

Public-Key Infrastructure (PKI): Concepts and Technology

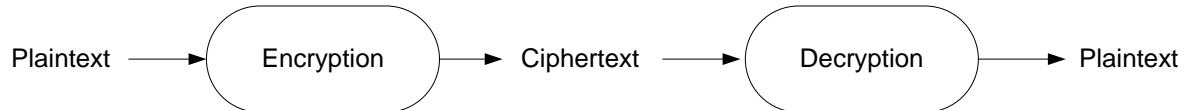
By Yashpal Matharu, Executive IT Management Consultant with Taos' Office of the CIO Practice

Security and the Internet

As all Internet communications are inherently insecure and subject to eavesdropping and tampering, the Internet model poses a number of security issues. Eavesdropping, tampering, insecure server, denial of service, downloadable code, insecure browser and impersonation and masquerading are some of the security risks the Internet Model poses. With the advent of e-commerce, more and more confidential and financial information is transmitted over public networks. The e-commerce (B2B, B2C, e-government, e-delivery) systems require that we can authenticate, identify, and create relative trust so that we can conduct business on the public network. That is where cryptography comes into play.

Cryptography

Cryptography is the science of making the cost of improperly acquiring or altering data greater than the potential value gained. Cryptography provides techniques for mangling a message or a block of data into an apparently unintelligible form and then recovering it from the mangled form, illustrated in Figure 1.



Encryption and Decryption of a Message

Figure 1

The original message is called **plaintext** or **cleartext**, and the unintelligible message is referred to as **ciphertext**. The mangling step is called **encryption** or **encipherment**; the demangling step is called **decryption** or **decipherment**. The mathematical function that identifies the encryption or decryption steps is termed the **cryptographic algorithm**, or simply the **cipher**. A **cryptosystem** is a collection of cryptographic algorithms, cryptographic keys, and all possible plain text and their corresponding ciphertext. All mathematical cryptosystems are based on only three cryptographic algorithms: **message digest**, **secret key**, and **public key**.

A **key** is a mathematical value that determines how a plaintext message is encrypted to produce ciphertext and its procession is required to decrypt the ciphertext and recover the original message. A key has a corresponding key length, which is the number of bits, or sometimes the bytes, in the key. Security of an algorithm is determined by its **key length** since an algorithm can be cracked by brute force. That implies that the security of an algorithm is also governed by the speed and the amount of processing power available to a hacker who is trying to break the key using brute force. As processing power doubles or so every 18 months, it becomes a necessity to increase key length every so many years. See appendix A for interesting facts that were published in 2006.

Message-digest algorithms map variable-length plaintext to fixed length ciphertext. They do not have any keys, and it is computationally infeasible to recover the original plaintext from ciphertext. Message-digest algorithms are usually used to convert large messages into messages of a smaller, more manageable size.

Secret-key algorithms encrypt plaintext messages to ciphertexts that are usually the same size. They use one secret key and it is not feasible to decrypt the message without knowledge of the secret key. Secret-key algorithms are generally used for bulk encryption.

Public-key algorithms are similar to secret-key algorithms with a major exception that they use two keys, one **public key** and one **private key**. These algorithms are frequently used to distribute secret keys.

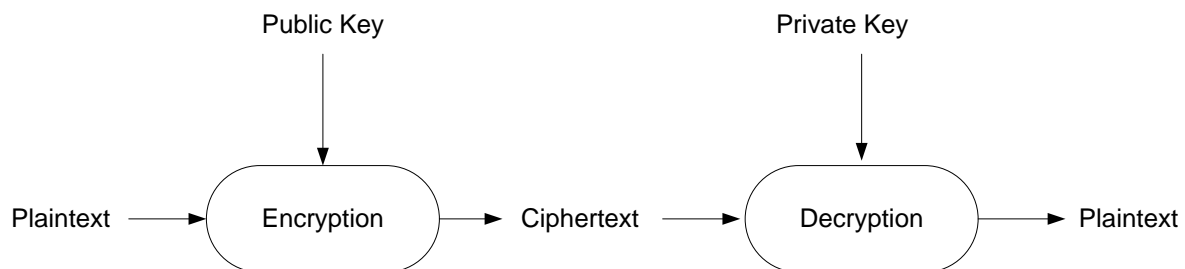
Challenges with Cryptography Systems

In secret-key cryptography systems, the biggest issue is how to exchange keys especially in today's Internet and e-business model. Different solutions were proposed, but they are not scalable and require a great amount of administration work to operate the solution. One solution that provides a mechanism to exchange and distribute keys in a safe fashion is the **Public Key Infrastructure**, which is based on the **Public-Key Cryptography**. Because the public-key algorithms use two keys, one of which can be shared, they make it possible to use cryptography in a practical, scalable way in today's digital world.

Public-Key Cryptography

Encryption and Decryption

Public-Key Cryptography was invented by Whitfield Diffie and Martin Hellman in 1975. It involves the use of two distinct keys, one public and one private. The **private key** is kept secret and must never be divulged; the **public key** is not secret and can be freely shared with anyone. The public and private keys are commonly called a **key pair**. There is a mathematical relationship between the key pair. The relationship is such that a message or a block of data encrypted by a public key can only be decrypted with the corresponding private key. See figure 2.



Encryption and Decryption using a private and public key pair

Figure 2

Digital Signatures

A digital signature is a data item that vouches for the origin and the integrity of a message. The originator of the message uses a signing key (private key) to sign the message and its digital signature to a recipient. The recipient uses a verification key (public key) to verify the original message has not been tampered with while in transit.

The public-key algorithm is reversible if it can be used for both confidentiality (encryption) and digital signatures. Public-key cryptography relies on these two things: keeping the private key secret and providing a mechanism to distribute the public key. **Public Key infrastructure** provides a scalable way to generate, store, validate and publish revoked certificate lists and distribute public keys through the use of **digital certificates** and **certification authorities (CA's)** which make up the PKI. I will discuss digital certificates and certification authorities later in this article. Please see Figure 3.

Public Key Cryptographic Standards (PKCS)

Public key cryptographic standards (PKCS) were developed in 1991 by RSA Laboratories in collaboration with a number of companies and academic institutions. PKCS has become the basis for many formal standards and is implemented in a variety of commercial applications.

PKCS addresses the pitfalls of public-key cryptography, such as the encryption of guessable messages and short messages, and provides standards for RSA encryption, Diffie-Hellman key agreement, password-based encryption, extended-certificate syntax, cryptographic message syntax, private key information syntax, certificate request syntax, etc.

PKCS is actually a collection of 12 papers on PKCS – PKCS#1 to PKCS#12. Additionally, PKCS provides two supplementary documents, “An Overview of the PKCS Standards” and “A Layman’s guide to a Subset of ASN.1, BER, and DER.”

Digital Certificates and Certification Authority

A public-key certificate is a binding between an entity’s public key and one or more attributes relating to its identity. The entity can be a person, a hardware device such as a router, or a software process. The public-key certificate provides assurance that the public key belongs to the identified entity and that the entity possesses the corresponding private key. A public-key certificate is loosely called a **digital certificate**, **digital ID** or a **certificate**; the entity identified in a certificate is referred to as **certificate subject**. If the entity is a person, it is often referred to as a **subscriber**.

Why do we need a certificate? A digital certificate serves as an identification card, such as a driver’s license, in the digital world. A driver’s license binds a photograph of a person to their personal information like name, address, height, weight, etc. It provides assurance that the person carrying the license is who he or she says they are. If an individual issues or creates their own driver’s license it is not of much use the same way an individual who creates their own certificates are not generally very useful in large communities. The department of motor vehicles (DMV) regulates the issuance of driver’s licenses and ensures the individual presents correct personal information when applying for a license. Similarly, trusted third-party **certification authorities (CA's)** confirm the identities of their subscribers and issue digital certificates. The next question that arises is how do we verify if the certificate issued is authentic or forged? The certification authority, after authenticating the identity of a subject, digitally signs the certificate, thereby incorporating its signature into the certificate. The certification calculates the digital signature by computing the message digest of the certificate and encrypting it with its private key. An attacker cannot forge a certification authority’s certificate because it does not know the certification

authority's private key and cannot generate the correct signature. Furthermore, if an attacker makes changes to a genuine certificate, the message digest of the certificate changes and no longer matches the certificate authority's signature.

Digital certificates have various applications in the digital world; they can be used to verify the identity of a person or device, authenticate software downloaded, send an encrypted or digitally signed email, control access to resources, implement nonrepudiation (**non-repudiation** is the concept of ensuring that a party in a dispute cannot repudiate, or refute the validity of a statement or contract. Although this concept can be applied to any transmission, including television and radio, by far the most common application is in the verification and trust of signatures.), etc. Digital certificates provide an alternative, scalable model in which third-party certification authorities authenticate persons and certify their public keys. A person can then find another person's public key by obtaining his certificate from a trusted certification authority and verifying the CA's signature. This model of public-key distribution is scalable because a small number of CA's can authenticate a large number of users in a community. Digital certificates can be transmitted via an insecure channel and stored in insecure mediums. Any changes made to the certificate can be detected. As a result, certificates can be stored in insecure repositories and freely transported across the Internet.

A CA can issue various classes of certificates. The information a subscriber or entity needs to provide to determine its identity depends on the class of the certificate. The measure the CA undertakes to confirm the identity of the subscriber also determines the level of assurance the certificate purports to provide.

A digital certificate is open data structure; it can contain domain specific knowledge and can be extended to contain application specific information. A variety of control information can be encoded in a certificate, such as security privileges, access privileges to use resources, etc. This allows us to build scalable and secure systems that are less server-centric and therefore more reliable. The X.509 is a standard and more elaborate format for the certificates. It is an International Telecommunication Union-Telecommunication Standardization Sector (ITU-T) certificate standard, first published in 1988 as part of X.500 Directory recommendations. The X.509 certificate is specified in a notation system called Abstract Syntax One (ASN.1).

Public Key Infrastructure (PKI)

The widespread adaptation of public-key technology requires a **public key infrastructure (PKI)** that defines a set of agreed-upon standards, certification authorities, structures between multiple certification authorities, methods to discover and validate certification paths, operational protocols, management protocols, interoperable tools, and supporting legislation. Operational protocols address the requirement to deliver certificate and **Certificate Revocation List (CRL's)** to certificate-using systems.

There are six key elements within a PKI. They are:

1. **The Security Policy**
 - Certificate Policy/Certificate Policy Statement (CP/CPS)
2. **The Certification Authority (CA)**
 - Issues and revokes certificates
 - Delivers the certificate to end user
 - Distributes certificates and the certificate revocation lists to certificate repository

- 3. The Registration Authority (RA)**
 - Responsible for identification and authentication of entities that want to enroll for a certificate
 - Enrollment and registration
 - Credential issuance, usage and revocation, re-issuance
 - RA is delegated certain tasks on behalf of an authorized CA
- 4. The Certificate Repository and Distribution System**
 - Enables both end-users and end-entities to search for certificates and CRL's
- 5. The End-user or subscriber**
 - Requests certificates from a CA
 - Receives the certificate from CA
 - Uses the certificate for authentication, encryption, and nonrepudiation
- 6. The Service Provider**
 - Provides application services on PKI, thus enabling support for strong authentication, encryption and nonrepudiation.

The basic key processes that are required to support PKI are:

- Certificate generation
- Certificate Signing Request (CSR)
- Issuing certificates
- Revoking certificates
- Certificate distribution
- Certificate Path discovery and validation
- Authentication/Verification
- Manage/Maintain and Publish Certificate Revocation Lists
- Publish Certificate Policy and Certificate Practices statement
- Conduct Key Ceremonies
- Key Escrow
- Software to generate random numbers, generate keys etc.

Public-key Cryptography and PKI have weaknesses. The whole system relies on keeping the private key of the end-user and the Certificate Authority secret. This is the weakest link in this security system. A lot of policies, procedures and technologies are employed to keep both the CA's private key and the end-entities key protected increasing the cost of the PKI making it less accessible. PKI has been around for two decades but has not taken off even though e-commerce has become a multi-billion dollar business. One of the key reasons is the complexity of the certificate lifecycle management and coming up with secure procedures to verify a person's or end-entity's credentials before issuing a certificate. These processes are time consuming, costly and not transparent to the end-user or end-entity. There is also a lack of definition of standard interfaces between applications and PKI that want to leverage PKI to enhance their security.

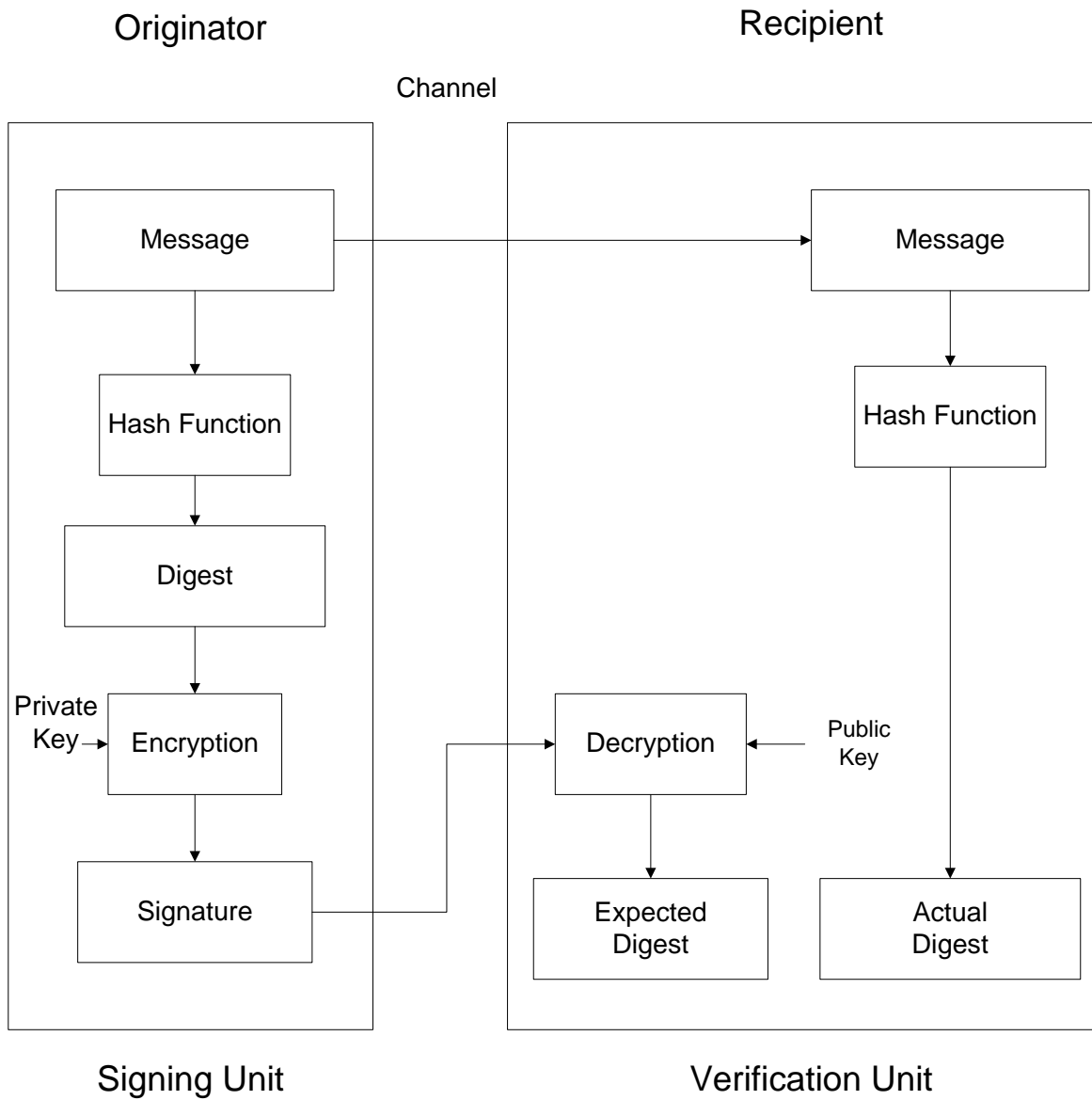


Figure 3 – Digital Signature

Appendix A

Here is an interesting story about key length and the time and resources it takes to break the key. Burt Kaliski first came up this characterization.

Imagine a computer that is the size of a grain of sand and that can test keys against some encrypted data. Also imagine that it can test a key in the amount of time it takes light to cross it. Then consider a cluster of these computers, so many that if you covered the earth with them, they would cover the whole planet to the height of 1 meter. The cluster of computers could crack a 128-bit key on an average of every 1,000 years.

However, 2^{32} isn't necessarily a very large set of keys when you're performing a brute force attack with a **worldwide distributed network of computers**. Below are some statistics from the RC5 distributed computing project. Here's what they've done so far:

- **56-bit** key was cracked in 250 days.
- **64-bit** key was cracked in 1,757 days.
- **72-bit** key is still being cracked; 1,316 days so far with 379,906 days remaining.

The earliest 56-bit challenge, which ended in 1997, tested keys at a rate of 1.6 million per second. The ongoing 72-bit challenge is currently testing keys at the rate of 139.2 million per second. We're testing keys 88 times faster than we were 10 years ago through natural increases in computing power and additional computers added to the distributed computing network.

And yet the RC5-72 project **still has 1,040 years to go before they test the entire key space**.

Remember, that's for a lousy 72-bit key! If we want to double the amount of time the brute force attack will take, all we need to do is tack on one teeny, tiny little bit to our key. 73-bit key? 2,080 years. 74-bit key? 4,160 years.

It's painfully clear that a brute force attack on even a 128 bit key is a fool's errand. Even if you're using a planet covered with computers that crack keys at the speed of light.