Offline Delegation*

Arne Helme† Tage Stabell-Kulø
Department of Computer Science
University of Tromsø, Norway
{arne,tage}@acm.org

Abstract

This article describes mechanisms for offline delegation of access rights to files maintained by a distributed "File Repository". The mechanisms are designed for a target environment where personal machines are used at times when critical services, such as authentication and authorization services, are not accessible. We demonstrate how valid delegation credentials can be transferred verbally without the use of shared secrets.

Our main result shows that delegation of access rights can be accomplished in a system that uses public-key encryption for secrecy and integrity, without forcing the user to rely on a trusted third party, and without requiring connection to the infrastructure.

The implementation runs on a contemporary Personal Digital Assistant (PDA); the performance is satisfactory.

1 Introduction

When personal machines are incorporated into distributed systems, privacy becomes important. In our view, the mere existence of personal machines implies a new direction in computer security research. Rather than solely protecting centralized resources from unauthorized access, protecting the interests of the individual becomes the focus. For example, users will want to decide for themselves when and to whom access to their resources should be granted. In contrast, consider a system with centralized control, such as Kerberos [Steiner et al., 1988]. In Kerberos, the trust relation between the system and its users is asymmetric and the organization maintaining the system dictates when and to whom access to resources should be granted. It is impossible for a user to generate credentials, either inside or outside Kerberos, that will be valid outside of its realms. Consequently, users cannot delegate access to their own resources at will. This lack of control over personal resources is an architectural concern, and not in any way related to the cryptographic technology that is applied in the system (symmetric or asymmetric cryptography).

Armed with a Personal Digital Assistant (PDA), users will challenge centralized models of authentication and access control by demanding to be in authority of their own resources. In essence, a PDA can provide the user with a Trusted Computing Base (TCB) [Department of Defense, 1985]. The TCB gives leverage in situations where the user accesses resources remotely or wants to delegate access rights to other users.

A system that relies on PDAs for a part of its security creates a new set of engineering challenges. This article examines a problem that is rather specific to mobile computing: how to cope with circumstances where there is low or no connectivity between a user's machine and the system components critical to some operations. Our focus is on how authority can be delegated from one user to another without the requirement of communication with any kind of server.

PDAs are often not connected to a computer network. Even so, delegation of authority should be possible. Delegation may have to take place indirectly and possibly over unconventional paths — such as part of a telephone conversation. Consider the following scenario: Alice and Bob are having a conversation on the phone, and Alice wants to grant Bob access to a file of hers. Since she has her TCB at hand, she should be able to generate sufficient credentials to enable Bob to access the file. This problem is denoted "offline delegation of access rights" (initially described in [Helme and Stabell-Kulø, 1996]). We describe how the problem comes about and manifests itself, what the implications are, and presents a so-

*Funded by the GDD project of the Research Council of Norway (project number 112577/431)

†Funded by the GDD-II project of the Research Council of Norway (project number 1119400/431)

lution to it. Some relevant details of an implementation are also presented.

The remainder of this article is structured as follows. Section 2 starts out by arguing why offline delegation is desirable, and presents the environment in which a solution has been implemented. Section 3 discusses the security constraints while Section 4 discusses the protocols that are used. Section 5, describes some relevant implementation details. Then, Section 6 discusses aspects of the TCB, the assumptions made about the PDA and the (secure) channels that are present in the system. Finally, the conclusions are drawn in Section 7.

2 Overview

The setting is one with a file repository (called FR) that manages replication and concurrency control of files [Stabell-Kulø and Fallmyr, 1998]. FR has been designed to support users with a variety of equipment, ranging from PDAs to workstations. FR is the research vehicle used to investigate the thesis that users should be involved at the places in a system where decisions (either implicit or explicit) are made. In particular, in a system with PDAs disconnection will be common and consistency problems occur frequently; FR has been designed to allow users to deal with them in any way they choose.

Along the same line of thought: Why should users be forced to contact FR simply to generate a delegation certificate? Stated differently: Is delegation only possible when there is connectivity? Systems that are constructed so that principals must be on line for delegation to take place, effectively exclude PDAs. How to delegate authority without connectivity is the theme central to offline delegation.

In FR, users are represented as principals by their public keys. Consequently, all communication channels can be authenticated for integrity and encrypted for privacy. An integrated part of FR´s design is that authority over files can be delegated freely. More specifically, a delegation certificate names a file and a user, together with an access right (read, write or both), and the time of creation (of the certificate) and when it will expire. The syntax and semantics of delegation certificates are discussed in detail below.

Making offline delegation possible requires that two problems are solved. First, one must be able to generate a valid certificate by means of a PDA without having access to FR. Second, it should be possible to convey the certificate verbally; without a computer network messages must be sent by means of the communication channel that is available; that is, by means of human speach. This constraint rules out every binary representation of certificates in so far that it is unlikely that anyone will be able or willing to read hundreds of digits over the phone. Similar arguments also rule out the use of digital signature schemes relying on long signatures (RSA [Rivest et al., 1978], for example, with 1024–2048 signature bits).

The following sections describe both the cryptographic techniques and the tools required in order to construct a practical solution to the problems described above.

3 Security considerations

Design of an offline delegation mechanism must carefully trade off convenience against availability without compromising the overall system security. This section explores the design space, and in particular the requirements for the delegation certificates. A solution must fulfill the following design criteria:

1. A delegation (i.e., a certificate) should not enable any principal to impersonate the delegator or delegatee.
2. The credentials must form valid and meaningful access rights. In particular, all objects (principals, machines and files) must be unambiguously named.
3. The authority granted by a signed statement should not be transferable, and a certificate containing the signature in question should be valid for usage only once.

To ease the task of verbally transferring access rights—while retaining security—the following strategy is used. Generally, of all the information in a typical certificate only the digital signature constitutes binary data. Because delegation certificates contain a wealth of information that can easily be read out, such as dates, file names, Internet addresses, domain names, etc, the amount of binary information that needs to be exchanged is limited. The key issue is then becomes to simplify and ease the exchange of the signature bits.

In order to delegate access rights based on digital signatures, the number of bits that needs to be exchanged is critical to the security of the system. Shared-key systems have shorter keys (64–128 bits) than public-key systems, but can not be used in this system, because they undermine the entire security regime in FR [Helme and Stabell-Kulø, 1997]; we elaborate on this in Section 6. Decreasing the length of the signature then implies using a crypto system with a denser key space. We have opted for the latter, and chosen a crypto system based on elliptic curves in finite fields. For a general description of elliptic-curve cryptography, see [IEEE, 1997, Menezes et al., 1996].

In addition to the requirements described above, to ensure that "once only" semantics can be enforced, each certificate must be unique. We know from experience that it takes (much) more than one second to generate a certificate, so rather than adding a serial number to them, we add the time of creation. With a resolution of the clock of (less than) one second, all certificates (from one principal) will be unique. To some extent, the time of creation can be viewed as a non-contiguous serial number. Including in each certificate its time of expiration, on the other hand, ensures an implicit revocation of old (unused) certificates.

4 Protocol description

This section describes the protocols used to delegate access rights from one user to another. In order to obtain a copy of a file a user must interact with FR by means of a protocol. The protocol is named File Repository Transport Protocol (FRTP) [Stabell-Kulø, 1995]. It resembles the Simple Mail Transport Protocol (SMTP) and Network News Transport Protocol (NNTP) protocols in that all commands and responses are encoded in ASCII with short numerical status-codes being returned for each command.

FR requires that users are authenticated and it provides users with the means to establish the authenticity of the server. At connection setup time, a secure channel is established between the user and FR. Certificates can be presented to FR without having to be signed by the delegatee when sent on an authenticated channel. The certificate can also be signed and subsequently sent unencrypted. In the latter case, the file will be returned on the same connection (but not on a secure channel). The latter approach is naturally equivalent with the former except that secrecy is not achieved. FR supports both kinds of interactions mentioned above.

4.1 Certificate creation

Before a protocol run can take place the delegator (file owner) creates a delegation certificate. This delegation certificate contains a digital signature that vouches for his delegation of access rights for a particular file to the other user. The certificate grants the delegatee access to the file. The certificate components that can easily be conveyed verbally are the following:

- the names of the file, the delegator and the delegatee,
- time of creation and expiration
- specification of delegated access rights. In the current design, access rights can be either *read* or *write*. A certificate will, thus, enable the delegatee to access the file according to the delegated access rights specified in the delegation certificate.

The verbal delegation ends with the exchange of the bits resembling the signature. Together with the other information exchanged, the signature enables the delegatee to (re)construct a machine readable representation of the certificate. These certificates, obviously, must be readable not only by computers but also by humans. To facilitate this, certificates are encoded in a syntax similar to SDSI [Rivest and Lampson, 1996].

4.2 Access-request protocol

The *access-request* protocol performs a forward authentication of access rights from the delegator to the server via another user (the delegatee). In short, to access a file the delegatee signs the delegation certificate with his own private key and transmits the result to the file server (FR). The following messages are exchanged:

$$\begin{array}{lll} \text{Message 1} & A \rightarrow B: \{A,B,F,AC,T,S\}_{K_A^{-1}} & (=X) \\ \text{Message 2} & B \rightarrow S: \{B,S,X,WD\}_{K_B^{-1}} & (=Y) \\ \text{Message 3} & S \rightarrow B: \{S,B,H(X),H(Y),RD,WD\}_{K_S^{-1}} & \end{array}$$

In the protocol description, A and B represent the users Alice and Bob, S is the FR (server), F is the name of the file in question, AC are the delegated access rights, and T are two time stamps, one making the certificate unique the other ensuring that the certificate expire. K_A^{-1} is the private key belonging to A. Message 1 (or X, in short) is the delegation certificate. $H(X)$ is the message

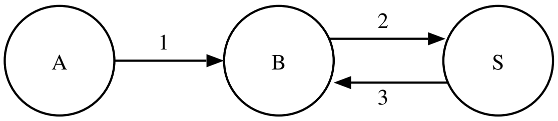


Figure 1: Messages in the protocol

digest of X, and is essentially B's receipt from S. The field *RD* is data that has been read while *WD* is data to be written. If the access-right is *read*, *WD* must be *nil*. The protocol does not distinguish between conventional (networked) or verbal transfer of the first message — in both cases the same information is presented to the server in Message 2.

4.3 Analysis

The protocol consists of three messages, as illustrated in Figure 1. The messages are explicit, and contain sufficient information that their meaning can be established without context. That is, the meaning of messages does not depend on a (set of) previous message(s).

A BAN logical analysis of the protocol begins with the observation that Message 1 (denoted X in the protocol description) is received by B over a "real time" channel, implying that $B \mid\equiv \sharp(X)$ (see [Abadi et al., 1991]). The assumption $B \mid\equiv \overset{K_A}{\mapsto} A$ makes it possible for B to believe that $A \mid\sim X$; a similar argument also holds for S. If B believes that S is honest [Syverson, 1992], we obtain $B \mid\equiv S \mid\equiv \mathit{data}$ because $B \mid\equiv \sharp(X)$ and $B \mid\equiv S \mid\sim (X, \mathit{data})$ (derived from Message 3). In other words, B believes that he has received the correct file from S.

At a higher level [Lampson et al., 1992], Message 1 contains two statements (we only deal with reading, writing is similar):

$$A \textbf{ says } B|A \Rightarrow B \textbf{ for } A \tag{1}$$

and

$$A \textbf{ says } B \textbf{ for } A \textit{ may read file } F. \tag{2}$$

Statement (1) says says that the combined principal $B|A$ may speak for B **for** A, while (2) says that B **for** A may read from file F. When B includes Message 1 in Message 2, it is interpreted as

$$B|A \textbf{ says } \textit{read } F. \tag{3}$$

Statement (3) enables S to deduce that B (by quoting A) indeed speaks for B **for** A. Consequently, if A owns the file F, then (3) should be honored.

5 Implementation details

This section describes the implementation of the offline delegation mechanism in greater detail. The implementation of the File Repository currently runs on Unix and Windows NT. As PDA we have used the Palm-III from 3Com with PalmOS as the operating system. The offline delegation mechanism is hosted both on the PDA and on Unix workstations.

In addition to FR itself, the implementation consists of a library with cryptographic functions, a library to parse and generate SDSI objects and a graphical user interface to create/send and receive/verify certificates on the PDA. First we will describe the user interface of the offline-delegation application running on the PDA. It is by means of this application that the users can *create*, *send* and *receive* certificates. Then we will focus on the two libraries that are part of our implementation. The first library contains cryptographic functions to create short digital signatures. The second library converts certificates between internal and external representations.

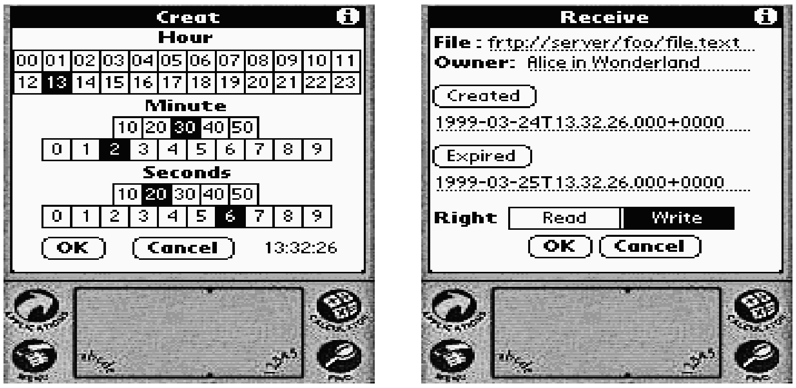
5.1 User interface

The bandwidth of the channel defined by human conversation is relatively low. The core of an offline delegation system is the dual ability to generate a valid certificate on a PDA and making it "readable" in a form which is simple to both send and receive on such low-bandwidth communication channels. To facilitate offline delegation, we have designed and implemented a supporting application on the PDA. The appearance is such that it enables two users to exchange sufficient information to send a certificate on one end, and receive it on the other.

In general, the process of building a certificate consists of entering the required information into the application, and then let the application generate it. The following information is required:

Filename: It is the users' responsibility to ensure that file names are unique within the scope of a server; the name of the file is prefixed by the name of a server. In the implementation, applications synchronize with the file server and has a cache of filenames available. If access is delegated to a file which name is not available, the name must be entered manually.

Delegatee: In all systems where users are represented by their public keys, names must be unique within



(a) Setting the time

(b) Information

Figure 2: Elements from the user interface components to create delegation certificates

the system. More precisely, the name appearing in the certificates must be unique. As with file names, the software keeps a cache of frequently used user names.

Created, Expire: In order to ease the process of entering dates, calendars are used (with the current date as default). Figure 2(a) shows how time is entered into the application; the user taps the boxes with the stylus.

Rights: A certificate can delegate authority [Lampson et al., 1992]. In our setting, such authority can be read or write (or both).

Figure 2(b) shows how general information is entered into the application.

The assumed operational procedure is that Alice enters information into the data fields on her PDA while she talks with Bob, and Bob enters the same information into his PDA. The software has been designed to be used in this way, that is, the order in which information is entered when a certificate is created is matched on both sides. By going through the fields together, they build up the certificate. When Alice is finished entering information, she will sign the data with her private key.

As will be explained below, the signature is 256 bits long. On the sending side, the bits are presented to Alice as 16 4-digit hexadecimal numbers; see Figure 3(a). Notice that the checksum in the right hand column is a simple error detection scheme. She can now read the signature bits out to Bob, one group at a time.

As Bob listens to Alice, he needs means to enter the signature bits as fast as Alice reads. To facilitate this, a dedicated form is presented on the PDA. By tapping on the screen of the PDA he is able to enter data (*receive* as it were). The design is such that users can receive bits fast enough for the system to be usable. See Figure 3(b) for a signature that has been partly received.

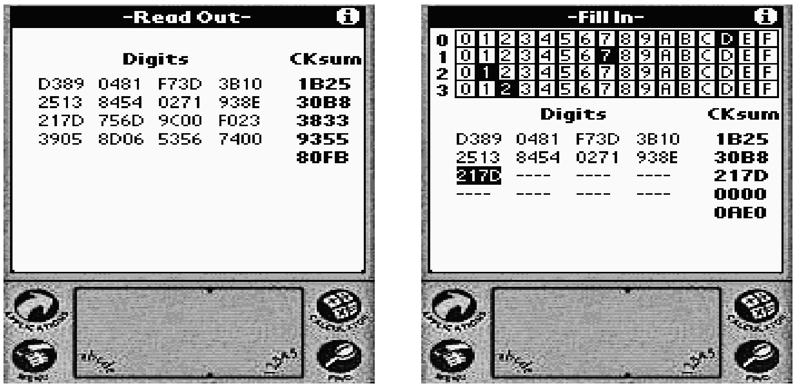
The checksum is calculated as data is entered. Although only a proper verification of the signature can determine whether it was properly transferred, the checksum is used to give Alice and Bob some confidence in its integrity. In this implementation the checksum is an exclusive-or function computed over the four 16-bit numbers. As a result, two bits in any number of row of bits will go undetected. If experience shows that a stronger integrity check is required, it can easily be incorporated into the application.

In the Unix client, the method used in the S/KEY one-time password system [Haller, 1994] has been adopted and the signature bits are conveyed in the form of English words.

5.2 Crypto library

The cryptographic library provides functions to create and verify digital signatures. Currently the library is available on both Unix and PalmOS platforms. The library provides the Nyberg-Rueppel version of elliptic curve signatures [Nyberg and Rueppel, 1993] and the SHA1 message digest function.

The implementation of elliptic curve cryptography is



(a) Sending

(b) Receiving

Figure 3: Elements from the user interface to send and receive delegation certificates

based on the algorithms for fast operations in finite fields. [Win et al., 1996] that has been tuned to fit the limited processor and memory resources on a small PDA. The current implementation uses a finite field of order $GF(136)$ with fast operations on elements in a subfield of order $GF(8)$. The order of the fixed point on the curve is a 241 bit prime number. This yields Nyberg-Rueppel scheme digital signatures with a total length of 256 bits for both components of the signatures.

The security of the elliptic-curve signature scheme relies on the difficulty of finding discrete logarithms in finite fields of special form. With the current field size and chosen curve parameters the security of the system is estimated to be at least of similar strength as 1024 bits RSA signatures. Digital signature schemes based on elliptic curves show not only promising results with respect to performance (signing speed, in particular), but also when it comes to strength per bit and memory utilization [IEEE, 1997].

5.3 SDSI library

All keys, certificates and protocol messages used in the implementation of the offline delegation mechanism are specified in a format similar to SDSI [Rivest and Lampson, 1996]. Only the syntax specification of SDSI has been adopted, however, and not its associated public-key infrastructure. We have chosen an external representation that is "human readable". That is, the majority of data in the certificate is represented in ASCII. In our experience it is valuable to be able to "look at" certificates to compare the fields one by one.

SDSIlib is based on the SEXP library designed and implemented by Ronald Rivest. The library has been extended to support a syntax similar to SDSI and contains sufficient functionality to build parsers and generators for new protocols with SDSI encoding of the protocol messages. The SDSIlib API [Helme, 1998] specifies the external representation of data in SDSI objects, and a set of library functions to manipulate such objects. The library contains basic functions to parse and generate basic SDSI objects. In the SDSIlib port for the Palm-III, most of the library's functionality has been retained. The library can be configured to read and write SDSI objects from and to TCP/IP streams, Unix-like file I/O, or PalmOS databases.

As an example, consider the specification of a `Signature` object containing a signed delegation statement in Figure 4. The `Signature` object contains an offline delegation certificate. More specifically, the signed object consists of the information that has been signed (`Object`), and information (`Algorithm`) about the signature algorithm that has been used to create the signature in question. The `Algorithm` field also holds information about the hash algorithm and signature algorithm that have been used to create the digital signature. The format of the `Algorithm` field is *signature-algorithm*`-with-`*hash-algorithm*. The name of the algorithm-dependent hash field is *hash-algorithm*`-Hash`. The name of the algorithm-specific signature field is *signature-algorithm*`-Signature`. The fields `Object-ref` and `Object-perms` identify the file subject to delegation and the delegated access rights.

```
( Signature:
  ( Object:
    ( Delegate-From: "Alice in Wonderland" )
    ( Delegate-To: "Bobs Country Bunker" )
    ( Object-ref:  frtp://server/foo/file.text )
    ( Object-perms: read )
    ( Created: "1999-03-24T13.32.26.000+0000" ) )
    ( Expire: "1999-03-24T13.32.26.000+0000" ) )
  ( Algorithm: ECNR-with-SHA1 )
  ( SHA1-Hash: |c+1xi/6oE4k5Hr8JPR1T4Q==| )
  ( ECNR-Signature:
    ( Signing-Key: |B3ZbHXKyBwQ=| )
    ( Galois-Field: #88# )
    ( R: |Gk/PrhBpnNdXinxC11krrQ==| )
    ( S: |DPI7aahT4A9c5MeG7EF/VQ==| ) ) )
```

Figure 4: Specification of a delegation certificate in a syntax similar to SDSI

6 Discussion

This section discusses some aspects of offline delegation in distributed systems, and in our system in particular. First we argue why offline delegation does not weaken the security of the system, then we discuss issues related to the trusted computing base, revocation of certificates and performance of the prototype implementation.

6.1 Security concerns

We believe that the provision of offline delegation does not weaken the security of FR in any way. This hinges on the fact that the file owner is solely responsible for the security policy he implements. The increased flexibility offered by offline delegation comes with the price of responsible and competent users. However, for each delegation there is only a single file involved, and other files are not involved in any way, so a user can not compromise the security of any other user. The security problem intrinsic to stolen PDAs is treated in the section on the TCB.

A file can be handed out erroneously if it is given the same name as an old file, and access to the old file has been delegated. A new file with the same name as an old file can be regarded as if the contents was altered on the old file. FR can thus not distinguish between rightful access to new data in the old file and incorrect access to a new file with the same name as an old file. By ensuring that all certificates expire properly (within a reasonable time frame) names of files can be reused after that time. We do not consider this a security problem.

The security regime of the system is built on public key cryptography. It is a goal that FR should be able to produce a certificate for each and every transaction that takes place. In such a system, a scheme built on shared keys is very hard to conceive. Any secret shared with FR can be used by FR to convince itself about the origin of a message, but such "proof" has no value to others.

6.2 Trusted Computing Base

The single most intriguing issue with the concept of PDAs is the possibility that every user can have a Trusted Computing Base (TCB) under his control which does not include resources controlled by others. That is, if the system is designed in such way that the user's PDA constitutes his TCB, then keys and credentials can safely be stored in it. More specifically, a trusted PDA can act on its owners' behalf when the owner delegates access rights. In particular, when it comes to the case of binding a public key to a human, the TCB consists only of the user's PDA. This, of course, stems from the use of public-key technology rather than offline delegation per se.

Recall the scenario with Alice delegating authority to Bob while speaking on the phone. In this scenario, FR is not part of the TCB because it is "only" used to store files. The result is that FR is unable to impersonate the user, neither when interacting with other instances of FR, nor when interacting with other users.

6.3 Revocation

Disconnected operation is common in our system and revocation of access rights is consequently a concern. Effective revocation of access rights in distributed systems is generally considered a hard problem to solve [Lampson et al., 1992], and lack of connectivity makes the problem even more difficult. This places limits on when revocation can be performed. In order to revoke a certificate there are essentially two approaches: either to

1. limit the time frame in which certificate is valid,
2. let the certificate be valid only once.

Both approaches have their merits and disadvantages [Helme, 1997]. FR provides offline delegation and it supports both mechanisms. To ensure that certificates are used only once, once they are used they are stored by the server until they expire. This policy facilitates that certificates have "once-only" semantics (see [Helme, 1997]). Users can not override this policy, but if there is a need to grant another user access on a more permanent basis, Access Control Lists (ACL) can be implemented. A discussion of ACLs in FR is beyond the scope of this article.

Timestamps are used as an additional source of information for revocation purposes. Since individual users specify access policies, the correctness of the time stamp encoded into each delegation certificate depends entirely on this user's ability to determine what the current time is. However, timestamps are only used to recognize and refuse old certificates. The use of time to discard once-only delegation certificates is not entirely without risks (for a discussion, see, for example, [Gong, 1992]).

6.4 Performance

On the platform for which we have made our implementation, entering information is time consuming. The graphical user interface is pen-based, and writing is rather slow. Some fairly long strings (such as file names with an associated path) have to be entered to uniquely identify a file, or selected from a list of frequently used strings.

On the PDA, to create a digital signature on a certificate takes about 5 seconds while verifying a signature takes about 10–15 seconds. While the process of signing certificates is mandatory, verifying the signature on the PDA is not. The receiver may choose to pass the certificate on to FR without verifying that it is correct. FR runs on Unix, and verification of a signature in this environment takes fractions of a second and does not pose a performance problem in the intended target environment.

6.5 Future work

The File Repository is the research vehicle for several other projects on data consistency and distributed applications, and the offline delegation mechanism will be used by these other projects whenever feasible.

One related project uses the File Repository to hold information about appointments. The offline delegation mechanism enables users to delegate access rights to other users in order to let them either read diary information or to (selectively) update it. The Global Distributed Diary (GDD) project explores these issues further.

7 Summary and conclusions

We have described how offline delegation is a natural extension to the services already offered by FR. The argument is focused on the concept of "user in the decision loop", and offline delegation is a consequence of this design philosophy. Offline delegation is used to delegate access rights from one user to another, in a setting where communication with FR is impossible at the time (or undesirable for some reason). To ease the exchange of delegation certificates, a method has been developed that enables two users to convey a certificate verbally. Cryptographic techniques based on elliptical curve encryption are used to facilitate short signatures so that as little data as possible has to be conveyed while maintaining a high level of security. The increased complexity in the implementation is outweighed by the advantage of short signatures.

The PDA of choice is the Palm-III, manufactured by 3Com. It was shown that performance is satisfactory even on this class of hardware. Furthermore, by utilizing the graphical user interface on the PDA, it is possible to transfer certificates (using speech) as part of a conversation between humans. Security is preserved and the performance is satisfactory.

Acknowledgements

The authors want to thank Feico Dillema, Jaap-Henk Hoepman, Åge Kvalnes, Sape Mullender, Per Harald Myrvang, and the anonymous referees for valuable comments. The first author also wants to thank George Barwood and Paulo Barreto for providing source code and useful information on how to implement elliptic curve cryptography.

References

[Abadi et al., 1991] Abadi, M., Burrows, M., Kaufman, C., and Lampson, B. (1991). Authentication and Delegation With Smart-cards. *Theoretical Aspects of Computer Software*, pages 326–345. Springer-Verlag.

[Department of Defense, 1985] Department of Defense (1985). DoD 5200.28-STD: Department of Defense (DoD) Trusted Computer System Evaluation Criteria (TCSEC).

[Gong, 1992] Gong, L. (1992). A Security Risk of Depending on Synchronized Clocks. *ACM Operating Systems Review*, 26(1).

[Haller, 1994] Haller, N. M. (1994). The S/KEY One-time Password System. In *Proceedings of the Internet Society Symposium on Network and Distributed System Security*, San Diego, CA.

[Helme, 1997] Helme, A. (1997). *A System for Secure User-controlled Eletronic Transactions*. PhD thesis, Informatica, University of Twente, Enschede, the Netherlands.

[Helme, 1998] Helme, A. (1998). SDSIlib API: Tutorial and Programmer's Reference Manual. In preparation.

[Helme and Stabell-Kulø, 1996] Helme, A. and Stabell-Kulø, T. (1996). Off-Line delegation in a File Repository. In *1996 DIMACS WorkShop on Trust Management in Networks*, Rutgers University. Work presented at workshop.

[Helme and Stabell-Kulø, 1997] Helme, A. and Stabell-Kulø, T. (1997). Security Functions for a File Repository. *ACM Operating Systems Review*, 31(2):3–8.

[IEEE, 1997] IEEE (1997). Standards Specifications for Public Key Cryptography (P1363, Draft Version 7). IEEE Standards Draft.

[Lampson et al., 1992] Lampson, B., Abadi, M., Burrows, M., and Wobber, E. (1992). Authentication in Distributed Systems: Theory and Practice. *ACM Transactions on Computer Systems*, 10(4):265–310.

[Menezes et al., 1996] Menezes, A. J., van Oorschot, P. C., and Vanstone, S. A. (1996). *Handbook of Applied Cryptography*. CRC Series on Discrete Mathematics and Its Applications. ACM Press.

[Nyberg and Rueppel, 1993] Nyberg, K. and Rueppel, R. (1993). A New Signature Scheme Based on the DSA Giving Message Recovery. In *First ACM Conference on Computer and Communications Security*, pages 58–61. ACM Press.

[Rivest et al., 1978] Rivest, R., Shamir, A., and Adleman, L. (1978). A Method for Obtaining Digital Signatures and Public-key Cryptosystems. *Communications of the ACM*, 21(2):120–126.

[Rivest and Lampson, 1996] Rivest, R. L. and Lampson, B. (1996). SDSI—A Simple Distributed Security Infrastructure. `http://theory.lcs.mit.edu/~rivest/sdsi10.ps`. (Version 1.0).

[Stabell-Kulø, 1995] Stabell-Kulø, T. (1995). File Repository Transfer Protocol (FRTP). Technical Report CS-TR 95-21, Department of Computer Science, University of Tromsø, Norway.

[Stabell-Kulø and Fallmyr, 1998] Stabell-Kulø, T. and Fallmyr, T. (1998). User controlled sharing in a variable connected distributed system. In *Proceedings of the seventh IEEE International Workshop on Enabeling Technologies: Infrastructure for Collaborative Enterprises (WETICE'98)*, pages 250–255, Stanford, California, USA. IEEE Computer Society.

[Steiner et al., 1988] Steiner, J. G., Neumann, B. G., and Schiller, J. I. (1988). Kerberos: An Authentication System for Open Network Systems. In *Proceedings of the Winter 1988 Usenix Conference*, pages 191–201.

[Syverson, 1992] Syverson, P. (1992). Knowledge, Belief and Semantics in the Analysis of Cryptographic Protocols. *Journal of Computer Security*, 1(3):317–334. IOS Press.

[Win et al., 1996] Win, E. D., Bosselaers, A., Vandenberghe, S., Gersem, P. D., and Vandewalle, J. (1996). A Fast Software Implementation for Arithmetic Operations in $GF(2^n)$. In Kim, K. and Matsumoto, T., editors, *Advances in Cryptology – ASIACRYPT'91*, volume 1163 of *Lecture Notes in Computer Science*, pages 65–76. Springer-Verlag.