Container

Howto

Prepping the base configuration

```
{
  /system logging action
  :foreach actionName in={ "ContainerLog"; "ContainerDebug" } do={
    :if ([:len [find where name="$actionName"]] = 0) do={
      add name=$actionName target=memory
      :put "[ADDED] $actionName action added"
    } else={ :put "[INFO] $actionName action already exists" }
  }

  /system logging
  :if ([:len [find where topics~"container"]] > 0) do={
    :put "[WARNING] Some logs dealing with the container topic already exist:"
    print where topics~"container"
  }

  /system logging
  :if ([:len [find where action="ContainerLog" and (topics~"container" and topics~"!debug")]]
= 0) do={
    add action=ContainerLog topics=container,!debug
    :put "[ADDED] ContainerLog logging added"
  } else={ :put "[INFO] ContainerLog=container,!debug logging already exists" }

  /system logging
  :if ([:len [find where action="ContainerDebug" and (topics~"container" and topics~"debug")]]
= 0) do={
    add action=ContainerDebug topics=container,debug
```

```
    :put "[ADDED] ContainerDebug logging added"
  } else={ :put "[INFO] ContainerDebug=container,debug logging already exists" }

}
```

```
/container config set registry-url=https://registry-1.docker.io
/container config set tmpdir=/z-pod/tmp

/container config set ram-high=500.0MiB
/container config set memory-high=500.0MiB
```

```
/interface veth
add address=100.122.31.11/24 gateway=100.122.31.1 gateway6="" name=veth1
add address=100.122.31.12/24 gateway=100.122.31.1 gateway6="" name=veth2
add address=100.122.31.13/24 gateway=100.122.31.1 gateway6="" name=veth3
add address=100.122.31.14/24 gateway=100.122.31.1 gateway6="" name=veth4
add address=100.122.31.15/24 gateway=100.122.31.1 gateway6="" name=veth5
add address=100.122.31.16/24 gateway=100.122.31.1 gateway6="" name=veth6
add address=100.122.31.17/24 gateway=100.122.31.1 gateway6="" name=veth7
add address=100.122.31.18/24 gateway=100.122.31.1 gateway6="" name=veth8
add address=100.122.31.19/24 gateway=100.122.31.1 gateway6="" name=veth9

/interface bridge
add comment="container lan - docker1" name=bridge-docker1

/ip address
add address=100.122.31.1/24 interface=bridge-docker1 network=100.122.31.0

/interface bridge port
add bridge=bridge-docker1 interface=veth1
add bridge=bridge-docker1 interface=veth2
add bridge=bridge-docker1 interface=veth3
add bridge=bridge-docker1 interface=veth4
add bridge=bridge-docker1 interface=veth5
add bridge=bridge-docker1 interface=veth6
add bridge=bridge-docker1 interface=veth7
add bridge=bridge-docker1 interface=veth8
add bridge=bridge-docker1 interface=veth9

/ip firewall nat
```

```
add action=masquerade chain=srcnat comment="wan - srcnat traffic from container network to the Internet" dst-address-list=!private-ip-subnets ipsec-policy=out,none out-interface=ether1 src-address=100.122.31.0/24
```

Alpine

To install a plain alpine:latest container on a Mikrotik RouterOS device, use the following:

```
remote-image=library/alpine:latest
```

```
/container
add name=alpine-1 \
  remote-image=library/alpine \
  interface=veth-alpine-1 \
  logging=yes \
  root-dir=sata1/c/alpine-diag-1 \
  cmd="tail -f /dev/null"
```

bluecrow76/alpine-netdiag

It's just the above but with updates and nmap installed.

```
/container
add name=alpine-1 \
  remote-image=bluecrow76/alpine-netdiag \
  interface=veth-alpine-netdiag-1 \
  logging=yes \
  root-dir=sata1/c/alpine-netdiag-1 \
  cmd="tail -f /dev/null"
```

Uptime Kuma

```
# mounts - pick the one you need, or edit to fit
```

```
# /
/container mounts add dst=/app/data name=uptime-kuma_app-data src=/c-mnt/uptime-kuma/app-data
/container mounts add dst=/app/data name=uptime-kuma_app-data-certs src=/c-mnt/uptime-kuma/app-data-certs

# /nvme1
/container mounts add dst=/app/data name=uptime-kuma_app-data src=/nvme1/c-mnt/uptime-kuma/app-data
/container mounts add dst=/app/data name=uptime-kuma_app-data-certs src=/nvme1/c-mnt/uptime-kuma/app-data-certs

# /pcie1
/container mounts add dst=/app/data name=uptime-kuma_app-data src=/pcie1/c-mnt/uptime-kuma/app-data
/container mounts add dst=/app/data name=uptime-kuma_app-data-certs src=/pcie1/c-mnt/uptime-kuma/app-data-certs

# /sata1
/container mounts add dst=/app/data name=uptime-kuma_app-data src=/sata1/c-mnt/upme-kuma/app-datat
/container mounts add dst=/app/data name=uptime-kuma_app-data-certs src=/sata1/c-mnt/upme-kuma/app-data-certs
```

```
# environment lists
/container envs
add key=NODE_EXTRA_CA_CERTS name=uptime-kuma-1 value=/app/data/certs/ca_bundle.pem
```

```
# build the actual container - use the one that suites or edit to fit

# onboard flash
{
:local image louislam/uptime-kuma:1
:local iface veth1
:local rootdir "/c/uptime-kuma-1"
/container/add name=uptime-kuma-1 remote-image=$image interface=$iface root-dir=$rootdir envlist=uptime-kuma-1 mounts=uptime-kuma_app-data,uptime-kuma_app-data-certs
}

# nvme1
{
```

```
:local image louislam/uptime-kuma:1
:local iface veth1
:local rootdir "/nvme1/c/uptime-kuma-1"
/container/add name=uptime-kuma-1 remote-image=$image interface=$iface root-dir=$rootdir
envlist=uptime-kuma-1 mounts=uptime-kuma_app-data,uptime-kuma_app-data-certs
}

# pcie1
{
:local image louislam/uptime-kuma:1
:local iface veth1
:local rootdir "/pcie1/c/uptime-kuma-1"
/container/add name=uptime-kuma-1 remote-image=$image interface=$iface root-dir=$rootdir
envlist=uptime-kuma-1 mounts=uptime-kuma_app-data,uptime-kuma_app-data-certs
}

# sata1
{
:local image louislam/uptime-kuma:1
:local iface veth1
:local rootdir "/sata1/c/uptime-kuma-1"
/container/add name=uptime-kuma-1 remote-image=$image interface=$iface root-dir=$rootdir
envlist=uptime-kuma-1 mounts=uptime-kuma_app-data,uptime-kuma_app-data-certs
}
```

-end

Revision #10
Created 6 September 2025 02:51:15 by bluecrow76
Updated 13 January 2026 22:02:59 by bluecrow76