NetYCE 7.1.1 Build_20190501

Release notes

Date: 2019-05-01

> Note: This general release is the result of all the various intermediate builds created under the version 7.1.0. The 7.1.0 was never general released due to the drive to create a very stable product. The 7.1.1 version label was used to signify the difference.

Enhancement

Management tab

The Node Details from now has a third tab 'Management'. This tab is defined to select from the various configured Loopback and Management interfaces which address to use for the device management. It allow to select one Ipv4 and one Ipv6 address for this purpose.

Previously the explicit selection was not possible and was performed by an algorithm.

Basic command jobs

The tool 'Basic command jobs' is totally reworked. The new tool allows the operator to create node jobs for all supported devices (vendors) that exist in the network. The tool is organised around nodes added to the 'CMDB' of NetYCE but also accepts ad-hoc nodes that are neither part of the YCE or CMDB environments.

The Basic command jobs support many of the capabilities of the fully modelled YCE nodes. It supports (multi-vendor) templates, command and configuration parsing, NCCM history and all scenario functions.

The main difference with the standard 'Command jobs' is that the amount of information about the fully modelled nodes is significantly more extended than those in the CMDB or ad-hoc nodes, resulting in less powerful templates. But for jobs requiring only a handful of variables, the basic command job will allow for very quick deployments.

IPv6 communication

NetYCE is now fully IPv6 compatible. The communication with all supported devices (vendors) can use both IPv6 and IPv4 depending on (configured) availability.

Also the NetYCE system can now be configured using dual-stack communication for IPv6 and IPv4.

RedHat / CentOS 7 support

The support of Linux versions is now extended with RedHat or CentOS 7. The distribution set will support both versions 6.x and 7.x. Installation guides for version 7.x are in progress.

> Update: The work on Rhel/CentOS 7.x support is as yet unfinished. The first production trials are currently initiated. Among the work in progress are the network-setup scripts and the installation documentation. A preliminary VM for trials is available on request.

Node Groups

Most of the tools in the 'Operate' menu are using a wizard-like style where the initial page is used to select the nodes the tool will work on. The selection of these nodes is predominantly Client-based. The operator selects the ClientCodes from a list and the corresponding nodes are selected by the tool.

This method has now been replaced by 'Node groups'. A Node group is a dynamically determined selection of nodes using a set of criteria. This set of criteria is defined by the customer and saved for use in all available tools.

In this initial release the form to define the Node groups is not yet available and only a predefined number of node groups will be created at installation time. These predefined groups will include a Node group for all existing Clients to offer a direct replacement for the Client-bases node selection the users are familiar with. In addition there will be groups for vendors, domains, regions and such. When the Node group setup form is available the customer can define an unlimited number of very specific groups using combinations of many different criteria.

The modifications to support these Node groups is a work in progress and has so far been incorporated in the tools 'Command jobs', 'Basic command jobs' and 'NCCM poller config'. Others will follow shortly, as will the Node group setup form.

> Update: The Node-group setup form and the Node-group test tool are incorporated in the latest versions. Extending Node-groups to other tools has not yet been realised.

CMDB Form

The CMDB form in the Operate menu has been re-implemented and redesigned. The form design is now more in line with most of the netYCE front-end and because the form fields now also align with the netYCE fields used in other objects, should be clearer and more intuitive to use.

The underlying cmdb database table was moved from the custom NMS database to its own CMDB database in preparation of extending the CMDB to include network discovery. The new 'CMDB.Cmdb_nodes' table uses different field names from the original 'NMS.CmdbNodes' table and is automatically migrated when updating the netYCE installation.

All tools, integrations and service-types using the old table were modified to use the new format.

Node group forms

A new form has been created to allow you to manage node groups. It can be found beneath the 'Logs'-option in the Operate menu. With the new form you can: - View, create, edit, duplicate, delete, import, export and search node groups - View, create, edit and delete node group rules - View, create, edit and delete node group conditions

Please consult the Wiki page on Node groups for details.

Default sub-menu tool selection

Many of the items in the 'Operate' menu have a selection of operational tools presented in a sub-menu. One of these tools is selected when selecting the menu.

The tool initially selected can now be customised per installation. To this end a series of Lookup entries have been added in the 'Tweaks' class, one for each sub-menu.

The tool name specified in the tweaks string-value will override the standard default tool selection. The various Lookup variable names all use the 'Default_tool_' prepended to the menu name. Thus, 'Default_tool_report' specifies the tool name initially selected in the 'Report' menu, and 'Default_tool_node_config' will do the same for the 'Node config' menu.

When changes are made to these settings, the action button `Regenerate config` from the 'Admin - System - System status' tool needs to be used. It executes the command 'yce_setup.pl -r' to re-create the menu and configuration files needed.

Systems-page Relaunch option

The NetYCE processes use several configuration files that are created at setup time and are regenerated after software installation updates. This regeneration of configuration files and relaunching of its processes is also required to activate some customisations.

For instance the changing of the default tools in sub-menus or the altering of the header bar for servers requires this action.

The 'Admin - System - System status' tool now has the action button labeled 'Regenerate config' that will perform these actions. The execution takes up to a minute during which the NetYCE application will not be available. Be aware that it might disrupt running tasks since many processes will be restarted.

Node type form persistent search

The node types, node type records and node classes grids now have persistent search.

Build service search

The search function in the main form has been updated to also include services. You can now search for service names. Since some of these searches can be quite long, these searches have been capped at 100 results. When searching for VPNs, you now also see a list of services belonging to the found VPNs.

Ldap and Active Directory support

NetYCE has been supporting Ldap for years and to some degree Microsoft Active Directory (AD) as well.

However, the AD implementations frequently were case for customisation due to the complex AD schemas used by the customer. To deal with these situations the LDAP and AD support has been reworked extensively in order to be more flexible towards AD schemas.

Most of the existing NetYCE ldap setup profiles will continue to work, but will require pre-production testing and, probably, tuning. The related Wiki documentation is already updated and should be reviewed before deployment. Please the consult Ldap / AD setup page.

Last login view

To provide a quick overview of the active NetYCE user accounts, the column 'Last login' is added to the user-grid of the User form. This column lists for all users the date and time of the most recent login. If the account was never successfully accessed, its value will be '-never-'.

The last login will show an extra asterix ('*') when that user has an active login token.

Arista EOS vendor-module

A preliminary vendor-module has been added to the list of supported vendor modules: Arista EOS. This vendor module is mostly functional and uses the Arista CLI. Support for OS upgrades and command parsing are not yet available.

It should be noted that the module has only been tested against the virtual Arista vEOS system. Any feedback on its functioning is welcome.

Scenario 'stop on-error'

The scenario syntax has been extended to allow the scenario to stop execution of the scenario and report a 'ABORTED' job status. Normally the scenario is always 'Successful' unless a 'stop' command is encountered, usually after testing if the 'error'-flag was set.

The command 'stop on-error' will from that point in the scenario automatically abort if any command fails. Note that the automatic abort will ignore any error handling that is present in the scenario.

The sample scenario…

```
reachable -n <node>
if error
   log -m "node <node> is non-responding"
   stop
endif

cmd_exec -n <node> -f <node>.cmd -v
if error
   log -m "command failure"
   stop
endif

end
```

...can be simplified using the new 'stop on-error'

```
stop on-error

reachable -n <node>

cmd_exec -n <node> -f <node>.cmd -v

end
```

At the point where the normal error handling by the scenario must be resumed the 'stop default' command can be inserted.

Patch administration

The NetYCE database is currently divided in several table schemas: YCE, LOGS, CMDB, NCCM, and NMS. Because each database schema can be restored individually, the existing method of keeping track of versions and patchlevels no longer suffices. Therefore, the patch administration was changed from a global patchlevel to a per-database schema patch administration.

Each database schema now has an additional table, named <schema>_patches, that keeps track of the patches installed. For patches that modify the server setup as similar administration is maintained in the file /etc/system_patches.xml.

This setup adds reliability when patches fail or partial archives are restored - it ensures all relevant patches are applied.

Debug logs

The new 'Admin - System - Debug logs' tool allows users with Manager of System privileges access to the log files of the NetYCE server. It also provides the user to enable the 'debug' mode which creates additional and extensive log files.

Log files can be viewed using a pop-up window from this tool. The 'follow' option shows the log file

in a scrolling window as information is added to it.

Additionally, the standard and debug log files combined with server configuration files can be converted into a 'NetYCE support file' to provide the support team with the relevant information to a reported problem. To ensure confidentially, each support file is encrypted on creation to allow secure email of the data.

Scenario Syntax Hash Variables

The syntax for hash variables has been altered. Hash variables used to be called with a '@'-character, like <@hash>, <row@hash> or <row.attribute@hash>. This has now been changed to use the '%' character: <%hash>, <row%hash> or <row.attribute%hash>.

A patch has been made to automatically convert all instances in Scenarios, but NOT Stored_jobs The syntax for calling relations has NOT been changed. In this way, it has become clearer what kind of namespace you're referring to.

Command parsing tester

A new tool has been included into netYCE in which you can test your command parsing templates. It can be found in the design menu, when you click on the template trace item. There, you can find the tool as the second option in the horizontal menu.

Search for your template, select it and put the output you're trying to parse in the 'Command output'-field, and click on Parse. This shows you the parsing process. You can also edit your template if something is wrong, and it will be updated in realtime.

Scenario Help

The scenario form has been extended with a context-sensitive help function. The function uses the current cursor location in the scenario field as you type or move. The help-function is intended to show you the available scenario commands and its associated options.

Whenever your cursor rests on, or one space right of a scenario command, it shows you a description of the command and the options supported by it. It includes a syntax help for all scenario functions like if, then, foreach, sort, grep, etc.

The help-function (and the syntax-highlighting) also supports (custom) tasker plugin modules that you might have created.

Automatic Node-groups for Clients

The definition and maintenance of Node-groups is primarily the task of the NetYCE operators as they can now create group definitions that best match their workflow and organisation. For this reason, Node-groups definitions and maintenance is a manual task that should only require occasional attention.

One exception was identified for some types of NetYCE customers: 'Clients'. When with high regularity new Clients are created or removed, support for automatic Node-group maintenance for Clients would be a time-saver.

For this purpose a Lookup tweak was added using the Lookup Class 'Node_group'. The entry for 'Update_clients' controls now if a new Node-group using the client's ClientCode is created or not. Likewise, it controls if on removal of the client any Node-group conditions using 'ClientCode=<client_code>' are to be removed. The 'Update_client' Str_value determines the Node-group tag and a non-zero Num_value enables this feature.

Service-type additions

A number of port-location Service-type commands have been added. All variants support wildcards (* and ?). The PORT_NAME versions can match either on the internal name (Gi01/00/24) or the name used by the vendor.

```
# LOCATE - PORTS - <node>     - PORT_NAME   - value  -> <portlist>
# LOCATE - PORTS - <portlist> - PORT_NAME   - value  -> <portlist>
# LOCATE - PORT  - <node>     - PORT_NAME   - value  -> <port>
# LOCATE - PORT  - <portlist> - PORT_NAME   - value  -> <port>
# LOCATE - PORTS - <node>     - -attribute- - value  -> <portlist>
# LOCATE - PORTS - <portlist> - -attribute- - value  -> <portlist>
# LOCATE - PORT  - <node>     - -attribute- - value  -> <port>
# LOCATE - PORT  - <portlist> - -attribute- - value  -> <port>
```

Additionally, the 'Assign - Ipv6_net - Port' service-type options have been extended to the following list.

```
# ASSIGN - IPV6_NET - <ipv6net>  - PORT     - <port>     -> x
# ASSIGN - IPV6_NET - <ipv6net>  - PORTS    - <portlist> -> x
# ASSIGN - IPV6_NET - <ipv6net>  - LINK     - <portlist> -> x
# ASSIGN - PORT     - <port>     - IPV6_NET - <ipv6net>  -> x
# ASSIGN - PORTS    - <portlist> - IPV6_NET - <ipv6net>  -> x
# ASSIGN - LINK     - <portlist> - IPV6_NET - <ipv6net>  -> x
```

Ipv6Add() template function

The function “Ipv6Add()” has been added to the configuration generator for use in templates.

The function adds an offset to an IPv6 base address. The Ipv6 base address can include a /prefix which causes the base address to be normalised to the corresponding size before the offset is added.

The offset may have an Ipv6 format (eg ::1 or ::abba) or numeric (0-65536). If the offset has a negative sign included, the offset is subtracted form the base address, should the base address include a /prefix, the offset is subtracted from the end of the normalised subnet.

Please consult the Wiki page on Functions - Ipv6Add for details.

Custom 'subnet-plan'

When creating a 'Custom' Ipv4 or Ipv6 subnet the Ip-plan with the id '0' was assigned. It is now possible to create subnet definitions within this 'Custom' ip-plan. Since the 'Custom' subnets are all compltely ad-hoc and not bound to network ranges an sizes, Ip-plan '0' does nothing to validate the network address and such, but it will help you in calculating the various offsets that are available for all subnets.

By creating a subnet type in ip-plan '0' of a chosen name, and assigning the subnet plan point-to-point and four-corners offsets, the creation of a custom subnet using that name will automatically set the appropriate ip-values using these offsets.

Likewise, the default set of offsets used for a subnet name that is not defined in ip-plan '0' will use the 'default-custom' subnet definitions. This applies to both Ipv4 and Ipv6.

Login form

In some cases the login in the front-end results in blank grids or immediate forced log-out after selecting a tool. The reasons for this behaviour differ, but are usually related to server setup or server processes not being available.

Where possible, the conditions that might lead to failing logins are now tested before the login page is presented to the user. If any of these prerequisites are not met, a message is displayed to inform the user. Logins are not possible until these conditions are satisfied. Every 10 seconds these tests are repeated and if successful, logins are granted again.

Custom subnet offsets

For the custom subnets (subnets not extracted from a supernet using an ip-plan) an additional function was (re)implemented: custom subnet offsets.

This is a feature available for ip-plan based subnets where subnet offsets are calculated from the network-address. Now custom subnets (that by definition always use plan-id '0') can be added to plan-id '0' and then be assigned a subnet plan with 'Point-to-point' and 'Four-corners' attributes.

When a custom subnet is created of the same name, the offsets are calculated using these definitions. Likewise the subnet 'Default-custom' can be created to have default offsets calculated.

This function is available for both IPv4 and IPv6.

Configuration generator

Templates using Relations that do net exist or resulted in SQL errors were quietly taken as Relations that had no results. And therefore (correctly) did not result in configuration lines. But since no errors were raised either, the user received no feedback on the failure.

Now missing or erroneous Relations will raise an error in similar fashion as the parameters in the

majority of cases where relations can be used. When the Relation truly has no results the configuration line is dropped as is expected.

Another enhancement made to the configuration generator is a special case of the iterative template generation ({<par@relation>, relation}). This syntax can be extended to include a filter by column value: {<par@relation>, relation, col='value'}. This syntax has been extended to allow the 'value' to be a variable or relation: {<par@relation>, relation, column=<var>} or {<par@relation>, relation, column=<var@relation2>}.

Such constructs could be used in previous versions with limited success, but now the scenario and configuration generator share variables these applications are more formal and require explicit implementations.

Scenario 'Relation'

In scenarios the 'relation' command can be used to extract a variable (list) from a named relation, and returns this to a list variable in the scenario.

This 'relation' command has been extended to support 'filter' (-f) options and a 'limit' (-l) option.

```
 [-f <column=value>   ]   define filter column=value pairs. The value
supports wildcards ('*' and '?')
                          the relation row must match all filters to be
included in the result
 [-l <limit>]             return the <limit> number of entries. The limit
must be > 0
```

Please consult the Wiki page on Scenario Commands details #Relation for details

Remote Custom reports

Custom reports allow for the periodic creation of database queries that can be retrieved using the NetYCE GUI or remotely using an API call. When using the API call `fetch_report` the latest result of the named report is converted to XML and downloaded.

For cases where the report results are required to be up-to-date, a new API call, `run_report` was created where the report is first executed (on demand) and immediately sent back in XML format.

SSL certificates

Administrators wishing to use HTTPS to communicate with the NetYCE front-end must install a SSL certificate. These certificates must be self-generated (self-signed) or authorised by a CA (Certificate Authority).

To simplify the creation of the self-signed-certificate or the certificate-request, the tool `mk_ssl_cert.pl` is available as a cli tool. This menu and prompt driven tool will help administrators to create the required CONF, KEY, CSR and CRT files.

Change

Job timeouts

When a job executes configuration or command lines on a devices CLI, the system expects to receive the device prompt within the timeout period. Since the timeout needed is different and hard to predict a job would occasionally fail die to a prompt timeout.

To reduce these occurrences, the way we determine a timeout was modified. Now when a command is in progress the timeout counter is restarted each time a bit of response is received. For lengthy commands issuing progress messages this will prevent timeouts. In the absence of progress messages the original timeout behaviour is unchanged.

Operator permissions

The permissions for 'Operator' level users were for several of the tools in the 'Operate' menu insufficient. The privileges for the operator level have been modified to include the execution of command jobs and all its related information.

New LOGS database

On systems with high levels of activity, the database tables supporting the four log types of activity logs can grow very fast. When combined with long ageing settings (using Lookup tweaks), these tables have a tendency to become very large.

Performing daily maintenance on these tables requires extensive resources, occasionally resulting in table corruption and database synchronisation issues (for redundant system setup).

To reduce these scalability issues, the four logging tables have been moved from the YCE database to a dedicated LOGS database. In this new database each log type will now have its data stored in week-tables. A table will be in existence for every week of the retention period for that type. The data is accessed though so-called 'merge' tables so that the data appears to have one source but is in fact using multiple files and indexes. This technique will reduce the maintenance to dropping a week-set of files once a week.

In addition, each log table is now used in a 'write-once' mode. Log entries cannot be individually removed from the database. The 'Custom data' tool grants read access to the individual and merged log tables.

Fix

Enable secret field

The 'Enable secret' field in the node details from would show you the secret as plain text if you had the permissions (manager) to change its value. The display behaviour has now been changed to show the value as a series of bullets. Only when the cursor is placed in this field will the value be shown as clear text. This behaviour is now in line with similar passwords fields in the Domain form.

MPLS Delete bug

There was a bug where MPLS Vrf's could not be deleted. This is fixed now.

Subnet Create Vlan Bug fixed

There was a bug where in some databases, the Create Ipv4 Subnet form vlan template dropdown showed duplicate entries. This is fixed now

Lookup admin filter

In the lookup form, you are now able to change your class filter back to nothing after using it.

Template port slot help

The template port slot forms pointed to outdated urls. This is fixed now.

Domain name change fix

Node types are now updated when you change the name of a domain. Plus the Domain form had some stability issues when doing this. This is fixed now.

Custom attribute menus double values

You could previously add profiles in custom attribute menu items that had duplicate key values. This is no longer possible.

Node type Par_group selection

When editing the Par_group value of a node type, you can now select parameter groups that do not have any values yet.

IPv4 Subnet Plan Grid

In the IPv4 Plan form, whenever you tried to fill in one of the values of a subnet plan, you would always have to enter this in a normal text form. This was rather inconvenient for topology positions, or scopes, since one small typo would result in an error elsewhere. For scopes and topology positions you now see a dropdown with options that you can select.

Service type form validation

There was a bug where if you edited a service type record and then deleted it, and then switched service types you would see a continuous message asking if you want to save your changes. This is fixed now.

Unresolvable fqdn fix

A nasty bug was introduced with the CMDB job handling. Jobs attempting to connect to a node that was configured to be connected using only a full-qualified-domain-name (fqdn) that was not DNS resolvable caused the job to fail with an obscure error message indicating an '_EXIT_' label could not be located.

This issue is fixed now.

Command and Config parsing on CMDB nodes

When executing a job scenario involving command parsing or configuration parsing on a CMDB node, this job would fail due to an undefined Vendor-type.

This issue was caused by an overlooked dependency when building the scenario support for CMDB nodes and is now fixed.

Job node-address resolving

All node communication sessions rely on resolving node name to an ip-address. This applies to stored node-fqdn and configured management addresses but also to explicitly provided node-fqdn and ip-addresses that can override the stored values. Only when all entries are resolved and prioritised (ipv6 precedes ipv4) will a session be attempted.

A problem was found in the most recent build when not explicitly providing overriding fqdn or

addresses: the stored node-fqdn would be interpreted as an overriding fqdn. If that fqdn could not be resolved in an ip-address no communication would be possible.

The proper use of stored and overriding fqdn fixed this problem

Port-template list

The port-list grid of a node could show multiple rows for the same port although the port really was configured to be unique. The issue was caused by using the same port-template name in different client-types. An additional row would be displayed for every client-type using the same port-template name.

By consistently restricting all port manipulations to the same client-type and vendor-type could the issue be removed.

Renaming templates

While testing the 'Port-template list' issue, it was noted that renaming templates would include ports from client-types and vendors that were not intended. This behaviour was like thee above port-list issue caused by inconsistently including client-type and/or vendor-type in the updates.

The issue is fixed.

Initial Ldap login

The recent Ldap / AD redevelopment was discovered to contain an issue when a user tried to login using Ldap for the first time. When a new user logs in for the first time, it is still unknown whether to validate as a local user or as an ldap user. But it was precisely this info that was supposed to determine the local or ldap approach. As a consequence no method could be selected and the user was denied access with an 'Unknown error'.

This issue has been fixed.

Stored jobs

The option to save and share command jobs for later use was broken. The tool appeared to save the job information on the from properly, but was found missing when attempting to use it.

The problem was resolved and jobs can now be properly saved and reused.

Custom subnets

When creating or editing 'Custom' Ipv4 or Ipv6 subnets using the front-end, several operations caused some ip-values to become inconsistent. Changing the prefix or net-address is now consistent with the calculated addresses and netmasks.

Vendor-modules

To ensure all our vendor-modules are stable, we retested them for their consistency and predictability in many different areas. All functions were re-aligned with our internal modeling and retested.

The areas most improvements were made are file (config) transfers using the various protocols (ftp, sftp, scp, tftp), Ipv6 support, logging and error-handling and OS-upgrades.

IP edit forms

The IPv4 and IPv6 edit subforms of the Service details were found to exhibit many inconsistencies or incomplete value updates after making changes. The 'inner workings' of these forms were extensively reworked to ensure consistent IP values.

Port-names for slot '0' and '[ ]'

When creating new ports on slot '0' or slot '[ ]', incorrect port-names were assigned. This has been corrected.

MariaDB conversions

Ever since NetYCE 6.x we the Mysql variant 'MariaDB' version 10.0. Later some deployments adapted MariaDB 10.2 which was equally stable.

When switching from MariaDB 10.0 to 10.2 an upgrade procedure had to be observed to convert the table formats. In environments where both database versions are operational and NetYCE database archives need to exchanged, this procedure was not observed and database issues could arise.

Now, when restoring a NetYCE database archive from an older MariaDB is restored (using the GUI), the upgrade conversion procedure is automatically performed. Note that an automatic downgrade procedure is not available. For those cases a manual procedure involving the cli script 'table_renew.pl' is made available.

Issue list

Apart form the problem-tickets and support requests we get from our customers and users, we also maintain an internal 'Issue-list' with bugs, problems and enhancements we experienced while developing or building solutions for customers. Usually the high-impact issues are fixed straight-away, but low-impact issues received lower priority.

Over time this the list of low-priority issues grew to exceed 150. However, for the 7.1.1 release we considered this status inappropriate to the maturity of NetYCE and resolved to eliminate all but the nice-to-haves. The resulting 7.1.1 version therefore incorporates the fixes and enhancements of

about 120 of these issues, only some of which were deemed relevant for these release notes.

Of course, there will be new bugs and issues or lacking options. Please help us in creating a better product and report them to us, the NetYCE development team, by emailing to **`support@netyce.com`**.

From:
https://wiki.netyce.com/ - **Technical documentation**

Permanent link:
https://wiki.netyce.com/doku.php?id=maintenance:releases:7.1.1_20190501

Last update: **2024/07/03 12:31**