

## Maintenance and Monitoring for Scalix

---

### Overview

The necessary maintenance and monitoring tasks for the Scalix server application is simplistic, and can be easily scheduled and automated by using the ``cron" facilities. Because elements of the host computer, and the physical network also provide vital dependent services and facilities for Scalix, consideration for a broader overall scope of maintenance and monitoring should be made. By using a combination of Scalix and Linux utilities, supplemented by 3<sup>rd</sup> Party monitoring software from either commercial or open source vendors, Scalix administrators can create a very comprehensive maintenance and monitoring solution for Scalix and these dependant elements. This technote will identify the necessary maintenance and monitoring tasks and events for the Scalix application and provide some further product suggestions for deeper levels of integration for a complete enterprise monitoring solution.

### Scalix Application Specific Tasks and Events

The following tasks and checks are vital to the health of the Scalix application itself and should be performed on the level of frequency as specified below. In most cases the output should be captured to a file where it can then be processed by other Linux utilities (such as grep, sed, awk) or even another Scalix utility.

#### Check disk space utilization

Always check available disk space on the volume where the Scalix message store resides. This check should occur quite frequently (hourly) and can occur using the Linux "df" command. Should a threshold be reached (%80-%90 of volume's space) an alert should be sent.

*Example:*

```
df -t ext2
```

#### Check I-node utilization

Always check inode utilization on the volume where the Scalix message store resides as well. This check should occur quite frequently (hourly) and can occur using the Linux "df" command. Should a threshold be reached (%80-%90 of volume's space) an alert should be sent.

*Example:*

```
df -i -t ext2
```

#### Check Scalix error queues

The Scalix error queue should be monitored frequently (hourly) to see if there are any messages. OMSTAT can be used to check for errors.

*Example:*

```
/opt/Scalix/bin/omstat -q
```

#### Check Scalix message queues

The Scalix message queues should be monitored frequently (hourly) to see if a build up of messages has occurred. Depending on the amount of traffic within your system, a threshold level could be used. For example, on a 1000 user Scalix server, a good threshold might be if any of the queues reach 100 messages. The four standard queues to check are the Router, Internet, Local and Sendmail queues, each can be checked with the OMSTAT utility.

*Example:*

```
/opt/Scalix/bin/omstat -s | grep -Ei 'router|local|internet|sendmail'
```

**Check Scalix services/daemons**

All Scalix services and daemons should be monitored frequently (hourly) to verify they are running. OMSTAT can be used to check both Scalix services and daemons (-s and -a switches of OMSTAT). It is important to look for “aborted” services and daemons as opposed to “stopped” services and daemons.

*Example:*

```
omstat -s | grep -i aborted
omstat -a | grep -i aborted
```

**Remove outdated messages**

One of the most typical tasks for any mail system is to set message retention limits for various containers, thus requiring a purging process on the outdated messages. The Scalix OMTIDYALLU utility can be used for this process.

*Example:*

```
/opt/Scalix/bin/omtidyallu -w 30 -d 2>&1 >> $MAINTLOG
```

**Age the Scalix audit log**

The Scalix audit log (/var/opt/scalix/logs/audit) should be aged on a daily basis. Use basic Linux scripting to roll the logs.

*Example:*

```
AUDFIL=/var/opt/Scalix/logs/audit
DAY=`date +%a`
SAVFIL="$AUDFIL.$DAY"
rm -f $SAVFIL 2>&1 >> $MAINTLOG
mv -f $AUDFIL $SAVFIL 2>&1 >> $MAINTLOG
touch $AUDFIL 2>&1 >> $MAINTLOG
chown Scalix $AUDFIL 2>&1 >> $MAINTLOG
chgrp sxoffice $AUDFIL 2>&1 >> $MAINTLOG
chmod 660 $AUDFIL 2>&1 >> $MAINTLOG
```

**Backup Scalix**

Backing up Scalix is arguably the most important task to execute and should occur at least once daily. Please review the Scalix Backup and Restore technote for further details.

**Scan the Scalix message store for inconsistencies**

The Scalix message store should be checked for any inconsistencies on a weekly basis. Unlike other operating systems, disk defragmentation is not required with Linux.

*Example:*

```
omscan -a -f -x 2>&1 2>&1 > /var/opt/Scalix/logs/omscan.out
```

**Practical Application - OMMAINT**

Scalix provides a sample script (OMMAINT) in the **/integration** folder on the Scalix Product CD. OMMAINT itself will execute all the maintenance and monitoring tasks necessary for the Scalix application as described above, and uses email to report the results of each task and event. It is recommended that other 3<sup>rd</sup> Party monitoring products be used to supplement the overall monitoring solution as described in the prior sections of this technote.

```
#!/bin/sh

#####
#####
## IMPORTANT: This script is not supported by Scalix Corp.. Use at your own
risk. ##
#####
#####

#
# Author : Scalix Technical Support, support@scalix.com
#
# NAME:      sxmaint - Perform Scalix periodic maintenance
#
# SYNOPSIS:  sxmaint -frequent | -daily | -weekly
#
# DESCRIPTION:
# Perform Scalix regular maintenance. I thought of everything I make
# Scalix do on a periodic basis and threw it in here. It takes frequent,
# daily, and weekly maintenance and puts them in a single script so that
# it's easier to e-mail around. You typically cron this to run as follows:
#
#      # minute hour monthday month weekday command
#      00,30 * * * * /usr/local/bin/ommaint -frequent
#      01 0 * * * /usr/local/bin/ommaint -daily
#      15 2 * * 0 /usr/local/bin/ommaint -weekly
#
# Exactly when you run everything doesn't matter. Frequent is for frequent
# ops such as checking for aborted queues or other errors. Daily empties
# the users' wastebaskets, rolls the audit logs, and does backups. Weekly
# runs omscan. You might be able to throw omscan in the daily part if you
# have a small installation.
#
# Over time you'll want to adapt this to your preferences. This is intended
# to provide a starting point for system admins new to Scalix. In
# particular, be sure to take a look at the backup section, since you'll
# probably want to modify that to your preferred way of doing backups. For
# simplicity I just use tar here because it's simple and it always exists.
#
# PLATFORMS TESTED: RedHat LINUX 6.1
#
# SUPPORT: None. However, if you make a useful addition that you think others
# can benefit from, or if you port it to a different platform, or
# if you find *and fix* a bug, please send me your modifications
# and I'll update the original, and thanks for your support.
#

#
# SECTION: General Configuration Section
#
#      General configuration for this script
#

# Address to mail reports to
MAIL_REPORTS="root"

# Device to use for backups
BACKUP_DEVICE=/dev/rmt0

# -----
# SECTION 1: FREQUENT OPERATIONS.
```

```

#
# Usage: ommaint -frequent
#
#   These operations should be done at frequent intervals, no less than
#   once/hour. Typically they do things such as checking for problems
#   such as running out of disk space - things that should be done
#   frequently. They also include operations that do not need to be
#   done frequently, but also don't require much time to complete.
#
ommaint_frequent ()
{
    MAINTLOG=/tmp/ommaint.$$
    STDERR=/tmp/ommaint.$$stderr
    STDOUT=/tmp/ommaint.$$stdout

    rm -f $MAINTLOG $STDERR $STDOUT
    touch $MAINTLOG $STDERR $STDOUT

    # ACTION: Check Disk Space Utilization
    #   Notify administrator if disk space utilization on any filesystem >= 80%
    #
    df -t ext2 | awk '{if (int($5)>=80) {print $0}}' > $STDOUT
    if [ -s $STDOUT ]
    then
        echo "These filesystems have >= 80% disk space utilization:" >>
$MAINTLOG
        echo "-----" >> $MAINTLOG
        df -t ext2 | head -1 >> $MAINTLOG
        cat $STDOUT >> $MAINTLOG
    fi

    # ACTION: Check Inode Utilization
    #   Notify administrator if inode utilization on any filesystem >= 80%
    #
    df -i -t ext2 | awk '{if (int($5)>=80) {print $0}}' > $STDOUT
    if [ -s $STDOUT ]
    then
        echo "These filesystems have >= 80% inode utilization:" >> $MAINTLOG
        echo "-----" >> $MAINTLOG
        df -i -t ext2 | head -1 >> $MAINTLOG
        cat $STDOUT >> $MAINTLOG
    fi

    # ACTION: Check the Error Queues
    #   Include the results in the report to the system admin only if there
    #   are some messages in the ERROR queues
    #
    for queue in ERROR SMERR
    do
        /opt/Scalix/bin/omstat -q $queue >$STDOUT 2>$STDERR
        if [ "$(cat $STDERR)" != "omstat : There are no messages on the queue" ]
        then
            echo "Messages on the Scalix $queue Queue" >> $MAINTLOG
            echo "-----" >> $MAINTLOG
            cat $STDERR >> $MAINTLOG
            cat $STDOUT >> $MAINTLOG
        fi
    done

    # ACTION: Check the Scalix Queues
    #   Report to the administrator if the number of messages queued up for
    #   delivery on any of the standard queues exceeds 100.
    #
    omstat -s | grep -Ei 'router|local|internet|sendmail' >$STDOUT 2>$STDERR
    cat $STDOUT | awk '{print $1" "substr($0,65,10)}' | while read line
    do
        queue=`echo $line | awk '{print $1}' | tr a-z A-Z`
        msgcount=`echo $line | awk '{print $2}'`
        if [ $msgcount -gt 100 ]
        then
            echo "Messages on the Scalix $queue Queue = $msgcount"
        fi
    done
}

```

```

# ACTION: Check for aborted services/daemons
# Report to the administrator any Scalix services or daemons
# which have become aborted
#
omstat -s | grep -i aborted > $STDOUT
omstat -a | grep -i aborted >> $STDOUT
if [ -s $STDOUT ]
then
  echo "The following services/daemons are ABORTED:" >> $MAINTLOG
  echo "-----" >> $MAINTLOG
  cat $STDOUT >> $MAINTLOG
  echo "" >> $MAINTLOG
fi

if [ -s $MAINTLOG ]
then
  mail -s "Scalix Frequent Maintenance Report" $MAIL_REPORTS <
$MAINTLOG
  fi

  rm -f $MAINTLOG $STDERR $STDOUT
}

# -----

# SECTION 2: DAILY OPERATIONS.
#
# Usage: ommaint -daily
#
# These operations should be done at daily intervals. Typically they do
# things such as maintaining log files so that log files do not grow
# to consume all available disk space.
#
# Daily operations should be performed around midnight. This is for two
# reasons. First, log files should be rolled around midnight so that
# audit.Wed is indeed Wednesday's data. Second, some daily maintenance
# operations are CPU and disk intensive, and should not coincide with
# periods of heavy system load. System load is usually highest during
# working hours when users actively using the server.
#
ommaint_daily ()
{
  DAY=`date '+%a'`
  MAINTLOG=/var/opt/Scalix/logs/ommaint.$DAY

  rm -f $MAINTLOG
  touch $MAINTLOG

  # ACTION: Remove items deleted > 30 days ago from users' trash folders
  #
  # CUSTOMIZATIONS AVAILABLE:
  # omtidyallu can also used to remove outdated items at customizable
  # periods from the trash, inbox, outbox, and/or folders. It's also possible
  # to create different retention schedules for individual users using
  # omtidyu, and many other customizations are available. See
  # "man omtidyallu" for these and other configurations.
  #
  /opt/Scalix/bin/omtidyallu -w 30 -d 2>&1 >> $MAINTLOG

  # ACTION: Age the Scalix Audit log
  #
  AUDFIL=/var/opt/Scalix/logs/audit
  DAY=`date '+%a'`
  SAVFIL="$AUDFIL.$DAY"
  rm -f $$SAVFIL          2>&1 >> $MAINTLOG
  mv -f $AUDFIL $$SAVFIL 2>&1 >> $MAINTLOG
  touch $AUDFIL          2>&1 >> $MAINTLOG
  chown Scalix $AUDFIL   2>&1 >> $MAINTLOG
  chgrp sxoffice $AUDFIL 2>&1 >> $MAINTLOG
  chmod 660 $AUDFIL      2>&1 >> $MAINTLOG
}

```

```

# ACTION: Backup Scalix
#
# WARNING:
# In order to guarantee Scalix database integrity, Scalix backups
# must be made from an offline copy of the data. This is because a
# change to the database is generally made by modifying multiple files
# simultaneously. If, during a backup, one of these files is modified
# after it is backed up, but the others are modified before they are
# backed up, than the backup will be corrupt. The options for backing
# up are therefore:
# 1. Shut down Scalix, back up, Restart
#   Advant: Backup not corrupt. Simple. Always works
#   Disadv: System will be unavailable during backup.
# 2. Use advanced filesystem such as ReiserFS, XFS, or Veritas
#   to make a "snapshot" filesystem. A snapshot filesystem holds
#   the changes made since the backup started, and allows the
#   backup program to access a "snapshot" of the system at the
#   time the backup started.
#   Advant: Backup not corrupt. No downtime.
#   Disadv: Requires advanced FS. Requires approximately extra
#   10% disk space to hold snapshot information.
# 3. Backup Scalix online from online data set. The restore
#   procedure will need to include commands to uncorrupt the
#   database. Those commands take time to run. This generally
#   works, but it is not guaranteed to work and is "unsupported".
#   The command to uncorrupt a database is omscan(8).
#   Advant: No downtime. Simple.
#   Disadv: Lengthy, unreliable restore. Unsupported.
#
# This backup script does option #1, shut down to backup, so that
# it will work even if you do not have an advanced filesystem. If
# you do have an advanced filesystem, comment out the default backup
# script and modify the steps below to build a backup script based
# on a snapshot filesystem.
#
# This sample backup script does a full backup of Scalix data in
# /var/opt/Scalix, and it uses tar. Scalix does not care if you
# do full or partials, if you backup /, or which backup program you use.
# However, please take note of the WARNING above.
#
# Standard backup script:
#
/opt/Scalix/bin/omshut          2>&1 >> $MAINTLOG
tar -cf $BACKUP_DEVICE /var/opt/Scalix 2>&1 >> $MAINTLOG
/opt/Scalix/bin/omrc          2>&1 >> $MAINTLOG

# # Backup script template for advanced filesystems:
# #
# /opt/Scalix/bin/omsuspend -s 300 & # Pause OM while making snapshot
# sync # This is a must!
# create_snapshot /var/opt/Scalix /var/optomsnap # Create snapshot FS
# /opt/Scalix/bin/omsuspend -r # Unpause OM after snapshot ready
# tar -cf $BACKUP_DEVICE /var/optomsnap
# cancel_snapshot /var/optomsnap

mail -s "Scalix Daily Maintenance Report" $MAIL_REPORTS < $MAINTLOG
rm -f $MAINTLOG
}

# -----

# SECTION 3: WEEKLY OPERATIONS.
#
# Usage: ommaint -weekly
#
# These operations should be done at weekly intervals. Typically they do
# things that are not necessary to perform daily and take up so much
# processing power that they should only be run infrequently and when
# when the system is lightly utilized (such as during a weekend).
#
ommaint_weekly ()
{
# ACTION: Scan the Scalix Message Store for corruptions, and e-mail a

```

```
#           report to the system administrator
#
# CONFIGURATION OPTIONS:
# See the manual entry for ommscan(8) for more options. Unless the system
# is very large, you can scan it daily if you prefer. However, daily
# scans take up CPU and I/O bandwidth and are generally not necessary.
#
ommscan -a -f -x 2>&1 2>&1 > /var/opt/Scalix/logs/ommscan.out
sed -e 's/^/ /' /var/opt/Scalix/logs/ommscan.out | mail -s "Scalix Weekly
Maintenance Report" $MAIL_REPORTS
}

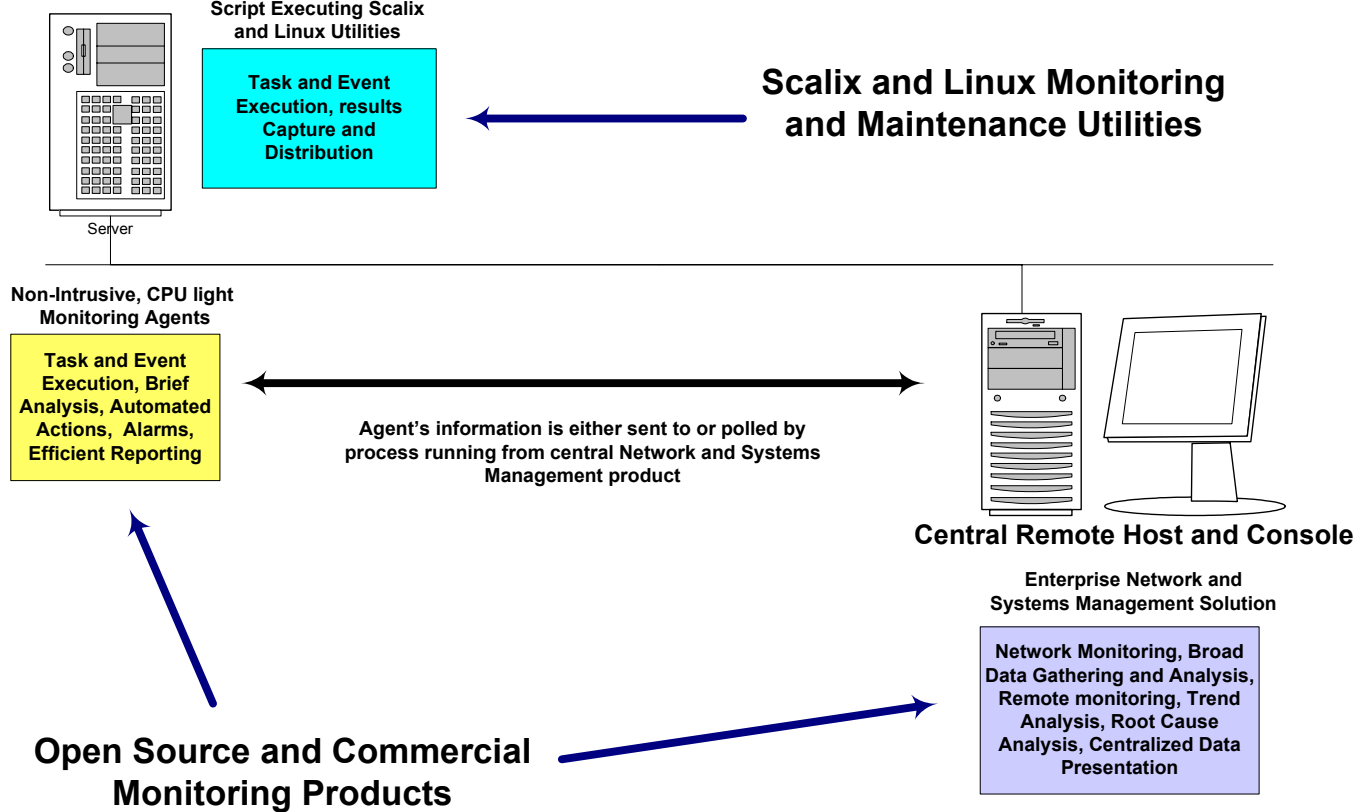
usage ()
{
    echo "USAGE: ommscan [ -frequent | -daily | -weekly ]" >&2
    exit 1
}

if [ $# -ne 1 ]; then usage; fi
case $1 in
    -frequent) ommscan_frequent ;;
    -daily)    ommscan_daily    ;;
    -weekly)   ommscan_weekly   ;;
    *)        usage             ;;
esac
```

### Understanding a Complete Solution

Scalix provides many proprietary utilities, which can be used to perform maintenance functions and report on various queues, containers, services and daemons of the Scalix application, while the Linux operating system can provide utilities to check disk space and inode utilization. The results of the execution of these utilities can then be captured and optionally distributed (typically via email); however this is only one half of a proper maintenance and monitoring solution. Expedient analysis of the maintenance and monitoring results, coupled with appropriate alerts and corrective measures for unfavorable results encompasses the second half of an effective solution. Because of inefficiencies in dedicating a human resource to the overall process, there exists a vast selection of Open source and commercial monitoring products designed to provide advanced levels of reporting, alerting and execution of corrective measures through automated action mechanisms. It is highly recommended that Scalix customers take advantage of these products to augment the existing basic capabilities provided by Scalix and Linux.

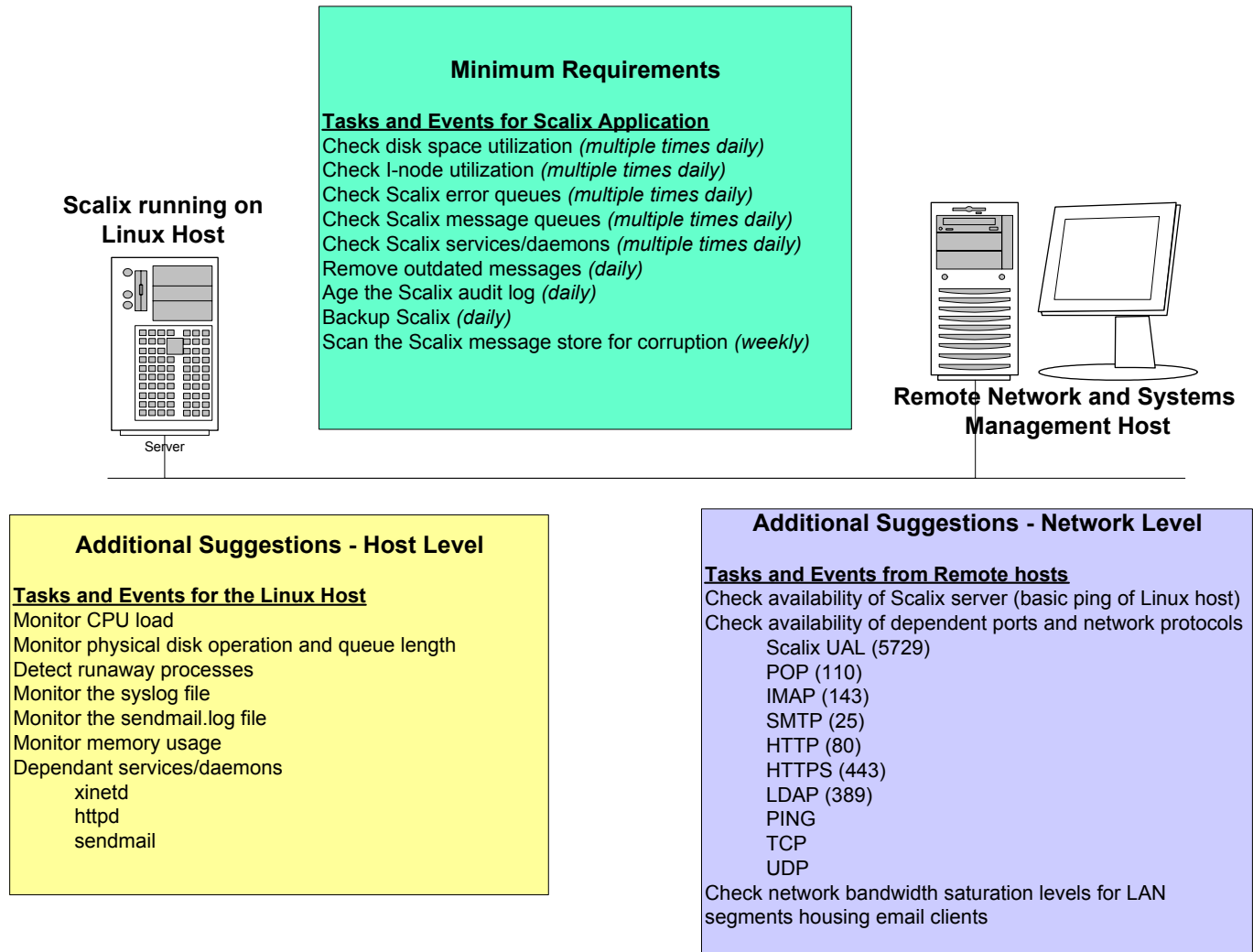
#### Scalix running on Linux Host



Many monitoring products use intelligent agents that could potentially be run on the Linux host, which is running Scalix. These agents typically provide monitoring, reporting, alert mechanisms and where applicable, some auto-actions. Often these agents can provide data back to a central network and systems management solution, which can process the reported data as well and provide a broader display of the overall health of all enterprise systems and components.

### Specific Items for a Complete Solution

Maintenance and monitoring of Scalix server involves a combination of hourly, daily and weekly tasks or events, the results of which should be reported and analyzed. Additionally, various components of both the host server and the physical network should be checked as well. The following graphic provides recommendations for a complete enterprise monitoring solution. The Scalix-specific maintenance and monitoring tasks and events will be discussed in greater detail in the subsequent section.



### Additional Suggestions

Many monitoring software packages provide intelligent agents that could potentially be run on the Linux host, which is running Scalix. These agents typically provide monitoring, reporting, alert mechanisms and where applicable, some auto-actions. Often these agents can provide data back to a central network and systems management product. Together these products can be use

## Linux Supporting Network & Systems Management Products

The following section provides some information from some of the better-known vendors of monitoring products. These range from commercial enterprise-wide multi-platform products which provide extended trend and root cause analysis, to simple but effective Linux specific monitoring products available through GNU general public licensing.

### IBM Tivoli

<http://www-3.ibm.com/software/tivoli/solutions/linux/>

Tivoli has always embraced an open management philosophy by offering solutions that integrate across heterogeneous environments. We continue to lead this philosophy by now extending e-business infrastructure management support to include the Linux operating system. Tivoli, along with IBM, believes so strongly in Linux that we provide the same support for Linux as for other, proprietary operating systems.

Whether, you are integrating Linux into your existing environment or have built your infrastructure solely on Linux, Tivoli offers the e-business infrastructure management expertise you need. Tivoli systems management software equips your company to handle all aspects of systems management: security, performance, availability, configuration, and systems operation. With solutions built to be flexible, reliable, and scalable, Tivoli can help keep your e-business applications and resources available and secure across your entire heterogeneous infrastructure.

Features	Advantages	Benefits
Resource monitoring	Out of the box and customizable	Detects outages fast and relieves operator from manual monitoring
Application recovery	Fast and consistent cluster-wide restart of components and whole applications	Reduces down time of your critical business applications. Maintains system availability in business context. Reduces system outages.
Automatic start, stop and move of your applications	Takes care of cluster-wide relationships, start/stop order, required pre and post start/stop actions	Relieves operator from manual command entry. Eliminates operator errors.
Resource grouping	Resources can be grouped into applications. Grouping can be cluster-wide. Group attributes are inherited. Multiple memberships and groups of groups are supported. Used to start, stop, and monitor and for policy definition.	Reduces complexity by operations at the application level, not at the resource level. Frees operators from remembering application components. Reduces operator interventions such as starting and stopping of resources
Resource relationships	Defines interdependent resource relationships and associates conditions with resources, i.e.: Start sequence relationship; Depends On relationships; Location relationship	Frees operators from remembering application components and relationships and therefore reduces operations errors. Can use sophisticated knowledge about application components and their relationships to decide corrective actions within the right context Lower-priority business applications can be shut down to keep higher-priority business applications functioning.

Policy-based automation	Policy consists of resource information, groups of resources, and relationships	Reduces automation implementation time, and coding and support effort. Leverage manpower through reduced education requirements. No programming skills are required for policy definition New resources or systems can be added without re-writing scripts, which can ease application growth and application scaling.
Reliable; Scalable	Cluster-wide heart-beating and reliable messaging service	Helps implementing a cluster. Basis for reliable automation and fast

### **Computer Associates - Unicenter Network and Systems Management 3.0**

<http://www3.ca.com/solutions/>

While the evolution of Linux as an open source community, an operating system and a viable eBusiness solution platform continues to transform the business landscape, it is also creating new management challenges. CA has recognized and responded to this trend by extending our industry-leading eBusiness management solutions to distributed and mainframe Linux implementations.

CA offers more than 60 solutions that can be deployed from the laptop to the data center - providing the most comprehensive portfolio of solutions to manage, secure, preserve and integrate Linux enterprise-wide. These solutions enable organizations to:

- Manage Linux to optimize the reliability, availability and performance of the operating environment.
- Secure and preserve the data, applications and systems running in their IT environments.
- Integrate the Linux environment within the heterogeneous enterprise to achieve an overall competitive advantage.

The Unicenter Network and Systems Management product manages the health and availability of operating systems and provides basic status management on all infrastructure elements such as network devices, business applications and database systems. Powerful auto-discovery builds a database with information on system elements and populates 2D and 3D system dynamic visualizations. Historian keeps you informed with past events and object status whereas predictive management capabilities inform you about possible bottlenecks in your systems and applications in future to take automated actions to avoid them. Portal technology provides personalized intuitive information for both technical and business focused administrators.

Tracking of a number of different elements is required in order to provide a valid view of system health. That's why Unicenter Network & Systems Management includes agents, which provide a holistic view of the many different platforms in the enterprise, which might network together to perform a business service. For example, information can be gathered from AS400, Linux, Tandem, Unix, OS/390, Windows and even MVS and brought together to a single management station. Accessing data is one of the most error prone and time-consuming elements of end-to-end application performance. That's why monitoring of databases such as Microsoft SQL, Oracle, Sybase, DB2 and others is a critical component of the network/systems picture.

## **NetIQ – AppManager 5.0**

<http://www.netiq.com/products/am/modules/xplatform/rhlinux.asp>

AppManager for Red Hat Linux is one component of NetIQ's AppManager Suite, the industry's leading solution for managing, diagnosing and analyzing the performance and availability of Windows- and UNIX-based systems, applications and server infrastructures.

In addition to delivering comprehensive event management and proactive alert messaging, this module checks for potential problems, triggers appropriate actions and gathers long-term data for planning, analysis and reporting. Here are some of the out-of-the-box management functions provided by AppManager for Red Hat Linux:

- Monitors an ASCII text file for specific strings and messages logged since the last monitoring interval.
- Monitors CPU usage for each process and total CPU usage for all processes.
- Monitors physical disk operation time and queue length.
- Monitors the number of failed logon and failed su attempts since the last interval.
- Detects runaway processes by sampling CPU usage and (optionally) terminates the process.
- Monitors the syslog file for specified search strings.
- Monitors total CPU used by all processes and which processes consume the most CPU resources.
- Monitors the total memory used by all processes and which processes consume the most memory.

## **Big Brother**

<http://www.quest.com/bigbrother/>

Big Brother™ Professional Edition is a lightweight, low cost tool that monitors system and network-delivered services for availability. Once Big Brother detects a problem, it immediately notifies the administrator by an e-mail, pager or text-messaging alert - allowing for quick response and resolution.

- Collect basic process and machine information with lightweight agent-based architecture
- Detect availability issues with color coded [Web-based interface](#) — red is bad, green is good
- Customize warning and alarm levels
- Customize additional scripts to monitor other metrics
- Maintain historical status information for the last 50 measurements
- Monitor network-based servers and devices
- [Integrates with Foglight®](#) for application-centric monitoring
- Works in conjunction with [Spotlight®](#) for real-time diagnostics

### *Foglight® Integration*

Big Brother offers seamless integration with [Foglight®](#), Quest's 24x7 application monitoring solution. The combination of Foglight and Big Brother allows administrators to monitor the servers and applications in their enterprise at the appropriate level.

## **WhatsUp**

<http://www.ipswitch.com/products/WhatsUp/index.html>

WhatsUp Gold enables you to keep your mission-critical network services up and running. It's ideal for businesses of all sizes: larger organizations can use WhatsUp Gold at the departmental level to monitor the availability of mission-critical applications or to complement the existing monitoring system; small-to-medium sized businesses can use it as their all-in-one network management system

## **Nagios**

<http://www.nagios.org/>

Nagios® is a host and service monitor designed to inform you of network problems before your clients, end-users or managers do. It has been designed to run under the Linux operating system, but works fine under most \*NIX variants as well. The monitoring daemon runs intermittent checks on hosts and services you specify using external "plugins" which return status information to Nagios. When problems are encountered, the daemon can send notifications out to administrative contacts in a variety of different ways (email, instant message, SMS, etc.). Current status information, historical logs, and reports can all be accessed via a web browser.

- Monitoring of network services (SMTP, POP3, HTTP, NNTP, PING, etc.)
- Monitoring of host resources (processor load, disk and memory usage, running processes, log files, etc.)
- Monitoring of environment / [temperature](#)
- Simple plug-in design that allows users to easily develop their own host and service checks
- Ability to define network host hierarchy, allowing detection of and distinction between hosts that are down and those that are unreachable
- Contact notifications when service or host problems occur and get resolved (via email, pager, or other user-defined method)
- Optional escalation of host and service notifications to different contact groups
- Ability to define event handlers to be run during service or host events for proactive problem resolution
- Support for implementing redundant and distributed monitoring servers
- External command interface that allows on-the-fly modifications to be made to the monitoring and notification behavior through the use of event handlers, the web interface, and third-party applications
- Retention of host and service status across program restarts
- Scheduled downtime for suppressing host and service notifications during periods of planned outages
- Ability to acknowledge problems via the web interface
- Web interface for viewing current network status, notification and problem history, log file, etc.
- Simple authorization scheme that allows you restrict what users can see and do from the web interface

## **Sysmon**

<http://www.sysmon.org/>

Sysmon is a network-monitoring tool designed to provide high performance and accurate network monitoring. Currently supported protocols include SMTP, IMAP, HTTP, TCP, UDP, NNTP, and PING tests.

## **OpenNMS**

<http://www.opennms.org/>

OpenNMS is an enterprise-grade network management system built using the open-source development model.

First: the name. The "NMS" part stands for Network Management System. An NMS is a system that monitors a computer network in an attempt to prevent and quickly identify problems that affect how well the computer network runs. The "Open" part stands for "open source", which means that the source code is available for no cost, and can be freely modified. OpenNMS is published under the [GNU General Public License](#) (GPL), which does impose some restrictions, usually involving trying to make the code less than open.

Second: enterprise-grade. There are a number of open-source NMS projects out there. OpenNMS does compete with them, but this does not mean that we are enemies. Many have been around longer than OpenNMS, but the main difference is that OpenNMS was designed from the beginning to manage a very large number of devices. With properly tuned hardware, OpenNMS can monitor over 15,000 services and collect data on over 8000 devices - the upper limit has yet to be determined.