Global Knowledge®

Expert Reference Series of White Papers

Fundamentals of the PKI Infrastructure

Fundamentals of the PKI Infrastructure

Boris Gigovic, Global Knowledge Instructor, CEI, CCI, MCT

Introduction

Securing corporate information can be a challenge these days, considering the numerous technologies and platforms that need to be protected—it can be especially challenging for companies that lack a unified security system that can circumvent and mitigate issues. It is well known that organizations have to rely on an in-depth defense approach to make sure access to information stays secure at all times.

In order to remain in a secured state, information needs to:

- Be accessible only by authorized entities (*confidentiality*)
- Be unable to be tampered with (*integrity*)
- Remain online without interruption (*availability*)

These three conditions are known as the CIA triad. One technology that definitely helps achieve some of these objectives is public key infrastructure (PKI). Although it does not guarantee data availability, the PKI is a mechanism that greatly enhances the security of our data by using advanced access methods and making sure the authenticity of the data flow is preserved.

Given a complex system, how does the PKI help achieve these objectives? How does it integrate into the CIA triad? What major components does it use, and what are system administrators required to perform to maintain this technology? This paper is all about the foundations of PKI, since it's becoming mainstream in many companies to implement PKI. But I've often seen a lack of real understanding of the PKI in order to insure proper integration of the system.

Let's explore and see how PKI works!

Symmetric and Asymmetric Cryptography

The PKI relies on cryptography to encrypt and decrypt data. At the very base of the PKI, we find different ciphers or encryption algorithms that are able to masquerade the data (initially known as the plaintext) and make it unreadable, if some conditions are not met.

To help us understand how a cipher works, let's take this basic example:

Plaintext (unsecured data): **PKI IS FUN!**
Cipher text (secured data): **CXV VF SHA!**

How did we actually generate that cipher text? If you take a deeper look, you will find we have used a working cipher by the name of ROT13. This cipher replaces a letter with the letter 13 letters after it.

Knowing this fact is the key that helps in the decryption of the message.

Of course, much stronger ciphers are now used to make the data unreadable in case the user or system does not have the key.

Within PKI, we need two kinds of encryption systems:

1. **Symmetric cryptography**: This type of encryption uses a very low overhead method to encrypt data and is very fast and efficient. We use symmetric encryption as a first layer of security over a plaintext. With this kind of encryption, the key that encrypts and decrypts the message is the same.

   Some examples of encryption algorithms within this category include AES, Blowfish, CAST, IDEA, and RC4.

   The main reason we have to use another kind of encryption on top of it is because of a fundamental problem this technology has: the use of the same key. Why is it an issue? Because there is no way to guarantee a safe delivery of the key to the other end. The key cannot be kept secure, presenting a problem when information is sent.

2. **Asymmetric cryptography**: This type of encryption adds much more overhead to the encryption process. It uses a pair of keys, making users performing the encryption and decryption process use different keys. One key is used to encrypt the data (public key), and the other is used to decrypt it (private key). These two keys are mathematically related and unlock each other. But, in strong ciphers, one key cannot be found by having the other from the pair.

   In the most basic sense, the asymmetric system has been developed to allow the secure transport of the first key, initialized by symmetric cryptography.

   Some examples of asymmetric algorithms include Diffie-Hellman, El Gamal, RSA, and Elliptic curve.

   When dealing with PKI, we mostly deal with asymmetric cryptography, as we do not have much control over the symmetric encryption process. But, with our own pair of keys, we can decide to encrypt data and select who else is able to read it through decryption.

Confusing? Not so much, you'll find. Let's discover the steps:

1. When Alice encrypts a message, the system uses symmetric algorithms to secure the message and generates the encryption key.
2. The encryption key is appended to the message.
3. Before the message is sent to Bob, asymmetric encryption is applied over it. Bob's **public key** is used to cipher the message.
4. The message is sent or stored into an unsecure network.
5. Upon receiving the message, Bob unlocks his public key with his private key (reversing asymmetric encryption).
6. Now that the encryption key is found appended to the message in clear text, Bob uses it to unlock the message itself (reversing symmetric encryption).

In this example, the objective was to secure the message by applying encryption on it (confidentiality). If the objective is integrity (make sure the message is not tampered with), then it is going to have a slightly different processing:

1. Alice creates a digest from the message (a hash—a fixed-length value derived from a hashing algorithm, for example 3cc31cd246149aec68079241e71e98f6).
2. Alice encrypts the digest/hash using asymmetric cryptography, but uses her own **private key**.

3. The message is sent to Bob, with the hash being encrypted, not the message itself.
4. Bob uses Alice's public key to decrypt the hash. The hash is then retained in memory, as Bob performs the same hash calculation over the plaintext message.
5. Bob compares the hash received within the message and the one he calculated.

If values of both digests match, it means the message has not been tampered with, as any attempt to change its content would have altered the initial digest and would have produced a new one upon Bob's verification of the digest.

In this case, we have made use of two concepts: non-repudiation and authentication, which both guarantee the message was sent by Alice, since her public key (that Bob had) was able to decrypt her private key. Knowing this, Alice cannot deny having sent the message, as well.

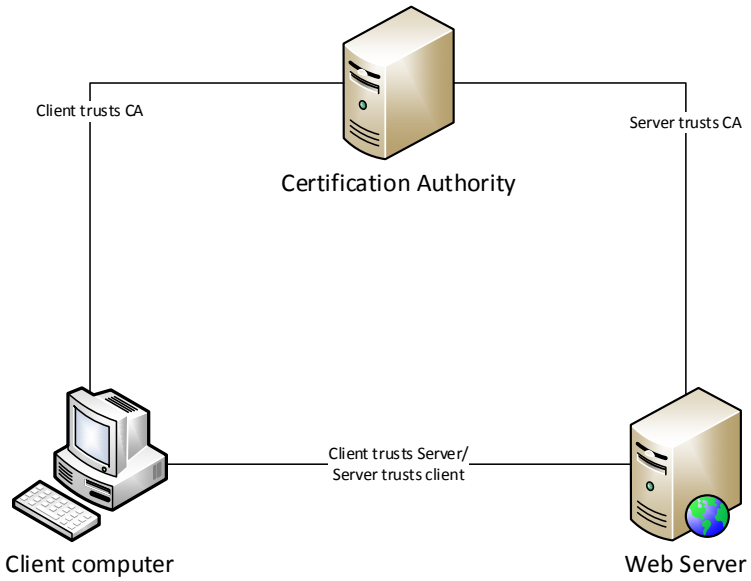
Why Use PKI?

The PKI role is useful in many scenarios, since the way we exchange data by default might be unsecured, while there is need to protect information in transit. For example, when you access web servers to transmit sensitive data, these should make use of the PKI through web server certificates.

If the communication stays unencrypted between your client and a web server, the following risks are occurring:

- No ability to prove you are sending information to the real server (susceptible to man-in-the-middle attacks)
- No data encryption to transmit your information in a confidential way (plaintext, can be captured on network)
- No server authentication, server cannot prove its identity to the client (again, susceptible to man-in-the-middle attacks).

When using the PKI to secure these types of servers, certificates that are generated will be used to provide an end-to-end secure flow of communication. In this particular case, PKI is all about establishing trust between the client, server, and issuer of certificates (certification authority), as well as insuring this protection of data in transit.



There are many other reasons to use PKI:

- Control data and disk encryption on your network through an Encrypting File System (EFS) and BitLocker
- Configure authentication between your network components (domain security)
- Secure your emails by using Secure/Multipurpose Internet Mail Extensions (S/MIME)
- Configure advanced authentication mechanisms, such as these requiring certificates when authentication across a virtual private network (VPN) link or from outside the organization
- Configure end-to-end communication (tunnel) between two organizations or sites (for example, a site-to-site VPN configured with Internet Protocol security (IPSec)
- Configure smart card authentication through enrollment agents
- Configure data recovery and key recovery agents, that are able to restore lost certificates or unencrypt data for emergency purposes
- Configure digital signatures, allowing recipients to validate from whom the information came

In all these cases, PKI will involve several components, which are explained within our next section.

Components of the Architecture

Certification Authorities

A certification authority (CA) is a server role that facilitates the functioning of all PKI components. By design, the PKI service is managed in a central way, and the CA is directly responsible for that: it issues certificates to clients (web servers, users, computers, and so forth) and deals with renewals, revocations, as well as general maintenance of the entire certificates infrastructure.

Public CA

This is a server that all clients trust, as its information is already integrated in the certificate store of every client that connects to it (this information is known as the CA's public key). In other words, certificates issued by this CA are automatically trusted by all your clients, thus no additional configuration is required to establish the trust we have previously discussed.

A public CA is the one we typically do not manage—it is rather maintained by specific organizations, such as Verisign, Entrust, and Thawte. Cost is involved in purchasing certificates from this type of CA. Typically, secure servers accessed by external users will require a certificate obtained from a public CA. Although certificates have to be acquired and are not free, a public CA is an interesting option when you do not plan to manage your own CA, which can become quite a complex task as we will see.

Private CA

In the case of private CA, we can have the full control over the setup or configuration of how certificates are distributed on clients. It is free to use but by design, servers and clients requesting certificates from this type of CA, are not trusted outside the company's infrastructure. This is an interesting option to use when issuing certificates to reinforce authentication, or to secure servers that are not accessible from the Internet. Depending on the product used, a private CA can integrate into Active Directory (or enterprise), for example, making certificate enrollment easier or be deployed as a standalone configuration.

Subordinate CAs

Although CAs issue certificates, it is unlikely that root CAs themselves are to issue certificates. Root CAs are typically kept offline, for security reasons. Subordinates, configured as delegates by the root for certificate enrollment, are the ones interacting with clients and performing the most of the work. Many subordinate servers are likely to exist, linking to the same root. They can deliver certificates for different sites, or be classified by certificate usage/templates or by types of clients.

Certificates templates

Certificate templates are used to define the building blocks of a future certificate issuance. They allow administrators to specify a certificate usage (web server, user authentication, IPSec, and so forth), customize the certificate, define mandatory information upon request, and apply auto-enrollment permissions. Certificate templates are available only when using an internal CA, which is specifically an enterprise type. One of the key functionalities certificate templates offer is the ability to auto-approve requests, which is of interest for reducing administrative overhead and some of the maintenance of a CA.

Certificates

A certificate is a piece of information containing private and public keys, which are used whenever a secure communication is requested. The certificate itself is unique and labeled with key information of the owner, such as its name, issuer and usage, the thumbprint, serial number, expiration date, as well as the CRL information. Certificates are found within computer and user stores on systems, and can be imported and exported. The way this information is presented in a certificate is defined by a standard named X.509 (not to confuse with X.500).

Certificate Revocation Lists

A certificate can be terminated or suspended: its private key can be compromised, expired, or temporarily disabled. In this case, there must be a location a client can refer to validate its state. Such location is a named a certificate revocation list (CRL). A CRL can be published within a Lightweight Directory Access Protocol (LDAP) server, or accessible from a web page, where the client can connect and retrieve information. For certificates issued by an internal CA, administrators must manually publish the CRL, while public CAs populate this information automatically for the customer.

How to Request a Certificate?

There are multiple ways to request a certificate. Typically, this will depend on the type of certificate that has been configured (public vs. private) or the type of CA (standalone vs. enterprise).

Web interface: This standard way of requesting a certificate is performed by browsing to a web page from where you are able to select a standard template available to you and perform the request. Since the web interface option is typically used in cases when no LDAP integration is performed, users have to provide information about them, and send their request for manual approval.

MMC consoles: In Windows systems, we can use perform certificate requests by using the management console and the certificates snap-ins. In this case, a list of available and ready-to-customize templates is available. Upon filling the information, a certificate can be issued automatically. This type of deployment is available only when using private CAs with certificate templates.

Auto enrollment feature: Based on certificate templates, administrators can configure auto deployments of certificates. A configuration of the certificate templates and group policies can help in massively deploying certificates transparently for users. Such option is very practical while configuring client authentication, for example. As this does not require any configuration from the client itself, it is a process fully administered from server-based tools.

Third-party tools: There are third-party utilities from which a certificate request can be performed. Typically, a procedure to create a certificate request can be done. That request can be in some cases sent directly from that interface, or the web page option can be used to send the request offline. For example, Microsoft's IIS console allows you to create a request in such way.

CA Maintenance

Maintaining a CA is critical, and it takes time. In fact, not only are we dealing with the CA, but also with every deployed certificate. We must ensure there is constant synchronization between these components, so they all run within the scope of best practices and recommendations. Below, we find some of these tasks:

Root CA certificate deployment: One of the critical tasks that must be performed is to make the client trust certificates issued by your internal CAs. Such a process needs to be done once, but also every time the root CA's public key expires. That public key from the CA needs to be exported, and then imported into the store of every client of the organization. Such configuration can be performed through group policies, for example.

Certificate approvals: If a standalone version of the private CA is installed, every certificate request needs to be manually processed by administrators. In that case, a constant monitoring is necessary, to make sure clients that requested certificates can get them on time. Moreover, it will be necessary to pay attention to the kind of certificate requested, as the system is not able to automatically apply permissions.

CRL updates: It is critical to perform updates of CRLs. Although there is an automated process for this task, there are critical situations when administrators need to perform it, such as after revoking a stolen certificate. There is a possibility of performing incremental updates on CRLs, that way the list of revoked certificates stays current and the updates are applied quickly.

Data recovery agents (DRAs) and key recovery agents (KRAs): KRAs need to be defined, and updated as new roles in organizations are emerging. It is critical to define a list of users that are able to recover lost certificates, so the end-users can continue working with their previous credentials. Otherwise, there would be loss of ability to open files, access remote systems, send secure email, and so forth. It is recommended to define entities that can open encrypted files and folders, if their creators lose access to keys.

Key archival: Some organizations require a specific copy of a key pair to reside in a particular storage space on the server. If that is a requirement, the store needs to be configured, and specific access needs to be given exclusively to delegates allowed to perform key recovery scenarios.

Certificate expiration and revocation: All certificates expire, and they need to get renewed. It is easier to renew a certificate before its expiration, and such process should be initiated in a timely fashion. Typically, we do not let a certificate expire and then renew, since there could then be interruption of service. This is particularly good to know for systems for which downtime is not an option.

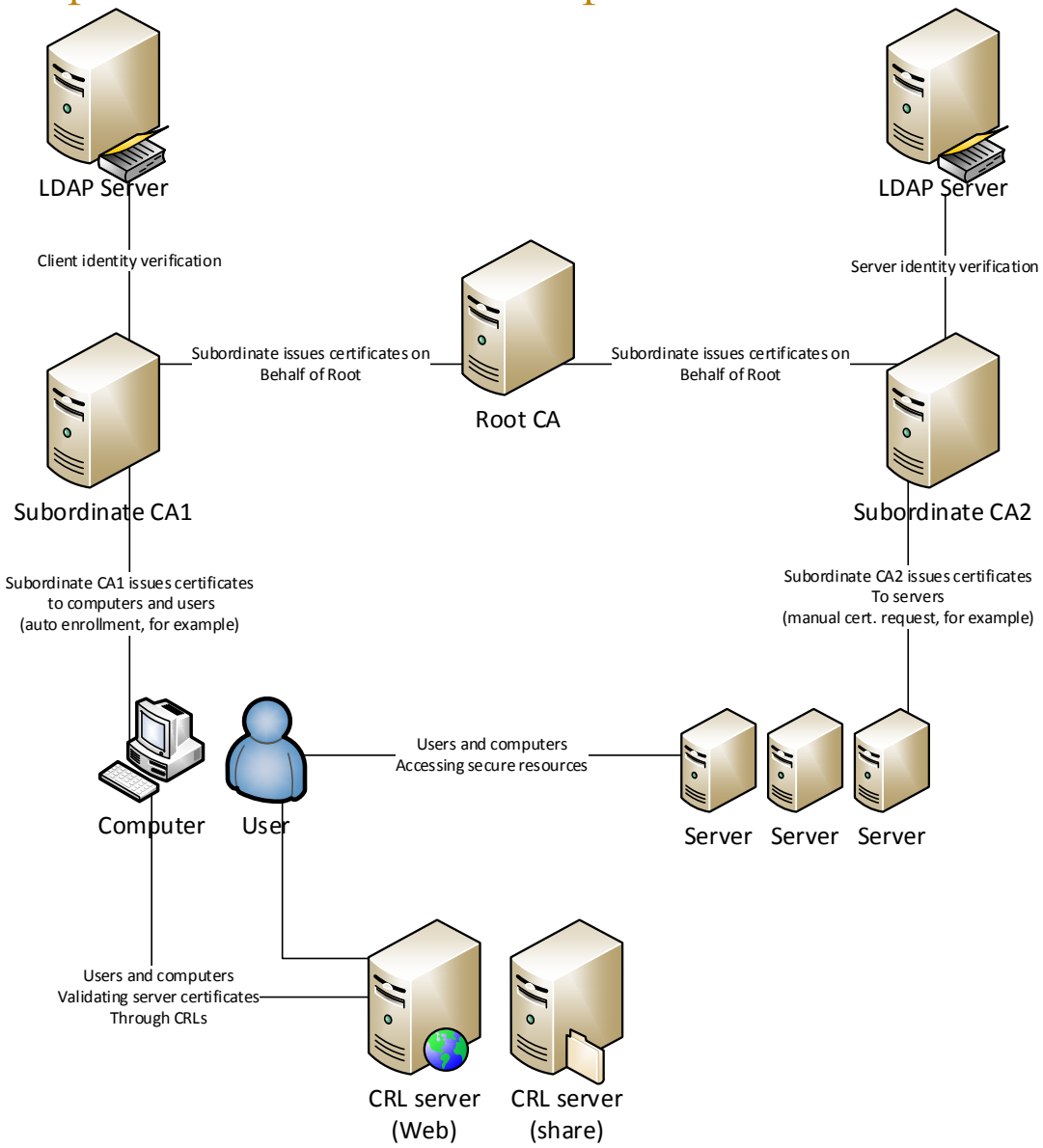
Disabling an account for which the certificate is used in case of termination is not sufficient. There is a need to monitor for which account's certificates need to be revoked or suspended. Note that you cannot re-activate a revoked certificate in many cases, except when the certificate is on hold (for employee's vacation, for example).

Backups: Backups are a critical part of every production system. Your CA needs to be backed up on a regular basis, especially if you are dealing with a dynamic environment (certificate templates, auto enrollment, and so forth)

Delegation of permissions: Although administration in a CA environment is typically static, it can happen when roles change, and we must in that case update the groups or users that are allowed to view or change the configuration of our PKI environment.

Certificate templates: New technologies are being tested and implemented, and for the most of these secure solutions, there could be the need to create or update certificate templates. Thus, management and updates of these templates are definitely tasks that need to be performed on a regular basis. Perhaps not at the daily level, but there could be requirements regarding a new type of VPN service that requires a completely new type of certificate that we have to create, for example.

Graphical Overview of Components' Interconnectivity



Conclusion

We have explored the basics of PKI and reviewed how cryptography is related to PKI. We now understand conceptually, how cryptography works within PKI and what key components are used to establish this kind of security within an environment. As we've seen, the foundation of the PKI is the implementation of certificates through a very hierarchical setup. An environment with a fully implemented CA infrastructure can enable auto enrollment, thus reducing the administrative overhead and automating processing in certificate issuance.

Learn More

Learn more about how you can improve productivity, enhance efficiency, and sharpen your competitive edge through training.

- Access Control, Authentication, and Public Key Infrastructure
- Security+ Prep Course (SYO-301)
- Certified Ethical Hacker v8

Visit **www.globalknowledge.com** or call **1-800-COURSES (1-800-268-7737)** to speak with a Global Knowledge training advisor.

About the Author

Boris Gigovic is a technical consultant and owner of Eccentrix doo, a well-established training center in Belgrade, Serbia. Being an active instructor for Global Knowledge, he specializes in networking courses as well as security. He is also owner of an organization providing consulting services in Quebec, Canada.

As a technical resource, Boris helps organizations maximizing their investment into technology products, as well as finding the best way to leverage key functionalities of these solutions. His areas of expertise are numerous, and include planning and designing computer networks, major infrastructure upgrade projects, administration, maintenance and support of these networks.

Being a Microsoft Certified Trainer, he delivers courses on various technologies, including Windows operating systems, messaging tools, as well as communication and collaboration platforms. Some of his time is also dedicated to security consulting (mostly security assessments), as well as VMware and Citrix software consulting.