Building an Open Source PKI using OpenXPKI

by Alexander Klink and Michael Bell
OpenXPKI

1. Some theory (concepts) by Michael

2. Some practical experiences (concepts + demo) by Alex
Concepts

- PKI Realm
- Crypto Abstraction
- Configuration Inheritance
- Workflow
Concepts – PKI Realm

CA Migration

- Root CA (e.g. DFN-PCA)
- Intermediate CA (e.g. HU-CA 1)
- End Entity CA (e.g. HU-DCA 3)
- Root CA (e.g. DFN-PCA Classic)
- End Entity CA (e.g. HU-CA 4)
Concepts – PKI Realm

Merger

- Root CA Provider LowLevel
- Root CA Provider SuperSecureLevel
- Alpha Bank CA
- Zulu Bank CA
- Alpha Zulu Bank CA
Concepts – PKI Realm

- You never want to lose certificates and CSRs because of a migration.
- Design or time-based rollovers should not influence the operations.
- How to (and why) import a PKI?
  - with/without CA key
- We simply need a playground ;-)
Concepts – PKI Realm

Multiple isolated CAs

- Root CA 1
- Root CA 2
- Intermediate CA 1
- Intermediate CA 2
- User CA
Concepts – PKI Realm

Domain/Group Concept

- Root CAs
- Intermediate CAs
- User CAs
- Server CAs
Concepts – PKI Realm

• PKI Realm Definition

  – A PKI realm is a namespace for CSRs, certificates, CA certificates, CRLs and any other PKI related information.
Concepts – PKI Realm

• Certificate Storage Definition

  – Every certificate exists only once.

  – Every certificate references its issuer via a SHA-1 hash of the issuer's certificate.

  – Every certificate is a member of exactly one PKI realm.
Concepts – PKI Realm

- Provider CA
- Company CA
- User CA
- AnotherTrustWorthy CA
- Server CA
- Michael Bell
- Alexander Klink
- John Doe

Null Reference
Concepts - PKI Realm

• Additional implementational sugar
  - aliases
  - special CA certificate handling
Concepts – PKI Realm

• Alias Definition

  – An alias is a reference to a certificate.

  – An alias consists of a name, a SHA-1 hash and a PKI realm.

  – Every alias is only valid inside of exactly one PKI realm.
Concepts – PKI Realm

● CA Handling

- Every CA Certificate must have set the PKI realm of its issuers PKI realm.

- The CA cert(s) inside of a PKI realm are only present as aliases.

- All self-signed (CA) certs are in a special PKI realm called “” (empty word).
Concepts – PKI Realm

(Too) Heavy stuff?
Concepts – PKI Realm

special PKI realm “”

Company CAs

User CAs

OpenXPKI CA

Root CA

???

OpenXPKI CA

Michael Bell

Alexander Klink

???
Concepts – PKI Realm

special PKI realm ""

Company CAs

Root CA

OpenXPKI User CA

OpenXPKI Server CA

User CAs

OpenXPKI CA

Server CAs

OpenXPKI CA
Concepts – PKI Realm
Do not try this at home!!!
Concepts – PKI Realm
Do not try this at home!!!

special PKI realm “”

Company CAs

OpenXPKI User CA

User CAs

OpenXPKI CA

Server CAs

Root CA

Root CA
Concepts – PKI Realm
Do not try this at home^Wwork!!!

special PKI realm “”

Company CAs

End Entity CAs
Concepts – PKI Realm

special PKI realm “”

Root CA 1

Root CA 2

Root CA

OpenXPKI User CA

Company CAs

OpenXPKI CA

End Entity CAs

Root CA
Concepts
Crypto Abstraction

- OpenSSL vs. OpenTLS vs. NSS
- patched toolkits (e.g. for GOST)
- different protection classes of used keys incl. m out of n passphrases for software keys (e.g. CA, SCEP, key backup)
- Hardware Security Modules (HSM)
Concepts
Crypto Abstraction

Crypto::API
- Crypto::Backend::API
- Crypto::Tool::SCEP::API

Crypto::Toolkit
- Crypto::Backend::OpenSSL
- Crypto::Tool::SCEP

Crypto::CLI
- Crypto::Backend::OpenSSL::CLI
- Crypto::Tool::SCEP::CLI

OpenSSL CLI
OpenCA SCEP Tool
Concepts
Crypto Abstraction

• Example: crypto algorithms
  - RSA
  - DSA/ECDSA
  - GOST algorithms

• Example: passphrase handling
  - hard coded
  - (splitted) plain
  - m of n as described by Adi Shamir
  - HSMs
Concepts
Crypto Abstraction

- TokenManager
  - Secret
    - Secret::Split
    - Secret::Plain
  - Backend::API
    - Backend::OpenSSL
      - Backend::OpenSSL::CLI
      - Backend::OpenSSL::Engine

OpenSSL CLI
Concepts

Config Inheritance

• Problems
  - many repetitions
  - new CA with only minimal changes
  - basic config and only small local additions

• Solution
  - configuration inheritance
  - attribute id ::= identify local nodes
  - attribute super ::= path to the parent
Concepts
Config Inheritance

<common>
  <test id="default">
    <value>My Test</value>
  </test>
</common>

<deep>
  <test super="../common/test{default}"/>
  <test super="/common/test"/>
</deep>
Concepts - Workflow

- Which problems are the most common ones with trustcenter software?
- Cool demo systems, BUT …
  - We need the following customization(s) …
  - We have an ERP/HR/CRM system XYZ …
  - My business process is the following one …

... AND THEN THE HORRORS BEGIN …
Concepts - Workflow

Server

- Server::API
- Server::API::Workflow
- Workflow

Clients:
- Conditions
- Validators
- Activities
- Config

Context:
- XML::Config
- DBI
- Session
- Crypto

Michael Bell, CMS Abt.1
Humboldt-Universität zu Berlin
Concepts – Workflow

How does this design help?

Please give Alex a chance ;-)
Structure
from abstract to concrete

- Workflows – example Certificate Request
- Simple Certificate Enrollment Protocol (SCEP)
- Smartcard personalization
- Support
- Hackers wanted!
Workflows

Example: Certificate Request

• How does it look from a user perspective?
• Certificate Request using a web interface
  • browser-based request (SPKAC/XEnroll)
  • PKCS#10 upload
• key generation on the CA
• Let’s have a look
Workflows

Certificate Request: User Experience
Workflows

Certificate Request: User Experience

Select the type of key generation

Please choose one of the following available options for the key generation. This is important because this influences the next steps. The options mean quite different things. So please read the descriptions carefully.

- SPKAC
- Microsoft Internet Explorer
- Server-side key generation
- PKCS#10
- Automatic browser detection

SPKAC

SPKAC was originally designed by Netscape. Today this format is used by several different browsers. Such browsers are Netscape, Mozilla, Firefox and Opera. If you have such a browser and you want to create the key using your client then please use this type of key generation.
Workflows

Certificate Request: User Experience

Creating the name of the certificate

The name of the certificate is built from the values of the following form fields. Please enter the required information. If you are not sure about what to enter then please read the descriptions of the fields at the end of this page.

uid = aklink + Common name = Alexander Klink
Organizational Unit = Development
Organization = OpenXPKI
dc = org

Descriptions of the form fields

Common name
The common name can be used to identify persons or objects by its name. Examples are John Doe or www.openxpk.org.
Workflows

Certificate Request: User Experience

Subject alternative name

A certificate may have several so called alternative names. These names give more information about the certificate use. Certificates for servers usually include the DNS names and IP addresses of the server. Normal user certificates include the email address. If you use Microsoft Smartcard login then you must set the UPN. Please set your required alternative names.

- emailAddress
- a.klink@cynops.de
- emailAddress
- DNS
- GUID
Workflows

Certificate Request: User Experience
Workflows

Abstract: visualization using GraphViz
Workflows

Abstract: visualization using GraphViz
Workflows

Abstract: visualization using GraphViz

Flexible Approval
Workflows

more concrete: XML configuration

```xml
<state name="APPROVAL">
  <action name="persist_csr"
    resulting_state="CSR_PERSISTED">
    <condition name="ACL::persist_csr"/>
    <condition name="Condition::check_csr_approvals"/>
  </action>
  <action name="approve_csr"
    resulting_state="APPROVAL">
    <condition name="ACL::approve_csr"/>
  </action>
  <action name="cancel_csr_approval"
    resulting_state="PENDING">
    <condition name="ACL::cancel_csr_approval"/>
  </action>
  <action name="reject_csr"
    resulting_state="REJECTED">
    <condition name="ACL::reject_csr"/>
  </action>
</state>
```

a snippet from workflow_def_certificate_signing_request.xml
Workflows

at the bottom: Perl & Workflow.pm

```perl
sub execute {
    my $self = shift;
    my $workflow = shift;
    my $context = $workflow->context();

    my $type = $context->param('csr_type');
    my $profile = $context->param('cert_profile');

    [ ... ]

    $dbi->insert(
        TABLE => 'CSR',
        HASH => {
            'PKI_REALM' => $pki.realm,
            'CSR_SERIAL' => $csr_serial,
            'DATA' => $data,
            [ ... ]
        },
    );
    $dbi->commit();
    $context->param('csr_serial' => $csr_serial);
}
```
SCEP
What is it all about?

• SCEP = Simple Certificate Enrollment Protocol
• IETF draft initiated by Cisco
• Protocol to automatically enroll for or renew certificates
• PKCS#10 in PKCS#7 over HTTP (so it could be worse)
• Implemented in many hardware devices
SCEP

The motivation

- Alternative: doing it manually
  - request certificate, install, note down expiry date in (probably at least next year’s) calendar
  - come expiry date, request new certificate, install
  - repeat, don’t make mistakes
  - pretty unsuitable for larger installations
SCEP

The “user” perspective: sscep

Request CA certificate from SCEP server

```
trinidad:~ klink$ sscep getca -u http://127.0.0.1:8042/cgi-bin/sscep -c cacert
sscep: requesting CA certificate
sscep: valid response from server

sscep: found certificate with
  subject: /DC=org/DC=OpenXPKI/OU=Development/UID=sscep/CN=SCEP Testserver
  issuer: /C=DE/O=Local Test/OU=PKI/CN=Local Test Root DUMMY CA 2
  usage: Key Encipherment
sscep: certificate written as cacert-0

sscep: found certificate with
  subject: /C=DE/O=Local Test/OU=PKI/CN=Local Test Root DUMMY CA 2
  issuer: /C=DE/O=Local Test/OU=PKI/CN=Local Test Root DUMMY CA 2
  usage: Certificate Sign, CRL Sign
sscep: certificate written as cacert-1
trinidad:~ klink$
```
SCEP
The “user” perspective: sscep

Send Certificate Signing Request to CA

Using a previously generated private key and PKCS#10
SCEP

The “user” perspective: sscep

Send Certificate Signing Request to CA

...wait for approval

... done!
SCEP
Workflows revisited

this looks familiar ...
SCEP

Workflows revisited

Approval again:
SCEP
Alternatives

• PKIX-CMP (Certificate Management Protocol)
  • standardized in RFC 4210, 4211, but not widely used ...

• CMC (CM over CMS, no transport defined)
  • RFC 2797, used by Microsoft CA with COM/DCOM as proprietary transport
  • we would like to see support for that to be able to easily replace a MS CA for domain controller enrollment
SCEP
(Santa’s?) Little helper: CertNanny

• Automatic renewal and keystore modification system (SCEP + signature using “old” certificate)

• Available on Unix (tested: Linux, AIX, Solaris, Darwin), Win32, Tandem NonStop

• Modifies OpenSSL keys, PKCS#8, Java Keystores, IBM GSKit, Windows Certificate Store

• widely deployed within a large financial institution
Smartcard personalization

What do we need smartcards for?

- Authentication:
  - Smartcard logon
  - (W)LAN authentication (802.1x)
  - Fileserver encryption (outsourcing!)
- (E-Mail)-Encryption
- you have to think about key recovery
Smartcard personalization

Why a self-service personalization?

- To get a huge set of smartcards in a working state
- Do It Yourself (DIY)
- Let the users do it
- Obvious choice :-)
- User only needs IE, Smartcard drivers and some time → screencast
Support
If RTFM does not help ...

- Actually, the fine manual is not yet finished ...
- ... but even when it is, OpenXPKI is still quite a complex piece of software
- Support via mailing list: openxpki-users@lists.sf.net
- Shameless plug: Cynops GmbH offers commercial support as well
Hackers wanted

Use it, break it, enhance it ...

• We are always looking for more people to help
  • Developers
  • Bug^H^H^H feature reporters :-)
  • people to WTFM (just kidding)
  • Code auditors
  • Users of hardware that talks SCEP

• Talk to us now, or later: openxpki-devel@lists.sf.net
Questions?
Comments? Confusion?

• Now is the time to ask ....
• But later is fine too:
  • Alex – DECT 2412, ak-23c3@cynops.de
  • Michael – michael.bell@cms.hu-berlin.de
• Thanks for your time!