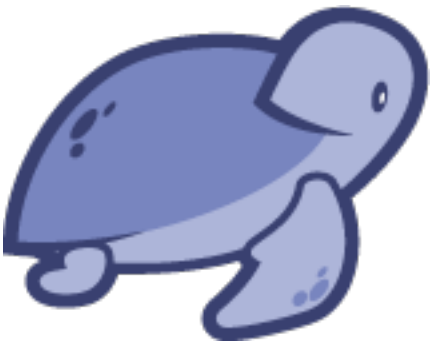
Keeto Documentation

Release 0.4.1-beta

Sebastian Roland

Apr 08, 2018

Contents



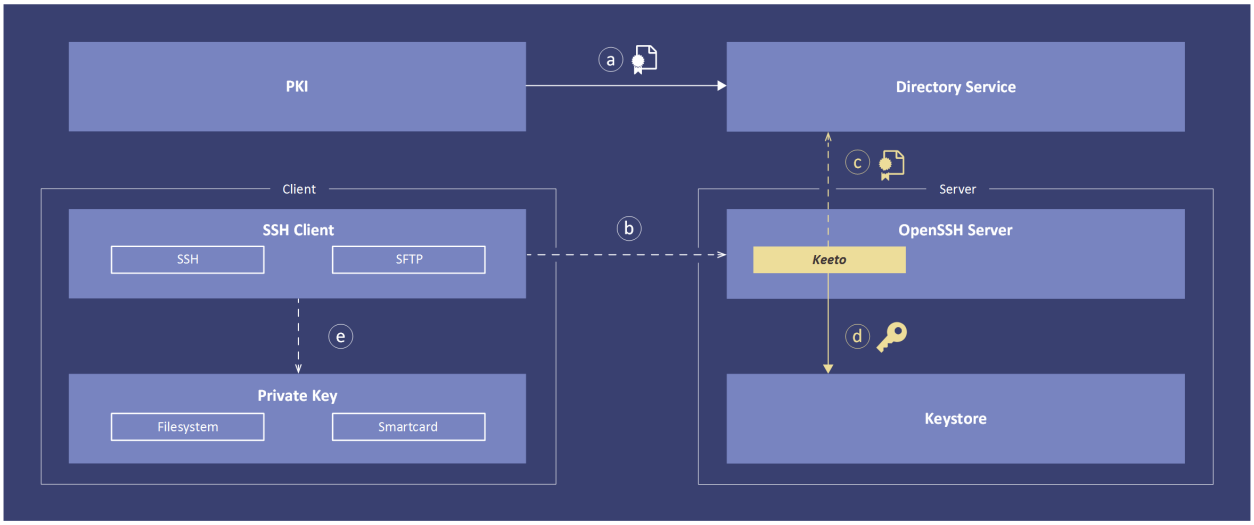
CHAPTER 1

Table of Contents

1.1 About

Keeto is a module for OpenSSH that enables profile-based administration of access permissions in a central LDAP aware Directory Service, adds support for X.509 certificates and handles the distribution of OpenSSH key material in an automated and secure manner.

1.2 Concept



The diagram shows a Keeto end to end flow. Each interaction is explained in the following. Note that flow **(a)** is periodically and all others are run on each SSH connection attempt.

(a) A Public Key Infrastructure (PKI) is responsible for managing the whole lifecycle of X.509 certificates. This managed X.509 certificates are distributed to a Directory Service where they can be used by Keeto.

(b) The SSH connection is triggered by a SSH protocol aware client such as PuTTY, FileZilla or WinSCP. An OpenSSH server receives the connection attempt.

(c) The control flow is passed over to Keeto which establishes a connection to the Directory Service and determines the current access permissions and retrieves relevant X.509 certificates for that connection.

(d) Keeto validates the X.509 certificates, extracts and transforms public keys and writes them to the appropriate authorized_keys file.

OpenSSH public key authentication is now taking over to authenticate the client against the freshly synced keys.

(e) The SSH client authenticates with the private key corresponding to the users X.509 certificate. Depending on the security requirements the private key can be held in software or hardware.

1.3 Getting Started

This chapter explains how to install and configure Keeto and its integration into OpenSSH. A description of the LDAP schema can be found in *LDAP Data Model*. You might also have a look at the fully configured Docker environment before starting from scratch yourself. A setup guide can be found at *Docker*.

1.3.1 Prerequisites

The following software packages are needed in order to build/run Keeto:

- OpenSSH >= 6.2
- Directory Service
- Syslog
- PAM
- pkg-config >= 0.9.9 (build only)
- libConfuse >= 2.7
- libcheck >= 0.9.9 (build only)
- OpenSSL >= 1.0
- libldap
- c_rehash

Make sure those components are installed and configured prior setting up Keeto.

1.3.2 Installation

Source

Grab the source tarball from https://keeto.io and unpack/build. Note that the library installation directory for PAM modules (–libdir) differs for various architectures/distros. Consult the documentation of your distro to figure out the right path:

```
<user>$ wget https://keeto.io/static/downloads/keeto-0.4.1-beta/keeto-0.4.1-beta.tar.
↪gz
<user>$ tar xvfz keeto-0.4.1-beta.tar.gz
<user>$ cd keeto-0.4.1-beta
```

```
<user>$ ./configure --libdir=/lib64/security
<user>$ make
<user>$ make check
<root>$ make install
```

RPM

Grab the RPM package from https://keeto.io and install:

```
<user>$ wget https://keeto.io/static/downloads/keeto-0.4.1-beta/keeto-0.4.1-0.1.beta.
↪el7.centos.x86_64.rpm
<root>$ rpm -i keeto-0.4.1-0.1.beta.el7.centos.x86_64.rpm
```

This installs the PAM modules and creates an initial configuration file keeto.conf as well as the authorized_keys and cert_store directories with the proper access permissions in /etc/ssh. Furthermore documentation is deployed.

1.3.3 Configuration

The following describes the configuration of the various components based on a installation from source. If an RPM package has been utilized certain steps do not need to be performed (see: *RPM*). Also notice that the location of the samples directory differs for an RPM based installation. It can be determined as follows:

```
<user>$ rpm -qd keeto
```

Keeto

Copy the configuration file keeto.conf from the samples directory into the OpenSSH configuration root folder and adjust it to your needs. An explanation of the various options can be found within the file. As the config contains sensitive data make sure it is only readable/modifiable by a privileged user:

```
<root>$ SSH_DIR=/etc/ssh
<root>$ cp samples/keeto.conf $SSH_DIR
<root>$ chmod 600 $SSH_DIR/keeto.conf
```

Furthermore create a directory where the authorized_keys files of the users are placed and a directory for trusted CA certificates and CRL's:

```
<root>$ mkdir $SSH_DIR/authorized_keys
<root>$ chmod 755 $SSH_DIR/authorized_keys
<root>$ mkdir $SSH_DIR/cert_store
<root>$ chmod 755 $SSH_DIR/cert_store
```

Copy all trusted CA certificates and CRL's (optional) for verifying the user certificates into the cert store. Make sure the whole chain except the end entity certificates are present. If StartTLS is used for the the LDAP connection also include the necessary certificates here. Finally create symlinks with:

```
<root>$ c_rehash $SSH_DIR/cert_store
```

OpenSSH

Make sure at least the following options are reflected in your sshd_config file:

```
PubkeyAuthentication yes
ChallengeResponseAuthentication yes
AuthorizedKeysFile /etc/ssh/authorized_keys/%u
UsePAM yes
AuthenticationMethods keyboard-interactive:pam,publickey
```

Optionally enable the processing of environment variables to export the real user (see: *What is the real user exported to the environment actually?*) to the environment and specify the algorithm used to hash the SSH keys fingerprint:

```
PermitUserEnvironment yes
#FingerprintHash md5
FingerprintHash sha25
```

If you are starting from scratch consider having a look at the sshd_config file provided in the samples directory as a starting point. Restart OpenSSH after all changes are made.

PAM

Copy the PAM configuration file for sshd to inject Keeto into the authentication process of OpenSSH:

```
<root>$ cp samples/sshd /etc/pam.d
```

Comment the optional audit module if you don't need the <uid, key fingerprint> mappings logged.

Syslog

Keeto logs to the syslog facility specified in keeto.conf. Adjust your syslog server accordingly. A sample config for syslog-ng can be found in the samples directory that logs Keeto output to a local file.

Directory Service

Keeto consults a Directory Service in order to obtain current access permissions and keys. The relevant entities and their relationship are described in *LDAP Data Model*. General configuration is software dependent and not not outlined here. The samples directory however contains relevant configuration files for the OpenLDAP Directory Service.

1.4 LDAP Data Model

Keeto obtains all access permissions and the key material from a central Directory Service. This section describes syntax and semantic of the entities involved and how they relate to each other. Certain attributes can be configured in the Keeto configuration file. Those attributes are of the form <x> where x is referring to the key in the Keeto configuration file.

1.4.1 SSH Server

The SSH server entry is the starting point for the determination of access permissions and key material. It specifies the relevant access profiles that shall be taken into account. Keeto locates the right SSH server entry through a unique identifier that is specified in the Keeto configuration file and must match the identifier within the SSH server entry in the Directory Service.

```
objectClass: top
objectClass: keetoSSHServer
```

Attribute	Mandatory	Single-Value	Description
cn	yes	no	RDN of SSH server entry.
uid	yes	no	Unique identifier of SSH server. See also: <ldap_ssh_server_uid>.
keetoAccessProfile	no	no	DN to Keeto access profile.
description	no	no	SSH server description.

1.4.2 Access Profiles

Keeto supports two different access profiles. Direct access profiles provide access to one's own account whereas access on behalf profiles provide access to someone else's account.

Direct Access Profile

A direct access profile specifies references to key providers either directly or through groups that shall be able to login with it's own account. Each key provider's UID is checked against the UID of the user about to login. If they match the X.509 certificates of the key provider will be taken into account. Optionally keystore options can be specified that are used for all key providers. A direct access profile can be enabled/disabled.

```
objectClass: top
objectClass: keetoAccessProfile
objectClass: keetoDirectAccessProfile
```

Attribute	Mandatory	Single-Value	Description
cn	yes	no	RDN of direct access profile entry.
keetoEnabled	yes	yes	Enable/Disable access profile.
keetoKeyProvider	no	no	DN to Keeto key provider.
keetoKeyProviderGroup	no	no	DN to Keeto key provider group.
keetoKeystoreOptions	no	yes	DN to Keeto keystore options.
description	no	no	Direct access profile description.

Access On Behalf Profile

Access on behalf profiles enable authentication on behalf of someone else. For that the UID of the user about to login is searched in the target keystores which are either directly linked to the profile or through groups. On match all valid keys of all key providers are synced into that target keystore. As with direct access profiles keystore options can be specified optionally and the profile can be enabled/disabled.

```
objectClass: top
objectClass: keetoAccessProfile
objectClass: keetoAccessOnBehalfProfile
```

Attribute	Mandatory	Single-Value	Description
cn	yes	no	RDN of access on behalf profile entry.
keetoEnabled	yes	yes	Enable/Disable access profile.
keetoKeyProvider	no	no	DN to Keeto key provider.
keetoKeyProviderGroup	no	no	DN to Keeto key provider group.
keetoTargetKeystore	no	no	DN to Keeto target keystore.
keetoTargetKeystoreGroup	no	no	DN to Keeto target keystore group.
keetoKeystoreOptions	no	yes	DN to Keeto keystore options.
description	no	no	Access on behalf profile description.

1.4.3 Key Provider

A key provider is an entry that provides the key material through X.509 certificate(s). All relevant attributes are specified in the Keeto configuration file to adjust it to different deployments. Key providers are relevant for both direct access profiles and access on behalf profiles.

Attribute	Mandatory	Single-Value	Description
<ldap_key_provider_uid_attr>	yes	no	UID of key provider.
<ldap_key_provider_cert_attr>	yes	no	X.509 certificate of user.

1.4.4 Target Keystore

A target keystore is only relevant for access on behalf profiles. It specifies the accounts that can be used on behalf of the key providers.

Attribute	Mandatory	Single-Value	Description
<ldap_target_keystore_uid_attr>	yes	no	UID of target keystore.

1.4.5 Groups

Key providers and target keystores can be linked to an access profile throug groups. For both cases the group member attribute is specified in the Keeto configuration file.

Attribute	Mandatory	Single-Value	Description
<ldap_key_provider_group_member_attr> / <ldap_target_keystore_group_member_attr>	yes	no	DN to key provider/target keystore.

1.4.6 Keystore Options

Keystore options can be optionally specified and linked to access profiles to restrict access with regard to the location a user is connecting from and the space of commands he is allowed to execute.

```
objectClass: top
objectClass: keetoKeystoreOptions
```

Attribute	Mandatory	Single-Value	Description
cn	yes	no	RDN of keystore options entry.
keetoKeystoreOptionFrom	no	yes	authorized_keys 'from' option entry. See also: man sshd.
keetoKeystoreOptionCommand	no	yes	authorized_keys 'command' option entry. See also: man sshd.
description	no	no	Keystore options description.

1.5 Docker

Keeto ships a fully configured Docker image providing an easy and fast way for the establishment of an environment in order to become familiar with Keeto. This chapter gives an overview about this environment, describes how to set it up and ultimately use it.

1.5.1 Overview

The Docker environment consists of the following containers:

Container	Description
keeto-openssh	Keeto-enabled OpenSSH server
keeto-openldap	Directory Service
keeto-syslog-ng	Central logging server
keeto-mariadb	Audit database

The OpenSSH server obtains access permissions and key material from the Directory Service and sends audit relevant SSH session information to the central logging server which stores the records in the audit database.

Services

The following ports are exposed by the Docker environment:

Container	Protocol	Ip:Port
keeto-openssh	SSH	127.0.0.1:1022
keeto-openldap	LDAP Plain/StartTLS	127.0.0.1:1389
keeto-openldap	LDAPS	127.0.0.1:1636
keeto-mariadb	MariaDB	127.0.0.1:13306

OpenLDAP Settings

Option	Value
LDAP URI	ldap://127.0.0.1:1389
Base DN	dc=keeto,dc=io
Bind DN	cn=directory-manager,dc=keeto,dc=io
Bind password	test123

MariaDB Settings

Option	Value
Ip:Port	127.0.0.1:13306
Database	keeto-audit
Username	root
Password	test123

OpenSSH Access Permissions

Key/User	Direct Access	Access On Behalf
birgit	Yes	-
bjoern	No	keeto
oliver	No	slapd, opendj
sebastian	No	keeto
trixi	No	slapd, opendj
wolfgang	Yes	-

1.5.2 Prerequisites

The following software packages are needed in order to run the Keeto Docker environment:

- Docker
- Docker Compose
- SSH client
- LDAP client (optional)
- MariaDB client (optional)

Although any LDAP client will do Apache Directory Studio is recommended as Keeto provides an export of the connection settings needed to configure the client for the usage with the Docker environment. The SSH client has to support either PKCS#8 or PuTTY's .ppk private key format for public key authentication.

1.5.3 Setup

Grab the source code tarball from https://keeto.io and unpack the distribution. All files needed to setup the Docker environment are included in the 'samples/docker' directory:

```
<user>$ wget https://keeto.io/static/downloads/keeto-0.4.1-beta/keeto-0.4.1-beta.tar.
↪gz
<user>$ tar xvfz keeto-0.4.1-beta.tar.gz
<user>$ cd keeto-0.4.1-beta/samples/docker
```

Now start the containers using Docker Compose with the following command:

```
<root>$ docker-compose up -d
```

Docker will download the images if they are not already available locally and subsequently start the environment. Finally you should see the following output:

```
Creating network "docker_keeto-net" with driver "bridge"
Creating keeto-mariadb
Creating keeto-openldap
Creating keeto-syslog-ng
Creating keeto-openssh
```

Thats it! The Keeto Docker environment is now fully operational.

1.5.4 Usage

Now that the environment is up and running you are able to play around and gain a better understanding of Keeto. Configure your favourite LDAP client with the settings described in *OpenLDAP Settings* and browse/modify the content in the OpenLDAP Directory Service. If you are using Apache Directory Studio you might wanna import the connection settings from the 'samples/docker/misc' folder. The environment comes with some predefined access permissions as described in *OpenSSH Access Permissions*. The private key material for the various logins is available in the 'samples/docker/keys' folder. Note that some SSH clients require the private key to have certain access permissions. If you are using such a client change permissions of the private key file accordingly.

The following two examples show logins with the OpenSSH SSH client for a user that has direct access and another one that has access on behalf of another account:

```
<user>$ chmod 600 keys/birgit-key.pem
<user>$ ssh -i keys/birgit-key.pem -p 1022 birgit@localhost
<user>$ chmod 600 keys/oliver-key.pem
<user>$ ssh -i keys/oliver-key.pem -p 1022 slapd@localhost
```

Connect to the database (see: *MariaDB Settings*) to have a look at the SSH session logging.

1.6 FAQ

1.6.1 How can I exclude accounts from being processed by Keeto?

Keeto is associated with keyboard-interactive authentication via PAM. Disabling keyboard-interactive authentication for specific users does the trick. This can be configured in the sshd_config file of OpenSSH through the Match directive as follows:

```
Match User foo,bar
    AuthenticationMethods publickey
    AuthorizedKeysFile .ssh/authorized_keys
```

1.6.2 Not all certificates/CRL's in the certificate store are utilized

Make sure that every certificate/CRL has a corresponding symlink created with the c_rehash utility. Beware that older version of c_rehash only processed files with the extension .pem. If you are using such a version consider symlinking or renaming the original file with the proper extension.

1.6.3 Keeto doesn't seem to check the LDAP server certificate chain against the CRL

If Keeto has been configured to check CRL's it depends on the crypto library libldap has been linked against. As for now CRL checking is only supported if libldap has been linked against OpenSSL. In any other case the CRL check is skipped. Keeto logs the following entry if the CRL check cannot be performed through libldap:

```
[C] failed to set ldap option: key 'LDAP_OPT_X_TLS_CRLCHECK', value '2'
```

Note that this only applies to the verification of the LDAP server certificate chain during secure connection establishment (StartTLS/LDAPS). Validation of user certificates against the CRL will always be performed if 'check_crl' has been set to '1' in the Keeto configuration file.

1.6.4 What is the real user exported to the environment actually?

The environment variable $KEETOREALUSER contains the UID of the user who owns the key that was used during public key authentication. This is meaningful for scenarios where an account is shared across different users and one would like to apply certain configuration based on the real entity.