AIIM

Association for Information and Image Management

1100 Wayne Avenue, Suite 1100
Silver Spring, Maryland 20910

301/587-8202

Centimeter

Inches

MANUFACTURED TO AIIM STANDARDS
BY APPLIED IMAGE, INC.

1 of 1

SAND 94-1552 C
Conf-9410105--4

Key Management for Large Scale End-to-End Encryption

Edward L. Witzke
RE/SPEC Inc.
4775 Indian School Road N.E., Suite 300
Albuquerque, New Mexico 87110
elwitzk@respec.com
(505)268-2661

Lyndon G. Pierson
Sandia National Laboratories
P.O. Box 5800
Albuquerque, New Mexico 87185-0807
lgpiers@sandia.gov
(505)845-8212

Abstract

Symmetric end-to-end encryption requires separate keys for each pair of communicating confidants. This is a problem of Order N^2. Other factors, such as multiple sessions per pair of confidants and multiple encryption points in the ISO Reference Model complicate key management by linear factors. Public-key encryption can reduce the number of keys managed to a linear problem, which is good for scaleability of key management, but comes with complicating issues and performance penalties.

Authenticity is the primary ingredient of key management. If each potential pair of communicating confidants can authenticate data from each other, then any number of public encryption keys of any type can be communicated with requisite integrity. These public encryption keys can be used with the corresponding private keys to exchange symmetric cryptovariables for high data rate privacy protection.

The Digital Signature Standard (DSS), which has been adopted by the United States Government, has both public and private components, similar to a public-key cryptosystem. The Digital Signature Algorithm of the DSS is intended for authenticity but not for secrecy.

In this paper, the authors will show how the use of the Digital Signature Algorithm combined with both symmetric and asymmetric (public-key) encryption techniques can provide a practical solution to key management scaleability problems, by reducing the key management complexity to a problem of order N, without sacrificing the encryption speed necessary to operate in high performance networks.

Keywords: Key management, End-to-end encryption, Scaleability, Public-key encryption, Asymmetric encryption, Two-key encryption, Exponential key exchange, Digital Signature Standard, Digital Signature Algorithm, Hybrid encryption systems, DSS, DSA, P-KE

Scope and Definitions

End-to-end encryption enciphers specific sessions between confidants on the source and destination nodes only [3]. Key management, for this paper, shall be limited to registration and distribution of keys to enable pairwise confidential communications for the number of confidants included in the system. This will be discussed in more detail later in this paper.

Key Management Complexity

Key management, in general, can be a difficult problem because of issues as:

- Distribution of keys;
- Distribution of lists containing revoked keys ("hot lists");
- Tracking which keys were valid during what period of time.

The items to be distributed must be communicated to the recipients in a tamper-proof manner. The complexity factor per confidant (G), which is due to this general nature of key management, can vary with implementation. To simplify analysis, assume that this complexity per confidant does not increase significantly as the number of confidants increase (as in a single database).

End-to-end session encryption requires separate keys for confidentiality between each pair of communicating confidants [3]. This is a combinatoric counting problem of N things taken 2 at a time. The general formula to compute the number of combinations of N things taken R at a time is $\binom{N}{R} = \frac{N!}{R!(N-R)!}$.

Specifically, $\binom{N}{2} = \frac{N!}{2!(N-2)!} = \frac{N^2 - N}{2}$, is the number of keys required for pairwise confidential communication between N confidants. (This is the same point made by Diffie and Hellman [2]. They strived to reduce this complexity by using public-key encryption, but at the time did not have a practical method to ensure authenticity of public keys.)
Each and any pair of communicating confidants can have multiple sessions (file transfer, virtual terminal, interprocess communication, etc.) proceeding simultaneously. Let S be the average number of simultaneous sessions requiring a key set, per pair of communicating confidants.

If end-to-end encryption is present at several layers of the ISO model (see Tanenbaum [6] for a description of the ISO Reference Model), each instance of end-to-end encryption *may* require a separate set of keys. Let K be the number of different key sets required per session, for a given cryptosystem due to encryption at multiple communication protocol layers.

The total number of key sets required is $\frac{SK(N^2 - N)}{2}$. In order to manage a predetermined key pair for each potential session, the overall complexity of the key management problem with respect to the number of confidants is C, where $C = \frac{GSK(N^2 - N)}{2}$.

G, S, and K are constants, not dependant on N, in the above formula. The key management complexity, C, is a problem of Order N^2.

The key management technique discussed in the balance of this paper, *as limited by the definition of key management earlier in this paper*, will convert the key management complexity problem to a problem of Order N.

Technique

What is presented in this paper is a hybrid encryption system. This in itself is not new [5]. Hybrid encryption systems have been proposed in the past as a way to speed up encryption while still taking advantage of the convenience offered by public-key cryptosystems. The authors recognize the value of this, but also wish to point out how public-key authentication can be used to control the key management complexity problems associated with classical cryptosystems. The technique presented here uses keys for the proposed Digital Signature Standard (DSS) [1], which does not provide secrecy, with public-key encryption to exchange a session key for a symmetric encryption algorithm. This technique also offers the flexibility to insert the encryption algorithms appropriate to the specific circumstances, based on application requirements and various cryptosystem design trade-offs. For example, since symmetric encryption algorithms typically have higher encrypted data throughput rates than public key algorithms, the session key for a fast symmetric encryption algorithm can be exchanged, in order to satisfy throughput performance requirements of today's high speed computer communication networks.

Once a method for assuring authenticity such as the Digital Signature Standard is in place, the type of public-key encryption (RSA, Diffie-Hellman, etc.) and if necessary, a public key (possibly generated specifically for this session) can be signed and transmitted. Next a symmetric, session encryption algorithm can be selected and a session key can be generated and encrypted for transport using the specified public-key encryption algorithm and key. Once the session algorithm and session key is known to both confidants, the selected symmetric encryption algorithm can be used to rapidly encrypt/decrypt the message traffic for this session. A specific implementation in which the Digital Signature Standard is used in conjunction with Diffie-Hellman public-key encryption to exchange a session key to a symmetric encryption algorithm will now be described.

Each confidant participating in this system must generate a pair of keys (one private, one public) suitable for use with the Digital Signature Standard. The public key, along with information to identify the confidant, shall be registered with a central authentication entity. The central authentication entity will also have a pair of DSS keys. The public key of the central authentication entity should be published, well known, and easily accessable. This deters spoofing as this main, public key can be verified through numerous independent means.

For each session, each member of a communicating pair (Alice and Bob) will generate a key-exchange key in the manner of Diffie-Hellman. Each confidant would request a certified copy of his partner's public DSS key. Each confidant validates his partner's public DSS key using the DSS certificate and the well-known, public DSS key of the certifying entity. Now that each of the communicating confidants has an authentic copy of the other's public DSS key, he can raise it to

the power of his own private DSS key, mod p, where p is one of the common and public parameters of the Digital Signature Algorithm. This works because the public component of the DSS key (y) is g^x mod p, where g is also a parameter of the Digital Signature Algorithm and x is the private component of the DSS key. It should now be obvious that the key-exchange key,

$$K = (g^{x_B} \bmod p)^{x_A} \bmod p = (g^{x_A} \bmod p)^{x_B} \bmod p,$$

for confidants Alice (A) and Bob (B) can be possesed by each.

Having computed a key-exchange key K, one confidant now selects or calculates a session key. This session key would be signed by Alice with her DSS key and then encrypted using K as a key to whatever symmetric encryption algorithm (DES, one time pad, etc.) has been selected for use in this system. When Bob receives the message containing the session key, he will first decrypt it using K and the symmetric algorithm, then he will validate the signature using Alice's public DSS component. After validation, Bob signs a message acknowledging receipt of the session key, and sends this to Alice. This acknowledgement does not necessarily need to be encrypted.

With the session key successfully exchanged (without tampering) and authenticated, private communication between Alice and Bob can now take place. The sending confidant can sign (using DSS) a message to assure authenticity. Next, he could encrypt the signed message using the exchanged session key, to achieve privacy. Now the message can be transmitted over an insecure communication channel, as shown in figure 1.

The receiving confidant decrypts the incoming message with the session key to "remove the privacy envelope." He then validates the digital signature of the message with the sending confidant's public DSS key (exchanged at the start of the session) to verify authenticity. At the end of the session, all session keys can be destroyed, unless the receiving confidant wishes to be able to verify the authenticity of a received document or message at a later time. By the receiver retaining the sending confidant's DSS key from this session (as obtained from the central authentication entity) along with the decrypted message, the receiver can verify (or re-verify) the signature at any time without having to determine which signature key was valid at the time of the original transmission.

Conclusions

Each confidant in the system would only have to register one key with the central authentication entity. That entity would manage *N* keys, where *N* is the number of confidants in the system. *S* and *K* will have the same values as stated earlier to represent the average number of sessions and the number of key sets per session. The factor due to general key management complexity (*G*) would not increase by adding the described encryption technique to a system already providing digital signature service. "Hot lists" related to encryption are no longer necessary, since encryption keys are used only for the current session. Likewise, it is not necessary to track which encryption keys were valid during what period of time, since any keys generated were used specifically for this session. It is only necessary to track the validity (history) of the DSS keys if they are used to provide authenticity for the data message traffic of this session. By definition the central authentication entity should always have the current DSS public component for each registered confidant. The only public key or DSS public component that needs to be checked to

ensure it is current, is the public component of the central authentication entity's DSS key. As stated earlier, this should be published, well known, and easily accessable. When a public key authentication system is used to exchange session variables, the overall complexity of the key management problem with respect to the number of confidants is C, where now $C = GSKN$, since only N public authentication variables must be maintained.

With G, S, and K independent of N in the above formula, and the complexity due to the number of confidants being reduced to N (regardless how many different pairs of them are communicating at any time), the overall complexity of the key management problem is linear. This will lend itself to scaling for large, high speed, communication networks because the key management complexity is not a function of the speed of the communications network and because the overhead attributable to the number of participants can be kept linear rather than exponential.

The Digital Signature Standard is (or will soon be) a national standard that provides an authentication mechanism which can be used for the exchange of cryptographic variables for public-key cryptosystems. Since the Digital Signature Algorithm does not provide secrecy, but may already be in use by a system for signature purposes, we recommend (although this method does not require) using the DSS keys in a Diffie-Hellman public-key encryption algorithm to encrypt the exchange of session keys used for classical encryption. This also avoids the possibly of maintaining separate key registries for signature and encryption keys. Now, by combining the Digital Signature Standard, public-key encryption (for key exchange), and classical session encryption, the means are available for scalable, high speed encryption and digital signature services between confidants who have never before communicated with each other. The general technique described in this paper provides for 1) flexibility regarding the choice of public-key encryption method for exchanging session key information, 2) allows dynamic generation and exchange of public-key cryptographic variables for session key exchange, 3) flexibility regarding the choice of session encryption method, and 4) dynamic generation and exchange of session keys, all with order N key management complexity.

Acknowledgements

The work described in this paper was performed by Sandia National Laboratories and RE/SPEC Inc. under RE/SPEC contract number 56-4484 to Sandia National Laboratories and Sandia contract number DE-AC04-94AL85000 to the United States Department of Energy. The authors would like to thank Ernest Brickell and Kevin McCurley, both of Sandia National Laboratories, for their contributions to this work.

Bibliography

1. "A Proposed Federal Information Processing Standard for Digital Signature Standard (DSS)," Federal Register, Vol. 56, No. 169, August 31, 1991, pp. 42980-42982.

2. Diffie, Whitfield, and Martin E. Hellman, "New Directions in Cryptography," IEEE Transactions on Information Theory, Vol. IT-22, No. 6, November 1976.

3. Pierson, Lyndon G., and Edward L. Witzke, "Data Encryption and the ISO Model for Open Systems Interconnection," ISE '84 Joint Proceedings, May 1984.

4. Rivest, R. L., et al., "A Method for Obtaining Digital Signatures and Public Key Cryptosystems," Communications of the ACM, Vol. 21, No. 2, February 1978.

5. Schneir, Bruce, Applied Cryptography, John Wiley & Sons, New York, 1994.

6. Tanenbaum, Andrew S., Computer Networks, Prentice-Hall, Englewood Cliffs, New Jersey, 1981.

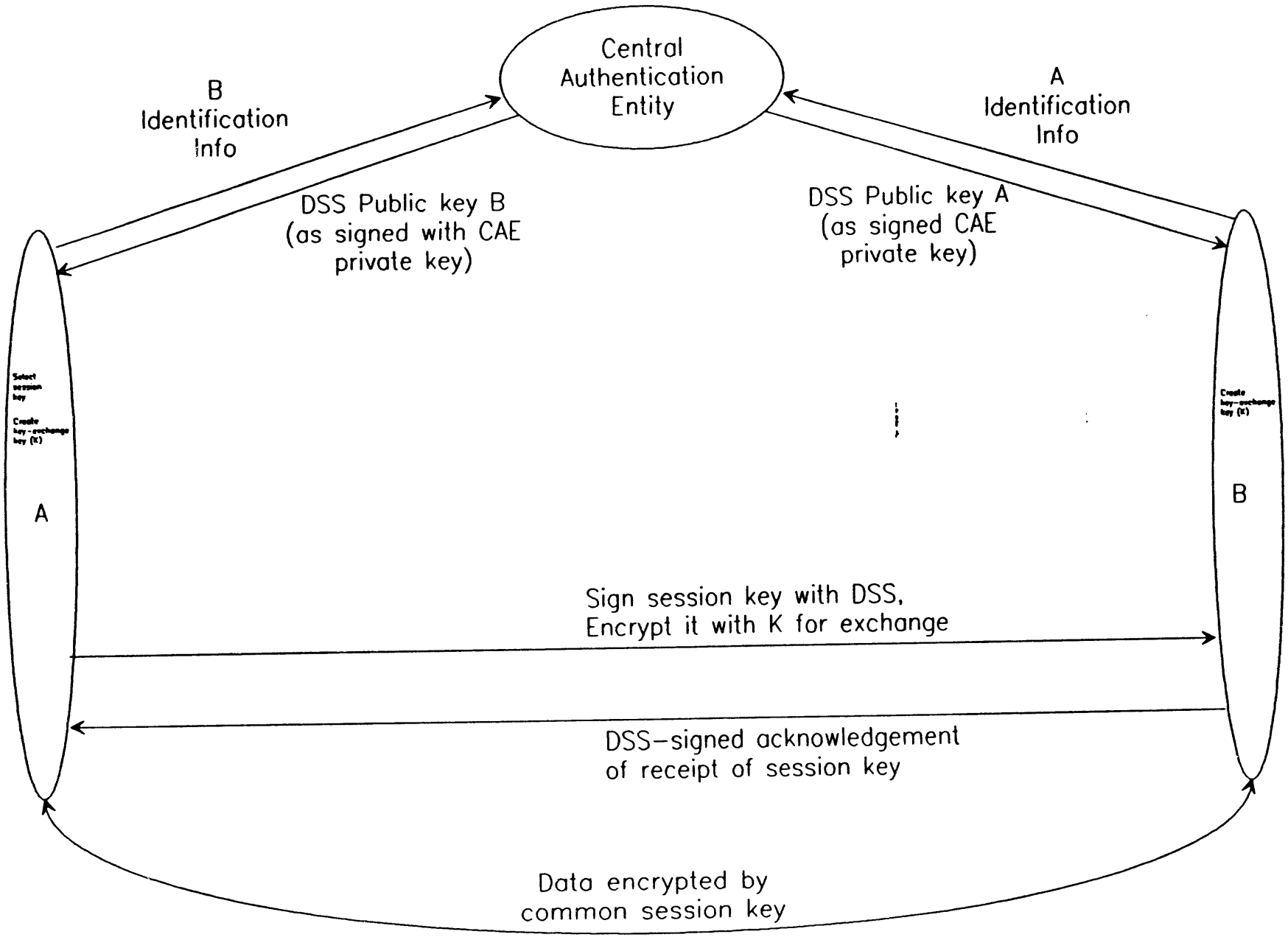

Figure 1. Key Management for Scalable End-to-End Encryption

DATE
FILMED
9/7/94
END