The OpenNMS Group, Inc.

WHITE PAPER

Configuring an OpenNMS Stand-by Server

Version 1.2

The OpenNMS Group, Inc. 220 Chatham Business Drive Pittsboro, NC 27312 **T** +1 (919) 533-0160 **F** +1 (503) 961-7746 david@opennms.com
URL: http://blogs.opennms.org/david

Executive Summary

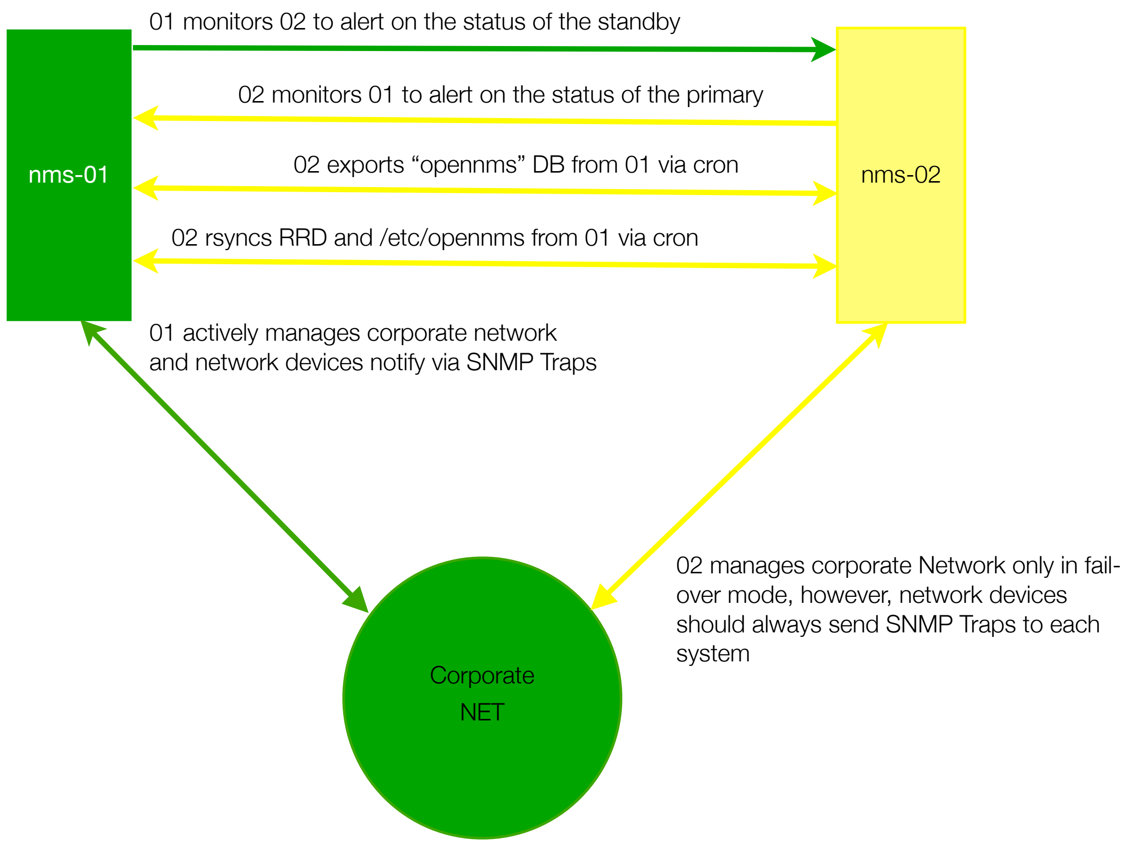
A large corporation contacted the OpenNMS Group for a consulting engagement requesting assistance with their new network and system management platform, OpenNMS. This engagement was scheduled for 3 days with the primary focus on developing and implementing a continuity of operations plan in the event of a system failure or localized disaster impacting OpenNMS.

Continuity of Operations Plan

This plan includes automated and manual instructions for creating, maintaining, and implementing two OpenNMS instances in primary, standby, and failover modes. For ease of communication we'll call this the HA configuration.

The HA configuration concepts

The OpenNMS software and software dependencies (i.e. PostgreSQL) will be installed on 2 geographically remote Debian Linux 4.0 servers: nms-01 and nms-02. The nms-01 server is the primary system that is responsible for monitoring the corporate network and the nms-02 server is the backup system that, when in stand-by mode, will be monitoring nms-01 and configured to notify operations staff when a failover should be initiated. The nms-02 server will monitor the corporate network in the failover mode.


01 monitors 02 to alert on the status of the standby
02 monitors 01 to alert on the status of the primary
02 exports "opennms" DB from 01 via cron
02 rsyncs RRD and /etc/opennms from 01 via cron
nms-01
nms-02
01 actively manages corporate network and network devices notify via SNMP Traps
02 manages corporate Network only in fail-over mode, however, network devices should always send SNMP Traps to each system
Corporate NET

There is a monitoring configuration on nms-02 and two main scripts located on nms-02 that support this configuration.

Script: sync.sh

This script is responsible for maintaining consistency of the live nms-01 OpenNMS instance with the stand-by OpenNMS' failover configuration resident on nms-02. This script is designed to be ran as a cron job and the crontab schedule should be maintained in the sync-on.crontab file in the $SYNC_HOME/etc/ directory and the on/off state is maintained by the sync-state.sh script found in the $SYNC_HOME/scripts directory.

(Note: This synchronization can be simplified tremendously if a shared file system can be mounted from a SAN and a DB server or DB server cluster is in place. The synchronization in this document assumes geographically separate and stand-alone OpenNMS instances.)

Phase 1 (Database Sync):

The script is responsible for synchronizing the OpenNMS database and configuration files on nms-01 without effecting nms-02's stand-by mission of only monitoring nms-01. This script is executed on a schedule defined in a crontab and is located on the file system in the ~opennms/scripts/failover directory. This script exports the live "opennms" database instance running on nms-01 to the file system on nms-02 and then imports that data to a PostgreSQL database instance called "opennms_failover" running on nms-02. This instance name is used to not conflict with the active 'opennms" instance that is used by OpenNMS in standby mode to only monitor nms-01.

Phase 2 (Configuration Sync):

The script is responsible for synchronizing the OpenNMS configuration files on nms-01 for use during failover mode of operation. The configuration files in $OPENNMS_ETC located on nms-01 are rsync'd, via and ssh tunnel, to $OPENNMS_ETC/opennms-failover directory on nms-02.

Phase 3 (RRD Repository Sync):

The script is responsible for synchronizing the OpenNMS RRD repository on nms-01 for use on nms-02 during failover. The RRD files in /var/lib/opennms/rrd located on nms-01 are rsync'd, via and ssh tunnel, to /var/lib/opennms/rrd-failover/ nms-02.

Script: failover.sh

This script is responsible for changing the state of OpenNMS running on nms-02 from "standby" mode to "failover" mode. The script requires that either "start" or "stop" is passed as a parameter. The parameter "start" stops the automatic synchronization and initiates the failover sequence causing the standby OpenNMS instance to begin actively monitoring the corporate network. The "stop" parameter deactivates the failover mode instance of OpenNMS and resumes OpenNMS in the standby mode of monitoring the nms-01 OpenNMS instance and reschedules the automatic synchronization.

This script when passed the "start" parameter:

- Stops OpenNMS on nms-02

- Changes symbolic links /etc/opennms and /var/lib/opennms/rrd to point to the respective failover directories created and synchronized by the sync.sh script so that OpenNMS configuration files synchronized from nms-01 will operate properly.
- Copies a special opennms-datasources.xml file to /etc/opennms/ (this file causes OpenNMS to use the "opennms_failover" database in PostgreSQL instance instead of the default "opennms" database)
- Starts OpenNMS

This script when passed the "stop" parameter restores the symbolic links and the opennms-datasources.xml file and resumes OpenNMS into "standby mode".

Script: sync-state.sh

This script is used to control the state of the automatic synchronization maintained in the system crontab of the $SYNC_USER on nms-02. Change the sync-on.crontab file in the $SYNC_HOME/etc/ directory to define the schedule.

HA Operational Modes:

For the nms-02 server, the OpenNMS instance's normal state is to operate in "standby mode". While in stand-by mode, OpenNMS is configured to monitor only nms-01 server via 2 of its services: the OpenNMS Web application (OpenNMSWebApp) and the ICMP service of the nms-01 ethernet interface.

When nms-02 detects that the nms-01 node is down (all services being monitored are down), it should generate a notification to an appropriate destination path. An OpenNMS administrator should take the following steps to begin the "failover mode" of operation on nms-02:

- ✓ Verify the nms-01 is actually down and not monitoring the corporate network
- ✓ Disable synchronization of nms-01 from nms-02 via the opennms user's crontab on nms-02
- ✓ Run the failover.sh script on nms-02 with the "start" parameter

When nms-01 is recovered, the OpenNMS administrator should take the following steps to resume the "stand-by mode" of operation on nsm-02:

- ✓ Run the failover.sh script on nms-02 with the "stop" parameter
- ✓ Re-nable synchronization of nms-01 from nms-02 via the opennms user's crontab

Required Configuration

- Access control to PostgreSQL on nms-01 to allow access from nms-02 (pg_hba.conf and iptables)
- The "rsync" package installed on both nms-01 and nms-02
- A synchronization user (SYNC_USER) account be created on both NMSs as a "best practice" as opposed to using root access and OpenNMS configuration and RRD structures changed to be owned by this user.
- The "sudo" package should be installed and/or configured on both NMSs to provide the SYNC_USER account root privileges when needed during failover.
- A pass-phrase-less RSA key to be used by the SYNC_USER and the sync.sh script. This is created so that the SYNC_USER user account on nms-02 could rysnc via an ssh tunnel without interaction.

NOTE*: The private key stored in the ~opennms/.ssh directory should be strictly controlled. As part of normal procedures taken following changes in personnel, this key should be regenerated and the opennms user account password changed.*

Appendix A

Failover Scripts

The scripts and files supporting this feature are installed using the following directory structure:

```
/Users/david/Documents/Business/Engineering/Support/kcpl/failover
.
./db
./etc
./etc/opennms-datasources-failover.xml
./etc/opennms-datasources-standby.xml
./etc/sync-off.cron
./etc/sync-on.cron
./scripts
./scripts/failover.sh
./scripts/sync-state.sh
./scripts/sync.sh
```

$SYNC_HOME/scripts/sync.sh

```
#!/bin/sh

# export data from other server
# pg_hba.conf on $PRIMARY_NMS must be changed
# to allow connections from $STANDBY_NMS.  Also
# postgresql.conf on $PRIMARY_NMS must be changed
# so that the postmaster process is configured
# to listen on all IP interfaces including
# localhost.

SYNC_USER=opennms
SYNC_HOME=~$SYNC_USER/failover/
PRIMARY_NMS=nms-01
STANDBY_NMS=`host`
OPENNMS_ETC=/etc/opennms
OPENNMS_ETC_STANDBY=/etc/opennms-standby

OPENNMS_ETC_FAILOVER=/etc/opennms-failover
OPENNMS_RRD=/var/lib/opennms/rrd
OPENNMS_RRD_STANDBY=/var/lib/opennms/rrd-standby
OPENNMS_RRD_FAILOVER=/var/lib/opennms/rrd-failover
FAILOVER_DB=opennms_failover

if [ -f $SYNC_HOME/bin/sync-envvars ]; then
        . $SYNC_HOME/etc/sync-envvars
fi

# export db data
dbSync()
{
  echo "Exporting data from $PRIMARY_NMS..."
  if ! test -d "${SYNC_HOME}/db"
  then
    mkdir $SYNC_HOME/db
  fi

  pg_dump -i -h $PRIMARY_NMS -U opennms opennms > $SYNC_HOME/db/opennms.sql

  if [ $? -eq 0 ]
  then
    echo "Dropping and recreating $FAILOVER_DB Database locally..."
    psql -U opennms template1 -c "DROP DATABASE $FAILOVER_DB;"
    psql -U opennms template1 -c "CREATE DATABASE $FAILOVER_DB ENCODING 'unicode';"
  else
    exit 1
  fi

  if [ $? -eq 0 ]
  then
    echo "Importing data exported from $PRIMARY_NMS into recreated opennms_failover DB..."
    psql -U opennms $FAILOVER_DB < $SYNC_HOME/db/opennms.sql
  else
    exit 1
  fi
}


# Now it is important that the opennms configuration files
# rsync'd from $PRIMARY_NMS
etcSync()
{
  echo "Rsync'ing configuration files from $PRIMARY_NMS..."
  rsync -e 'ssh -i ~$SYNC_USER/.ssh/id-rsa-key' -avz $SYNC_USER@$PRIMARY_NMS:$OPENNMS_ETC*
$OPENNMS_ETC_FAILOVER/
}


# Sync the RRD files
rrdSync()
{
    echo "Rsync'ing RRD data files from $PRIMARY_NMS..."
    rsync -e 'ssh -i ~$SYNC_USER/.ssh/id-rsa-key' -azv $SYNC_USER@$PRIMARY_NMS:$OPENNMS_RRD/*
$OPENNMS_RRD_FAILOVER/
}

# Synchronize the DB
dbSync
if [ $? -ne 0 ]
then
  echo "Failed db sync of $PRIMARY_NMS!"
  exit 1
fi

# Synchronize the failover $OPENNMS_ETC_FAILOVER with the $PRIMARY_NMS:$OPENNMS_ETC/
etcSync
if [ $? -ne 0 ]
then
  echo "Failed etc sync of $PRIMARY_NMS!"
  exit 1
```

```
fi

# Synchronize the failover $OPENNMS_RRD_FAILOVER/ with the primary $OPENNMS_RRD/
rrdSync
if [ $? -ne 0 ]
then
  echo "Failed etc sync of $PRIMARY_NMS!"
  exit 1
fi
```

$SYNC_HOME/scripts/failover.sh

```
#!/bin/bash

# This script will start opennms
# in in a configuration that has been
# duplicated from the primary opennms
# system.

# Must first stop OpenNMS that is running and
# monitoring the primary server.

SYNC_USER=opennms
SYNC_HOME=~$SYNC_USER/failover
PRIMARY_NMS=nms-01
STANDBY_NMS=`hostname`
OPENNMS_ETC=/etc/opennms
OPENNMS_ETC_STANDBY=/etc/opennms-standby

OPENNMS_ETC_FAILOVER=/etc/opennms-failover
OPENNMS_RRD=/var/lib/opennms/rrd
OPENNMS_RRD_STANDBY=/var/lib/opennms/rrd-standby
OPENNMS_RRD_FAILOVER=/var/lib/opennms/rrd-failover
OPENNMS_BIN=/usr/share/opennms/bin

if [ -f $SYNC_HOME/etc/sync-envvars ]; then
        . $SYNC_HOME/etc/sync-envvars
fi

case "$1" in

  start)
    echo "Stopping automatic synchronization..."
    $SYNC_HOME/scripts/sync-state.sh off

    if [ $? -ne 0 ]
    then
      echo "ERROR:Failed to stop synchronization cron job; not failing over!"
      exit 1
    fi

    echo "Stopping OpenNMS in preparation of failover..."
    $OPENNMS_BIN/opennms stop
    if [ $? -ne 0 ]
    then
```

```
      echo "ERROR:Failed to stop standby OpenNMS instance; not failing over!"
      $SYNC_HOME/scripts/sync-state.sh on
      exit 1
    fi

    echo "Changing the $OPENNMS_ETC symbolic link to $OPENNMS_ETC_FAILOVER..."
    rm $OPENNMS_ETC
    if [ $? -ne 0 ]
    then
      echo "ERROR:Failed to remove symbolic link to the $OPENNMS_ETC; not failing over!\nVerify
OpenNMS state."
      $SYNC_HOME/scripts/sync-state.sh on
      exit 1
    fi

    ln -s $OPENNMS_ETC_FAILOVER $OPENNMS_ETC
    if [ $? -ne 0 ]
    then
      echo "ERROR:Failed to create symbolic link to the $OPENNMS_ETC_FAILOVER; not failing
over!\nVerify OpenNMS state."
      $SYNC_HOME/scripts/sync-state.sh on
      exit 1
    fi

    echo "Changing the $OPENNMS_RRD symbolic link to $OPENNMS_RRD_FAILOVER..."
    rm $OPENNMS_RRD
    if [ $? -ne 0 ]
    then
      echo "ERROR:Failed to remove symbolic link to the $OPENNMS_RRD; not failing over!\nVerify
OpenNMS state."
      $SYNC_HOME/scripts/sync-state.sh on
      exit 1
    fi

    ln -s $OPENNMS_RRD_FAILOVER $OPENNMS_RRD
    if [ $? -ne 0 ]
    then
      echo "ERROR:Failed to create symbolic link to the $OPENNMS_RRD_FAILOVER; not failing
over!\nVerify OpenNMS state."
      $SYNC_HOME/scripts/sync-state.sh on
      exit 1
    fi

#    echo "Setting failover data sources configuration..."
#    cp -p ~$SYNC_HOME/etc/opennms-datasources-failover.xml $OPENNMS_ETC/opennms-
datasources.xml

    echo "Restarting OpenNMS..."
    $OPENNMS_BIN/opennms start
    ;;
  stop)

    echo "Stopping OpenNMS in failover mode, returning to standby mode..."
    $OPENNMS_BIN/opennms stop
    if [ $? -ne 0 ]
    then
      echo "ERROR:Failed to stop failover OpenNMS instance; not failing over!"
      $SYNC_HOME/scripts/sync-state.sh off
      exit 1
    fi

    echo "Changing the $OPENNMS_ETC symbolic link to $OPENNMS_ETC_STANDBY..."
    rm $OPENNMS_ETC
    if [ $? -ne 0 ]
    then
      echo "ERROR:Failed to remove symbolic link to the $OPENNMS_ETC; not starting standby
mode!\nVerify OpenNMS state."
      $SYNC_HOME/scripts/sync-state.sh off
      exit 1
    fi

    ln -s $OPENNMS_ETC_STANDBY $OPENNMS_ETC
    if [ $? -ne 0 ]
    then
      echo "ERROR:Failed to create symbolic link to the $OPENNMS_ETC_STANDBY; not starting
standby mode!\nVerify OpenNMS state."
      $SYNC_HOME/scripts/sync-state.sh off
      exit 1
    fi

    echo "Changing the $OPENNMS_RRD symbolic link to $OPENNMS_RRD_STANDBY..."
    rm $OPENNMS_RRD
    if [ $? -ne 0 ]
    then
      echo "ERROR:Failed to remove symbolic link to the $OPENNMS_RRD; not starting in standby
mode!\nVerify OpenNMS state."
      $SYNC_HOME/scripts/sync-state.sh off
      exit 1
    fi

    ln -s $OPENNMS_RRD_STANDBY $OPENNMS_RRD
    if [ $? -ne 0 ]
    then
      echo "ERROR:Failed to create symbolic link to the $OPENNMS_RRD_STANDBY; not starting in
standby mode!\nVerify OpenNMS state."
      $SYNC_HOME/scripts/sync-state.sh off
      exit 1
    fi
    ln -s $OPENNMS_RRD_STANDBY $OPENNMS_RRD

#    echo "Chaning failover data sources configuration to stand-by configuration..."
#    cp -p $SYNC_USER/failover/etc/opennms-datasources-standby.xml $OPENNMS_ETC/opennms-
datasources.xml

    echo "Restarting OpenNMS..."
    $OPENNMS_BIN/opennms start

    echo "Restarting automatic synchronization..."
    $SYNC_HOME/scripts/sync-state.sh on
    ;;
```

```
  *)
    echo "Usage: $0 {start|stop}"
    exit 1
    ;;
esac
```

synch-state.sh

```
#!/bin/sh
SYNC_USER=opennms
SYNC_HOME=~$SYNC_USER/failover
PRIMARY_NMS=nms-01
STANDBY_NMS=`hostname`
OPENNMS_ETC=/etc/opennms
OPENNMS_ETC_STANDBY=/etc/opennms-standby

OPENNMS_ETC_FAILOVER=/etc/opennms-failover
OPENNMS_RRD=/var/lib/opennms/rrd
OPENNMS_RRD_STANDBY=/var/lib/opennms/rrd-standby
OPENNMS_RRD_FAILOVER=/var/lib/opennms/rrd-failover
OPENNMS_BIN=/usr/share/opennms/bin

if [ -f $SYNC_HOME/etc/sync-envvars ]; then
        . $SYNC_HOME/etc/sync-envvars
fi

case "$!" in

  on)
    echo "Enabling cron..."
    crontab -u $SYNC_USER ~$SYNC_HOME/etc/sync-on.cron
    crontab -l -u $SYNC_USER
    ;;
  off)
    echo "Enabling cron..."
    crontab -u $SYNC_USER ~$SYNC_HOME/etc/sync-on.cron
    crontab -l -u $SYNC_USER
    ;;
  *)
    echo "Usage: $0 {on|off}"
    exit 1
    ;;
esac
```

$SYNC_HOME/etc/sync-envvars

```
SYNC_USER=opennms
SYNC_HOME=~$SYNC_USER/failover/
PRIMARY_NMS=nms-01
STANDBY_NMS=`host`
OPENNMS_ETC=/etc/opennms
OPENNMS_ETC_STANDBY=/etc/opennms-standby
OPENNMS_ETC_FAILOVER=/etc/opennms-failover
OPENNMS_ETC_FAILOVER=/etc/opennms-failover
OPENNMS_RRD=/var/lib/opennms/rrd
```

```
OPENNMS_RRD_STANDBY=/var/lib/opennms/rrd-standby
OPENNMS_RRD_FAILOVER=/var/lib/opennms/rrd-failover
OPENNMS_BIN=/usr/share/opennms/bin
FAILOVER_DB=opennms_failover
```

$SYNC_HOME/etc/sync-on.crontab

```
#Crontab entry for automatic synchronization of OpenNMS
MAILTO=root
SYNC_USER=opennms
0 0 * * *       ~$SYNC_USER/failover/scripts/sync.sh
```

$SYNC_HOME/etc/sync-off.crontab

```
#Crontab entry for automatic synchronization of OpenNMS
MAILTO=root
SYNC_USER=opennms
#0 0 * * *      ~$SYNC_USER/failover/scripts/sync.sh
```