

# A Primer on Public Key Infrastructures

## Version 15, May 2004

**M. E. Kabay, PhD, CISSP**

<mailto:mkabay@norwich.edu>

Associate Professor, Information Assurance

Program Director, Master of Science in Information Assurance

Division of Business & Management

Norwich University, Northfield, VT 05663-1035 USA.<sup>1</sup>

### **INTRODUCTION**

Public key infrastructures (PKI) are the foundation on which the digital signatures and secure Internet transactions will be built. This brief introduction to PKI touches on the main issues and provides pointers for further reading.

A Public Key Cryptosystem (PKC) uses asymmetric encryption to provide secure encryption of confidential messages and transactions, to authenticate the origin of such data, and to guarantee data integrity. For each user, two keys are generated at a time; each can decrypt only what the other encrypts; that is, each key cannot decrypt what it encrypts. One key is kept secret; the other becomes a public key known to anyone who wants to use the PKC. To send a message readable only by a specific PKC user, we can encrypt the cleartext with the recipient's public key; only the corresponding secret key can decrypt the ciphertext. Similarly, to authenticate the origin of a message, we can encrypt the cleartext or a randomized extract of the text (*a hash*) using our own secret key. Anyone can then decrypt the message using our public key--and only that key. Both of these methods also guarantee the integrity of the cleartext while the ciphertext is in transit, since any tampering with the ciphertext causes errors during verification of the digital signature or decryption of the ciphertext.

However, a PKC depends on trust. For example, in the case of a digital signature, the PKC provides proof only of the *secret key* used to sign a given document. What if a signing key were actually issued to an imposter? What if a person's secret key were compromised? The PKC can be trusted only if there is a trusted link from a public key to a known individual, organization, or device. It is the chief function of a PKI to document a trustworthy linkage between the ostensible owner of a secret key and that key.

There are many questions raised when discussing the PKI. For example,

- What should a certificate that links identity to a key pair contain?
- How should we validate a public key to prevent spoofing (impersonation)?
- How should we handle revocation of certificates?

---

<sup>1</sup> The author thanks R. G. Moskowitz [rgm@icsa.net](mailto:rgm@icsa.net), Senior Technical Director, ICSA Labs, for contributions to the original edition of this paper.

- What happens to documents signed with keys that have been revoked?
- Should organizations build their own PKI or use third-party certificates?
- Can proprietary formats for certificates lead to successful interoperability?

This overview ends with a brief summary of some of the costs and benefits of third-party PKIs versus in-house PKIs.

## **CERTIFICATES**

In the real world, we are used to many different kinds of certificates. For example, we carry passports, drivers' licenses, car registration, employee cards, membership cards, and credit cards. In cyberspace, we accomplish the same purposes as those of real-world certificates using the electronic equivalent. Ford and Baum write, "A certificate is a collection of information to which a digital signature has been affixed by some authority who is recognized and trusted by some community of certificate users. "

Any certificate, whether electronic or physical, must answer at least the following questions:

- Whom does the certificate identify?
- What is it certifying?
- Who certified the bearer's identity and the object certified?
- How is the authenticity of the certificate established?

The International Standards Organization (ISO) defined the X.500 standard for identifying individuals and other entities (e.g., companies, government departments, and even devices such as routers or printers). This standard used a simple scheme involving the country, organization, and common name to identify entities; e.g., C=US, O=ICSA, CN=Mich Kabay. However, this structure did not guarantee uniqueness. For example, a different person with the same common name could easily end up working for the same organization at the same time or in the future. More recent versions of the X.500 standard include unique identifiers; organizations can use these fields to insert such elements as employee numbers, Internet e-mail address, Internet domain name, X.400 e-mail address, X.500 directory name, Internet IP address, or any other unique, unambiguous name registered as an object.

The X.509 version 3 certificate goes beyond the X.500 standard by its provisions for unique identifiers; organizations can use these fields to insert such elements as employee numbers, Internet e-mail address, Internet domain name, X.400 e-mail address, X.500 directory name, Internet IP address, or any other unique, unambiguous name registered as an object. At this time, the X.509 certificate consists of the following fields:

- Version of certificate format
- Certificate serial number
- Signature algorithm identifier
- Certificate authority (CA) X.500 name
- Validity period (start, expiration)
- Subject X.500 name
- Subject public key info (algorithm, public key)
- Issuer unique identifier (optional and NOT recommended, name should be unique over time.)
- Subject unique identifier (optional and NOT recommended, name should be unique over time within issuer.)
- Extensions
- CA's digital signature.

What is certified by a certificate? Certificates attest to a linkage or an association, an origin or ownership. For example, a passport links a person to citizenship. A driver's license links a person to the right to drive in a given state. The car registration links and automobile, an owner, and the payment of appropriate fees to the states. An employee covered links person, and employer, and confers authority. A membership card relates a person to a club and associates that person with a member's rights. A credit card links a person, a financial institution, a credit limit, and the right to buy services or products up to that credit limit. Similarly, a certificate in a PKI links a person or device to a key pair through the signature of a certificate authority (CA).

## **THE FUNCTIONS OF ANY PKI**

The functional requirements for a PKI include the following:

- Issuing a certificate
- Locating a certificate
- Trusting a certificate
- Renewing a certificate
- Relations among CAs
- Revoking a certificate
- Supporting non-repudiation of signatures by public key systems
- Scalability
- Interoperability (sometimes).

*When a certifying authority issues a certificate*, the CA must gather all necessary information, verify the accuracy of that information, create and sign a certificate with the CA's private signing key, send the certificate to its owner, optionally deposit a copy of the new certificate in a public repository, archive the certificate, and record an entry in an audit trail.

*What defines the authenticity of a certificate?* Users of any certificate trust the procedures used to authenticate the identity of the certificate error. For example, the passport office demands other documents, photographs signed by a photographer and salon. This kind of procedure implies a *chain of trust*; for example, a passport official must trust the signature of the photographer and must trust the authenticity of a birth certificate. The chain of trust assumes that authenticate documents are difficult to forge; in the real world, documents are printed in special formats on recognizable unique paper, have serial numbers that can be verified and a variety of stamps that are presumed to be difficult to duplicate without authorization. In cyberspace, a Public Key Cryptosystem guaranties strong authentication as long as the confidentiality of the signing key is protected. However, there is always a possibility of fraud; for example, a stolen passport can be modified by criminals. In the PKC, all confidence in a certificate is lost if the signing key used to authenticate the certificate is divulged.

*Authenticating a certificate* in a PKI requires access to the signer's public key. The user can either store all public keys needed in a single list or--more likely when documents from strangers have to be authenticated--there has to be a system of key servers that can provide public keys securely on demand. As the number of users grows, managing public keys becomes more onerous; therefore, a fully developed PKI naturally involves a directory structure for locating the right keys.

*Trusting a certificate* always involves following a chain of authentication. For example, suppose Alice and Charlie do not know each other but both know and trust Bob. If Bob signs Alice's public key, Charlie can verify Bob's digital signature on Alice's key; if Charlie assumes that Bob took proper precautions before signing Alice's key, then Charlie can trust Alice's public key. The same principle applies to trusting a passport, a driver's license, and so on.

The chain of trusted signatures can be arbitrarily long. Suppose Dave's key were signed by Alice. Would Charlie trust Dave's key? Yes, since Dave's key was signed by Alice and Alice's key was signed by Bob, and Charlie trusts Bob. This process is known as following the trail of trust. It occurs regardless of the particular structure of a PKI hierarchy of signers.

There are several models for the relationships among CAs. The most common thus far is the top-down hierarchical structure. Certification flows only downward in this structure; all certificate users must know and trust the top-level CA public key. The main advantages of this model are that there is one and only one certification path for each entity; it is easy to locate these paths; and the structure is consistent with naturally hierarchical organizations and systems. Examples of top-down hierarchical CA structures include

- SET (Secure Electronic Transactions) for VISA & MasterCard
- U.S. Department of Defense MISSI (Multilevel Information Systems Security Initiative)
- PEM (Privacy Enhanced Mail, which failed to be accepted as a standard).

A major implication of the top-down hierarchy is that protection of the signing keys becomes increasingly important at higher levels of the hierarchy. A compromised high-level key could destroy trust in a very large number of certificates.

In practice, the top-down hierarchy will likely not be implemented worldwide--or at least, not soon (some say, not without a world revolution); because of national sovereignty considerations, few governments would accept a single root certificate authority that would certify national CAs. Because of the reluctance to have national root CAs certified by a single international authority, it is likely that the global PKI will consist of many independent hierarchies. To enable interoperation, it is likely that national CAs cross-certify — that is, will mutually sign — each other's public keys.

Another model for strengthening trust in public keys is the *web of trust* promoted by PGP users. PGP (Pretty Good Privacy) is an encryption tool originally invented by Philip Zimmermann as freeware in the early 1990s and now being sold by Network Associates, Inc. PGP users depend on a manual process for signing each other's public keys. In this model, there is no centralized signing function. The software allows keys to be marked with four different levels of trust. However, there is no formal accountability for bad decisions. A single careless certification of a public key could allow malefactors to hijack other people's identities by signing false public keys.

In yet another function for a PKI, *it may be necessary to stop trusting a certificate*; for example, the secret signing key might be compromised or there could be a change of relationship between the certificate bearer and the CA. Announcing that a data certificate is no longer valid is called revocation. In addition to revocation for cause, most certificates have a limited period of validity as defined by policy; typically the longevity of a certificate is measured in months to years and is indicated in a field of the certificate itself.

Both the bearer (owner, subject) of a certificate and the certifying CA must be able to revoke a certificate. However, revocation in itself would count for little without some mechanism for distributing news of this revocation. Therefore, a PKI distributes a list of revocations--the certificate revocation list, or CRL. The CRL is published periodically and can be distributed by posting for pull access or through a directory tree just like certificates. With this mechanism in place, every verification of the digital signature includes comparison with a recent CRL. How often this CRL is issued is a matter of policy and depends on the criticality of the certificate.

*Even without revocation, eventually a key pair expires.* How can digital signatures with the old key pair be verified in the future? What if an encrypted message were stored that required an expired decryption key? A PKI must include provisions for archiving expired public keys. Such keys are necessary in cases of litigation for verification of historical audit records.

Unfortunately, there are still many questions that have no consensus. For example, how long should keys be kept (according to the American Bar Association, probably around 30 years)? How would one know if a stored key were altered (we need methods for testing the validity of ancient signatures)? What would happen if the encryption algorithms in use were to be shown to be crackable?

*Non-repudiation* is familiar from the legal arena; we are used to the legal requirement of accepting responsibility for contracts that we have signed. Similarly, a PKI provides a basis for insisting that a document signed by a particular private key represents acknowledgement by the private key owner. Repudiation of such a signature necessarily implies immediate revocation of the corresponding public key. So non-repudiation depends entirely on the security of the private key; if anyone else has a copy of the private signing key, the owner can legitimately repudiate her signature. Therefore, no one should make a copy of a signing key unless its safety is as strongly protected as that of the original key. This principle also implies a distinction between a signature key and an encryption key.

*Scalability* is the capacity for acceptable performance regardless of size. One of the problems of growth for a PKI is finding a way of verifying a certificate sent by another user; another problem is trusting an increasing number of CAs to follow strict certification policies. In particular, with many governments in the world today consisting of the equivalent of armed criminal conspiracies, national signing authorities cannot automatically be trusted. Scalability is supported by an effective directory structure permitting verification of any public key or certificate and by effective standards enforced by complete records and meticulous audits.

Worldwide interoperability is a long-term goal for e-commerce and communications. However, some PKI hierarchies are deliberately kept from interoperating (e.g., SET) to make it harder for subversion of their certificates. PGP is not interoperable.

## **PKI SERVICES VS IN-HOUSE PKIs**

One of the biggest questions PKI users have to answer is whether to use a certification system implemented within their organizations or to use its third-party verification instead. An external PKI has some advantages. The commercial company typically claims to provide 100% availability of operations with high scalability. Certificates are purchased, with volume-based pricing available. The CA functions reside largely in secure facilities with disaster recovery plans, bonded personnel, and independent auditing of the certification procedure. The commercial PKI can offer key backup and recovery, support for secure key escrow of encryption keys (if anyone chooses to trust such escrow), revocation management, multiple directory standards, and interoperability (assuming customers push for this feature). Costs vary widely; some published figures range from less than \$1 per year per user for 500,000 users up to around \$16 per year per user for 5,000 users. All of these estimates need to factor in the costs of identification and authentication (I&A) technology, which can run from dozens to hundreds of dollars per user in initial costs (with lower on-going costs) depending on the technology used.

The alternative is to create and manage one's own PKI. The purchaser is responsible for buying software for every client system. There is a potentially heavy investment in special hardware and software. According to some analysts, rolling your own PKI is suitable for organizations who are sure of their direction and do not want to pay per-user fees to a third-party. Costs of in-house PKI are more difficult to estimate; some published studies have found costs of about \$7 per year per user for 500,000 users and up to around \$70 per year per user for 5,000 users. On the other hand, the costs of identification and authentication for in-house access to the PKI are integrated into I&A that the organization has to provide anyway for controlled access to its other information systems. In contrast, equally believable reports have found in-house PKI more economic than third-party services. We need better data.

## **CONCLUSION**

Public Key Infrastructures will soon become a ubiquitous feature of cyberspace. Public and private organizations will be forced to choose among competing models to create an interoperable basis for secure interactions in all spheres of e-commerce and communications.

## **LIST OF SOME PKI SUPPLIERS (VENDORS AND OTHERS)**

Baltimore Unicert CA  
Belsign (Belgium)  
Celo  
CertCo  
Digital Signature Trust  
Entegrity  
Entrust PKI  
Eurotrust  
GTE CyberTrust  
IBM Vault registry  
iD2  
Lockstar  
Lotus  
Microsoft Certificate Server  
Netrust (Singapore)  
Netscape Certificate Server  
NIST's MISPC  
PGP Certificate Server  
RSA Keon Certificate Server  
South African Certificate Agency  
Spyrus S2CA  
Telecash (Germany)  
U. S. Postal Service  
Valicert  
Verisign  
XCert Sentry CA



## FOR FURTHER READING

### Books

Bosworth, S. & M. E. Kabay (2002), eds. *Computer Security Handbook, 4th Edition*. Wiley (New York). ISBN 0-471-41258-9. 1184 pp. Index.

Ford, W. & M. S. Baum (1997). *Secure Electronic Commerce: Building the Infrastructure for Digital Signatures and Encryption*. Prentice Hall (Upper Saddle River, NJ). ISBN 0-13-476342-4. xxv + 470. Index.

Garfinkel, S. & G. Spafford (1997). *Web Security and Commerce*. O'Reilly & Assoc (Sebastopol, CA). ISBN 1-56592-269-7. 483 pp. Index.

Ghosh, A. K. (1998). *E-Commerce Security: Weak Links, Best Defenses*. Wiley (New York). ISBN 0-471-19223-6. xv + 288 pp. Index.

Khare, R. (1998), ed. *Web Security: A Matter of Trust* [World Wide Web Journal 2(3)]. O'Reilly (Sebastopol, CA). ISSN 1085-2301, ISBN 1-156592-329-4. ix + 272 pp.

RSA (1999). RSA Laboratories' Frequently Asked Questions About Today's Cryptography.  
<http://www.rsasecurity.com/rsalabs/faq/questions.html>

Schneier, B. (1995). *Applied Cryptography: Protocols, Algorithms, and Source Code in C, Second Edition*. John Wiley & Sons (New York). Hardcover, ISBN 0-471-12845-7, \$69.95; Softcover, ISBN 0-471-11709-9. xviii + 618. Index.

Schneier, B. (2000). *Secrets & Lies: Digital Security in a Networked World*. Wiley (New York). ISBN 0-471-25311-1. xvii + ~400. Index.

Stein, L. D. (1998). *Web Security: A Step-by-Step Reference Guide*. Addison-Wesley (Don Mills, ON). ISBN 0-201-62489-9. 448 pp.

Tipton, H. F. & M. Krause (2000), eds. *Information Security Management Handbook, 4th edition*. Auerbach (Boca Raton, FL). ISBN 0-8493-9829-0. xiii + 711. Index.

### Useful Web links (please report broken links to [mkabay@norwich.edu](mailto:mkabay@norwich.edu))

A Solution to Open Standard of PKI. [http://www.cs.cmu.edu/~qihe/paper/open\\_solution/](http://www.cs.cmu.edu/~qihe/paper/open_solution/)

Baltimore Products: PKI-Plus. <http://www.baltimore.ie/products/pki-plus/>

Building a Corporate Public Key Infrastructure. <http://www.infoseceng.com/corppki.htm>

IAL - Initiatives: Common Data Security Architecture — CDSA Specifications.  
<http://www.intel.com/ial/security/specifications.htm>

Cylink Resource Library. <http://www.cylink.com/library/library.htm>

Design Issues in a PKI. <http://www.dstc.qut.edu.au/MSU/projects/pki/Design/PKIPaper-cover.html>

US DoD PKI (1998). [http://www.telestrat.com/oss\\_up/640/harris/sld001.htm](http://www.telestrat.com/oss_up/640/harris/sld001.htm)

ENTRUST links to White Papers. <http://www.entrust.com/resourcecenter/>

Govt Canada PKI Secretariat. [http://www.cio-dpi.gc.ca/pki/pki\\_index\\_e.html](http://www.cio-dpi.gc.ca/pki/pki_index_e.html)

Intel and RSA to Accelerate Delivery of New PC and Application Security Products.

<http://www.intel.com/pressroom/archive/releases/cn11899b.htm>

Non-Repudiation in the Digital Environment (Adrian McCullagh & William Caelli).

[http://firstmonday.org/issues/issue5\\_8/mccullagh/](http://firstmonday.org/issues/issue5_8/mccullagh/)

PKI Law - Public Key Infrastructure. <http://www.pkilaw.com/>

PKI White Paper (VeriSign). <http://www.verisign.com/whitepaper/enterprise/difference/index.html>

Public Key Infrastructure (PKI) Technology (NIH slide show, 1998).

<http://www.oir.nih.gov/seconf/t1s1/tsld001.htm>

SearchSecurity.com — enter “PKI” in search field on <http://www.searchsecurity.com>

UMS Public Key Infrastructure (PKI) Analysis. <http://www.cusys.edu/~security/pki/dtreport/>

# Appendix: Enabling a New PGP Key

## ***Maintaining the Web of Trust***

I received an e-mail message from an old friend today (as I write this text); he said that he had lost his PGP keyrings and had to generate a new keypair, so here was his new public key.

You will recall that Pretty Good Privacy (see < <http://www.pgp.com/products/dtop-security-data/default.asp> > for more information about the commercial version and < <http://www.pgp.com/products/freeware/default.asp> > for the free non-commercial version) generates two keys at a time (a key pair) that are complementary: what one key encrypts, the other decrypts and vice versa. One of the keys is made public; the other is kept secret by its user. This asymmetric encryption algorithm makes possible the public-key cryptosystem and that is very useful indeed.

Suppose you want to send a message securely using PGP (or any other public-key cryptosystem) so that only the desired recipient can read it: what do you do? Answer: you encrypt the message with the recipient's public key; then only the recipient knows the secret key with which to decrypt that message. (Actually it's a bit more complicated than that – the message gets encrypted with a temporary key – a session key – and then that key gets encrypted using the recipient's public key and sent along with the encrypted message.) The encryption process also ensures that the recipient can verify the integrity of the message: any change to the ciphertext – as the encrypted text is known – is detected during the decryption phase.

Now suppose you want to send a non-confidential message to one or more recipients; you want to maintain proof of integrity and you want your correspondents to be sure the message actually came from you. PGP generates a function of your message called a hash; it then encrypts the hash using your own private (secret) key. After delivery, the recipient's PGP program generates the hash of your message again; it also decrypts the hash you sent along with your message using your public key. If your decrypted hash matches the recipient's hash, then (a) nobody changed the message in transit; and (b) the message must have come from a person with access to your secret key. To the extent that the recipients are confident in your ability to protect your secret key against unauthorized use, they can have the same confidence that the message actually came from you.

Well now we come to the reason I winced when my buddy (let's call him "Bob") sent me that new public key. How confident was I that the key genuinely came from him?

But what would it matter if someone else were sending me a key falsely claiming to come from Bob?

Suppose a man in the middle of the communications link and pretending to be Bob sent me a key falsely claiming it was Bob's public key? Then later, he (let's call him "Darth") could impersonate Bob and read encrypted messages from me to "Bob" and read them. If Darth could intercept PGP-signed messages from Bob to me (and prevent my receiving the authentic ones from Bob) then Darth could alter them before sending them on to me with a signature using the false "Bob" key. Darth could also generate completely fraudulent messages claiming to be from Bob and my PGP signature verification would tell me that they were authentically from what I incorrectly thought was Bob's private key.

Now Bob had actually signed his own public key using his secret key, but the valid signature only proved that the public key had been signed by the corresponding private key; it in no way guarantees the authenticity of that keypair as really coming from Bob.

Solution: the recipient of a new public key must establish to the desired level of confidence that it really comes from the putative sender. In my case, it was good enough to call Bob using the phone number I already knew from our previous interactions (not a phone number in the message bearing the new key).

When I was confident that I was speaking with Bob, I asked him to bring up his new PGP key in the PGPKKeys program. By looking at the properties of the keys on his computer and on mine, we could verify that the fingerprint on his version of the key matched the fingerprint of the version of the key I had received by e-mail. He read me the fingerprint; it matched what I saw, so I was satisfied that the key was genuine.

Going one step further, I signed Bob's key using my own PGP secret key. Anyone trusting me would know that I had established to my own standards of rigor that the key was legitimate. If these third parties decided to trust my judgement, they could then use Bob's signed public key without having to check it further. Thus my verifying that fingerprint was an essential element in the non-hierarchical web of trust that underlies PGP and similar public-key cryptosystems.

Going one step further, I signed Bob's key using my own PGP secret key. Anyone trusting me would know that I had established to my own standards of rigor that the key was legitimate. If these third parties decided to trust my judgement, they could then use Bob's signed public key without having to check it further. Thus my verifying that fingerprint was an essential element in the non-hierarchical web of trust that underlies PGP and similar public-key cryptosystems.

Should someone who had no idea who I am trust my signature on Bob's public key? Not necessarily. They could look at who signed my public key (the public key should be stored on a public keyserver for access by anyone) and if they saw a valid signature from someone whom they did trust, then maybe they could hope that I would maintain the same level of trust. However, none of this provides formal guarantees of trustworthiness. It's an informal web of trust and it works only as well as the honesty and care of the people involved. At a fundamental level, exactly the same issues of probity and trustworthiness underlie other mechanisms for defining the level of trust in any public key infrastructure.

Anyone critically concerned with the validity of a public key can check its fingerprint by contacting the owner of that key before accepting its authenticity.

Finally, I gave my friend some good-natured ribbing: there is no reason he should have lost his PGP keyrings at all. They should be backed up safely.

In summary, there are several major lessons here for anyone using PGP:

- (1) If you receive a new public key from someone you know, communicate with the ostensible owner using a trustworthy channel and check the key fingerprints before trusting the new key.
- (2) When deciding whether to trust a public PGP key, you can examine who has signed the key and check the validity of those signatures.
- (3) You may go so far as to check a proposed public key by verifying that key's fingerprint with its owner using a trustworthy channel of communication.
- (4) If you know the owner of a new PGP key personally and you have verified the key to the maximum level of confidence that you deem appropriate, you may sign the new key if you feel that others know you and trust your judgement in guaranteeing the new key's authenticity.
- (5) Back up your PGP keyrings.

