

Comdasys FAQ: SNMP

Comdasys AG
Rüdesheimer Str. 7
D-80686 München
Tel.: +49.89.5484333-0
Fax: +49.89.5484333-29

support@comdasys.com

<http://www.comdasys.com>

Disclaimer

We have taken all possible care to ensure that this manual contains correct, accurate information. However, the manufacturer cannot assume liability for any possible errors. In addition, the manufacturer cannot guarantee that the hardware will meet the purpose you require.

Comdasys reserves the right to make changes according to technical progress at any time. Brand names may be registered trademarks and must be treated as such.

© Copyright 2005-2009
Comdasys AG
80686 München, Germany

All rights reserved. No part of this manual may be reproduced, processed or distributed in any form (print, photocopy, microfilm or any other process) or processed by an electronic system without prior written permission from the manufacturer.

Content

1. Synopsis.....	4
2. General.....	4
3. Reacting on Traps.....	4
4. Writing SNMP traps.....	5
4.1. Creating a MIB file.....	5
4.2. Creating the trap.....	6
4.3. Making the trap start automatically.....	7
5. Monitoring.....	7
6. Survivability Mode Traps.....	8
6.1. Creating a Sample Trap.....	8
6.2. Creating a Sample Trap.....	8

1. Synopsis

This document concentrates on the usage of Comdasys products as SNMP Trap daemons. All of our Convergence and Mobile Convergence products can be configured to send and forward SNMP Traps via their web interfaces. Find this SNMP facility in the Diagnostics tab of the GUI.

It is also possible to use your Comdasys device to receive and react to SNMP Traps from other devices. However, this is only possible via the command line interface (CLI). This document is therefore directed at administrators who have familiarity with the CLI, as well as basic knowledge of SNMP.

2. General

There is a status page on each Comdasys product which shows the status of all services. If a service is running it is marked with a green LED, if it is currently inactive it will be gray.

Both SNMP daemons, the SNMP Daemon (for sending and forwarding Traps), as well as the SNMPTRAPD Daemon (for the reception of and reaction to Traps) are displayed there (Diagnostics). As soon as the SNMPTRAPD component is started via the CLI, the respective LED on the status page will turn green after a restart of the services.

3. Reacting on Traps

The configuration file for SNMPTRAPD Daemon is located at: `/etc/snmptrapd.conf`. It has the following format:

```
traphandle .1.3.6.1.4.1.2021.13.990.0.17 \  
/bin/touch /tmp/yes  
traphandle default /tmp/trapthetrap
```

At first you have to make sure that the trap daemon is running. This can be accomplished by making sure that the `/etc/sysconfig/goodies` file contains the following line:

```
ENABLE_SNMP=yes
```

This line will make sure that the SNMP Daemon and the Trap Handling Daemon will come up after the next reboot. To start the daemons quickly just type in:

```
restartservices
```

If you have at least basic knowledge of SNMP, the syntax for this file is quite clear. The number in the first line is the SNMP OID (object identifier). The second line uses the keyword `default` to match with all traps. The semantic within this file is first match. For every trap, there will be exactly *one* match.

The second parameter in this line refers to a helper script, that will help you find out the OID of a trap in case you do not know it. It will catch all traps and log the trap OID into a file. In order to accomplish this, create the following script in `/tmp`

```
#!/bin/sh

read host
read ip
vars=

while read oid val
do
  if [ "$vars" = "" ]
  then
    vars="$oid = $val"
  else
    vars="$vars, $oid = $val"
  fi
done

echo trap: $1 $host $ip $vars >>/tmp/trap.log
```

First you have to make sure that the script above is executable. This is done by invoking:

```
chmod 755 /tmp/trapthetrap
```

Now send the trap to the Comdasys product from wherever you want to send it. The trap OID can then be found in /tmp/trap.log file. Use this number and enter it in the snmptrapd.conf to capture traps from a specific object, and to invoke a specified reaction on it.

4. Writing SNMP traps

This article describes how to write a custom SNMP trap on a Comdasys product. We assume you have basic knowledge about SNMP and Linux.

4.1. Creating a MIB file

Before we can write a trap we need to create a MIB (management information base) file in /usr/share/snmp/mibs (we'll name ours TRAP-TEST-MIB.txt):

```
TRAP-TEST-MIB DEFINITIONS ::= BEGIN
  IMPORTS ucdExperimental FROM UCD-SNMP-MIB;

  demotraps OBJECT IDENTIFIER ::= { ucdExperimental 990 }

  demo-trap TRAP-TYPE
    STATUS current
    ENTERPRISE demotraps
    VARIABLES { sysLocation }
    DESCRIPTION "This is just a demo"
    ::= 17
```

```
END
```

4.2. Creating the trap

Our next step is writing the trap. The goal in this example is to monitor a process (in this case "ser") and send a trap when the process is not running any more. It is not desirable to have constant traps sent when the process is not running. A trap should only be sent once we can't solve the problem with a cron (a Chronograph, which is a job scheduler for commands and shell scripts), but have to write a script that will run in the background.

We want the trap script to be run automatically, which is why we'll make use of the `/etc/rc.custom` directory

The next section contains more about the above directory, but now we create our trap script `/etc/rc.custom/exampletrap`:

```
#!/bin/bash

PROCESSTOWATCH=ser
HOSTTONOTIFY=10.0.0.197
PROCESSDOWN=

while true ; do

    TMPDOWN=`pidof $PROCESSTOWATCH`
    if [ ! "$TMPDOWN" = "$PROCESSDOWN" ] ; then
        PROCESSDOWN=$TMPDOWN

        if [ -z "$PROCESSDOWN" ] ; then
            snmptrap -v 1 -c public $HOSTTONOTIFY TRAP-TEST-
MIB::demotraps ' 6 17 ' SNMPv2-MIB::sysLocation.0 s "Process
went down"
            fi
            fi

        sleep 5

    done
```

After this script is written we need to make it executable with by calling:

```
chmod +x /etc/rc.custom/exampletrap
```

The important line is the one with the `snmptrap` command. Note that everything between `snmptrap` and "Process went down" is one line!

More about the `snmptrap` command and its usage can be found on the net-SNMP homepage and their SNMPTrap Tutorial on which the above code is based:

<http://net-snmp.sourceforge.net/wiki/index.php/Tutorials>

4.3. Making the trap start automatically

Since r671 Convergence products support custom *rc scripts* that are started during boot time and stopped on shutdown. These scripts reside in `/etc/rc.custom`, the `README` file in this directory explains the usage of this directory.

To make our trap start during boot time we need to write an rc script, which we'll name `/etc/rc.custom/rcexampletrap`:

```
#!/bin/bash

case "$1" in
  start) killall -0 exampletrap &>/dev/null
        if [ $? -gt 0 ] ; then
            /etc/rc.custom/exampletrap &
        fi
        ;;
  stop)  killall -9 exampletrap &>/dev/null
        ;;
esac
```

Then we need to make the rc script executable and create the necessary links:

```
chmod +x /etc/rc.custom/rcexampletrap
cd /etc/rc.custom
ln -s rcexampletrap S01rcexampletrap
ln -s rcexampletrap K99rcexampletrap
```

5. Monitoring

With the following bash script TRAPs will be sent if the number of configured users falls under a preconfigured limit.

Add the file `/etc/cron.d/check_registered` with the following content:

```
*/10 * * * * /etc/customscripts/check_registered.sh
```

Then add another file to `/etc/customscripts/check_registered.sh` with this content:

```
#!/bin/bash
HOSTTONOTIFY= 10.0.0.197
SERACTUAL=`/usr/sbin/openssl online | wc -l | tr -d " "`
SEREXPECTED=4
if [ $SERACTUAL -lt $SEREXPECTED ]
then
snmptrap -v 1 -c public $HOSTTONOTIFY \
```

```
TRAP-SURV-MIB::survtraps \
'' 6 18 '' SNMPv2-MIB::sysLocation.0 s \
"Only $SERACTUAL from $SEREXPECTED numbers registered"
fi
```

Just like the other scripts above, this script has to be made executable with this line:

```
chmod +x /etc/customscripts/check_registered.sh
```

After this is done traps will be sent to 10.0.0.197 every 10 minutes if less than 4 numbers are registered!

6. Survivability Mode Traps

In some cases it is desirable to have the Convergence send traps for switching from/to survivability mode. A single trap will then be sent for every mode change.

6.1. Creating a Sample Trap

Two traps are needed to make this function work properly. In the example below these two traps are saved to /usr/share/snmp/mibs (the name of the file is: TRAP-SURV-MIB.txt):

```
TRAP-SURV-MIB DEFINITIONS ::= BEGIN
    IMPORTS ucdExperimental FROM UCD-SNMP-MIB;

    survtraps OBJECT IDENTIFIER ::=
    { ucdExperimental 990 }

    surv-enter TRAP-TYPE
        STATUS current
        ENTERPRISE survtraps
        VARIABLES { sysLocation }
        DESCRIPTION "Enter Survivability Mode"
        ::= 17

    surv-exit TRAP-TYPE
        STATUS current
        ENTERPRISE survtraps
        VARIABLES { sysLocation }
        DESCRIPTION "Exit Survivability Mode"
        ::= 18

END
```

6.2. Creating a Sample Trap

The traps will be initiated by two hook scripts that have to be added to the directory. The first would be the script. The `HOSTNOTIFY` method has to be set to the destination for your trap.

As always, the \ means that the content should be a single line that has been formatted here for better readability.

```
#!/bin/bash

HOSTTONOTIFY=10.0.0.197

snmptrap -v 1 -c public $HOSTTONOTIFY \
  TRAP-SURV-MIB::survtraps \
  ' 6 17 ' SNMPv2-MIB::sysLocation.0 s \
  "Survivability Mode Enter"
```

You have to make sure that this script has execute permissions. If this is not the case, use chmod to set them. The second script would be identical to the one above, but the variable should be set to the destination for your traps.

```
#!/bin/bash
/etc/customscripts survivability-enter +x survivability-
enter/etc/customscripts/survivability-leave. HOSTTONOTIFY
HOSTTONOTIFY=10.0.0.197

snmptrap -v 1 -c public $HOSTTONOTIFY \
  TRAP-SURV-MIB::survtraps \
  ' 6 18 ' SNMPv2-MIB::sysLocation.0 s \
  "Survivability Mode Exited"
```