Mike Burmester and Yvo G. Desmedt

Hierarchical public-key Certification: The Next Target for Hackers?

Several organizations are setting up hierarchical authentication structures for secure communication. We argue that such structures are not sufficiently secure for open networks such as the Internet and discuss alternatives.

The past few years have seen a remarkable growth of computer networks, with many new applications such as electronic commerce, e-government, digital libraries, etc. 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. Digital signatures require two keys: a public 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 of) the signed data. See Figure 1 for an illustration. If the signature is valid then the data is authentic, provided of course that the public-key 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. 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. In this report we overview some of these issues and propose several new solutions.

Public-key Cryptosystems and Certificates

The most popular digital signatures are the RSA (Rivest–Shamir–Adleman) cryptosystem and the DSA (Discrete Signature Algorithm) [11]. The RSA cryptosystem can also be used for

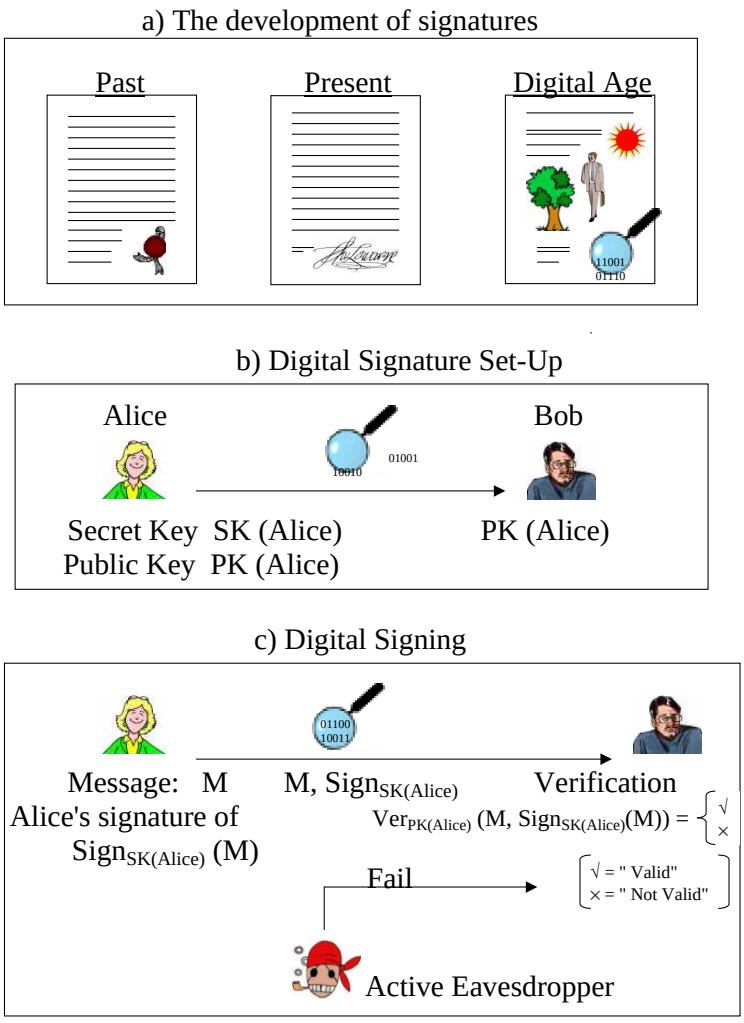


Figure 1: Digital signatures. a) A digital signature 1100101... attached to a (digital) document. b) The secret key of Alice is SK(Alice) and the public key PK(Alice) = 01001100... is made known to Bob. c) Bob verifies the signature $\text{Sign}_{\text{PK(Alice)}}(\text{M}) = 01100100\ldots$ on message M; a forged signature will fail the verification test.

encryption (alternatively, the ElGamal cryptosystem [11] may be used).

With public-key signatures, anyone can verify a signature but only the possessor of the secret key can sign. For encryption it is the other way round: anyone can encrypt, but only the possessor of the secret key can decrypt.

Several alternatives are available for establishing confidence in the public-keys in large open communication networks. These 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 do not contain sensitive information. They have a validity period and may be revoked by the issuing entity if required. The X509 international standard [5] is an example of 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 [11]).

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* which can be used to model the trust of the entities in the public-keys. With X509, this infrastructure is hierarchical, spanned by a tree with root, the *Root* Certification Authority (RCA). Figure 2a 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 2a is: RCA $\rightarrow$ CA2 $\rightarrow$ Carol. There is no need for the users to certify the public-key of the RCA, because it is assumed that it 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 cross-certify their keys.

How secure is a Certifying Authority

A Certification Authority 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

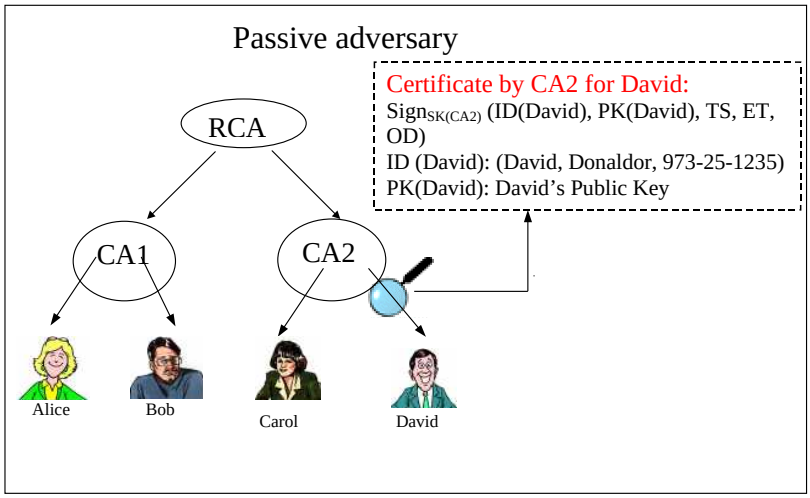


Figure 2: a) A hierarchical authentication infrastructure. b) The infrastructure under attack.

systems is to use a mixture of security tools and security policies. This approach focuses on the *reliability* of the security tools and implicitly assumes that the security policies are *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 ad hoc networks.

Several recent attacks, such as the penetration of the Web sites of the Interior Department, the Senate, the FBI, and the hacking of the secret key of the State Department support the case that this approach to secure communication may be inadequate.

Setting up an inherently vulnerable system such as the X509 hierarchical infrastructure will attract hackers due to the importance of such structures in e-commerce and similar applications (*e.g., e-credentials*). The problem with X509 is that it cannot tolerate even one penetration: each node is a single point of failure.

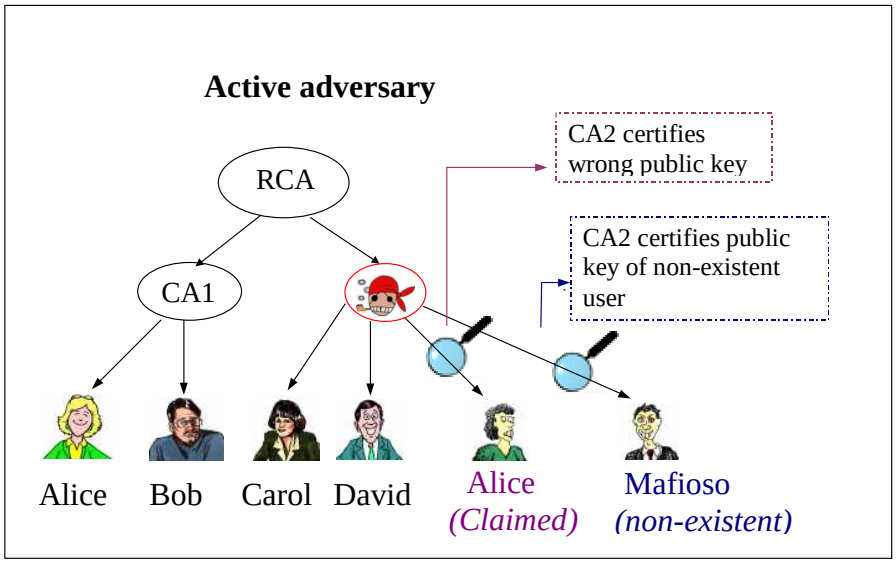
Hacked Certificates

An entity (*e.g.* a CA, or even worse an RCA) which is penetrated either at the organizational level (*e.g.* by an insider) or through its computer system, may issue fraudulent certificates that authenticate fake public-keys. These may certify actual users, or non-existent users. Figure 2b illustrates this. A penetrated Certifying Authority CA2 issues two fraudulent certificates: one for the actual user Alice and one for the non-existent user Mafioso.

Impact on authenticity: If a hacked certificate certifies a fake signature key of a user (e.g., Alice) then the hacker can impersonate that user. Alternatively, the hacker may acquire legitimacy for that key (e.g., Mafioso's key).

Impact on privacy: If a hacked certificate certifies a fake encryption key of a user then the hacker can eavesdrop on all communication intended for that user (e.g., Alice).

These threats should not be underestimated. For example, in e-commerce applications the



adversary may obtain sensitive information about a customer (*e.g.*, credit card details) from a merchant, or impersonate the merchant. Even if we assume that it is not possible to compromise a CA at the organizational level (many security analysts would find it hard to believe this), a hacker can always exploit weaknesses in the supporting computer systems, in particular with wireless or ad hoc networks. For a list of threats and attacks on open networks the reader is referred to [3]. Many computer vulnerabilities are also listed at [Cert: http://www.cert.org/]

Such problems have been pointed out elsewhere (see [9,1] and also [4]). 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 against CAs that forfeit their obligations, including the required security checks [4]. Moreover, such an online/offline approach will increase the delay in certificate-update which may cause problems in large dynamic networks, such as wireless networks with a fast subscriber turnover.

Hierarchies: a disaster in the making

The hierarchy consisting of the RCA and all the CAs is a clear target for hackers. If a hacker succeeds in penetrating the RCA (either at the organization level, or through its computer system) then the security of the system is totally broken. If this is not possible, hackers may concentrate on CAs that are lower down in the hierarchy. A penetrated CA will compromise the public-keys of all its descendants, and also those that the hacker *claims to be* descendants of the CA. This makes hierarchical structures very vulnerable when used in open networks. A similar argument applies to the case when the trust-graph is spanned by a forest with several cross-certified RCAs.

Solving the problem – trust-graphs

Several structures will support public-key certification mechanisms. We consider these, and discuss their suitability for open networks. First we introduce the concept of trust-graphs.

Certificates 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 Certification Authorities) and edges correspond to the certificates. An edge links node A to 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 (*e.g.,* 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 if it believes that its key is authentic. Therefore the edges of the trust-graph reflect direct confidence. This confidence is established by non-cryptographic means (*e.g.,* 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. The following example illustrates this. Suppose that Bob is a friend of Alice and that Alice knows his public-key. Then Alice may be willing to certify Bob's key, even though Bob may be far away on a long journey, with no communication link between them.

A horizontal approach – trust-graphs with multiple connectivity

If a public-key is authenticated via two trust-paths which have a common node (other than the end-nodes), then a hacker will target this common 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 which authenticate the same public-key. The public-key can then be determined by taking a majority vote over the trust-paths. Figure 3a illustrates this for 3 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 $2k+1$ 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 that hackers cannot penetrate more than k nodes, then any two entities can communicate securely [1,2]. If the trust-graph is *known*, then a public-key is certified via $2k+1$ node-disjoint trust-paths. Since at most k of these disjoint trust-paths are faulty, voting will allow parties

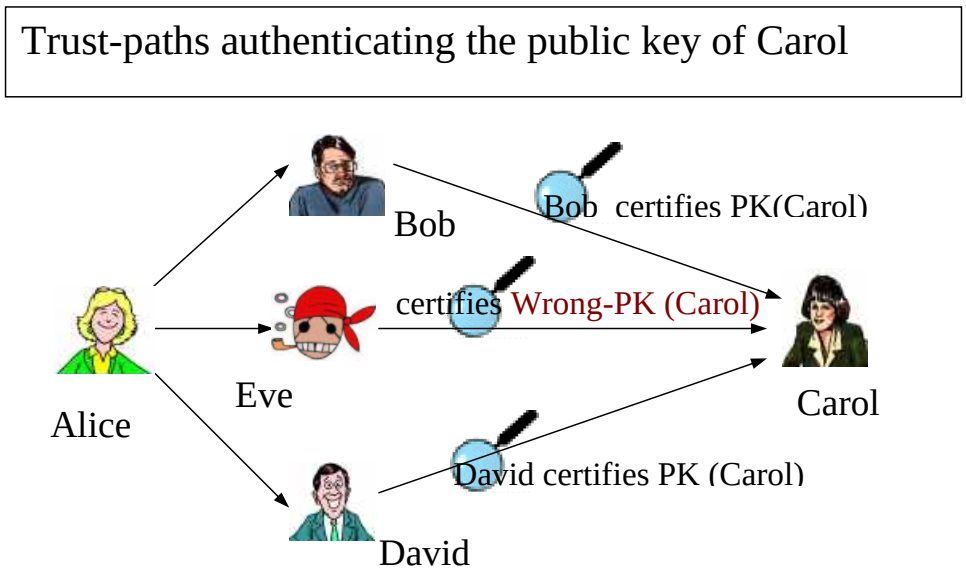
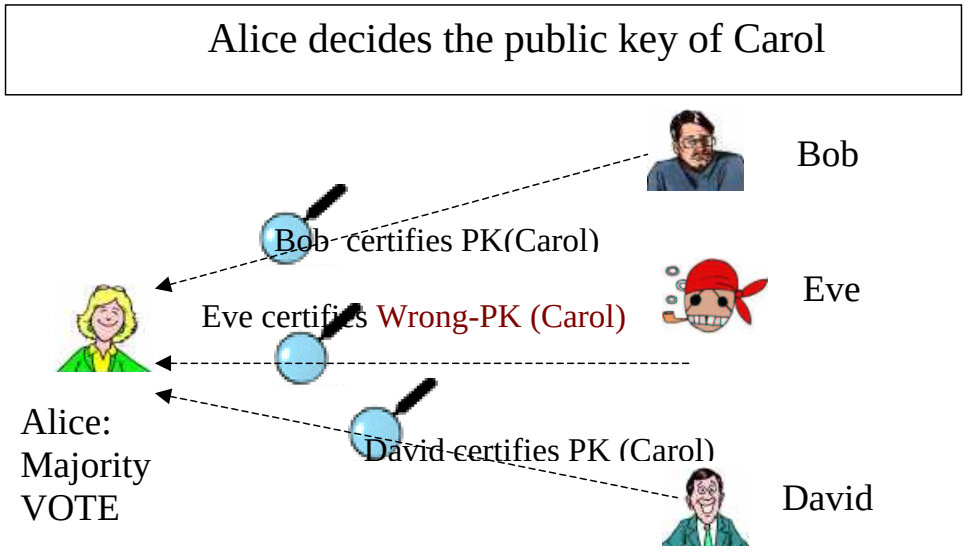


Figure 3: a) Three disjoint trust-paths authenticating the pubic-key of Carol. b) Alice decides which public-key is Carol's.

to obtain the correct public-key. Figure 3b illustrates this for 3 trust-paths: Alice decides that PK(Carol) is the real key of Carol. The case when the trust-graph is *not known* is more complex. This is because the trust-graph is dynamic, and is modified continuously as entities join or leave the network. Furthermore, a hacker may destroy some certificate databases and/or entities. For survivability, 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 its cost. We shall discuss this issue shortly.

An unstructured approach

Pretty Good Privacy (PGP) [11] is an electronic mail 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. 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 is that with unstructured trust-graphs the connectivity may be insufficient to endow legitimate users with enough trust to distinguish them from hackers [2].



How many hackers?

The number of hackers can be quite large, particularly if there is an incentive to penetrate the system. If the network is used for financial transactions, wholesale transactions, contracts, etc, then clearly one should expect the number of attempted penetrations to increase significantly. This raises the question of how large should we choose the upper bound k for the number of expected penetrations.

A bug in the operating system of a Certification Authority (or a fault at the organization level) may be exploited by the hackers. Such an attack will effect *all* servers that employ this operating system. Moreover such an attack may be automated using computer viruses and/or worms. In this case the question of how large k should be chosen is of little relevance.

Several approaches can be used to address such attacks. In this report we consider one which is based on *platform independent* infrastructures, for which communication is via trust-paths on disjoint platforms. Figure 4 illustrates an infrastructure in which trust is established via three platform independent trust-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 that the cost of penetrating two platforms is sufficiently larger than that of a penetrating one.).[1]

The argument for using independent-platform structures extends beyond information security. For example, to critical infrastructures. Indeed, the fact that the same platform was used in the tragic events of September 11th, 2001 (airports with similar security policies, airplanes of a similar type, similar weapons, etc) made it much easier for the terrorists to carry out their

[1] It is not necessary to use disjoint-platform certificate paths as in Figure 4 – however such details go beyond the scope of this discussion.

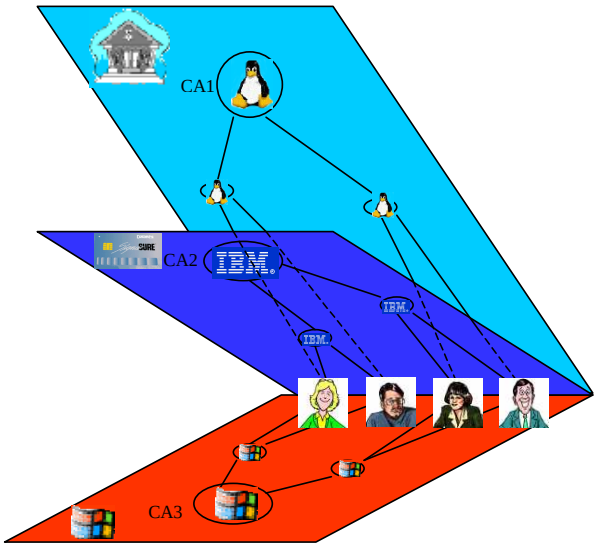


Figure 4: Combining hierarchical and horizontal infrastructures.

attack.

Discussion

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 maliciously faults (Byzantine) and survivability with the destruction of parts of the structure. The destruction may affect entities (*e.g.* the 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).

Clearly there is a tradeoff between the security requirements and the complexity of a communication system. Hierarchical structures sacrifice security for managerial convenience. By authenticating via single paths they are very efficient. However they are vulnerable to single penetrations. By having multiple connectivity one can design scalable communication systems that are reliable, robust and survivable.

References

1. M. Burmester, Y. Desmedt, and G. Kabatianski. Trust and security: A new look at the Byzantine generals problem. *Network Threats, DIMACS, Series in Discrete Mathematics and Theoretical Computer Science vol 38*, AMS, 1998.

2. M. Burmester and Y. Desmedt. Secure communication in an unknown network using certificates, Proceedings, *Advances in Cryptology – Asiacrypt'99,* **1716**, Springer, 1999, pp. 274–287.
3. D.E. Denning and P.J. Denning. Internet Besieged, ACM Press, New York 1998
4. C. Ellison and B. Schneier. Ten Risks of PKI: What You're Not Being Told About, *Computer Security Journal* 16(1), pp. 1–7, 2000.
5. ISO/IEC 9594-8, Information technology, Open Systems Interconnection. International Organization for Standardization, Geneva, Switzerland, 1995.
6. U. Maurer. Modeling public-key infrastructure. Proceedings, *Computer Security - ESORICS 96*, LNCS **1146**, Springer, 1996, pp. 325–350.
7. Trusted Computer System Evaluation Criteria (TCSEC). U.S. Department of Defense, 1985, 5200.28-STD (Orange Book).
8. M. Pease, R. Shostak, and L. Lamport. Reaching Agreement in the Presence of Faults, *Journal of ACM* 27(2), pp. 228–234, 1980.
9. M. K. Reiter and S. G. Stubblebine. Path independence for authentication in large scale systems. *Proceedings, 4th ACM Conference on Computer and Communications Security*, pp. 57–66, 1997.
10. R. L. Rivest and B. Lampson. SDSI–A Simple Distributed Security Infrastructure http://theory.lcs.mit.edu/~cis/sdsi.html
11. B. Schneier, Applied Cryptography, John Wiley & Sons, 1996.

Mike Burmester (burmester@cs.fsu.edu) and Yvo G. Desmedt (desmedt@cs.fsu.edu) are in the Department of Computer Science at Florida State University, Tallahassee, Florida.

This research was funded in part by DARPA F30602-97-1-0205 and by EPSRC GR/23301.