

Monitoring NetApp Filers with the EDDIE Tool

Chris Miles <chris@chrismiles.info>

September 2004

Introduction

Monitoring Network Appliance Filers using The EDDIE Tool [1] is very easy. SNMP is used to poll the Filers and EDDIE can make decisions based on the results of the queries. Checks can be made on the health of the Filer and alerts generated if necessary. Filer system statistics can be stored in RRD files, useful for graphing and analyzing visually. This document describes how to configure both The EDDIE Tool monitoring agent, and the Netapp Filers, to perform these tasks.

Enabling SNMP on the Filer

SNMP must first be enabled on the Filers themselves. EDDIE will use the SNMP protocol to request information from (poll) the Filers which can then be used to make decisions, such as sending alerts to humans or sending statistics to a database.

Open a command-prompt on the Filer (using telnet or ssh) and see if SNMP is enabled using options `snmp`, for example:

```
filer01> options snmp
snmp.access      legacy  (same value required in local+partner)
snmp.enable      on      (same value required in local+partner)
filer01>
```

If `snmp.enable` is set to `on` then it has already been enabled. If not, then enter `options snmp.enable on` and check the settings again. Note that if your Filers are clustered you should make exactly the same settings on the partner.

Next you should examine the SNMP settings using the `snmp` command. For example,

```
filer01> snmp
contact:

location:
    My Company
authtrap:
    0
community:
    ro public
traphost:
init:
    0
filer01>
```

If the *community* is not set to “ro public” then you should set it by using the command `snmp community add ro public`. Note that you can (and really should) use a different community string other than “public”, for better security. Any community string can be used, for example “MySecretCommunity”. Whatever community string is used needs to be remembered, as the SNMP client (EDDIE in this case) needs to know it. The community string “public” will be used for the rest of this document.

That is all that is required to enable and configure SNMP on the Filer. Remember to make the same settings to both Filers if they are clustered.

Testing SNMP access to the Filer

If you have a computer with SNMP client tools installed you can now test SNMP access to the Filer. For example, using a Redhat Linux box with `net-snmp-utils` installed, the command `snmpwalk` can be used to perform the test. The command would be entered like `snmpwalk -v 1 -c public filer01`. Remember to replace “public” with the community string that was used on the Filer. For example,

```
chris[~] snmpwalk -v 1 -c public filer01
SNMPv2-MIB::sysDescr.0 = STRING: NetApp Release 6.4.4: Fri Mar  5 10:49:23 PST 2004
SNMPv2-MIB::sysObjectID.0 = OID: SNMPv2-SMI::enterprises.789.2.3
SNMPv2-MIB::sysUpTime.0 = Timeticks: (1177035721) 136 days, 5:32:37.21
SNMPv2-MIB::sysContact.0 = STRING:
SNMPv2-MIB::sysName.0 = STRING: filer01
SNMPv2-MIB::sysLocation.0 = STRING: My Company
SNMPv2-MIB::sysServices.0 = INTEGER: 72
IF-MIB::ifNumber.0 = INTEGER: 8
IF-MIB::ifIndex.1 = INTEGER: 1
IF-MIB::ifIndex.2 = INTEGER: 2
IF-MIB::ifIndex.3 = INTEGER: 3
```

and so on (there would be a lot more output than this). If the test works fine then the Filer is configured properly and it is time to create EDDIE rules. If the test fails, the Filer settings should be checked again. The problem could also be caused by the test client or other elements of the network (such as firewalls) so some investigation may be required (e.g. can the client poll other SNMP-enabled devices?).

EDDIE rules for SNMP checks

The first check that is recommended is to verify that the Filer is visible on the network. This can easily be done using the PING directive. Obviously, if this check fails, then something significant is wrong. Assuming no other network problems (which EDDIE could also check for) perhaps the Filer has crashed, has been switched off, or has been unplugged from the network. In any case, this check will be used as a dependency for the others, which will prevent the SNMP checks from polling the Filer if it is not reachable on the network.

The EDDIE directive to perform the ping check would be like:

```
PING filer01_ping:
  host='filer01'
  numpings=10
```

```

rule='not alive'
action=COMMONALERT(commonmsg.ping,2)
act2ok=COMMONFIXED(commonmsg.ping,2)

```

The next directive uses SNMP to check the Filer's *Global Status*. This is an indication of the global health of the Filer, and will normally return "The system's global status is normal. " If the global status returns anything else, the rule will evaluate to true and the action will be called (i.e., sending an alert). The global status will be changed whenever something is wrong with the Filer (e.g., disk broken, filesystem full, cluster failover, etc).

```

SNMP filer01_global_status:
  host='filer01'
  oid='1.3.6.1.4.1.789.1.2.2.25.0'
  community='public'
  maxretry=10
  rule='not failed and response != "The system\'s global status is normal. "'
  checkdependson='filer01_ping'
  action=email( ALERT_EMAIL, "%(host)s status: %(response)s", \
    "Netapp %(host)s global status has changed to: %(response)s" )
  act2ok=email( ALERT_EMAIL, "%(host)s status OK", \
    "Netapp %(host)s global status is now OK: %(response)s" )

```

This simple directive is enough to monitor the Filer for any major problems. Thanks to the informative messages returned by this query, if there are any problems with the Filer, the messages are usually enough to indicate exactly what that problem is. The action, in this case, is simply to email that response to the relevant people.

For more detailed monitoring the LOGSCAN directive can be used to watch the Filer's logs

```

LOGSCAN filer01_messages_error:
  file='/netapp/filer01/etc/messages'
  regex=':warning|:error|:CRITICAL'
  action=email(ALERT_EMAIL, "%(file)s", "%(lines)s")

```

This directive needs to be run on a host that is allowed to mount the Filer's root volume. This would usually be the Filer's "admin" host. It uses a simple regular expression to match any messages that are logged with either "warning", "error" or "CRITICAL" states.

Gathering Filer statistics

Filer statistics can be collected and stored for further analysis or graphing. Useful statistics to store are: CPU load; NFS/CIFS/HTTP ops per second; volume and snapshot space used; network/disk/tape throughput; and cache age.

SNMP directives are used to fetch the statistics and actions are used to store the data in some way. The examples below use the elvinrrd action to send the statistics over the Elvin messaging network [2] to an elvinrrd [3] listener. elvinrrd stores the statistics in RRD files to be graphed by other applications.

The directives are shown below, followed by examples of the graphs that can be produced by rrdtool.

```

SNMP filer01_cpu_rrd:
  host='filer01'
  oid='1.3.6.1.4.1.789.1.2.1.2.0, 1.3.6.1.4.1.789.1.2.1.4.0'
  community='public'
  maxretry=10
  rule='not failed'
  scanperiod='1m'
  checkdependson='filer01_ping'
  action=elvinrrd("cpunetapp-%(host)s", "cpu_busy=%(response1)s", "cpu_idle=%(response2)s")

SNMP filer01_vol0_rrd:
  host='filer01'
  oid='1.3.6.1.4.1.789.1.5.4.1.4.1, 1.3.6.1.4.1.789.1.5.4.1.3.1'
  community='public'
  maxretry=10
  rule='not failed'
  scanperiod='5m'
  checkdependson='filer01_ping'
  action=elvinrrd("fs-%(host)s_vol0", "used=%(response1)s", "size=%(response2)s")

SNMP filer01_vol0snap_rrd:
  host='filer01'
  oid='1.3.6.1.4.1.789.1.5.4.1.4.2, 1.3.6.1.4.1.789.1.5.4.1.3.2'
  community='public'
  maxretry=10
  rule='not failed'
  scanperiod='5m'
  checkdependson='filer01_ping'
  action=elvinrrd("fs-%(host)s_vol0snap", "used=%(response1)s", "size=%(response2)s")

SNMP filer01_volweb01_rrd:
  host='filer01'
  oid='1.3.6.1.4.1.789.1.5.4.1.4.3, 1.3.6.1.4.1.789.1.5.4.1.3.3'
  community='public'
  maxretry=10
  rule='not failed'
  scanperiod='5m'
  checkdependson='filer01_ping'
  action=elvinrrd("fs-%(host)s_volweb01", "used=%(response1)s", "size=%(response2)s")

SNMP filer01_volweb01snap_rrd:
  host='filer01'
  oid='1.3.6.1.4.1.789.1.5.4.1.4.4, 1.3.6.1.4.1.789.1.5.4.1.3.4'
  community='public'
  maxretry=10
  rule='not failed'
  scanperiod='5m'
  checkdependson='filer01_ping'
  action=elvinrrd("fs-%(host)s_volweb01snap", "used=%(response1)s", "size=%(response2)s")

SNMP filer01_nfsops_rrd:
  host='filer01'
  # enterprises.netapp.netapp1.sysStat.misc.miscHighNfsOps.0
  # enterprises.netapp.netapp1.sysStat.misc.miscLowNfsOps.0
  oid='1.3.6.1.4.1.789.1.2.2.5.0:1.3.6.1.4.1.789.1.2.2.6.0'
  community='public'
  maxretry=10
  rule='not failed'
  scanperiod='5m'
  checkdependson='filer01_ping'
  action=elvinrrd("nfsops-%(host)s", "nfsops=%(response)s")

```

```

SNMP filer01_cifsops_rrd:
    host='filer01'
    # enterprises.netapp.netapp1.sysStat.misc.miscHighCifsOps.0
    # enterprises.netapp.netapp1.sysStat.misc.miscLowCifsOps.0
    oid='1.3.6.1.4.1.789.1.2.2.7.0:1.3.6.1.4.1.789.1.2.2.8.0'
    community='public'
    maxretry=10
    rule='not failed'
    scanperiod='5m'
    checkdependson='filer01_ping'
    action=elvinrrd("cifsops-%(host)s", "cifsops=%(response)s")

SNMP filer01_httppops_rrd:
    host='filer01'
    # enterprises.netapp.netapp1.sysStat.misc.miscHighHttpOps.0
    # enterprises.netapp.netapp1.sysStat.misc.miscLowHttpOps.0
    oid='1.3.6.1.4.1.789.1.2.2.9.0:1.3.6.1.4.1.789.1.2.2.10.0'
    community='public'
    maxretry=10
    rule='not failed'
    scanperiod='5m'
    checkdependson='filer01_ping'
    action=elvinrrd("httppops-%(host)s", "httppops=%(response)s")

SNMP filer01_netbytes_rrd:
    host='filer01'
    # enterprises.netapp.netapp1.sysStat.misc.miscHighNetRcvdBytes.0
    # enterprises.netapp.netapp1.sysStat.misc.miscLowNetRcvdBytes.0
    # enterprises.netapp.netapp1.sysStat.misc.miscHighNetSentBytes.0
    # enterprises.netapp.netapp1.sysStat.misc.miscLowNetSentBytes.0
    oid='1.3.6.1.4.1.789.1.2.2.11.0:1.3.6.1.4.1.789.1.2.2.12.0, \
        1.3.6.1.4.1.789.1.2.2.13.0:1.3.6.1.4.1.789.1.2.2.14.0'
    community='public'
    maxretry=10
    rule='not failed'
    scanperiod='5m'
    checkdependson='filer01_ping'
    action=elvinrrd("net-%(host)s_total", "ibytes=%(response1)s", "obytes=%(response2)s")

SNMP filer01_diskbytes_rrd:
    host='filer01'
    # enterprises.netapp.netapp1.sysStat.misc.miscHighDiskReadBytes.0
    # enterprises.netapp.netapp1.sysStat.misc.miscLowDiskReadBytes.0
    # enterprises.netapp.netapp1.sysStat.misc.miscHighDiskWriteBytes.0
    # enterprises.netapp.netapp1.sysStat.misc.miscLowDiskWriteBytes.0
    oid='1.3.6.1.4.1.789.1.2.2.15.0:1.3.6.1.4.1.789.1.2.2.16.0, \
        1.3.6.1.4.1.789.1.2.2.17.0:1.3.6.1.4.1.789.1.2.2.18.0'
    community='public'
    maxretry=10
    rule='not failed'
    scanperiod='5m'
    checkdependson='filer01_ping'
    action=elvinrrd("disk-%(host)s_disk", "rbytes=%(response1)s", "wbytes=%(response2)s")

SNMP filer01_tapebytes_rrd:
    host='filer01'
    # enterprises.netapp.netapp1.sysStat.misc.miscHighTapeReadBytes.0
    # enterprises.netapp.netapp1.sysStat.misc.miscLowTapeReadBytes.0
    # enterprises.netapp.netapp1.sysStat.misc.miscHighTapeWriteBytes.0
    # enterprises.netapp.netapp1.sysStat.misc.miscLowTapeWriteBytes.0
    oid='1.3.6.1.4.1.789.1.2.2.19.0:1.3.6.1.4.1.789.1.2.2.20.0, \
        1.3.6.1.4.1.789.1.2.2.21.0:1.3.6.1.4.1.789.1.2.2.22.0'

```

```

community='public'
maxretry=10
rule='not failed'
scanperiod='5m'
checkdependson='filer01_ping'
action=elvinrrd("tape-%(host)s_tape", "rbytes=%(response1)s", "wbytes=%(response2)s")

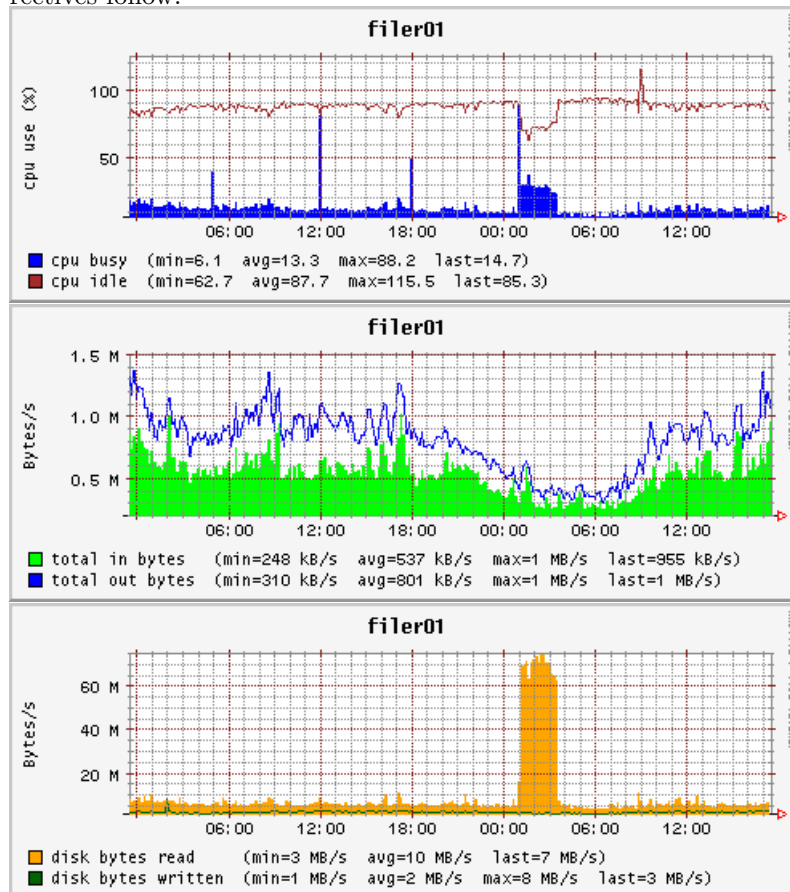
```

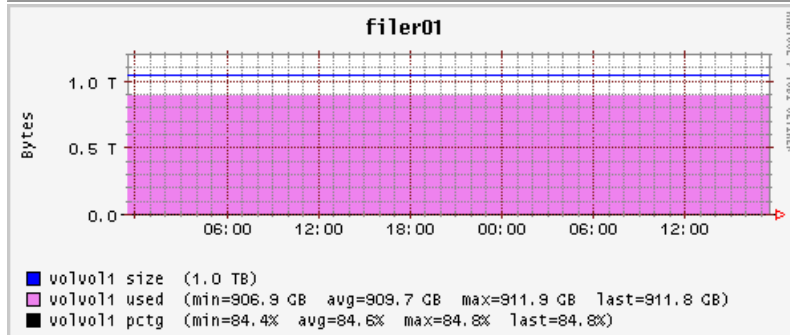
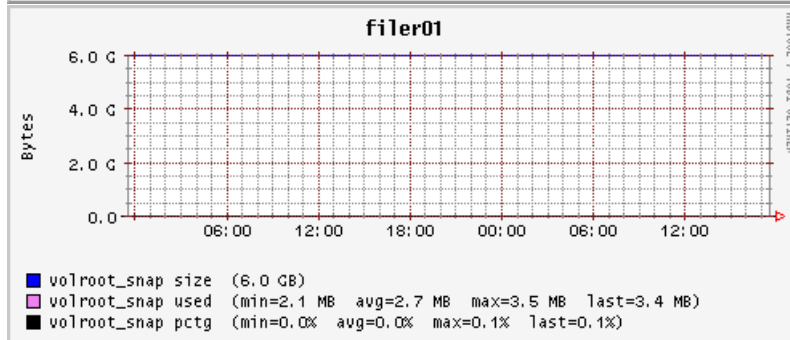
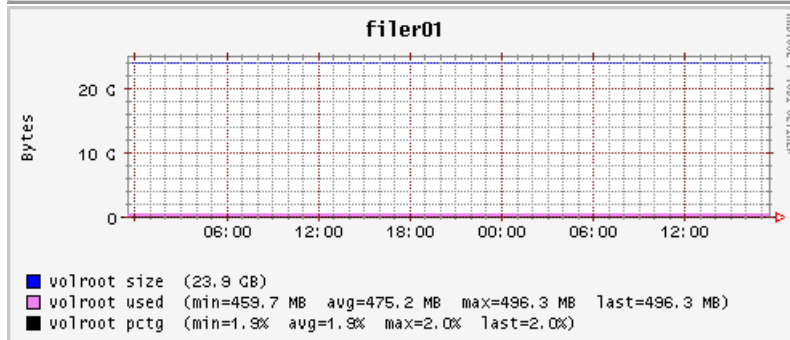
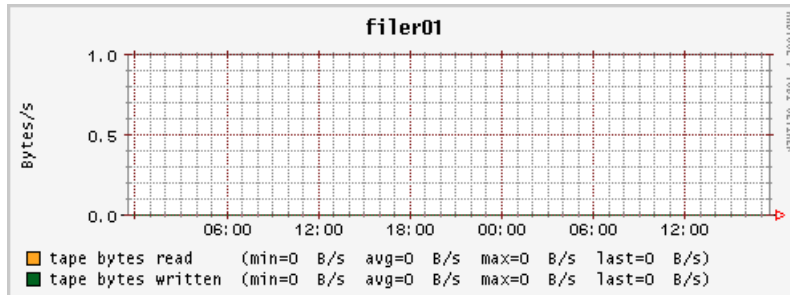
```

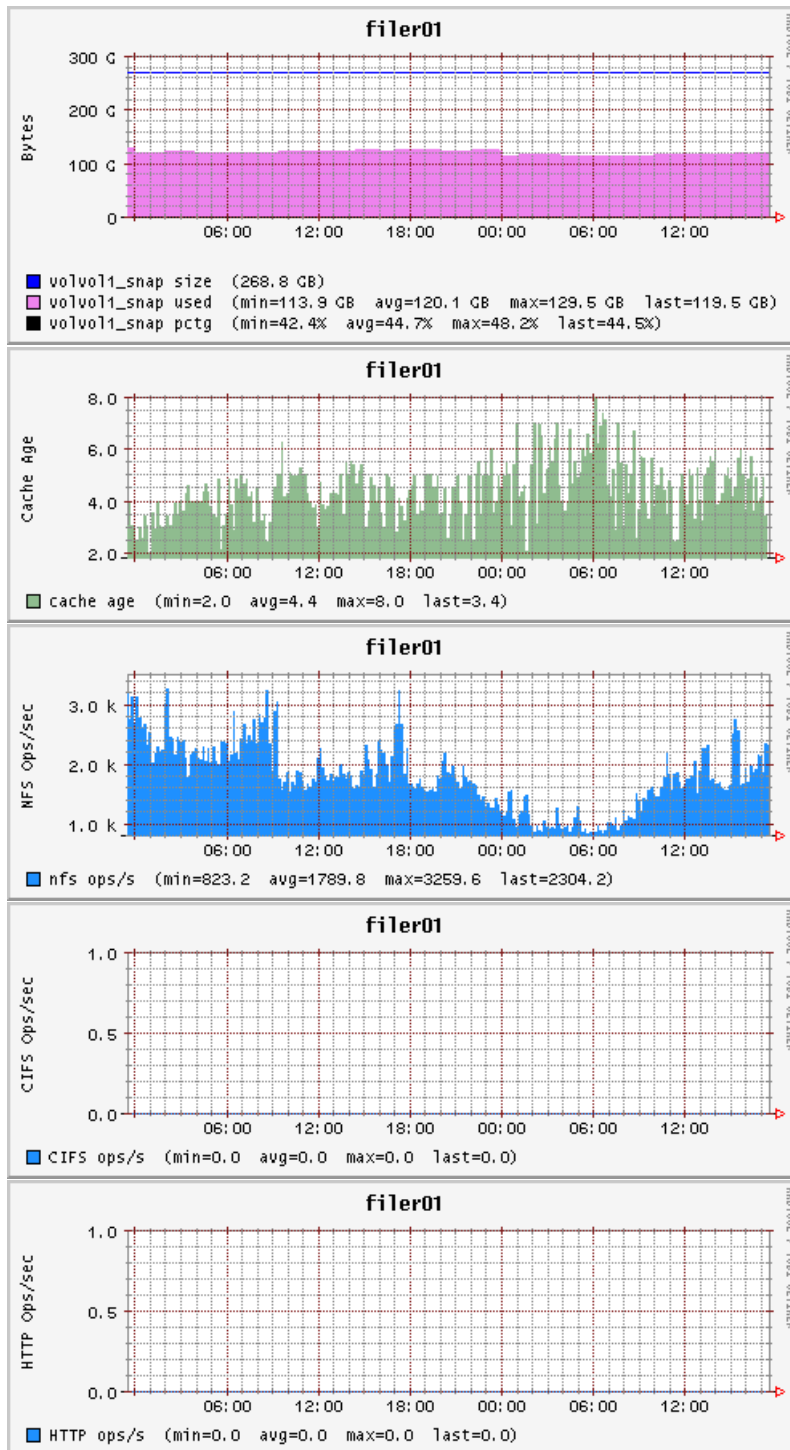
SNMP filer01_cacheage_rrd:
host='filer01'
# enterprises.netapp.netapp1.sysStat.misc.miscCacheAge.0
oid='1.3.6.1.4.1.789.1.2.2.23.0'
community='public'
maxretry=10
rule='not failed'
scanperiod='5m'
checkdependson='filer01_ping'
action=elvinrrd("netappcacheage-%(host)s", "cacheage=%(response)s")

```

Examples of rrdtool graphs produced using data collected by the above directives follow.







Setting up Elvin messaging and the elvinrrd listener is beyond the scope of this document but they do provide a very convenient means of storing and graphically displaying system statistics for trend analysis.

Conclusion

EDDIE can be very easily used to monitor Netapp Filers using its ping, SNMP polling and log file scanning capabilities. Filer system statistics can also be collected and stored for later analysis, including using rrdtool to graph the data.

Appendix A - Configs for elvinrrd

The relevant configuration entries for an elvinrrd listener to receive and store Netapp Filer stats are given below.

```
# Netapp CPU [1min]
elvinrrd=cpunetapp-*
rrdfile=/export/rrd/cpunetapp/cpunetapp-*.rrd
store=cpu_busy,cpu_idle
create=-s 60 DS:cpu_busy:COUNTER:120:U:1000 DS:cpu_idle:COUNTER:120:U:1000 \
  RRA:AVERAGE:0.5:1:1440 RRA:AVERAGE:0.5:10:1008 RRA:AVERAGE:0.5:60:720 \
  RRA:AVERAGE:0.5:480:1095

# Filesystems [5min]
elvinrrd=fs-*
rrdfile=/export/rrd/fs/fs-*.rrd
store=used,size
create=-s 300 DS:used:GAUGE:600:0:U DS:size:GAUGE:600:0:U RRA:AVERAGE:0.5:1:600 \
  RRA:AVERAGE:0.5:6:600 RRA:AVERAGE:0.5:24:600 RRA:AVERAGE:0.5:288:600 \
  RRA:MAX:0.5:24:600 RRA:MAX:0.5:288:600

# NFS ops/sec [5min]
elvinrrd=nfsops-*
rrdfile=/export/rrd/nfsops/nfsops-*.rrd
store=nfsops
create=-s 300 DS:nfsops:COUNTER:600:0:100000 RRA:AVERAGE:0.5:1:600 RRA:AVERAGE:0.5:6:600 \
  RRA:AVERAGE:0.5:24:600 RRA:AVERAGE:0.5:288:600 RRA:MAX:0.5:24:600 RRA:MAX:0.5:288:600

# CIFS ops/sec [5min]
elvinrrd=cifsops-*
rrdfile=/export/rrd/cifsops/cifsops-*.rrd
store=cifsops
create=-s 300 DS:cifsops:COUNTER:600:0:100000 RRA:AVERAGE:0.5:1:600 RRA:AVERAGE:0.5:6:600 \
  RRA:AVERAGE:0.5:24:600 RRA:AVERAGE:0.5:288:600 RRA:MAX:0.5:24:600 RRA:MAX:0.5:288:600

# HTTP ops/sec [5min]
elvinrrd=httpops-*
rrdfile=/export/rrd/httpops/httpops-*.rrd
store=httpops
create=-s 300 DS:httpops:COUNTER:600:0:100000 RRA:AVERAGE:0.5:1:600 RRA:AVERAGE:0.5:6:600 \
  RRA:AVERAGE:0.5:24:600 RRA:AVERAGE:0.5:288:600 RRA:MAX:0.5:24:600 RRA:MAX:0.5:288:600

# network throughput [5min]
elvinrrd=net-*
rrdfile=/export/rrd/net/net-*.rrd
store=ibytes,obytes
create=-s 300 DS:ibytes:COUNTER:600:0:U DS:obytes:COUNTER:600:0:U RRA:AVERAGE:0.5:1:600 \
  RRA:AVERAGE:0.5:6:600 RRA:AVERAGE:0.5:24:600 RRA:AVERAGE:0.5:288:600 RRA:MAX:0.5:24:600 \
  RRA:MAX:0.5:288:600

# disk throughput (bytes) [5min]
elvinrrd=disk-*
```

```

rrdfile=/export/rrd/disk/disk-*.rrd
store=rbytes,wbytes
create--s 300 DS:rbytes:COUNTER:600:0:U DS:wbytes:COUNTER:600:0:U RRA:AVERAGE:0.5:1:600 \
RRA:AVERAGE:0.5:6:600 RRA:AVERAGE:0.5:24:600 RRA:AVERAGE:0.5:288:600 RRA:MAX:0.5:24:600 \
RRA:MAX:0.5:288:600

# tape throughput (bytes) [5min]
elvinrrd=tape-*
rrdfile=/export/rrd/tape/tape-*.rrd
store=rbytes,wbytes
create--s 300 DS:rbytes:COUNTER:600:0:U DS:wbytes:COUNTER:600:0:U RRA:AVERAGE:0.5:1:600 \
RRA:AVERAGE:0.5:6:600 RRA:AVERAGE:0.5:24:600 RRA:AVERAGE:0.5:288:600 \
RRA:MAX:0.5:24:600 RRA:MAX:0.5:288:600

# Netapp Cache Age [5min]
elvinrrd=netappcacheage-*
rrdfile=/export/rrd/netappcacheage/netappcacheage-*.rrd
store=cacheage
create--s 300 DS:cacheage:GAUGE:600:0:10000 RRA:AVERAGE:0.5:1:600 RRA:AVERAGE:0.5:6:600 \
RRA:AVERAGE:0.5:24:600 RRA:AVERAGE:0.5:288:600 RRA:MAX:0.5:24:600 RRA:MAX:0.5:288:600

```

Appendix B - Configs for eddiebrowser.cgi

One way to generate graphs of the RRD data that look like those shown in this document is to use the eddiebrowser.cgi interface.

Configurations are given below for using this interface to generate the RRD graphs.

cpunetapp.cfg:

```

NAME=CPU Usage
GROUP=System
FILES=cpunetapp-(?P<hostname>[a-zA-Z0-9._-]+)\.rrd

```

```

GRAPH_TITLE=%(hostname)s
GRAPH_VERTICAL_LABEL=cpu use (%)
GRAPH_DEFS=A,B

```

```

GRAPH_LABEL_A=cpu busy
GRAPH_TYPE_A=AREA
GRAPH_COLOR_A=0000FF
GRAPH_SOURCE_A=cpu_busy
GRAPH_SOURCETYPE_A=AVERAGE
GRAPH_GPRINT_A1=MIN:(min=%.11f
GRAPH_GPRINT_A2=AVERAGE:avg=%.11f
GRAPH_GPRINT_A3=MAX:max=%.11f
GRAPH_GPRINT_A4=LAST:last=%.11f)

```

```

GRAPH_LABEL_B=cpu idle
GRAPH_TYPE_B=LINE1
GRAPH_COLOR_B=A52A2A
GRAPH_SOURCE_B=cpu_idle
GRAPH_SOURCETYPE_B=AVERAGE
GRAPH_GPRINT_B1=MIN:(min=%.11f
GRAPH_GPRINT_B2=AVERAGE:avg=%.11f
GRAPH_GPRINT_B3=MAX:max=%.11f
GRAPH_GPRINT_B4=LAST:last=%.11f)

```

fs.cfg:

```

NAME=Filesystem Usage
GROUP=Disk

```

```
FILES=fs-(?P<hostname>[a-zA-Z0-9.-]+)(?P<fsname>[a-zA-Z0-9._]+)\.rrd
```

```
GRAPH_TITLE=(hostname)s  
GRAPH_VERTICAL_LABEL=Bytes  
GRAPH_DEFS=A,B,C,D,E,F  
GRAPH_BASE=1024
```

```
GRAPH_SOURCE_A=used  
GRAPH_SOURCETYPE_A=AVERAGE
```

```
GRAPH_SOURCE_B=size  
GRAPH_SOURCETYPE_B=AVERAGE
```

```
GRAPH_LABEL_C=  
GRAPH_TYPE_C=LINE1  
GRAPH_COLOR_C=000000  
GRAPH_RPN_C=A,A,-
```

```
GRAPH_LABEL_D=(fsname)s size  
GRAPH_TYPE_D=LINE1  
GRAPH_COLOR_D=0000FF  
GRAPH_RPN_D=B,1024,*  
GRAPH_GPRINT_D1=LAST:(%.11f %sB)
```

```
GRAPH_LABEL_E=(fsname)s used  
GRAPH_TYPE_E=AREA  
GRAPH_COLOR_E=EE82EE  
GRAPH_RPN_E=A,1024,*  
GRAPH_GPRINT_E1=MIN:(min=%.11f %sB  
GRAPH_GPRINT_E2=AVERAGE:avg=%.11f %sB  
GRAPH_GPRINT_E3=MAX:max=%.11f %sB  
GRAPH_GPRINT_E4=LAST:last=%.11f %sB)
```

```
GRAPH_LABEL_F=(fsname)s pctg  
GRAPH_TYPE_F=LINE1  
GRAPH_COLOR_F=  
GRAPH_RPN_F=100,A,B,/,*  
GRAPH_GPRINT_F1=MIN:(min=%.11f%%  
GRAPH_GPRINT_F2=AVERAGE:avg=%.11f%%  
GRAPH_GPRINT_F3=MAX:max=%.11f%%  
GRAPH_GPRINT_F4=LAST:last=%.11f%%)
```

nfsops.cfg:

```
NAME=NFS Ops  
GROUP=NFS  
FILES=nfsops-(?P<hostname>[a-zA-Z0-9.-]+)\.rrd
```

```
GRAPH_TITLE=(hostname)s  
GRAPH_VERTICAL_LABEL=NFS Ops/sec  
GRAPH_DEFS=A
```

```
GRAPH_LABEL_A=nfs ops/s  
GRAPH_TYPE_A=AREA  
GRAPH_COLOR_A=1E90FF  
GRAPH_SOURCE_A=nfsops  
GRAPH_SOURCETYPE_A=AVERAGE  
GRAPH_GPRINT_A1=MIN:(min=%.11f  
GRAPH_GPRINT_A2=AVERAGE:avg=%.11f  
GRAPH_GPRINT_A3=MAX:max=%.11f  
GRAPH_GPRINT_A4=LAST:last=%.11f)
```

cifsops.cfg:

```
NAME=CIFS Ops
GROUP=Netapp
FILES=cifsops-(?P<hostname>[a-zA-Z0-9._-]+).rrd
```

```
GRAPH_TITLE=%(hostname)s
GRAPH_VERTICAL_LABEL=CIFS Ops/sec
GRAPH_DEFS=A
```

```
GRAPH_LABEL_A=CIFS ops/s
GRAPH_TYPE_A=AREA
GRAPH_COLOR_A=1E90FF
GRAPH_SOURCE_A=cifsops
GRAPH_SOURCETYPE_A=AVERAGE
GRAPH_GPRINT_A1=MIN:(min=%.11f
GRAPH_GPRINT_A2=AVERAGE:avg=%.11f
GRAPH_GPRINT_A3=MAX:max=%.11f
GRAPH_GPRINT_A4=LAST:last=%.11f)
```

httpops.cfg:

```
NAME=HTTP Ops
GROUP=Netapp
FILES=httpops-(?P<hostname>[a-zA-Z0-9._-]+).rrd
```

```
GRAPH_TITLE=%(hostname)s
GRAPH_VERTICAL_LABEL=HTTP Ops/sec
GRAPH_DEFS=A
```

```
GRAPH_LABEL_A=HTTP ops/s
GRAPH_TYPE_A=AREA
GRAPH_COLOR_A=1E90FF
GRAPH_SOURCE_A=httpops
GRAPH_SOURCETYPE_A=AVERAGE
GRAPH_GPRINT_A1=MIN:(min=%.11f
GRAPH_GPRINT_A2=AVERAGE:avg=%.11f
GRAPH_GPRINT_A3=MAX:max=%.11f
GRAPH_GPRINT_A4=LAST:last=%.11f)
```

net.cfg:

```
NAME=Network Throughput
GROUP=Network
FILES=net-(?P<hostname>[a-zA-Z0-9._-]+)(?P<ifname>\w+).rrd
```

```
GRAPH_TITLE=%(hostname)s
GRAPH_VERTICAL_LABEL=Bytes/s
GRAPH_DEFS=A,B
```

```
GRAPH_LABEL_A=%(ifname)s in bytes
GRAPH_TYPE_A=AREA
GRAPH_COLOR_A=00FF00
GRAPH_SOURCE_A=ibytes
GRAPH_SOURCETYPE_A=AVERAGE
GRAPH_GPRINT_A1=MIN:(min=%.01f %sB/s
GRAPH_GPRINT_A2=AVERAGE:avg=%.01f %sB/s
GRAPH_GPRINT_A3=MAX:max=%.01f %sB/s
GRAPH_GPRINT_A4=LAST:last=%.01f %sB/s)
```

```
GRAPH_LABEL_B=%(ifname)s out bytes
GRAPH_TYPE_B=LINE1
GRAPH_COLOR_B=0000FF
GRAPH_SOURCE_B=obytes
GRAPH_SOURCETYPE_B=AVERAGE
GRAPH_GPRINT_B1=MIN:(min=%.01f %sB/s
```

```
GRAPH_GPRINT_B2=AVERAGE:avg=%.01f %sB/s
GRAPH_GPRINT_B3=MAX:max=%.01f %sB/s
GRAPH_GPRINT_B4=LAST:last=%.01f %sB/s)
```

disk.cfg:

```
NAME=Disk Throughput
GROUP=Disk
FILES=disk-(?P<hostname>[a-zA-Z0-9.-]+)(?P<diskname>\w+).rrd
```

```
GRAPH_TITLE=%(hostname)s
GRAPH_VERTICAL_LABEL=Bytes/s
GRAPH_DEFS=A,B
```

```
GRAPH_LABEL_A=%(diskname)s bytes read
GRAPH_TYPE_A=AREA
GRAPH_COLOR_A=FFA500
GRAPH_SOURCE_A=rbytes
GRAPH_SOURCETYPE_A=AVERAGE
GRAPH_GPRINT_A1=MIN: (min=%.01f %sB/s
GRAPH_GPRINT_A2=AVERAGE:avg=%.01f %sB/s
GRAPH_GPRINT_A3=MAX:max=%.01f %sB/s
GRAPH_GPRINT_A3=LAST:last=%.01f %sB/s)
```

```
GRAPH_LABEL_B=%(diskname)s bytes written
GRAPH_TYPE_B=LINE1
GRAPH_COLOR_B=006400
GRAPH_SOURCE_B=wbytes
GRAPH_SOURCETYPE_B=AVERAGE
GRAPH_GPRINT_B1=MIN: (min=%.01f %sB/s
GRAPH_GPRINT_B2=AVERAGE:avg=%.01f %sB/s
GRAPH_GPRINT_B3=MAX:max=%.01f %sB/s
GRAPH_GPRINT_B4=LAST:last=%.01f %sB/s)
```

tape.cfg:

```
NAME=Tape Throughput
GROUP=Tape
FILES=tape-(?P<hostname>[a-zA-Z0-9.-]+)(?P<diskname>\w+).rrd
```

```
GRAPH_TITLE=%(hostname)s
GRAPH_VERTICAL_LABEL=Bytes/s
GRAPH_DEFS=A,B
```

```
GRAPH_LABEL_A=%(diskname)s bytes read
GRAPH_TYPE_A=AREA
GRAPH_COLOR_A=FFA520
GRAPH_SOURCE_A=rbytes
GRAPH_SOURCETYPE_A=AVERAGE
GRAPH_GPRINT_A1=MIN: (min=%.01f %sB/s
GRAPH_GPRINT_A2=AVERAGE:avg=%.01f %sB/s
GRAPH_GPRINT_A3=MAX:max=%.01f %sB/s
GRAPH_GPRINT_A4=LAST:last=%.01f %sB/s)
```

```
GRAPH_LABEL_B=%(diskname)s bytes written
GRAPH_TYPE_B=LINE1
GRAPH_COLOR_B=006420
GRAPH_SOURCE_B=wbytes
GRAPH_SOURCETYPE_B=AVERAGE
GRAPH_GPRINT_B1=MIN: (min=%.01f %sB/s
GRAPH_GPRINT_B2=AVERAGE:avg=%.01f %sB/s
GRAPH_GPRINT_B3=MAX:max=%.01f %sB/s
GRAPH_GPRINT_B4=LAST:last=%.01f %sB/s)
```

netappcacheage.cfg:

```
NAME=Cache Age
GROUP=Netapp
FILES=netappcacheage-(?P<hostname>[a-zA-Z0-9._-]+).rrd
```

```
GRAPH_TITLE=%(hostname)s
GRAPH_VERTICAL_LABEL=Cache Age
GRAPH_DEFS=A
```

```
GRAPH_LABEL_A=cache age
GRAPH_TYPE_A=AREA
GRAPH_COLOR_A=8FBC8F
GRAPH_SOURCE_A=cacheage
GRAPH_SOURCETYPE_A=AVERAGE
GRAPH_GPRINT_A1=MIN:(min=%.11f
GRAPH_GPRINT_A2=AVERAGE:avg=%.11f
GRAPH_GPRINT_A3=MAX:max=%.11f
GRAPH_GPRINT_A4=LAST:last=%.11f)
```

References

- [1] <http://eddie-tool.net/>
- [2] <http://elvin.dstc.edu.au/>
- [3] <http://www.psychofx.com/elvinrrd/>