nShield Container Option Pack

nCOP v1.1.1 User Guide

09 April 2024

© 2025 Entrust Corporation. All rights reserved.

Table of Contents

1. Introduction

The nShield Container Option Pack (nSCOP) provides application developers, within a container-based environment, the ability to access the cryptographic functionality of an nShield Connect HSM. This release of nSCOP 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. nSCOP has been tested with Security World v12.40 and v12.60, and Docker Engine 19.03.

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

1. Install the nShield Security World software on the client host machine.
2. Configure a Remote File System (RFS) for the nShield Connects that you wish to use nSCOP with.
3. Set up the nShield Connect/s to have the IP address of your client host machine config ured as a client of the nShield Connect.

The client host machine is the machine on which you will run the nShield hardserver and application containers.

To access and use cryptographic keys from within a Security World you will need to load or create a Security World on the nShield Connect and map the key management data folder (kmdata) from your client host machine into the running application containers.

For further information on configuring and managing nShield Connects, Security Worlds and Remote File Systems, please consult the nShield Connect 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 nSCOP. For example:

```
mkdir -p /opt/nfast/nscop
```

2. Untar the option pack to this directory. For example:

```
tar xf nscop-1.0.0.tar -C /opt/nfast/nscop
```

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.

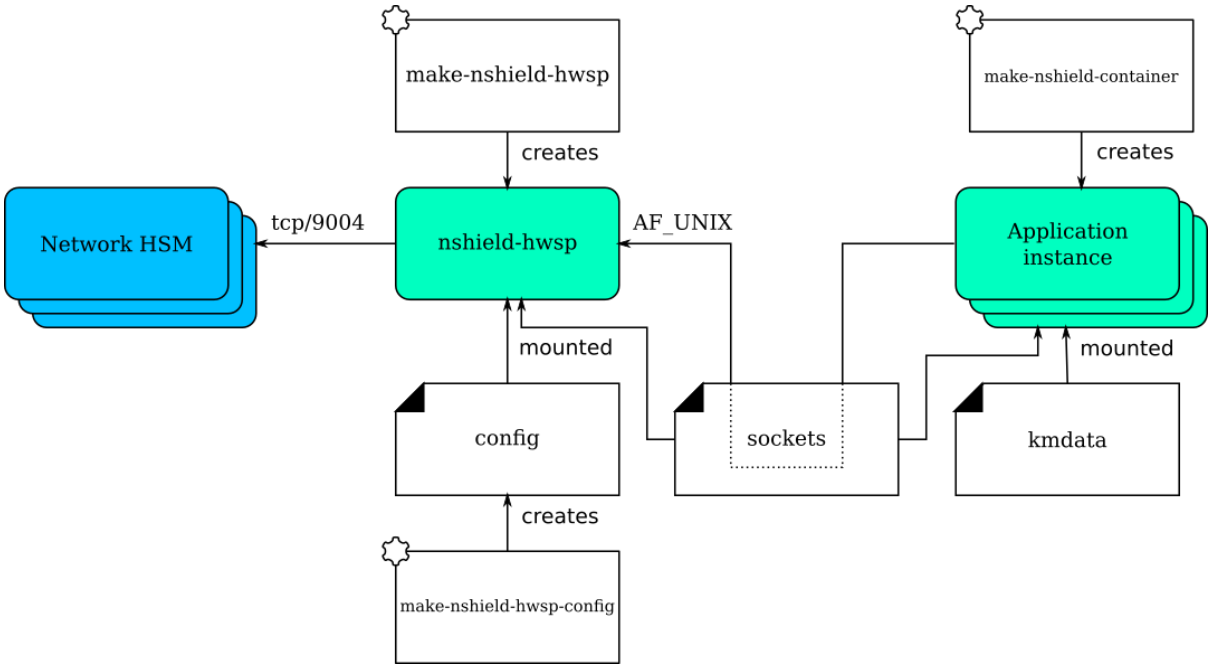
Script	Purpose
make-nshield-hwsp	Make an nShield hardserver Docker image
make-nshield-hwsp-config	Generates a hardserver configuration file for an nshield-hwsp container
make-nshield-application	Create a new Docker image with the nShield support software installed
extend-nshield-application	Install the 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. They are supplied with the socket used to connect to the hardserver and access to the key management data files (to use the World 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`

`make-nshield-hwsp` can be used to create the hardserver container. The only required argument is the path to a mounted nShield Security World ISO.

For example:

```
$ mkdir SecWorld-12.60.3
$ sudo mount -o loop SecWorld_Lin64-12.60.3.iso SecWorld-12.60.3 mount: /dev/loop0 is write-protected, mounting
read-only
$ make-nshield-hwsp SecWorld-12.60.3
[...]
Successfully tagged nshield-hwsp:12.60.3
```

The default base image for nShield hardserver containers is `ubuntu:bionic`. 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.

5.2. Users and Groups

By default the `nfast` user and group in the container will match those on the host machine. If they do not exist on the host, or if this is a bad fit for deployment, the `--uid` and `--gid` options should be used to set them.

5.3. Configuring `nshield-hwsp`

`make-nshield-hwsp-config` can be used to create the hardserver container's configuration (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.

For example:

```
$ sudo mkdir -p /opt/nfast/kmdata/config.hwsp
$ sudo make-nshield-hwsp-config --output /opt/nfast/kmdata/config.hwsp/config 192.168.0.10
$ cat /opt/nfast/kmdata/config.hwsp/config syntax-version=1

[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 compared against values recorded from the front panel, or some other trusted path.

5.4. Running `nshield-hwsp`

To run the hardserver container, you must:

- Supply the generated hardserver configuration to the container.
- Expose the container's client socket.

Both can be done with the `-v` option. For example:

```
$ sudo mkdir -m755 -p /opt/nfast/sockets.hwsp
$ sudo chown -R nfast:nfast /opt/nfast/sockets.hwsp
$ docker run \
    -v /opt/nfast/kmdata/config.hwsp:/opt/nfast/kmdata/config:ro \
    -v /opt/nfast/sockets.hwsp:/opt/nfast/sockets \
    nshield-hwsp:12.60.3
Hardserver INIT: Notice: Hardserver using priority class queueing algorithm: 0 classes and 0 modules total.
[...]
```

This makes `nshield-hwsp`'s hardserver available via the sockets in `/opt/nfast/sockets.hwsp`. You can test this from the host:

```
$ NFAST_SERVER=/opt/nfast/sockets.hwsp/nserver enquiry -m0
Server:
 enquiry reply flags  none
 enquiry reply level  Six
 serial number        1111-2222-3333
 mode                 operational
 version              12.60.3
[...]
```

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-12.60.3
[...]
Successfully tagged nshield-centos7:12.60.3
```

This container can be run directly:

```
$ docker run -it \
    -v /opt/nfast/kmdata:/opt/nfast/kmdata:ro \
    -v /opt/nfast/sockets.hwsp:/opt/nfast/sockets \
    nshield-centos7:12.60.3
[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` in Example applications.

The default base image for nShield application containers is `centos:centos7`. 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

- CentOS 7
- Ubuntu Bionic
- frolvlad/alpine-glibc
- Debian Stretch
- nginx 1.17.4
- Apache 2.4.41

Other base images may work but are untested.

6.2. API Support (CHIL, Java, PKCS#11)

Depending on the application's requirements, additional components may be installed. The possible options are `--chil`, `--java` and `--pkcs11`. Any combination is allowed.

```
$ make-nshield-application --java SecWorld-12.60.3
[...]
Successfully tagged nshield-centos7:12.60.3-java
```

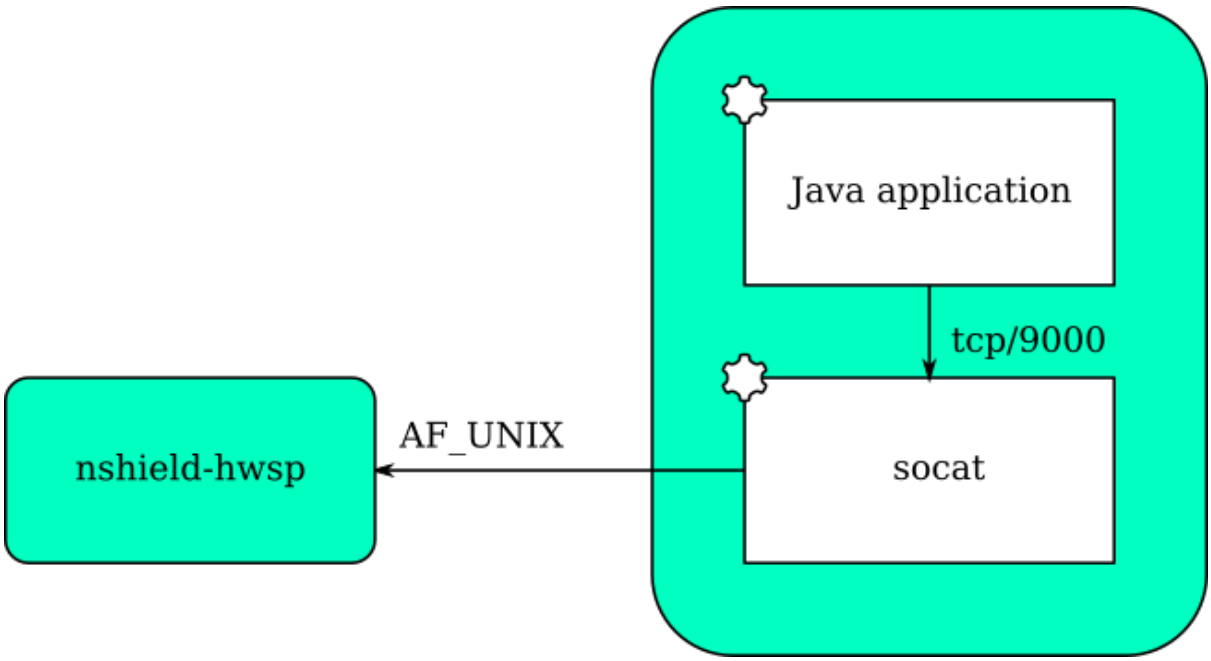
The set of supported APIs is appended to the nShield software version in the container tag.

API support is dependant on the version of nShield software used. Note that with Security World v12.60, PKCS11 is included by default and is not configurable with the nSCOP scripts, and CHIL is not supported. With earlier versions you can configure any combination of APIs.

6.3. Java applications

Java applications expect to connect to localhost:9000. This must therefore be forwarded to the hardserver socket.

`/opt/nfast/sbin/nshield-forward` implements this forwarding and 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.



6.4. Deriving from application containers

The alternative strategy is to build the application container and then install the nShield sup port 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 crypto-graphic backends with nShield being just one choice.

For an example use of `extend-nshield-application`, see `examples/nfkmverify` in Example applications.

6.5. Example applications

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

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

Directory	Description
`examples/nfkminfo`	Simple example of running an nShield application in an application con-tainer created using `make-nshield-application`.
`examples/javaenquiry`	Example Java application where the application container is derived from the nShield Java container using `make-nshield-application`.
`examples/nfkmverify`	Example where the application container is extended from an existing container to add nShield Container using `extend-nshield-application`.

Directory	Description
`examples/nfweb`	Example web server exposing basic information about the connected nShield modules.