

How to Keep Authenticity Alive in a Computer Network

Fritz Bauspieß

Hans-Joachim Knobloch

Universität Karlsruhe
Europäisches Institut für Systemsicherheit
European Institute for System Security

Kaiserstr. 8
D-7500 Karlsruhe 1

Abstract

In this paper we present a cryptographic scheme that allows to ensure the ongoing authenticity and security of connections in a computer network. This is achieved by combining a zero-knowledge authentication and a public key exchange protocol. It is noteworthy that due to the combination both protocols gain additional security against attacks that would otherwise be successful. The scheme is applicable to both local area networks and internetworks.

1 Introduction

In recent computer networks there are two developments demanding for the use of new methods of user authentication. On one hand advanced workstations largely increased the number of machines in a network and internetworking LAN-LAN connections or remote access telecommunication lines increased the connectivity of these machines, thus making the network much more vulnerable against unauthorized manipulation. On the other hand through the widespread use of computers more and more people gain the knowledge, the wish and the possibility to carry out such en-vogue manipulations like *hacking* or *programming computer viruses*.

A network administrator wants methods to prevent these manipulations or at least to be able to sue an individual user for damages if manipulations occurred, thus deterring other possible ab-users of his network. Likewise network users want methods to prevent them from being accused of manipulations other people made pretending their authentication.

Even if encryption is used to protect an interactive session there is a fundamental problem whenever key exchange for the session encryption and authentication of the session are two separate protocols.

The possible intruder who is capable of suppressing and forging messages will let the authentication data pass unchanged. But he will try to intercept a session key instead, so that his requests encrypted under this key appear authentic.

If e.g. the Diffie-Hellman key exchange protocol [3] were used, the intruder may perform the well-known switching-in attack resulting in two keys, one between user and intruder and the other between intruder and host. The key shared with the user might be used to ask him for authentication data as the pretended host. The same attack would also work against the key exchange using function composition [5].

So it appears to be essential that key exchange and authentication are inseparably combined.

2 Authentication by key exchange

Suppose all users are known to the host before they attempt to establish a session. Then secret information common between the user and the host can be used to generate a session key. The user is under this approach authenticated by using the correct key and thereby showing his knowledge of the secret information he was given in advance.

For example the shared secret information may be a master key to encrypt and decrypt the session key generated by the host. There are numerous slight modifications of this scheme.

Another possibility is to use a public key exchange protocol where an essential part of information is not transmitted publicly but kept as common secret. Using the Diffie-Hellman protocol the steps could be:

- Host B chooses a prime power q and a primitive element ω of $\text{GF}(q)$
- Upon registration user A chooses his secret $a \in \mathbf{Z}_{q-1}$ and passes $\omega^a \in \text{GF}(q)$ to B (along with his identification)
- Every time a session is established, B generates a random number $s \in \mathbf{Z}_{q-1}$ and sends $\omega^s \in \text{GF}(q)$ to A. Only A and B are then capable to compute the session key ω^{as}

One major disadvantage of this protocol is that B has to keep a datafile of all the possible users A_i and their ω^{a_i} . This will be a large and often changing file. Moreover if ω^{a_i} is used to ensure the user that he is connected to the correct host (as proposed in SELANE [1]) this file contains sensitive data and thus is an additional breakpoint into the system.

SELANE therefore proposes a CSC (Central Security Controller) to hold these data. Updating and securing this file is then made easy. In addition this CSC can be used to note all connections made in the network (for later reconstruction) - no one can bypass it!

3 Key exchange by authentication

An alternative possibility to combine authentication and key exchange is to use data exchanged during the authentication protocol, and therefore authentic data, to construct the session key. Under this approach the use of a zero-knowledge authentication protocol provides additional advantages, since then the host does not have to know all possible users but only the secure key issuing authority (SKIA). All

sensitive data is either offline within the SKIA or protected by the zero-knowledge scheme. Also concerning internetworking this solution has advantages over the one mentioned before, because there are not many SKIAs within a whole internetwork nor is there a large fluctuation of their public data as compared to user data.

A zero-knowledge identification scheme that is very well suited for our purposes is the one Beth introduced at the Eurocrypt '88 [2] which we will recall here for the reader's convenience.

The scheme consists of three phases. In the first phase the SKIA chooses some constants that are common to all participants of the scheme. In the next phase the SKIA computes data that serve as the individual user's credentials and issues them in a secure token device. The last phase is the authentication itself.

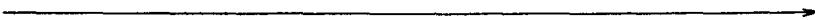
Initiation:

SKIA chooses a finite field $\text{GF}(q)$ with primitive element ω ,
 random $x_1, \dots, x_m \in \mathbf{Z}_{q-1}$ and
 a oneway function $f : \mathbf{Z} \times \mathbf{Z}_m \rightarrow \mathbf{Z}_{q-1}$
 computes $y_j := \omega^{x_j}$ in $\text{GF}(q)$ for $j = 1, \dots, m$
 q, w, y_1, \dots, y_m and f are published, x_1, \dots, x_m are kept secret

Registration of user A:

SKIA checks A and gives her a $\text{name}_A \in \mathbf{Z}$
 computes $I_{A,j} := f(\text{name}_A, j)$ for $j = 1, \dots, m$
 chooses a random $k_A \in \mathbf{Z}_{q-1}$ and computes $r_A := \omega^{k_A}$ in $\text{GF}(q)$
 determines solutions $s_{A,j}$ of $x_j r_A + k_A s_{A,j} \equiv I_{A,j} \pmod{(q-1)}$ for $j = 1, \dots, m$
 issues a token device to A containing $\text{name}_A, r_A, s_{A,1}, \dots, s_{A,m}$

Authentication of user A versus host B:

A		B
	name_A, r_A	
		
		for $j = 1, \dots, m$ computes $I_{A,j} = f(\text{name}_A, j)$

for $i = 1, \dots, h$ do

chooses random $t_{A,i} \in \mathbf{Z}_{q-1}$

computes $z_{A,i} := r_A^{-t_{A,i}}$ in $\text{GF}(q)$

$z_{A,i}$

chooses random

$(b_{A,1,i}, \dots, b_{A,m,i}) \in \mathbf{Z}_{q-1}^m$

$(b_{A,j})_i$

computes

$u_{A,i} := t_{A,i} + \sum_j b_{A,j,i} s_{A,j} \pmod{(q-1)}$

$u_{A,i}$

computes (in $\text{GF}(q)$)

$\gamma_{A,i} := \left(\prod_j y^{r_A b_{A,j,i}} \right) r_A^{u_{A,i}} z_{A,i} - \omega^{\sum_j b_{A,j,i} I_{A,j}}$

od

accepts the authentication if

$\gamma_{A,i} = 0$ for all $i = 1, \dots, h$

Now consider the case $m = 1$ and $h = 1$.

Observation: There is the value $z_A = r^{-t_A}$ the form of which resembles the public parts of the key in the Diffie-Hellman protocol.

Idea: Use z_A as part of the Diffie-Hellman key. B chooses a random $d_B \in \mathbf{Z}_{q-1}$ to generate and exchange the other part $e_B := r^{d_B}$ of the key. A computes $e_B^{-t_A}$, B computes $z_A^{d_B}$, both of which are the same key $\omega^{-k_A t_A d_B}$.

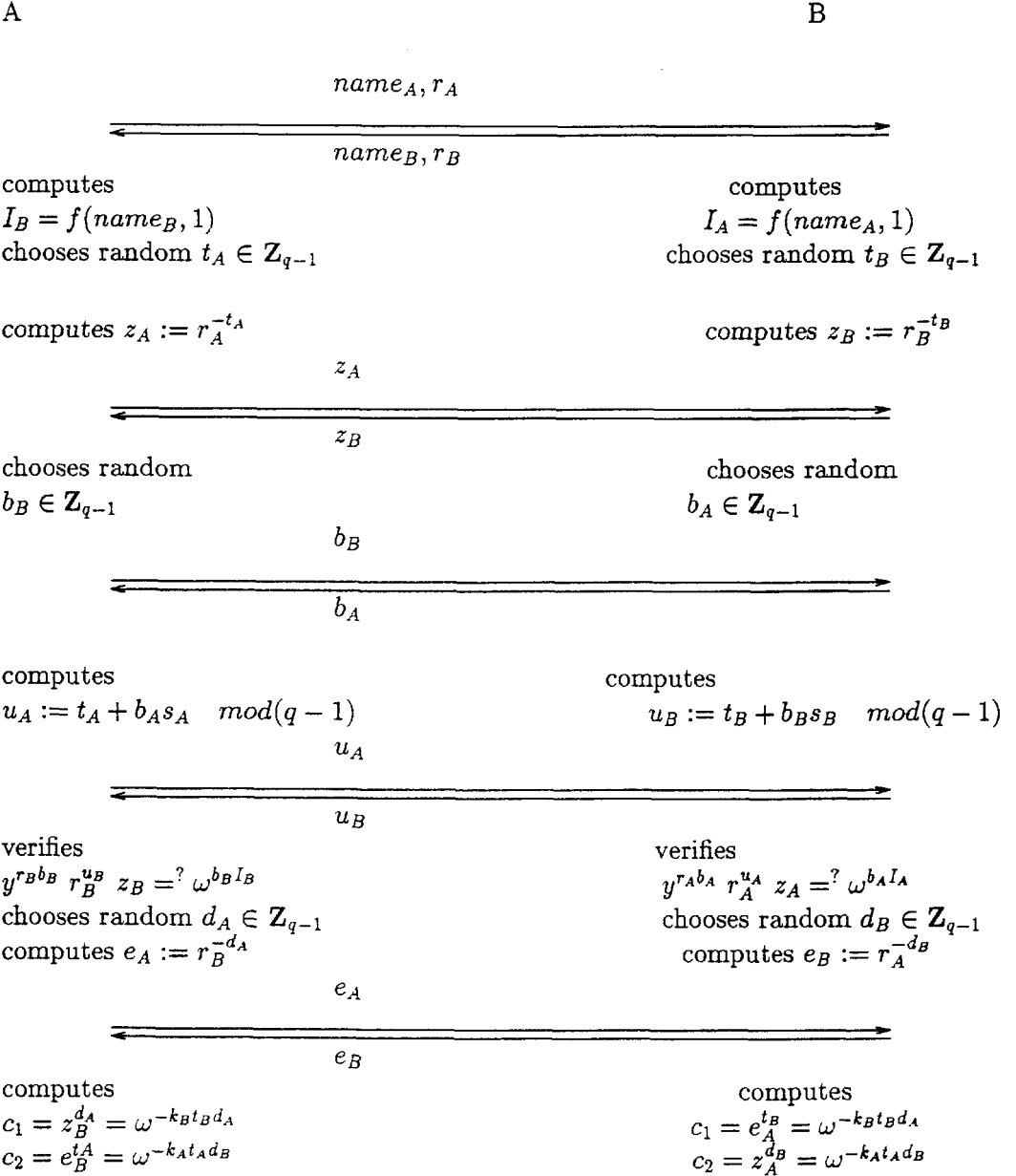
Notes:

1. The value z_A is an integral part of the authentication protocol and therefore it cannot be altered by an intruder without causing the authentication to fail. Using it as a part of the session key is an effective way to combine authentication and key exchange as demanded above.

2. SKIA must take care that the multiplicative order of $r_A = \omega^{k_A}$ is as high as possible, which can be guaranteed if $\text{GF}(q)$ is a Fermat-Field.
3. Neither A nor B need the knowledge of k_A . On the other hand they must not have this knowledge, otherwise x could be reconstructed from r_A, s_A, k_A and I_A .
4. The trivial cases $t_A = 1$ or $b_A = 0$ have to be avoided.
5. $m > 1$ or $h > 1$ significantly decrease the probability that (ab-)user C pretending to be A can successfully guess the matrix $((b_{A,j,i}))$ and "tune" his $z_{A,i}$ and $u_{A,i}$ accordingly.
6. If $m > 1$, the $s_{A,j}$ must be stored in secure memory. Otherwise a number of conspiring users would be able to pool their information to create new valid IDs without participation of SKIA.
7. $h > 1$ leads to the problem of having to combine several $z_{A,j}$ during the key exchange. Otherwise only the security of one pass of the authentication protocol would apply to the key.
8. Even if, for $m, h = 1$ a forger C could guess the right challenge b_A and would send $z'_A := \omega^{b_A I_A} y^{-r_A b_A} r_A^{-u_C} = (r_A^{s_A})^{b_A} r_A^{-u_C}$ instead of z_A and $u'_A := u_C$ instead of u_A for a chosen u_C he could only gain a correct authentication. There seems to be no effective way for him to compute the discrete logarithm of his certain z'_A which he needs to compute the correct session key. So he cannot make use of the authenticated session. Thus, the security of the combined protocol seems to be independent from the difficulty of guessing $((b_{A,j,i}))$ and $m, h = 1$ may be used without decreasing the overall security.
9. In Beth's original publication the choice of $((b_{A,j,i}))$ is limited to a "suitably chosen subset" $R^{m \times h}$ of $\mathbf{Z}_{q-1}^{m \times h}$ for proof technical reasons. Increasing the choice space for a practical implementation seems to result in higher, not in lower security.
10. As the amount of data needed for the protocol is small and these data are rarely changing and not sensitive, there is no direct demand for a CSC as mentioned in the previous section. Even if a CSC were used for logging established connections, its failure would not crash the whole cryptographic system. So its availability is less important than it was in SELANE.

The protocol in this form ensures B that A is authentic but not vice versa. This is acceptable if B is a host and A a user. However it is desirable that both communicating parties are sure about each other's identity, especially in a network with equivalent nodes that are clients as well as servers to other nodes.

So we propose a symmetric version of the above scheme (for $m, h = 1$):



Notes:

1. The protocol provides two keys. z_A and z_B cannot be used to construct one key as they are powers of different bases r_A and r_B and their logarithms k_A and k_B are unknown to A and B.
2. A is sure that z_B is authentic so she knows that she shares key c_1 with B. But she is not sure who sent e_B and thus with whom she shares key c_2 . The same holds for B concerning z_A and e_A and respectively keys c_2 and c_1 .
However A can be sure that if she would share key $c'_2 := e_C^{t_A} = \omega^{-k_A t_A d_C}$ with an intruder C then C cannot share another key $c''_2 := z_C^{d_B}$ with B, because B uses the authentic z_A to generate his key c_2 and C does not know the corresponding t_A . Again the same holds for B and key c_1 .

To overcome the problem mentioned in the note above consider the following extension of the protocol:

- A chooses a random string g and sends it to B encrypted under key c_2 .
- B receives and decrypts g under key c_2 , encrypts it under key c_1 and returns it to A.
- If A receives g encrypted under key c_1 , then key c_2 is considered valid and will be used as the session key.

Notes:

1. If A receives g encrypted under c_1 , she is sure that it was sent to her by B. So B must have received it from her under key c_2 as mentioned above in note 2.
2. Nobody else besides A and B can share key c_2 unless the Diffie-Hellman protocol is broken.
3. Repeat the same protocol starting with B and a different random string h to verify key c_1 . The second key may e.g. be used for synchronisation or as replacement of c_2 during the session. If the involved cipher is a stream cipher or equivalent, c_1 can be verified even with only one additional transmission.
4. An implementation must provide a suitable timeout when A waits for the return of g (and of course also for any other transmission).

4 Acknowledgements

The authors would like to thank Th. Beth, M. Clausen, D. Gollmann, H.-P. Rieß and S. Stempel for interesting discussions about the ideas presented in this paper.

It is noteworthy that Christoph Günther devised a surprisingly similar scheme for combined authentication and key exchange under a different approach [4]. We wish to thank him for the time he spent working with us on the differences and similarities of the two approaches.

References

- [1] F. Bauspieß: *SELANE - SEcure Local Area Network Environment*, Studienarbeit, Universität Karlsruhe, 1988
- [2] Th. Beth: *Efficient Zero-Knowledge Identification Scheme for Smart Cards*, Proc. of Eurocrypt '88, Springer LNCS 330, pp. 77-84, 1988
- [3] W. Diffie, M. Hellman: *New Directions in Cryptography*, IEEE Trans. on Information Theory, IT-22, pp. 644-654, 1976
- [4] Ch. Günther: *Diffie-Hellman and El-Gamal protocols with one single authentication key*, Proc. of Eurocrypt '89
- [5] R. Rueppel: *Key Agreements based on Function Composition*, Proc. of Eurocrypt '88, Springer LNCS 330, pp. 3-10, 1988