How-To Create parser files and compliance policy files

In this section we are assuming that you already create the command set to extract the configuration that you need. Please check this article How-To Create a Compliance Policy in opConfig Step 1-b

Parser files.

Non-structured command outputs need to be condensed and transformed before opConfig can make compliance assessments in an efficient manner. This operation is performed by any number of 'configuration parsers', small components (written in Perl) that act as Subject-matter Experts and digest the textual input into one precise, unambiguous and structured document that's minimal in the sense of only containing relevant facts and properties.

For example, only small parts of the output of a Cisco's "`show interfaces`" command would be relevant for detecting configuration mistakes like directed broadcasts being allowed. opConfig itself cannot contain the domain-specific expert knowledge for all kinds of devices, therefore we decided to make these config parsers extensible components: anybody can provide their own parsers for a particular type of command output.

opConfig's configuration parsers are more than just parsers and we might have called them ***"knowledge extractors"*** instead. The purpose of such a config parser is to consume a particular type of command output and transform (relevant parts of) it into a tree structure (json output).

Parsers are installed in the directory /usr/local/omk/conf/config_parsers and they must have the .pm extension. All config parsers must be valid Perl scripts.

Step 1. Identifying what information do you want to extract from a command output.

In this example we want to extract the Netflow cofiguration on a Fortigate device. Fortigate command that have the information we need "**show full-configuration system netflow**"

```
FGVM04TM22002236 # show full-configuration system netflow
config system netflow
    set collector-ip 192.168.0.104
    set collector-port 2055
    set source-ip 192.168.0.105
    set active-flow-timeout 1800
    set inactive-flow-timeout 15
    set template-tx-timeout 1800
    set template-tx-counter 20
    set interface-select-method auto
end
```

Step 2. Create the parser file.

Now we have to create the parser for the "show full-configuration system netflow" command. We can copy the cisco-config.pm parser to create fortigate-netflow.pm parser.

```
# cp cisco-interface.pm fortigate-netflow.pm
```

Edit the fortigate-netflow.pm parser file in order to find the following values.

- collector-ip
- collector-port
- source-ip
- active-flow-timeout
- inactive-flow-timeout
- template-tx-timeout
- template-tx-counter
- interface-select-method

```
# vi fortigate-netflow.pm
```

```
# this is a Fortigate "show full-configuration system netflow" parser for opconfig, which amends the
config_features->config section
our $VERSION = "1.0.0";

my %config;
for my $line (split(/\r?\n/,$input)) # the $input vriable contains all the command output from show full-
configuration system netflow
{
        if ($line =~ /.*set collector-ip (.+)/)  # the (.+) is the $1 variable
        {
                $config{collectorip} = $1;    # collectorip variable store the value 192.168.0.104
        }
        if ($line =~ /.*set collector-port (.+)/)
        {
                $config{collectorport} = $1;
        }
        if ($line =~ /.*set source-ip (.+)/)
        {
                $config{sourceip} = $1;
        }
        if ($line =~ /.*set active-flow-timeout (.+)/)
        {
                $config{activeFtimeout} = $1;
        }
        if ($line =~ /.*set inactive-flow-timeout (.+)/)
        {
                $config{inactFtimeout} = $1;
        }
        if ($line =~ /.*set template-tx-timeout (.+)/)
        {
                $config{txtimeout} = $1;
        }
        if ($line =~ /.*set template-tx-counter (.+)/)
        {
                $config{txcounter} = $1;
        }
        if ($line =~ /.*set interface-select-method (.+)/)
        {
                $config{intmethod} = $1;
        }
}

return { config_features => { netflow => \%config } };
```

We can check if the syntaxis is ok using this:

```
# perl -c fortigate-netflow.pm
fortigate-netflow.pm syntax OK
```

To create parsers files you need to know the basic concepts of perl programin and regular expresions

Step 3. Add parser file to opCommon file.

Now we need to add the fortigate-netflow.pm file to the opCommon file.

```
# cd /usr/local/omk/conf/
# vi opCommon.json
```

```
.
.
.
      "opconfig_url_base" : "",
      "opconfig_disable_ios_ssh_connection_discovery" : "false",
      "opconfig_parsers" : [
         [
            "^show full-configuration system netflow$",          # Command that we are using to extract the
information
            "config_parsers/fortigate-netflow.pm"                # File to parse the command output
         ]
      ],
      "opconfig_queue_expire_after_seconds" : 691200,
      "opconfig_audit_import" : 1,
.
.
.
```

We can check if the syntax is is ok using this:

```
# json_xs < opCommon.json
```

Note that *all* matching parsers will be applied for a particular command, in the order they are given in the configuration.

Step 4. Execute the update_config_status and export_config_status commands.

To verify that your parsers have correctly extracted the expected properties, you can update and export the newest version of the config status document.

```
# /usr/local/omk/bin/opconfig-cli.pl act=update_config_status node=FortinetTest force=1 debug=9
```

```
[root@localhost conf]# /usr/local/omk/bin/opconfig-cli.pl act=update_config_status node=FortinetTest force=1
debug=9
opconfig-cli.pl Version 3.420.0

Copyright (C) 2015 Opmantek Limited (www.opmantek.com)
This program comes with ABSOLUTELY NO WARRANTY;
See www.opmantek.com or email contact@opmantek.com

opConfig is licensed to Opmantek Internal for 50 Nodes - Expires 15-Aug-2023

[2023-01-20 17:26:48.57293] [7905] [debug] new opConfig: require_db
[2023-01-20 17:26:48.71776] [7905] [debug] Creating NMISx
[2023-01-20 17:26:48.95578] [7905] [debug] getting newest config status for node FortinetTest
[2023-01-20 17:26:48.96042] [7905] [debug] Node status for FortinetTest needs updating
[2023-01-20 17:26:48.96154] [7905] [debug] found no config parser for command "diagnose ip address list" on
node FortinetTest
[2023-01-20 17:26:48.96165] [7905] [debug] found no config parser for command "get system status" on node
FortinetTest
[2023-01-20 17:26:48.96168] [7905] [debug] found no config parser for command "nmap -T4 -F" on node FortinetTest
[2023-01-20 17:26:48.96170] [7905] [debug] found no config parser for command "nmap -T4 -O -F --version-light"
on node FortinetTest
[2023-01-20 17:26:48.96173] [7905] [debug] found no config parser for command "ping" on node FortinetTest
[2023-01-20 17:26:48.96176] [7905] [debug] found no config parser for command "show full-configuration" on node
FortinetTest
[2023-01-20 17:26:48.96178] [7905] [debug] found no config parser for command "show full-configuration system
interface" on node FortinetTest
[2023-01-20 17:26:48.96181] [7905] [debug] getting newest command output for show full-configuration system
netflow
[2023-01-20 17:26:48.96393] [7905] [debug] newest command output for show full-configuration system netflow is
in
[2023-01-20 17:26:48.96404] [7905] [debug] processing command show full-configuration system netflow, node
FortinetTest, revision 1, input length 289, structured no
[2023-01-20 17:26:48.96451] [7905] [debug] running parser /usr/local/omk/conf/config_parsers/fortigate-netflow.
pm for node FortinetTest and command "show full-configuration system netflow"
[2023-01-20 17:26:48.96492] [7905] [debug] parser finished, merging results
[2023-01-20 17:26:48.96511] [7905] [debug] found no config parser for command "traceroute" on node FortinetTest
```

Please avoid to use "-" or "_" in the parser files variable. Example:

```
my %config;
for my $line (split(/\r?\n/,$input)) # the $input vriable contains all the command output from show full-configuration system netflow
{
    if ($line =~ /.*set collector-ip (.+)/)  # the (.+) is the $1 variable
    {

        $config{collector-ip} = $1;    # collector-ip variable store the value 192.168.0.104
    }
```

Error that you could get when execute the update_config_status command:

```
[2023-01-20 17:41:31.89140] [9699] [debug] running parser /usr/local/omk/conf/config_parsers/fortigate-netflow.pm for node FortinetTest and
command "show full-configuration system netflow"
[2023-01-20 17:41:31.89179] [9699] [warn] Config parser /usr/local/omk/conf/config_parsers/fortigate-netflow.pm failed to execute: Bareword
"collector" not allowed while "strict subs" in use at (eval 1231) line 7.
Bareword "ip" not allowed while "strict subs" in use at (eval 1231) line 7.
Bareword "collector" not allowed while "strict subs" in use at (eval 1231) line 12.
```

```
# /usr/local/omk/bin/opconfig-cli.pl act=export_config_status node=FortinetTest debug=true
```

```
[root@localhost conf]# /usr/local/omk/bin/opconfig-cli.pl act=export_config_status node=FortinetTest debug=true
opconfig-cli.pl Version 3.420.0

Copyright (C) 2015 Opmantek Limited (www.opmantek.com)
This program comes with ABSOLUTELY NO WARRANTY;
See www.opmantek.com or email contact@opmantek.com

opConfig is licensed to Opmantek Internal for 50 Nodes - Expires 15-Aug-2023

[2023-01-20 17:32:28.57037] [8596] [debug] new opConfig: require_db
[2023-01-20 17:32:28.71590] [8596] [debug] Creating NMISx
{
   "config_features" : {
      "netflow" : {
         "activeFtimeout" : "1800",
         "collectorip" : "192.168.0.104",
         "collectorport" : "2055",
         "inactFtimeout" : "15",
         "intmethod" : "auto",
         "sourceip" : "192.168.0.105",
         "txcounter" : "20",
         "txtimeout" : "1800"
      }
   }
}
```

Compliance Policy files.

Compliance policy language is very similar to opEvents language.

Here is a quick overview of the structural rules:

- A policy consists of one hash (or "associative array"). All hash keys (=rule numbers) must be numeric, and the keys control the order of rule evaluation.
  Rule numbers do not have to be globally unique, just within the enclosing subpolicy.
- Each hash element must describe either one IF/THEN clause or one EACH/BLOCK iteration.
- THEN statements can be either a single string (describing the actions to take) or a nested sub-policy (in the form of a nested hash).
- EACH/BLOCK iterations always require a nested sub-policy.
- IF statements are single strings, made up from structure or variable selector expressions and Perl operators and expressions.
- The available actions for THEN statements are `ok()`, `exception()`, `CONTINUE()` and `LAST()`.
- EACH statements consist of a variable name (for the iterator variable to be) and a structure selector expression (for the objects to iterate over).
- The policy engine invokes policy rules with a number of pre-defined structure variables, to provide access to the configuration status document, the current node name and a few others

Compliance policy files are installed in the directory /usr/local/omk/conf/compliance_policies and they must have the .json extension.

Step 1. Create compliance policy file.

With the Compliance policy file we are going to evaluate the information that we got on step 4 - Execute the update_config_status and export_config_status commands. We have to create a compliance policy file, navigate to /usr/local/omk/conf/compliance_policies.

```
# cd /usr/local/omk/conf/compliance_policies
```

We can copy the cisco_nsa.json policy to create fortigate-netflow.json policy.

```
# cp cisco_nsa.json fortigate-netflow.json
```

In our case we want to valided if the Fortigate has the collector IP and collector port configured.

let's edit the fortigate-netflow.json file

```
# vi fortigate-netflow.json
```

```
{
   "10" : {
      "IF" : "not defined(${NODEINFO}.os_info)",
      "THEN" : "exception(\"Node has no os_info\",0,node=$NODENAME) AND LAST()",
      "Comment" : "If the device does not have a OS information you will get an exception "
   },
   "20" : {
      "IF" : "$NODEINFO.os_info.os eq \"Fortinet\"",
      "Comment" : "In our case OS info must macth with Fortinet in order to apply this policy, if you have
cisco device you should have IOS or IOS-EX etc",
      "Comment" : "This policy will apply only for Fortigate devices",
      "THEN" : {
         "201" : {
            "IF" : "$NODE.config_features.netflow.collectorip eq \"192.168.0.104\"",
            "Comment" : "If you want to check if the value on collectorip variable is equal to 192.168.0.104",
            "Comment" : "all the variables stored in the inventory are strings, you should evaluate with eq ne
etc",
            "THEN" : "ok(\"Collector IP for Netflow is OK\",4,node=$NODENAME,config=$NODE.config_features.
netflow.collectorip)",
            "Comment" : "Output that you will get in opConfig NODE.config_features.netflow.collectorip"
         },
         "202" : {
            "IF" : "$NODE.config_features.netflow.collectorip ne \"192.168.0.104\"",
            "THEN" : "exception(\"Collector IP for Netflow is not correct\",3,node=$NODENAME,config=$NODE.
config_features.netflow.collectorip)"
         },
         "203" : {
            "IF" : "not($NODE.config_features.netflow.collectorip)",
            "Comment" : "if the collectorip is not present on the inventory",
            "THEN" : "exception(\"Collector IP for Netflow is not configured\",3,node=$NODENAME,config=$NODE.
config_features.netflow.collectorip)"
         },
         "204" : {
            "IF" : "$NODE.config_features.netflow.collectorport eq \"2055\"",
            "THEN" : "ok(\"Collector port is OK\",4,node=$NODENAME,config=$NODE.config_features.netflow.
collectorport)"
         },
          "205" : {
            "IF" : "$NODE.config_features.netflow.collectorport ne \"2055\"",
            "THEN" : "exception(\"Collector port is not correct\",3,node=$NODENAME,config=$NODE.config_features.
netflow.collectorport)"
         },
         "206" : {
            "IF" : "not($NODE.config_features.netflow.collectorport)",
            "THEN" : "exception(\"Collector port is not configured\",3,node=$NODENAME,config=$NODE.
config_features.netflow.collectorport)"
         }
       }
    }
}
```

Compliance policy file does not allow comments. If you copy the code with comments you will get this error when you will try to use the check_compliance command.

```
[2023-01-23 12:24:01.24850] [68968] [debug] Operating on node FortinetTest
[2023-01-23 12:24:01.24909] [68968] [debug] iterating through policy rules, now at nr. 10, substvars
NODE, NODEINFO, NODENAME
[2023-01-23 12:24:01.24926] [68968] [debug] Operating on node FortinetTest
[2023-01-23 12:24:01.24939] [68968] [debug] Taking exception action exception, Rule Error
check_compliance failed for node FortinetTest: Policy unparseable: Unknown keywords present.
```

We can check if the syntaxis is ok using this:

```
# json_xs < fortigate-netflow.json
```

We get the variable $NODE.config_features.netflow.collectorip from the inventory output:

```
[root@localhost compliance_policies]# /usr/local/omk/bin/opconfig-cli.pl act=export_config_status node=FortinetTest debug=true
opconfig-cli.pl Version 3.420.0

Copyright (C) 2015 Opmantek Limited (www.opmantek.com)
This program comes with ABSOLUTELY NO WARRANTY;
See www.opmantek.com or email contact@opmantek.com

opConfig is licensed to Opmantek Internal for 50 Nodes - Expires 15-Aug-2023

[2023-01-23 11:53:09.48243] [64984] [debug] new opConfig: require_db
[2023-01-23 11:53:09.60927] [64984] [debug] Creating NMISx
{
  "config_features" : {
     "netflow" : {
        "activeFtimeout" : "1800",
        "collectorip" : "192.168.0.104",
        "collectorport" : "2055",
        "inactFtimeout" : "15",
        "intmethod" : "auto",
        "sourceip" : "192.168.0.105",
        "txcounter" : "20",
        "txtimeout" : "1800"
     }
  }
}
[root@localhost compliance_policies]#
```

Step 2. Execute the import_policy and check_compliance commands.

All compliance policies are named and versioned.

To make a policy available to opConfig, it must be imported, we are going to import the fortigate-netflow.json policy like this:

```
# /usr/local/omk/bin/opconfig-cli.pl act=import_policy name="netflow" file=/usr/local/omk/conf
/compliance_policies/fortigate-netflow.json
```

List the compliance policies

```
# /usr/local/omk/bin/opconfig-cli.pl act=list_policies
```

```
[root@localhost compliance_policies]# /usr/local/omk/bin/opconfig-cli.pl act=list_policies
opconfig-cli.pl Version 3.420.0

Copyright (C) 2015 Opmantek Limited (www.opmantek.com)
This program comes with ABSOLUTELY NO WARRANTY;
See www.opmantek.com or email contact@opmantek.com

opConfig is licensed to Opmantek Internal for 50 Nodes - Expires 15-Aug-2023

Policy     Version   Date
banner01   1         2022-11-17T09:37:12
conf01     1         2023-01-17T11:21:10
conf02     1         2023-01-17T11:38:25
conf03     1         2023-01-17T11:48:25
conf04     1         2023-01-17T12:02:11
conf05     1         2023-01-17T12:06:06
conf06     1         2023-01-17T12:11:22
conf06     2         2023-01-17T12:13:57
conf06     3         2023-01-17T12:18:36
conf06     4         2023-01-17T12:19:54
conf06     5         2023-01-17T13:01:36
conf06     6         2023-01-17T13:14:20
int01      1         2023-01-04T14:31:21
netflow    1         2023-01-23T12:05:49
```

Always that you change the compliance policy file you must import the policy in order to update it. You will have a new version.

```
[root@localhost compliance_policies]# /usr/local/omk/bin/opconfig-cli.pl act=list_policies
opconfig-cli.pl Version 3.420.0

Copyright (C) 2015 Opmantek Limited (www.opmantek.com)
This program comes with ABSOLUTELY NO WARRANTY;
See www.opmantek.com or email contact@opmantek.com

opConfig is licensed to Opmantek Internal for 50 Nodes - Expires 15-Aug-2023

Policy     Version   Date
banner01   1         2022-11-17T09:37:12
conf01     1         2023-01-17T11:21:10
conf02     1         2023-01-17T11:38:25
conf03     1         2023-01-17T11:48:25
conf04     1         2023-01-17T12:02:11
conf05     1         2023-01-17T12:06:06
conf06     1         2023-01-17T12:11:22
conf06     2         2023-01-17T12:13:57
conf06     3         2023-01-17T12:18:36
conf06     4         2023-01-17T12:19:54
conf06     5         2023-01-17T13:01:36
conf06     6         2023-01-17T13:14:20
int01      1         2023-01-04T14:31:21
netflow    1         2023-01-23T12:05:49
netflow    2         2023-01-23T12:26:53
```

At this time compliance policy assessments are not performed automatically but have to be triggered with opconfig-cli.pl:

We have to execute this command "/usr/local/omk/bin/opconfig-cli.pl act=check_compliance"

```
# /usr/local/omk/bin/opconfig-cli.pl act=check_compliance name='netflow' node=FortinetTest debug=9
```

Step 3. View Compliance Status

Now you can check the Complaince Status in the opConfig GUI. Access the opConfig GUI at http://YOUR_SERVERNAME/omk/opConfig, login and then from the Menu Bar "Views -> Compliance Status".