KeySafe 5

KeySafe 5 v1.1 Quick Start Guide

21 September 2023

Table of Contents

1. Quick Start Guide

1.1. Overview and prerequisites

The following steps provide a quick-start guide to installing Keysafe 5 and its dependencies using the provided deploy script.

The script is designed to be run on UNIX/Linux based systems by a non-root user. The script may call `sudo` as required.

These steps install Keysafe 5 and its dependencies. The included deploy script (`deploy.sh`) will provide a substitute installation for the various dependencies but they are only suitable for evaluation purposes, and should *not* be used for production environments.

Please see Hardening The Deployment for steps to harden the deployment. Entrust recommends these steps as a minimum and that additional hardening may be required dependent on your own requirements.

A production deployment will have as a minimum the following:

- Maintained and patched versions of all the dependencies
- A secure CA with TLS v1.3 support for certificates. The deploy script can provide a local insecure CA.
- A secure Kubernetes installation. The deploy script can install K3s locally.
- A secure MongoDB database. The deploy script can provide a replicated MongoDB with X.509 authentication running in Kubernetes.
- A secure RabbitMQ message broker. The deploy script can provide a RabbitMQ with X.509 authentication running in Kubernetes.
- HTTPS secured by a trusted certificate for the Keysafe 5 endpoints. The deploy script will enable HTTPS connections with a self-signed insecure certificate.
- Require authentication to access Keysafe 5. OIDC & OAUTH2 are currently supported in Keysafe 5. The deploy script will not set up authenticated access.

This release includes Docker images that need to be pushed to a Docker registry. If you have a private registry you may push the images from a different machine.

1.2. Hardware Requirements

Entrust recommends the following hardware specification to ensure smooth running of Keysafe 5.

- CPU: 2 minimum (4 recommended)
- RAM: 8GB minimum
- Disk Storage: 64GB minimum
    - For optimal performance Entrust recommends use of an SSD.

Requirements will vary depending on the size of the nShield estate being managed and if services, such as MongoDB and RabbitMQ, are being hosted on the same machine as the Kubernetes cluster or externally.

1.3. Unpack the release

```
$ mkdir keysafe5-install
$ tar -xf nshield-keysafe5-1.1.1.tar.gz -C keysafe5-install
$ cd keysafe5-install
```

1.4. Existing infrastructure

This section describes the interaction with infrastructure that you may like to use when installing Keysafe 5 via the deployment script. Otherwise skip to Authentication.

1.4.1. Docker images

If you have a private registry you may push the images to it like so:

```
# Load the Docker images to your local Docker
$ docker load < docker-images/hsm-mgmt.tar
$ docker load < docker-images/sw-mgmt.tar
$ docker load < docker-images/ui.tar

# We need to use a private registry in many places
$ export DOCKER_REGISTRY=private.registry.local
```

```
# Tag the Docker images for a private registry
$ docker tag hsm-mgmt:1.1.1 $DOCKER_REGISTRY/keysafe5/hsm-mgmt:1.1.1
$ docker tag sw-mgmt:1.1.1 $DOCKER_REGISTRY/keysafe5/sw-mgmt:1.1.1
$ docker tag mgmt-ui:1.1.1 $DOCKER_REGISTRY/keysafe5/mgmt-ui:1.1.1

# Log in to ensure pushes succeed
$ docker login $DOCKER_REGISTRY

# And push
$ docker push $DOCKER_REGISTRY/keysafe5/hsm-mgmt:1.1.1
$ docker push $DOCKER_REGISTRY/keysafe5/sw-mgmt:1.1.1
$ docker push $DOCKER_REGISTRY/keysafe5/mgmt-ui:1.1.1
```

By setting the `DOCKER_REGISTRY` environment variable the deploy script will pull images from that registry. Otherwise the deploy script will set up a local insecure Docker registry. In this case, ensure that Docker is installed.

1.4.2. Kubernetes

If you have a Kubernetes cluster available, ensure that `kubectl` points to it, and that `kubectl get pods -A` returns a list of pods. Otherwise the deploy script will install K3s locally to `/usr/local/bin` and create a `~/.kube/config` to point to it. If you want to use the K3s installed by the script, the environment variable `KUBECONFIG` will need to be set to `~/.kube/config`. For this setting to persist it needs to be added to your shell's configuration file.

1.4.3. Istio

If you have Istio installed, ensure that `istioctl` is on your path. Otherwise the deploy script will download a local copy of `istioctl`, and install Istio as required.

1.4.4. RabbitMQ

Entrust recommends that you use your standard secure RabbitMQ installation, along with your policies for authentication and virtual hosts on your production system; this is only a demo system.

If you have an existing RabbitMQ environment:

- Set the environment variable `RABBIT_URL` to point to the RabbitMQ server, port, and vhost like: `rabbitmq.example.com:5671/keysafe5`.
- You will also need to create a Kubernetes generic secret in the `nshieldkeysafe5` namespace with `ca.crt`, `tls.crt`, and `tls.key` for a user that has full access to the vhost. Set `RABBIT_SECRETS` to the name of this generic secret.

- If you set the environment variable `AGENT_USER` to a value, and create another set of credentials `ca.crt`, `$AGENT_USER.crt` and `$AGENT_USER.key` then the deploy script will create an `agent-config.tar.gz` with a configuration for deploying an agent.

If you do not have an existing RabbitMQ server, the deploy script will set one up on the Kubernetes cluster along with the secrets for both the server and agents. By default, neither RabbitMQ's management interface nor peer discovery interface for Kubernetes will be enabled. Should either of these be required, set the environment variable `RABBITMQ_PLUGINS`. To enable both, set it to `"rabbitmq_management rabbitmq_peer_discovery_k8s"`. To enable only one, set it to the appropriate value. For example, to enable just the management interface, set it to `"rabbitmq_management"`.

1.4.5. MongoDB

Entrust recommends that you use your standard secure MongoDB Replica Set installation.

If you have an existing MongoDB deployment:

- Set the environment variable `MONGODB` to a backslash-comma separated list of servers, along with their port numbers in the form: `mongo-1.example.com:27017\,mongo-2.example.com:27017` The backslash should be visible when running `echo $MONGODB`. A quick tip: using single-quotes `'` will prevent the bash command line acting on the backslash you have typed.
- You will also need to create a Kubernetes generic secret in the `nshieldkeysafe5` namespace with `ca.crt`, `tls.crt`, and `tls.key` for a user that has readWrite roles on the databases: `hsm-mgmt-db` and `sw-mgmt-db`. Set `MONGO_SECRETS` to the name of this generic secret.

If you do not have an existing MongoDB deployment, the deploy script will set one up on the Kubernetes cluster along with the secrets for both the server and backend services.

1.5. Authentication

To disable OIDC authentication, set the environment variable `DISABLE_AUTHENTICATION` to `yes`, and you may move on to Install Keysafe 5.

To configure authentication for Istio, the environment variable `AUTH_ISSUER_URL`

needs to point at the issuer URL. Additionally, either `AUTH_JWKS` (for the payload) or `AUTH_JWKS_URL` (for the URL) also needs to be set. `AUTH_AUDIENCES` should be a comma-delimited list. The deploy script will automatically add the fully qualified domain name for the host to this list if not already present.

For UI authentication the deploy script requires an `OIDCProviders.json` file. Its location should be set in the environment variable `OIDC_PROVIDERS_FILE_LOCATION`.

Further details on configuring authentication for Keysafe 5 can be found in *Helm Chart Installation*.

1.6. Install Keysafe 5

It is now possible to run the deploy script.

Do not run the deploy script under `sudo`. If `sudo` permissions are required it will be called by the script and you will be prompted for your credentials.

The deploy script must be run from inside the directory to which it is extracted. Running with the `-n` flag will perform a set of pre-flight checks and show what will happen then exit without taking any action.

```
$ ./deploy.sh -n
```

To disable authentication set the environment variable `DISABLE_AUTHENTICATION` to `yes`. Otherwise you may follow the instructions in the Authentication section.

You may now perform the deployment with the `-y` flag.

```
$ ./deploy.sh -y
```

The script will take a few minutes to run, showing what actions are taking place. You may be prompted for your password by `sudo`, for example when installing K3s.

1.7. Configure nShield client machines

To configure a host machine to be managed and monitored by this deployment, install the Keysafe 5 agent on the nShield client machine containing the relevant Security World or HSMs.

Ensure no firewall rules are blocking the AMQP port communication between the machine exposing the AMQP port from Kubernetes and the machine running the agent.

The deploy script will output a tar archive called `agent-config.tar.gz` that contains the agent configuration file and TLS certificates for authentication to RabbitMQ.

1.7.1. Install on Linux

On a machine with a supported Security World installation installed:

```
$ sudo tar -xf keysafe5-install/keysafe5-agent/keysafe5-1.1.1-Linux-keysafe5-agent.tar.gz -C /
$ sudo tar -xf agent-config.tar.gz -C /opt/nfast/keysafe5/
```

If the hardserver is already running, use the Keysafe 5 install script to not restart it:

```
$ sudo /opt/nfast/keysafe5/sbin/install
```

Otherwise use the nShield install script which will start all the services:

```
$ sudo /opt/nfast/sbin/install
```

1.7.2. Install on Windows

1. Run the `Keysafe5-agent.msi` installer if the Keysafe 5 agent is not already installed.

   This creates the nShield Keysafe 5 agent service but does not start it.

2. Populate the Keysafe 5 agent configuration using the files from the generated `agent-config.tar.gz`

   

   The generated `config.yaml` sets the log file location for the agent (`log.file.path`) to a Unix filepath. You will need to manually update the path to a file in `%NFAST_LOGDIR%`, e.g. `C:\ProgramData\nCipher\Log Files\KeySafe5-agent.log`.

3. Start the nShield Keysafe 5 agent service using Windows Service Manager.

1.8. Access Keysafe 5

You can now access Keysafe 5 through the URL provided by the deploy script. For example, you could send curl requests:

```
$ curl -s --cacert ca.crt https://$(hostname -f)/mgmt/v1/hsms | jq
$ curl -s --cacert ca.crt https://$(hostname -f)/mgmt/v1/pools | jq
```

You can access the Management UI in a web browser at `https://$(hostname -f)`.

1.9. Uninstall

If Kubernetes was not provided and K3s was installed by the deploy script, you may simply uninstall K3s which will clear up all the installed helm charts.

```
/usr/local/bin/k3s-uninstall.sh
```

This will request `sudo` permissions.

If a private Docker Registry was not provided, the deploy script will have created a local one and it will be removed when the script finishes. Should this fail, you may uninstall it manually by running:

```
docker stop registry
docker rm registry
```

If a Kubernetes installation was provided then the helm charts will need to be uninstalled individually.

```
helm --namespace nshieldkeysafe5 uninstall keysafe5-istio
helm --namespace nshieldkeysafe5 uninstall keysafe5-backend
helm --namespace nshieldkeysafe5 uninstall keysafe5-ui
```

If an existing RabbitMQ installation was not provided, then the deploy script will have installed a RabbitMQ helm chart that should be uninstalled. The same is also true for MongoDB.

```
helm --namespace rabbitns uninstall rabbit-chart
helm --namespace mongons uninstall mongo-chart
```

If Istio was installed by the deploy script, it may be uninstalled by running:

```
keysafe5-install/istioctl uninstall --purge
```

To uninstall the Keysafe 5 agent, run the Keysafe 5 uninstaller:

```
$ sudo /opt/nfast/keysafe5/sbin/install -u
```

2. Security Guidance

Your nShield HSM protects the confidentiality and integrity of your Security World keys. Keysafe 5 allows an authorized client to remotely configure and manage an estate of nShield HSMs. All network traffic between Keysafe 5 and clients using the UI, the REST API, or both, passes through a secure channel. This TLS based secure channel is set up using token-based client authentication. The administrator of the Keysafe 5 system must remain diligent concerning the entities who are given access to the system and the secure configuration of the system.

Entrust recommends the following security-related actions for Keysafe 5 deployments:

- Ensure that log levels are set appropriately for your environment.

  More verbose log levels might expose information that is useful for auditing users of Keysafe 5, but the log information also reveals which REST API operations were performed. While this log information might be useful for diagnostics, it could also be considered sensitive and should be suitably protected when stored.

- Rotate the logs regularly. The log files could grow quickly if left unattended for a long time. The system administrator is responsible for log rotation.
- Verify the integrity of the Keysafe 5 tar file before executing it. You can verify the integrity of this file with the hash provided with the software download.
- Suitably protect the network environment of Keysafe 5 to maintain its availability, for example using firewalls and intrusion detection and prevention systems.
- Ensure that the Keysafe 5 platform's system clock is set accurately and only authorized system administrators can modify it so that the platform correctly interprets certificate and token lifetimes.
- Ensure that only authorized system administrators have access to the Keysafe 5 system, and only trusted software is run on the platform hosting Keysafe 5.
- Take standard virus prevention and detection measures on the platform hosting Keysafe 5.
- The system administrator should consider whether threats in the Keysafe 5 deployment environment would justify the encryption of the sensitive configuration data held in Kubernetes secrets, see Kubernetes documentation.

3. Hardening The Deployment

To harden the demo deployment there are a number of steps to follow. The documentation below requires modifying the configuration of the Helm charts installed by following the Manual Install steps or running the `deploy.sh` script. To obtain the installed configuration for each installed Helm chart, run the following commands:

```
$ helm -n nshieldkeysafe5 get values --all --output yaml keysafe5-backend > keysafe5-backend-values.yaml
$ helm -n nshieldkeysafe5 get values --all --output yaml keysafe5-ui > keysafe5-ui-values.yaml
$ helm -n nshieldkeysafe5 get values --all --output yaml keysafe5-istio > keysafe5-istio-values.yaml
```

Documentation for each configurable value in the Keysafe 5 Helm charts can be found by untarring the chart.tgz and viewing the contents of either README.md or the default values.yaml file.

3.1. Certificates

The Manual Install steps and `deploy.sh` script will generate and install a number of short-lived demo certificates. These must be replaced to continue using the system.

Your certificates will need to adhere to X.509 v3, sometimes known as a multiple-domain certificates, or SAN certificates. The X.509 extension Subject Alternative Name (SAN) allows specifying multiple hostnames, and has replaced Common Name as the source of the hostname.

3.1.1. External Keysafe 5 Server TLS Certificate

To update the TLS certificate used for the Keysafe 5 Server (for HTTPS connections to the REST API or User Interface) you must create a Kubernetes Secret containing the new certificate/private key and redeploy the `keysafe5-istio` Helm chart.

For more information on enabling HTTPS for helm-keysafe5-istio, see the Helm Chart Installation section of the Installation Guide.

The Manual Install steps and `deploy.sh` script will create a Kubernetes Secret called `keysafe5-server-credential`. You can either delete the existing Secret as shown below, or use a different name for the Secret containing your new TLS certificate.

```
$ kubectl --namespace istio-system delete secret keysafe5-server-credential

$ kubectl --namespace istio-system create secret tls keysafe5-server-credential \
  --cert=path/to/cert/file \
  --key=path/to/key/file
```

Before running `helm upgrade`, set the following values in your `keysafe5-istio-values.yaml`:

- httpsEnabled=true
- tls.existingSecret=keysafe5-server-credential (or the name you used when creating the Kubernetes Secret containing your certificate/private key)

```
$ helm upgrade --install keysafe5-istio \
  --namespace=nshieldkeysafe5 \
  --values keysafe5-istio-values.yaml \
  --wait --timeout 1m \
helm-charts/nshield-keysafe5-istio-1.1.1.tgz
```

3.1.2. Internal Certificates

The Manual Install steps and `deploy.sh` script installs Keysafe 5 using TLS for the communications between the central platform and MonogDB/RabbitMQ.

You can refresh these internal certificates by running the `updateinternalcerts.sh` script, specifying the number of days the new certificates will be valid for.

The `updateinternalcerts.sh` script must be run from a directory containing the Keysafe 5 Helm charts (for example, from the root directory of the untarred Keysafe 5 package).

If an error occurs during certificate update you can restore the previous setup by rolling back Helm chart installations to a previous release, see Helm Chart Upgrade in the Upgrade section of the Installation Guide.

3.1.2.1. MongoDB TLS Certificates

The Manual Install steps and `deploy.sh` script installs the Bitnami MongoDB Helm chart with TLS enabled for the connections between Keysafe 5 and the MongoDB server and also with TLS used for authentication. These certificates will initially be valid for 30 days from the time the process was run.

You can refresh the MongoDB certificates by running the `updateinternalcerts.sh` script, specifying the number of days the new certificates will be valid for.

```
$ updateinternalcerts.sh mongodb 365
```

This script will:

- Generate the new TLS certificates
- Update the MongoDB helm chart to use the new certificates
- Update the `keysafe5-backend` helm chart to use the new certificates

3.1.2.2. RabbitMQ

The Manual Install steps and `deploy.sh` script installs the Bitnami RabbitMQ Helm chart with TLS enabled.

You can refresh the RabbitMQ certificates by running the `updateinternalcerts.sh` script, specifying the number of days the new certificates will be valid for.

```
$ updateinternalcerts.sh rabbitmq 365
```

This script will:

- Generate the new TLS certificates
- Update the RabbitMQ helm chart to use the new certificates
- Update the `keysafe5-backend` helm chart to use the new certificates
- Output a new `agent-config.tar.gz` that contains the agent configuration file and TLS certificates for authentication to RabbitMQ

You will need to update your client machines with a Keysafe 5 agent installed to use the updated config and restart the agent so that the new configuration is applied.

```
$ sudo tar xf agent-config.tar.gz -C /opt/nfast/keysafe5/
$ /opt/nfast/scripts/init.d/keysafe5-agent restart
```

3.2. Authentication

If you chose to install the demo deployment without authentication you should enable authentication for accessing the Keysafe 5 REST API and User Interface.

For how to configure authentication for the KeySafe 5 REST APIs see `Helm Chart Installation: helm-keysafe5-istio authentication` in the Installation Guide.

To update the `keysafe5-istio` Helm chart installed by the demo deployment, set the following values in `keysafe5-istio-values.yaml`.

- requireAuthn=true
- issuer[0].authIssuer="https://foobar.auth0.com"
- issuer[0].authJWKsURI="https://www.googleapis.com/oauth2/v1/certs"
- issuer[0].authAudiences[0]="https://keysafe5.location"

Then run `helm upgrade`.

```
$ helm upgrade --install keysafe5-istio \
  --namespace=nshieldkeysafe5 \
  --values keysafe5-istio-values.yaml \
  --wait --timeout 1m \
helm-charts/nshield-keysafe5-istio-1.1.1.tgz
```

To update the `keysafe5-ui` Helm chart installed by the demo deployment, set the following values in `keysafe5-ui-values.yaml`:

- authMethod=oidc

Untar the chart and copy your OIDC provider config file (`OIDCProviders.json`) into the config directory:

For more details on how to populate `OIDCProviders.json` and how to configure authentication for the Keysafe 5 User Interface see `Helm Chart Installation: Configure UI authentication` in the Installation Guide.

```
$ tar -xf helm-charts/nshield-keysafe5-ui-1.1.1.tgz -C helm-charts
$ cp my-oidc-provider-config.json helm-charts/nshield-keysafe5-ui/config/OIDCProviders.json
```

Then run `helm upgrade`:

```
$ helm upgrade --install keysafe5-ui \
  --namespace=nshieldkeysafe5 \
  --values keysafe5-ui-values.yaml \
  --wait --timeout 3m \
helm-charts/nshield-keysafe5-ui
```

3.3. K3s

If not using your own Kubernetes cluster, the `deploy.sh` script will create one using K3s. To harden this K3s install follow the official documentation at K3s Hardening Guide.

The `deploy.sh` script installs K3s with `traefik` and `metrics-server` explicitly disabled.