nShield Container Option Pack

nCOP v1.1.2 User Guide

09 April 2024

Table of Contents

1. Introduction

The nShield Container Option Pack (nCOP) provides application developers, within a container-based environment, the ability to access the cryptographic functionality of an nShield Connect HSM. This release of nCOP has been tested with Docker containers.

The nShield Container Option Pack is installed on top of your existing Security World Software installation, allowing you to continue using your existing Security World and keys.

2. Software Prerequisites

The nShield Container Option Pack requires nShield Security World Software and Docker to be installed prior to the use of the nShield Container Option Pack scripts.

We have successfully tested the following configurations:

nShield HSM	Security World Software Version	nCOP Version	Container Product
Connect+ Connect XC nShield 5c	v12.60 v12.71 v12.81 v13.3 v13.4	1.1.2	Docker Engine 20

When you are using `podman` on RedHat Enterprise Linux, you should install `podman-docker` to provide the Docker alias.nCOP has been tested with `podman-docker 4.0`

Before you can begin using nCOP you must complete the following steps:

1. Set up the HSM(s). For instructions, see the Installation Guide of your HSM(s).
2. Using its IP address or soft Kneti from v12.80, configure your container host machine as a client of the HSM(s). The container host machine is the machine on which you will run the nShield hardserver and application containers.
3. To access and use cryptographic keys from a Security World, load or create a Security World on the HSM, and map the key management data folder (`kmdata`) from your container host machine into the running application containers.

For further information on configuring and managing nShield HSMs, Security Worlds, and Remote File Systems, see the User Guide supplied with your Security World software

3. Installation

To install the nShield Container Option Pack:

1. Create the directory where you wish to install nCOP. For example, `mkdir -p /opt/ncop`.
2. Untar the option pack to this directory: `tar xf ncop-1.1.2.tar -C /opt/ncop`

You should ensure that any users that will use the nShield Container Option Pack scripts have permission to execute the installed scripts and run Docker.

The following Bash scripts are provided in this Option Pack.

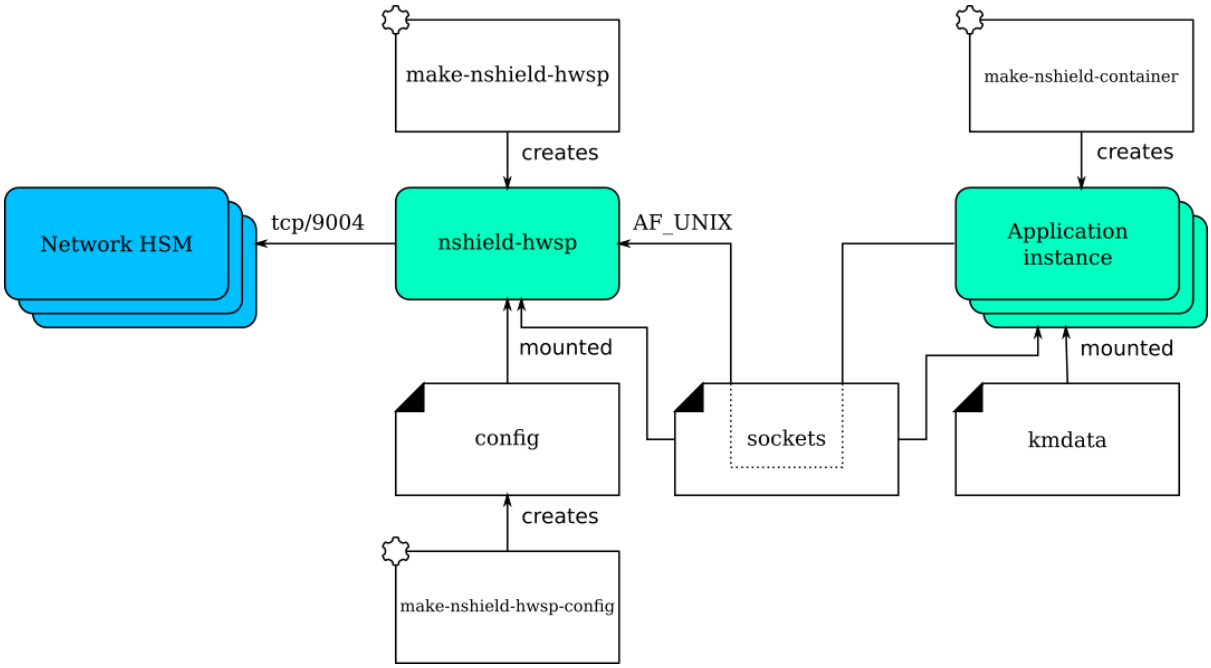
Script	Purpose
make-nshield-hwsp	Make an nShield hardserver Docker image
make-nshield-hwsp-config	Generates a hardserver configuration file for an nshield-hwsp container
make-nshield-application	Create a new Docker image with the nShield support software installed
extend-nshield-application	Install the nShield support software to an existing Docker image

3.1. Uninstalling the nShield Container Option Pack

To uninstall, delete the directory containing the nShield Container Option Pack from your system.

You should remove any built Docker images and/or containers from your system if they are no longer needed. Consult the documentation for Docker on how to delete Docker images and containers.

4. Deployment Architecture



The `nshield-hwsp` container runs the hardserver. It is supplied with configuration to connect to one or more network HSMs (nShield Connects). It exposes the hardserver via an `AF_UNIX` socket.

Access to the hardserver socket must be restricted to trusted users.

Application instances are any containers that include applications that use the nShield software stack. The applications are supplied with the socket used to connect to the hardserver and access to the security World key management data files and associated cryptographic keys.

The key management data files, including encrypted copies of keys, are located in `kmdata`. A container mounting `kmdata` as a volume will be able to spoof the nShield Connect client. Therefore, access to files in `kmdata` must be controlled and restricted to trusted users.

5. The Hardserver Container

The hardserver container `nshield-hwsp` controls communication between the configured nShield Connect/s and application containers. Only one hardserver container is required per deployment, regardless of the number of nShield Connects or application containers.

5.1. Creating `nshield-hwsp`

Create the hardserver container using `make-nshield-hwsp`. The only required argument is the path to a mounted nShield Security World ISO.

For example:

```
$ mkdir SecWorld-12.70.4
$ sudo mount -o loop SecWorld_Lin64-12.70.4.iso SecWorld-12.70.4
mount: /dev/loop0 is write-protected, mounting read-only
$ make-nshield-hwsp SecWorld-12.70.4
[...]
Successfully tagged nshield-hwsp:12.70.4
```

The default base image for nShield hardserver containers is RedHat UBI7. The default tag reflects the version of nShield Security World software that the container was built from.

If you want to use a different base image, or specify a different tag, use the `--from` and `--tag` options. See `make-nshield-hwsp --help` for more information.

5.1.1. Users and Groups

By default the `nfast` user and group in the container will match those on the host machine. If the `nfast` user and group do not exist on the host, either:

- create the `nfast` user and group on the host, or
- if this is a bad fit for deployment, use the `--uid` and `--gid` options to set them instead.

5.2. Configuring `nshield-hwsp`

Create the hardserver container's configuration using `make-nshield-hwsp-config`. The hardserver container's configuration is the "config" component in the

Deployment Architecture diagram.

- Use the `--output` option to specify the filename.
- List IP addresses of network HSMs on the command line.

Different configuration files can be used for different container deployments.

Running the following example requires the nShield support software installed on the host. You can also create the config file based on the template below by filling in the `esn/ip/port/keyhash` values in the `nethsm_imports` section.

For example:

```
$ sudo mkdir -p /opt/ncop/config1
$ sudo make-nshield-hwsp-config --output /opt/ncop/config1/config 192.168.0.10
$ cat /opt/ncop/config1/config

[nethsm_imports]
local_module=1
remote_esn=1111-2222-3333
remote_ip=192.168.0.10
remote_port=9004
keyhash=000102030405060708090a0b0c0d0e0f10111213
privileged=0
```

Module numbers are assigned in order.

Note that key hash values are retrieved from remote HSMs without any trust. The generated configuration file should be checked against values recorded from the front panel, or some other trusted path.

5.3. Running `nshield-hwsp`

To run the hardserver container, you must:

1. Supply the generated hardserver configuration to the container.
2. Mount a volume for the /opt/nfast/sockets folder.
3. Mount a volume for the /opt/nfast/sockets-priv folder, if required.

This can be done with the `-v` option.

For example, using a Docker volume for the /opt/nfast/sockets folder:

```
$ docker volume create socket1
$ docker run \
    -v /opt/ncop/config1:/opt/nfast/kmdata/config:ro \
    -v socket1:/opt/nfast/sockets \
    nshield-hwsp:12.70.4
Hardserver INIT: Notice: Hardserver using priority class queueing algorithm: 0 classes and 0 modules total.
```

```
[...]
```

This makes the hardserver in `nshield-hwsp` available via the sockets in the Docker volume socket1. If the nShield support software is installed, this can be tested from the host:

First obtain the mount point for the Docker volume and use this for the NFAST_SERVER environment variable:

```
$ docker volume inspect --format '{{ .Mountpoint }}' socket1
/var/lib/docker/volumes/socket1/_data

$ NFAST_SERVER=/var/lib/docker/volumes/socket1/_data/nserver /opt/nfast/bin/enquiry -m0
Server:
    enquiry reply flags  none
    enquiry reply level  Six
    serial number        1111-2222-3333
    mode                 operational
    version              12.70.4
[...]
```

6. Application Containers

An nShield application container is a container with the nShield Security World software installed.

Two strategies for creating nShield application containers are supported:

- Create an nShield base container, and derive application containers from it.
- Derive a container with nShield Security World software from an existing application container.

6.1. nShield base container

The base container can be created using `make-nshield-application`. The only required argument is the path to a mounted Security World ISO.

```
$ make-nshield-application SecWorld-12.70.4
[...]
Successfully tagged nshield-ubi7:12.70.4
```

To run the base application container, you must:

- Supply a kmdata folder (if you wish to perform operations that require a Security World).
- Mount a volume for the sockets folder.

Both can be done with the -v option. Different application containers can use different kmdata folders. For example, you could create a new folder:

```
$ mkdir -p /opt/ncop/app1/kmdata/local
```

You can then copy the desired Security world and module files for your application into this folder.

Using this folder and the Docker volume created for the hardserver container in section 5.3 above, this container can be run directly:

```
$ docker run -it \
    -v /opt/ncop/app1/kmdata:/opt/nfast/kmdata:ro \
    -v socket1.hwsp:/opt/nfast/sockets \
    nshield-ubi7:12.70.4
[root@075c41761e0f /]# /opt/nfast/bin/enquiry
Server:
    enquiry reply flags  none
    enquiry reply level  Six
    serial number        1111-2222-3333
```

```
[...]
```

It can also be used as the base for an application container. (See examples/nfkminfo.)

The default base image for nShield application containers is RedHat UBI7. The default tag reflects the version of nShield Security World software that the container was built from.

If you want to use a different base image, or specify a different tag, use the `--from` and `--tag` options. See `make-nshield-application --help` for more information. nShield application containers have been tested with the following base images:

- RedHat UBI 7/8 (including "minimal")
- CentOS 7
- Ubuntu Bionic 18.04
- Ubuntu Focal 20.04
- Debian Stretch 9 (including slim)
- Debian Buster 10 (including slim)
- OpenSUSE 15.1/2
- Alpine (frolvlad-glibc)
- Nginx 1.18
- Apache 2.4.43

Other base images may work but are untested.

6.1.1. API Support (Java)

Depending on the application's requirements, the additional Java component may be installed with option `--java`.

```
$ make-nshield-application --java SecWorld-12.70.4
[...]
Successfully tagged nshield-ubi7:12.70.4-java
```

The supported API is appended to the nShield software version in the container tag.

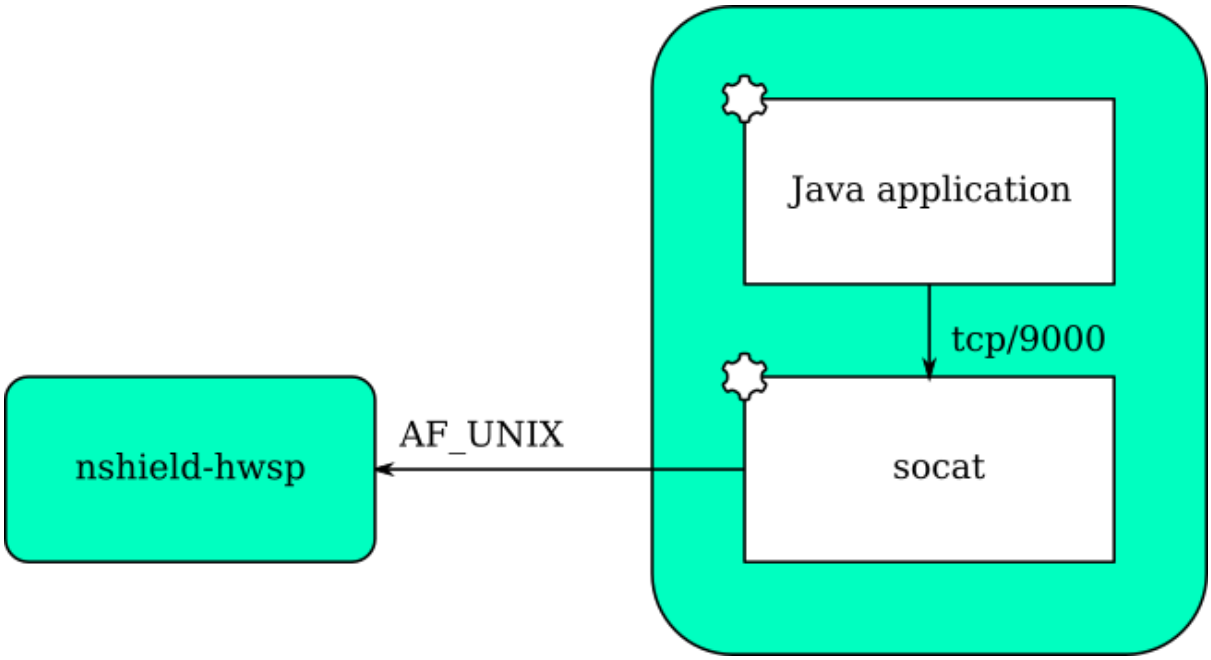
Note that with Security World v12.60 and later:

- PKCS11 is included by default and is not configurable with the nCOP scripts.

- CHIL is not supported.

Java applications expect to connect to localhost:9000. This must therefore be forwarded to the hardserver socket using the `/opt/nfast/sbin/nshield-forward` utility.

- This utility is included in any container built using the `make-nshield-application` or `extend-nshield-application` scripts.
- This utility depends on `socat` being installed in the application container. The nShield base container includes `socat` but `extend-nshield-application` will not install it. You must install it yourself.



6.2. Deriving from application containers

The alternative strategy is to build the application container and then install the nShield support software into it. This can be done with `extend-nshield-application`. This strategy might be preferred if the application container already exists or if it supports many cryptographic backends with nShield being just one choice.

See examples/nfkmverify for an example use of `extend-nshield-application`.

6.3. Example applications

A set of example application containers are provided within the examples directory.

For further information on building and running the examples, see the Readme within each example directory.

Example	Description
`nfkminfo`	Simple example of running an nShield application in an application container created using `make-nshield-application`
`javaenquiry`	Example Java application where the the application container is derived from the nShield Java container using `make-nshield-application`
`nfkmverify`	Example where the application container is extended from an existing container to add nShield Container using `extend-nshield-application`
`nfweb`	Example web server exposing basic information about the connected nShield modules