CNaaS Tools

A collection of tools to help out when working with CNaaS.

Faster template development

Test out new templates by doing a dry_run on a device without commiting the new template to git by using template_dry_run.py from https://raw.githubusercontent.com/SUNET/cnaas-nms/develop/src/cnaas_nms/tools/template_dry_run.py

You must have python3 with pip modules: requests, jinja2 and yaml installed. Then export ENV variables CNAASURL (URL to CNaaS API) and JWT_AUTH_TOKEN (a valid JWT token), change directory to the root of your templates and run the script like so:

/path/to/script/template_dry_run.py <dryrun_hostname>

Old config and new config for the device based on your un-commited templates will be printed to stdout.

Easier settings editing

Verify settings syntax inside the VScode editor without saving or committing to git by using https://github.com/SUNET/cnaas-nms-vscode

Download CNaaS-NMS VScode plugin (v0.0.3 - 2023-01-02)

After installing the plugin in VScode you have to configure the URL to your API. Go into settings of VScode and search for cnsaas, and enter the base URL

After everything is setup, open a YAML file from your settings repository and then run: Ctrl-Alt-P , enter cnaas syntax check, press enter. A small box will appear in the bottom right corner saying CNaaS syntax check: success/error message.

Better access-control list editing

Generate ACLs for Arista, Cisco, Juniper, IPtables etc from one single definition. Configure networks, groups and services and combine it with policies to generate ACLs that can be pasted into templates.

See Howto capirca

Automatic generation of ssh-key users

Generate template config for ssh-key users based of a a .ssh/authorized_keys file:

usage: ./sunet-arista-sshkeys.py < ~/.ssh/authorized_keys > users_template.j2

Download sunet-arista-sshkeys.py

A collection of tools to help out when working with CNaaS.

- Faster template development
- Easier settings editing
- Better access-control list editing
- Automatic generation of ssh-key users
- Faster template development
- Easier settings editing
- Better access-control list editing
- Automatic generation of ssh-key users

Faster template development

Test out new templates by doing a dry_run on a device without commiting the new template to git by using template_dry_run.py from https://raw.githubusercontent.com/SUNET/cnaas-nms/develop/src/cnaas_nms/tools/template_dry_run.py

You must have python3 with pip modules: requests, jinja2 and yaml installed. Then export ENV variables CNAASURL (URL to CNaaS API) and JWT_AUTH_TOKEN (a valid JWT token), change directory to the root of your templates and run the script like so:

/path/to/script/template_dry_run.py <dryrun_hostname>

Old config and new config for the device based on your un-commited templates will be printed to stdout.

Easier settings editing

Verify settings syntax inside the VScode editor without saving or committing to git by using https://github.com/SUNET/cnaas-nms-vscode

Download CNaaS-NMS VScode plugin (v0.0.2 - 2021-08-17)

After installing the plugin in VScode you have to configure the URL to your API. Go into settings of VScode and search for cnsaas, and enter the base URL

After everything is setup, open a YAML file from your settings repository and then run: Ctrl-Alt-P , enter cnaas syntax check, press enter. A small box will appear in the bottom right corner saying CNaaS syntax check: success/error message.

Better access-control list editing

Generate ACLs for Arista, Cisco, Juniper, IPtables etc from one single definition. Configure networks, groups and services and combine it with policies to generate ACLs that can be pasted into templates.

See Howto capirca

Automatic generation of ssh-key users

Generate template config for ssh-key users based of a a .ssh/authorized_keys file:

usage: ./sunet-arista-sshkeys.py < ~/.ssh/authorized_keys > users_template.j2

Download sunet-arista-sshkeys.py