Public Key Infrastructures for the Financial Services Industry

William E. Burr

Kathy L. Lyons-Burke

National Institute of Standards and Technology

August 31, 1999

PUBLIC KEY INFRASTRUCTURE (PKI) BASICS

A Public Key Infrastructure (PKI) allows individuals to know who they are communicating with electronically and enables confidential communications across an open network. A PKI also enables the application of a technically non-repudiable digital signature to specific information. This means that there is only one key and one message that could have been used together to generate a specific digital signature. Therefore, if the signatory did not share his digital key with another, he must have signed the information. In addition, since the information that was signed was input to the digital signature process, the information could not have been modified and is exactly what the individual signed.

These PKI capabilities allow a financial institution to replace typically paper based business processes with electronic processes. They can also enable an institution's participation in all aspects of electronic commerce.

Security Services

A PKI relies on cryptography to help accomplish the following security services:

- *Authentication*: the ability to corroborate the identity of a party that originated particular data, or participates in a protocol;
- *Confidentiality*: the ability to ensure that unauthorized individuals are not able to access protected data;
- *Access Control*: the prevention of unauthorized use or access to a resource;
- *Non-repudiation*: the ability to prove to a fair, neutral second party that a first party performed or is responsible for some action. This is sometimes called "technical non-repudiation" to distinguish it from legal concepts that allow one, for example, to repudiate an action performed under the compulsion of some threat; and
- *Integrity*: the ability to know that data has not been altered in any way.

Cryptography

Cryptographic algorithms perform mathematical transformations on data. These transformations have valuable security properties. *Encryption* algorithms mathematically transform *plaintext* data under control of a *key* (a number, typically from 40 to 1024 bits) into *ciphertext*, so that intruders who steal the ciphertext learn nothing about its meaning without knowing the right key. *Decryption* reverses encryption under control of the right key, and turns ciphertext back into plaintext.

For hundreds of years, cryptographic algorithms all used a single key to both encrypt and decrypt data. The parties encrypting and decrypting messages had to have a common key that was not shared with others, known as a secret key. These *symmetric key* algorithms can be made to work at high speed, and are widely used for "bulk encryption."

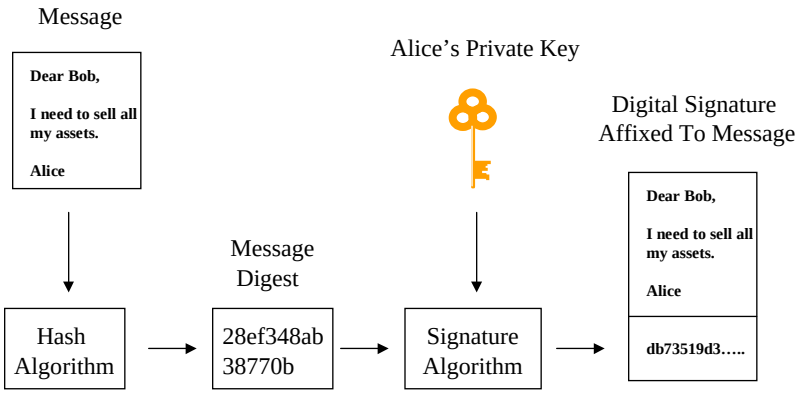
Public Key Cryptography

The concept of *public key* cryptography was introduced about 30 years ago. Public key cryptography uses public/private key-pairs. The public key can be made known to whoever needs it, but the private key is kept secret. The public key is mathematically related to the private key, but the private key cannot be computed from the public key. Key-pairs are used with specific cryptographic algorithms for digital signing, signature verification, encryption and decryption operations.

However, public key algorithms are relatively slow, therefore we don't usually sign or encrypt a long message directly with a public key algorithm. Rather a long message is encrypted with a symmetric key algorithm, and the encryption key is then encrypted with at public key algorithm. For digital signatures, we first generate a relatively small message-digest (typically a 160-bit value) from a message of any size, using a cryptographic *hashing* algorithm. The hashing algorithms are relatively fast and designed so that it is infeasible to find a second message with the same digest (one would have to hash about 1,000,000,000,000,000,000,000,000 different messages on average to find two with the same 160-bit message digest). Therefore the message digest can substitute for the message in the signature.

Figure 1 illustrates how a digital signature is generated using a commonly used algorithm, called RSA [RIV 78]. First, a signatory, Alice, applies a cryptographic hashing algorithm to a message (a message in this context can be any digital file) to generate a message digest. Alice (see box Alice and Bob), then applies her private key to encrypt the message digest, producing a digital signature unique to the message and Alice's private signing key. This digital signature can then be appended to the message.

Figure 1: RSA Digital Signature Creation

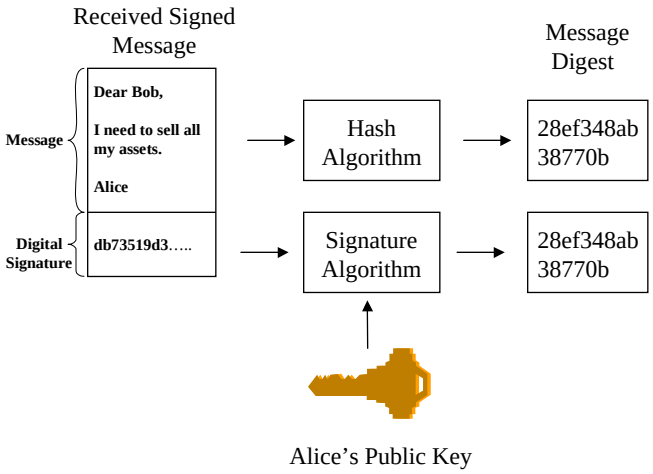


Alice and Bob

Because it is difficult to follow discussions written in abstract terms such as "relying parties," and "signatories," we have chosen to follow the convention common in cryptographic papers of using "Alice" and "Bob." In this paper Alice is always the signatory, who uses her private key, while Bob is always a relying party, someone who relies on Alice's public key. Alice and Bob are not necessarily people, but may be web servers, routers, or processes in a computer, that either have key-pairs or rely on them.

Figure 2 shows the digital signature verification process that a relying party, Bob, follows to verify the signature. Bob first applies the hash algorithm to the received message to obtain the message digest. He then applies Alice's public key to the digital signature to decrypt the message digest used to create the digital signature. Bob then compares the two message digests. If the message digests are identical, the digital signature is verified. If Alice's public key is known, the digital signature provides both integrity and non-repudiation services. Moreover, it's easy to design a protocol where Alice digitally signs a random challenge, to provide authentication. The same RSA algorithm or other algorithms can be used to encrypt a message to provide confidentiality (although separate keys for digital signatures and encryption are strongly recommended).

Figure 2: RSA Digital Signature Verification



Key Management

Users must manage their keys. How users generate and store their private keys is significant. A PKI depends on the quality and secrecy of the private keys. Alice's private signature key is solely for the purpose of generating Alice's signature. There is no reason that another party ever

needs to know Alice's private signature key. Therefore, signature keys are best generated under the control of the key's owner. Other (trusted) parties may know private encryption keys, to protect against loss of encrypted data. In addition, old signature keys are usually destroyed to prevent post-dated signatures, while old encryption keys must be kept as long as files are encrypted with them. For these reasons, the best practice is to use different keys for encryption and signatures.

In many cases, private keys are stored in an encrypted file on the user's computer. To use the private key, the user decrypts it using a password or phrase. Alternatively, private keys may be kept on a hardware cryptographic token, typically a smartcard, with signature operations performed on the token. A PIN or password is typically required to activate the key for signing. This promotes mobility (at least between machines with token readers) and may be more secure. However, tokens and readers must be purchased, and this will increase costs. See the box " Private Key Protection" for more information.

Private Key Protection

The strongest line of defense for private keys is encryption. Keys are stored encrypted under a key that is generated from a password or phrase. An attacker, even one with vast resources, who succeeds in obtaining a "well encrypted" private key will generally fail to decrypt the key, unless the attacker can guess the password or phrase. Well-chosen pass phrases are vital!

When an encrypted private key is stored on a disk in a general-purpose computer, it may be easy to steal the encrypted password. However, the real vulnerability exists when the pass-phrase is entered and the key is decrypted for use in a software implementation. At this point, a "Trojan Horse" program active on the system can steal either the decrypted key or the pass-phrase.

Hardware tokens that hold the private key and perform signing operations on the token offer additional protection. The tokens are often implemented as smartcards. Smartcards can be designed to prevent the key from ever leaving the physical environment of the token. Smartcards typically retain the private keys in encrypted form and only decrypt the keys to perform signing operations. Private keys can often be extracted from stolen smartcard tokens in a laboratory setting, but the attacker only gets an encrypted key.

Cryptographic tokens for applications where the attacker is likely to be a legitimate token user, such as cash cards that hold some monetary value or tokens for satellite TV use, are notoriously difficult to protect. This is because the legitimate token user can activate the key, but is not supposed to know what the key's actual value is. If the user knows the key value, he may, for example, be able to add cash value to the card himself. Many such systems have fallen to determined attacks. PKI tokens are generally not as vulnerable, because the token user is not the attacker.

Whether implemented in software or on a hardware token, private keys should be stored on and cryptographic functions should be implemented in a separate, carefully designed cryptographic module. In the United States and Canada, the FIPS 140-1 [FIPS 140] cryptographic module validation program tests cryptographic modules in independent laboratories and validates them as meeting one of four levels of security.

Elements of a PKI

A PKI is based on a digital document called a *public key certificate*, or simply, a "certificate." A public key certificate is issued to a subscriber and contains the subscriber's public key, specific information about the subscriber, an expiration date, and perhaps, additional information to be used for specific applications. The certificate is digitally signed by a trusted entity called a *Certification Authority (CA)* and issued to the subscriber. Public key certificates are formatted according to an accepted international standard called X.509 [X.509 97]. Figure 3 shows the elements of an X.509 public key certificate.

A CA is the fundamental component of a PKI. All trust in a PKI ultimately begins with the public keys of a CA. CA products are designed to protect their public and private keys and provide careful controls over the use of the CA's key to sign certificates. CAs are typically operated in secure, controlled environments.

VERSION: v3
SERIAL NUMBER: *1123*
ISSUER NAME: *Country=US, Organization=Big Bank, Common Name=CA1*
VALIDITY PERIOD: *1/1/1999 to 12/31/2000*
SUBJECT NAME: *Country =US, Organization =Big Bank, Common Name =Alice Smith*
SUBJECT PUBLIC KEY: *x"0FF01ACD6723A....." a large binary number*
STANDARDIZED OPTIONAL EXTENSIONS: *other information to help the relying party, such as the location to find Certificate Revocation Lists, the Certificate Policy under which the certificate was issued, or the intended use for the subject public key*
APPLICATION SPECIFIC EXTENSIONS: *usually information about the subject, perhaps an account number, a credit limit or some access privilege*
SIGNATURE: *hash of shaded area, encrypted under Big Bank CA's private key*

Figure 3: X.509 Certificate

Another PKI component is known as a *Registration Authority (RA).* The RA is responsible for verifying the identity of certificate requesters, and vouching for that identity to CAs. Figure 4 shows the manner in which Alice obtains a certificate. Alice generates her own signature key-pair. She then requests a certificate and supplies all required identification information and her public key to an RA. Alice does not give her private key to the RA or CA. The RA verifies the identification information and requests that the CA issue Alice her certificate. The CA issues Alice a certificate signed by the CA that binds her name, and perhaps other attributes, to her public key. The certificate is then provided to Alice and placed into a repository that is accessible to users of the PKI. A subscriber generally requests separate certificates for signature and encryption capabilities.

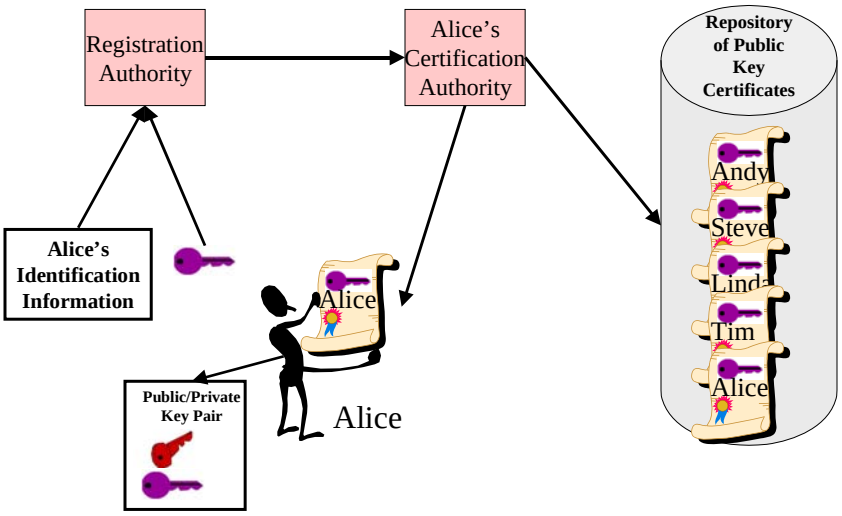
Alice and Bob use a PKI client, a program that uses certificates and private keys to sign messages, verify signatures, and encrypt and decrypt messages. PKI clients are now built into the major e-mail and web browser products and into some servers, such as secure web servers. In

addition, some CA vendors offer toolkits that applications vendors may use to "PKI enable" their applications.

How a PKI Works

Figure 5 illustrates a typical scenario for using a PKI. Alice needs to send a loan application to Bob. She needs to make sure that Bob receives the exact application and she doesn't want her competitor to be able to discover that she is applying for a loan. Alice uses Bob's public key from his public key certificate to encrypt the application so that only Bob can decrypt it. She

Figure 4: Obtaining a Certificate



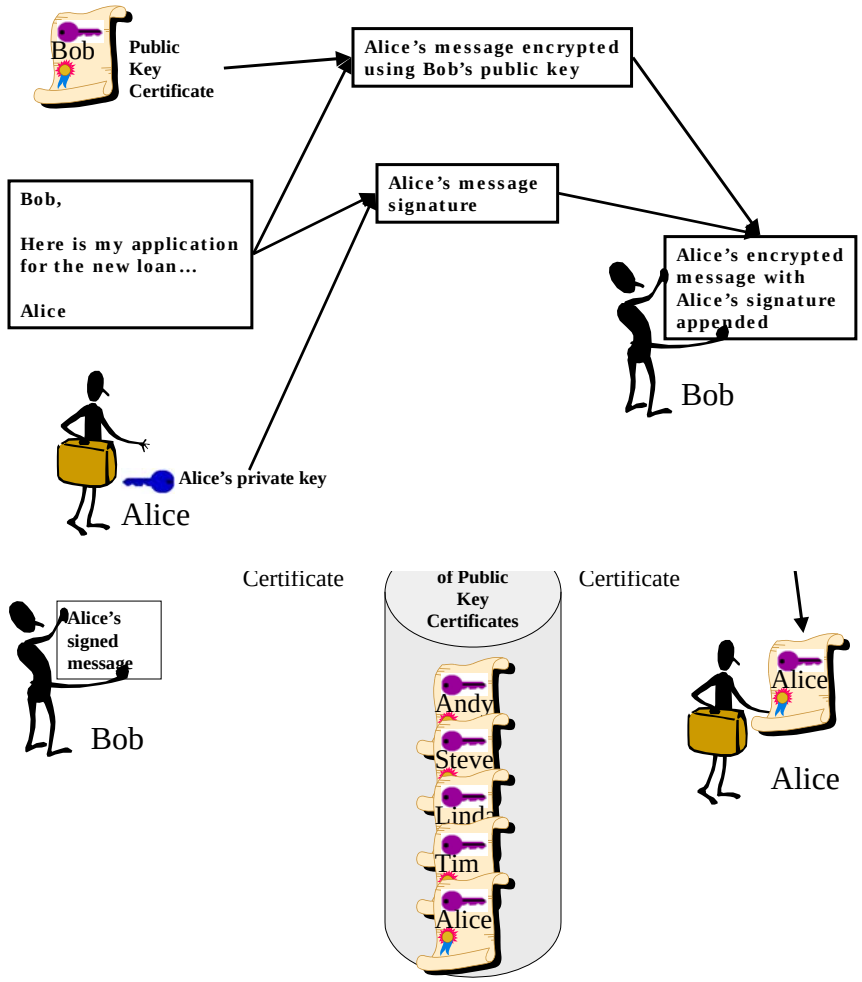
also wants to sign the application so that Bob knows the application is from her and not from an imposter. Bob receives an encrypted and signed message. He decrypts the message using his private key and he verifies that Alice sent it by using Alice's public key from Alice's certificate to verify her signature. Now Bob can have confidence that the message was not altered after it was signed and that it really came from Alice.

In the example above, Alice had Bob's public key certificate. If Alice and Bob share the same CA and the same repository, it is relatively easy for each to get the other's certificate. If, however, they have separate CAs with separate repositories, finding the needed certificates can become more complicated. Bob has to access the repository associated with Alice's CA to get her certificate if she does not include it with her message. However, even if they use separate CA's, Alice and Bob may share a repository or their repositories may be connected.

Once Bob and Alice have each other's certificates, the issue becomes do they trust them? In a PKI, trust usually starts with the public key of a trusted CA, which is called a *trust anchor*. When Alice and Bob went to their RA to get their certificates, they were given a trust anchor key in the form of a *self-signed* certificate issued by the CA to itself. The trust anchor may be the

key of the issuing CA, or it may be the key of some superior *root CA* in a hierarchy of CAs. The normal mechanism for indicating trust between CAs is for one CA to issue a certificate called a *cross-certificate* to another CA. These cross-certificates can be chained together to form a certification path. A relying party starts from a trust anchor and validates a certification path from this starting point, through cross-certificates, until he can validate a user certificate. A PKI is a systematic structure of CAs, the trust anchors, the cross-certificates between CAs, and user certificates. Figure 6 illustrates a certification path of cross-certificates.

Figure 5: Alice Sends Bob a Signed and Encrypted Message



Certificate Status

It does not matter how a relying party gets a certificate, since the CA's signature on the certificate authenticates it, and the certificate indicates its issuance and expiration date. However, it is still

necessary to check the status of certificates before relying on them, because the issuing CA may have revoked them. There are many reasons why a certificate may be revoked. For example, if Alice's certificate was issued to her as a customer or employee, and Alice has closed her account or left the firm, her certificate should be revoked. If Alice's private key has been compromised, then her certificate should also be revoked.

There are two standardized ways to verify current certificate status. The first method is a signed *Certificate Revocation List (CRL)* that the CA issues periodically. A CRL lists the serial numbers of certificates revoked by a CA. A relying party verifies the signature on the current CRL to confirm its authenticity and if a certificate is not listed, the certificate is considered valid. As with a certificate, CAs post CRLs in repositories. It doesn't matter how a relying party gets a CRL, because the issuing CA's signature authenticates it. But old CRLs may be missing recent revocations. CRLs may be the only practical alternative for relying parties who do not always have online access.

The second method to verify current certificate status requires a trusted online server using the *Online Certificate Status Protocol (OCSP)* [RFC 2560]. This server, called an OCSP responder, produces a signed certificate status response message. While OCSP requires an online server (a potential performance bottleneck), it may offer more current status information than CRLs. In its simplest form, an OCSP responder replaces the CRL and answers only the question, "has this certificate been revoked?" Bob trusts the responder because it has a certificate from a CA that Bob trusts.

The simple OCSP responder concept can be carried farther to become a *Validation Authority (VA)*. The VA may offer validation services that go beyond indicating that the certificate is not revoked by the issuing CA. The VA may, for example, indicate whether the certificate can be relied upon for a particular transaction, based on a current account balance, or may undertake to insure or guarantee a particular transaction. The VA's public key can replace a CA's key as a trust anchor. Bob can trust a VA to tell him whether he can rely on a certificate without worrying about the CA that issued it or finding a valid certification path. In effect, he passes that responsibility to his VA. The VA has a list of the public keys of the CAs it trusts and has access to the status of certificates issued by those CAs.

The two approaches tend to favor rather different trust and business models. Figure 7 illustrates the traditional PKI model where a subscriber has an account with her CA, which posts certificates and CRLs in a repository that is usually a directory server. Directory servers may be connected together to provide an overall directory service that makes the certificates and CRLs of many CAs available. Bob, the relying party, does not necessarily have an account or any relationship with the CA that issued Alice's signature certificate. He goes to the directory service to find all the certificates and CRLs he needs. So, the relationship between Alice's CA and Bob may be distant. If Alice's CA or the directory service wants to charge for each certificate verification, it may be impractical to charge Bob to validate Alice's certificate, since Bob has no account with Alice's CA, and the necessary CRLs and certificates are self-validating.

Figure 7: Classical Directory Oriented PKI

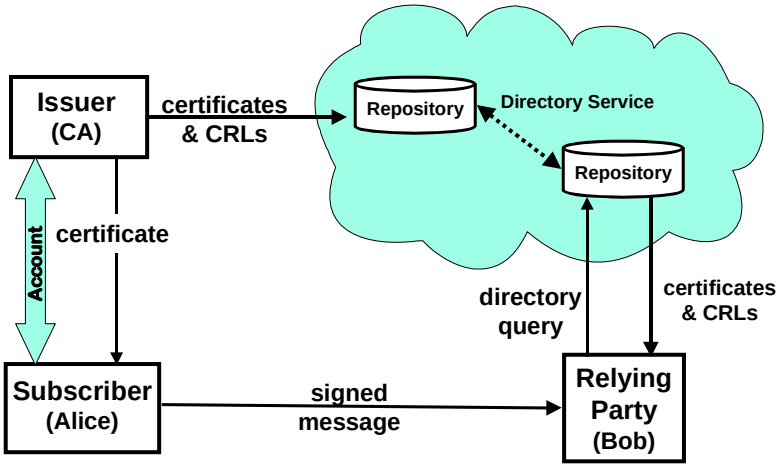
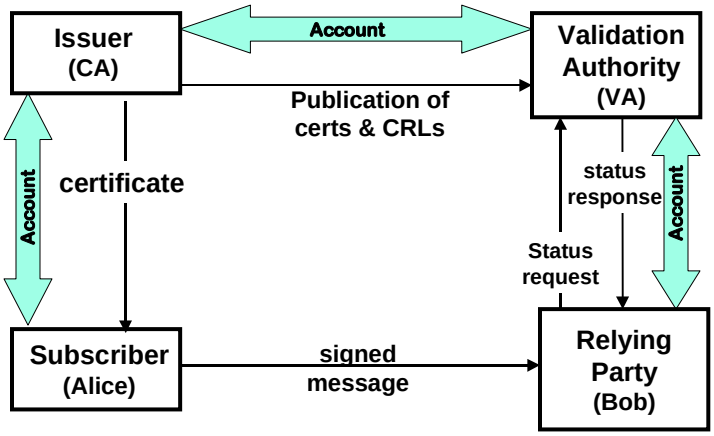


Figure 8 illustrates a different kind of model, a four corner model, which relies on a VA. Bob has an account with a VA. He asks the VA if Alice's certificate is valid. He may ask a more complex question: "Is Alice's certificate valid for transactions of $250?" Bob may have to pay a fee to get a status response, and Bob's VA probably has a business relationship with Alice's CA.

Figure 8: Four Corner OCSP Oriented PKI



In an internal enterprise PKI, or a PKI intended only for the use of an institution's customers, the traditional model may be easier to implement and less expensive, since it does not need a trusted online server. The necessary certificates and CRLs are widely available and can be posted

anywhere, which offers cost and performance advantages. All subscribers and relying parties have an established relationship with the CA.

The four-corner PKI model with a VA may do a better job of reflecting the more complex business, liability, and trust issues where Alice and Bob are customers of different institutions. Bob, now has an account with his VA, and it's the VA that he relies on to validate Alice's certificate. It's easier, in this context, to charge Bob for each validation of a certificate, and the validation can cover more than the revocation status of Alice's certificate. The model explicitly recognizes that Alice and Bob may be customers or employees of different organizations. This is similar to the traditional credit card system, where the cardholder gets his card from one institution, while the merchant relies on a clearinghouse to validate transactions with many different institutions. Therefore, the four-corner PKI model may facilitate business models similar to those now used by financial institutions for payment systems.

PKI CHOICES

An institution must make certain choices in order to use a PKI. These choices are explored below.

Registration

When Alice registers to receive a certificate, her identity is verified. An RA usually performs the identity proofing. Many PKI products have a separate RA capability. However, for large-scale public CAs, the identification process may be largely automated. If an institution is issuing certificates to customers or employees, no additional identity proofing may be required. The purpose of the certificate will usually dictate the level of identification required.

PKI products generally take one of two approaches to registration:

- There is a separate RA that prepares certificate issuance requests and sends them off to a CA, or;

- The applicant sends the request directly to the CA, where it is provided to a RA. The RA then reviews the requests in the queue and approves or disapproves them based upon specific guidelines.

Ubiquitous PKI Applications

Although many specific applications are "PKI enabled," there are now several nearly ubiquitous security applications or protocols that use a PKI. Once an institution has a PKI is in place, and certificates are available, these applications are readily used for many purposes. Three such functions are built into current browser products (or operating systems so they are available for other applications to use):

- **SSL** (Secure Socket Layer, also now known as TLS or Transport Layer Security): this protocol, which is also implemented in web servers, is the basis for "secure web pages." SSL has many options, but in the most commonly used case, a server certificate is used to

authenticate the server to the web browser client and to set up an encrypted channel. Additionally, a certificate in the browser can be used to cryptographically authenticate the client to the server.

- **S/MIME** (Secure Multipart Internet Mail Extension) [RFC 2632]: This protocol allows browser or other e-mail clients with certificates to digitally sign and encrypt e-mail messages.

- **Code Signing**: On Wintel platforms, this allows Java applets, Active-X controls and other programs to be signed and the signature verified before the code is allowed to run. Other platforms may implement similar controls on code destined for their platform.

In addition, **IPSEC** (the Internet Protocol Security standard) [RFC 2401] is also becoming very widely used as a vehicle for constructing virtual private networks (VPN). Although not mandatory, public key certificates can be used with TLS for authentication to the VPN, and encryption.

The following two examples illustrate applications with very different needs and how these are reflected in the registration process:

Customer PKI Example

Alice is a customer of Big Bank. Big Bank offers Internet banking services. Big Bank already knows Alice. Big Bank sends Alice a postal mail notice that she is eligible for secure Internet banking, with some instructions and a one-time authenticator code. Alice goes to a secure web page and fills in a form, including her name, account number, and the authenticator. The browser's normal certificate request process is then activated; the browser generates Alice's key-pair and sends only the public key to Big Bank. Alice's private key is stored encrypted on the disk drive of her computer, and she must enter a pass phrase to decrypt and use the private key. The Big Bank CA then generates a certificate and sends it back to Alice's browser, which adds the Big Bank issued certificate to its certificate store. When Alice visits the Big Bank Internet banking page, her certificate is used to authenticate her to the Big Bank server, through a mutually authenticated SSL [RFC 2246] session (see SSL in the Ubiquitous PKI Applications box).

Open PKI Example

Alice operates a small business and deals with customers and suppliers around the world via the Internet. She needs to be able to establish her identity with strangers worldwide, and to file reports and applications with several governments. Alice needs a widely recognized, highly secure certificate that vouches for her identity and can be used to do business with strangers. She applies to Public CA, Inc. which does a credit check on Alice and her business, and then sends Alice a *smartcard* token, a smartcard reader for her computer, and some software. A smartcard token is a computer chip, in a credit card size plastic package, that can hold Alice's key and perform cryptographic operations with

the key. Alice initializes her token with a key-pair. She now takes her token, plus identification papers (birth certificate, passport, business licenses, etc.) to the local registration office of Public CA, where they are examined. If they are satisfactory, the RA inserts Alice's token into a workstation that extracts Alice's public key (the token is designed so that the private key never leaves the token). The RA sends the public key and other information to the central CA of Public CA, which generates and returns Alice's certificate. The RA loads the certificate onto Alice's token. Now the token itself signs messages for Alice and holds her private key.

These examples illustrate some of the issues, processes and alternatives that apply to registration, which may be the most costly and difficult part of operating a PKI. There are many variations. The Customer PKI example illustrates a self-contained, limited application that is not difficult to implement today. Only Big Bank customers use the PKI, and only to interact with Big Bank. The Open PKI example stretches the envelope today, but illustrates some of the possibilities. The major problem is propagating trust very broadly across institutions and borders. This example requires either a broadly cross-certified public CA, or one that is very widely recognized. Trust propagation is discussed further under Isolated or Interconnected PKI below.

Operate CA or Outsource

An institution can operate its own CA or use a CA service provider. If the institution decides to operate its own CA, the institution can make all policy and procedure decisions regarding certificate issuance, use, and revocation. In this case, the institution installs, operates, and maintains the CA. If the institution uses a CA service provider, it gets the RA capability to control the certificates that are issued and how they are issued, but the service provider remotely issues the certificates. A service provider usually has its own policies and procedures that it follows and the institution may or may not find them acceptable. If an organization wants certificates to be used for both internal and external purposes, they can be issued under the service provider's key, which may be more widely known and trusted.

Isolated or Interconnected PKI

A CA or PKI can be an isolated trust island or it can provide trust connections to other CAs or PKIs. Will a PKI or its CAs cross-certify with external CAs and under what circumstances? An isolated PKI provides authentication and data integrity within the institution and with an identified set of users, but does not help in dealings with other institutions or organizations. It is simpler to implement an isolated CA and the risks are less.

If an institution needs a PKI that is connected in a trust sense to external PKIs, how trust propagates and how far it propagates becomes a big issue. CAs can cross-certify with each other, and relying parties can attempt to find and follow a certification path of certificates from their trust anchor to any certificate. But if CA 1 cross-certifies with CA 2, which, in turn, cross-certifies with CA 3, does trust propagate from CA 1 to CA 3?

Web browser clients and web servers now largely rely on a trust list, a store of many trust anchor self-signed certificates. The two major browser products ship with a set of 40 to 50 trust anchor certificates already installed and they offer some tools for managing the certificates, either

individually at the desktop or through an enterprise configuration and management system. There are obvious business implications to this as well as security concerns. Individual CAs desiring wide public recognition need their self-signed certificates shipped with the browser. But, should a relying party trust a self-signed certificate because it was included in a download from a browser vendor's website?

There are several strategies for dealing systematically with cross-certification in PKIs with many CAs. One is a hierarchy, where the trust anchor for all relying parties is a single root CA at the apex of the hierarchy, and all other CAs are subordinate to the root. Another method is to have each CA be the trust anchor for its subscribers, and to cross-certify with a single bridge CA. A bridge CA is a CA that serves to establish a trust relationship between other CAs. In the bridge CA concept, a common set of policies and practices is established for a community of interest, such as the banking community. All CAs within the community agree to PKI transactions that meet the specified policies and practices. The community of interest CAs each cross-certify with the single bridge CA. Members of the community of interest are then able to perform PKI-based transactions with each other, even if they have different CAs.

Several optional fields in the X.509 certificate standard are intended for managing trust in cross-certified environments, including name constraints in certificates, certification path length constraints, and certificate policy fields. Unfortunately, few clients today process these fields. Differences between the clients can create interoperability issues that confound trust management in heterogeneous client environments. Several “certificate profiles” have been developed that recommend the inclusion of particular optional fields in certificates and processing of these fields by clients as a solution to these interoperability issues.

The U.S. Federal Government is building an overall PKI between its different components (which are large enterprises in their own right) through agency CA cross-certification with a Federal bridge CA. Similarly, the U.S. Automotive Network Exchange (ANX) is building a PKI for the automotive industry through cross-certification with an automotive bridge CA. The Canadian Government is farther along in implementing such a cross-certified PKI, but with the simplification of a single CA and PKI client vendor.

There is little experience with large cross-certified PKIs. Today, the only truly widely recognized certificates are the ones issued by the CAs that have their certificates included in the trusted certificate stores of browser products.

An alternative to cross-certification is to use a VA as the trust anchor, and leave the decision about trust propagation to the VA and its arrangements with different CAs. This has the advantage of simplifying clients, while concentrating trust decisions under the management of a central authority.

Policies and Practice Statements

A CA issues certificates under specified Certificate Policies. The X.509 standard defines a field to identify all the Certificate Policies that apply to a certificate. A Certificate Policy is a written document of the policy under which a certificate was issued and the purposes for which it may be used. For example, an institution’s policy may indicate that certificates issued by the

institution are only for use in transactions with the institution, or that the certificates may be used for electronic commerce transactions up to $5,000. The policy may also indicate the degree of trust someone may place in the certificate. This document usually addresses legal policy issues and disclaimers.

A Certification Practice Statement (CPS) is a detailed statement of the practices and procedures a CA uses to issue certificates. The CPS usually identifies procedures the CA uses to identify individuals and entities, how it generates and protects its key-pair, its physical and network access controls, and procedures it follows such as separation of duties and backup procedures.

Both of these documents are generally referenced during a cross-certification procedure. This procedure determines whether CA 1 will accept a cross certificate from CA 2 and vice versa. The cross-certificate implies a level of trust between the two CAs.

INTERACTIONS USING A PKI

The interactions that a financial institution engages in have been divided into five basic types for the purposes of this paper. They are described below. Each has different requirements for a PKI.

Internal Enterprise Interactions

Interactions internal to the organization involving the institution's internal business procedures

Inter-Enterprise Interactions

Interactions with other enterprises, particularly other financial institutions, where there is an existing peer relationship

Government and Regulatory Interactions

Interactions between a financial institution and a government agency or regulator

Customer Interactions

Interactions between a financial institution and its own customers

Open Interactions

Interactions between a financial institution and anyone with which they do not have an existing relationship. For example, interactions with potential new customers.

Typically, a PKI exists to provide support to the underlying financial institution's business. It should provide some feature or advantage that, at least in the long run, will compensate for the cost of implementing and maintaining a PKI.

Internal Enterprise PKI

A PKI for strictly internal use by an institution is the easiest PKI to implement. An institution already knows its own employees. It can train its personnel on how to use the PKI. It controls both certificate holders and relying parties and can enforce administrative procedures on both. CA liability is easily managed. Hardware tokens, if warranted, can be issued to hold private keys. The institution can control the PKI clients its employees use and employ "value added" clients that have more capabilities and are more secure than generic browser clients. There are many internal business applications for a PKI, including:

- digital signatures for approvals when electronic documents replace paper documents;
- digital signatures to protect the integrity of records;
- strong user authentication to log onto systems or access restricted files, or pass through firewalls, and;
- data encryption to protect sensitive information.

Inter-Enterprise PKI

A PKI that spans institutions is more complex. In many cases, only certain employees will be authorized to participate in inter-enterprise applications. If the PKI that supports these applications is built by cross-certification of general corporate CAs, then some mechanism (typically a policy field in certificates) is needed to identify how certificates can be used. Relying party clients must recognize and interpret the policies.

One method of building an inter-enterprise PKI is the bridge CA concept discussed earlier. Another alternative is to set up a separate PKI for an association or group of businesses. The separate PKI would issue certificates directly to those individuals from each institution that are authorized to speak for their institution in that association. Alice now may have one certificate issued by her own institution and several others issued by external organizations.

Liability is a major concern in such an inter-enterprise PKI. In what ways can Alice obligate her institution? These issues are not new with a PKI, and may already be addressed by business agreements. Certificates may include fields that identify business roles, but this is only useful with client software that can recognize these fields.

Government or Regulatory PKI

Financial institutions are often heavily regulated and may operate in many jurisdictions. In the United States, the Government Paperwork Elimination Act will force government agencies to make electronic filing available for citizens and businesses, where paper is now required. Even if the law did not require it, the government would be moving this way for the sake of efficiency and better service. State and Federal Courts are also developing procedures for filing legal documents. A PKI and digital signatures are an excellent way to authenticate these transactions.

It is generally not clear what kind of PKI will be created for governments and courts and who will create it. Some U.S. states and some countries have laws for licensing or regulating public CAs. In the U.S., some Federal or state agencies may choose to issue certificates to businesses and financial institutions to sign filings. Others may cross-certify directly with institutional CAs.

The U.S. General Services Administration is developing a program to issue certificates to citizens and businesses that can be used with government agencies. In this case, the certification process will be contracted to private CAs, who will verify the identities of certificate holders and recover their costs by using an OCSP-like protocol to validate certificates with each use. Government agencies will pay for each validation.

Customer PKI

Many financial institutions now offer online services to their customers over the Internet. Institutions often have secure webservers that use the SSL protocol (see the box Ubiquitous PKI Applications) with a server certificate to establish an encrypted link for the banking service. The customer is usually authenticated with a PIN.

A few financial institutions are now issuing certificates to customers to use for online banking. The SSL protocol can use a client certificate to authenticate the client to the server. This method offers stronger customer authentication than a PIN. In most cases, certificates are strictly for use with the issuing institution. Since these applications generally plug into and require an online system with access to account information, certificate revocation is handled automatically. There may not be any information about the customer except an account number in the certificate. Everything else is available through access to account information. The business model is very simple, because the issuer and the relying party are the same institution.

The primary barrier to the use of customer PKIs is current PKI clients. They are not as user friendly and as mature as they need to be to allow ordinary customers to be comfortable getting, managing, and using their certificates.

The Secure Electronic Transaction (SET) protocol combines a special purpose PKI with the existing credit card infrastructure to allow more secure use of credit card services over the Internet. This goes beyond a single financial institution, but its use is still limited to credit card transactions. SET is still relatively new and is not widely used today.

Open Public PKI

In the modern online world, the Internet makes business or legal transactions between strangers practical, or even routine. A PKI is needed that spans the globe and supports authentication of identity, signatures, and encryption between strangers, who may be businesses, individuals or governments. A few public CA start-ups have addressed this market, and at least one has a successful business providing certificates that identify secure webservers.

As online services replace in-person services, some financial institutions may wish to offer online “walk-in” services that require open online interaction, but also require that identities be authenticated. For such services, the institution’s PKI would accept any valid certificate issued by a reputable CA. To ensure maximum business opportunities, the institution would have to be aware of most public CAs and would need some understanding of their trustworthiness. The institution could then establish a list of trusted public CAs whose certificates it recognizes.

It is possible that financial institutions themselves could fill the role of public CAs. Financial institutions know who their customers are, and are largely in the business of being trusted, at least in a financial sense. They would be well positioned to issue general use identity certificates to their customers. However the business models and potential liabilities of such a public CA service are not familiar and well understood.

SUMMARY AND CONCLUSIONS

PKI technology is here now, already in use and widely available. An emerging PKI industry includes a number of vendors of PKI products, as well as commercial CA services. Secure web pages that use public key certificates are already a linchpin of electronic commerce. Laws in many jurisdictions recognize digital signatures. Useful PKI clients are already embedded in ubiquitous web browsers, and there are a good number of "PKI enabled" software applications.

But, PKI technology is still in its infancy and has many challenges to overcome before it reaches it's full potential as one of the great enablers of the online world and electronic commerce. Client and application software is probably the weakest link at the present time. The software tends to be relatively immature and often has poor user interfaces. These clients may not yet offer needed features, such as the ability to automatically find certification paths or check certificate status. PKI vendors are working to resolve some of these issues and they may disappear over time.

PKI technology presents financial institutions with many opportunities to improve their internal operations and to deal with other institutions more securely. It offers financial institutions better ways to serve their customers in the online world, and PKI may offer financial institutions business opportunities that extend the traditional concepts of financial services.

REFERENCES

[FIPS 140] FIPS 140-1, *Security Requirements for Cryptographic Modules*, NIST, 1994, available at: http://csrc.nist.gov/cryptval/.

[RFC 2246] T. Dierks and C. Allen, *The TLS Protocol, Version 1.0*, RFC 2246, Internet Engineering Task Force, November 1998, available at: http://www.ietf.org/html.charters/smime-charter.html

[RFC 2401] S. Kent and R. Atkinson, *Security Architecture for the Internet Protocol*, RFC 2041, Internet Engineering Task Force, November 1998, available at: http://www.ietf.org/html.charters/ipsec-charter.html.

[RFC 2560] M. Myers, R. Ankney, A. Malpani, S. Galperin, and C. Adams, *Internet Public Key Infrastructure Online Certificate Status Protocol - OCSP*, RFC 2560, Internet Engineering Taskforce, June 1999, available at: http://www.ietf.org/html.charters/pkix-charter.html.

[RFC 2632] B. Ramsdell, *S/MIME Version 3 Certificate Handling*, RFC 2632, Internet Engineering Task Force, June 1999, available at: http://www.ietf.org/html.charters/smime-charter.html.

[RIV 78] Rivest, Ronald L., Adi Shamir and Leonard L. Adleman, "A Method for Obtaining Public Key Signatures and Public Key Cryptosystems," *Communications of the ACM,* 21 (1978).

[X.509 97] ITU-T Recommendation 509, *The Directory: Authentication Framework*, ITU, International Telecommunications Union, June, 1997.