Table of Contents

Settings Tweaks

NetYCE supports some settings that can be modified by the customer to change specific behaviour. Usually these are procedure or policy related. They can be found in the Admin → Setup → Settings section of the main menu.

This article describes the Tweaks that are listed under the "Tweaks" class.

See the generic article on the General settings form on how to use it.

General Tweaks

AllowCustomReportsDecrypt

Custom reports will decrypt passwords only for those user-levels present in String (eg: '456'). Default is '456'. The Num value has no function and is ignored.

The NetYCE database has many encrypted columns that are decrypted for front-end and back-end purposes on the fly. This tweak controls the same dynamic decryption behaviour for user custom reports. Given the confidentiality of some of the information, these reports could easily circumvent the security policies of the customer. By specifying the user-levels permitted to get access to this dynamic encryption, the policies can be enforced.

These user-levels correspond to the Global Permission Level defined in the User Group the operator is a member of. The levels range from 0 to 6, mapping respectively to: 'disabled', 'browser', 'operator', 'engineer', 'modeller', 'manager', and 'system'.

Email_report_body_limit

Custom reports can be emailed after scheduled execution. The CSV output is included in the email body AND as an attachment. This tweak controls how many CSV lines may be included in the email body before truncating. The default is 500 lines.

Setting this value to "0" will result in an email with no CSV lines in the body and only an attachment. The attached CSV file is never truncated and always included.

AllowTemplateEdit

When number set <>0, editing of production template revision is allowed. Default is '0'. The String value has no function and is ignored.

The tweak removes the strict enforcing of the requirement that for any template change, a new revision must be created, providing a fully traceable change log of the templates. However for test- and development-environments this quickly results in overkill and a cumbersome process.

Allow_topo_multipoint

For Client_type in <String>, permit Service_type to add many topo links on a port when <Num> not 0.

The front-end and the Service-types do not allow a device port to have multiple topology connections since they are point-to-point by nature.

However, for migration purposes a feature to selectively allow multiple links per port is desirable. For this purpose the Lookup tweak 'Allow_topo_multipoint' is intended. It can allow or deny this behaviour per Client-type.

The default Tweak has no Client-type specified in the String value, indicating its setting applies to all Client-types. The Num value determines the setting itself: a zero for the normal not-permitted action (conform existing policies) and a non-zero for the permitted action.

A tweak record with this name (Allow_topo_multipoint) can be added for each Client-type, overruling this setting using the Num value. This permits the configuration where the default action is not-permitted, but where for one or two Client-types this operation is granted.

Change_hostname

When Num value is not 0, a Hostname change is not restricted to its node-type function

The Node details form will normally not allow you to alter a Hostname because it was generated

within the constraints of the Node-type definitions.

If this tweak is enabled, the Hostname can be changed into any free hostname.

Without this tweak, the permitted name changes will be limited to the Node-type definition restraints.

Login_legal_message

If the description is set, and the numerical value is set to 1 then this text is shown at the bottom of the login page. This allows you to specify a legal message to show to users attempting to authenticate.

This feature supports html tags such as <b> and

Debug_max_days

Maximum age in days before ending any NetYCE debug mode. Default = 1 day (24h), Forever = 0

NetYCE uses two flag-files to control the debugging level of nearly all of its components. The **debug** level is enabled by creating the file /opt/yce/etc/yce_debug. When it exists, the various daemons will create additional log files in /var/opt/yce/logs/ which will have file

names like `yce_<daemon>_debug.log`.

The **development** level is enabled by creating the file `/opt/yce/etc/yce_development`. When it exists, the file `/var/opt/yce/yce_dev_debug.log` is created. It will mostly contain very detailed information on the interaction with the devices.

Due to the amount of information that will be created in these log files, they are automatically emptied out daily by the `log_maint.pl` script at 22:15 (see [Logging aging tweaks](#)).

To prevent the generation of debug and development log files over longer periods of time, the tweak 'Debug_max_days' is by default set to 1 day. This implies that the yce_debug and yce_development flag files will be removed 24 hours after they were created.

Only when these logging levels are required for longer periods should this tweak be set to more days. If set to '0', the flag files will never be removed.

Default_site_status

Num_value defines the initial site status (1=active, 2=planned, 0=obsolete, ..

When creating a new site it must have an initial status. This initial status is defined using this tweak. The default is '1' which is translated to 'active'. These translations are also defined in the Lookup form but under the class 'Translation'. The entries using de variable `SiteStatus` define the available site statuses and their corresponding translations.

Lookup

Filter class	Translation

Class	Variable	Num	String	Description
Translation	Service_status	5	update	Service status translation
Translation	SiteStatus	0	obsolete	Site status translation
Translation	SiteStatus	1	active	Site status translation
Translation	SiteStatus	2	planned	Site status translation
Translation	SiteStatus	3	merging	Site status translation
Translation	SiteStatus	4	migrating	Site type translation
Translation	SiteStatus	5	inactive	
Translation	SiteType	3	CN-	Site type translation
Translation	SiteType	11	HK+	Site type translation

+ 🗑

Variable	SiteStatus	Class	Translation
String	active	Num	1
String b			
Description	Site status translation		

Apply   Close

Remove_deleted_supernets

Enabling this tweak will delete the supernets from the database when no longer in use. The default behavior will not delete these supernets, so they are available for re-use without re-adding them. This also allows for reservations which are to be used at a later point in time.

Send_email_to

With this setting you are able to define when an email is send, if it should go to the user defined email address, the group defined email address or both.

When the setting is set to 'user' and the email address is not defined it is send to the group email address.

Sched_ignore_locks

This setting can disable the distributed schedulers "node-lock" behaviour. Jobs will be kept in a 'waiting' state while another job on that same node is still running. These global node-locks are released after 10 minutes.

The scheduler will not wait for jobs running on same node when Num_value > 0. The default is '0'. This setting applies to all NetYCE servers and requires the 'yce_sched' process on each server to be restarted to take effect.

Main_substring_search

The 'Search' function of the main Build overview form allows the user to locate clients, sites, nodes, services and vpns using several fields of these objects. The search string can optionally include wildcards (* and ?).

When NOT using wildcards, the search string is generally taken as a literal match, not as a substring. The 'Main_substring_search' tweak, when set, will change this into a substring match.

The initial value of this tweak enables the 'substring' match.

Filter_cmdb_nodes

CMDB nodes that are assigned a client-type will be included in the node grid of the main Build overview form. As these CMDB nodes cannot be manipulated from within the grid, it might be desirable to filter the CMDB nodes from the Node grid.

The 'Filter_cmdb_nodes' tweak, when set, will hide the CMDB nodes from the node grid.

The initial value of this tweak permits the CMDB nodes to be included in the view.

Template_markers

We added support for the variables <current_template> and <current_vendor> that will have the active template name and the verdor during its generation. This allows for marking the start and end of each template and can be used in conditionals.

However, a few shortcomings with this approach were observed: First, all templates will need modification - an unpleasant task. Second, when a sub-template is done, config generation returns to its parent template to process the remaining lines of that template. It is often hard to locate which template that was, especially when the template tree has multiple levels.

To simplify these issues, this "Template_markers" Tweak inserts automatic 'current template' comments that use these new variables. It will insert comment lines in the generated output at the template 'start', template 'done' and template 'resume' places.

The inserted comments are controlled by the Num_value field of the "Template_markers" Tweak: When it equals 1 - print 'start' template info; when 2 - also include 'done'; when 3 - also include 'resume' comments. The default is to include none (0).

With hierarchical templates, the message text indentation follows the levels. The example below shows the inserted comments when all three types are enabled. The normally generated configuration lines were removed for calrity:

```
# -- Template start - HP5800_24_core | HP_C5
# --  Template start - Basis | HP_C5
# --  Template done - Basis
# -- Template resume - HP5800_24_core
# --  Template start - Network_mgmt | HP_C5
# --  Template done - Network_mgmt
# -- Template resume - HP5800_24_core
# --  Template start - Sflow | HP_C5
# --  Template done - Sflow
# -- Template resume - HP5800_24_core
# --  Template start - DHCP | HP_C5
# --   Template start - DHCP_RCO | HP_C5
# --   Template done - DHCP_RCO
# --  Template resume - DHCP
# --  Template done - DHCP
# -- Template resume - HP5800_24_core
::etc::
# -- Template resume - HP5800_24_core
# -- Template done - HP5800_24_core
```

Logging aging/ageing tweaks

NetYCE maintains many log files and logging data tables on its system(s) and database. All logs are size/age rotated or capped by a maximum age. The Tweaks in this category allow the system owner to modify the default aging parameters for the various logging types.

All (numeric) values are in days.

The logging aging is maintained by the daily logging cleanup script, `bin/log_maint.pl`, that is executed daily using the `cron` facility. It starts at 22:15 by default.

This logging cleanup script itself logs all its actions in `/var/opt/yce/logs/yce_maint.log`. This logfile rotates at 500.000 bytes.

The default crontab entry (from the `yce` user!) is listed below:

```
#--- YCE daily maintenance -------------------------------
#    Log, Job and Output directory cleanup
15 22 * * * /opt/yce/bin/log_maint.pl > /dev/null 2>&1
```

Age_action_logs

Do not keep Action_log entries older than Num_value days. Default = 400. The String value has no function and is ignored.

The Action_log is a table in the database where user activity is logged. It also has the entries created by jobs using scenarios with the "log_action" command.

This tweak is used by the daily logging cleanup script, `bin/log_maint.pl`.

Age_config_logs

Do not keep Config_log entries older than Num_value days. Default = 100. The String value has no function and is ignored.

The Config_log is a table in the database where Job configuration sessions are logged.

This tweak is used by the daily logging cleanup script, `bin/log_maint.pl`.

Age_nccm_data

Do not keep Nccm configuration entries older than Num_value days. Default = 800. The String value has no function and is ignored.

The Nccm log is a table in the database where polled node configuration changes are logged.

This tweak is used by the daily logging cleanup script, `bin/log_maint.pl`.

Age_node_logs

Do not keep Node_log entries older than Num_value days. Default = 400. The String value has no function and is ignored.

The Node_log is a table in the database where the responses to node 'show'-type commands are logged.

This tweak is used by the daily logging cleanup script, `bin/log_maint.pl`.

Age_custom_reports

Do not keep Custom report results older than Num_value days. Default = 30. The String value has no function and is ignored.

Custom report results are removed from the database when they exceed this age. This applies to all individual report result entries. A one-off report is removed when it was created after this many days as is the dated report (with the <date> appended). The date listed in the YCE.Images.Image_filename is used as the creation date and applies only where YCE.Images.Image_name ends in ".rep".

This tweak is used by the daily logging cleanup script, `bin/log_maint.pl`.

Age_result_files

Do not keep Job log files older than Num_value days. Default = 30. The String value has no function and is ignored.

The result files are primarily the local Job log files. Each job that is executed on a server has a unique environment (the Job_id) under `/var/opt/yce/jobs/` where running files are maintained. This tweak affects the period that these files are preserved.

This tweak is used by the daily logging cleanup script, `bin/log_maint.pl`.

Age_task_logs

Do not keep Task_log entries older than Num_value days. Default = 100. The String value has no function and is ignored.

The Task_log is a table in the database where the AMP API requests and responses are logged.

This tweak is used by the daily logging cleanup script, `bin/log_maint.pl`.

Age_temp_files

Do not keep temp files older than Num_value days. Default = 15. The String value has no function and are ignored.

The temp files are primarily the local Job session files. Each job that is executed on a server has a unique environment (the Job_id) under `/var/opt/yce/jobs/` where running files are maintained. This tweak affects the period that these files are preserved. This tweak is used by the daily logging cleanup script, `bin/log_maint.pl`.

Archive_count_yce

The NetYCE database archive tools maintain the number of archives that may exist for the archive types, YCE and NCCM. Automatic daily archives of both YCE and NCCM are now standard and takes place at 23:00 or 23:20 by default (for primary or secondary server respectively).

The maximum number of archives for each type are controlled using Tweaks in the Lookup form. The '**Archive_count_yce**' variable defines the number of historical YCE database archives that are maintained. The default is '15'.

All archives of each type count against these values, regardless if they were created manually, uploaded, or created automatically. The only archives NOT counted (and therefore not deleted) are uploaded archives where the filename was modified.

The deletion of excess archives takes place after each archive creation, regardless of type. This tweak is used by the database archive tool `/opt/yce/bin/dbarchive.pl`.

Archive_count_nccm

The NetYCE database archive tools maintain the number of archives that may exist for the archive types, YCE and NCCM. Automatic daily archives of both YCE and NCCM are now standard and takes place at 23:00 or 23:20 by default (for primary or secondary server respectively).

The maximum number of archives for each type are controlled using Tweaks in the Lookup form. The '**Archive_count_nccm**' variable defines the number of historical NCCM database archives that are maintained. The default is '15'.

All archives of each type count against these values, regardless if they were created manually, uploaded, or created automatically. The only archives NOT counted (and therefore not deleted) are uploaded archives where the filename was modified.

The deletion of excess archives takes place after each archive creation, regardless of type. This tweak is used by the database archive tool `/opt/yce/bin/dbarchive.pl`.

Ldap_idle_account_age

This setting allows NetYCE administrators to automatically remove Ldap-based user accounts that have been idle for a defined period. The 'Ldap_idle_account_age' Tweak variable can be set to the number of days that an Ldap account is considered idle. If the last-login date of an account was before this age, the account is removed. This will be performed daily as part of the nightly maintenance tasks.

By default this setting has the value '0', which will skip Ldap account aging completely. This setting applies only to 'ldap'-type user accounts, 'local' users are not affected.

Splunk log export

NetYCE uses its database to maintain its main logging entries. Since these logs are not directly available for logging analysis tools like **Splunk**, a facility was created to export these logging tables to parsable logging files. The next three tweaks enable or disable this facility for each of the three logging tables.

The exporting facility is implemented in the NetYCE daemon process `yce_skulker`. Amongst its many tasks is the exporting of the log files. It will examine the logging tables marked for export every five minutes and append the new items to the export files.

At date rollover, new files will be opened and the older renamed to provide a 10-day history (as all log files are rotated within NetYCE).

Any NetYCE system that updates to a version made after 17 Jan 2023 will automatically enable the following three tweaks:

Export_action_logs

When number set to 1, the Action_logs (users) will be exported to log files

The exported log file is `/var/opt/yce/logs/yce_action.log`. Logrotation will create ten files using the extensions '.0' thru '.9'.

Export_config_logs

When number set to 1, the Config_logs (jobs) will be exported to log files

The exported log file is `/var/opt/yce/logs/yce_config.log`. Logrotation will create ten files using the extensions '.0' thru '.9'.

Export_task_logs

When number set to 1, the Task_logs (api) will be exported to log files

The exported log file is `/var/opt/yce/logs/yce_task.log`. Logrotation will create ten files using the extensions '.0' thru '.9'.

Password Tweaks

Password validation rules are defined using the Lookup class "**Password**". These settings are described in Lookup Password

Vendor module tweaks

Commit_pending_policy

When Num_value set to 1, commit based vendors will issue commit commands during command jobs: in case this is not desired, set the Num_value to 0.

Only commit based vendors support this option, which are currently: Juniper, Palo Alto, Huawei CE and Cisco XR.

Backup / CMPL Tweaks

Backups

| Polling groups | Polling status | Configuration | Search | Command replies | **Settings** |

Class: [▾]

Variable	Str value	Num value	Class	Description
Cmpl_max_children	*	3	Constants	Compliance max children
Default_cmpl_rule_s…		1	Constants	Default severity for compliance rules (Medium)
Nccm_max_children	*	3	Constants	Nccm deamon max children alive at the same time
Nccm_poll_interval	*	300	Constants	Nccm deamon poll cycle length
Cmpl_condition_incl…	Must contain	0	Translation	Compliance condition include type
Cmpl_condition_incl…	Must not cont…	1	Translation	Compliance condition include type
Cmpl_condition_incl…	Must contain …	2	Translation	Compliance condition include type
Cmpl_condition_incl…	Must contain …	3	Translation	Compliance condition include type
Cmpl_node_scope	yce	0	Translation	Compliance node scope
Cmpl_node_scope	cmdb	1	Translation	Compliance node scope
Cmpl_policy_schedu…	disabled	0	Translation	Compliance policy schedule status

[New] [Delete] [Search]

Variable: [Nccm_max_children]   **Class:** [Constants ▾]

Str value: [*]   **Num value:** [3 ▴▾]

Str value b: [ ]

Description: [Nccm deamon max children alive at the same time]

[Save]

The backup and compliance tweaks can be found in a different section of the application. They can be found in the Backups → Settings form. They are separate in order to allow for a separated database for these functionalities, since they deal with a lot of data, making the overall product more stable.

NCCM tweaks

The NCCM process retrieves periodically the life configuration of devices and stores them to report on observed differences over time.

The NCCM uses a number pollers to retrieve the configuration from the devices. Each poller is tasked

to sequentially contact a series of nodes and interactively retrieve the config, find any differences and stores them if any.

The user defines the polling interval and the total of number of nodes the server has to poll using the NCCM front-end tools. The system then determines how many pollers is needs based on the number of nodes and the time a poll requires.

Each time a configuration is retrieved, the new configuration is compared against a **base-configuration**. Any differences between the the base and the current is then stored. After a while a new base config is established and the differences against that base are then stored. This method of configuration storage has the benefit of low storage volume and minimal retrieval processing (never more than 4 records and two re-builds).

To determine if a new base configuration should be created or only the differences stored, three criteria are used: number of diffs, age of the base, and size of the diff. These three criteria are set using three corresponding tweaks.

To control the number of pollers two tweaks are provided: to limit the maximum number of pollers and the average time to poll a node.

Nccm_max_age

Maximum age in days of a base-configuration. Set the Num_value. Default = 7

When the current base configuration exceeds this age in days, a new base configuration is created.

Nccm_max_diffs

Maximum number of stored config-diffs before a new base-config is created. Set Num_value. Default = 36

When there are more than this number of diffs stored for the current base configuration, a new base configuration is created.

Nccm_max_diff_size

Maximum size in bytes of a config-diff before a new base-config is created. Set Num_value. Default = 8192

When the diff exceeds this number of bytes, a new base configuration is created.

Nccm_max_pollers

Maximum number of NCCM pollers per system. Set the Num_value. Default = 12

By default no more than 12 pollers are permitted to run on a server. If more pollers are required

for the number of nodes, additional server(s) could be configured to poll the remaining nodes. Or the polling interval could be increased.

The limit of 12 is based on the common hardware specifications of the NetYCE servers, but could require tuning for the systems used or the load given for other tasks.

Nccm_poll_duration

Average duration in seconds of a NCCM config-poll. To calculate required number of pollers. Default = 15

In practice, the average duration of a configuration retrieval is 15 seconds. For slower devices, larger configurations or restricted bandwidth, this number might be increased.

If set too low, the pollers will still poll all their nodes but will simply skip any idle periods that is will otherwise use to maintain the desired pace. The effect is that the polling interval will be (somewhat) longer than the configured number of hours.

NCCM_max_children

The maximum number of cmpl children the nccmd daemon can spawn. The nccmd daemon spawns a number of compliance and nccm children to take care of its tasks concurrently, and they both have their own upper limit on how many children can be spawned. For compliance this is Cmpl_max_children.

Delete_polling_group

Whenever you delete a polling group:

- 0: delete all its nodes from Nccm_selection or
- 1: delete only nodes that do not belong to a polling group anymore.

Polling_interval_translations

Controls the values for the next poll interval dropdown in the polling groups form. Options are separated by '|'-pipes, and separated by commas. The first item denotes the number of hours until the next poll, the second the label.

Note that if you enter the wrong syntax, it will mess up the form, so be sure to make a backup if you want to modify this value.

CMPL Tweaks

Cmpl_max_children

The maximum number of cmpl children the nccmd daemon can spawn. The nccmd daemon spawns a number of compliance and nccm children to take care of its tasks concurrently, and they both have their own upper limit on how many children can be spawned. For nccm this is Nccm_max_children.

Cmpl_rule_severity

Rules have their own severity, and their own colors to denote low priority compliance erros, medium, high and serious. You can customize this and their colours with these options.

Default_cmpl_rule_severity

When a new rule is created, this is the severity that it will have by default, corresponding to the numerical value of Cmpl_rule_severity. By default it is set to 1 (Medium)

Default_signal_trigger

When a new policy is created, these are its signal triggers that are set by default. There is a record set for every state change possible. The string values correspond to:

- **C2C**: Compliant to compliant
- **NC2C**: Non-compliant to compliant
- **C2NC**: Compliant to non-compliant
- **NC2NC**: Non-compliant to non-compliant

If the value is set to 1, a signal will be triggered on this state change. If not, a signal will not be triggered.

Default_signal_type

Sets the default signal type. A signal type is a bitwise value, where the first four bits matter. Their meanings are:

- **Bit 1**: A trap message
- **Bit 2**: A syslog message
- **Bit 3**: An email message
- **Bit 4**: A Rest-api 'post' call using Json

The default is set to 1, sending just a trap message.

NOTE: we do not recommend setting the third bit, Email. Compliance result signals are sent out per node, per policy. If you don't watch out you could set off a ddos attack with an overwhelming number of emails.

Cmpl_censor_config

Denotes whether or not we should use censored configs (1) or uncensored configs (0) when checking compliance for a node.

The default value is 0.

Censored configs have all passwords and sensitive information filtered away as **censored**, preventing this information from returning into your reports. If this is not an issue you can set it to 0.