To appear in *Journal of Computer Security, 1996.*

A Calculus for Security Bootstrapping in Distributed Systems[1]

Ueli M. Maurer

Department of Computer Science
ETH Zürich
CH-8092 Zürich, Switzerland

Pierre E. Schmid

Omnisec AG
Trockenloostrasse 91
CH-8105 Regensdorf, Switzerland

Abstract

A calculus of channel security properties is presented which allows the analysis and comparison of protocols for establishing secure channels in a distributed open system at a high level of abstraction. A channel is characterized by its direction, its time of availability and its security properties. Cryptographic primitives as well as trust relations are interpreted as transformations for channel security properties, and a cryptographic protocol can be viewed as a sequence of such transformations. A protocol thus allows to transform a set of secure channels established during an initial setup phase, together with a set of insecure channels available during operation of the system, into the set of secure channels specified by the security requirements. The necessary and sufficient requirements for establishing a secure channel between two entities A and B are characterized in terms of secure channels to be made available during the initial setup phase and in terms of the minimal trust A and B must have into other entities or into trusted third parties.

Keywords. Security, Distributed systems, Cryptography, Key management, Trust, Security transformations.

1. Introduction

The fast development of worldwide distributed information systems and communication networks (e.g. the World Wide Web on the Internet) and their use for electronic commerce leads to security problems whose solution becomes increasingly important as the use of these systems grows [7]. While for some small-scale systems security can perhaps be viewed simply as a desirable feature, integral security is a mandatory requirement for any large-scale information system to be merely operational, and for security-critical systems to be acceptable to its users.

Relevant security goals include the protection of sensitive data against eavesdropping, the authentication of messages (including sender and receiver identity, content, time of generation and possibly other parameters of a message), the non-repudiation of sending or receipt

[1] A preliminary version of this paper appeared in Proc. of the 1994 European Symposium on Research in Computer Security (ESORICS' 94), D. Gollmann (Ed.), Lecture Notes in Computer Science, Berlin: Springer-Verlag, vol. 875, pp. 175-192.

of a message, the security and possibly anonymity of payments over a network, etc. Substantial progress has been made in the security-relevant technologies, which include tamper-proof device technology (e.g. smartcards) biometric technology and, as a key technology, cryptography. Powerful cryptographic mechanisms like encryption algorithms, message authentication codes, public-key cryptosystems and digital signature schemes are available. In contrast trust, which is another fundamental ingredient for distributed systems security, seems to be less well understood [26], [1]. One of the goals of this paper is to discuss in a simple framework the interworking of cryptography and trust relations in distributed systems security.

The importance of security in large distributed systems and communication networks has long been identified and addressed by academic and industrial research (e.g., see [2], [4], [8], [10], [11], [25], [26]), and several solutions have been proposed [21], [16], [13], [12], [24], [27], some of which have been implemented in commercial products. These systems are based on various key management paradigms, and it remains to be evaluated for which application scenarios approaches based on on-line servers [21], [13], centralized and hierarchical certification authorities [24], [28], decentralized public key certification [27], or a combination of these models, are best suited. These approaches differ with respect to the required communication, the user responsibility and the necessary trust relations. For example, while centralized certification management may be suitable for large organizations, a distributed trust management approach may be needed in a world-wide network like the Internet.

The process of establishing security in a distributed system can be viewed as a two-phase process:

- **Initialization phase:** During this phase, communication channels with security properties (e.g. a trusted courier, a user's personal registration at the premises of a trusted third party, mutual authentication by speaker identification on an otherwise insecure voice channel, etc.) are available for setting up security parameters like a shared secret encryption key or an authenticated public key.

- **Communication phase:** During this phase, entities can typically communicate only over insecure channels. These channels can be protected by application of cryptographic techniques, which can be interpreted as transferring the security properties of the channels available during the initialization phase to the insecure channels available during the communication phase, thus making the latter secure.

The purpose of this paper is to provide a simple and straight-forward formalism for illustrating and comparing the various approaches to security in distributed systems. The emphasis of our approach is on the simplicity and expressive power of the model. Unlike many previous formal treatments of security and authentication in distributed systems [4], [9], [23], [26], it is not intended to be applied for the design or the security verification of protocols. It rather allows to illustrate in a simple manner several interesting aspects of distributed system security: the minimal requirements and trust relations necessary for achieving a secure channel between two entities in a distributed system, the timing constraints on the involved communication channels, the duality between authenticity and confidentiality, and the distinguishing features between secret-key and public-key cryptography. It also provides a black-box view of cryptographic primitives, showing that a digital signature scheme can be interpreted as the dual of

a symmetric cryptosystem and a public-key cryptosystem can be interpreted as the dual of a message authentication code.

Cryptographic primitives and trust relations are both interpreted as transformations for channel security properties, and a cryptographic protocol can be viewed as a sequence of such transformations of both types. A protocol thus makes it possible to transform a set of secure channels established during an initial setup phase, together with a set of insecure channels available during operation of the system, into the set of secure channels specified by the security requirements.

The minimal requirements for achieving a secure channel between two entities in a distributed system are characterized in terms of secure channels to be made available in an initial setup phase and in terms of the minimal required trust relations between users and/or between users and trusted authorities. Several types of protocols are reviewed within the presented framework, but it is not a goal of this paper to develop new protocols. We do not distinguish between different types and degrees of trust [26], but our model could be extended in this direction.

Many of the issues addressed in this paper have been discussed before [3], [18], [20]. These models differ with respect to their expressive power, the degrees of formality, and the types of channels that are modeled. Rueppel's approach [20] illustrates the importance of confidentiality and authenticity as basic security goals but does not address trust as a fundamental ingredient of security bootstrapping in distributed system. In contrast, Rangan [18] addresses trust relations between users and authentication servers in a model that is based on a logic of belief. The most advanced model, based on the formal specification language Z, is due to Boyd [3]. Our independently developed framework has several similarities with Boyd's, which was first brought to the authors' attention after a preliminary version of this paper had been presented at ESORICS '94.

Despite its relations and similarities to these previous approaches, this paper differs in several aspects. Cryptographic primitives are characterized completely in terms of the transformations they permit, and this leads to insights into the duality between certain primitives. The clear distinction between initialization and communication phases, the explicit consideration of the insecure communication channels needed in the communication phase, and the explicit representation of trust-based transformations allow the comparison of various key management approaches. While in [3] the intuitive aspects of secret keys and secure channels are made precise in formal definitions, we emphasize their intuitive nature by stating them as axioms. We hope that the intuitive nature of our model nicely complements previous approaches.

The paper is organized as follows. Section 2 describes a classification of channel security properties and Section 3 provides a complete list of cryptographic transformations of such channel security properties. Sections 6 and 4 discuss the necessary and sufficient condition for establishing a secure channel between two users in an open network, with and without exploiting trust relations, respectively, and security transformations based on trust relations are introduced in Section 5. In Section 7, several approaches to bootstrapping security in an open network are compared, and Section 8 concludes the paper with a brief comparison between secret-key and public-key cryptography in the context of distributed systems security.

2. Classification of channel security properties

At a high level of abstraction, a communication channel can be viewed as a means for transporting a message from a source entity (at the channel input) to a destination entity (at the channel output). In this paper we will remain at this high level of abstraction without considering any details of communication protocols. The term entity can thus either refer to a user, to an application, to the network or physical layer (according to the OSI model) of a systems communications interface, or to any other logical endpoint of a communication.

In a communication system it is possible that, either intentionally or due to information leakage, a message is received by several entities. Similarly, it is possible that, either by design of the system or due to a security problem, one of several entities could potentially be the sender of a message received by the receiver(s).

The duality of source and destination of a message or, equivalently, the duality of input and output of a channel, is reflected in the duality of the two fundamental security goals for messages (or channels). They refer to the exclusiveness of access to the channel input or the channel output:

- **Confidentiality:** A channel provides confidentiality if its *output* is exclusively accessible to a specified receiver and this fact is known to (or believed by) a sender on the channel.
- **Authenticity:** Similarly, a channel provides authenticity if its *input* is exclusively accessible to a specified sender and this fact is known to (or believed by) a receiver.

It is important to note that confidentiality and authenticity refer to a sender's and a receiver's state of belief, respectively. It is usually impossible to establish the truth of such a belief, and in practical applications it is the goal of the communicating parties to collect sufficient evidence for justifying such a belief. Such evidence for channel security attributes can either be based on physical assumptions (e.g. authenticity guaranteed by speaker identification, confidentiality guaranteed by the tamper-proofness of a module), on cryptography (e.g. encrypted communication over an otherwise insecure channel, digital signatures), or a combination of these.

As mentioned above, confidentiality is a security property that refers to a sender's viewpoint: the sender wants to assure that the channel's output is accessible only to the intended receiver. However, there are more subtle aspects to confidentiality which will not be discussed in detail here. For instance, the sender may implicitly have to trust the receiver to keep the message confidential. Here we will assume that a sender who wishes to establish a confidential channel has decided beforehand that the receiver can be trusted to see the message. Moreover, addressing the question of who is actually interested in the confidentiality of a message, it could actually be the receiver who wants to keep a message (e.g. concerning him or her) to be received confidentially. However, our viewpoint will be that this is only possible when he can convince the sender to assure the confidentiality of the channel, thus again turning confidentiality into the sender's security goal.

Confidentiality and authenticity are independent and dual security properties. One can be available without the other, as will be discussed below. Based on these two fundamental security properties, channels can be classified into four different types according to whether they provide none, one or both of the properties.

Channels are denoted by the symbol $\longrightarrow$ and allow to transmit, at a given time, a message of unspecified length from an input to an output. The symbol $\bullet$ attached to one side of the channel symbol $\longrightarrow$ indicates that the user at the corresponding end of the channel has exclusive access to the channel from the opposite entity's point of view. The symbols for the four types of channels from an entity A to an entity B are:

$A \longrightarrow B$	channel that provides no security
$A \longrightarrow\!\!\bullet\, B$	channel that provides confidentiality but not authenticity
$A \bullet\!\!\longrightarrow B$	channel that provides authenticity but not confidentiality
$A \bullet\!\!\longrightarrow\!\!\bullet\, B$	channel that provides both confidentiality and authenticity

We further use the following symbol for a bidirectional secure channel between A and B:

$$A \bullet\!\!\longleftrightarrow\!\!\bullet\, B$$

Note again that a security symbol $\bullet$ refers only to the opposite entity's state of belief. For instance, the $\bullet$ in $A \bullet\!\!\longrightarrow B$ refers to B's belief that information received on this channel originates from A.

An illustrative "real-world" example of a $\longrightarrow\!\!\bullet$ channel is a mailbox for which only a designated person possesses a key. Anyone who believes that the key is held exclusively by this person can put a letter into the mailbox and be assured of the message's confidentiality. However, the recipient has no direct means for verifying the sender's authenticity. An example of a $\longrightarrow\!\!\bullet$ channel which may be more realistic in a network scenario will be mentioned in Section 8. Examples of $\bullet\!\!\longrightarrow$ channels are a bulletin board that is physically protected by a glass cover and a lock (whose key is available only to a designated person), or an insecure telephone line combined with reliable speaker identification. Examples of a $\bullet\!\!\longrightarrow\!\!\bullet$ channel are a trusted courier, an optical fiber (under certain assumptions) or an insecure channel protected by encryption with a secret key shared by sender and receiver.

Extending our classification of channels, a parameter above the channel symbol $\longrightarrow$ will indicate the time of availability. The symbols $\overset{t}{\longrightarrow}$, $\bullet\!\overset{t}{\longrightarrow}$, $\overset{t}{\longrightarrow}\!\!\bullet$ and $\bullet\!\overset{t}{\longrightarrow}\!\!\bullet$ will denote channels that are available at time t. For example, the availability of a trusted courier from A to B at time t is denoted by $A \bullet\!\overset{t}{\longrightarrow}\!\!\bullet\, B$. Such a channel can be used to send a secret key which can thereafter be used to encrypt and authenticate messages exchanged between A and B; hence an insecure channel $A \overset{t'}{\longrightarrow} B$ from A to B available at some later time $t' > t$ can be converted into a secure channel from A to B available at time t', denoted $A \bullet\!\overset{t'}{\longrightarrow}\!\!\bullet\, B$. Similarly, a telephone call at time t between persons A and B, where B can reliably recognize A's voice, corresponds to a $A \bullet\!\overset{t}{\longrightarrow} B$ channel.

We will also need to consider channels that introduce a delay, i.e., for which the input message must be specified at some time t_1 while the message appears at the output only at some later time $t_2 > t_1$. For the various types of security properties, such channels will be denoted by $A \overset{t_2[t_1]}{\longrightarrow} B$, $A \bullet\!\overset{t_2[t_1]}{\longrightarrow} B$, $A \overset{t_2[t_1]}{\longrightarrow}\!\!\bullet\, B$ and $A \bullet\!\overset{t_2[t_1]}{\longrightarrow}\!\!\bullet\, B$, where the notation $t_2[t_1]$ implies that $t_2 > t_1$ and where the bracketed time can be omitted when $t_1 = t_2$. For example, in a standard public-key certification scenario where a trusted third party T digitally signs an entity

A's public key and returns the certificate to A, the channel on which an entity B receives the certificate for A from T is actually a concatenation of two channels, $T \xrightarrow{t_1} A$ and $A \xrightarrow{t_2} B$, i.e. one from T to A at some time t_1 on which the certificate for A is returned, and one at some (possibly much later) time t_2 from A to B on which A sends his or her certificate to B. The resulting channel is hence $T \xrightarrow{t_2[t_1]} B$ (see Section 5).

Before we characterize cryptographic primitives as transformations of channel security properties, we list a few trivial non-cryptographic transformations. Transformations are denoted by the preconditions for the transformation followed by the symbol $\Longrightarrow$, and followed by the resulting channel. A given channel can be used as input to an arbitrary number of transformations, meaning that the actual message transmitted on that channel consists of several components.

When a $A \xrightarrow{t_2[t_1]} B$ channel is available then so is a $A \xrightarrow{t_2[t_1']} B$ channel for all $t_1' \leq t_1$; this is also true for the other types of channels. Hence we have

$$\left.\begin{array}{c} A \xrightarrow{t_2[t_1]} B \\ t_1' \leq t_1 \end{array}\right\} \quad \Longrightarrow \quad A \xrightarrow{t_2[t_1']} B$$

Furthermore, the symbol $\bullet$ can always be dropped when it is not needed in a transformation. For instance, when only authenticity but not confidentiality provided by a secure channel from A to B is exploited, we make use of

$$A \bullet\!\!\xrightarrow{t}\!\!\bullet B \quad \Longrightarrow \quad A \bullet\!\!\xrightarrow{t} B \tag{1}$$

If channels are available from A to B and from B to C at some times t_2 and t_4, respectively (where possibly the messages must be fixed at earlier times t_1 and t_3, respectively), then B can relay a message from A to C, provided that $t_3 > t_2$. Formally we write:

$$\left.\begin{array}{c} A \xrightarrow{t_2[t_1]} B \\ B \xrightarrow{t_4[t_3]} C \\ t_3 > t_2 \end{array}\right\} \quad \Longrightarrow \quad A \xrightarrow{t_4[t_1]} C \tag{2}$$

Note that the input message for the resulting virtual channel $A \xrightarrow{t_4[t_1]} C$ from A to C must be fixed by A at time t_1 while it is received by C only at the later time t_4.

It is appropriate to discuss briefly *availability* as a third fundamental security requirement (in addition to confidentiality and authenticity). Availability of an information or communication system, (for example the prevention of system failures and of denial of service attacks [14]) is often even more crucial than for instance the confidentiality of data. However, the techniques for achieving availability are completely different. In most cases, a system's availability, even in the presence of malicious attacks, can in principle be made arbitrarily high by introducing a sufficient amount of redundancy. Our goal is to achieve security in an insecure but reasonably reliable open network. Therefore the reliability of channels and entities is not explicitly represented in our model. For example, for the $A \xrightarrow{t_4[t_1]} C$ channel in transformation (2) to be available, B must be reliable and available. However, this subtlety will not arise in

this paper because transformation (2) will only be used in situations where B (rather than A) wants the message to reach C, and hence is reliable in its own interest.

When the channels from A to B and from B to C in transformation (2) both provide either confidentiality or authenticity or both, then so would the channel from A to C, but only when A or C, or both A and C, respectively, believe they can trust B. Trust-based transformations will be discussed in Section 5.

3. Cryptographic security transformations

In this section we present a systematic discussion of the well-known cryptographic primitives (symmetric encryption, message authentication code, public-key cryptosystem and digital signature scheme) by interpreting them as transformations of channel security properties. In this paper, cryptographic systems are treated as black boxes without considering their mathematical realization.

3.1. Symmetric encryption and message authentication codes

A symmetric cryptosystem or cipher is often assumed to provide implicit message authentication; the fact that a message is encrypted with a certain key "proves" the sender's knowledge of the key. However, this observation is valid only under the assumption that plaintext is sufficiently redundant; this allows to distinguish meaningless messages (resulting from the decryption of a fraudulent ciphertext) from valid messages. Moreover, certain types of ciphers (e.g. additive stream ciphers) provide no implicit message authentication because individual bits in the ciphertext can be flipped selectively, resulting in the corresponding plaintext bits to be flipped. The latter problem can be solved by appending to a given message a cryptographic hash value of the message. In the sequel we therefore assume without loss of generality that a symmetric cipher provides both confidentiality and authenticity. A message authentication code (MAC), also based on a secret-key cryptosystem, is used to assure a message's authenticity only.

In our framework, a symmetric cipher can be interpreted as follows. The basic security transformation provided by a symmetric cipher is to transfer the security of a channel available at some time t_2 to an insecure channel available at some later time t_4. The times t_1 and t_3 are included for the sake of generality and will be used later, but the reader can for the remainder of Section 3 just as well assume that $t_2 = t_1$ and $t_4 = t_3$.

$$\left.\begin{array}{l} A \overset{t_2[t_1]}{\bullet\!\!\longrightarrow\!\!\bullet} B \\ A \overset{t_4[t_3]}{\longrightarrow} B \\ t_4 \geq t_2 \end{array}\right\} \quad \Longrightarrow \quad A \overset{t_4[t_3]}{\bullet\!\!\longrightarrow\!\!\bullet} B \tag{3}$$

When the insecure channel is directed from B to A rather than from A to B, then so is the resulting secure channel:

$$\left.\begin{array}{l} A \overset{t_2[t_1]}{\bullet\!\!\longrightarrow\!\!\bullet} B \\ A \overset{t_4[t_3]}{\longleftarrow} B \\ t_3 > t_2 \end{array}\right\} \quad \Longrightarrow \quad A \overset{t_4[t_3]}{\bullet\!\!\longleftarrow\!\!\bullet} B \tag{4}$$

It is interesting to notice that a symmetric cryptosystem can also be used to transfer confidentiality without authenticity:

$$\left.\begin{array}{c} A \overset{t_2[t_1]}{\longrightarrow\!\bullet} B \\ A \overset{t_4[t_3]}{\longrightarrow} B \\ t_4 \geq t_2 \end{array}\right\} \quad \Longrightarrow \quad A \overset{t_4[t_3]}{\longrightarrow\!\bullet} B \tag{5}$$

To achieve this transformation, A can use the confidential channel $A \overset{t_2[t_1]}{\longrightarrow\!\bullet} B$ for transferring a (not authenticated) cipher key. Messages encrypted with this key can only be decrypted by B; hence the sender is assured of a message's confidentiality. However, although B can check that the message was sent by the same person who previously sent the secret key, the $A \overset{t_2[t_1]}{\longrightarrow\!\bullet} B$ channel provides no authenticity and hence nor does the $A \overset{t_4[t_3]}{\longrightarrow\!\bullet} B$ channel. On the other hand, if the second channel provides authenticity, then so does the resulting channel:

$$\left.\begin{array}{c} A \overset{t_2[t_1]}{\longrightarrow\!\bullet} B \\ A \overset{t_4[t_3]}{\bullet\!\longrightarrow} B \\ t_4 \geq t_2 \end{array}\right\} \quad \Longrightarrow \quad A \overset{t_4[t_3]}{\bullet\!\longrightarrow\!\bullet} B \tag{6}$$

The above transformation can also be achieved when the authenticated channel is available prior to the confidential channel. For this purpose, a secret key K is generated by A who uses the $A \overset{t_2[t_1]}{\bullet\!\longrightarrow} B$ channel to send $f(K)$, where f is a one-way function. At this point, B does not know K. Later, K is sent (confidentially) over the $A \overset{t_4[t_3]}{\longrightarrow\!\bullet} B$ channel together with the actual message authenticated (using a MAC) with the same key K. B can authenticate this key as well the message by computing $f(K)$ and comparing it with the value received before.

$$\left.\begin{array}{c} A \overset{t_2[t_1]}{\bullet\!\longrightarrow} B \\ A \overset{t_4[t_3]}{\longrightarrow\!\bullet} B \\ t_4 \geq t_2 \end{array}\right\} \quad \Longrightarrow \quad A \overset{t_4[t_3]}{\bullet\!\longrightarrow\!\bullet} B \tag{7}$$

While a symmetric cryptosystem can transfer confidentiality without authenticity (transformation (5)), it is important to note that it cannot transfer authenticity without confidentiality. The latter transformation is achieved by digital signatures and is one of the major advantages of public-key cryptography.

On the other hand, a symmetric cryptosystem or a MAC-scheme can also be used to convert a confidential channel into an authenticated channel. If A sends a secret key to B over the confidential channel, then A can later recognize messages encrypted with this key as authentic from B. However, since B cannot verify that A is indeed the sender of the secret key, the confidentiality of encrypted messages is not guaranteed.

$$\left.\begin{array}{c} A \overset{t_2[t_1]}{\longrightarrow\!\bullet} B \\ A \overset{t_4[t_3]}{\longleftarrow} B \\ t_3 > t_2 \end{array}\right\} \quad \Longrightarrow \quad A \overset{t_4[t_3]}{\longleftarrow\!\bullet} B \tag{8}$$

Note that when B receives several (not authenticated) secret keys from various entities he can authenticate a message for each key separately without need for knowing which entities had sent the keys. When the second channel in transformation (8) provides confidentiality, then so does the resulting channel:

$$\left.\begin{array}{c} A \overset{t_2[t_1]}{\longrightarrow\!\bullet} B \\ A \overset{t_4[t_3]}{\bullet\!\longleftarrow} B \\ t_3 > t_2 \end{array}\right\} \quad \Longrightarrow \quad A \overset{t_4[t_3]}{\bullet\!\longleftarrow\!\bullet} B \tag{9}$$

The timing constraint in (4), (8) and (9) differs from that in (3), (5) and (6) because in the former three B can send the reply only after receiving the message from A.

3.2. Public-key cryptosystems

The basic transformation provided by an (asymmetric) public-key cryptosystem is to transform the authenticity of a channel into the confidentiality of a channel available at some later time:

$$\left.\begin{array}{c} A \overset{t_2[t_1]}{\bullet\!\longrightarrow} B \\ A \overset{t_4[t_3]}{\longleftarrow} B \\ t_3 > t_2 \end{array}\right\} \quad \Longrightarrow \quad A \overset{t_4[t_3]}{\bullet\!\longleftarrow} B \tag{10}$$

When the second channel provides authenticity then so does the resulting channel:

$$\left.\begin{array}{c} A \overset{t_2[t_1]}{\bullet\!\longrightarrow} B \\ A \overset{t_4[t_3]}{\longleftarrow\!\bullet} B \\ t_3 > t_2 \end{array}\right\} \quad \Longrightarrow \quad A \overset{t_4[t_3]}{\bullet\!\longleftarrow\!\bullet} B \tag{11}$$

Our framework demonstrates that a public-key distribution system as defined by Diffie and Hellman [6], if combined with a symmetric cryptosystem for encryption with the generated bilateral secret key, is equivalent to a public-key cryptosystem in the sense that it provides exactly the same transformations (10) and (11). Transformation (11) corresponds to the usual description of a public-key distribution system as allowing two entities to generate a shared secret key by communicating only over insecure but authenticated channels. Transformation (10) can be achieved as follows: User A generates a Diffie-Hellman key pair and sends the public key to B (in an authenticated manner). User B also generates a Diffie-Hellman key pair, computes the secret key shared with A, encrypts the message with this secret key and sends the encrypted message and the public key over the insecure channel to A.

A comparison of transformations (8) and (10) suggests that a public-key cryptosystem is, in a certain sense, the dual of a symmetric message authentication code (MAC): the former permits the conversion of authenticity into confidentiality while the latter provides the inverse conversion.

3.3. Digital signature schemes

There are four possible security transformations on a channel with one security symbol $\bullet$, and three of them have been discussed: transforming $\bullet\!\!\longrightarrow$ into $\bullet\!\!\longleftarrow$, $\longrightarrow\!\!\bullet$ into $\longrightarrow\!\!\bullet$ (transferring the confidentiality of a channel to a later available insecure channel), and $\longrightarrow\!\!\bullet$ into $\longleftarrow\!\!\bullet$. One such transformation is still missing, namely transferring the authenticity of a channel to an insecure channel available at some later time. This missing transformation is provided by a digital signature scheme:

$$\left.\begin{array}{c} A \overset{t_2[t_1]}{\bullet\!\!\longrightarrow} B \\ A \overset{t_4[t_3]}{\longrightarrow} B \\ t_4 \geq t_2 \end{array}\right\} \quad \Longrightarrow \quad A \overset{t_4[t_3]}{\bullet\!\!\longrightarrow} B \tag{12}$$

When the second channel is confidential, then so is the resulting channel:

$$\left.\begin{array}{c} A \overset{t_2[t_1]}{\bullet\!\!\longrightarrow} B \\ A \overset{t_4[t_3]}{\longrightarrow\!\!\bullet} B \\ t_4 \geq t_2 \end{array}\right\} \quad \Longrightarrow \quad A \overset{t_4[t_3]}{\bullet\!\!\longrightarrow\!\!\bullet} B$$

which is identical to (7).

A comparison of transformations (5) and (12) demonstrates that a digital signature scheme is, in a certain sense, the dual of a symmetric cryptosystem: they allow to transfer authenticity and confidentiality, respectively.

4. Limitations of cryptographic security transformations

The transformations discussed in the previous section can be interpreted as tools for moving or replacing channel symbols ($\longrightarrow$) while keeping the security symbols $\bullet$ attached to the corresponding users. For example, transformation (3) can be interpreted as replacing the channel $\overset{t_2[t_1]}{\longrightarrow}$ in $A \overset{t_2[t_1]}{\bullet\!\!\longrightarrow\!\!\bullet} B$ with the channel $\overset{t_4[t_3]}{\longrightarrow}$ while keeping the $\bullet$'s attached to A and B. Hence this transformation permits to transfer the security properties from an initially available secure channel to a later available insecure channel between the same two entities. In this context it is important to notice that a security symbol $\bullet$ is attached to the corresponding entity rather than to the corresponding channel and is valid only with respect to the opposite entity.

It is obvious that $\bullet$'s cannot be "created" or copied from one user to another by cryptographic transformations. Furthermore, the fact that an entity A is exclusive in a certain sense (with respect to another entity B) cannot be transferred to a third entity C (unless C trusts B, see Section 5). Hence security symbols $\bullet$ must be established by non-cryptographic means such as authentication based on speaker identification or on the verification of a passport. These observations are obvious but appear to be impossible to prove formally and are therefore stated as an axiom. By a cryptographic transformation we mean one that does not exploit trust relations.

Axiom 1. *There exists no cryptographic transformation allowing to "create" a $\bullet$ or to copy a $\bullet$ from one user to another. Furthermore, a cryptographic transformation can copy a $\bullet$ only between channels with identical endpoints.*

A typical security goal for a distributed system is that every pair of entities (e.g., A and B) can communicate securely (i.e., over a $A \bullet\!\!\longleftrightarrow\!\!\bullet B$ cannel) at any time after some system setup time t_0. Of course, a necessary condition is that they be able to communicate at all, i.e., that there exists at least an insecure channel $A \overset{t}{\longleftrightarrow} B$ at any time $t > t_0$. For the remainder of this section we therefore make the following assumption, which can be interpreted as a characterization of a reliable open network. The assumption will be dropped again in Section 5.

Full connectivity assumption. *Insecure channels ($\longleftrightarrow$) between every pair of users are always available.*

Proposition 1. *Under the full connectivity assumption it is a sufficient condition for achieving a secure channel between A and B from time t_0 onwards ($A \bullet\!\!\overset{t}{\longleftrightarrow}\!\!\bullet B$ for $t \geq t_0$) that the following two conditions are satisfied:*

(1) there exists a $A \bullet\!\!\overset{t_1}{\longrightarrow} B$ or a $A \bullet\!\!\overset{t_1}{\longleftarrow} B$ channel for some time $t_1 < t_0$, and

(2) there exists a $A \overset{t_2}{\longrightarrow}\!\!\bullet B$ or a $A \overset{t_2}{\longleftarrow}\!\!\bullet B$ channel for some time $t_2 < t_0$.

When only cryptographic transformations are available, this condition is also necessary.

Proof. The fact that the conditions are necessary follows from Axiom 1. For showing that they are also sufficient it is easy to verify that for each of the four combinations of preconditions consistent with conditions (1) and (2) of the proposition there exists a transformation or a sequence of transformations for creating a secure channel. For instance, every confidential channel can be transformed into an authenticated channel by application of transformation (8) and the obtained scenario consisting of two complementary authenticated channels can be transformed into a secure channel by transformation (11). Now transformations (3) and (4) can be applied to complete the proof. □

Remark. A secure channel $A \bullet\!\!\overset{t}{\longrightarrow}\!\!\bullet B$ (e.g. a trusted courier) can be interpreted as two channels $A \bullet\!\!\overset{t}{\longrightarrow} B$ and $A \overset{t}{\longrightarrow}\!\!\bullet B$; hence transformation (3) does not contradict the proposition.

5. Trust-based security transformations

From the point of view of security in distributed systems, Proposition 1 is pessimistic: it states that in order to establish a secure channel between a pair of entities by cryptographic means there must previously exist some secure channel(s) between these entities. This implies that the number of secure channels needed to achieve security between any pair of entities grows quadratically in the number of entities.

In a large distributed system one therefore needs transformations based on trust relations that are not restricted by Axiom 1. Trust is a fundamental ingredient for secure communications in large distributed systems. Various types and degrees of trust can be distinguished (e.g., see [26]), but for the sake of simplicity of the model, such a distinction will not be made in this paper, although our model could be extended in this direction.

When a user B trusts another user or third party T to send only authenticated information (i.e., T is trusted to properly authenticate its sources of information as well as not to fraudulently distribute inaccurate information), then B can use T to connect two authenticated channels $A \overset{t_2[t_1]}{\bullet\!\!\longrightarrow} T$ and $T \overset{t_4[t_3]}{\bullet\!\!\longrightarrow} B$ to result in an authenticated channel from A to B, provided that $t_3 > t_2$, i.e., provided that the message on the $T \overset{t_4[t_3]}{\bullet\!\!\longrightarrow} B$ channel need not be fixed before the first message is received by T on the $A \overset{t_2[t_1]}{\bullet\!\!\longrightarrow} T$ channel:

$$\left.\begin{array}{c} A \overset{t_2[t_1]}{\bullet\!\!\longrightarrow} T \\ T \overset{t_4[t_3]}{\bullet\!\!\longrightarrow} B \\ t_3 > t_2 \\ B \text{ trusts } T \end{array}\right\} \quad \Longrightarrow \quad A \overset{t_4[t_1]}{\bullet\!\!\longrightarrow} B \tag{13}$$

This transformation is a generalization of transformation (2). Note that A need not trust T because authenticity refers to the receiver's (not the sender's) state of belief.

Similarly, when a user A trusts another user or third party T to treat secret information confidentially and to send it only to entities approved by A, then A can use T to connect two confidential channels, provided that $t_3 > t_2$:

$$\left.\begin{array}{c} A \overset{t_2[t_1]}{\longrightarrow\!\!\bullet} T \\ T \overset{t_4[t_3]}{\longrightarrow\!\!\bullet} B \\ t_3 > t_2 \\ A \text{ trusts } T \end{array}\right\} \quad \Longrightarrow \quad A \overset{t_4[t_1]}{\longrightarrow\!\!\bullet} B \tag{14}$$

The following transformation cannot be derived from the previously described transformations because the two messages sent over the channels $A \overset{t_2[t_1]}{\longrightarrow\!\!\bullet} T$ and $T \overset{t_4[t_3]}{\longrightarrow\!\!\bullet} B$ must be correlated. This transformation corresponds to the classical secret key distribution by a trusted third party and requires that A and B both trust T to generate a random session key and to keep it confidential. It also requires the use of a symmetric cryptosystem and is hence both cryptographic and trust-based.

$$\left.\begin{array}{c} T \overset{t_2[t_1]}{\bullet\!\!\longrightarrow} A \\ T \overset{t_4[t_3]}{\bullet\!\!\longrightarrow} B \\ A \overset{t_5}{\longrightarrow} B \\ t_5 > t_2, t_5 \geq t_4 \\ A \text{ and } B \text{ trust } T \end{array}\right\} \quad \Longrightarrow \quad A \overset{t_5}{\bullet\!\!\longrightarrow} B \tag{15}$$

Instead of generating and distributing a secret key for a symmetric cryptosystem, a trusted third party T could generate a secret-key/public-key pair for a public-key cryptosystem or a digital signature scheme and send the secret key to an entity A over a confidential channel and the public key to other entities (e.g. B) over authenticated channels. When B trusts T then

A can use an insecure channel to establish an authenticated channel to B.

$$\left.\begin{array}{c} T \overset{t_2[t_1]}{\longrightarrow\!\bullet} A \\ T \overset{t_4[t_3]}{\bullet\!\longrightarrow} B \\ A \overset{t_5}{\longrightarrow} B \\ t_5 > t_2, t_5 \geq t_4 \\ B \text{ trusts } T \end{array}\right\} \quad \Longrightarrow \quad A \overset{t_5}{\bullet\!\longrightarrow} B \tag{16}$$

A similar transformation would yield a $A \overset{t_5}{\longleftarrow\!\bullet} B$ channel.

The two channel uses by T in transformations (15) and (16) are correlated. This is generally undesirable because in a large scale system different entities' interactions with T should be independent.

The probably most important application of digital signatures (transformation (12)) is for achieving the trust-based transformation (13) even when $t_3 < t_2$. Clearly, no communication can take place from A to B in this case unless some additional insecure channels are used. A first approach to introducing additional insecure channels could be to use an insecure channel $T \overset{t}{\longrightarrow} B$ from T to B at some time $t > t_2$. Thus transformation (12) would yield a channel $T \overset{t}{\bullet\!\longrightarrow} B$ and hence, by application of (13), a channel $A \overset{t[t_1]}{\bullet\!\longrightarrow} B$. The drawback of this approach is that T would have to participate actively in the communication from A to B.

A more realistic scenario for achieving transformation (13) for the case $t_3 < t_2$ is shown in Figure 1. It corresponds to the well-known certification of public keys by a trusted user or trusted third party. Here we make use of insecure channels $T \overset{t_5}{\longrightarrow} A$ and $A \overset{t_6}{\longrightarrow} B$. The interaction between A and T is independent of the message sent by T over the $T \overset{t_4[t_3]}{\bullet\!\longrightarrow} B$ channel.

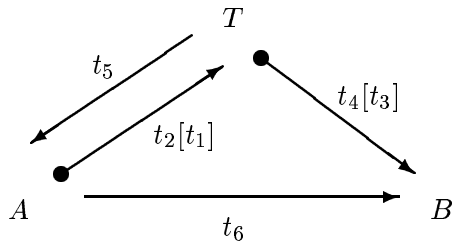

Figure 1. A channel scenario for establishing an authenticated channel $A \bullet\!\longrightarrow B$ from A to B. A third party T trusted by B can issue a public-key certificate to A and thereby provide a link between the two authenticated channels $A \bullet\!\longrightarrow T$ and $T \bullet\!\longrightarrow B$.

We can apply the following sequence of transformations. For $t_6 > t_5$ transformation (2) gives

$$\left.\begin{array}{c} T \xrightarrow{t_5} A \\ A \xrightarrow{t_6} B \\ t_6 > t_5 \end{array}\right\} \quad \stackrel{(2)}{\Longrightarrow} \quad T \xrightarrow{t_6[t_5]} B \tag{17}$$

Now transformation (12) for digital signatures yields

$$\left.\begin{array}{c} T \overset{t_4[t_3]}{\bullet\!\!\longrightarrow} B \\ T \xrightarrow{t_6[t_5]} B \\ t_6 > t_4 \end{array}\right\} \quad \stackrel{(12)}{\Longrightarrow} \quad T \overset{t_6[t_5]}{\bullet\!\!\longrightarrow} B \tag{18}$$

When B trusts T we can now apply transformation (13), provided that $t_5 > t_2$. This results in

$$\left.\begin{array}{c} A \overset{t_2[t_1]}{\bullet\!\!\longrightarrow} T \\ T \overset{t_6[t_5]}{\bullet\!\!\longrightarrow} B \\ t_5 > t_2 \\ B \text{ trusts } T \end{array}\right\} \quad \stackrel{(13)}{\Longrightarrow} \quad A \overset{t_6[t_1]}{\bullet\!\!\longrightarrow} B \tag{19}$$

which together with transformation (12) for digital signatures gives the desired result, an authenticated channel from A to B at time t_6:

$$\left.\begin{array}{c} A \overset{t_6[t_1]}{\bullet\!\!\longrightarrow} B \\ A \xrightarrow{t_6} B \end{array}\right\} \quad \stackrel{(12)}{\Longrightarrow} \quad A \overset{t_6}{\bullet\!\!\longrightarrow} B \tag{20}$$

Because we have assumed that $t_3 > t_2$, the applications of transformations (2) and (13) are unavoidable. Hence the above derivation illustrates that an authenticated channel $A \overset{t_6}{\bullet\!\!\longrightarrow} B$ can be derived from the scenario of Figure 1 if and only if $t_2 < t_5 < t_6$ and only if B trusts T. Two applications of a digital signature scheme are required, first by T for certifying A's public key and secondly by A to authenticate actual messages. In this model of public-key certification, user A serves as a relay according to transformation (2) for a message (his own public key) conceptually sent from T to B.

6. The necessary and sufficient condition for a secure channel $A \bullet\!\!\longleftrightarrow\!\!\bullet B$ between two entities

For two reasons a model based on a single trusted third party T as discussed in the previous section is unrealistic in many real-world scenarios of large distributed systems. First, there generally exists no single third party that is trusted by every entity. Second, a secure channel between A and T is usually established by A visiting T, presenting a passport and exchanging the necessary messages with T. In a world-wide system it is impractical for every entity to visit such a world-wide trusted third party T.

The second problem can be solved by using more than one trusted third party with a geographically sufficiently dense representation and typically organized in a hierarchy. However,

in applications where entities cannot be requested to trust certain specified authorities, solving the first problem typically requires a very large number of trusted entities with a decentralized trust management (see Section 7).

Inspecting the trust-based transformations of the previous section leads to the following characterization of the limitations of trust-based transformations. It appears impossible to prove formally and is therefore stated as an axiom. The symbol $\bullet\!\!-\!\!\circ$ stands for any channel of either direction where the symbol $\circ$ can either be deleted or replaced by $\bullet$.

Axiom 2. *There exists no trust-based transformation allowing to "create" a $\bullet$ or to copy a $\bullet$ from one user to another. However, a trust-based transformation can copy the $\bullet$ from a channel $A \bullet\!\!-\!\!\circ T$ to become the $\bullet$ in the channel $A \bullet\!\!-\!\!\circ B$, but only if at least two conditions are satisfied: (1) there also exists a channel $T \bullet\!\!-\!\!\circ B$, and (2) B trusts T.*

When a security derivation for a channel leading to or from entity B implies that B must trust some entity T' and further implies that T' must trust some entity T, then this derivation also implies that B must trust T. If B would not trust T it would be useless that T' trusts T. The fact that T' trusts T can rather be seen as a possible reason for B to trust T in which case B follows a recommendation by T'. This fact is restated as the following axiom. Note that the trust between entities can be formalized as a relation

$$\mathcal{T} \subseteq \mathcal{E} \times \mathcal{E},$$

where $\mathcal{E}$ is the set of entities. For two entities i and j, $(i,j) \in \mathcal{T}$ if and only if i trusts j.

Axiom 3. *The trust relation $\mathcal{T}$ needed as a precondition in a security derivation (consisting of a sequence of transformations) must be extended to its transitive closure.*

We also refer to [18] for a discussion on the transitivity of the trust relation. The following proposition characterizes the necessary and sufficient conditions for establishing an authenticated channel between two entities in a distributed system.

Proposition 2. *Under the full connectivity assumption it is the necessary and sufficient condition for establishing an authenticated channel from A to B from time t_0 onwards ($A \bullet\!\!\xrightarrow{t} B$ for $t \geq t_0$), or a confidential channel from B to A from time t_0 onwards ($A \xleftarrow{t}\!\!\bullet\ B$ for $t \geq t_0$), that there exists a connected path of channels (of arbitrary individual directions) from A to B such that*

(1) every channel in the path is available at some time earlier than t_0 and has a $\bullet$ attached to that end of the channel which is closer to A, and

(2) B trusts every intermediate entity on the path.

Proof. Assume that every confidential but not authenticated channel ($\rightarrow\!\!\bullet$) is converted by transformation (8) into an authenticated channel ($\leftarrow\!\!\bullet$). After these transformations a path according to the proposition is a complete chain of authenticated channels from A to B where B trusts all the intermediate entities in the chain. According to the sequence of transformations (17) to (20) discussed in the previous section one can repeatedly combine the two links of the

chain closest to B to a single authenticated channel. This finally results in a single authenticated channel $A \bullet\!\!\longrightarrow B$ from A to B.

In order to show that the conditions stated in Proposition 2 are also necessary, first observe that all the exclusiveness symbols ($\bullet$) are needed because if one of the entities (say T) in a chain had no exclusiveness symbol on the link $T \text{ —— } T'$ leading towards B, then T could be impersonated by an opponent. More formally, after combining each of the two subchains of authenticated channels into a single authenticated channel ($A \bullet\!\!\longrightarrow T$ and $T' \bullet\!\!\longrightarrow B$), it would be a contradiction to Axiom 2 when a channel $A \bullet\!\!\longrightarrow B$ could be constructed without a $T \bullet\!\!-\!\!\circ T'$ channel.

It remains to be shown that B must trust every entity in the chain. No matter what sequence of transformations is used to consecutively combine two authenticated channels to a new authenticated channel, finally resulting in a single channel $A \bullet\!\!\longrightarrow B$, the involved trust requirements are such that for every entity T on the path, either B must trust T or B must trust an entity T' on the path connecting T and B, where T' must trust T. According to Axiom 3 the latter case also implies that B must trust T. Hence B must trust every entity on the path. □

The following proposition follows from Propositions 1 and 2. It is important to note that the two paths can be disjoint except for the endpoints A and B.

Proposition 3. *Under the full connectivity assumption it is the necessary and sufficient condition for achieving a secure channel between A and B from time t_0 onwards ($A \bullet\!\!\overset{t}{\longleftrightarrow}\!\!\bullet B$ for $t \geq t_0$) that there exist two paths of channels according to Proposition 2, one from A to B and one (with symmetric conditions) from B to A.*

Example: Consider the somewhat artificial scenario of Figure 2: Assume for simplicity that T_1, T_2 and T_3 are trusted by both A and B, that the channels are available at the indicated times and that insecure channels are freely available (full connectivity assumption). In order to determine the earliest time after which A and B can communicate securely, one has to find two paths as required by Proposition 3 with the smallest possible maximal path time on the paths. Remember that because of the full connectivity assumption the direction of the channels is irrelevant. In this example there exist two paths from A to B, namely $A \bullet\!\!\overset{t_2}{\longleftarrow}\!\!\bullet T_3 \bullet\!\!\overset{t_6}{\longrightarrow} B$ and $A \bullet\!\!\overset{t_1}{\longrightarrow} T_1 \bullet\!\!\overset{t_3}{\longrightarrow} T_3 \bullet\!\!\overset{t_6}{\longrightarrow} B$, and one path from B to A, namely $B \bullet\!\!\overset{t_7}{\longleftarrow} T_2 \bullet\!\!\overset{t_4}{\longrightarrow} T_1 \bullet\!\!\overset{t_3}{\longrightarrow} T_3 \bullet\!\!\overset{t_2}{\longrightarrow}\!\!\bullet A$. Hence the earliest time for secure communication between A and B is $\max(t_2, t_3, t_4, t_6, t_7, \min(t_2, \max(t_1, t_3)))$.

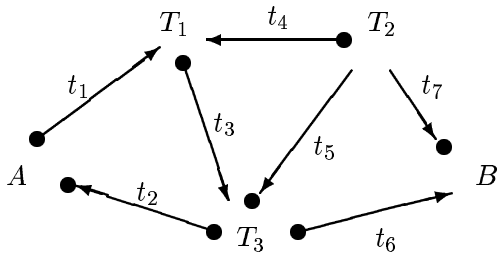

Figure 2. A security bootstrapping scenario involving three third parties trusted by both A and B.

7. Bootstrapping security in distributed systems

In the previous section we have discussed the theoretical limitations to establishing secure channels in a distributed system. This section reviews, within our framework and at a high level of abstraction, some previously proposed protocols for establishing a secure channel between two entities.

7.1. Protocols based on symmetric cryptography

Consider a trusted third party T to which every user establishes a secure channel during the initialization phase. Hence we can exploit channels $T \bullet\!\!\xrightarrow{t_1}\!\!\bullet\, A$ and $T \bullet\!\!\xrightarrow{t_2}\!\!\bullet\, B$ for some t_1 and t_2. The first two protocols are discussed for illustration purposes only and for the sake of completeness, but these protocols are not used in distributed systems.

7.1.1. Distribution of bilateral secret keys

The probably simplest protocol, which is used in some military and diplomatic applications with small user groups, is for T to generate a bilateral secret key for every pair of entities and to use the secure channels for sending each entity the corresponding subset of bilateral keys. This approach is completely impractical in large networks because for every pair (A, B) of entities the communication on the two channels $T \bullet\!\!\xrightarrow{t_1}\!\!\bullet\, A$ and $T \bullet\!\!\xrightarrow{t_2}\!\!\bullet\, B$ is necessarily correlated, which implies a quadratic (in the number of entitites) growth of complexity. A and B can either use their bilateral key directly for encrypting the communications or, preferably, for the secure exchange of a new encryption key for each session.

7.1.2. Message or session key relaying by a trusted server

The major drawback of the above approach is the quadratic growth of the number of session keys generated and distributed by T. This problem can be avoided when T is available *on-line* for all the entities. The simplest such protocol, involving transformations (9), (13) and (14), is when T serves as a relay for messages. When A wants to send a message to B, he or she encrypts it with the secret key shared with T and sends it to T using a channel $A \longrightarrow T$. T can decrypt the message, reencrypt it with the secret key shared with B, and send the new ciphertext to B using a $T \longrightarrow B$ channel.

A more reasonable protocol additionally requiring direct interaction between A and B can be derived from the so-called wide-mouthed-frog protocol proposed by Burrows [4]. Here, T merely relays a session key generated by A for communication between A and B, and the subsequent encrypted communication between A and B takes place over an insecure channel.

7.1.3. Session key distribution by a trusted server

The following key distribution paradigm is used in the Kerberos [21], [16] and in the KryptoKnight [13] systems. One of its advantages is that users need not be capable of generating "good" encryption keys. Otway and Rees [17] have proposed a similar protocol with a different sequence of interactions with the trusted server T. Figure 3 illustrates the scenario: During the initialization phase, T agrees on a bilateral secret key with every entity (in our example at

time t_1 with A and at time t_2 with B).

When A wants to communicate with B she uses the $A \xrightarrow{t_3} T$ channel for sending a request to T. T generates a session key, sends the encrypted session key to A, together with the same session key encrypted for B (i.e., encrypted with the key shared by T and B). A can then initiate a communication with B by sending the encrypted session key. Note that this protocol, like that discussed in Section 7.1.1, makes use of transformation (15), but in a quite different manner. In particular, T does not send the session key directly to B.

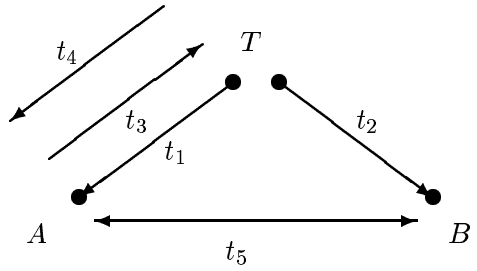

Figure 3. A channel scenario for the distribution of a session key by a trusted server T to A and B, as it is used in Kerberos [21] and KryptoKnight [13].

The sequence of transformations used for obtaining a secure channel $A \overset{t_5}{\bullet\!\!\longrightarrow\!\!\bullet} B$ is as follows: When $t_5 > t_4$ then the channels $T \xrightarrow{t_4} A$ and $A \xrightarrow{t_5} B$ can be connected by transformation (2):

$$\left.\begin{array}{c} T \xrightarrow{t_4} A \\ A \xrightarrow{t_5} B \\ t_5 > t_4 \end{array}\right\} \overset{(2)}{\Longrightarrow} \quad T \xrightarrow{t_5[t_4]} B$$

Now transformation (3) applied to channels $T \xrightarrow{t_5[t_4]} B$ and $T \overset{t_2}{\bullet\!\!\longrightarrow\!\!\bullet} B$ gives

$$\left.\begin{array}{c} T \overset{t_2}{\bullet\!\!\longrightarrow\!\!\bullet} B \\ T \xrightarrow{t_5[t_4]} B \\ t_5 \geq t_2 \end{array}\right\} \overset{(3)}{\Longrightarrow} \quad T \overset{t_5[t_4]}{\bullet\!\!\longrightarrow\!\!\bullet} B$$

Channels $T \overset{t_1}{\bullet\!\!\longrightarrow\!\!\bullet} A$ and $T \xrightarrow{t_4} A$ are combined by a second application of (3):

$$\left.\begin{array}{c} T \overset{t_1}{\bullet\!\!\longrightarrow\!\!\bullet} A \\ T \xrightarrow{t_4} A \\ t_4 > t_1 \end{array}\right\} \overset{(3)}{\Longrightarrow} \quad T \overset{t_4}{\bullet\!\!\longrightarrow\!\!\bullet} A$$

Now $T \overset{t_4}{\bullet\!\!\rightarrow\!\!\bullet} A$, $T \overset{t_5[t_4]}{\bullet\!\!\rightarrow\!\!\bullet} B$ and $A \overset{t_5}{\longrightarrow} B$ can be used for key distribution according to (15):

$$\left.\begin{array}{c} T \overset{t_4}{\bullet\!\!\rightarrow\!\!\bullet} A \\ T \overset{t_5[t_4]}{\bullet\!\!\rightarrow\!\!\bullet} B \\ A \overset{t_5}{\longrightarrow} B \\ t_5 > t_4 \\ A \text{ and } B \text{ both trust } T \end{array}\right\} \overset{(15)}{\Longrightarrow} \quad A \overset{t_5}{\bullet\!\!\rightarrow\!\!\bullet} B$$

It may be desirable for A to generate the session key herself. In this case, a modified version of the above protocol in which $A \overset{t_3}{\bullet\!\!\rightarrow\!\!\bullet} T$ is created from $T \overset{t_1}{\bullet\!\!\rightarrow\!\!\bullet} A$ and $A \overset{t_3}{\longrightarrow} T$ by application of (4), $A \overset{t_5[t_3]}{\bullet\!\!\rightarrow\!\!\bullet} B$ is created from $A \overset{t_3}{\bullet\!\!\rightarrow\!\!\bullet} T$ and $T \overset{t_5[t_4]}{\bullet\!\!\rightarrow\!\!\bullet} B$ by application of (13) and (14), and $A \overset{t_5}{\bullet\!\!\rightarrow\!\!\bullet} B$ is created from $A \overset{t_5[t_3]}{\bullet\!\!\rightarrow\!\!\bullet} B$ and $A \overset{t_5}{\longrightarrow} B$ by application of (3). This sequence of transformations requires that $t_1 < t_3 < t_4 < t_5$.

7.2. Protocols based on public key certification

A drawback of the protocols described in Section 7.1 (and of all protocols based solely on symmetric cryptography) is that either a trusted third party T must be available on-line to distribute session keys, or the initial secure communications between different users and T must be correlated, resulting in a quadratic growth of complexity of the key distribution problem. This problem of requiring a third party for every session establishment can be solved by using certified public keys as mentioned in Section 4. However, in a very large network such as the Internet, several trusted authorities are required to make the system practical, i.e., to provide all the paths of channels required by Proposition 3.

7.2.1. Hierarchical public key certification

The certification authorities can be organized in a hierarchy as suggested in [28], in which each authority can certify the public key of lower-level authorities.

A simple scenario with a two-level hierarchy is shown in Figure 4: T_1 is a system-wide authority which certifies public keys of regional authorities (T_2 and T_3). A and B obtain certificates for their public keys from T_2 and T_3, respectively. Such a certification step consists of sending the public key, over an authenticated channel, to the higher-level authority and receiving from it, over another authenticated channel, the certified public key together with the public keys and certificates of all authorities on the path to the top authority T_1. For instance, the message transmitted on the $T_2 \overset{t_2}{\bullet\!\!\rightarrow\!\!\bullet} A$ channel consists of T_1's public key, T_2's public key and a certificate (issued by T_1) for it, and the certificate (issued by T_2) for A's public key. (The certificates could actually be sent over insecure channels.) Typically, the authenticated channels needed in a certification step could be realized by mutual identification when a user registers with an authority.

A user's public key may consist of two components, the public keys for a digital signature scheme and for a public-key cryptosystem (or a public-key distribution system). When the RSA system [19] is used, one public key is sufficient because this system can be used both as a digital signature scheme and as a public-key cryptosystem.

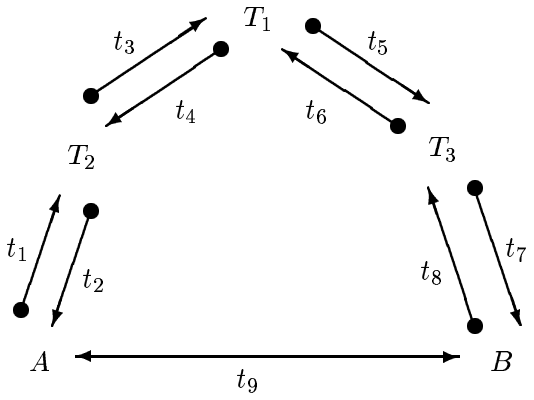

Figure 4. The channel scenario needed for hierarchical public key certification involving three trusted third parties with T_1 at the top of the hierarchy.

A formal derivation using security transformations demonstrates the constraints on the timing of the channels and the trust relations required to achieve a secure channel between A and B: both A and B must trust each authority on the path between A and B. The sequence of transformations resulting in an authenticated channel $A \overset{t_9}{\bullet\!\!\longrightarrow} B$ from A to B is shown in Figure 5.

$$\left.\begin{array}{c} T_2 \overset{t_2}{\bullet\!\!\longrightarrow} A \\ A \overset{t_9}{\longrightarrow} B \\ t_9 > t_2 \end{array}\right\} \overset{(1),(2)}{\Longrightarrow} T_2 \overset{t_9[t_2]}{\longrightarrow} B \qquad \left.\begin{array}{c} T_2 \overset{t_3}{\bullet\!\!\longrightarrow} T_1 \\ T_1 \overset{t_9[t_4]}{\bullet\!\!\longrightarrow} B \\ t_4 > t_3 \\ B \text{ trusts } T_1 \end{array}\right\} \overset{(13)}{\Longrightarrow} T_2 \overset{t_9[t_3]}{\bullet\!\!\longrightarrow} B$$

$$\left.\begin{array}{c} T_1 \overset{t_4}{\bullet\!\!\longrightarrow} T_2 \\ T_2 \overset{t_9[t_2]}{\longrightarrow} B \\ t_2 > t_4 \end{array}\right\} \overset{(1),(2)}{\Longrightarrow} T_1 \overset{t_9[t_4]}{\longrightarrow} B \qquad \left.\begin{array}{c} T_2 \overset{t_9[t_3]}{\longrightarrow} B \\ T_2 \overset{t_9[t_2]}{\longrightarrow} B \end{array}\right\} \overset{(12)}{\Longrightarrow} T_2 \overset{t_9[t_2]}{\bullet\!\!\longrightarrow} B$$

$$\left.\begin{array}{c} T_1 \overset{t_5}{\bullet\!\!\longrightarrow} T_3 \\ T_3 \overset{t_7}{\bullet\!\!\longrightarrow} B \\ t_7 > t_5 \\ B \text{ trusts } T_3 \end{array}\right\} \overset{(13)}{\Longrightarrow} T_1 \overset{t_7[t_5]}{\bullet\!\!\longrightarrow} B \qquad \left.\begin{array}{c} A \overset{t_1}{\bullet\!\!\longrightarrow} T_2 \\ T_2 \overset{t_9[t_2]}{\bullet\!\!\longrightarrow} B \\ t_2 > t_1 \\ B \text{ trusts } T_2 \end{array}\right\} \overset{(13)}{\Longrightarrow} A \overset{t_9[t_1]}{\bullet\!\!\longrightarrow} B$$

$$\left.\begin{array}{c} T_1 \overset{t_7[t_5]}{\bullet\!\!\longrightarrow} B \\ T_1 \overset{t_9[t_4]}{\longrightarrow} B \\ t_9 > t_7 \end{array}\right\} \overset{(12)}{\Longrightarrow} T_1 \overset{t_9[t_4]}{\bullet\!\!\longrightarrow} B \qquad \left.\begin{array}{c} A \overset{t_9[t_1]}{\bullet\!\!\longrightarrow} B \\ A \overset{t_9}{\longrightarrow} B \end{array}\right\} \overset{(12)}{\Longrightarrow} A \overset{t_9}{\bullet\!\!\longrightarrow} B$$

Figure 5. The sequence of transformation for exploiting the certification hierarchy of Figure 4 to establish an authenticated channel $A \overset{t_9}{\bullet\!\!\longrightarrow} B$ from A to B.

Note that this sequence of transformations requires that $t_1 < t_2$, $t_3 < t_4$, $t_5 < t_7$ and $t_4 < t_2 < t_9$ and also that B trusts T_1, T_2 and T_3. The condition $t_4 < t_2 < t_9$ is an interpretation of the fact that T_2 must provide A with T_1's public key as well as with T_1's certificate for his own public key.

A similar sequence of transformations results in a channel $A \xleftarrow{t_9}\bullet\ B$ under the conditions $t_8 < t_7$, $t_6 < t_5$, $t_4 < t_2$ and $t_5 < t_7 < t_9$ and provided that A trusts T_1, T_2 and T_3. Finally, a secure channel $A \bullet\!\!\xleftrightarrow{t_9}\!\!\bullet\ B$ can be obtained by application of transformation (11).

7.2.2. Non-hierarchical public key certification

In large and heterogeneous distributed systems, hierarchical certification schemes with a tree-shaped topology are critical for two reasons. First, if a single third party is compromised, this leads to the failure of a substantial part of a system. Second, and more importantly, both users must trust *every* third party on the certification path, i.e., a certification path can be at most as strong as its weakest link. As a consequence, a user's trust in an third party T_i is in most cases based on a higher-level authority's recommendation issued according to some policy. Such delegation of trust within a rigid hierarchical structure appears to be quite critical in many application scenarios. Trust management is an important emerging research area in distributed system security (e.g., see [26], [1]).

It is crucial in large and heterogeneous distributed systems that not only the communication links, but also the certification paths be highly redundant. Zimmermann's Pretty Good Privacy (PGP) software [27] (see also [22]) allows for a very flexible use of certificates, leaving the responsibility completely in the hands of the users. PGP exploits Proposition 2 and Corollary 3 in their full generality. The need for a international public-key infrastructure is also discussed in [5]. However, it is possible that a more general approach allowing to fully express and exploit various degrees of trust and combine certificates of different trust levels is needed. A first approach in this direction is described in [1].

8. Concluding remarks

When compared to secret-key cryptography, the application of public-key cryptography offers two major potential advantages in key management protocols.

First, protocols based on secret-key cryptography require a trusted third party T to be available *on-line* [21], [13] for all entities or, alternatively, that the interactions between individual entities and T are correlated, thus creating a key management complexity that is quadratic in the number of entities (see Section 7.1). In contrast, the protocols based on public-key certificates described in Section 7.2 require each entity only to perform an initial interaction with one or several certification authorities, which are not involved later in establishing sessions between entities. However, it should be pointed out that this advantage may have to be sacrificed to some extent in a large-scale application because the *revocation* of invalid certificates may require an on-line examination of a certificate revocation list.

The second and more crucial advantage of public-key cryptography is that it permits a completely flexible trust management. A certificate-based protocol allows every entity to individually decide which entities it trusts for issuing certificates [27] or, more generally, for

issuing recommendations. In contrast, secret-key protocols are based on a fixed structure of key servers. An entity can generally not choose which inter-realm path of servers [10] should be used for establishing a session key, in particular because the two communicating entities' trust relations may not be compatible. An entity is hence forced to trust all servers in a path specified by the system rather than by itself. Many of these servers (or the organizations running these servers) may be unknown to this user. As an alternative, it is conceivable that a global distributed web of trust and public-key certificates will emerge, as was suggested by Phil Zimmerman [27].

The discussion of Sections 2 and 3 and Proposition 1 illustrate the duality between authenticity and confidentiality. While authenticated channels without confidentiality ($\bullet\!\!\longrightarrow$) are used routinely in distributed systems, Propositions 1,2 and 3 suggest that confidential channels without authenticity ($\longrightarrow\!\!\bullet$) could be used equally well. It remains an interesting open question how such channels could be obtained in a realistic scenario.

To illustrate that such channels need not be completely unrealistic, consider for instance a user B with several accounts on various machines on a large network with cryptographically protected channels between his personal computer and these machines. It may be reasonable to assume that an eavesdropper could simultaneously access messages sent by another user A to a few of these machines, but that he is unable to access all the messages sent to these machines at a given time. In such a scenario, a $A \longrightarrow\!\!\bullet\ B$ channel to be exploited in one of the transformations (5), (6), (8), (9), and (14) could for instance be established by A by dividing a secret key S into various pieces such that all pieces are required to compute S, and sending (without authenticity, and preferably from accounts on different machines) the individual pieces to B's mailboxes on the various machines.

Acknowledgements

We would like to thank Colin Boyd for drawing our attention to reference [3] and an anonymous referee for several suggestions for improving content and presentation of the paper.

References

[1] T. Beth, M. Borcherding and B. Klein, Valuation of trust in open systems, *Computer Security – ESORICS '94*, D. Gollmann (Ed.), Lecture Notes in Computer Science, Berlin: Springer-Verlag, Vol. 875, pp. 3-18.

[2] A. Birell, B. Lampson, R. Needham and M. Schroeder, A global authentication service without global trust, *Proc. IEEE Symposium on Research in Security and Privacy*, 1986, pp. 223–230.

[3] C. Boyd, Security architectures using formal methods, *IEEE Journal on Selected Areas in Communications*, Vol. 11, No. 5, 1993, pp. 694-701.

[4] M. Burrows, M. Abadi and R. Needham, A logic of authentication, *ACM Transactions on Computer Systems*, Vol. 8, No. 1, 1990, pp. 18–36.

[5] S. Chokhani, Towards a national public-key infrastructure, *IEEE Communications Magazine*, Vol. 32, No. 9, 1994, pp. 70-74.

[6] W. Diffie and M.E. Hellman, New directions in cryptography, *IEEE Transactions on Information Theory*, Vol. 22, No. 6, 1976, pp. 644–654.

[7] R. Ganesan and R. Sandhu, Securing Cyberspace, *Communications of the ACM*, Vol. 37, No. 11, 1994, pp. 29-31.

[8] M. Gasser, A. Goldstein, C. Kaufman and B. Lampson, The Digital distributed system security architecture, *Proc. 12th National Computer Security Conference*, NIST/NCSC, Baltimore, 1989, pp. 305–319.

[9] J. Glasgow, G. MacEwen and P. Panangaden, A logic for reasoning about security, *ACM Transactions on Computer Systems*, Vol. 10, No. 3, 1992, pp. 226–264.

[10] V.D. Gligor, S.-W. Luan and J.N. Pato, On inter-realm authentication in large distributed systems, *Proc. IEEE Conference on security and privacy*, 1992, pp. 2–17.

[11] B. Lampson, M. Abadi, M. Burrows and E. Wobber, Authentication in distributed systems: theory and practice, Proc. 13th ACM Symp. on Operating Systems Principles, 1991, pp. 165–182.

[12] J. Linn, Privacy enhancement for internet electronic mail: Part I, Message encipherment and authentication procedures, Internet RFC 1421, Feb. 1993.

[13] R. Molva, G. Tsudik, E. Van Herreweghen and S. Zatti, "KryptoKnight Authentication and Key Distribution System", *Proc. 1992 European Symposium on Research in Computer Security (ESORICS 92)*, Toulouse (Nov.92).

[14] R.M. Needham, Denial of service: an example, *Communications of the ACM*, Vol. 37, No. 11, 1994, pp. 42–46.

[15] R.M. Needham and M.D. Schroeder, Using encryption for authentication in large networks of computers, *Communications of the ACM*, Vol. 21, 1978, pp. 993–999.

[16] B.C. Neuman and T. Ts'o, Kerberos: an authentication service for computer networks, *IEEE Communications Magazine*, Vol. 32, No. 9, 1994, pp. 33–38.

[17] D. Otway and O. Rees, Efficient and timely mutual authentication, *Operating systems review*, Vol. 21, No. 1, 1987, pp. 8–10.

[18] P.V. Rangan, An axiomatic theory of trust in secure communication protocols, *Computers & Security*, Vol. 11, 1992, pp. 163–172.

[19] R.L. Rivest, A. Shamir, and L. Adleman, A method for obtaining digital signatures and public-key cryptosystems, *Communications of the ACM*, Vol. 21, No. 2, 1978, pp. 120–126.

[20] R.A. Rueppel, A formal approach to security architectures, *Advances in Cryptology - EUROCRYPT '91*, Lecture Notes in Computer Science, Vol. 547, pp. 387–398, Berlin: Springer-Verlag, 1991.

[21] J.G. Steiner, B.C. Neuman and J.I. Schiller, Kerberos: An authentication service for open network systems, Proceedings of *Winter USENIX 1988*, Dallas, Texas.

[22] W. Stallings, *Network and Internetwork Security*, Englewood Cliffs, NJ: Prentice Hall, 1995.

[23] P. Syverson and C. Meadows, A logical language for specifying cryptographic protocols requirements, *Proc. IEEE Conf. on Research in Security and Privacy*, 1993, pp. 165–180.

[24] J.J. Tardo and K. Alagappan, SPX: Global authentication using public key certificates, *Proc. IEEE Conf. on Research in Security and Privacy*, 1991, pp. 232–244.

[25] V. Voydock and S. Kent, Security mechanisms in high-level network protocols, *ACM Computing Surveys*, Vol. 15, No. 2, 1983, pp. 135–171.

[26] R. Yahalom, B. Klein and T. Beth, Trust relationships in secure systems – a distributed authentication perspective, *Proc. IEEE Conf. on Research in Security and Privacy*, 1993, pp. 150–164.

[27] P. Zimmermann, PGP User's Guide, Vol. I and II, Version 2.6, May 22, 1994.

[28] ISO/IEC International Standard 9594-8, Information technology – open systems interconnection – the directory, Part 8: Authentication framework, 1990.