SNMP Agents for Solaris and their Integration with NetView and Mid Level Managers

Jane Curry, Skills 1st Limited
jane.curry@skills-1st.co.uk

Abstract

Simple Network Management Protocol (SNMP) is the basic protocol of network management. In order to deploy Tivoli network management products such as NetView and Mid Level Managers (MLMs), a reliable base SNMP agent is a prerequisite on the management system.

Sun Microsystems provide the Solstice Enterprise Agent (SEA) with the Solaris operating system, which offers basic functionality but whose customisation is rather complex. The net-snmp agent is publicly available from www.net-snmp.com and includes more comprehensive support with a simpler customisation process.

These two agents can be combined to provide the best of both worlds and to offer a robust platform on which to build network management stations.

This paper describes both agents, including customisation examples. It also discusses the integration with Tivoli NetView and Tivoli MLM and the configuration required to take best advantage of the functionality offered.

Introduction

Any device that is to be managed using the SNMP protocol must have an SNMP agent installed and configured. An SNMP agent must support a minimum level of functionality:

- Respond to requests from an SNMP manager using the SNMP protocol - SNMP GETs for read access to information, and SNMP SETs for write access
- Ability to issue SNMP TRAPs to an SNMP manager
- Provide a mechanism to customise security access to the SNMP agent
- Support access to information defined by the MIB-2 standard (RFC 1213) - this largely defines *network* information such as bytes in to an interface, TCP segments sent, etc...

Tivoli NetView is an SNMP manager capable of managing many thousands of SNMP devices. In addition, NetView also offers a Mid Level Manager (MLM) which is code that can be deployed throughout a large network, to offload some of the network manager's job, to MLMs closer to the devices being managed. A basic prerequisite of deploying either NetView or an MLM, is that the machine should already have an active and reliable SNMP agent. The more functionality is supported by a manager's SNMP agent, the better management one has of one's manager!

There are a number of different versions of the SNMP protocol. Version 1, published in 1988 with updates in 1990 and 1991, is still the only full standard. SNMP V2 was designed in the mid-1990s and reached "Proposed" status in 1996 before being superseded by work on SNMP

V3 in 1999. V3 has now reached "Draft" status. The main areas of enhancement in later versions of SNMP address moving large amounts of data more effectively, and with security issues. The major problem with a V1 SNMP agent is that, although it has a password with each data packet which must match customisations at the manager and agent, this password (known as a *community name*) is not encrypted, so any network sniffing tool can see the password.

Basic SNMP agents typically only support access to data defined by MIB-2. There are a number of mechanisms available to extend the scope of an agent to provide further information. Extra functionality is usually implemented via a master / subagent paradigm which means that the standard SNMP manager / agent mechanism is unchanged but the standard agent has a way to devolve responsibility for some MIB queries, to a subagent. Such extensions must meet the following requirements:

- The standard (master) SNMP agent must accept SNMP requests from an SNMP manager for other MIB variables
- It will pass the MIB request to the subagent extension that will handle the request and return a response
- The subagent will have instrumentation to query the system to supply data, in MIB format, back to the master agent
- The master agent then returns the response to the SNMP manager

For example, the Sun MIB is supported by Sun's *mibiisa* subagent. mibiisa interacts with the Operating System and can provide information on CPU utilisation and swap data.

A number of standardised and non-standardised solutions exist for extending a basic SNMP agent:

- SMUX defined in RFC 1227 - SNMP multiplexing protocol, historic status. SMUX is used by the IBM AIX SNMP agent
- DPI defined in RFC 1228 & 1592 - Distributed Protocol Interface, experimental status
- AgentX defined in RFC 2741 - proposed status. Various publicly available toolkits are available, eg. net-snmp
- Various, non-standard SNMP pass-through mechanisms eg. SUN SEA

Background to the scenarios discussed

In the following sections, the network management setup is as shown below:

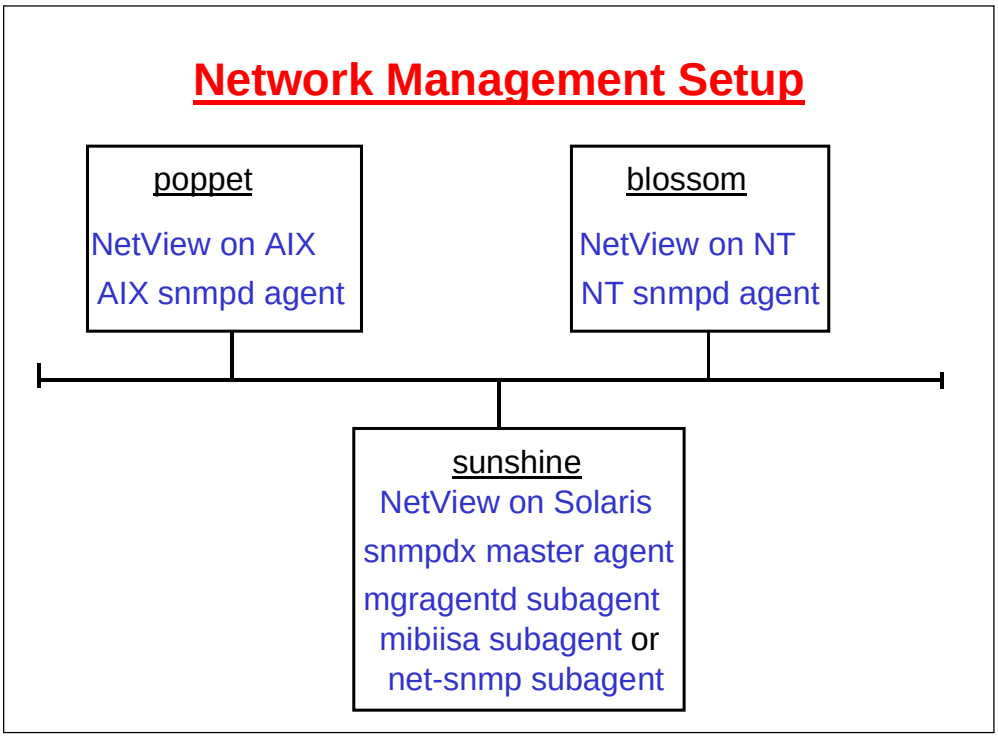

Figure 1. Network management setup for scenarios

It would be unusual to have so many network managers on a typical production network but this setup permits demonstration of a number of points.

The Solaris system called **sunshine** runs the snmpdx master agent and *either* Sun's mibiisa subagent *or* the publicly-available net-snmp agent. Either of these can respond to MIB-2 queries (but not both at the same time!). **sunshine** can also run NetView Server. The mgragentd subagent is a NetView process that handles communication between NetView Server and NetView Client.

For each network manager, community names need to be configured on the SNMP agents, both for read-only access and for read-write access. Many organisations prefer to block access to agents with the default community name of **public**, choosing "good" community name passwords instead. One method of selecting memorable passwords that are meaningless (and un-rememberable!) to others, is to choose the first letter of a well-known phrase. Hence, you will see read-only community names in the samples, of **rrwatr** (**r**omeo **r**omeo **w**herefore **a**rt **t**hou **r**omeo) and read-write community passwords of **fraclmye** (**f**riends **r**omans **a**nd **c**ountrymen **l**end **m**e **y**our **e**ars).

Be very aware that one of the major drawbacks with SNMP V1 is that community names are **not** encrypted so anyone with a network sniffer tool or a packet analysis utility can see these community name passwords. For example, it would be a very bad idea to use the easily-snooped snmp password as the root access password for the operating system!

Sun's Solstice Enterprise Agent (SEA)

The SNMP agent shipped with Sun's Solaris operating system is known as the **Solstice Enterprise Agent (SEA)**. Details in this paper relate to Solaris 2.8 with the SEA at version 1.0.3, with patch 108869-10.

The SEA supports SNMP V1 and deploys a non-standard master agent / subagent architecture. The master agent, **snmpdx**, opens UDP port 161 (the standard port for SNMP GETs and SETs), and accepts SNMP queries. Configuration files, found by default in **/etc/snmp/conf** , allow various subagents to register with the master agent so that specified MIB queries are passed through from snmpdx to the subagent. Each subagent must use a different, configurable port to communicate with snmpdx (**not** UDP/161).

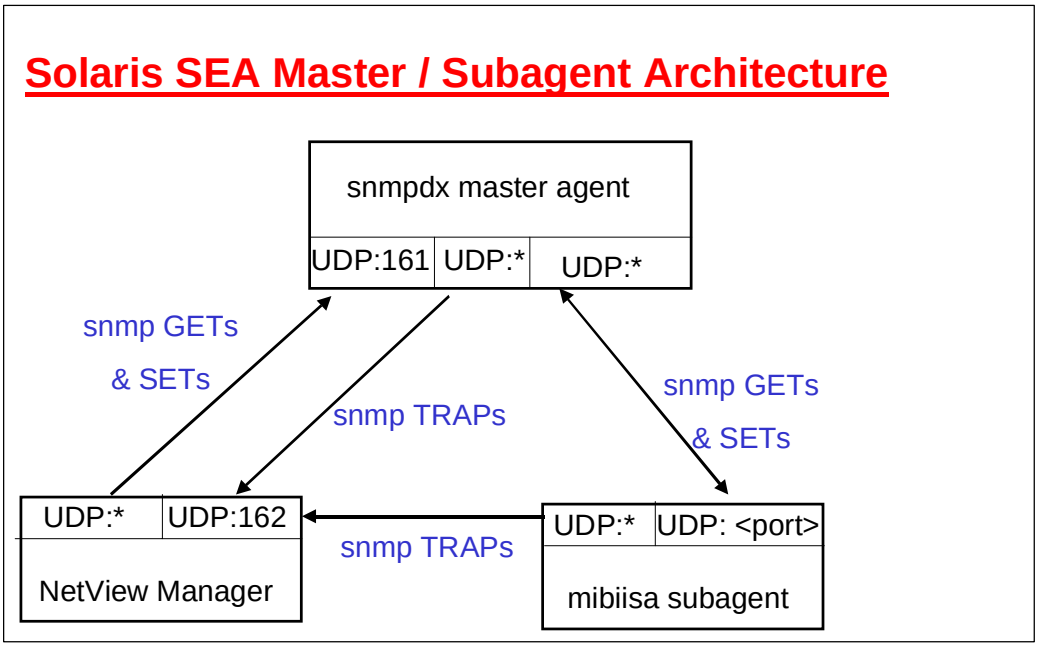

Figure 2. Solaris SEA Master / Subagent Architecture

Figure 2 shows the SEA master / subagent architecture. The master snmpdx agent and the mibiisa subagent are 2 logical entities on the same system. Typically, the NetView manager will be a different system but for a machine running NetView, all 3 logical entities will be within the same machine.

The ports marked **UDP:*** are random, ephemeral ports; for a given system; obviously these ports will all be different for different conversations. The **UDP: <port>** notation represents the *customisable* port that snmpdx will use to talk to a subagent. This **UDP:<port>** combination must be different for each subagent and will be configured in the subagent's registration file (see later examples).

The mibiisa subagent will handle MIB requests for MIB-2 variables and for the Sun enterprise-specific MIB. It sends TRAPs direct to the NetView manager, not via snmpdx.

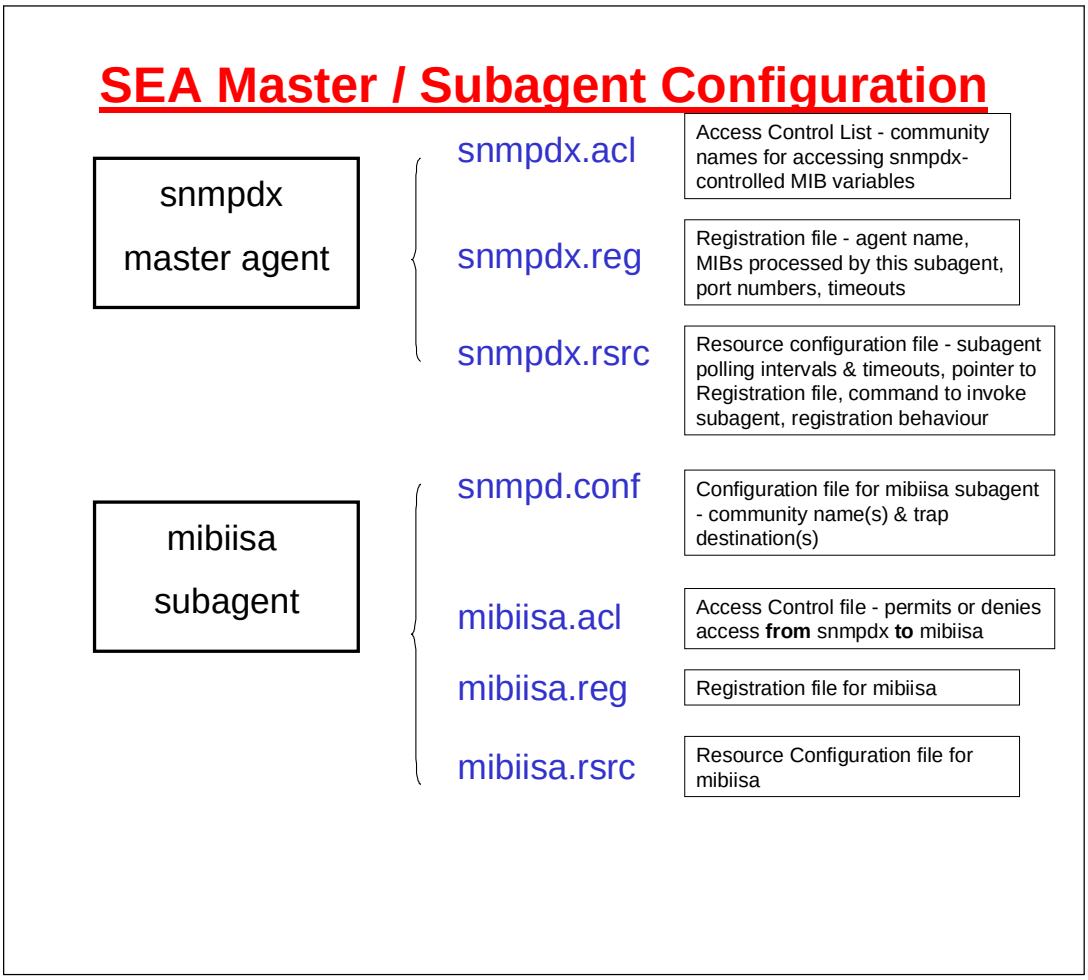

Figure 3. SEA Master / Subagent Configuration

Figure 3 shows the different customisation files for each of the snmpdx master agent and the mibiisa subagent. By default, all SEA customisation files are in **/etc/snmp/conf**.

snmpdx master agent

When the snmpdx master agent is started, it first initialises itself. **snmpdx.reg** is consulted which specifies that the snmpdx master agent itself handles queries for the MIB subtree **sun.2.15** - these are the MIB variables for snmpdx. snmpdx will open port UDP 161 and listen for SNMP requests from network managers.

```
#ident "@(#)snmpd.snmprelay     1.3 25 Jan 1996 SMI"
#
# Copyright 25 Jan 1996 Sun Microsystems, Inc. All Rights Reserved
#  Personal/Configuration file of the relay agent
# snmpdx itself handles queries for the Sun snmpdx MIB - sun.2.15
#  It is snmpdx that gets UDP port 161
#
##########
# agents #
##########

agents =
{
       {
              name = "relay-agent"
              subtrees = { sun.2.15 }
              timeout = 900000000
              port = 161
       }
}
```

Figure 4. snmpdx.reg

snmpdx.acl specifies which managers with which community names are permitted access to the MIB tree specified in snmpdx.reg ie. **sun.2.15**. snmpdx.acl can also specify where to send on, any traps that it receives. Note that although snmpdx is the master agent, **snmpdx.acl** is **NOT** a blanket authorisation file for all MIBs - just those specified in snmpdx.reg.

```
# Copyright 12/11/96 Sun Microsystems, Inc. All Rights Reserved.
#pragma ident  "@(#)snmpdx.acl   1.5 96/12/11 Sun Microsystems"
#
#  Configuration file of the snmpdx master agent
#
###################
# access control #
###################
# The list of community names needed for read/write access
#     to the MIBs specified in snmpdx.reg ie. sun.2.15   (snmpdx)
# If the list is empty, the only valid community name is "public"
#        and its access type is read-only
acl = {
        {
                communities = rrwatr
                access = read-only
                managers = localhost, poppet, blossom, sunshine
        }

        {
                communities = fraclmye
                access = read-write
                managers = localhost, poppet, blossom, sunshine
        }
      }
####################
# trap parameters #
####################
#All subagents send traps directly so no trap customisation required for snmpdx
#
trap = {
}
```

Figure 5. snmpdx.acl

snmpdx.rsrc is the resource configuration file for snmpdx; it contains timeout and polling parameters.

```
#ident "@(#)snmpd.snmprelay     1.3 25 Jan 1996 SMI"
#
# Copyright 25 Jan 1996 Sun Microsystems, Inc. All Rights Reserved
#  Configuration file of the SNMP Relay
#  for the SNMP agent bundled with SNM
#
##########
# agents #
##########

environment =
{
       max-agent-timeout = 999999999
       poll-interval = 240
}
```

Figure 6. snmpdx.rsrc

The **max-agent-timeout** parameter signifies the maximum allowed time-out a subagent may request during registration - this timeout is the number of microseconds that snmpdx will wait before it assumes that the subagent is not going to respond.

The **poll-interval** parameter specifies the number of seconds after which the master agent will perform housekeeping queries to subagents - for example, checking that the subagent is still alive.

Once snmpdx is initiated, it consults *all* other acl, reg and rsrc configuration files in /etc/snmp/conf and brings up each subagent according to those files.

mibiisa subagent

The subagent implemented by the executable mibiisa (but rather confusingly named snmpd), will handle queries for MIB-2 and the Sun MIB. The Sun MIB provides *system* information such as:

- CPU utilisation
- Disk transfers
- Swap & virtual memory performance information

```
#ident "@(#)snmpd.snmprelay      1.3 25 Jan 1996 SMI"
# Copyright 25 Jan 1996 Sun Microsystems, Inc. All Rights Reserved
#
#  Configuration file of the SNMP Relay
#  for the SNMP agent bundled with SNM
# mibiisa handles MIB-2 queries and Sun MIB queries
#
##########
# agents #
##########

agents =
{
      {
            name = "snmpd"
            subtrees = { mib-2, sun }
            timeout = 2000000
            watch-dog-time = 86400
      }
}
```

Figure 7. mibiisa.reg

```
################################################################
#  Access control file for Solaris mibiisa agent
# The list of community names needed for read/write access
#      to MIBs defined in mibiisa.reg - mib-2 and sun
# If the list is empty, the only valid community name is "public"
#     and its access type is read-only.
# snmpdx will check these authorisations, on behalf of mibiisa,
#    once mibiisa has registered with snmpdx

acl = {
        {
                communities = fraclmye
                access = read-write
                managers = localhost, poppet, blossom, sunshine
        }
        {
                communities = rrwatr
                access = read-only
                managers = localhost, poppet, blossom, sunshine
        }
}

# Traps do NOT go via snmpdx - they go straight to NetViews
trap = {
 {
   trap-community = rrwatr
   hosts = poppet, blossom, sunshine
 }
}
```

Figure 8. mibiisa.acl

By default, the SEA package does not include a mibiisa.acl. This means that access to those MIBs that mibiisa handles, will not be filtered by snmpdx - they will simply be passed straight to mibiisa. If the configuration file for mibiisa (snmpd.conf) does **not** permit access to it's MIB with the community name supplied, it never responds to snmpdx. This can result in snmpdx timing-out the mibiisa subagent and even deleting mibiisa from the snmpdx list of subagents. Thus all subsequent queries for MIB-2 and Sun MIBs will fail.

Creating a mibiisa.acl, as shown above, ensures that, once mibiisa has registered with snmpdx, community names for MIB-2 and Sun MIB requests will be checked at snmpdx and blocked before ever reaching mibiisa, if community names are wrong.

A corollary to this behaviour is that you are unlikely to see Authentication TRAPs from the mibiisa subagent if community names match in mibiisa.acl and snmpd.conf. This is because bad community names will simply be blocked at snmpdx (which **doesn't** issue an Authentication TRAP). If you wish to test this behaviour, add another "communities" stanza to mibiisa.acl with, say, read-only access with community name of "fred". A MIB-2 request from NetView on poppet with community name of "fred" will now be forwarded to mibiisa. If the mibiisa snmpd.conf file does **not** permit access with "fred", it will generate an Authentication TRAP.

Do ensure that mibiisa.rsrc (shown below) is modified to include a "security" statement.

```
#ident "@(#)snmpd.rsrc     1.3 25 Jan 1996 SMI"
#
# Copyright 25 Jan 1996 Sun Microsystems, Inc. All Rights Reserved
#
# Resource configuration file for mibiisa.  Port used by mibiisa is
ephemeral
# Note that the "-r" parameter has been removed from the command  line
#         to permit read-write access
# Also a security line has been specifically added to control the
access
#      from snmpdx to mibiisa - this helps prevent hang-ups on
timeouts

resource =
{
        {
                registration_file = "/etc/snmp/conf/mibiisa.reg"
                security = "/etc/snmp/conf/mibiisa.acl"
                policy = "spawn"
                type = "legacy"
                command = "/usr/lib/snmp/mibiisa -p $PORT"

        }

}
```

Figure 9. mibiisa.rsrc

A further quirk to watch for is that mibiisa.rsrc, as shipped, includes a "-r" parameter in the **command** statement that starts up the mibiisa subagent. This flag sets the whole subagent into read-only mode, even if correct community names are supplied. Removing the "-r" flag ensures that it is *possible* to perform SNMP SETs on some MIB variables controlled by mibiisa.

Beware! If editing this **command** statement, ensure that there is only one remaining space between "mibiisa" and "-p $PORT"; otherwise snmpdx seems to die off immediately!

The mibiisa subagent has it's own configuration file, **/etc/snmp/conf/snmpd.conf**, by default.

```
# Copyright 1988 - 09/23/99 Sun Microsystems, Inc. All Rights
Reserved.
#pragma ident  "@(#)snmpd.conf   2.23 99/09/23 Sun Microsystems"
#
sysdescr        Sun SNMP Agent, Ultra-5_10
syscontact      Jane Curry, Skills 1st Limited
sysLocation     System administrators office
#
system-group-read-community     rrwatr
system-group-write-community    fraclmye
#
read-community  rrwatr
write-community fraclmye
#
# All requests to mibiisa will come from snmpdx on localhost
managers        localhost
#
# mibiisa will send traps direct to NetViews, not via snmpdx
trap            poppet blossom sunshine
trap-community  rrwatr
```

Figure 10. snmpd.conf for mibiisa subagent

snmpd.conf only needs to specify community names for the manager **localhost** as all requests will come via the local snmpdx master agent. TRAPs, on the other hand, are sent directly to NetView managers.

TRAPs and the SEA agents

In common with many other SNMP agents and managers, community names on TRAP Protocol Data Units (PDUs) are not checked. Although they have been configured in the sample files above, the community name will be ignored.

The SEA documentation suggests that TRAPs can be forwarded from subagents to the snmpdx master agent. In practise, this is only possible for subagents that have been developed with Sun's SEA development toolkit. Legacy subagents such as mibiisa, net-snmp and mgragentd, must forward their TRAPs directly to network management station(s).

Controlling snmpdx

The snmpd master agent is controlled via the script in **/etc/init.d/init.snmpdx**.

```
#!/sbin/sh
#
# Copyright (c) 1997 by Sun Microsystems, Inc.
# All rights reserved.
#
#ident "@(#)init.snmpdx    1.12   97/12/08 SMI"
#
# -f 0 parameter added to "start" to prevent snmpdx killing
#    mibiisa on bad community name

case "$1" in
'start')
      if [ -f /etc/snmp/conf/snmpdx.rsrc -a -x /usr/lib/snmp/snmpdx ]; then
       /usr/lib/snmp/snmpdx -f 0 -d 4 -y -c /etc/snmp/conf > /tmp/snmpdx.log 2>&1 &
#            /usr/lib/snmp/snmpdx -f 0 -y -c /etc/snmp/conf
      fi
      ;;

'stop')
#     /usr/bin/pkill -9 -x -u 0 '(snmpdx|snmpv2d|mibiisa)'
      /usr/bin/pkill -9 -x -u 0 '(snmpdx|snmpv2d|mibiisa|snmpd)'
      ;;

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

Figure 11. snmpdx control file, /etc/init.d/init.snmpdx

The sample above has been modified in 3 ways from the default as shipped:

- a “-f 0” has been added to the startup of snmpdx. This is an undocumented parameter which ensures that subagents do not get deleted from snmpdx’s table of subagents if bad community names are used. Check the SEA 1.0.3 Release Notes for further information. **Note: This parameter seems to significantly improve stability of snmpdx.**
- An alternative line has been added for snmpdx startup which will turn on debugging and write output to /tmp/snmpdx.log. Ensure that only one startup line is uncommented!
- An alternative line has been added to stop snmpdx. It will kill off net-snmp subagent processes in addition to the SEA subagents.

Debugging can be turned on at different logging levels where 0 represents no debugging (the default) and 4 represents the highest level. The latter is very useful in understanding the master / subagent registration process and authentications, and to see the detail of all SNMP conversations.

The net-snmp Agent

net-snmp is an extensible SNMP agent that has evolved from the work done at Carnegie Mellon University (CMU) and the subsequent SNMP agent developed at University of California at Davis (UCD) - see **http://www.net-snmp.com**. Recently, the UCD development has metamorphosed into net-snmp.

It is publicly available from **http://sourceforge.net/project/net-snmp**. This paper was written using Version 4.2.1 - 4.2.2 has just (October 10th, 2001) become available. The FAQ is a good place to start for information (**http://www.net-snmp.com/FAQ.html**).

Some advantages of this agent include:

- Support for Solaris, AIX, HP-UX, Ultrix, Linux and several other Unixes (it is standard with Red Hat Linux)
- Some support is available for Windows NT
- Support for SNMP V1, V2 and V3 (the FAQ has some good information on these different versions)
- SNMP libraries and SNMP tools are included (snmptrap, snmpget, etc)
- Support for MIB-2, UCD agent extensions and Host Resources MIB (RFC 1514)
- Standardised support for extensible agents - SMUX (RFC 1227), AgentX (RFC 2741)
- Pass-through support permits commands or shellscripts to be run in response to MIB queries
- The SNMP agent is configurable with regard to ports used, and can utilise either UDP or TCP
- UCD MIB extensions offer CPU, disk space and process monitoring (the Host Resources MIB offers many more *system* management variables)
- Sample configuration file with LOTS of examples
- Good track-record

Building the net-snmp Agent

Once the source code has been downloaded, it is a relative simple process to build the agent. The GNU gcc compiler is recommended. To include support for SMUX, AgentX and the Host Resources MIB, use the following commands:

configure --with-mib-modules=agentx,smux,host
make
umask 022
make install

The agent binary will be in **/usr/local/sbin/snmpd** by default. The **snmpd.conf** configuration file should be in the **/usr/local/share/snmp** directory.

Note that later versions of SNMP require stronger authentication methods so Secure Sockets Library (SSL) support will be required, and access to SSL libraries will be necessary for the snmpd agent executable.

Customising the net-snmp Agent

The net-snmp agent has a file called **snmpd.conf** in **/usr/local/share/snmp**, by default.

```
# Configuration file for the UCD SNMP agent
#
###############################################################################
# Access Control
###############################################################################
# By default, the agent responds to the "public" community for read
# only access, if run out of the box without any configuration file in
# place.
####
# First, map the community name (COMMUNITY) into a security name
#       sec.name              source          community
com2sec localhost_read        localhost      rrwatr
com2sec localhost_write       localhost      raclmye
com2sec sunshine_read         sunshine       rrwatr
com2sec sunshine_write        sunshine       fraclmye
com2sec poppet_read           poppet         rrwatr
com2sec poppet_write          poppet         fraclmye
com2sec blossom_read          blossom        rrwatr
com2sec blossom_write         blossom        fraclmye

####
# Second, map the security names into group names:
#                              sec.model  sec.name
group sunshine_read_group     v1         sunshine_read
group sunshine_write_group    v1         sunshine_write
group poppet_read_group       v1         poppet_read
group poppet_write_group      v1         poppet_write
group blossom_read_group      v1         blossom_read
group blossom_write_group     v1         blossom_write
group localhost_read_group    v1         localhost_read
group localhost_write_group   v1         localhost_write

####
# Third, create a view for us to let the groups have rights to:
#           incl/excl subtree                            mask
view all    included  .1                                 80

####
# Finally, grant the groups access to the 1 view with different
# write permissions:
#                       context sec.model sec.level   match  read   write  notif
access sunshine_read_group ""      any       noauth    exact  all    none   none
access sunshine_write_group ""     any       noauth    exact  all    all    none
access poppet_read_group ""        any       noauth    exact  all    none   none
access poppet_write_group ""       any       noauth    exact  all    all    none
access blossom_read_group ""       any       noauth    exact  all    none   none
access blossom_write_group ""      any       noauth    exact  all    all    none
access localhost_read_group ""     any       noauth    exact  all    none   none
access localhost_write_group ""    any       noauth    exact  all    all    none

# You can modify the port used here, or use the "-p" parameter when calling
#    the snmp executable
# agentaddress udp:161
# agentaddress 1000
#
# Trap section
# Changing authtrapenable to 2 (default) disables authentication traps
authtrapenable 1
#
trapcommunity rrwatr
#
# trapsink has format:
#  trapsink <host> [ <community>  [<port> ]
# The default trap port is UDP 162
trapsink poppet
trapsink blossom
trapsink sunshine
# -----------------------------------------------------------------------------
```

Figure 12. net-snmp configuration file (part 1)

```
##############################################################################
# System contact information
#
# It is also possible to set the sysContact and sysLocation system
# variables through the snmpd.conf file:
syslocation Cedar Chase
syscontact Jane Curry
##############################################################################
# disk checks
# The agent can check the amount of available disk space, and make
# sure it is above a set limit.
# disk PATH [MIN=DEFDISKMINIMUMSPACE]
#
# PATH:  mount path to the disk in question.
# MIN:   Disks with space below this value will have the Mib's errorFlag set.
#        Default value = DEFDISKMINIMUMSPACE.
# Check the /tmp partition and make sure it contains at least 725 megs.
disk /tmp 725000
##############################################################################
# load average checks
# load [1MAX=DEFMAXLOADAVE] [5MAX=DEFMAXLOADAVE] [15MAX=DEFMAXLOADAVE]
# 1MAX:   If the 1 minute load average is above this limit at query
#         time, the errorFlag will be set.
# 5MAX:   Similar, but for 5 min average.
# 15MAX:  Similar, but for 15 min average.
# Check for loads:
#load 12 14 14
```

Figure 13. net-snmp configuration file (part 2)

This agent can be run stand-alone *instead* of the Solaris SEA agent. By default it will listen on UDP 161 for SNMP GETs and SETs and will issue TRAPs on UDP 162 (but both are configurable).

The net-snmp agent is started via:

/usr/local/sbin/snmpd -p <port no> the port number will be 161 by default

To start the agent with debugging turned on, to a logfile **/tmp/snmp_ucd.log**, use:

/usr/local/sbin/snmpd -d -l /tmp/snmp_ucd.log

Combining the Sun SEA agent and the net-snmp agent

It is possible to achieve the best of both worlds with these 2 agents. The net-snmp agent can replace the Sun mibiisa agent, handling queries for MIB-2, UCD enterprise-specific MIBs and MIB queries for system information from the Host Resources MIB.

The SEA snmpdx master agent doesn't use either SMUX or AgentX to communicate with it's subagents. It is effectively a "pass-through" mechanism.

Simply remove the mibiisa files and snmpd.conf from /etc/snmp/conf to a directory for safekeeping, and create a set of files for the net-snmp agent, eg. **ucd.acl, ucd.reg** and **ucd.rsrc**. The **snmpdx** customisation files can remain unchanged.

```
################################################################
#  Registration configuration file for UCD SNMP agent
#  The UCD agent handles queries for MIB-2 and UCD MIBs
#  It runs on port 1161
#

macros =
{
ucd = 1.3.6.1.4.1.2021
host = 1.3.6.1.2.1.25
}

agents =
{
       {
              name = "ucd-snmp"
              subtrees = { ucd, mib-2 }
              timeout = 200000
              watch-dog-time = 86400
              port = 1161
       }
}
```

Figure 14. Registration file for net-snmp agent - ucd.reg

The registration file specifies that this agent, known to snmpdx as **ucd-snmp**, will handle MIB queries for MIB-2 and the UCD enterprise. MIB queries for the Host Resources MIB are a subtree of MIB-2.

A crucial configuration here is that the net-snmp agent will listen for MIB requests on port UDP 1161 (snmpdx already has the traditional UDP 161).

```
################################################################
#  Access control file for UCD SNMP agent
#
# The list of community names needed for read/write access
#      to the MIB defined in ucd.reg - mib-2 and UCD
# If the list is empty, the only valid community name is "public"
# and its access type is read-only.
# snmpdx will check these authorisations on behalf of UCD snmp,
#     once UCD snmp has registered with snmpdx

acl = {
        {
                communities = fraclmye
                access = read-write
                managers = localhost, sunshine, poppet, blossom
        }
        {
                communities = rrwatr
                access = read-only
                managers = localhost, sunshine, poppet, blossom
        }
}

trap = {
 {
   trap-community = rrwatr
   hosts = poppet, blossom, sunshine
 }
}
```

Figure 15. Access Control File for net-snmp agent - ucd.acl

```
#################################################################
#  Resource file for UCD SNMP agent
# Note that the command line must specify the correct port number
#    for the UCD agent to run on.  It must match that specified in ucd.reg


resource =
{
      {

            registration_file = "/etc/snmp/conf/ucd.reg"
            security = "/etc/snmp/conf/ucd.acl"
            policy = "spawn"
            type = "legacy"
            command = "/usr/local/sbin/snmpd -d -l /tmp/snmp_ucd.log -p 1161"
      }
}
```

Figure 16. Resource Configuration File for net-snmp agent - ucd.rsrc

The resource configuration file includes the command to startup the net-snmp agent, including the port that it will listen on - this must match the configuration in ucd.reg. This example also shows the agent started with debugging turned on ("-d" flag), to a logfile /tmp/snmp_ucd.log.

Simply stopping and starting the snmpdx agent via:

/etc/init.d/init.snmpdx stop

/etc/init.d/init.snmpdx start

should startup the snmpdx master agent with the net-snmp agent as a registered subagent.

Integrating the Combined SEA / net-snmp Agents with Tivoli NetView

Tivoli NetView is the network management suite within the Tivoli family of enterprise management products. A basic prerequisite for installation of NetView, is to have an SNMP agent that responds at least to MIB-2 queries, on port UDP 161. During installation, NetView will test for an snmp agent using:

ps -ef | grep snmpd

This test should be successful whichever combinations of the above agents you are running; however if no "snmpd" process is found, the installation process will try to start snmpdx.

******** WARNING ************ WARNING ************* WARNING *****

If you have been following best practices and configured your snmp agent(s) not to respond to a read-only request with a community name of **public**, your NetView installation will **FAIL**. This is because the installation process attempts to start the NetView processes in order to complete customisation. Some of the NetView daemons need to communicate with the local SNMP agent. By default (which is all you can possibly have at this stage!), NetView will use a community name of **public**. You **MUST** ensure that your local SNMP agent will permit read-only access with **public** (you can change it to something safer once NetView installation is complete and it's community files can also be modified to keep in synchronisation).

Before performing a NetView installation, run the **NVprereq.sh** prerequisite check script that can be found under the **TOOLS** directory of the NetView CD. This checks that an SNMP daemon exists but it is does **not** test community name access.

The NetView installation process ensures that NetView is started up after a system reboot by adding various files into the **/etc/rc.d** directories. Each of these files is a symbolic link to **/etc/init.d/netnmrc**.

Note!! It appears to take a little time for snmpdx to initialise the net-snmp agent and that the net-snmp agent may not be running by the time NetView's netmon process tries to start (so netmon dies). If this happens, netmon can easily be started subsequently from the command line. Ideally, the startup scripts should be slightly modified to ensure net-snmp is running before netmon attempts to start.

Customising NetView for Solaris SNMP agents

Once NetView is installed, probably one of the first things to do is to improve security on community names. For the agent(s), this is as discussed earlier. For NetView, take the **Options -> SNMP Configuration** menus from the NetView GUI and create an entry for both the Domain Name Service (DNS) name of the NetView system, and also an entry for loopback, ensuring that these match the agent customisations. Don't forget to click on "OK"!

The NetView installation process installs a daemon called **mgragentd** which handles communication between NetView Server processes and NetView Client processes. mgragentd registers as a subagent to the master snmpdx agent and has the usual three types of files created in /etc/snmp/conf.

```
#  Configuration file for TME-10 NetView


macros =
{
mgragentd = 1.3.6.1.4.1.2.6.4.6
}

agents =
{
       {
              name = "mgragentd"
              subtrees = { mgragentd }
              timeout = 3000000
              watch-dog-time = 31536000
              port = 1670
       }
}
```

Figure 17. Registration file for mgragentd - mgragentd.reg

The mgragentd is started automatically with other NetView processes, on port 1671. Queries from the mgragentd enterprise, 1.3.6.1.4.1.2.6.4.6, are forwarded through snmpdx to mgragentd.

```
#  Access control file for TME 10 NetView mgragentd

# The list of community names needed for read/write access
# to the mgragentd MIB.
# If the list is empty, the only valid community name is "public"
# and its access type is read-only.

acl = {
        {
                communities = fraclmye
                access = read-write
                managers = localhost, sunshine, poppet, blossom
        }
        {
                communities = rrwatr
                access = read-only
                managers = localhost, sunshine, poppet, blossom
        }
}

trap = {
}
```

Figure 18. Access Control file for mgragentd - mgragentd.acl

The access control file for mgragentd is shipped with community names of **public** for read-only and **private** for read-write, with **managers = *.** From inspecting the snmpdx log files, it would appear that the *only* authorisation for mgragentd is at *snmpdx* via the mgragentd.acl file. This file could reasonably be customised with the same values as general MIB-2 access. There is no customisation option in NetView to modify community names used by the mgragentd subagent. The mgragentd MIB can be inspected and access tested from the NetView MIB Browser by following:

private -> enterprises -> ibm -> ibmProd -> netView6000SubAgent -> mgrAgentd

```
#  Resource file for TME-10 NetView mgragentd SNMP daemon


resource =
{
       {

              registration_file = "/etc/snmp/conf/mgragentd.reg"
              security = "/etc/snmp/conf/mgragentd.acl"
              policy = "spawn"
              type = "legacy"
              command = "/usr/bin/echo mgragentd registered with snmpdx"
       }
}
```

Figure 19. Resource Configuration file for mgragentd - mgragentd.rsrc

The mgragentd.rsrc has a dummy command line that simply echos a message. mgragentd is actually started with other NetView processes.

By default, NetView creates a SmartSet (collection) of NetViews. Membership of this SmartSet depends on the “isManager” flag being set for a node in the NetView object database. The “isManager” flag depends on communication with the mgragentd subagent so invalid community names in mgragentd.acl will result in the Node **not** being recognised as a NetView system. It will also prevent the NetView Backup facility from working correctly.

Customising appearance of Solaris agents in NetView

If using NetView with the default snmpdx / mibiisa combination, NetView will display the local Solaris system as a "Workstation" icon, based on the mibiisa agent reporting that it's Object ID (OID) is 1.3.6.1.4.1.42.2.1.1 ie. the basic SunOS agent. The NetView file **/usr/OV/conf/C/oid_to_sym** provides the mapping between agent OIDs and the symbols that represent those agents.

The net-snmp package provides an **ov** directory with files to customise NetView for the net-snmp agent. A README is included but this is specific to HP OpenView; customisation for NetView is provided below.

1. Change to **/usr/OV/fields/C** and take a copy of **ovw_fields** and **snmp_fields** outside the /usr/OV/fields/C directory hierarchy

2. Edit **/usr/OV/fields/C/ovw_fields** and add to the "Vendor" Enumeration list the entry found in **ov/UCD-fields** for "UCDavis". It is recommended that these entries be in alphabetical order. Be very careful with the syntax at the end of the lines in this file.

3. Edit **/usr/OV/fields/C/snmp_fields** and add to the "SNMPAgent" Enumeration list the entries found in **ov/UCD-fields** for each of the UCD agents. It is recommended that these entries be in alphabetical order. Be very careful with the syntax at the end of the lines in this file.

4. Modify the oid_to_type capabilities file

   A. Change to **/usr/OV/conf** and take a copy of **oid_to_type**
   B. Edit **oid_to_type** and add to the end of the file, the lines found in **ov/oid_to_type**

5. Modify the oid_to_sym symbol-mapping file

   A. Change to **/usr/OV/conf/C** and take a copy of **oid_to_sym**
   B. Edit **oid_to_sym** and add to the end of the file, the lines found in **ov/oid_to_sym**

6. Add the UCD subclass to the Computer class definition:

   A. Change to **/usr/OV/symbols/C** and take a copy of **Computer** outside the /usr/ OV/symbols/C directory hierarchy
   B. Edit **/usr/OV/symbols/C/Computer** and append to the end of the file, the lines found in **ov/UCD-Computer**

7. Setup the UCD bitmap files:

   A. Copy all the ".p" and ".m" files from **ov/bitmaps** to **/usr/ OV/bitmaps/C**
   B. Compile the changes
      - ovw -fields

8. You will need to stop and restart the NetView GUI before you see these new symbols

9. You will probably also need to stop and restart the NetView daemons before the changes to oid_to_type take effect

10. From the GUI, check **Help -> Legend** to see your new bitmap

11. You will need to Demand Poll a device running the UCD agent, before it's symbol changes. Alternatively a Configuration Poll (every 24 hours by default), will achieve this. If all else fails, delete the node from the NetView object database and it should have the UCD icon when it is rediscovered.

Loading MIBs into NetView for Solaris agents

If you run with the SEA snmpdx / mibiisa combination, the SNMP V1 Sun MIBs should be loaded into NetView. There are 2 MIBs, **snmpdx.mib** and **sun.mib** which can be found on the Solaris system under **/var/snmp/mib**. If using NetView V7.1, from the NetView GUI, select the **Options -> Load / Unload SNMP V1 MIBs** menu option. Check values using **Tools -> MIB Browser SNMP v1**. For earlier versions of NetView, the menu paths will be **Options -> Load / Unload MIBs -> SNMP** for loading, and **Tools -> MIB Browser -> SNMP** to view.

If you deploy the combined snmpdx / net-snmp agent you will want to load the UCD MIBs which are SNMP V2 MIBs. With NetView V7.1, there is a new application to load SNMP V2 MIBs under the **Tools -> Web Console MIB Loader** menu. The MIB source *must* be located in /usr/OV/www/mibs and must have a filename suffix of ".mib". The UCD MIBs are part of the net-snmp package under the **mibs** directory. SNMP V2 MIBs can only be viewed through the NetView Web Interface with the **Tools -> MIB Browser** menus.

With NetView prior to V7.1, load V2 MIBs via **Options -> Load / Unload MIBs -> SNMP v1/SNMP v2** and view them via **Tools -> MIB Browser -> SNMP v1 / SNMP v2.**

Introducing Mid Level Manager (MLM) into this Scenario

The Tivoli NetView Mid Level Manager code is shipped at no extra cost, with NetView. Versions are available for AIX, HP-UX, Solaris and NT. The purpose of an MLM is to offload from a central NetView, one or more of:

- Status polling
- Node discovery
- Polling for performance data
- Local reception of SNMP TRAPs

Typically, an MLM will be installed on a departmental system to handle local management; there is also sometimes a good argument for installing MLM on the same system as NetView -

perhaps to do local status polling, trap filtering and/or collection of performance data.

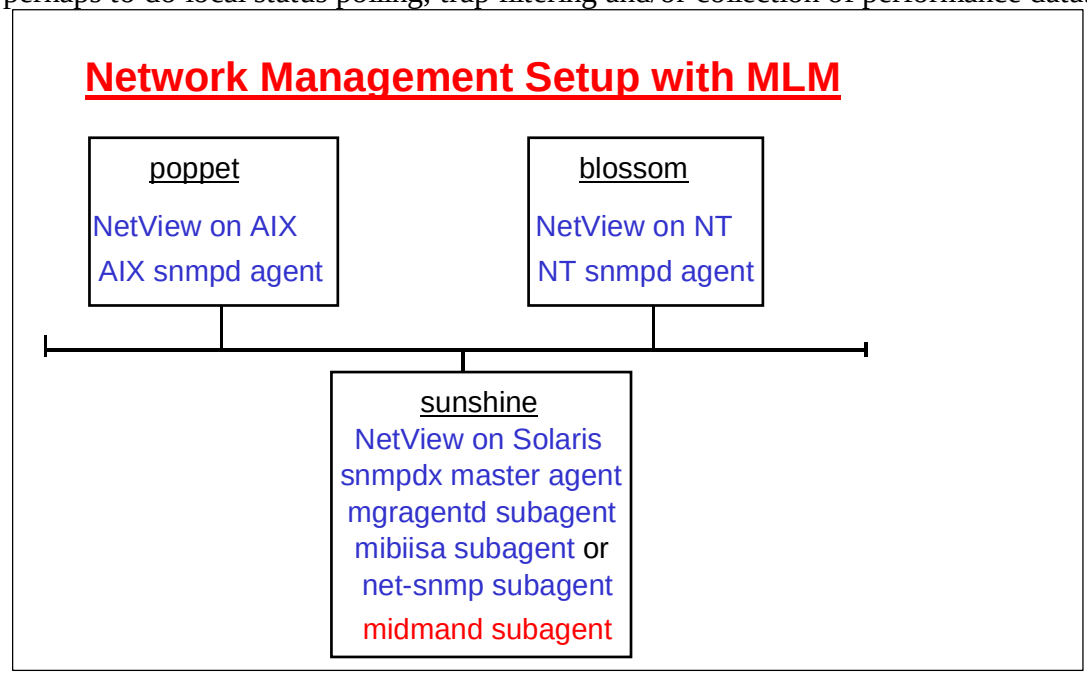

Figure 20. Network Management Scenario with MLM

A Solaris MLM can be installed either remotely from the NetView Server icon on a Tivoli Desktop, or it can be installed locally through command line methods. If using the remote install from the Tivoli Desktop, ensure that the file **/etc/ftpusers** permits root to logon via ftp (ie. root should **not** be listed in /etc/ftpusers). The main MLM executable is **midmand**.

There is a little confusion for the unwary in that the MLM installation also introduces new versions of the following SNMP utilities:

- snmpget
- snmpset
- snmpnext
- snmptrap
- snmpwalk

These files overwrite existing files in **/usr/bin** (these may well be your net-snmp versions of the same utilities!). Further, the manual pages for the MLM versions go under **/usr/man/cat** (whereas the net-snmp man pages are typically under **/usr/local/man**). Depending on your PATH and MANPATH variables, you can end up with man pages for net-snmp but MLM binaries. (This isn't a huge issue, but you do need to get things in synchronisation!). As an additional confusion, *NetView* also has versions of the above 5 SNMP tools in **/usr/OV/bin**. One way to find out if you have the MLM binaries is to simply run the command with no arguments - it will echo a message saying it was distributed with MLM.

The MLM installation process adds the usual three configuration files for midmand to **/etc/snmp/conf**.

```
#
# COMPONENT_NAME: midmand.reg
#
# Licensed Program Product: Tivoli NetView MLM for Solaris
#
# (C) COPYRIGHT International Business Machines Corp. 1997
# All rights reserved
# Licensed Material - Property of IBM
#
# US Government Users Restricted Rights - Use, duplication or
#
# disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
################################################################
#  Configuration file for Tivoli NetView MLM
macros =
{
ibm = 1.3.6.1.4.1.2
sysmon = 1.3.6.1.4.1.2.6.12
}

agents =
{
      {
            name = "midmand"
            subtrees = { sysmon.1.1, sysmon.1.2, sysmon.1.3, sysmon.1.4,
sysmon.3, sysmon.5, sysmon.6, sysmon.7, sysmon.8, sysmon.9, sysmon.10,
sysmon.12, sysmon.13 }
            timeout = 30000000
               watch-dog-time = 31536000
            port = 1668
      }
}
```

Figure 21. Registration file for midmand - midmand.reg

The Mid Level Manager executable, **midmand**, registers for subtrees of the IBM sysmon MIB (sysmon was an earlier name for Mid Level Manager). midmand will run on port 1668.

```
# COMPONENT_NAME: midmand.acl
#
# Licensed Program Product: Tivoli NetView MLM for Solaris
#
# (C) COPYRIGHT International Business Machines Corp. 1997
# All rights reserved
# Licensed Material - Property of IBM
#
# US Government Users Restricted Rights - Use, duplication or
#
# disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
#############################################################
#  Access control file for Tivoli NetView MLM
# The list of community names needed for read/write access
# to the parts of the MLM MIB - see midmand.reg
# If the list is empty, the only valid community name is "public"
# and its access type is read-only

acl = {
        {
                communities = fraclmye
                access = read-write
                managers = localhost, poppet, blossom, sunshine
        }
        {
                communities = rrwatr
                access = read-only
                managers = localhost, poppet, blossom, sunshine
        }
}

trap = {
}
```

Figure 22. Access Control file for midmand - midmand.acl

As shipped, midmand.acl simply has the following community stanza:

```
communities = public, private
access = read-write
managers = *
```

The NetView Release Notes suggest changing this to 2 separate stanzas, one for read-only access with **public** and one for read-write with **private**. For a security minded organisation, any community name of **public** is anathema. The sample shown in the box above works fine.

The midmand daemon itself uses the *NetView* configuration file **/usr/OV/conf/ovsnmp.conf** to decide what community names to use. If there is no /usr/OV/conf/ovsnmp.conf (because the MLM is installed on a non-NetView system), midmand will check for an **ovsnmp.conf** in **/var/adm/smv2** . Basically, your ovsnmp.conf and your midmand.acl must agree.

```
#
# COMPONENT_NAME: midmand.rsrc
#
# Licensed Program Product: Tivoli NetView MLM for Solaris
#
# (C) COPYRIGHT International Business Machines Corp. 1997
# All rights reserved
# Licensed Material - Property of IBM
#
# US Government Users Restricted Rights - Use, duplication or
#
# disclosure restricted by GSA ADP Schedule Contract with IBM Corp.
#
#
############################################################

#  Resource file for Tivoli NetView MLM

resource =
{
       {
              registration_file = "/etc/snmp/conf/midmand.reg"
              security = "/etc/snmp/conf/midmand.acl"
              policy = "spawn"
              type = "legacy"
              command = ""
       }
}
```

Figure 23. Resource configuration file for midmand - midmand.rsrc

The snmpdx daemon is not generally expected to initiate the midmand daemon, hence the dummy command parameter.

The Mid Level Manager is generally controlled via the **/usr/etc/smmlm** shellscript, which takes various parameters to start, stop and show status of the midmand daemon.

There is one small remaining problem with changing community names away from **public**. The script that starts the midmand, **/usr/etc/smmlm**, checks that the midmand process has started and, if it isn't yet registered with snmpdx but was started independently (ie **not** by snmpdx resource files), then it send a SIGHUP signal to snmpdx to make it re-read all it's configuration files. OK so far - this method ensures a user can manually stop and start the MLM. Unfortunately, the code that does the "is midmand registered with snmpdx" check uses the *MLM* snmpwalk binary and hardcodes a community name of **public**!!!! The neatest way to get around this is simply to modify **/usr/etc/smmlm** to omit the community name parameter and to use the *NetView* snmpwalk binary, rather than the MLM binary as the NetView snmpwalk consults **/usr/OV/conf/ovsnmp.conf**. Obviously this only works where an MLM is on a system that also has NetView installed. Otherwise you will need to hardcode a community name in smmlm or, perhaps, have it look up in MLM's **/var/adm/smv2/ovsnmp.conf**.

```
  .....

#Check if midman was indeed able to start
midmand_pid=$(/usr/bin/ps -e | grep " midmand$" | awk '{print $1}')

#Send SIGHUP to snmpdx if:
#   midmand is now running and
#   this script was not executed by snmpdx and
#   midmand is not in snmpdx's list of daemons
if [ -n "$snmpdx_pid" ]; then
  if [ -n "$midmand_pid" -a $PPID != "$snmpdx_pid" ]; then
midmand_state=`/usr/OV/bin/snmpwalk 127.0.0.1 .1.3.6.1.4.1.42.2.15.8.1.10 | grep midmand`
#midmand_state=`/usr/bin/snmpwalk 127.0.0.1 public .1.3.6.1.4.1.42.2.15.8.1.10 | grep midmand`
    if [ -z "$midmand_state" ]; then
      kill -1 $snmpdx_pid
    fi
  fi
fi

exit
```

Figure 24. End of /usr/etc/smmlm file

The code snippet above shows the original line commented out and the active line using NetView's snmpwalk command with no community name supplied.

By default on installation, a file **/etc/rc3.d S99smmlm.ibm** is created which ensures that the MLM is started at system reboot. There is a corresponding file, **K10smmlm.ibm** which stops the MLM when a system is taken down.

Summary

Combining SNMP agents and managers on a Solaris system is complex but perfectly achievable. Either the Sun-supplied snmpdx / mibiisa agents can be used, or the mibiisa functionality can be replaced with other SNMP agents, such as net-snmp, to provide responses to MIB-2 queries, and to extended functionality, particularly if any *system* management is required. An alternative MIB-2 subagent **must** be configurable to use a port other than UDP/161.

snmpdx should **always** be run as the master SNMP agent as NetView's mgragentd and the Mid Level Manager midmand expect to use the snmpdx master / subagent mechanism. snmpdx, always occupies the SNMP port UDP/161. It receives requests from network managers and, based on registration files, decides whether to handle those queries itself or whether to pass the requests to registered subagents. snmpdx and each subagent has an access control file that specifies what community names are acceptable and from which managers. These files **only** relate to MIB queries from that part of the MIB tree referenced in the associated subagent registration file. Good practise suggests that access with a community name of **public** should be prohibited and this is possible for all scenarios.

NetView can be installed on top of either master / subagent combination but it is essential that access is permitted to MIB-2 with a community name of **public**, during installation. With care, Mid Level Managers can also be installed on top of either master / subagent combination.