# Nimsoft/UIM/DX IM Alarm Reporter

Table of Contents

[Nimsoft/UIM/DX IM Alarm Reporter 1](#_Toc106181177)

[1.1 Alarm Reporting 3](#_Toc106181178)

[1.1.1 Example 1: run a detail report of the last 24 hours. 6](#_Toc106181179)

[1.1.2 Example 2: report on all net\_connect messages 8](#_Toc106181180)

[1.1.3 Example 3: report on all assigned alarms from last 4 hours 9](#_Toc106181181)

[1.1.4 Example 4: report on all level 4 and 5 alarms from last hour 9](#_Toc106181182)

[1.1.5 Example 5: report on first occurrences of an alarm from last hour 9](#_Toc106181183)

[11.1.6 Example 6: report to find detail what happened with an alarm 9](#_Toc106181184)

[11.1.7 Example 7: report on all assigned alarms from last 12 hours 10](#_Toc106181185)

[1.2 Open Alarm Report 11](#_Toc106181186)

[1.2.1 Example: open alarms for printer probe 12](#_Toc106181187)

[2 - nimsoft\_generic.dat 13](#_Toc106181188)

[3 - Create Windows Perl environment 15](#_Toc106181189)

[4 - Create Linux Perl environment 18](#_Toc106181190)

[4.1 install Perl base 18](#_Toc106181191)

[4.2 Install C compiler 19](#_Toc106181192)

[4.3 Install additional Perl Packages 19](#_Toc106181193)

[4.3.1 Perl modules MySQL 20](#_Toc106181194)

[4.3.2 Compiling Perl 21](#_Toc106181195)

Initially this tool was written to obtain an exception report of all not known alarms for the last 12 hours. This was done by adding all “normal” alarms in the message exclude filter –me.

Afterwards more and more options were added to give more flexible reporting possibilities.

This reporting tool will give you the possibility to report in a very flexible way (via SQL and regex filters) on the Nimsoft/UIM/DX IM Alarms stored in your MSSQL/MySQL/Mariadb database.

As result you can have:

* HTML report
* csv file
* return code equal to the selected # of exception messages

This tool can be used on:

* Windows to MSSQL and MySQL (Mariadb)
* The compiled version can be used on a standalone machine with my Perl or database client installed (Windows and Linux)
* Linux to MSSQL and MySQL (Mariadb) (older version that is compiled)

**Note1**: Linux version was tested on Centos7, Windows version was tested on W2016 with MSSQL 2016 and UIM 20.4.

**Note2:** the possibility to check the return code of the tool, that will contain the # of selected messages/alarms, was used to send an email in the morning that indicated in the subject line how many exceptions occurred the last 12 hours.

## 1.1 Alarm Reporting

Your console only contains the open alarms. The SQL tables:

* **NAS\_TRANSACTION\_LOG**: the detail messages
* **NAS\_TRANSACTION\_SUMMARY**: similar messages are listed once with the suppcount field representing the number of occurrences
* NAS\_ALARMS: the open alarms (see next section)
* **ALARMTRANSACTIONLOG**: this table is only created/used when you activate the "network transaction logging". This makes it possible that remote NAS alarms are also logged in a central SQL table.

The tool: **nimsoft\_alarm\_reporter.exe/pl/bin** will create reports based on the NAS\_TRANSACTION\_LOG or NAS\_TRANSACTION\_SUMMARY table.

The goal of this reporting tool is to create exception reports. Try to filter all unneeded messages and report only on those messages that are important.

The only required parameter is: –bh”nn” (or another parameter to indicate the reporting period).

nimsoft\_alarm\_reporter -bh"number of hours"

**Input parameters:**

-rs: (optional) number of seconds between report + html updates

-wr: (optional) create overvieW report (Y, N) Default Y

-al: (optional) t: nas\_transaction\_log, s: nas\_transaction\_summary, y: query alarmtransactionlog table. Default: t (nas\_transaction\_log)

**Date Selection:** (one date option must be selected)

-bh: (optional) number of hours to go back to start report

-bm: (optional) number of minutes to go back to start report

-mm:(optional) 0: report current month, 1: report previous month

-1: (optional) start date-time (format: "yyyy-mm-dd hh:mm")

-2: (optional) start hour filter (format hh:mm)

-3: (optional) end date-time (format: "yyyy-mm-dd hh:mm")

-4: (optional) end hour filter (format hh:mm)

**Report Columns:**

-co: give the columns you want in your report: (duration,robot,source,hostname,subsys,probe,severity,level,visible,suppcount,message,suppkey,sid,acknowledged,user\_tag1,user\_tag2,hub,origin,

level,assigned\_by,assigned\_to)

**note**: when you use the nas\_transaction\_summary table as input (-al"s") you can use the columns: custom1, custom2, custom3, custom4 and custom5

**SQL where like filters** (to limit the amount of selected records)

-lm: like Message (as where like clause in sql)

-lh: like Hostname (as where like clause in sql)

-lg: like oriGin (as where like clause in sql)

-lu: like hUb (as where like clause in sql)

-lt: like roboT (as where like clause in sql)

-lo: like prObe (as where like clause in sql)

**Search Criteria:** (regex)

-mi/me: message text include/exclude

-hi/he: hostname include/exclude

-gi/ge: oriGin include/exclude

-ui/ue: hUb include/exclude

-ti/te: roboT include/exclude

-oi/oe: prObe include/exclude

-ai/ae: assigned\_by include/exclude

-si/se: assigned\_to include/exclude

-ki/ke: acknowledged\_by include/exclude

-li/le: level include/exclude (0-5)

-cv: (suppress)count value regex filter

**note**: cv"1" will select alarms with count blank, 0 and 1

-vi: Visible y: show visible (default) n: no visible reported o: only visible

**Output:**

-di: output directory (default: c:\temp or /opt/temp)

-fi: output file (default: report\_nimsoft\_alarm\_reporter)

-9: View report directly (y,n) Default: n

-x: create csv file (y) or html report (n) Default: n

-nb: maximum reported lines. Default: 5000

-fq: strip fqdn from hostname column (y,n) default: y

Goal of this tool is to:

* create reports by exception, so that you can define a nexec profile that executes the report at 08:00 to produce all abnormal (= not known) messages that happened during the night.
* create overview reports that can show different counters of the most active probe, robot or message.

The program will end with a return code that is equal to the selected messages.

**Note1**: see next section that describes the nimsoft\_generic.dat file. This file contains all user, server variables used by these custom reporting tools.

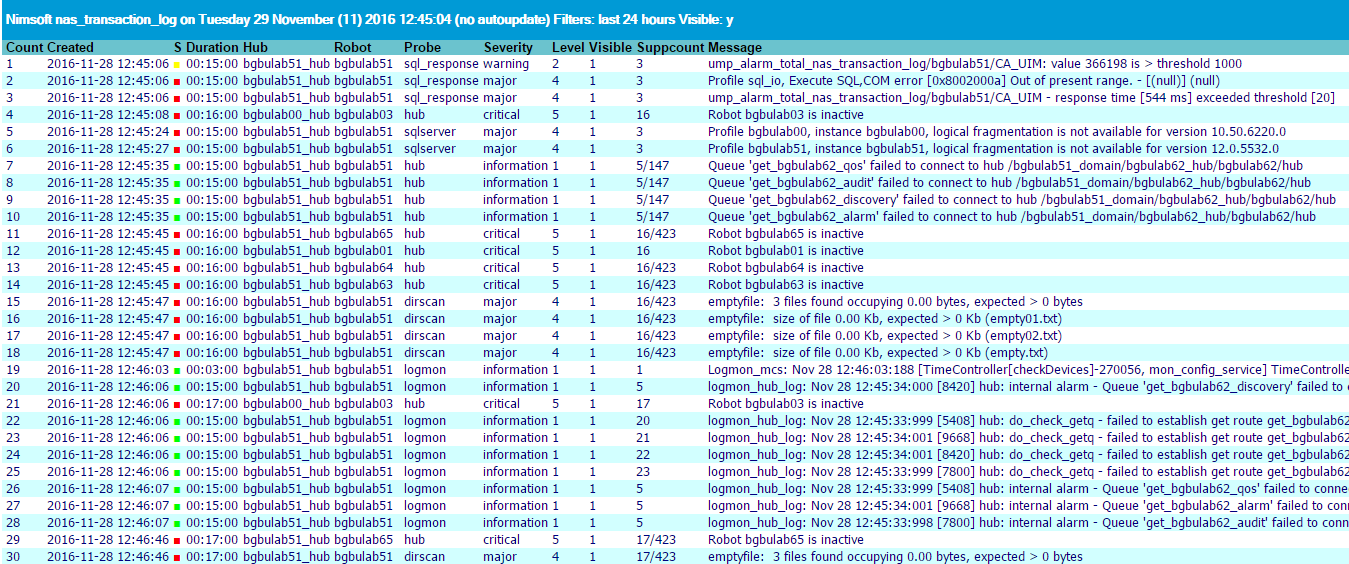
**Note2:** if you use only -2 (start hour) or -4 (end hour) as date selection parameter(s) we will use the todays date in the SQL query. In case you use other date selection parameters, the -2 and/or -4 parameters will be used in the regex filtering

**Note3**: where possible use the SQL filters, this will improve performance because only a limited set of records are selected.

**Note4:** if you combine SQL and Regex filters, it’s important to note that the SQL filters are determining the selected records and that the regex filters are applied only to this resulting record set.

### 1.1.1 Example 1: run a detail report of the last 24 hours.

nimsoft\_alarm\_reporter.exe -bh"24"



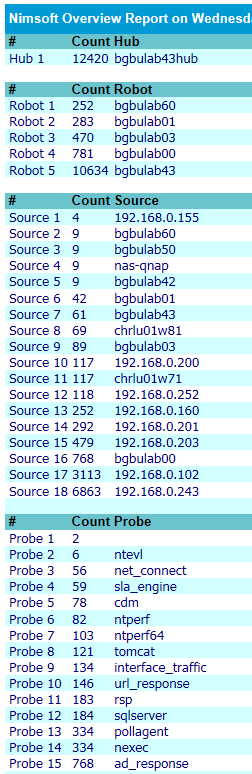
File: c:\temp\report\_nimsoft\_alarm\_reporter.html

Once the 'alarm logging' into the db tables is activated, this tool can see all messages, also those messages flagged with a count or invisible.

**Note1:** a first decision you will have to take is if you want to see all detail messages (-al"t") or only 1 line for all occurrences (-al"s"). Default is: -al"t".

**Note2**: a second decision is how you want your output. Do you want a HTML report (default) or do you want a csv file (-x"y").

**Note3**: a possible 3th decision is if you want the overview reports, by default these are generated (–wr”y”)Sample overview report(s): (file: c:\temp\report\_nimsoft\_alarm\_reporter\_overview.html)



These overview views can help you in finding abnormal alarm generators.

### 1.1.2 Example 2: report on all net\_connect messages

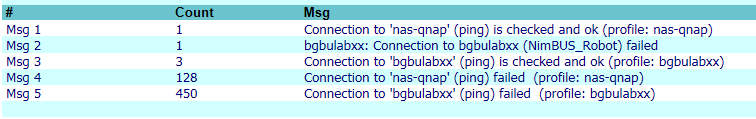
**Note**: By default there is a pre-processing rule that makes all net\_connect messages invisible.

If you are testing or expecting the ping alarms from net\_connect it can be confusing that you don't see all messages. (because some messages are "cleared")

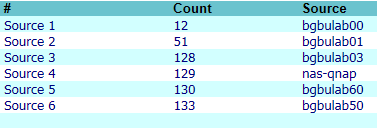
nimsoft\_alarm\_reporter.exe -lp"net\_connect" -bh"2"

The SQL parameter –lp is more performing than the regex filter –oi.

The different overview reports give a good view of what type of messages you selected:

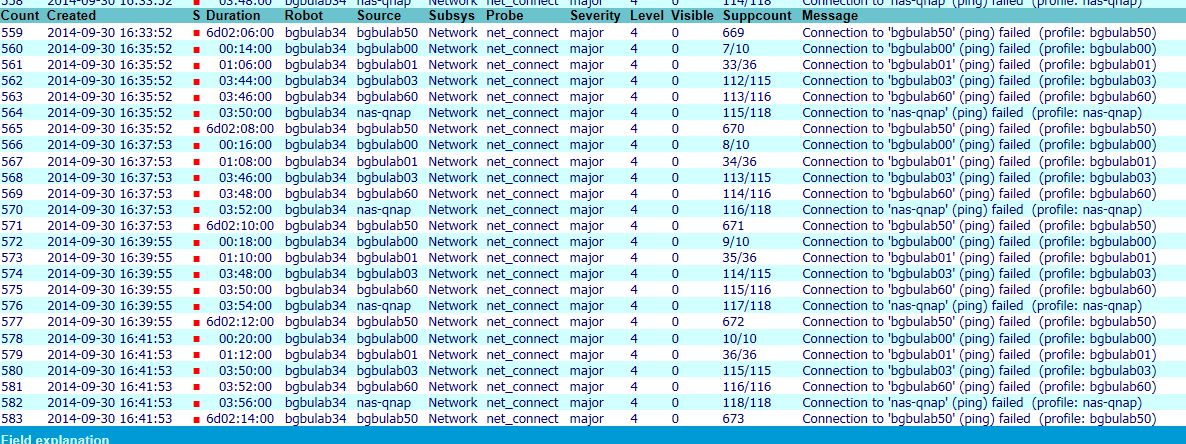


You can see that almost all messages are ping failed messages.



Here we see the source servers for all messages.

In the detail report we see all ping failed that are not appearing in the console:



### 1.1.3 Example 3: report on all assigned alarms from last 4 hours

perl nimsoft\_alarm\_reporter.pl -si"\w" -bh"4"

With **-si** we create an include filter on assigned\_to. The **\w** regex filter means "any word".

### 1.1.4 Example 4: report on all level 4 and 5 alarms from last hour

perl nimsoft\_alarm\_reporter.pl -li"4|5" -bh"1"

With **-li** we create an include filter on level. All include and exclude filters are in regex format.

### 1.1.5 Example 5: report on first occurrences of an alarm from last hour

perl nimsoft\_alarm\_reporter.pl -bh"1" -cv"1"

The keyword **-cv**"1" will test on occurrence blank, 0 and 1.

### 11.1.6 Example 6: report to find detail what happened with an alarm

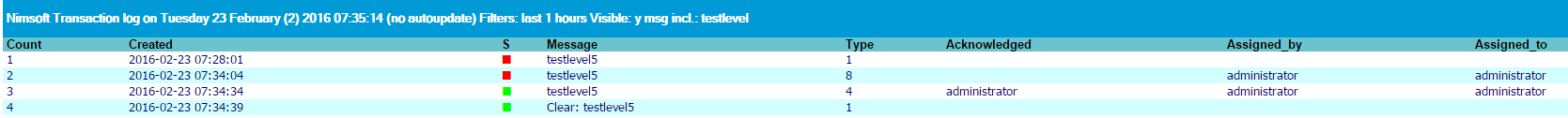
Sometimes you need to know why an alarm was not handled correctly by operations.

* did the message arrive?
* was it visible?
* was is acknowledged?

We will change the default columns from the report to add assigned\_by, assigned\_to and type:

perl nimsoft\_alarm\_reporter.pl -mi"testlevel" -bh"1" -co"message,type,a

cknowledged,assigned\_by,assigned\_to"



By adding the type column you can follow the live steps for an alarm:

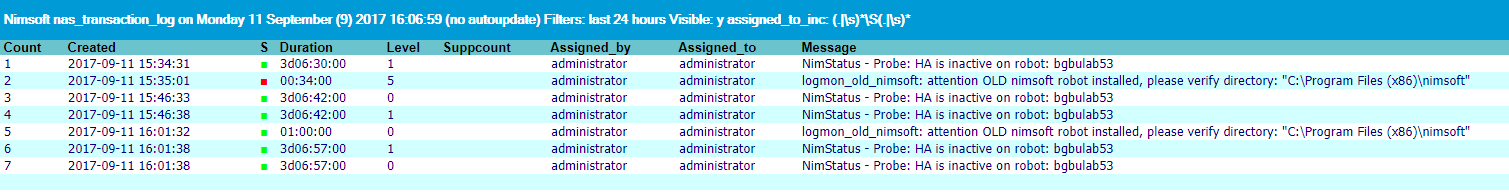
* 1: alarm created
* 2: alarm updated
* 4: acknowledged
* 8: assigned/unassigned

### 11.1.7 Example 7: report on all assigned alarms from last 12 hours

With the -ai switch (“assigned by” include) and -si (“assigned to” include) you can select the messages that are assigned to somebody specific, or to any value:

perl nimsoft\_alarm\_reporter.pl -bh"2" -si"(.|\s)\*\S(.|\s)\*" -co"duration,level,suppcount,assigned\_by,assigned\_to,message"

The "-si" regex filter will select the “assigned to” field for any non-space character. The -co parameter will customize the columns that you will receive in the report.



## 1.2 Open Alarm Report

The tool: **nimsoft\_open\_alarms.exe** will read the nas\_alarms table to get all open alarms.

This tool uses the file: nimsoft\_generic.dat to store all user variables. This file is common between several utilities. Please edit, once, this file and adapt all parameters.

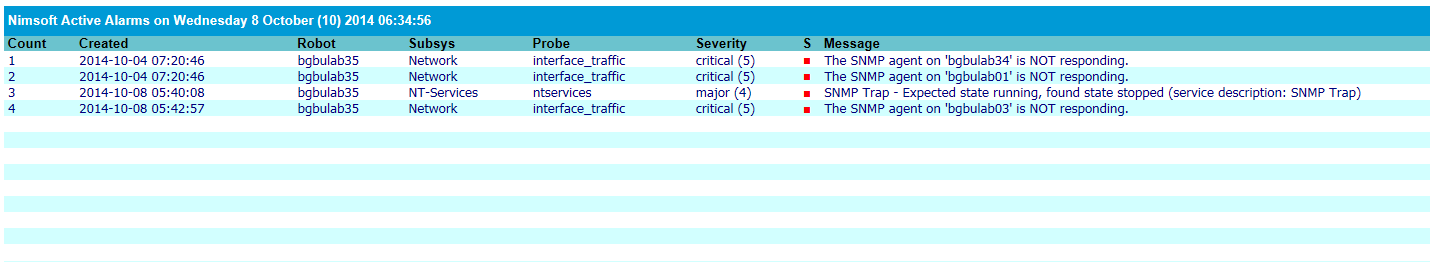
Execution:

* **-d**:(optional) output directory (default: c:\temp)
* **-f**: (optional) output file (default: report\_nimsoft\_open\_alarms.html)

Search Criteria/Filters:

* **-mi**: (opt) regex message include filter (use -m"all" to see all open alarms)
* **-me**: (opt) regex message exclude filter
* **-hi**: (opt) regex hostname include filter
* **-he**: (opt) regex hostname exclude filter
* **-ti**: (opt) regex roboT include filter
* **-te**: (opt) regex roboT exclude filter
* **-oi**: (opt) regex probe include filter
* **-oe**: (opt) regex probe exclude filter

nimsoft\_open\_alarms.exe -mi"all"



The tool will end with a return code equal to the # of selected messages. This gives you the possibility to email the report and use the %errorlevel% variable in the subject line of your email.

### 1.2.1 Example: open alarms for printer probe

You want to mail to your helpdesk an hourly report with all open alarms from the printer probe.

1. First we run the Perl: nimsoft\_open\_alarms

nimsoft\_open\_alarms.exe -oi"printer"

1. Save the return code

set rc=%errorlevel%

1. Send customized mail with report embedded in mail (so they don't need to open an attachment)

blat - -to james.bond@ca.com -body "In attachment the %rc% open printer Alarms" -subject "We found: %rc% open printer alarms" -embed c:\temp\report\_nimsoft\_open\_alarms.html

**Note**: Blat is an open source mailing tool with all needed switches and parameters.

# 2 - nimsoft\_generic.dat

All custom reporting tools use a common parameter file: nimsoft\_generic.dat

# -- UMP server & port

uim\_server=ump\_server\_name

uim\_port=80

uim\_https=http

# --- Nimsoft userid and crypted password (via nimsoft\_crypt.exe)

uim\_user=administrator

**uim\_password**=gWL/M/ij/Dvonp0=

# --- Nimsoft domain, hub and robot to create address to the main hub

uim\_domain=xxx\_domain

uim\_hub=xxx\_hub

uim\_robot=xxx

# --- SQL server, userid, crypted password (via nimsoft\_crypt.exe) and database name

sql\_server=sql\_server\_name

sql\_user=sa

**sql\_password**=gWL/M/ij/Dvonp0=

sql\_db=CA\_UIM

# - sql\_type: mssql or mysql

sql\_type=mssql

# - sql\_driver: "SQL Server" (=default) or a manual installed newer driver, example: "ODBC Driver 17 for SQL Server" (\* no quotes around driver name \*)

sql\_driver=SQL Server

# --- end of parameters ---

The 2 passwords are stored in an encrypted form. To generate this encrypted password, use:

nimsoft\_crypt your\_password

As output you will receive the string that you can enter as password in the above file.

**Note1**: some of these variables are perhaps not needed for this tool and are used by other custom utilities.

**Note2:** if you use sql\_user=trusted and use also sql\_password= we will connect to MSSQL via a trusted connection. (=your logged on userid)

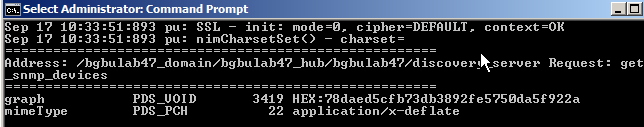
**Note3**: the uim\_robot must be defined in the format/case that UIM recognize them. (like it's displayed in IM)

**Note4:** sql\_driver: previously we used the hardcoded sql driver “SQL Server” that is by default installed in Windows systems. But when you want to connect to TLS 1.2 MSSQL you need to install (manually) a newer MSSQL driver (example: ODBC Driver 17 for SQL Server)

**Note5**: In case you receive a "communication error" while using a non-simulation execution of the tool, try to use the: /uim\_domain/uim\_hub/uim\_robot values like you use it in nimsoft\_generic.dat in the command:

pu -u administrator -p ??? /bgbulab47\_domain/bgbulab47\_hub/bgbulab47/discovery\_server get\_snmp\_devices

This commands must give a normal output like:



These names are case sensitive.

# 3 - Create Windows Perl environment

This tool contains the Perl source and compiled Perl.

If you want to create a Perl environment that can run this Perl source & optionally compile the source yourself you can follow the documented steps.

Once the Perl environment is created you can compile the Perl source:

pp -C -o c:\unibat\nimsoft\_alarm\_reporter.exe -l="C:\strawberry\c\bin\libmysql\_\_.dll" c:\unibat\nimsoft\_alarm\_reporter.pl

**Note1**: you must execute this PP command from a command prompt with as directory where you placed:

* nimsoft\_alarm\_reporter.pl
* nimsoft\_generic.pm

**Note2:** the -l option is needed to include all dll modules to be able to execute the compiled module on an external servers without Perl installed.

* + download from: http://strawberryperl.com/releases.html the file:
    - strawberry-perl-5.14.2.1-64bit.msi (it's a **must** that you download version 5.14.2)
  + install the msi in: c:\Strawberry64 (as an example)
  + verify that the following directories are in the system path:
    - C:\strawberry64\perl\bin
    - C:\strawberry64\perl\site\bin
    - C:\strawberry64\c\bin
  + refresh the command prompt or reboot the activate the path
  + deploy the probe: SDK\_Perl to the main UIM server (this creates: C:\Program Files (x86)\Nimsoft\perllib)
  + copy the directories under: C:\Program Files (x86)\Nimsoft\perllib to C:\strawberry64\perl\lib
    - now you are ready to install additional packages via cpan: (you need internet access because these modules are get directly from cpan):
      * cpan install Crypt::RC4
      * cpan install XML::Simple
      * cpan install HTTP::Request
      * cpan install DBD::ODBC (here you will receive some messages, but it's normal)
      * cpan install MIME::Base64
      * cpan install Time::Piece
      * cpan install Time::Seconds
      * cpan install LWP::UserAgent
      * (cpan install -f WWW::Mechanize)
      * (cpan install PAR::Packer)
      * (cpan install MIME::Lite)
      * (cpan install Data::GUID)
      * (cpan install Sys::HostAddr)
      * cpan install XML::LibXML
      * (cpan force install Statistics::LineFit)
      * (cpan install Math::Spline)
      * cpan install DBD::CSV
      * (cpan install DBD::Chart)
      * cpan install Net::SNMP
      * pip http://strawberryperl.com/package/kmx/perl-modules-patched/Crypt-OpenSSL-Random-0.04\_patched.tar.gz
      * (cpan -fi Net::SSL::ExpireDate) (generates a lot of messages and can take some time)
      * cpan install Win32::Console
      * cpan install Term::ANSIColor
      * cpan install Win32::Console::ANSI

Note: the lines between () are not needed to run the probe or utility but are packages that are also installed on the original Perl directory received from CA services. (and can be needed if you run other customized reports/tools)

# 4 - Create Linux Perl environment

## 4.1 install Perl base

- deploy perl probe package (perl\_linux\_23\_64)

- deploy Perl UIM SDK probe (SDK\_Perl)

- add in /etc/bashrc:

PATH=/opt/nimsoft/perl/bin:$PATH

- add perl5lib

\* Add a file in directory: /etc/profile.d example: setperl.sh

\* add in this file:

export PERL5LIB=/opt/nimsoft/perllib:/root/perl5/lib/perl5:/root/perl5/lib/perl5:/root/perl5/lib/perl5

- login again or reboot to activate settings

## 4.2 Install C compiler

- to install the c compiler (to install other packages):

Yum group install “Development Tools”

Yum install expat-devel

Probably the "Development Tools" will have added also version of Perl installed, but uim perl is first in definitions.

## 4.3 Install additional Perl Packages

- Note: in this order this will install perl in /opt/nimsoft/perl

- Install now the additional Perl packages (needed by several chrlu01 utilities), all done by:

**cpan install package (the first cpan install will ask to configure cpan)**

- cpan install Encode::Locale

- cpan install File::Listing

- cpan install HTML::Entities

- cpan install HTML::HeadParser

- cpan install IO::HTML

- cpan install LWP::MediaTypes

- cpan install URI

- cpan install HTTP::Cookies

- cpan install HTTP::Daemon

- cpan install HTTP::Headers::Util

- cpan install HTTP::Negotiate

- cpan install YAML

- cpan install Net::HTTP

- cpan install WWW::RobotRules

- cpan install LWP::UserAgent

- cpan install XML::Simple

- cpan install Crypt::RC4

- cpan install File::Which

- ( check yum install [perl-XML-LibXML-2.0018-5.el7.x86\_64](http://linuxsoft.cern.ch/cern/centos/7/updates/x86_64/Packages/perl-XML-LibXML-2.0018-5.el7.x86_64.rpm)

- yum install libxml2

- yum install libxml2-devel

- cpan install XML::LibXML

- cpan install Win32::Console

- cpan install Term::ANSIColor

- cpan install Win32::Console::ANSI

### 4.3.1 Perl modules MySQL

If you need the MySQL/MariaDB interface on a server without the DB itself:

* goto directory:  /etc/yum.repos.d/ and create a file (example) mariadb.repo
* copy the following in the file (or generate the content via: https://downloads.mariadb.org/mariadb/repositories/#mirror=nucleus

# MariaDB 10.1 CentOS repository list - created 2016-11-18 06:47 UTC #<http://downloads.mariadb.org/mariadb/repositories/> [mariadb] name = MariaDB baseurl = <http://yum.mariadb.org/10.1/centos7-amd64> gpgkey=<https://yum.mariadb.org/RPM-GPG-KEY-MariaDB> gpgcheck=1

* add in bashrc:

export DBD\_MYSQL\_CFLAGS=-I/usr/local/mysql/include/mysql

export DBD\_MYSQL\_LIBS="-L/usr/local/mysql/lib/mysql -lmysqlclient"

export DBD\_MYSQL\_EMBEDDED=

export DBD\_MYSQL\_CONFIG=mysql\_config

export DBD\_MYSQL\_NOCATCHSTDERR=0

export DBD\_MYSQL\_NOFOUNDROWS=0

export DBD\_MYSQL\_NOSSL=

export DBD\_MYSQL\_TESTDB=test

export DBD\_MYSQL\_TESTHOST=bgbulab67

export DBD\_MYSQL\_TESTPASSWORD=Unicenter10

export DBD\_MYSQL\_TESTPORT=3306

export DBD\_MYSQL\_TESTUSER=root

* run: yum install MariaDB-client
* yum install perl-DBD-mysql
* yum install mysql-devel
* yum install zlib-devel
* yum install openssl-devel
* cpan install DBD::mysql

**Note**: you will remark that sometimes a pre-requisite package can be missing, you will need to install the missing pre-requisite also. (here it's a big difference with Strawberry Perl on Windows that installs automatically all missing pre-requisites)

**Note:**If cpan is not initialized correctly, remove /root/.cpan (rm –rf /root/.cpan) and excute cpan again to reinit

### 4.3.2 Compiling Perl

**Note**: if you want to compile Perl sources and you don't have the PP command you can install:

- Cpan install PAR

- cpan install PAR::Packer

To compile a module:

pp -C -o test.bin test.pl

this to compile a very simple Perl

pp -C -o nimsoft\_alarm\_reporter.bin -l /usr/lib64/libodbc.so.2 -l /usr/lib64/libmysqlclient.so nimsoft\_alarm\_reporter.pl

this to compile a Perl that will access MySQL and you want to be able to execute this .bin from a standalone machine with no Perl and no MySQL client.