Scalable Policy Driven and General Purpose Public Key Infrastructure (PKI)

Vishwa Prasad[1] vmp@attlabs.att.com
Sreenivasa Potakamuri[2] psrao@attlabs.att.com
Michael Ahern[2] Michael@attlabs.att.com

Michah Lerner[1] Michah@attlabs.att.com
Igor Balabine[2] igorb@attlabs.att.com
Partha Dutta[2] pdd@attlabs.att.com

AT&T Labs
[1]Lincroft New Jersey
[2]San Jose, California

ABSTRACT

This paper describes a flexible and general purpose PKI platform providing an easily interoperable security infrastructure. Developed at AT&T Labs, the architecture is part of the UCAID/Internet2 efforts in PKI and scalable security. The architecture can host multiple certificate authorities (CAs) from different vendors in a uniform and scalable manner. This facilitates scalable operation with third-party CA systems. It acts as a CA distributor driven by uniform enrollment procedures based on vendor independent PKI policies. The design of seamless integration facilitates easy integration with third party CA services such as Verisign. The architecture adapts software components into a framework for secure, authenticated IP services over the open Internet or within internal intranets. Policy descriptions, written in XML, support explicit controls upon certificate sources and contents. These XML-encoded policies define issuance and acceptance of X.509v3 certificates from multiple CAs supporting the "obligations and warrantees … even if the policy is neither recorded anywhere nor referenced in the certificate" [1][2]. The PKI component has been developed within a general middleware platform [6].

Keywords

PKI, CA, policy, trust, XML, account, naming, grammar, X509, PKCS, CRML, OCSP, CRL, authentication, rights, revocation, certificate.

1. INTRODUCTION

The convergence and growth of communications enables interactions among multiple carriers, business and users at global through local scale. *Trust* – rather than mere *technology* – solidifies these activities. Indeed, trust has been the basis for commercial and social relationships through the ages. Communications-based systems introduce new challenges through an "electronic persona" freed from the traditional constraints of unchangeable identities, once physically present at concrete locations. Instead, communications-based systems facilitate flexible identities, mobile both in destination and in origin. Roaming over foreign networks – both through a communications-based presence as well as through physical mobility – draws upon new notions of trust. Sustaining this trust means that users, service providers, merchants and businesses may adapt their traditional methods of managing risks, identities and rights. The benefits of standardized methods for such activities are tangible yet complex.

PKI and digital certificates are emerging as one of the foundation technologies in the emerging online economy. These are a set of security services leveraging public-key cryptography and X.509 certificates through protocols supporting strong authentication, data encryption, digital signatures and access control functions in a networked environment. These functions are increasingly gaining momentum for applications in the Extranet, Intranet, and VPN areas, as the Internet is becoming a primary media for communications and electronic commerce. Currently this technology is being used in communication between Web browsers and Servers through the SSL protocol, secure email exchanges with digital signatures through the S/MIME protocol, and downloading of trusted and signed application code to run in users' browsers.

Yet, the deployment of PKI in the existing form introduces new problems and issues. Infrastructure for wide deployment of PKI is far from ubiquitous. Standards are still evolving. Vendor's PKI implementations may not always support PKI in a standard and interoperable fashion. As the application needs for PKI support grow, so do the incentives for a general-purpose scalable and interoperable PKI solution.

2. Current approach

This section describes a general view of the PKI platforms available from different vendors. Individual solutions from different vendors differ in some aspects from the functionality and implementation standpoint, not discussed here. In general, a PKI solution contains the following:

- A certificate authority (CA), which can issue and revoke a certificate
- A registration authority (RA), which handles identity verification in order for a certificate to be issued by the CA to a user. The RA also issues a certificate request on behalf of the user following identity verification, among other tasks. The RA is often considered an optional PKI component
- Directories that serve both as the repositories for certificates issued, and also for publication of certificate revocation lists (CRLs)
- Mechanisms to establish hierarchical and cross-certification trust relationships among CAs
- Policies that govern the operation and use of the PKI
- Software development kits and application program interfaces to PKI-enabled applications

Figure 1 shows a general high-level view of data flow in a PKI framework, with the following certificate flow:

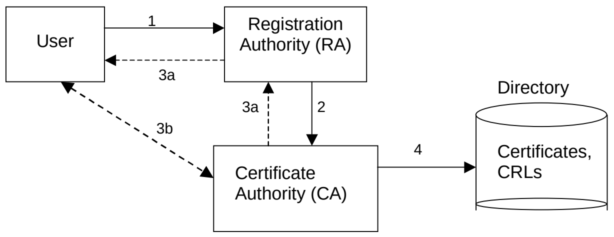

Figure 1: PKI Structure

1. User makes a request to the RA for certificate issuance/revocation
2. RA, upon verifying the user's identity, has the user issue a certificate request to the CA

3a. CA then either issues the certificate to the user or sends him a PIN and a directory location. The content of the issued certificate adheres to the policy of the CA. In the case of revocation the CA revokes the certificate

3b. If the CA issued a PIN and a directory location, the user uses these to pick up his certificate later

4. The CA publishes the certificate CRLs to a directory. Publication is periodic at well-defined intervals

2.1 Limitations of the Current Approach

This section addresses some of the fundamental issues surrounding the currently available PKI solutions mainly from a service provider standpoint. As the value of electronic commerce on the Internet grows, these challenges will become even more apparent.

2.1.1 Interoperability and Manageability

There are multiple PKI vendors today. Deploying multiple solutions will result in the operation of multiple disjointed PKI environments, each with its own policies, management utilities and trust management practices. Interoperability standards between the various PKI components are just in their early stages of definition. This locks the service operator into a particular PKI vendor, and requires a significant amount of customization work with each of the different PKI components to meet the service/enterprise specific needs. Complexities increase if a service provider hosts an existing PKI infrastructure.

2.1.2 Naming Conventions

Directory design affects certificate naming schema and management. Standards require the subject name equal the distinguished name (DN) in the directory, and this binding of certificates to DNs challenges the manageability of PKI. An organization's issued certificates may require revocation and reissue when the organizational structure changes. This inflexibility of the naming convention presents challenges to the hosting of PKI services for multiple organizations each with its own directory convention. This is a major issue in user authentication where the user presents a certificate issued by a third party CA such as Verisign.

2.1.3 Certificates

Each certificate has basic certificate fields and extension fields. The X.509v3 specification also enables organizations to define their own extension fields and encode information specific to their needs in a certificate. Although these extensions may reference policies of the CA, actual policy enforcement requires additional infrastructure. Currently, interoperability standards for certificate requests, revocation, and validation operations are still emerging. As a result, based on each organization's needs and dependent on its Certificate Authority (vendor dependence), a significant amount of custom work is required between the Registration Authority (RA) and the Certificate Authority (CA) to meet a particular organization's requirements. This work is very specific for each PKI vendor.

2.1.4 Validation

The cryptographic integrity of a signed document is typically insufficient grounds to trust a signed certificate for a given use. Compromise of the private key, for example, invalidates basic trust assumptions and revocation of the affected certificate is often appropriate under such conditions. Certificate revocation uses protocols that identify revoked certificates by placing them onto a certificate revocation list (CRL).

The CA publishes CRLs in a well-known location, yet the availability of such information depends on the frequency of publication as well as network connectivity. Consequently, the most recently revoked certificates may not appear in the CRLs. In this situation, a PKI application may potentially accept an invalid certificate. Suitable protocols reduce this risk. This may adversely affect performance through mandatory delay, as well as the costs associated with consulting multiple CRL distribution points.

2.1.5 Trust Models

Policies regulate the specific trust relationships based on the needs of an enterprise. Various legal and technical methods reinforce the underlying models. Trust may draw upon CA-issued procedural documents such as formal certificate practice statements (CPS) that describe mandatory practices for issuance of certificates and their subsequent use. Formal security models [8] describe trusts with mathematical precision, although the use of such models is often complex and specialized.

The main trust relationships in PKI are *certificate hierarchy*, and *cross-certification*. In the case of the certificate hierarchy model, it involves an agreement on policy with concomitant dependence upon the Root CA. If the hierarchy is deep, there are overhead and trust dilution in traversing the certificate path for validating a certificate. In addition, cascading failures in the trust chains occur upon the compromise of either the Root CA, or any of the intermediate CAs in the chain.

In cross-certification, each CA must again be comfortable with the security policies and practices of each other. In many cases, the representatives of each CA organization sign legal documents that specify security policies in both domains, and define specific liabilities or limitations. Complexity and financial cost increase with greater numbers of trust relationships between the CAs. The management and risk assessment become expensive, not to mention the associated legalese between the multiple organizations in establishing the cross-certification trust.

Enforcement of security policy may also limit the scalability. In particular, certificate path processing and other attribute-based techniques add substantial complexity. Path processing mechanisms derive success or failure of a validation through domain-specific policy sets and critical key extensions; these are neither standardized nor widely interoperable between domains.

2.1.6 Scalability

In contrast to the above problems workable, real-world solutions should be simple and affordable. A generalized and scalable PKI infrastructure is one approach to wider deployment of PKI solutions. This facilitates an open environment composed of multiple CAs from different vendors in a uniform and interoperable way. Such a platform will enhance the ability of service providers to host PKI services for organizations, as well as integrate existing PKI infrastructures.

3. Heterogeneous Approach

We discuss here our design approach in addressing some of the limitations mentioned in the previous sections mainly from a service provider's perspective. Our emphasis is upon policies, and in general PKI. As part of this effort, we developed a flexible and general purpose PKI platform, which can host multiple CAs from different vendors in a uniform and scalable manner. It acts as a CA distributor driven by uniform enrollment procedures based on vendor independent PKI policies. It is also designed to seamlessly integrate third party CA services such as Verisign. The proposed PKI infrastructure has been built as part of the platform code-named GeoPlex – a concept Advanced Network Service Platforms providing a core set of authentication, accounting, security, access control, and user registry functions that support diverse IP-based services, and allow corporations and providers to create new services rapidly.

Clients obtain certificates through signed PKCS10 requests originating from the client browser, with required fields and values provided to the client by the platform infrastructure. This facilitates the client's request for certificates that comply with platform policy, yet it does not constrain any third-party requests that the client issues independently. Certificates of either third-party or platform-supported CAs are suitable for authentication to valid accounts.

3.1 Principles of Open PKI

Based on the above assumptions, we enforce four general principles in support of the open PKI:

Vendor neutrality: The infrastructure permits all legal actions of internally hosted or externally accessed certificate authorities. This ensures a "level playing field" in which any entity may establish a certificate authority, as well as create services that require certificates. The subscribers and providers of middleware services may utilize all certificates and CAs without platform-imposed constraints.

Select trusted entities: The trusted CAs define the acceptable sources of certificates that are eligible for enrollment; eligibility relies upon administrative controls over accounts and sub accounts. There is no *a priori* restriction upon the trustable CAs.

Select entity role: The selection of the issuing CAs and certificate content enables platform mediation of actions. Platform-mediated issuance of certificates constrains the mandatory or forbidden content of requested certificates, as well as the authorities that may issue certificates on the behalf of the network. This enables standardized services that leverage multiple CAs through uniform content.

Enhanced Usage: The middleware or service may grant any local privilege to certificate holders. The granting and exercise of privileges are under the control of the platform or account administrator.

3.2 Benefits of these Principles:

Mobility: Mobility between PKI providers – regardless of the certificate interface or protocol issues – with continued use of existing X.509 certificates. Vendor independence prevents "legacy lock-in" with concomitant substandard service or excessive price. Lock-in occurs when a customer is compelled to keep using a provider simply due to the costs of changing to a new provider.

Independence: PKI implementation is independent of any particular CA vendor. The platform is able to host multiple CAs from different vendors. The platform provides uniform enrollment and revocation procedures for the multiple CA vendors it hosts. The PKI places minimal requirements on the CA products from different vendors for hosting on this platform

Management: Certificate management ensures that all users have ample notification of any potential problems with their certificates. This includes notification of impending expiration or revocation of a client's certificates. These management services are essential to ensure uninterrupted availability despite dependence on external authorities.

Issuance Policies: Certificate issuance adheres to policies that define the permitted providers, as well as the content of X.509 certificates. The PKI integrates with the customer information profile thereby providing uniform policy definition and enforcement.

Usage Policies: Allows multiple certificates per user or per account, as well as use of the same certificate for multiple accounts. The certificates can be issued either by the platform hosted CAs, or a third party CA service. This property eases formation of Intranet, Extranet and VPN spaces.

Policy Content: The administrator through definition of preferred policies determines certificate contents. For example, the customer rather than the CA vendor defines the naming structure.

Preferences: The administrator through definition of preferred policies determines the maximum permissible and minimum acceptable parameters for service-specific extensions.

Innovation: A PKI provider through improved software or hardware may modify their internal operations freely without requiring any customer changes. The certificate infrastructure accommodates the "front end" changes.

Compliance: A platform-issuing CA obtains well-defined and accountable pre-screened applicants. Middleware-mediated requests are guaranteed to be "in compliance" with the policies and procedures of the user's community or organization.

3.3 Certificate Policy Defines Content

The open PKI defines the permissible certificate types, either for issuance on a client's behalf, or acceptance for authentication purposes. We find substantial value in a formal representation of these certificates. We therefore developed the Credential Management Policy Information (CMPI) block. This describes the essential elements of policy through a grammar supporting the exact interpretation of CMPI. Structuring through reuse of symbolic labels provides consistent reference to policy elements, and block-structured inheritance defines a structure of values and constraints. Database query may, in the future, also retrieve CMPI blocks through skeletal patterns that match a database of policies. This provides an interoperable alternative to policy definitions written in English prose, spreadsheets, policy description languages [10] or resolution theorem provers [11].

The system reported here encodes the policy into the CMPI XML representation, and associates these blocks with the accounts of the domain structure. Future work may generate the policy blocks directly from queries against policy description through the XML Query Language described by W3C query data model (http://www.w3.org/TR/query-datamodel/), and automatically attach these blocks to the domain structure. The language supports value assignment as well as various transforms and conformance tests. It contains the following elements:

- *tags* that identify statements
- *name and type* of items
- *permissible values* (or ranges of values)
- *function* definitions
- *source* information, for example display and input from a tagged field on the user's browser, or a reference to the domain's database of account information
- *transformation* functions that modify values; these can be JavaScript, VB or the URI of a resource containing the function
- *filter* functions that validate values
- *destination* information, specifically the location in the issued certificate

In CMPI, as with other languages, terminal symbols [9] may denote variables of the generated language. For example, the sequence `CN=$CA_FN$ $CA_LN$` indicates the `commonName` (CN) is formed from variables that store values of the subject's first and last names (`CA_FN` and `CA_LN`). When requesting a certificate we bind the client's values to these variables, and emit a suitable PKCS10 request.

This approach constrains the contents of permissible X.509 certificates through the XML-encoded CMPI. An initial grammar production defines specific types of certificates. Subsequent productions describe the mandatory and optional fields and values of each certificate type.

3.4 Interaction with Existing PKI

Certificate registration policy is managed at the individual account, user, and service level through a policy hierarchy. All accounts, users, and services exist as sub-entities to a master root account. Accounts form a tree structure among the various user, account, and service entities. Any sub-account may have multiple sub-accounts, users and service entities.

The system directory stores the specific certificate policy in a hierarchical manner, beginning with the master root account and descending to sub-entities. Every child object inherits the policy of its parent object unless administrative overrides have previously granted the child the ability to define specific aspects of the policy. The account tree defines all atomic attributes, and the unique path from the entity to the root defines a complete certificate policy. For example, if an entity's validity period is unspecified, the credential manager uses the first ancestor occurrence that defines the validity period. Entities inherit the root account policy attributes, as well as refinements of the entities' hierarchical predecessors in the account tree. These refinements include both limitations and extensions.

Our credential management structure achieves value through a standardized XML structure and an inheritance object model. Large and small groups of entities are organized as branches within the object tree, with policy management via inheritance to individual objects. Tools include a basic "administrator only" editor of policy attribute values including but not limited to the certificate management policy XML document. API extenders include a parser object that creates a data structure from which new certificates are constructed and existing certificates validated against an administrative policy.

4. Architecture

The high-level block diagram of Figure 2 outlines the architecture.

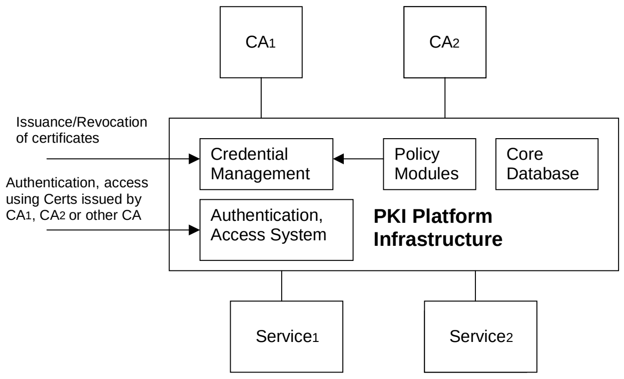
CA1
CA2
Issuance/Revocation of certificates
Authentication, access using Certs issued by CA1, CA2 or other CA
Credential Management
Policy Modules
Core Database
Authentication, Access System
PKI Platform Infrastructure
Service1
Service2

Figure 2: PKI platform Infrastructure

The above diagram shows multiple certificate authorities – CA_1, CA_2 and services $Service_1$ and $Service_2$ – hosted by the platform. A user via a browser requests the issuance of a certificate from the Credential Management system, or is redirected to it by the Registration system (the registration system is responsible for verification of identity). The Core database in the above diagram contains the user account information and any associated PKI policies for the user obtained by the registration process. The credential system issues the user a temporary password for certificate issuance. The user directs the browser to the platform and authenticates with the temporary password with privileges limited to certificate enrollment.

Based on the policies of the user's account and the particular issuing CA, the credential system performs the certificate enrollment procedure. The user submits the necessary data by filling out HTML forms provided by the enrollment service. A user may change some of the fields based on the enforced policy before submitting a certificate request. Upon submission of the certificate generation request, the credential management system sends the certificate request in PKCS10 format to the relevant Certificate Authority. Additional fields, not part of the PKCS10 request, are sent in CRMF format to the Certificate Authority. The CA then returns the signed certificate in PKCS7 format to the credential management system, which then returns the certificate to the user's browser.

The user can now register the new certificate with the platform, and thereby obtain the privilege to authenticate to the platform later using this certificate as a credential. The certificate registration process also enables authentication through certificates issued by multiple issuing authorities, without going through the explicit cross-certification process. Policy defines both the number of certificates per user and the trusted issuing authorities. For the case where the user requires use of an existing third-party certificate as a credential to access services hosted on the platform, he registers the existing certificate by logging on to the platform using a temporary password, and then carries out the registration steps discussed above. The credential system only allows registration if the user's policy trusts the third-party issuing authority.

Efficient authentication and revocation utilize two thumbprints, both computed during certificate registration. Certificate thumbprints, stored in the Core database, make platform authentication independent from the certificate contents. A *full certificate thumbprint* supports authentication, and a *partial thumbprint* enables revocation using the Issuer name and Serial number fields. The platform reserves the right to revoke the authentication privilege of any certificate.

4.1.1 Policy Module

This section describes storage of PKI policies on a generic and per user/account basis for different types of certificate authorities, through repositories of Credential Management Policy Information (CMPI) blocks in the core database. The credential management system uses these definitions as policies for certificate issuance, registration, revocation and supplying values as defined in X.509 for certificate fields and extensions. The account structure is hierarchical, where each user/account contains the PKI policies defined in an XML template.

The root account contains the default PKI policies for the account hierarchy. Each sub-account or user account inherits the parent's PKI policy, unless it has defined its own policy. In this case, the defined policy overrides the parent's definition. Each CA hosted by the platform also has an associated CMPI block. The CA's CMPI block can also restrict the permissible relevant attributes.

Figure 3 shows an example account hierarchy with the CMPI blocks, including the MSCA or Verisign certificates

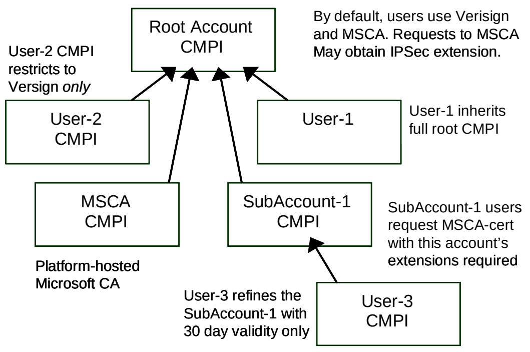
Root Account CMPI
By default, users use Verisign and MSCA. Requests to MSCA May obtain IPSec extension.
User-2 CMPI restricts to Versign *only*
User-2 CMPI
User-1
User-1 inherits full root CMPI
MSCA CMPI
SubAccount-1 CMPI
SubAccount-1 users request MSCA-cert with this account's extensions required
Platform-hosted Microsoft CA
User-3 refines the SubAccount-1 with 30 day validity only
User-3 CMPI

Figure 3: Policy Hierarchy

and a platform-hosted MSCA implemented by a Microsoft certificate server. $User_1$ and $User_3$ can receive a certificate issued by the hosted MSCA based on the policy defined in CMPI block. $User_1$ and $User_2$ can also register a Verisign issued certificate with the platform because their policy allows it. This enables $User_1$ and $User_2$ to use a Verisign issued certificate to access services hosted by the platform.

4.1.2 Certificate Enrollment

We attain interoperable certificate enrollment by means of the CMPI block written as XML templates. These define generic PKI policies. The credential management system interprets the CMPI definitions and provides the user with appropriate pages during certificate enrollment. The credential management is also able to enforce policies at the granularity level of individual X.509 v3 certificate fields and extensions, based on the CMPI policy definitions. This allows per-field definition by policy, user preference or registration data. The X.509v3 extensions enable organizations to define their own extension fields and encode information specific to their needs in a certificate in an interoperable fashion independent of a CA. The CMPI denotation of figure 3 is:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<CMPI>
 <VERSION>1.00</VERSION>
 <CREDTYPE>
   CERTIFICATE
   <VERSION>1.00</VERSION>
   <TRUSTEDCAS>
     <HANDLE>verisign</HANDLE>
     <HANDLE>msca</HANDLE>
   </TRUSTEDCAS>
   <CERTINFO>
     <HANDLE>msca</HANDLE>
     <USETEMPLATES>
       <NAME REF="IPSEC"> ipsec </NAME>
     </USETEMPLATES>
   </CERTINFO>
   <USEHOSTEDCAS>
     <HANDLE>msca</HANDLE>
   </USEHOSTEDCAS>
   <MAXCERTS> 6 </MAXCERTS>
  </CREDTYPE>
</CMPI>
```

The above template is an example CMPI block definition for the Root account as shown in Figure 3. The tag `CERTINFO` defines the `msca` handle for a CA known as `msca`, and requires use of a template for that describes the configuration information.

The following example template is a CMPI policy definition for User$_2$ as defined in Figure 3. User$_2$ can only register a Verisign issued certificate; he cannot use the platform hosted CA:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<CMPI>
  <VERSION>1.00</VERSION>
  <CREDTYPE>
   CERTIFICATE
   <VERSION>1.00</VERSION>
   <TRUSTEDCAS>
     <HANDLE>verisign</HANDLE>
   </TRUSTEDCAS>
  </CREDTYPE>
</CMPI>
```

The platform PKI provides a robust mechanism that allows adding data for the standard certificate fields and arbitrary certificate extensions, specifying values of the fixed and hidden certificate fields, and presenting user-modifiable data items in the certificate enrollment form. For example, a certificate enrollment form item that puts user e-mail address in the `rsadsi-pkcs9-e-mail` certificate extension is formulate as follows:

```
<FIELD>
  <TAG LABEL=user_email DISPLAY_PROP=hidden>
  </TAG>
  <ATTRIBUTE>
    <SOURCE>
       <TAG_LABEL>@emailAddress </TAG_LABEL>
    </SOURCE>
    <TRANSFORM_FUNCTION>
      <FUNCTION_REF NAME=toUpperTruncate/>
      <PARAMETER>20</PARAMETER>
    </TRANSFORM_FUNCTION>
    <DESTINATION>
      <EXTENSION_OID CRITICAL=yes>
        1 2 840 113549 1 9 1
      </EXTENSION_OID>
    </DESTINATION>
  <ATTRIBUTE>
</FIELD>
```

This XML data object element specifies extraction of the extension from the `emailAddress` field in the account database. It declares that the value neither displayed nor modifiable. The value from the database field is converted to uppercase and truncated to 20 characters by means of an external `toUpperTruncate` JavaScript function. The resulting data is placed in certificate as the critical extension OID 1 2 840 113549 1 9 1 (older "`rsadsi-pkcs9-e-mail`" extension). We note that the W3C standards body has just finalized the XSL proposal, and we are adapting to these XSL conventions.

4.1.3 CA integration

The platform is independent of any particular CA service implementation. The architecture for the integration of a CA to the platform has been described here. Most of the certificate authorities have programmable entry, policy and exit modules. All CA service vendor-related dependencies are handled in these modules.

The platform interfaces with the CA using the PKCS10 format to request a certificate. The platform expects delivery of the issued certificate in the PKCS7 format. The platform uses the CMP format to revoke certificates issued by the platform hosted CA. Any additional data required for certificate issuance by the local policy which cannot be delivered as a part of the PKCS10 request, is provided in the CRMF format. The CA service provides native policy modules to implement the policies received from the platform PKI.

4.1.4 Expiration Management

The platform also facilitates expiration management for the CAs. This mechanism can redirect issuance of a certificate for users to a new CA, without having to update the account hierarchy. Certificate expiration management utilizes the feature of multiple certificates for a user account.

An authentication-time notification warns the user of upcoming certificate expirations, and instructs him to obtain another certificate. The user may obtain a replacement certificate from a platform hosted CA using the current unimpaired certificate, or the user may request a certificate from a platform-trusted third party. The user's presently valid certificate provides a credential for registering the new certificate. This provides seamless continuation of a service for the user without going through customer service or other out of band mechanisms.

4.1.5 Revocation

Revocation of a certificate occurs as the immediate effect of un-registration of the certificate by removal of certificate thumbprint from the core database. Without the presence of the certificate thumbprint, a user will not be able to authenticate to the platform. Un-registration of a

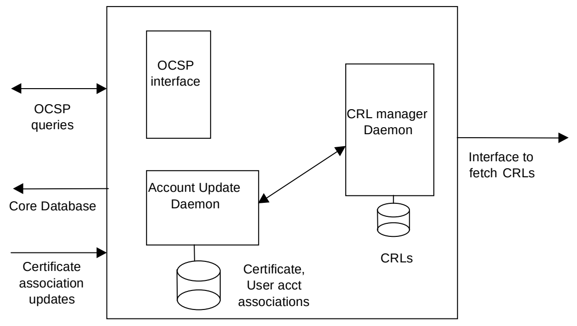

Figure 4: Certificate Revocation

certificate is the only function that performed for third party issued certificates. For platform hosted CA, a revocation request is also propagated to the CA simultaneously. Third-party CRLs are mapped to a local store of certificate thumbprints and accounts. The Account Update daemon deletes the certificate thumbprints from the core, thus blocking further authentication

The CRL service management also includes an OCSP interface for certificate status queries by any service. This is used by the Credential Management service to check the CRL before registering a certificate. Appropriate design abstractions ensure the future utility of these mechanisms. The CRL Management Service can easily adapt as OCSP services provide ubiquitous support for all CA services, as well to invention of future status checking mechanisms.

5. Conclusions

In this paper, we have discussed the shortcomings of current PKI solutions. We believe resolution of these shortcomings is crucial to create a viable foundation for the new digital economy. Our solution provides a flexible, interoperable, scalable and robust PKI platform.

Currently, the first implementation of the platform runs on Solaris 2.6. We are in the process of hosting and integrating multiple CA vendors on this platform.

6. References

[1] RFC2459: Internet X.509 Public Key Infrastructure Certificate and CRL Profile. R. Housley, W. Ford, W. Polk, D. Solo. January 1999.

[2] TU-T Recommendation X.509: Information technology – Open systems interconnection – The directory: public-key and attribute certificate frameworks. March 2000 with amendments and corrigenda.

[3] RFC2511: Internet X.509 Certificate Request Message Format. M. Myers, C. Adams, D. Solo, D. Kemp. March 1999.

[4] RFC2527: Internet X.509 Public Key Infrastructure Certificate Policy and Certification Practices Framework. S. Chokhani, W. Ford. March 1999.

[5] RFC2560: X.509 Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP. Myers, R. Ankney, A. Malpani, S. Galperin, C. Adams. June 1999.

[6] M Lerner, *et. al.*, Middleware Networks: Concept, Design and Deployment of Internet Infrastructure. Kluwer Academic. April 2000.

[7] IETF *work in progress* DRAFT-IETF-MOSKOWITZ-CMPINTEROP-00.TXT.

[8] D. Gollman, Computer Security, John Wiley and Sons Ltd., West Sussex, England, 1999.

[9] A. Aho, *et. al.*, Compilers – Principles, Techniques and Tools. Addison-Wesley, 1988.

[10] A Herzberg, *et. al.*, *Access Control Meets Public Key Infrastructure, Or: Assigning Roles to Strangers*. http://www.hrl.il.ibm.com/TrustEstablishment/paper.asp and 2000 IEEE Symposium on Security and Privacy.

[11] M Blaze, *et. al.*, *The Role of Trust Management in Distributed Systems Security* in Secure Internet Programming – Security Issues for Mobile and Distributed Objects Lecture Notes in Computer Science 1603, Springer Verlag, 1998.

7. Appendix: Policy Storage Architecture

The PKI platform stores policy information. The root account contains the default PKI policies for the whole account hierarchy. Each sub-account or user account inherits its parent's PKI policy, unless an administrator attaches PKI policies that supercede its parent definition. Each certificate authority (CA) hosted by the platform also has an associated CMPI block. The following is the CMPI for the configuration described above in Figure 3:

Root Account
```
<?xml version="1.0" encoding="ISO-8859-1"?>
<CMPI>
 <VERSION>1.00</VERSION>
 <CREDTYPE>
  CERTIFICATE
   <VERSION>1.00</VERSION>
   <TRUSTEDCAS>
     <HANDLE>verisign</HANDLE>
     <HANDLE>msca</HANDLE>
   </TRUSTEDCAS>
   <CERTINFO>
     <HANDLE>msca</HANDLE>
     <USETEMPLATES>
       <NAME REF="IPSEC">
         ipsec
       </NAME>
     </USETEMPLATES>   </CERTINFO>
   <USEHOSTEDCAS>
     <HANDLE>msca</HANDLE>
   </USEHOSTEDCAS>
   <MAXCERTS> 6 </MAXCERTS>
  </CREDTYPE>
</CMPI>
```

CMPI for SubAccount-1
```
<?xml version="1.0" encoding="ISO-8859-1"?>
<CMPI>
  <VERSION>1.00</VERSION>
  <CREDTYPE>
   CERTIFICATE
   <VERSION>1.00</VERSION>
   <TRUSTEDCAS>
     <HANDLE>msca</HANDLE>
   </TRUSTEDCAS>
  </CREDTYPE>
</CMPI>
```

CMPI for User-1
```
<?xml version="1.0" encoding="ISO-8859-1"?>
<CMPI>
```

```
  <VERSION>1.00</VERSION>
</CMPI>
```

CMPI for User-2

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<CMPI>
  <VERSION>1.00</VERSION>
  <CREDTYPE>
   CERTIFICATE
   <VERSION>1.00</VERSION>
   <TRUSTEDCAS>
     <HANDLE>verisign</HANDLE>
   </TRUSTEDCAS>
  </CREDTYPE>
</CMPI>
```

CMPI For User-3

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<CMPI>
 <VERSION>1.00</VERSION>
  <CREDTYPE>
   CERTIFICATE
   <VERSION>1.00</VERSION>
   <CERTINFO>
     <HANDLE>msca</HANDLE>
     <USETEMPLATES>
      <DEFAULT REF="USR_DEFAULTTEMPL"> CUSTOM3
      </DEFAULT>
      <NAME> CUSTOM3 </NAME>
      <NAME> IDENTITY </NAME>
      <UI>
      <FILTERCLASS>
            geo.CAPolicy.TSFilter
      </FILTERCLASS>
      <PAGE>
       1
       <HTMLDATA>
        <FILENAME>
          /opt/geoplex/config/certs.html
        </FILENAME>
       </HTMLDATA>
     </PAGE>
    </UI>
   </USETEMPLATES>
   <TEMPLATEDEFS>
    <TEMPLATE>
     <NAME REF="USR_CUSTOM3NAME">CUSTOM3</NAME>
     <DESC REF="USR_CUSTOM3DESC"> Custom for
        user3 - IPSEC for 3 months and comment
     </DESC>
     <TEMPLATEEXTEND>IPSEC</TEMPLATEEXTEND>
     <VALIDPERIOD REF="USR_CUSTOM3VP">129600
     </VALIDPERIOD>
     <NSCOMMENT>
       <VALUE REF="USR_CUSTOM3NSCMNT">
             User3-Type Your comment Here
       </VALUE>
     <ENABLE>Y</ENABLE>
     </NSCOMMENT>
     <UI>
      <PAGE>
        1
       <HTMLDATA>
        <FILENAME>
    /opt/geoplex/config/certs/html/tdpguser3a.html
        </FILENAME>
       </HTMLDATA>
      </PAGE>
     </UI>
    </TEMPLATE>
   </TEMPLATEDEFS>
  </CERTINFO>
 </CREDTYPE>
</CMPI>
```

For MSCA hosted on the Platform

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<CMPI>
 <VERSION>1.00</VERSION>
 <CREDTYPE>
  CERTIFICATE
 <VERSION>1.00</VERSION>
 <INTERFACECLASS>geo.CAInterface.CAIHttpImpl
 </INTERFACECLASS>
 <CAPOLICYCLASS>geo.CAPolicy.CPPImpl
 </CAPOLICYCLASS>
 <MAPTOCA>
  <NEWCAHANDLE> msca2 </NEWCAHANDLE>
   <POLICYMAPCLASS>
    geo.CAPolicy.MapPolicyFromMSCAToMsca2
   </POLICYMAPCLASS>
  </MAPTOCA>
  <LOGLEVEL>2</LOGLEVEL>
  <PKCS10URL>
http://msca.dev4.ipo.att.com/ca_poe/issuecert.asp
  </PKCS10URL>
  <SPKACURL>
http://msca.dev4.ipo.att.com/ca_poe/issuecert.asp
  </SPKACURL>
  <REVOKEURL>
    http://msca.dev4.ipo.att.com/ca_poe/revoke.asp
  </REVOKEURL>
  <CRLURL>
     http://msca.dev4.ipo.att.com/ca_poe/crl.asp
  </CRLURL>
  <CACERTURL>
    http://msca.dev4.ipo.att.com/ca/cacert.asp
  </CACERTURL>
  <ENTITYINFORMATION>
   <ATTRIBUTE REF="CA_FN"> First_Name
   </ATTRIBUTE>
   <ATTRIBUTE REF="CA_MN"> Middle_Name
   </ATTRIBUTE>
   <ATTRIBUTE REF="CA_LN">
      Last_Name </ATTRIBUTE>
   <ATTRIBUTE REF="CA_E"> Email </ATTRIBUTE>
   <ATTRIBUTE REF="CA_O">
      Affliation </ATTRIBUTE>
   <ATTRIBUTE REF="CA_OU">
     Organization_Unit </ATTRIBUTE>
   <ATTRIBUTE REF="CA_L"> City </ATTRIBUTE>
   <ATTRIBUTE REF="CA_ST"> Province </ATTRIBUTE>
   <ATTRIBUTE REF="CA_C"> Country </ATTRIBUTE>
   </ENTITYINFORMATION>
   <CERTINFO>
    <HANDLE>msca</HANDLE>
    <USETEMPLATES>
     <DEFAULT> IDENTITY </DEFAULT>
     <NAME> IDENTITY </NAME>
     <NAME> IPSEC </NAME>
    </USETEMPLATES>
    <TEMPLATEDEFS>
     <TEMPLATE>
      <NAME REF="CA_IDENTITYNAME">
       IDENTITY </NAME>
      <DESC REF="CA_IDENTITYDESC">
       User Identity Certificate</DESC>
      <SUBJECT>
        CN=$CA_FN$ $CA_MN$ $CA_LN$,OU=People,
        OU=$CA_OU$,O=$CA_O$,L=$CA_L$,S=$CA_
  ST$,C=$CA_C$,E=$CA_E$
</SUBJECT>
<STARTOFFSETIME>5</STARTOFFSETIME>
<VALIDPERIOD>525600</VALIDPERIOD>
 <EXTKEYUSAGE>
  <VALUE>CA</VALUE>
 </EXTKEYUSAGE>
</TEMPLATE>
<TEMPLATE>
 <NAME REF="CA_IPSECNAME"> IPSEC </NAME>
 <DESC REF="CA_IPSECDESC">
     User IPSEC Certificate</DESC>
 <SUBJECT>
   CN=$CA_FN$ $CA_MN$ $CA_LN$, OU=$CA_OU$,
   O=$CA_O$,L=$CA_L$, S=$CA_ST$, C=$CA_C$,
   E=$CA_E$
 </SUBJECT>
 <STARTOFFSETIME>10</STARTOFFSETIME>
 <VALIDPERIOD>525600</VALIDPERIOD>
 <EXTKEYUSAGE>
  <VALUE>CA</VALUE>
  <VALUE>IU</VALUE>
 </EXTKEYUSAGE>
</TEMPLATE>
<TEMPLATE>
 <NAME REF="CA_CUSTOMNAME">CUSTOM</NAME>
 <DESC REF="CA_CUSTOMEDESC">
   Custom – select allowable parameters
 </DESC>
 <SUBJECT REF="CA_CUSTOMSUBJ">
    CN=$CA_FN$ $CA_MN$ $CA_LN$,OU=$CA_OU$,
    O=$CA_O$,L=$CA_L$,S=$CA_ST$,
    C=$CA_C$,E=$CA_E$
  </SUBJECT>
  <STARTOFFSETIME REF="CA_CUSTOMOT">
   10 </STARTOFFSETIME>
  <VALIDPERIOD REF="CA_CUSTOMVP">
   129600</VALIDPERIOD>
  <KEYUSAGE>
   <VALUE REF="CA_CUSTOMKU1"></VALUE>
   <VALUE REF="CA_CUSTOMKU2"></VALUE>
   <CRITICAL REF="CA_CUSTOMKUC">
     F</CRITICAL>
  </KEYUSAGE>
  <EXTKEYUSAGE>
   <VALUE REF="CA_CUSTOMEKU1"></VALUE>
   <CRITICAL REF="CA_CUSTOMEKUC">
     F</CRITICAL>
  </EXTKEYUSAGE>
  <SUBJECTALTNAME>
   <RFC822NAME REF="CA_CUSTOMALTEMAIL">
   </RFC822NAME>
   <DNSNAME REF="CA_CUSTOMALTDNSNAME">
   </DNSNAME>
   <URI REF="CA_CUSTOMALTURI"></URI>
   <OID REF="CA_CUSTOMALTOID"></OID>
   <CRITICAL REF="CA_CUSTOMALTC">
   </CRITICAL>
  </SUBJECTALTNAME>
  <NETSCAPETYPE>
```

```
         <VALUE REF="CA_CUSTOMNS1"></VALUE>
          <CRITICAL REF="CA_CUSTOMNSC"> F
          </CRITICAL>
        </NETSCAPETYPE>
        <CUSTOMEXTENSIONS>
         <EXTENSION>
          <OID REF="CA_CEXT1O"> 1.2.3.4</OID>
          <TYPE REF="CA_CEXT1T">STRING</TYPE>
          <VALUE REF="CA_CEXT1V"> Extension  string
          </VALUE>
          <DATE>
           <DATEFORMAT REF="CA_CEXT1DF">
              dd/mm/yyyy hh:mm:ss aa zz
           </DATEFORMAT>
           <DATEFORMATDESC REF="CA_CEXT1DFD">
             Format for date is dd/mm/yyyy hh:mm:ss
               AM|PM EST|CST|PST|GMT[+|-]hh:mm
           </DATEFORMATDESC>
          </DATE>
         <CRITICAL REF="CA_CEXT1C">F</CRITICAL>
         </EXTENSION>
        </CUSTOMEXTENSIONS>
        <NSCOMMENT>
        <VALUE REF="CA_CUSTOMNSCMNT">
          Type Your Comment Here
        </VALUE>
        <ENABLE>Y</ENABLE>
        </NSCOMMENT>
        <UI>
         <FILTERCLASS> geo.CAPolicy.TDFilter
         </FILTERCLASS>
          <PAGE> 1
           <HTMLDATA>
            <FILENAME>
         /opt/geoplex/config/certs/html/tdpg1.html
            </FILENAME>
           </HTMLDATA>
          </PAGE>
          <PAGE> 2
         <HTMLDATA>
          <FILENAME>
         /opt/geoplex/config/certs/html/tdpg2.html
         </FILENAME>
        </HTMLDATA>
       </PAGE>
       <PAGE> 3
        <HTMLDATA>
         <FILENAME>
          /opt/geoplex/config/certs/html/tdpg3.
         </FILENAME>
        </HTMLDATA>
       </PAGE>
      </UI>
    </TEMPLATEDEFS>
   </CERTINFO>
   <MAXCERTS> 6 </MAXCERTS>
  </CREDTYPE>
</CMPI>
```