By Mike Burmester and Yvo G. Desmedt

Is Hierarchical Public-Key Certification THE NEXT TARGET FOR HACKERS?

Considering alternatives to hierarchical authentication structures that are not sufficiently secure for communication on open networks such as the Internet.

The past few years have seen a remarkable growth of computer networks, with many new groundbreaking applications such as e-commerce, e-government, and digital libraries. However, networks—and in particular large open networks—are inherently insecure. A hacker can corrupt data, steal sensitive information, or masquerade as another user. The authenticity and/or integrity of data is essential for commercial transactions. To protect data one may use cryptographic mechanisms such as digital signatures (see Figure 1). Digital signatures require two keys: a public key and a secret key. The signer has both. The secret key is used to digitally sign the data while the public key is made known to the receiver. The public key is used to authenticate (that is, verify the correctness and origin of) the signed data. If the signature is valid then the data is authentic, provided of course that the public key

Illustration By Walter Sipser

used for verification is the real key of the sender. If a hacker can convince the receiver that a fake key, one made by the hacker, is the public key of the sender, then the receiver will be fooled into accepting as authentic data created by the hacker.

Public-key cryptography can also be used for encryption to protect the privacy of the data. In this case the public key is used to encrypt and the secret key to decrypt. The sender must use the public key of the receiver. If a hacker can convince the sender that a fake key is the public key of the receiver, then the hacker will be able to eavesdrop on all communication intended for the receiver.

The authenticity of the public key of a user can be established with public-key certificates. These are issued by Certification Authorities (CAs). Large networks have several CAs, which may be linked in different ways. The traditional approach is to use hierarchical architectures. While this seems to be a natural and efficient approach, there are some fundamental security issues that must be addressed. Here, we survey some of these issues and propose several new solutions.

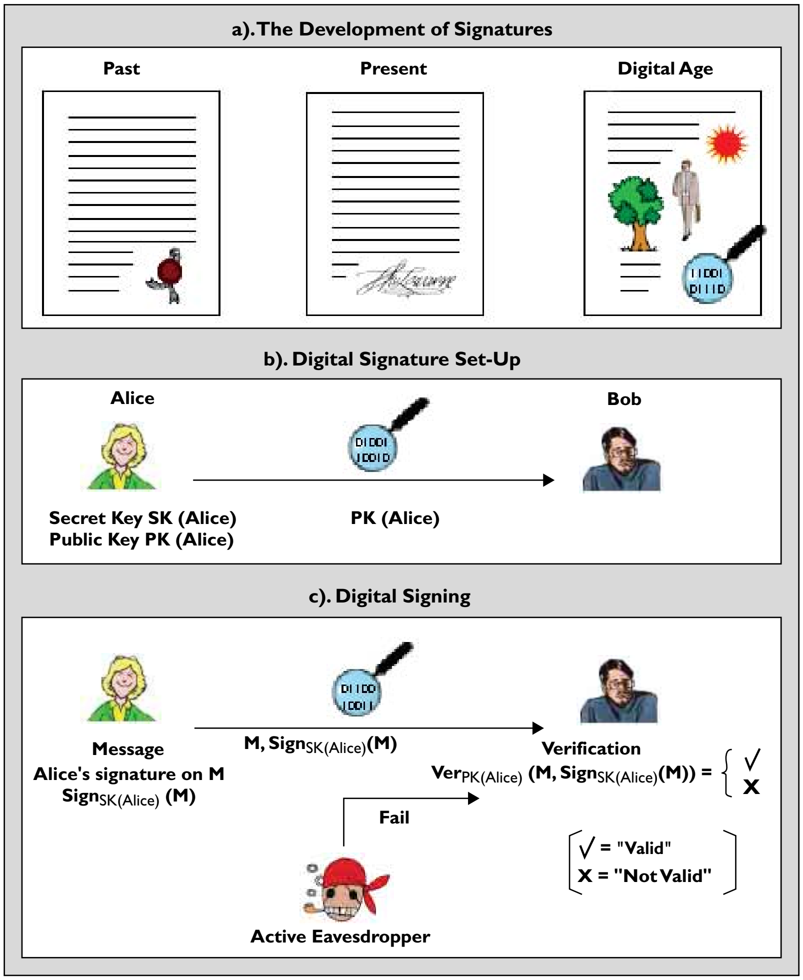


Figure 1. Digital signatures. a) The evolution of signatures to digital ones. The digital signature 1100101... is attached to a (digital) document. b) The secret key of Alice is SK(Alice); the public key PK(Alice) = 01001100 ... is made known to Bob. c) Bob verifies the signature SignSK(Alice)(M) = 01100100 ... on message M; a forged signature will fail the verification test.

Public-Key Cryptosystems and Certificates

The most popular digital signatures schemes are the RSA cryptosystem and the DSA (Digital Signature Algorithm) [10]. The RSA cryptosystem can also be used for encryption (an alternative is the ElGamal cryptosystem [10]). With public-key signatures, anyone can verify a signature but only the possessor of the secret key can sign. For encryption, anyone can encrypt, but only the possessor of the secret key can decrypt.

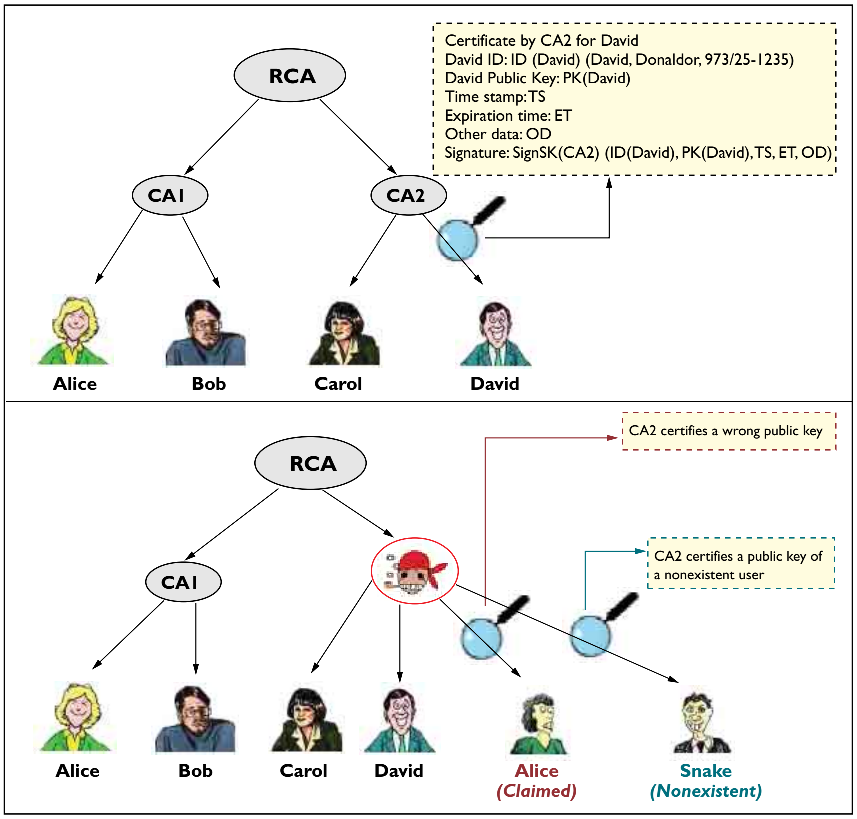
Several alternatives are available for establishing confidence in public keys in large open communication networks. These techniques employ public-key certificates and make use of key-management techniques. A public-key certificate provides a means by which public keys can be stored in insecure repositories or transmitted over insecure channels. Certificates are not usually confidential and typically do not contain sensitive information. They have a validity period and may be revoked (offline or online) by the issuing entity if required. The X509 international standard [5] is a good example of a certificate format.

Public-key certificates have two parts: data and a signature. The data contains information about the identity of an entity, the public key, the validity period, and other relevant details. The signature is a digital signature on the data by a certifying entity. For the X509 certificates this is a Certification Authority (CA), but in general it may be another user (as in PGP [10]).

Certificates are stored in a directory by the issuing entity. Usually they are either sent out upon creation or periodically, to all entities, or stored in a database from which they can be obtained. Alternatively, certificates may be sent to individual users.

Hierarchical Authentication Structures

The public-key certificates of a network define an authentication infrastructure that can be used to model the trust of the entities in the public keys. With


Certificate by CA2 for David
David ID: ID (David) (David, Donaldor, 973/25-1235)
David Public Key: PK(David)
Time stamp: TS
Expiration time: ET
Other data: OD
Signature: SignSK(CA2) (ID(David), PK(David), TS, ET, OD)
RCA
CA1
CA2
Alice
Bob
Carol
David
CA2 certifies a wrong public key
RCA
CA1
CA2 certifies a public key of a nonexistent user
Alice
Bob
Carol
David
Alice (Claimed)
Snake (Nonexistent)

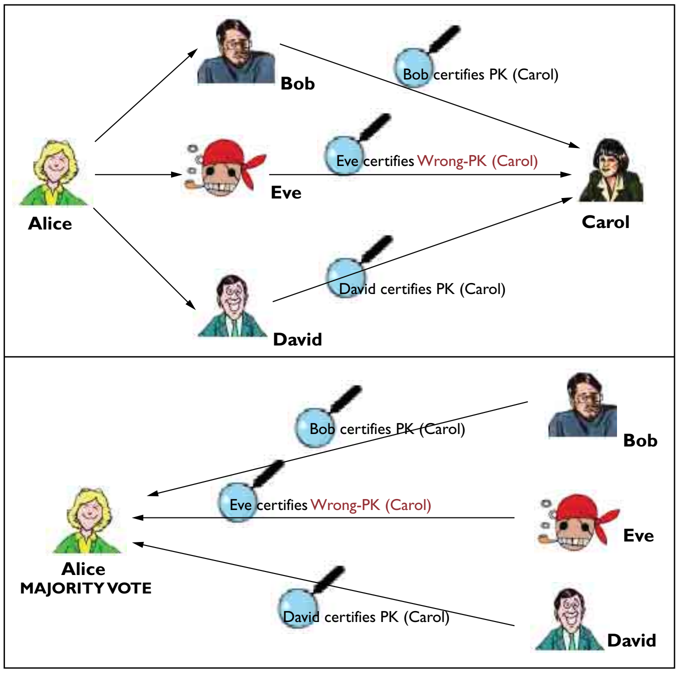

Bob
Bob certifies PK (Carol)
Eve certifies Wrong-PK (Carol)
Eve
Alice
Carol
David certifies PK (Carol)
David
Bob certifies PK (Carol)
Bob
Eve certifies Wrong-PK (Carol)
Eve
Alice
MAJORITY VOTE
David certifies PK (Carol)
David

Figure 2a. (top) A hierarchical authentication infrastructure is shown and the impact when CA2 is penetrated. Figure 2b. (right) Three disjoint trust-paths authenticating the public key of Carol (top). Alice decides the public key of Carol based on three certificates (bottom).

X509, the infrastructure is hierarchical, spanned by a tree with root, the Root Certification Authority (RCA). Figure 2 illustrates such a tree. In this case the trust is centered at the RCA, and is transferred hierarchically to all the users in the network via Certification Authorities. More specifically, the public key of the RCA is known a priori to all users, and this knowledge is used to induce confidence in the public keys of the other entities via trust-paths. For example, the trust-path authenticating the public key of Carol to Bob in Figure 2 is: RCA –> CA2 –> Carol. There is no need for the users to certify the public key of the RCA, because it is assumed this key is known to all.

For applications in which it would be unreasonable to expect that all entities trust the same RCA, one may use hierarchical authentication structures with several RCAs. The users are partitioned into domains, each one under the control of a single RCA, and the RCAs should cross-certify their keys.

How Secure is a Certifying Authority?

A CA is subject to insider and outsider attacks. Insider attacks enable the adversary to access its organization, computer system, and data. Outsider attacks involve the network system. The traditional approach for protecting an organization and its computer systems is to use a mixture of security tools and security policies [7]. This approach focuses on the reliability of the security tools and implicitly assumes the security policies will be adhered to. It is designed for networks in which the trust is concentrated in a few well-protected CAs, and does not take into account the kind of threats that are possible in open dynamic networks such as the Internet and wireless networks.

Several attacks, such as the penetration of Web sites of the FBI and the unauthorized access to the secret key of the Web server of the U.S. State Department, support the case that this approach for secure communication may be inadequate. Moreover, setting up an inherently vulnerable system such as the X509 hierarchical infrastructure will attract hackers, particularly if it is used for financial transactions. The problem with X509 is that it cannot tolerate even one penetration: each node is a single point of failure.

Hacked Certificates

An entity (a CA, or even worse an RCA) that is corrupted by an insider may issue fraudulent certificates that authenticate fake public keys. These may certify actual users or nonexistent users, as illustrated in Figure 2a. In this figure, a penetrated CA (CA2) issues two fraudulent certificates: one for the actual user Alice and one for the nonexistent user Snake.

Impact on authenticity: If a hacked certificate certifies a fake signature key of a user (Alice) then the hacker can impersonate that user. Alternatively, the hacker may acquire legitimacy for a key corresponding to a false identity (Snake's key). Impact on privacy: If a hacked certificate certifies a fake encryption key of a user then the hacker can eavesdrop on communication intended for that user (Alice).

These threats should not be underestimated. For example, in e-commerce applications the adversary may obtain sensitive information about a customer (such as credit card details) from a merchant, or impersonate the merchant. Even if we assume it is not possible to compromise a CA from the inside (many security analysts would find this hard to believe), a hacker can always exploit weaknesses in the supporting computer systems. For a list of threats and attacks on open networks, see [3]; many computer vulnerabilities are also listed at the CERT security site (see www.cert.org).

Many of these problems have been pointed out elsewhere (see [1, 4, 8]). It has been argued that, to deal with hackers, CAs should not sign online and that only the certificates (created offline) should be online. However this approach does not protect users from CAs that forfeit their obligations [4] and does not allow for online revocation. Moreover, such an online/offline approach will increase the delay in certificate updating, which may cause problems in large dynamic networks, such as wireless networks with frequent subscriber turnover.

Hierarchies: A Disaster in the Making

The hierarchy that consists of a RCA and its CAs is a clear target for hackers. If a hacker succeeds in penetrating the RCA (either from the inside, or through its computer system) then the security of the system is completely broken. A penetrated CA can compromise the public keys of all its descendants, as well as those that the hacker claims to be descendants of that CA. This situation makes hierarchical structures particularly vulnerable when they are used in open networks. A similar argument applies to the case when the structure is spanned by a forest with several cross-certified RCAs.

Solving the Problem (Trust-Graphs)

Several structures support public-key certification mechanisms [1, 2, 6, 9]. We consider these structures, and discuss their suitability for open networks, after introducing the concept of trust-graphs.

Certificates may be used to model the confidence of a network in its public keys by a directed trust-graph whose nodes correspond to the entities of the network (users and/or CAs) and whose edges correspond to certificates. An edge links node A to node B if there is a certificate in which A authenticates (digitally signs) the public key of B.

The confidence that an entity has in the public key of another entity (a CA) may be based on direct knowledge or on induced knowledge. Certificates corroborate direct knowledge: an entity will only certify the public key of another entity if it believes the key is authentic. Therefore the edges of the trust-graph reflect direct confidence. This confidence is established by noncryptographic means (by checking personal details). Induced confidence is established via trust-paths that link nodes in the trust-graph. In a trust-path each node certifies the authenticity of the public key of the next node on the path. In this manner trust is induced by a trust-chain.

The trust-graph should be distinguished from the communication network, because its edges correspond to trust relations and are not necessarily communication paths. Furthermore, the nodes of the trust-graph may not be communication nodes, as illustrated in the following example. Suppose that Bob is a friend of Alice and that Alice knows his public key. Alice may be willing to certify Bob's key, even though Bob may be located a long distance away, with no communication link.

A Horizontal Approach (Trust-Graphs with Multiple Connectivity)

If a public key is authenticated via two trust-paths with a common node (other than the end nodes), then a hacker will target this node. So the trust induced via such paths is no more than that of a single path. To increase the trust we need to have several node-disjoint (excluding end nodes) trust-paths that authenticate the same public key. The public key can then be determined by taking a majority vote over the trust-paths. Figure 2b illustrates this for three trust-paths that authenticate the key of Carol. Such an approach is successful if the number of penetrated nodes is less than half the number of disjoint trust-paths that authenticate the same key, since each penetrated node cannot be on more than one path.

A trust-graph is (2k+1)-connected if there are 2k+1 node-disjoint trust-paths that connect any two nodes. With such graphs the induced trust in public keys is distributed horizontally via at least 2k + 1 node-disjoint paths to all nodes. Attacking such structures requires the penetration of more than k nodes. It has been shown that if a trust-graph is (2k +1)-connected and if we assume hackers cannot penetrate more than k nodes, then any two entities can communicate securely [1, 2]. If the trust-graph is known, then trust in a public key can be established by majority vote over 2k+1 disjoint trust-paths, since at most k of these paths will be faulty. Figure 2b illustrates this for three trust-paths. The case when the trust-graph is not known is more complex. This is because the trust-graph may be dynamic, and its structure is modified continuously as entities leave or join the network. Furthermore, hackers may destroy certificate databases and/or entities. For survivability, given such threats, it should be possible for the remaining entities to recover enough of the authentication structure to be able to communicate securely. Of course there is a trade-off between the required security and the cost.

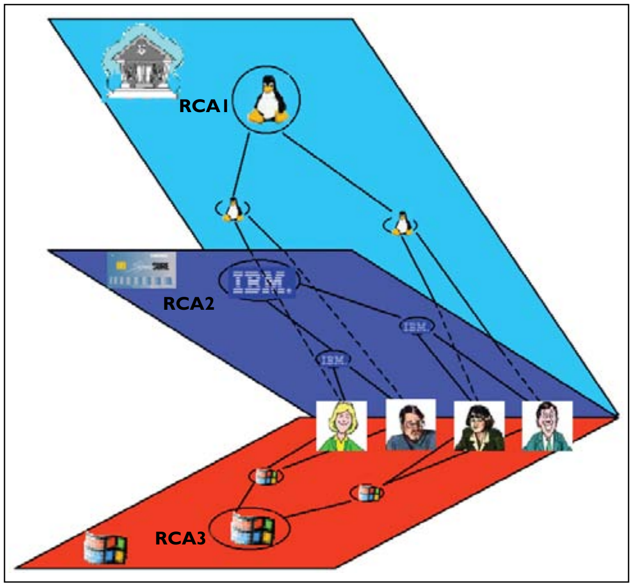


Figure 3. Combining hierarchical and horizontal infrastructures.

An Unstructured Approach

Pretty Good Privacy (PGP) [10] is an email system that uses an unstructured authentication framework. Users are free to decide whom they trust: PGP does not specify any structure for its trust-graph, and for this reason it is vulnerable. There are two major concerns. The first is that the trust-graph may not be known to legitimate users and, because there is no structure, it may be impossible for a legitimate user to construct it when the adversary is active [2]. Observe that there is no centrally trusted authority with unstructured frameworks. The second concern is that with unstructured trust-graphs the connectivity may be insufficient to endow legitimate users with enough trust to distinguish them from hackers [2] (for example, if the adversary controls too many of its neighbors).

How Many Hackers?

The number of hackers can be quite large, particularly if there are incentives. For example, if the network is used for wholesale transactions, contract signing, and similar activities. To guarantee security one may therefore have to choose a large value for the upper bound k on the number of expected penetrations.

However, this approach does not take into account attacks in which common weaknesses of network nodes are exploited by hackers. For example, a bug in the operating system of a CA. Such attacks will affect all servers that employ this operating system. Moreover they may be automated by using computer viruses and/or worms. In this case the question of how large k should be is of little relevance.

Several approaches may be used to address the common platform attacks. Here, we propose a platform-independent approach in which trust in public keys is established via paths on disjoint platforms. Figure 3 illustrates an infrastructure in which trust is established via three platform-independent paths. The number of platforms must be such that the resource required to penetrate half of them exceeds that available to the hackers. (We assume the cost of penetrating two platforms exceeds significantly that of penetrating one platform.)[1]

The argument for using platform-independent structures extends far beyond information security. For example, to critical infrastructures. Indeed, the fact that the same platform was used in the terrorist attacks of Sept. 11, 2001 (airports with similar security policies, similar aircraft, similar weapons) made it easier for the terrorists to carry out their attacks.

Conclusion

A secure authentication infrastructure must be reliable, robust, and survivable. Reliability deals with faults that occur in a random manner, and is achieved by replication. Robustness deals with malicious (Byzantine) faults, and survivability deals with the destruction of parts of the structure. The destruction may affect entities (for example, CAs) as well as stored data, and may be malicious. For survivability, the remaining entities should be able to recover enough of the authentication infrastructure to communicate securely. Robustness and survivability are assured by using horizontal authentication structures (provided the number of expected penetrations is bounded).

---

[1] It is not necessary to use disjoint-platform certificate paths as in Figure 3, however, such details go beyond the scope of this article.

Clearly there is a trade-off between the security requirements and the complexity of an authentication infrastructure. Hierarchical structures sacrifice security for managerial convenience. They achieve efficiency by single-path authentications. Hierarchical structures also are less expensive. Indeed, 2k + 1 certificates are needed for a robust approach. However, hierarchical structures are vulnerable to single penetrations. Moreover, a single sloppy CA can do serious damage. Therefore, several organizations are setting up their own public-key infrastructures. Taking this into account, the extra cost to set up a structure as proposed in Figure 3 may not be too excessive and the degree of security obtained is higher. **C**

REFERENCES

1. Burmester, M., Desmedt, Y., and Kabatianski, G. Trust and security: A new look at the Byzantine generals problem. *Series in Discrete Mathematics and Theoretical Computer Science 38*, AMS, 1998.
2. Burmester, M. and Desmedt, Y. Secure communication in an unknown network using certificates. In *Proceedings, Advances in Cryptology* (Asiacrypt'99), Springer, 1999.
3. Denning, D.E. and Denning, P.J. *Internet Besieged*. ACM Press, NY, 1998.
4. Ellison, C. and Schneier, B. Ten risks of PKI: What you're not being told about. *Computer Security Journal 16*, 1 (Jan. 2000).
5. ISO/IEC 9594-8. *Information Technology Open Systems Interconnection*. International Organization for Standardization, Geneva, Switzerland, 1995.
6. Maurer, U. Modeling public-key infrastructure. In *Proceedings, Computer Security–ESORICS 96*, LNCS 1146, Springer, 1996.
7. *Trusted Computer System Evaluation Criteria (TCSEC)*. U.S. Department of Defense, 1985, 5200.28-STD (Orange Book).
8. Reiter, M.K. and Stubblebine, S.G. Path independence for authentication in large scale systems. In *Proceedings of the 4th ACM Conference on Computer and Communications Security*, (1997), 57–66.
9. Rivest, R.L. and Lampson, B. *SDSI: A Simple Distributed Security Infrastructure*; theory.lcs.mit.edu/~cis/sdsi.html.
10. Schneier, B. *Applied Cryptography*. Wiley, NY, 1996.

MIKE BURMESTER (burmester@cs.fsu.edu) is a professor in the Department of Computer Science at Florida State University in Tallahassee.

YVO G. DESMEDT (desmedt@cs.fsu.edu) is a professor in the Department of Computer Science at Florida State University in Tallahassee and a visiting professor of Information Security at Royal Holloway, University of London.

Research for material appearing in this article was funded in part by DARPA F30602-97-1-0205, by EPSRC GR/23301, and by NSF 0209092.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

© 2004 ACM 0001-0782/04/0800 $5.00