Node template YML syntax

This document describes the syntax of node template YML files used by EVE-NG. Template files are stored under html/templates/<platform>/ (e.g. templates/intel/ or templates/amd/) and must have the .yml extension. The filename without extension is the **template name** (e.g. docker.yml → template docker).

File format

- **Format:** YAML 1.x.
- **Preamble:** Optional copyright/comment lines at the top. The actual configuration must start with a YAML document; use --- at the beginning of the config block if you have comments above.
- **Encoding:** UTF-8.

Example minimal structure:

```
---
type: docker
description: My Docker template
name: MyNode
# ... rest of keys
...
```

Common keys (all node types)

These keys are read at startup from every .yml file in the template directory and define how the template appears and behaves in the UI and backend.

Key	Type	Description
type	string	**Required.** Node type: docker, qemu, iol, dynamips, vpcs. Determines which emulator/runtime is used and which type-specific keys apply.
description	string	Short description shown in the "Add node" / template list (e.g. "Docker.io", "Arista cEOS").
name	string	Default name prefix for new nodes (e.g. "Docker", "cEOS").
icon	string	Icon filename for the topology canvas (e.g. Misc-2D-Docker-S.svg, Router-2D-Gen-Grey-S.svg).

Key	Type	Description
config_script	string	Config script name (e.g. config_docker.py, embedded). Used when the user applies a startup config to the node.
prep	string	Optional. Name of a **preparation script** run before the node is started (see Scripts prep, init, cstart).
init	string	Optional. Name of an **init script** run after preparation (see Scripts prep, init, cstart).
cstart	string	Optional. **QEMU-only.** Extra QEMU command-line flags used only on the first start when a startup config is applied (see Scripts prep, init, cstart).
cpulimit	number	CPU limit (e.g. 1).
ethernet	number	Default number of Ethernet interfaces.
console	string	Default console type: e.g. rdp, rdp-tls, vnc, telnet.
cpu	number	Default CPU count/limit for the node.
ram	number	Default RAM in MB.

Type-specific keys (e.g. qemu_arch, nvram, dock_args) are documented below per type.

Interface naming (optional)

For node types that support custom interface naming:

Key	Type	Description
eth_format	string	Format string for interface labels, e.g. Ethernet{1}, Gi0/0/0/{0}, eth{0}. Placeholders like {0}, {1} are replaced by interface index.
eth_name	list of strings	Optional list of custom names per interface (one per interface). Can be used together with eth_format; eth_name takes precedence for the given index.

These are supported for docker and other types that use the universal naming logic; templates with hardcoded interface naming may ignore them.

Scripts prep, init, cstart

These three keys control preparation and first-start behaviour. Scripts are looked up under the directory **/opt/unetlab/config_scripts/** on the EVE-NG server. The value is the **script filename** (e.g. prep_xrd.sh), not a path.

prep (preparation script)

- **Type:** string (script filename).
- **When it runs:** During the **prepare** phase, before the node process is started. The node's running directory already exists; TAP interfaces (for non-Docker types) are created before the prep script is called.
- **Invocation:** The script is executed with a single argument: the node's **running directory** (e.g. /opt/unetlab/tmp/0/<lab-uuid>/<node-id>).
- **Purpose:** Prepare files in that directory (e.g. generate env files, build config.iso from startup-config, create virtiob.qcow2 or other images). For Docker nodes on a remote satellite, the content of the node directory is synced to the satellite after prep.
- **Example values (from shipped templates):** prep_xrd.sh, prep_junosvex.sh, prep_isrv.sh, prep_c8000v.sh, prep_cat9kv.sh, prep_cat9kvq200.sh, prep_cat9kvuadp.sh.

Example in template:

```
prep: prep_xrd.sh
```

init (init script)

- **Type:** string (script filename).
- **When it runs:** After the prepare phase. For **Docker** nodes on a **remote satellite**: after the container has been **created** but **not yet started**. For local nodes or non-Docker nodes: after prepare, before the node process is started.
- **Invocation:** The script is called with the node's **running directory** as first argument. For Docker on a remote satellite, a second set of arguments is passed: -s followed by the satellite IP (e.g. -s 172.29.130.2). The script can use this to run Docker commands on the satellite (e.g. docker -H ssh://root@<satellite_ip>).
- **Purpose:** One-time setup inside the node or container (e.g. inject a first-boot config file into the container, create files that the node expects at first boot).
- **Example values (from shipped templates):** init_xrd.sh, init_srlinux-ixr-d3.sh, init_srlinux-ixr-d3l.sh, init_srlinux-ixr-d2l.sh, init_cat9kv.sh, init_cat9kvq200.sh, init_cat9kvuadp.sh.

Example in template:

```
init: init_xrd.sh
```

cstart (custom start flags, QEMU only)

- **Type:** string (literal QEMU command-line flags).
- **When it is used:** Only for nodes of type **qemu**, and only when **both** are true: the node has a **startup config** attached (config ≠ 0), and the node has **not** yet been marked as configured (no .configured file in the node directory). So it applies to the **first** start when a config is being applied.
- **Effect:** The string is **appended** to the QEMU command line (the "flags" passed to the emulator). Typical use: add a CD-ROM with a config ISO (-cdrom config.iso) or a USB stick image so the guest can load the startup config on first boot.

- **Not a script:** Unlike prep and init, cstart is not a script name; it is the exact options passed to QEMU.

Examples in templates:

```
# Attach config as CD-ROM on first start
cstart: -cdrom config.iso
# Attach a virtio block device (e.g. USB stick image) on first start
cstart: -drive if=none,id=stick,format=raw,file=virtiob.qcow2 -device nec-usb-xhci,id=xhci -device usb-storage,bus=xhci.0,drive=stick
```

Docker-specific parameters (template YML)

When type: docker, the following keys in the **template YML** are read and applied as defaults for nodes using this template. They control how the container is created and how the console shell is started. They are the main way to configure "new" docker behaviour (replacing legacy image-name heuristics).

sysbox (integer, optional)

- **Values:** 0 (default) or 1.
- **Effect:** If 1, the container is created with --runtime=sysbox-runc. If 0 or omitted, the default runtime is used.
- **Immutability:** Set from the template only; not changed from the node/lab data after creation.
- **Typical use:** Docker-in-Docker (DinD) or other workloads that need the Sysbox runtime.

Example:

```
type: docker
sysbox: 1
# ...
```

dock_args (string, optional)

- **Effect:** Extra arguments passed to docker create (e.g. --device, --cap-add, --net, -it, --shm-size). They are appended after the common options (e.g. --cpus, -m, --name, -h) and before the image name.
- **Default:** Empty string if omitted.
- **Example:**
  --device /dev/fuse --cap-add=SYS_ADMIN --cap-add=NET_ADMIN -it --net=none --shm-size 1G

Example in template:

```
type: docker
dock_args: --device /dev/fuse --cap-add=SYS_ADMIN --cap-add=NET_ADMIN -it --net=none --shm-size 1G
# ...
```

dock_cmd (string, optional)

- **Effect:** Command and arguments appended **after** the image name in docker create. Used as the container's main process/entrypoint when needed (e.g. /sbin/init with systemd env vars).
- **Default:** Empty string if omitted.
- **Example:**
  /sbin/init systemd.setenv=INTFTYPE=eth systemd.setenv=MGMT_INTF=eth0

Example in template:

```
dock_cmd: /sbin/init systemd.setenv=INTFTYPE=eth systemd.setenv=MGMT_INTF=eth0
```

docker_shell (string, optional)

- **Effect:** Command run inside the container when the user opens the console (e.g. shell or CLI launcher). Replaces the default /bash-static -l.
- **Default:** /bash-static -l if omitted.
- **Example:**
  echo Press Enter to start CLI && read -r && Cli
  or
  pgrep sr_linux 1>/dev/null && echo Press Enter to start CLI && read -r && sr_cli || (echo waiting system && sleep 10)

Example in template:

```
docker_shell: echo Press Enter to start CLI && read -r && Cli
```

Complete docker template example

```
---
type: docker
config_script: config_docker.py
description: Docker.io
name: Docker
cpulimit: 1
icon: Misc-2D-Docker-S.svg
```

```
ethernet: 1
console: rdp
cpu: 1
ram: 1024
dock_args: --device /dev/fuse --cap-add=SYS_ADMIN --cap-add=NET_ADMIN -it --net=none --
shm-size 1G
...
```

Example with optional docker-only keys and interface naming:

```
---
type: docker
config_script: config_ceos.py
description: Arista cEOS
name: cEOS
cpulimit: 1
icon: Router-2D-Gen-Grey-S.svg
ethernet: 4
eth_format: Ethernet{1}
eth_name:
  - Management0
console: telnet
cpu: 0
ram: 4096
disable_offload: 1
mtu: 9500
dock_args: --privileged -v /lib/modules:/lib/modules:ro -i -t -e MGMT_INTF=eth0 -e
container=docker
docker_shell: echo Press Enter to start CLI && read -r && Cli
dock_cmd: /sbin/init systemd.setenv=MGMT_INTF=eth0 systemd.setenv=container=docker
...
```

QEMU-specific parameters (type: qemu)

When type: qemu, the following keys define the QEMU binary, architecture, NIC model, and extra options. Users can override them per node; the template provides defaults.

qemu_arch (architecture)

- **Type:** string.
- **Meaning:** Target CPU architecture for the QEMU system emulator (qemu-system-<arch>).
- **Allowed values:**

Value	Description
i386	32-bit x86.
x86_64	64-bit x86 (most common).

- **Default (if omitted):** No default in template; must be set in template or per node. Common template default: x86_64.

qemu_version (QEMU version)

- **Type:** string.
- **Meaning:** Which QEMU version to use. The system looks for the binary under /opt/qemu-<version>/bin/qemu-system-<arch> or, if version is empty, /opt/qemu/bin/qemu-system-<arch>.
- **Allowed values (versions known to the UI/API):**

Value
1.3.1
2.0.2
2.2.0
2.4.0
2.5.0
2.6.2
2.12.0
3.1.0
4.1.0
5.2.0
6.0.0
7.2.9
8.2.1
9.2.2

- **Default (if omitted):** 2.4.0 when no value is set.

qemu_nic (NIC model)

- **Type:** string.
- **Meaning:** The QEMU network device model used for the node's Ethernet interfaces (e.g. -device <model>,netdev=...). Affects compatibility and performance inside the guest.
- **Allowed values:**

Value	Description / typical use
virtio-net-pci	Virtio paravirtualized NIC; good performance, supported by most Linux and many other guests.
e1000	Intel E1000 emulation; widely compatible (default when not set).
e1000-82545em	Intel 82545EM; alternative E1000 variant.
i82559er	Intel i82559ER; older Intel NIC.
rtl8139	Realtek 8139; legacy compatibility.
vmxnet3	VMware vmxnet3; useful for VMware-style images.

- **Default (if omitted):** e1000.

qemu_options (extra QEMU options)

- **Type:** string.
- **Meaning:** Extra arguments passed to the QEMU command line (e.g. machine type, CPU, KVM options, serial, boot order). Appended to the generated QEMU invocation.
- **Example:**
  -machine type=pc,accel=kvm -vga std -usbdevice tablet -boot order=cd -cpu host

shutdown (optional)

- **Type:** number (0 or 1).
- **Meaning:** Shutdown mode (e.g. poweroff vs. ACPI shutdown). Used by the template; exact behaviour is runtime-dependent.

Other node types (brief)

- **IOL** (type: iol): Uses e.g. nvram, serial, keepalive from the template.
- **Dynamips** (type: dynamips): Uses dynamips_options (string) for extra dynamips command-line flags.
- **VPCS** (type: vpcs): Uses the common keys only; no extra template keys documented here.