X.509 Certificate Management with FreeIPA

Thorsten Scherf
Red Hat EMEA

SLAC 2016 - Berlin

Problem statement

We need certificates for:

- Services/Hosts
- Users

to provide:

- Confidentiality
- Integrity
- Authenticity

X.509 Certificates are widely used within Enterprises, but it's a complex technology.

Only very few OpenSource solutions are available.

Yes, there is OpenSSL and friends, but those tools don't scale very well.

Ideally we want to have a solution which is integrated into a larger IdM framework.

FreeIPA

FreeIPA provides an Identity-Store for users, groups, hosts and services.

It also contains authorization policies to control access to hosts and services.

A PKI component is included to issue and manage X.509 certificates.

FreeIPA PKI component

The FreeIPA PKI is based on the Dogtag project.

Dogtag provides various APIs to access internal functions.

The PKI comes with a CA and KRA (also known as DRM) component.

Two setup options exists:

- Root-CA
- Subordinate-CA

Certificates can be published in a replicated LDAP-database.

Client tools exist to request, renew and revoke certificates.

Certificate profiles

FreeIPA comes with default profiles primarily used for Server- and Client TLS authentication.

Additional profiles can be created and imported into FreeIPA.

```
$ ipa certprofile-show caIPAserviceCert --out caIPAserviceCert.cfg
```

```
$ ipa certprofile-import caIPAClientAuthSigning \
--file caIPAClientAuthSigning \
--desc "Profile for user authentication and signing" \
--store TRUE
```

Certificate ACLs

Users are not allowed to request certificates on their own.

ACLs can be used to give access to certain profiles.

Create a new ACL:

```
$ ipa caacl-add acl_ClientAuthSigning
```

Assign a group to the ACL:

```
$ ipa caacl-add-user acl_ClientAuthSigning --group ipausers
```

Assign a profile to the ACL:

```
$ ipa caacl-add-profile acl_ClientAuthSigning \
--certprofile caIPAClientAuthSigning
```

Certmonger

Certmonger is a D-BUS based service which talks via https/xmlrpc to the FreeIPA server.

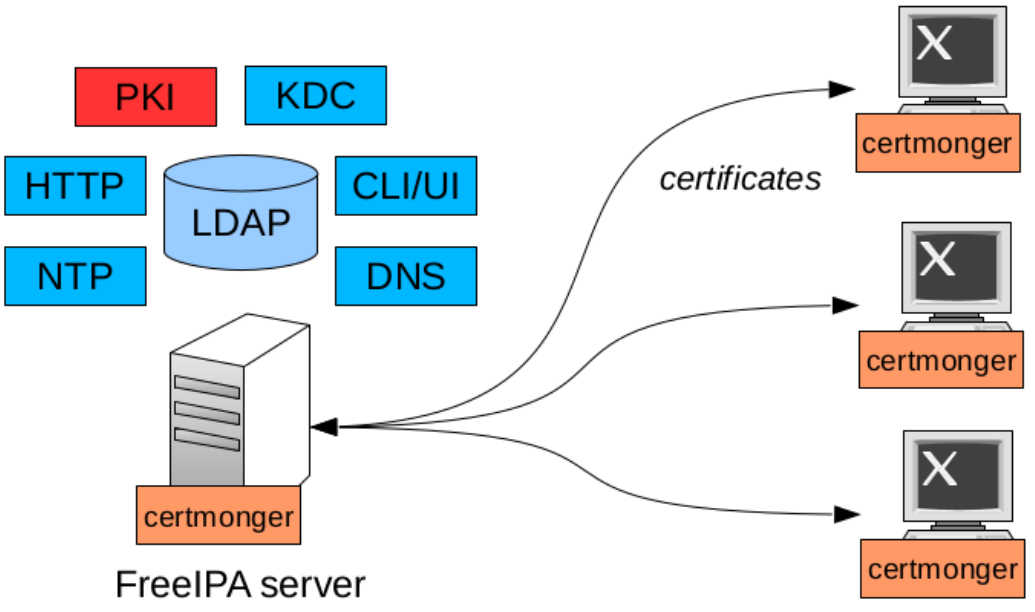
The certmonger client tool can be used to request, renew and revoke certificates.

Certmonger can renew certificates automatically before they expire.

The tool works with PEM-encoded files or NSS databases.

Certmonger supports the Simple Certificate Enrollment Protocol (SCEP).

Highlevel Architecture



Service Certificates

How to request a PEM-encoded certificate

```
$ ipa-getcert request -f /path/to/server.crt -k /path/to/private.key
```

How to request a NSS-based certificate

```
$ ipa-getcert request -d /path/to/database -n 'test-cert'
```

```
$ ipa-getcert list [ 'cert-id' ]
Request ID '20150901145517':
    status: MONITORING
    stuck: no
    key pair storage: type=NSSDB,location='/etc/httpd/alias',nickname='Server-Cert',token
    certificate: type=NSSDB,location='/etc/httpd/alias',nickname='Server-Cert',token='NSS
    CA: IPA
    issuer: CN=Certificate Authority,O=TUXGEEK.DE
    subject: CN=tiffy.tuxgeek.de,O=TUXGEEK.DE
    expires: 2016-08-04 11:19:38 UTC
    key usage: digitalSignature,nonRepudiation,keyEncipherment,dataEncipherment
    eku: id-kp-serverAuth,id-kp-clientAuth
    [...]
```

User Certificates

- Smart Card Authentication
- Client Certificate Authentication
- S/MIME Signing Certificates

Currently the CSR has to be created outside of FreeIPA.

```
$ ipa cert-request tscherf.csr --principal tscherf --profile-id caIPAuserCert
  Certificate: MIIDgDCCAmigAwIBAgIBDzAN[…]
  Subject: CN=tscherf,OU=pki-ipa,O=IPA
  Issuer: CN=Certificate Authority,O=VIRT.TUXGEEK.DE
  Not Before: Thu Jun 09 17:33:42 2016 UTC
  Not After: Sun Jun 10 17:33:42 2018 UTC
  Fingerprint (MD5): 87:93:bc:cb:5b:d6:d1:10:40:2f:29:1a:b1:8b:0b:98
  Fingerprint (SHA1): fc:83:40:dc:de:4a:6b:1b:ec:db:7d:03:f8:fa:55:b3:ce:fb:cb:37
  Serial number: 15
  Serial number (hex): 0xF
```

Smart Cards

The primary use case for user certificates is to support Smart Card Authentication.

SSSD provides a new PKCS#11 interface. Mapping between certificates and users is much easier than with pam_pkcs11.

Certificate is matched based on Kerberos Principal Name. Other options are in the works.

SSSD only requires "*pam_cert_auth = True*" in sssd.conf **[pam]** section.

Provision a Smart Card

Create the CSR based on an existing key:

```
OpenSSL> req -engine pkcs11 -new -key slot_1-id_11 -keyform engine -out
/home/tscherf/tscherf.csr -text -config /home/tscherf/openssl.cfg
```

Request the certificate from FreeIPA:

```
$ ipa cert-request ~/tscherf.csr --principal tscherf \
--profile-id caIPAClientAuthSigning

$ ipa user-show tscherf --out=tscherf.pem
```

Transfer the new certificate to the card:

```
$ pkcs15-init --store-certificate /home/tscherf/tscherf.pem \
--auth-id 01 --id 11 --format pem
```

Certificate based SSH-Logins

Remote hosts need to have access to the certificate.

The solution to the problem is to either:

- Put them into a local *~/authorized_keys* file
- Put them in LDAP and tell sshd to download them.

FreeIPA has user certificates already stored as part of the user entry.

OpenSSH uses the *AuthorizedKeysCommand* option to download certificates:

```
AuthorizedKeysCommand /usr/bin/sss_ssh_authorizedkeys
```

New Features

- Lightweight Sub-CAs
- Certs in ID overrides for AD users
- pkinit support for Smart Card use case
- Support for SCEP (RfE)
- Support for ACME (RfE)

freeIPA

identity | policy | audit

Thank you!

Thorsten Scherf

tscherf@redhat.com