Building an Open Source Public Key Infrastructure using OpenXPKI

Alexander Klink, Cynops GmbH (a.klink@cynops.de)
Martin Bartosch, Cynops GmbH (m.bartosch@cynops.de)
Michael Bell, HU Berlin (michael.bell@cms.hu-berlin.de)

Introduction

OpenXPKI is an open source trust center solution developed by the OpenXPKI Project. It aims at creating an enterprise-scale PKI/trust center software running on Unix-based systems supporting well-established infrastructure components such as RDBMS and Hardware Security Modules. It is the successor of OpenCA and builds on the experience gained while developing it.

Note that when we say enterprise-scale, we actually mean it. OpenXPKI is not yet another one of those projects for setting up the self-signed CA of the geek next door. Not that OpenXPKI might not appeal to geeks, but it aims to provide a different class of Certificate Authority. Real open source competition is not visible on the market, whereas commercial PKI solutions usually cost a fortune and offer less flexibility. These are some of the reasons why a large financial corporation plans to use OpenXPKI in production »pretty soon now«.

Written in object-oriented Perl, it has quite a flexible architecture which makes hacking it to your liking pretty easy and fun.

Features

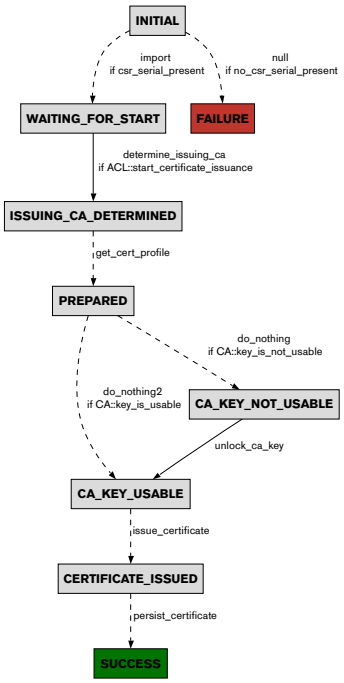
OpenXPKI has quite an advanced feature set, supporting all of the basic operations a Public Key Infrastructure has to offer, including modular authentication, a user interface API and is designed for scalability.

But there are some more features which we believe distinguish ourself from the competition.

Workflow engine

We use the Workflow.pm module from CPAN as a workflow engine, which allows us to create much more flexible code and configuration. Basically, the workflow engine provides a state machine – a workflow can be in a certain state, from which it can change into a different state using an activity.

Activities are only available if certain conditions are met. Input data for a workflow activity can be validated using so-called »validators«. Each workflow has its own context, which is where it stores the data associated with it. As an example, here is a visualization of our certificate issuance workflow and a snippet from the corresponding XML configuration file:




```
<state name="INITIAL">
 <action name="import"
      resulting_state="WAITING_FOR_START">
  <condition name="csr_serial_present"/>
 </action>
 <action name="null"
  resulting_state="FAILURE">
  <condition name="no_csr_serial_present"/>
 </action>
</state>

<state name="WAITING_FOR_START">
 <action name="determine_issuing_ca"
      resulting_state="ISSUING_CA_DETERMINED">
  <condition name="ACL_cert_issuance"/>
 </action>
</state>
```

Figure 1 Graphviz rendering and a snippet from the XML definition

As you can see, defining such workflows is pretty straightforward. The activities (in Workflow.pm they are called »actions«) and the conditions map to Perl modules which transform data from the workflow context or checks whether certain conditions are met. The workflow context is saved in a database, so that retrieving the context data or searching for it is pretty easy and »outsourced« to the database layer.

This gives us a much better infrastructure for custom definitions than was possible with OpenCA. Pre-defined workflow definition and implementations include certificate requests using different methods, certificate issuance, CRL issuance, SCEP, Smartcard personalization, etc. These can be easily re-used in custom workflow definitions – normally, a customized workflow is only a few changes in the XML file and a few lines of Perl away.

PKI Realms & automatic CA rollover

Most commercial PKI solution vendors will want to sell you a new piece of software for a new Certificate Authority. Contrary to that, OpenXPKI offers to run several completely independent CAs within the same installation. This leads to what we call »PKI Realms«, which groups together CAs with the same task – you might have a PKI realm for your employees, one for your servers and one for your customers for example. Within these realms, you can define CAs which might even be valid at the same time.

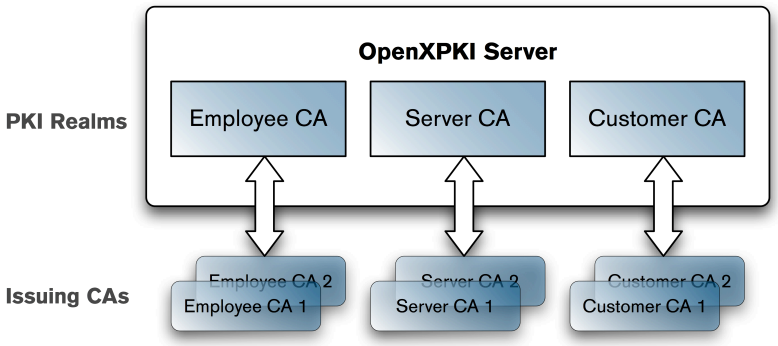


Figure 2 OpenXPKI structure: PKI Realms

Maybe you are wondering why you would want to have CAs that are valid at the same time. We implemented this to solve one of the problems most PKI solutions have – the expiry of the CA certificate.

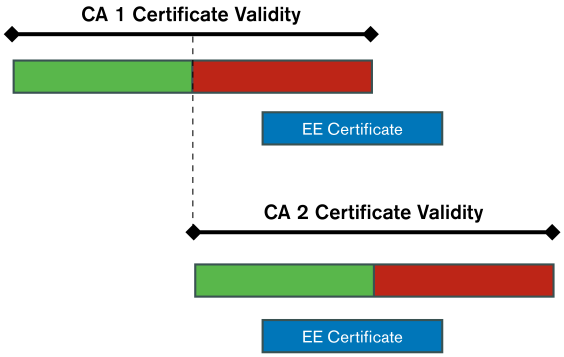


Figure 3 Automatic CA rollover

Say you have a CA certificate with a lifetime of two years. Now, one year and a day has gone by after the creation of the CA certificate. Given a request for an end-entity (EE) certificate with a

lifetime of one year (and your belief in the layer-based certificate validity validation model), you are in a bit of trouble: your CA certificate is still valid, but you can not accomodate the request, as the end-entity certificate would not be valid for a whole year.

OpenXPKI offers the automatic CA rollover feature, where more than one valid CA certificate can be in operation at any time. The PKI then decides at the time of the request which CA certificate to use for issuing the end-entity certificate. Hence, setting up a new CA certificate can be done at any time and does not need a complete hotswapping re-deployment.

Hardware Security Modules

OpenXPKI has support for some well-known Hardware Security Modules (HSMs), such as the nCipher nShield or the Chrysalis-ITS Luna CA. Hardware Security Modules are pretty interesting pieces of hardware – they provide a secure external storage for cryptographic keys and can perform the cryptographic operations in a protected environment. Think of it as a giant smartcard, if you like (though, some HSMs actually use smartcards for authentication as well, so you'd rather have to think of it as a smartcard with a smartcard slot).

Unluckily, HSMs are not something you add to your geek hardware collection at christmas – actually they are quite expensive. This is why we are looking into a pretty interesting solution which provides adequate security as well and is much more cost-effective.

If HSMs are out of your reach, we provide the interesting possibility to split the password for your encrypted software key into pieces using Shamir's secret splitting algorithm. In that way, you can still use the dual control principle to secure access to your CA key without dedicated hardware.

Self-Service Smartcard Solution

Imagine you have a batch of several thousand smartcards lying around your office for your company's employees. Would you rather generate and install the certificates for all of them or wouldn't it be nice to just give them out and point people to a website where they could do these initial steps themself?

Yes, we would have guessed so. This is where our self-service smartcard personalization application comes in. It offers the possibility to automatically create a key and a corresponding certificate request for the CA. The CA then signs the request and within a few seconds, the certificate is returned to the user and is automatically installed on the user's smartcard. For all of this, the user just needs a browser (well, OK, it needs to be Internet Explorer) and a few clicks. The necessary data which is to be included in the certificate can be retrieved from an LDAP directory, so that user interaction is kept to a minimum.

OpenXPKI

Cynops
network security engineering

Hacking OpenXPKI

Hacking OpenXPKI is actively encouraged by the current developers. We are always curious as to what ideas can be realized using our current infrastructure. Replacing the cryptographic layer should be pretty easy to do and we would definitely love to see something else than the usual OpenSSL-based implementation – maybe using Mozilla's Network Security Services (NSS) library or even something completely different.

Further ideas to be developed in the future include integration with management systems such as Tivoli or Nagios, clustering mechanisms to support the issuance of more than 500.000 certificates per day. One particularly interesting idea is to support CMC (the Certificate Management protocol using CMS) over COM, as this could be used to seamlessly replace a Microsoft CA.

If you are interested in some of these ideas or have your own thing that you like to work on, talk to us on the mailing list and we will try to provide you with the needed support in starting your development.

Contact

If you want to read more about the project, please see our websites at

- http://www.openxpki.org (main project website) or
- http://www.sf.net/projects/openxpki/ (Sourceforge development site).

There you can download the source, read more documentation or submit bug reports.

In addition we are always interested in talking to people interested in or using our software. Project communication mostly takes place on the mailing lists, which are at

- openxpki-users@lists.sourceforge.net (end-user support and discussion) and
- openxpki-devel@lists.sourceforge.net (developer discussion)

If you are curious about the current progress, you can also track our Subversion checkins using the openxpki-svn mailing list.

If all of this looks interesting to you, but you are unsure about whether you can shoulder the installation yourself, you need a PKI concept first or need to do some custom development, commercial support is available – take a look at http://www.openxpki.org/support/commercial.html for your options.