Authentication and key distribution in computer networks and distributed systems

Rolf Oppliger
University of Berne
Institute for Computer Science and Applied Mathematics (IAM)
Neubrückstrasse 10, CH-3012 Bern
Phone +41 31 631 89 57, Fax +41 31 631 39 65, oppliger@iam.unibe.ch

Abstract
Authentication and key distribution systems are used in computer networks and distributed systems to provide security services at the application layer. There are several authentication and key distribution systems currently available, and this paper focuses on Kerberos (OSF DCE), NetSP, SPX, TESS and SESAME. The systems are outlined and reviewed with special regard to the security services they offer, the cryptographic techniques they use, their conformance to international standards, and their availability and exportability.

Keywords
Authentication, key distribution, Kerberos, NetSP, SPX, TESS, SESAME

1 INTRODUCTION

Authentication and key distribution systems are used in computer networks and distributed systems to provide security services at the application layer. There are several authentication and key distribution systems currently available, and this paper focuses on Kerberos (OSF DCE), NetSP, SPX, TESS and SESAME. The systems are outlined and reviewed with special regard to the security services they offer, the cryptographic techniques they use, their conformance to international standards, and their availability and exportability. It is assumed that the reader of this paper is familiar with the fundamentals of cryptography, and the use of cryptographic techniques in computer networks and distributed systems (Oppliger, 1992 and Schneier, 1994). The following notation is used in this paper:

- Capital letters are used to refer to principals (users, clients and servers). Letters starting from A are used to refer to users, whereas C (S) is used to refer to a client (server).

- K is used to refer to a key from a secret key cryptosystem. Principals may be added as subscripts; K_p is only known to P (and maybe some central authentication server), whereas K_{pq} is shared between P and Q.
- (k, k^{-1}) is used to refer to a public key pair; k is the public key, and k^{-1} is the corresponding private key. Again, principals may be added as subscripts.
- The expression $\{M\}K$ is used to refer to a message M that is encrypted with a secret key K. The same key is used for decryption, so $\{\{M\}K\}K$ equals M. Similarly, $\{M\}k$ refers to a message M that is encrypted with a public key k. The message can only be decrypted with the corresponding private key k^{-1}. If a public key cryptosystem is used to digitally sign messages, the private key is used for signing, and the corresponding public key is used for verifying the signatures. In this case, $\{M\}k^{-1}$ refers to a digital signature giving message recovery, and $[M]k^{-1}$ to a digital signature with appendix. In both cases, the signature can be verified only with the corresponding public key k.
- T is used to refer to a timestamp. Subscripts may imply a temporal ordering.
- A nonce is a fresh and unpredictable random number. N is used to refer to a nonce. In this case, a subscript may specify a principal.

An authentication and key distribution system implements cryptographic protocols. In general, a protocol specifies the format and relative timing of information exchanged between communicating parties. The expression i: P $\longrightarrow$ Q : M is used to denote step i, in which P transmits a message M to Q. Note that the notation of $\longrightarrow$ must be interpreted with care. The messages are sent in environments, where error, corruption, loss and delay may occur. There is nothing in the environment to guarantee that messages are really made in numerical order by the principals indicated, received in numerical order or at all by the principals indicated, or received solely by the principals indicated.

2 KERBEROS

The authentication and key distribution system Kerberos has been developed at the Massachusetts Institute of Technology (MIT) to protect the emerging network services provided by the Athena project (Schiller, 1994). Kerberos versions 1 through 3 were used at the MIT only. Kerberos version 4 (V4) was made publicly available and is in widespread use today. Work on Kerberos version 5 (V5) commenced in 1989, fueled by discussions with V4 users and administrators about their experiences with the protocol and MIT's implementation. In September 1993 Kerberos V5 was specified as an Internet standards track protocol (Kohl and Neuman, 1993). Today, several vendors provide their own versions of Kerberos V4 and V5.

Kerberos is organized in realms. In every realm, there is a central and physically secure authentication server (AS) to share a secret key K_p with every principal P. If P is a user, then K_p is derived from P's password by applying a one-way hash function to it; otherwise K_p is explicitly given.

Kerberos works by providing principals with both tickets that they can use to identify themselves to other principals and secret keys for secure communication with other principals. The AS authenticates users as they sign-on, and provides them with a ticket granting ticket (TGT). The TGT can be used to get tickets from a ticket granting server (TGS), and these tickets can further be used as credentials to contact particular servers.

$T_{c,s} = \{U, C, S, K, T_{start}, T_{expire}\}K_s$ refers to a ticket that a client C can use to contact a server S on a user U's behalf. The ticket includes the principal identifiers of both U and S, C's network address, a session key K, a start time T_{start}, and an expiration time T_{expire}. The ticket is encrypted with K_s, the secret key of S, so C won't be able to read or modify it. In order to protect it against replay attacks, C generates and additionally sends an authenticator $A_{c,s} = \{C, T\}K$ that includes both C's network address and a timestamp T. The authenticator is encrypted with the session key K, too.

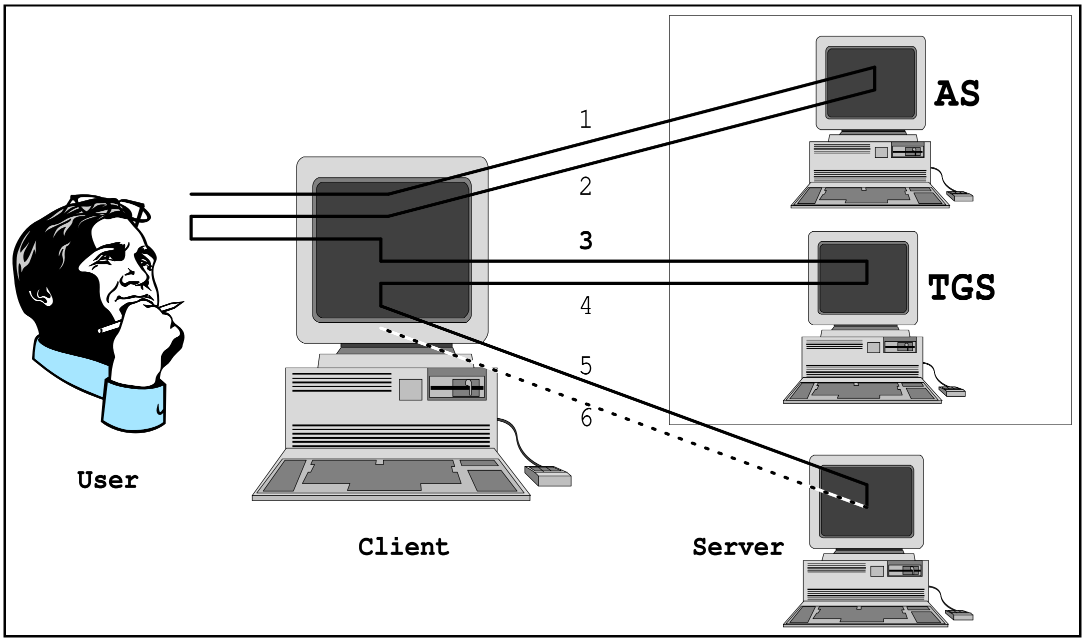

Figure 1 Kerberos protocol.

The Kerberos protocol is based on key distribution protocols that were originally proposed by Needham and Schroeder (Needham and Schroeder, 1978), and later modified to include timestamps (Denning and Sacco, 1981 and Needham and Schroeder, 1987). The protocol is illustrated in figure 1; it can be summarized as follows:

$$
\begin{array}{llll}
1: \mathrm{C} & \longrightarrow & \mathrm{AS} & : \mathrm{U,TGS} \\
2: \mathrm{AS} & \longrightarrow & \mathrm{C} & : T_{c,tgs}, \{TGS, K, T_{start}, T_{expire}\}K_u \\
3: \mathrm{C} & \longrightarrow & \mathrm{TGS} & : S, T_{c,tgs}, A_{c,tgs} \\
4: \mathrm{TGS} & \longrightarrow & \mathrm{C} & : T_{c,s}, \{S, K', T'_{start}, T'_{expire}\}K \\
5: \mathrm{C} & \longrightarrow & \mathrm{S} & : T_{c,s}, A_{c,s} \\
6: \mathrm{S} & \longrightarrow & \mathrm{C} & : \{T'\}K'
\end{array}
$$

The protocol basically consists of three exchanges: An AS exchange between the client C and the AS in steps 1 and 2, a TGS exchange between C and a TGS in steps 3 and 4, and a client/server exchange between C and a server S in steps 5 and 6.

In step 1, C sends the principal identifiers of U and a TGS to the AS. The AS generates a TGT $T_{c,tgs} = \{U, C, TGS, K, T_{start}, T_{expire}\}K_{tgs}$, and returns both $T_{c,tgs}$ and

$\{TGS, K, T_{start}, T_{expire}\}K_u$ to C in step 2. C asks U to enter this password, and if U enters it correctly, C can generate K_u and decrypt $\{TGS, K, T_{start}, T_{expire}\}K_u$ accordingly. C generates an authenticator $A_{c,tgs} = \{C, T\}K$, and forwards S, $T_{c,tgs}$ and $A_{c,tgs}$ to the TGS in step 3. The TGS decrypts $T_{c,tgs}$ with K_{tgs}, extracts K, decrypts $A_{c,tgs}$ with K, and checks the validity of the timestamp T. If both the ticket and the authenticator are valid, the TGS generates a ticket $T_{c,s} = \{U, C, S, K', T'_{start}, T'_{expire}\}K_s$, and returns $T_{c,s}$ and $\{S, K', T'_{start}, T'_{expire}\}K$ to C in step 4. C can decrypt the message with K and extract the new session key K'. He generates a new authenticator $A_{c,s} = \{C, T'\}K'$, and forwards it together with $T_{c,s}$ to S in step 5. S decrypts $T_{c,s}$ with K_s, extracts K', decrypts $A_{c,s}$, and checks the validity of T'. Again, if both the ticket and the authenticator are valid, S assumes C to authenticly acting on U's behalf. If mutual authentication were required, S would return $\{T'\}K'$ to C in step 6.

After step 5, C and S share K' as a session key . Equipped with this key, they can provide data authentication, confidentiality and integrity services. All current implementations of Kerberos use the Data Encryption Standard (DES) as secret key cryptosystem, and DES-CBC, MD4 and MD5 as one-way hash functions.

Bellovin and Merritt discussed environmental shortcomings and technical deficiencies of Kerberos V4 in (Bellovin and Merritt, 1990). The discussion has led to some modifications and changes in Kerberos V5. Two drawbacks haven't been addressed yet:

- The first drawback is related to the fact that in step 2 the AS returns a message that is encrypted with K_u. As K_u is derived from the user's password, an intruder who has been able to grab the message can start an off-line dictionary attack ("verifiable password"-attack). At least two solutions have been proposed to overcome the "verifiable password"-attack problem (Bellovin and Merritt, 1993 and Gong et. al., 1993).
- The second drawback is related to the fact that Kerberos is based on secret key cryptography, and that the AS must be trusted by all principals.

Work is currently going on at the MIT, to extend Kerberos for authentication over secondary networks (Atkins, 1993), and to use Kerberos to set up a PGP server in a corporate environment (Schiller and Atkins, 1995). Recently, a variant of Kerberos V5 was proposed as Yaksha (Ganesan, 1995). Yaksha uses as its building block a generalization of the RSA cryptosystem. The user's private key is split into two parts; one part is becoming the user's password, and the other part the AS' password for that particular user. Together, the user and the AS can digitally sign messages.

3 NETSP

Based on the fact that U.S. export restrictions primarily hold for cryptographic facilities that can be used for bulk data encryption, IBM has developed a family of authentication and key distribution protocols that use keyed one-way hash functions to generate message authentication codes (MAC). The protocols were first prototyped in a KryptoKnight security system (Molva, Tsudik and Van Herreweghen, 1992), and later used in IBM's Network Security Program (NetSP).

The core of the family is a 2-party authentication protocol (2PAP) that has been shown to be resistant against various kinds of attacks (Bird et. al., 1995):

$$
\begin{array}{lcl}
1: \mathrm{A} & \longrightarrow & \mathrm{B}: N_a \\
2: \mathrm{B} & \longrightarrow & \mathrm{A}: N_b, MAC_{ba}(N_a, N_b, B) \\
3: \mathrm{A} & \longrightarrow & \mathrm{B}: MAC_{ab}(N_a, N_b)
\end{array}
$$

In step 1, A challenges B with a nonce N_a, and in step 2, B responds with both another nonce N_b and $MAC_{ba}(N_a, N_b, B)$. The MAC is generated by applying a keyed one-way hash function on N_a, N_b and B. A can verify the MAC, and authenticate B accordingly. To authenticate himself to B, A generates $MAC_{ab}(N_a, N_b)$, and sends it to B in an additional step 3. Note that step 3 is only required for mutual authentication. Also note that the keys that are used to generate $MAC_{ba}(N_a, N_b, B)$ in step 2 and $MAC_{ab}(N_a, N_b)$ in step 3 needn't be the same. They are the same, if A and B use a secret key cryptosystem and share a session key. But 2PAP can also be used with a public key cryptosystem, and MAC_{ba} (MAC_{ab}) then implies a MAC that is generated by B (A), and that can be verified by A (B).

The following 2-party key distribution protocol (2PKDP) is derived from 2PAP by replacing B with a key distribution center (KDC):

$$
\begin{array}{lcll}
1: \mathrm{A} & \longrightarrow & \mathrm{KDC} & : N_a \\
2: \mathrm{KDC} & \longrightarrow & \mathrm{A} & : N_k, MAC_a(N_a, N_k, KDC) \oplus K'_a \\
3: \mathrm{A} & \longrightarrow & \mathrm{KDC} & : MAC_a(N_a, N_k)
\end{array}
$$

Step 1 is identical to 2PAP. In step 2, the KDC chooses a fresh nonce N_k and a new key for A, namely K'_a. N_k is returned in plaintext, whereas K'_a is added bitwise modulo 2 to $MAC_a(N_a, N_k, KDC)$. After receiving message 2, A generates the MAC, and extracts K'_a from $MAC_a(N_a, N_k, KDC) \oplus K'_a$ accordingly. If A were to confirm the receipt of the new key, he would return $MAC_a(N_a, N_k)$ in an additional step 3. Again, step 3 is optional. Note that after receiving N_k and $MAC_a(N_a, N_k, KDC) \oplus K'_a$ in step 2, A has no possibility to verify the authenticity of the message. The 2-party authenticated key distribution protocol (2PAKDP) has been designed to overcome this drawback:

$$
\begin{array}{lcll}
1: \mathrm{A} & \longrightarrow & \mathrm{KDC} & : N_a \\
2: \mathrm{KDC} & \longrightarrow & \mathrm{A} & : MAC(A), E_a(MAC(A)) \oplus N_k \\
3: \mathrm{A} & \longrightarrow & \mathrm{KDC} & : MAC_a(N_a, N_k)
\end{array}
$$

In this notation, $MAC(A)$ abbreviates $MAC_a(N_a, N_k, KDC)$, and N_k simultaneously represents K'_a. Note that only step 2 has changed; instead of replying with $N_k, MAC_a(N_a, N_k, KDC) \oplus K'_a$, the KDC returns $MAC(A), E_a(MAC(A)) \oplus N_k$. Upon receiving the message, A encrypts $MAC(A)$ and uses $E_a(MAC(A))$ to factor out the new key N_k from $E_a(MAC(A)) \oplus N_k$. A then computes $MAC_a(N_a, N_k, KDC)$ and checks, whether it matches $MAC(A)$ that he has been given by the KDC.

Several 3-party-key distributuion protocols (3PKDP) can be constructed by combining 2PAP and 2PKDP (or 2PAKDP). The following 3PKDP is a combination of two 2PAKDP (A-KDC and B-KDC) and one 2PAP (A-B):

$$
\begin{array}{llll}
1: \mathrm{A} & \longrightarrow & \mathrm{B} & : N_a \\
2: \mathrm{B} & \longrightarrow & \mathrm{KDC} & : N_a, N_b, A \\
3: \mathrm{KDC} & \longrightarrow & \mathrm{B} & : MAC(A), E_a(MAC(A)) \oplus K_{ab}, \\
: & & & : MAC(B), E_b(MAC(B)) \oplus K_{ab} \\
4: \mathrm{B} & \longrightarrow & \mathrm{A} & : MAC(A), E_a(MAC(A)) \oplus K_{ab}, \\
: & & & : N_b, MAC_{ab}(N_a, N_b, B) \\
5: \mathrm{A} & \longrightarrow & \mathrm{B} & : MAC_{ab}(N_a, N_b), MAC_a(N_a, K_{ab}) \\
6: \mathrm{B} & \longrightarrow & \mathrm{KDC} & : MAC_a(N_a, K_{ab}), MAC_b(N_b, K_{ab})
\end{array}
$$

Again, $MAC(A)$ ($MAC(B)$) abbreviates $MAC_a(N_a, K_{ab}, B)$ ($MAC_b(N_a, K_{ab}, A)$). After step 3, B is able to extract K_{ab} from $MAC(B), E_b(MAC(B)) \oplus K_{ab}$, and after step 4, A is able to do the same with $MAC(A), E_a(MAC(A)) \oplus K_{ab}$.

Obviously, 2PAKDP could be used for single sign-on (SSO), too. The user changes his weak password-derived long-term key to a strong session key. But because 2PAKDP is not providing authenticity in step 1, any intruder can start a protocol run to get information that might be useful for a dictionary-attack. The following SSO protocol provides preauthentication in step 1:

$$
\begin{array}{llll}
1: \mathrm{U} & \longrightarrow & \mathrm{KDC} & : N_u, T, MAC_u(N_u, T, U) \\
2: \mathrm{KDC} & \longrightarrow & \mathrm{U} & : MAC(U), E_u(MAC(U)) \oplus N_k
\end{array}
$$

$MAC(U)$ abbreviates $MAC_u(N_u, N_k, KDC)$, and N_k represents a session key for U, and T a timestamp. Note that in spite of its preauthentication, the protocol is still vulnerable to "verifiable-password" attacks. An attacker who is able to grab the message in step 1 knows N_u, T and U. He can try password candidates, until he finds a match with $MAC_u(N_u, T, U)$. The same mechanisms that could be used in Kerberos to protect against "verifiable-password" attacks could be used in this protocol, too. In environments with high-level security requirements, SSO in general is not a good idea, and one-time password systems should be used instead.

NetSP provides authentication and data integrity services. It can be enhanced to provide data confidentiality services, too. The confidentiality services are based on IBM's Commercial Data Masking Facility (CDMF). CDMF is a variant of DES with a restricted key length of 40 bit.

4 SPX

DEC has specified a Distributed Authentication Security Service (DASS) as part of its Distributed System Security Architecture (DSSA). The service has been prototyped in an authentication and key distribution system named SPX (Tardo and Alagappan, 1991).

SPX follows a hybrid approach. It uses secret key (DES) and public key cryptography (RSA). The overall architecture of SPX is strongly influenced by the ITU-T recommendation X.509. Certificate Distribution Centers (CDC) are to certify and distribute private and public keys, and Login Enrollment Agent Facilities (LEAF) are to authentication users as they sign-on. The SPX user initialization protocol is illustrated in figure 2; it can be described as follows:

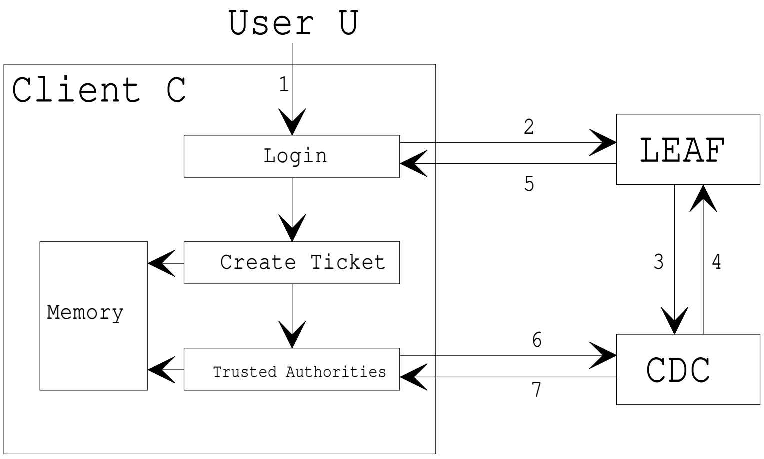

Figure 2 SPX user initialization protocol.

$$
\begin{array}{llll}
1: \mathrm{U} & \longrightarrow & \mathrm{C} & : \mathrm{U}, P' \\
2: \mathrm{C} & \longrightarrow & \mathrm{LEAF} & : \{U, T, N, h_1(P')\}k_{leaf} \\
3: \mathrm{LEAF} & \longrightarrow & \mathrm{CDC} & : \mathrm{U} \\
4: \mathrm{CDC} & \longrightarrow & \mathrm{LEAF} & : \{\{k_u^{-1}\}h_2(P), h_1(P), U\}K, \{K\}k_{leaf} \\
5: \mathrm{LEAF} & \longrightarrow & \mathrm{C} & : \{\{k_u^{-1}\}h_2(P), U\}N \\
6: \mathrm{C} & \longrightarrow & \mathrm{CDC} & : \mathrm{U} \\
7: \mathrm{CDC} & \longrightarrow & \mathrm{C} & : \{U, TA_u, L, k_{TA_u}\}_{k_u^{-1}}
\end{array}
$$

In step 1, U signs-on a client C by providing his username and password P'. C uses a one-way hash function h_1 to hash P'. He encrypts U, a timestamp T, a nonce N and $h_1(P')$ with k_{leaf}, and sends the result to the LEAF in step 2. In step 3, the LEAF contacts the CDC, and in step 4, the CDC returns $\{\{k_u^{-1}\}h_2(P), h_1(P), U\}K$ and $\{K\}k_{leaf}$. With its private key, the LEAF can decrypt $\{K\}K_{leaf}$ and extract K accordingly. It can use this key to decrypt $\{\{k_u^{-1}\}h_2(P), h_1(P), U\}K$, and to extract $\{k_u^{-1}\}h_2(P)$, $h_1(P)$ and U accordingly. If $h_1(P)$ matches $h_1(P')$, the LEAF assumes C to be authenticly acting on U's behalf. In step 5, the LEAF provides C with both $\{k_u^{-1}\}h_2(P)$ and U. The whole message is encrypted with N. C knows N and can decrypt $\{\{k_u^{-1}\}h_2(P), U\}N$ accordingly. $\{k_u^{-1}\}h_2(P)$ can then be decrypted by applying h_2 to P', and using the result as a decryption key. In addition to that, C needs the public key k_{TA_u} of a trusted authority for U (TAu). He therefore contacts the CDC in step 6, and in step 7, the CDC returns a message that is encrypted with the private key of U. C can verify the signature, and extract k_{TA_u} accordingly.

Figure 3 illustrates the SPX authentication protocol; it can be described as follows:

$$
\begin{array}{llll}
1: \mathrm{C} & \longrightarrow & \mathrm{CDC} & : S \\
2: \mathrm{CDC} & \longrightarrow & \mathrm{C} & : \{TA_u, S, L_1, k_s\}k_{TA_u}^{-1} \\
3: \mathrm{C} & \longrightarrow & \mathrm{S} & : \{T\}K, \{U, L_2, k_d\}k_u^{-1}, \{K\}k_s, \{k_d^{-1}\}K \\
4: \mathrm{S} & \longrightarrow & \mathrm{CDC} & : U \\
5: \mathrm{CDC} & \longrightarrow & \mathrm{S} & : \{TA_s, U, L_3, k_u\}k_{TA_s}^{-1} \\
6: \mathrm{S} & \longrightarrow & \mathrm{C} & : \{T+1\}K
\end{array}
$$

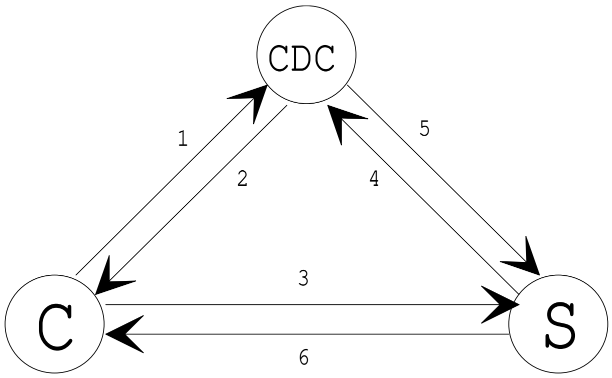

Figure 3 SPX authentication protocol.

In step 1, C contacts the CDC to get the public key k_s of S. In step 2, the CDC returns a certificate for k_s. The certificate is encrypted with TAu's private key, and can be decrypted with k_{TA_u}. The lifetime of the certificate is restricted with L_1. C now generates a session key K and a RSA delegation key pair (k_d, k_d^{-1}). In step 3, he sends $\{T\}K$, $\{U, L_2, k_d\}k_u^{-1}$, $\{K\}k_s$ and $\{k_d^{-1}\}K$ to S. S can decrypt $\{K\}k_s$, and use K to decrypt both $\{k_d^{-1}\}K$ and $\{T\}K$. For the decryption of $\{U, L_2, k_d\}k_u^{-1}$, S needs the public key of U. In step 4, S contacts the CDC, and in step 5, the CDC returns a certificate for k_u. This certificate is encrypted with $k_{TA_s}^{-1}$, the private key of the trusted authority for S (TAs). S is assumed to be provided with this key, and he can decrypt $\{TA_s, U, L_3, k_u\}k_{TA_s}^{-1}$ accordingly. Equiped with k_u, S can decrypt $\{U, L_2, K_d\}K_u^{-1}$ and extract k_d. For mutual authentication, S has to send $\{T+1\}K$ to C in an additional step 6.

SPX has historically been the first authentication and key distribution system to follow a hybrid approach. Unfortunately, DEC has canceled the project before SPX was turned into a commercial product. Today, SPX is publicly available. Recently, a simplified version of SPX has been specified as Simple Public-Key GSS-API Mechanisms (SPKM).

5 TESS

The Exponential Security System (TESS) has been developed at the European Institute for System Security of the University of Karlsruhe (Beth and Gollmann, 1994). TESS includes both an authentication and key distribution system named SELANE (Secure Local Area Network Environment), and a digital signature system named EES (Exponential Electronic Signature). The core of TESS is a KATHY (key exchange with embedded authentication) protocol family (Bauspiess and Knobloch, 1989). A SKIA (Secure Key Issuing Authority) registers users, and generates, certifies and distributes personal tokens (smartcards or PCMCIA cards).

During the SKIA initialization phase, a large prime number p, a generator $\alpha \in GF(p)$, and a private key $X \in GF(p)$ are chosen. The SKIA's public key is calculated as $Y = \alpha^X \bmod p$. The system parameters p, α and Y are made public, whereas X is kept secret. To register user U_i, the SKIA generates an identity string m_i and codes it in $GF(p)$. The

SKIA then chooses a $k_i \in_R GF^*(p)$, calculates $r_i = \alpha^{k_i} \bmod p$ and solves $Xr_i + s_i k_i \equiv m_i (\bmod p - 1)$ for s_i. So (r_i, s_i) represents an ElGamal digital signature for m_i. Every user U_i is equipped with a personal token that holds m_i, r_i and s_i. A and B can use the following protocol to do an authenticated key exchange:

$$\begin{array}{lcl} 1: \mathrm{A} & \longrightarrow & \mathrm{B}: m_A, r_A \\ 2: \mathrm{B} & \longrightarrow & \mathrm{A}: v_B = r_A^{z_B} \bmod p \end{array}$$

In step 1, A provides B with m_A and r_A. B then chooses a $z_B \in_R GF^*(p)$, and returns $v_B = r_A^{z_B} \bmod p$ to A in step 2. If A calculates $v_B^{s_A} \bmod p$, and B calculates $(\alpha^{m_A} Y^{-r_A})^{z_B} \bmod p$, they come out with the same session key. If A and B are to play an equal part in the key generation process, the protocol has to be run a second time (in reverse order):

$$\begin{array}{lcl} 1: \mathrm{B} & \longrightarrow & \mathrm{A}: m_B, r_B \\ 2: \mathrm{A} & \longrightarrow & \mathrm{B}: v_A = r_B^{z_A} \bmod p \end{array}$$

Again, A and B come out with the same session key. They can use any publicly known function to hash the two session keys to one.

If the SKIA stores all the values that it generates during the user registration phase, it is able to reconstruct all session keys that are subsequently exchanged. If the users don't trust the SKIA, they may want to have their identity strings certified by the SKIA by using covert signatures.

6 SESAME

Based on the OSI security architecture, the European Computer Manufacturer Association (ECMA) has developed a security framework for the application layer. SESAME (A Secure European System for Applications in a Multi-vendor Environment) is a project that is partly sponsored by the European Union (EU). The aim of the project is to implement and further develop the ECMA security framework (McMahon, 1995). The implementation work is mainly done by Bull SA, International Computers Limited (ICL) and Siemens Nixdorf Informationssysteme (SNI) AG.

SESAME V1 implemented the ECMA authentication facility. With regard to the SESAME V2 authentication service, a migration to Kerberos V5 was decided in 1991. Since January 1994, SESAME V2 is available for Beta testers. The SESAME V2 toolkit is supposed to be released later this year.

The overall architecture of SESAME V2 is illustrated in figure 4. The user contacts his User Sponsor. Together with the Authentication and Privilege Attribute (APA) Client, the User Sponsor contacts the Kerberos AS in order to get a TGT. Having received a TGT, it is up to the Secure Association Context Manager (SACM) to store the ticket, and to use it to get a Privilege Attribute Certificate (PAC) from a Privilege Attribute Server (PAS). PACs are digitally signed and their valid lifetimes are restricted, too. Together with the TGT, the PAC can be used to get a ticket from a Key Distribution Server (KDS). A SACM is running on the server's side, too. A PAC Validation Facility (PVF) is to verify PACs and tickets. The PVF therefore interacts with the KDS.

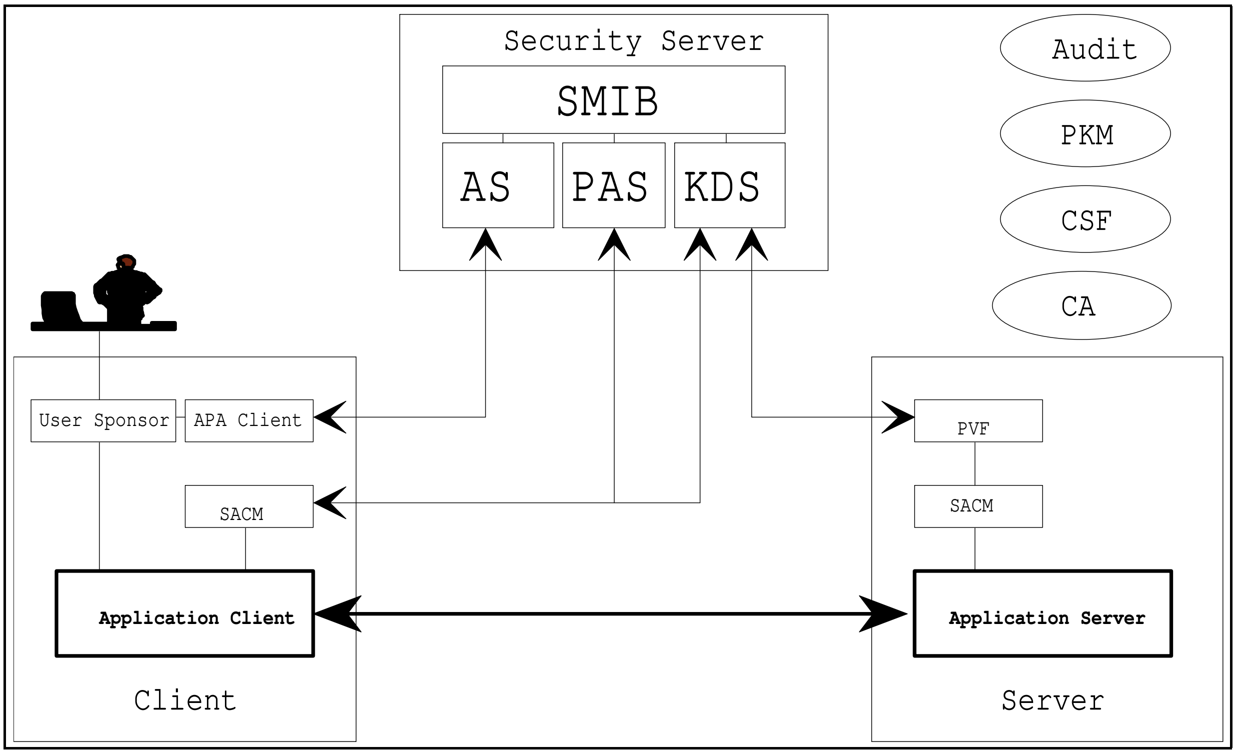

Figure 4 Architecture of SESAME.

A Certification Authority (CA), a Cryptographic Support Facility (CSF), a Public Key Management (PKM), and a system audit are providing supporting functionalities in the background .

7 CONCLUSIONS

Authentication and key distribution systems are used in computer networks and distributed systems to provide security services at the application layer. The systems that are outlined in this paper differ in various respects:

Security services: All systems provide authentication, data confidentiality and integrity services. (Note that NetSP provides data confidentiality services only with a restricted key length of 40 bit.) In addition to that, TESS provides non-repudiation and SESAME access control services.

Cryptographic techniques: NetSP uses keyed one-way hash functions only, and Kerberos uses secret key cryptography. All other systems follow a hybrid approach, and use public key cryptography, too.

Conformance to standards: Kerberos and SPX are both specified as Internet Request for Comments (RFC). SPX and SESAME are compliant with ITU-T recommendation X.509, and SESAME is compliant with the ECMA security framework, too. Except

TESS, all systems provide security services at an application programming interface (API) that is compliant to the Generic Security Services API (GSS-API). The GSS-API has been specified by the Internet Engineering Task Force (IETF) Common Authentication Technology (CAT) Working Group (WG) in RFC 1508 and 1509.

Availability: Kerberos, SPX and TESS are publicly available, whereas NetSP and SESAME are distributed by vendors.

Exportability: The issue of exportability is relevant only for U.S. American systems: The export of Kerberos and SPX is restricted, whereas NetSP is exportable (due to its restricted key length).

The Open Software Foundation (OSF) has recognized the importance of security in distributed computing. As a matter of fact, it has adapted Kerberos V5 as authentication service for the Distributed Computing Environment (DCE) version 1.x (with some extensions to access control mainly taken from SESAME V2). It is assumed that OSF DCE will migrate to a hybrid approach in a later version.

ACKNOWLEDGEMENTS

The author would like to thank Prof. Dr. Dieter Hogrefe, Dr. Andreas Greulich and Arnold Grossmann from the University of Berne, Daniel Bleichenbacher from the ETH Zürich, and Peter Trachsel and Marcel Frauenknecht from the Swiss Federal Office of Information Technology and Systems (BFI) for their encouragement and support.

REFERENCES

Atkins, D. (1993) Charon: Kerberos Extensions For Authentication Over Secondary Networks. Massachusetts Institute of Technology (MIT), Cambridge, MA.

Bauspiess, F., and Knobloch, H.J. (1990) How to Keep Authenticity Alive in a Computer Network. In *Proceedings of EUROCRYPT '89*, 38-46.

Bellovin, S.M., and Merritt, M. (1990) Limitations of the Kerberos Authentication System. *ACM Computer Communication Review 20(5)*, 119-32.

Bellovin, S.M., and Merritt, M. (1993) Augmented Encrypted Key Exchange. In *Proceedings of the 1st ACM Conference on Communications and Computing Security.*

Beth, Th., and Gollmann, D. (1994) Security Systems Based on Exponentiation Primitives: TESS — The Exponential Security System. In *Proceedings of IFIP SEC'94.*

Bird, R., Gopal, I., Herzberg, A., Janson, P.A., Kutten, S., Molva, R., and Yung, M. (1995) The KryptoKnight Family of Light-Weight Protocols for Authentication and Key Distribution. *IEEE/ACM Transactions on Networking 3(1)*, 31-41.

Denning, D.E., and Sacco, G. (1981) Timestamps in Key Distribution Protocols. *Communications of the ACM 24(8)*, 533-6.

Ganesan, R. (1995) Yaksha: Augmenting Kerberos with Public Key Cryptography. In *Proceedings of the Internet Society Symposium on Network and Distributed System Security*, 132-43.

Gong, L., Lomas, M., Needham, R.M., and Saltzer, J. (1993) Protecting Poorly Chosen Secrets from Guessing Attacks. *IEEE Journal on Selected Areas in Communications 11(5)*, 648-56.

Kohl, J., and Neuman, B.C. (1993) The Kerberos Network Authentication Service (V5). Request for Comments 1510.

McMahon, P. (1995) SESAME V2 Public Key and Authorisation Extensions to Kerberos. In *Proceedings of the Internet Society Symposium on Network and Distributed System Security*, 114-31.

Molva, R., Tsudik, G., Van Herreweghen, E., and Zatti, S. (1992) KryptoKnight Authentication and Key Distribution System. In *Proceedings of ESORICS '92*, 155-74.

Needham, R.M, and Schroeder, M.D. (1978) Using Encryption for Authentication in Large Networks of Computers. *Communications of the ACM 21(12)*, 993-9.

Needham, R.M., and Schroeder, M.D. (1987) Authentication Revisited. *ACM Operating Systems Review 21(1)*, 7.

Oppliger, R. (1992) *Computersicherheit — Eine Einführung*. Vieweg-Verlag, Wiesbaden.

Schiller, J.I. (1994) Secure Distributed Computing. *Scientific American* November, 72-6.

Schiller, J.I., and Atkins, D. (1995) Scaling the Web of Trust: Combining Kerberos and PGP to Provide Large Scale Authentication. In *Proceedings of the Technical Conference on UNIX and Advanced Computing Systems*, 83-94.

Schneier, B. (1994) *Applied Cryptography: Protocols, Algorithms, and Source Code in C*. John Wiley & Sons, Inc., New York.

Tardo, J., and Alagappan, K. (1991) SPX: Global Authentification Using Public Key Certificates. In *Proceedings of the IEEE Symposium on Security and Privacy*, 232-44.

BIOGRAPHY

Rolf Oppliger studied computer science, mathematics, and economics at the University of Berne (Switzerland), where he received his Ph.D. from the Institute for Computer Science and Applied Mathematics (IAM) in June 1993. His interests are directed towards cryptography, and the use of cryptographic techniques in computer networks and distributed systems. He just returned from a post-doctoral stay at the International Computer Science Institute (ICSI) in Berkeley, where he mainly focused on the security requirements of multi-party communication and conferencing systems. He's a current member of IFIP TC 11/WG 4.