TROUBLE-SHOOTING

PID Files

If MRTG won't start and is complaining about another MRTG process using that configuration, look at the PID in `/home/netops/etc/mrtg-routers.pid` and make sure there isn't another MRTG process running with that configuration file. If not, then simply remove `/home/netops/etc/mrtg-routers.pid` and try starting MRTG again.

Examine Log Files

After MRTG has been running for a few minutes, examine the contents of `/home/netops/var/log/*.log` -- look for error messages.

APPENDIX

build_octet_config.conf

```
# cc_mrtg configuration file.
#
# Format:
# group name ; [host RE ;] [host list ;] [ifref ;] [ifdesc ;] [iffilter ;] [community;] [log]
#
# The string 'group' is a keyword and must begin the line.
#
# If you want to skip on optional part, you must pad it with a ; as
# the fields are position dependent. Either (or both) host RE or
# host list must be defined.
#
# The host regular expression will be evaluated with no case sensitivity.
#
# The host list is comma delimited and the hostnames are not case sensitive.
#
# If both a host RE and list are specified, then all matches are used.
#
# The 'ifref' and 'ifdesc' fields must be chosen from nr|ip|eth|desr|name|type
# If not specified, they default to 'name' and 'alias'
#
# The 'community' field defaults to 'public'
#
# The 'iffilter' field defaults to '$if_is_ethernet or $if_is_wan' if not
# specified
#
# You can specify the log to use according to shell syntax (e.g. prepend
# >> to the log file for appending). If no redirect characters are detected
# in front of the log (>, >> or |), then &> is used. If no log is specified,
# then stdout is assumed. Be careful about sending it to stdout, as the
# output is collected via `` and can eat up a lot of memory. If you don't
# want to keep the output, use /dev/null as the log.
#
# There is a special group called default, if it is specified here,
# then all hosts that did not end up in another group gets put here.
```

```

# The hosts RE and list are ignored in the default group.  If no
# group named default is found, then the extra hosts are ignored.
#
# Examples:
# group core ; ^core- ; ; ip ; ; public; /var/tmp/core_mrtg.out
# group core ; ; core1, core2, core3 ; ; alias ; ; >> /var/tmp/core_mrtg.out

group test ; ; pit-a-esx, pit-b-esx ; ; ; ; u-guessed-it ; /home/netops/var/log/test_mrtg.out

group core ; ; core-a-rtr, core-b-rtr ; ; ; ; /home/netops/var/log/core_mrtg.out

group distribution ; ^(bag|cf|df|ga|ja|le4|md)-\w-rtr$ ; ; ; ; ;
/home/netops/var/log/distribution_mrtg.out

group dmz ; ^(gigapop|internap|dmz)-\w-rtr$|pit-\w-esx) ; crab-rtr, site-vpn ; ; ; ;
/home/netops/var/log/dmz_mrtg.out

group j_blg ; (^j\d-esx|^j4sr-\w-esx) ; j4-test-esx ; ; ; ; /home/netops/var/log/j_blg_mrtg.out

group abc_blg ; (^a\d-esx|^adsr-\w-esx|^b\d-esx|^bd-\d-esx|^c\d-esx) ; cf-esx ; ; ; ;
/home/netops/var/log/abc_blg_mrtg.out

group bag ; ((annex|cabrini|hschool|rivkin)-esx$(annex|cabrini|hschool|rivkin)-rtr$|^l[afmv]-
esx|harrison-rtr$|mmz-\w-esx) ; hkids-esx, sat-rtr ; ; ; ; /home/netops/var/log/bag_mrtg.out

group d_blg ; (^d\d-esx|^dfs-\w-esx) ; d5sr-esx, de-esx ; ; ; ;
/home/netops/var/log/d_blg_mrtg.out

group g_blg ; (^g\d-esx|^gb.*-esx|^gcore-\w-esx) ; lg-esx ; ; ; ;
/home/netops/var/log/g_blg_mrtg.out

group sixteen ; (le\d-esx) ; le5-test-esx ; ; ; ; /home/netops/var/log/sixteen_mrtg.out

##group m_blg ; (^m\d[ew]-esx|^m[de][ew]-esx|^m\dsr-\w-esx) ; ; ; ; ;
/home/netops/var/log/m_blg_mrtg.out

group m_sr_blg ; (^m[de][ew]-esx|^m\dsr-\w-esx) ; ; ; ; ;
/home/netops/var/log/m_blg_two_mrtg.out

group m_idf_blg ; (^m\d[ew]-esx) ; ; ; ; ; /home/netops/var/log/m_blg_one_mrtg.out

#group default ; ; ; ; ; ; /home/netops/var/log/default_mrtg.out

```