Using one-way chains to provide message authentication without shared secrets

Bogdan Groza
Politehnica University of Timisoara, Faculty of Automatics and Computers
Bd. Vasile Parvan nr. 2, 300223 Timisoara, Romania,
Email: bogdan.groza@aut.upt.ro

Abstract

The objective of this paper is to propose a cryptographic protocol which provides authenticity in the exchange of information between two entities without using any shared secret and by using only one-way chains. Such a protocol may have many applications and may be of interest especially in constrained environments where computational power is limited since one-way chains may be constructed using some of the simplest cryptographic one-way functions. We propose and investigate two approaches based on one-way chains, which we call: delayed message authentication and direct message authentication. Both of them have some shortcomings and a final hybrid approach, which combines their advantages without inheriting their weaknesses, appears to be quite useful and effective.

1. Introduction

Cryptographic techniques have an important role in assuring security objectives that may vary from application to application. Among all security objectives, authentication seems to play the most important role, since other objectives, such as keeping information confidential, might not matter much when some one has no certainty on the source of the information.

The history of one-way chains, in the context of authentication, probably begins with the work of Lamport [7], which proposed the use of a one-way chain in order to authenticate a user to a remote system. Lamport's idea was to generate the one-way chain $f^0(x), f^1(x), f^2(x),..., f^i(x)$, where f is a one-way function, x is a secret value, $f^i(x)$ denotes the composition of f with itself for i times $f^i(x) = f\left(f^{i-1}(x)\right)$, of course $f^0(x) = x$; and then to use every value from the one-way chain as a password. We will define the length of this chain as the number of function compositions necessary to obtain the highest order element, therefore the length of the previous chain is i, also note that in fact the chain contains $i+1$ elements.

In order to construct such a one-way chain a one-way function is required. This is not a very restrictive condition since most cryptographic primitives behave as one-way functions. The first solution to this purpose is to use hash functions because they require low storage space and low computational power. But if the chain is too short it can be exhausted too quickly, while if it is too long it requires more computational power to be computed. In respond to this, some optimized solutions were proposed for efficient computation and traversal of hash chains [3], [6], [14]. However hash functions still have the disadvantage that the resulting chain has a fixed length. In order to remove this disadvantage the use of functions from public-key encryption can be an alternative. By using such functions the length of the one-way does not influence the computational cost and therefore extreme lengths may be chosen as well - but these functions require much more computational power. Using simple functions over groups of integers, such as the squaring function can offer some advantages [4], still this remains unsuitable for environments such as the sensor networks from [9]. For the rest of the paper we will not be concerned with how the one-way chain is generated, the proposed protocols work for one-way chains constructed on any of these methods.

One-way chains prove to have a number of advantages from which the most important: they require low computational power and their use does not depend on shared secrets. For this reason they were used in some applications, for example in the S-Key system by Haller [5] or in the electronic payment scheme proposed by Rivest and Shamir [13]. Some recent results show an increased interest for using one-way chains to assure authenticity in constrained environments, such as sensor networks, where computational power and communication abilities are drastically limited [9]. Other schemes that address the problem of message authentication in group or broadcast communication by using hash-chains are in [2], [11].

The objective of this paper is to propose a cryptographic protocol based on one-way chains that can be used in the exchange of authentic information between two entities (other proposals that address this problem are based on hash chains, we prefer the notion of one-way chain since, as stated, any one-way function can be used for this purpose). The proposed protocol will respond to the following necessities:

1) It is based only on one-way functions which do not require significant computational resources and therefore can be applicable in a large variety of environments.

2) It provides data authenticity which implies that information was not altered and originates from a particular entity.

3) It does not depend on shared secrets; every entity stores only its own secrets. This means that the proposed protocol is not based on a secret key model, however compromising any secret from any side leads to security loss on both sides so the proposed protocol is not a public-key model either.

4) It does not require a time synchronization between entities involved in a communication, e.g. there is no need for any entity to keep a secure clock on its side which is loosely synchronized with others entities clocks.

5) It provides a secure timeline which implies that messages arrive in the order they were sent and their order cannot be switched by an intruder.

Section 2 proposes a first solution for this purpose, which we call Delayed Message Authentication. This solution is shown to have a shortcoming and therefore in section 3 a different approach is proposed, which we call Direct Message Authentication. This solution also experiences some shortcomings and therefore in section 4 we propose a hybrid solution which combines the advantages of the previously proposed protocols without inheriting their weaknesses. Section 5 holds the conclusions of this paper.

2. The Delayed Message Authentication Protocol

2.1. The description of the protocol

In order to establish an authentic communication channel between two entities denoted A and B, we will use a one-way chain on each side. The protocol consists in a variable number of sessions and each session consists in exactly two rounds. Each session provides the necessary information to prove the authenticity of the message from the previous session and in particular each round is the confirmation of the previous round.

Suppose that A randomly chooses x_A and B randomly chooses x_B, both these values will be kept secret. We will define the session keys, which are elements of the one-way chains, for A and B by the following relations:

$$\sigma_A(k) = f^{\eta-k}(x_A), 0 \le k \le \eta \quad (1)$$

$$\rho_B(k) = f^{\eta-k}(x_B), 0 \le k \le \eta \quad (2)$$

Here k is the session number and η is an integer fixed by common agreement such that it would be feasible for any of the entities to manipulate a chain with the respective length. In session 0 the entities inform each other, in a secure manner to guarantee the authenticity of this information, of the values of $\sigma_A(0)$ and respectively $\rho_B(0)$ which represent the tips of the one-way chains and will be later used to verify the authenticity of the new session keys, this information is not secret but its integrity must be preserved.

Entity A starts the k^{th} communication session by sending a package with this structure: $M_{A,k}, MAC\left(M_{A,k}, \sigma_A(k+1)\right), \sigma_A(k)$. The significance of the notations is the following: $M_{A,k}$ denotes the message from session k, MAC is a message authentication code computed on $M_{A,k}$ with the key $\sigma_A(k+1)$ (which will be disclosed in session $k+1$) and $\sigma_A(k)$ is the current session key. Upon receiving the package from session k, entity B must verify that the session key $\sigma_A(k)$ is correct by checking that $f\left(\sigma_A(k)\right) = \sigma_A(k-1)$, where $\sigma_A(k-1)$ is the key disclosed in session $k-1$. If the session key proves to be correct then this package is memorized and the authenticity of the message $M_{A,k-1}$ from session $k-1$ can now be verified with the disclosed key $\sigma_A(k)$ by checking the message authentication code $MAC\left(M_{A,k-1}, \sigma_A(k)\right)$. Entity B will confirm the arrival of a correct session key by sending its own session key $\rho_B(k)$. The next communication session $k+1$ will be started by A only if the received value of $\rho_B(k)$ is correct and this can be easily verified by checking that $f(\rho_B(k)) = \rho_B(k-1)$. It might be also convenient for entity B to include an authentic message in its package from the second round; therefore the two communication rounds for session k can be as follows:

Session $k, 1 \le k \le \eta$

Round 1 $A \to B$:

$$M_{A,k}, MAC(M_{A,k}, \sigma_A(k+1)), \sigma_A(k)$$

Round 2 $B \to A$:

$$M_{B,k}, MAC(M_{B,k}, \rho_B(k+1)), \rho_B(k)$$

It is very important to note that each round will play the role of a confirmation for the previous round, for example round 1 of session k is the confirmation of round 2 from session $k-1$ while round 2 from session k is the confirmation of round 1 from session k, and such confirmation will be sent only if the session key from the confirmed round was correct.

The protocol can be stopped at any session by A, obviously the authenticity of the message from that session will be proved only when the protocol starts again and the next session key is disclosed. When A decides to restart the conversation it will send a new package with a new session key only if the package from the previous round was confirmed, otherwise it has to resend the package from the previous round. The same policy should be applied in the case of accidental stops of the protocol which may be caused by some communication failures or by an intruder. To be more accurate we will suppose that the conversation was stopped at session k described previously. If A has received the correct $\rho_B(k)$ the protocol can be restarted later by computing and sending the package for session $k+1$ otherwise the protocol will be restarted by resending the package from session k until a correct $\rho_B(k)$ is received. These rules must be strictly followed, since if A computes and sends the package for session $k+1$ without having the confirmation key $\rho_B(k)$ from session k, in the case that the package from session k was not received by B this package can now be forged by an attacker who will now be in possession of the session key $\sigma_A(k+1)$. The same rule must be strictly followed by B, the confirmation $\rho_B(k)$ is sent in the second round only if $\sigma_A(k)$ proves to be correct.

It may also appear convenient to let any entity start the protocol after it was stopped in some session; however, this is a challenging task and it is not easy to solve. Suppose that any entity can restart the protocol by sending the package for the next session if the package from the previous session was confirmed or otherwise by resending the last unconfirmed package - this will allow an attacker to replay every unconfirmed package until a confirmation is received from the other entity, and so on, making the protocol never stop. Therefore in the DeMA protocol only one of the entities, in our case A, will have the ability to start and stop the protocol in some session.

The informal argument on the security of the DeMA protocol is that a new element of the one-way chain is disclosed only when the other party has already received and store a particular message, thus being to late for the intruder to affect the authenticity of the message. A pre-play attack such as suggested in Note 10.7 from [8] does not influence the security of the scheme since A will reveal a new password only when he has correctly received the confirmation from B. The best thing an attacker can do is to alter the messages sent between the entities – but these messages will prove in the next not to be authentic.

2.2. The shortcoming of this approach

The limitation of the protocol occurs when the one-way chains are exhausted and the entities run out of session keys after the completion of η communication sessions. In order to prevent exhaustion, the chains must be re-initialized. This can be achieved by computing a new chain and sending an authentic message that contains the tip of the new chain. However the security problem which occurs is the following: when the entities will try to reinitialize their chains, even if an attacker would not be able to break the authenticity of the messages, he can simply replace the value of the tip of the chain with an arbitrary value and in the next session the authenticity of this value will fail while the entities will probably already run out of session keys (i.e. elements of the one-way chain) and will not be able to continue the communication. If chains constructed from public-key primitives are used the chain will never be exhausted and this limitation will not occur. However this approach might not be suitable when we cannot afford enough computational power because asymmetric primitives are more computational intensive than symmetric primitives. Therefore, a different approach will be proposed in the next section.

3. The Direct Message Authentication Protocol (DiMA)

3.1. Overview of the protocol

A simple explanation for the shortcoming in the previously described protocol is the following: any entity which receives a message in some round will confirm the arrival of such a message in the next round even if it was unable to verify its authenticity. In order to overcome this, we have to provide a mechanism for any entity to check the authenticity of the message before confirming that it received such message. A solution for this purpose is to use a one-time digital signature mechanism in order to prove the authenticity of the message in the session when it is received. There is an extensive work on one-time signatures and their use, a proposal that is closed to optimal and uses directed acyclic graphs to construct one-time signatures is in [1], protocols which use one-time signatures can be found in [10], [12], [15].

We will avoid using secret shared keys by using two different one-way chains on each side. The protocol will again consist in a variable number of sessions and each session will again have two rounds. In each round two elements from each one-way chain will be sent and the authentic message will be recovered from these two elements. A positive integer λ is fixed by common agreement such that it would be easy for any entity to compute the value of $f^{\lambda}(x)$ with λ successive compositions of the one-way function. If we consider the messages from each round represented as integers then they will be smaller than λ, if $m_{A,k}$ denotes the message then $0 \le m_{A,k} < \lambda$, of course the bit length of such a message will be $\lfloor \log_2(\lambda-1) \rfloor + 1$ bits.

Suppose that A randomly chooses x_A, y_A and B randomly chooses x_B, y_B, all these values will be kept secret. We will define the session key pair as $\theta_A(k, m_{A,k})$, $\omega_A(k, m_{A,k})$ and $\theta_B(k, m_{B,k})$, $\omega_B(k, m_{B,k})$ by the following relations:

$$\theta_A(k, m_{A,k}) = f^{\eta - m_{A,k} - (k-1)\cdot\lambda - 1}(x_A) \qquad (3)$$

$$\omega_A(k, m_{A,k}) = f^{\eta + m_{A,k} - k\cdot\lambda}(y_A) \qquad (4)$$

$$\theta_B(k, m_{B,k}) = f^{\eta - m_{B,k} - (k-1)\cdot\lambda - 1}(x_B) \qquad (5)$$

$$\omega_B(k, m_{B,k}) = f^{\eta + m_{B,k} - k\cdot\lambda}(y_B) \qquad (6)$$

For all four relations we assume $1 \le k \le \frac{\eta}{\lambda}, 0 \le m_{A,k} < \lambda$. Here k is the session number, η is an integer fixed by common agreement such that it would be feasible for any of the entities to manipulate a chain with the respective length and $m_{A,k}$, $m_{B,k}$ are the authentic messages from session k. We can also observe that the one-way chains are split into sequences of length λ and each such sequence will correspond to a particular session; thus, given the length of the one-way chains and the length of the sequences, the maximum number of communication sessions for each entity until these chains exhausts will be $\frac{\eta}{\lambda}$ (of course it will be natural for the value of η to be chosen as a multiple of λ since the number of sessions is an integer number).

In session 0, the entities will inform each other, in a secure manner to guarantee the authenticity of this information, of the values of the pairs $\theta_A(0, \lambda-1) = f^{\eta}(x_A)$, $\omega_A(0,0) = f^{\eta}(y_A)$ and $\theta_B(0, \lambda-1) = f^{\eta}(x_B)$, $\omega_B(0,0) = f^{\eta}(y_B)$ which represent the tips of the one-way chains. The structure of the one way chains is also depicted in figure 1.

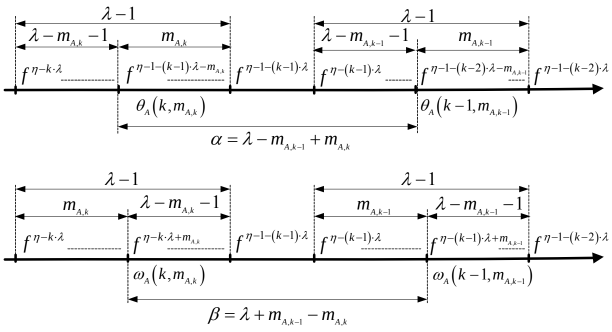
$\lambda-1$
$\lambda - m_{A,k} - 1$
$m_{A,k}$
$\lambda - m_{A,k-1} - 1$
$m_{A,k-1}$
$f^{\eta-k\cdot\lambda}$
$f^{\eta-1-(k-1)\cdot\lambda-m_{A,k}}$
$f^{\eta-1-(k-1)\cdot\lambda}$
$f^{\eta-(k-1)\cdot\lambda}$
$f^{\eta-1-(k-2)\cdot\lambda-m_{A,k-1}}$
$f^{\eta-1-(k-2)\cdot\lambda}$
$\theta_A(k, m_{A,k})$
$\theta_A(k-1, m_{A,k-1})$
$\alpha = \lambda - m_{A,k-1} + m_{A,k}$
$f^{\eta-k\cdot\lambda+m_{A,k}}$
$f^{\eta-(k-1)\cdot\lambda+m_{A,k-1}}$
$\omega_A(k, m_{A,k})$
$\omega_A(k-1, m_{A,k-1})$
$\beta = \lambda + m_{A,k-1} - m_{A,k}$

Figure 1. The structure of the one-way chains

In the first round of every communication session, A represents the message as an integer value $m_{A,k} \in [0, \lambda)$, computes the pair $\theta_A(k, m_{A,k}), \omega_A(k, m_{A,k})$ as described in (3),(4) and send this to B. After receiving the pair $\theta_A(k, m_{A,k}), \omega_A(k, m_{A,k})$, B will determine by successive compositions of function f two positive integers α, β with the following properties: if $k=1$ then $\alpha + \beta = \lambda + 1$ else $\alpha + \beta = 2 \cdot \lambda$, $f^{\alpha}(\theta_A(k, m_{A,k})) =$ $\theta_A(k-1, m_{A,k-1})$ and also

$f^{\beta}\left(\omega_A\left(k,m_{A,k}\right)\right) = \omega_A\left(k-1,m_{A,k-1}\right)$. If such integers exist then the authentic message is: if $k=1$ then $m_{A,1}=\alpha-1$ else $m_{A,k}=\alpha-\lambda+m_{A,k-1}$. Entity B must confirm the arrival of such message by sending a confirmation which consists in two elements of its one-way chains $\theta_B\left(k,m_{B,k}\right),\omega_B\left(k,m_{B,k}\right)$ computed for any message $m_{B,k}$ (in particular if B does not need to send any information to A a default message from the interval $\left[0,\lambda\right)$ could be used). The next communication session $k+1$ will be started by A only if the package received from B was authentic, A will verify the authenticity and recover the message in the same way as B did.

For the k^{th} session, the communication will be as follows:

Session $k, 1 \le k \le \frac{\eta}{\lambda}$

Round 1**:** $A \to B$: $\theta_A\left(k,m_{A,k}\right)$, $\omega_A\left(k,m_{A,k}\right)$

Round 2**:** $B \to A$: $\theta_B\left(k,m_{B,k}\right)$, $\omega_B\left(k,m_{B,k}\right)$

Again, it is important to note that each round will play the role of the confirmation for the previous round, and such confirmation will be sent only if the message from the confirmed round was authentic. As discussed for the previous protocol, the protocol can be stopped by A at any point. When A decides to restart the conversation it will send a new package with two new session keys only if the package from the previous round was confirmed, otherwise it has to resend the package from the previous round until a valid confirmation is received.

The same restriction, as in the case of the DeMA protocol, will remain and only one of the entities (in our case A) can start and stop the protocol.

Of course the exhaustion of the chains does not cause any problem since the entities can reinitialize their chains by computing two new chains and sending their tips as authentic messages as long as the current chains are not exhausted; the attack over the DeMA protocol will not hold any more.

Parameter λ can improve the flexibility of the communication. Suppose that one entity needs to send a message which is an integer greater than λ, let the message be $M>\lambda$. Since by using this protocol only small messages situated in the interval $\left[0,\lambda\right)$ can be sent in each round a chain of length $\lambda\cdot\left(\left\lfloor \log_{\lambda} M\right\rfloor+1\right)+1$ is needed in a number of $\left\lfloor \log_{\lambda} M\right\rfloor+1$ communication sessions with a computational effort of $2\cdot\lambda$ compositions of the one-way function to recover the small messages from each round. Indeed, by increasing the value of λ the number of communication rounds decreases while the length of the chain and the computational effort to recover the message increases.

Since using large chains requires more computational power it is likely that smaller values of λ will be preferable; however in environments where communication abilities are drastically limited increasing the value of λ may be considered at the price of more computational power.

3.2. Shortcomings of the protocol

The DiMA protocol will resist in the case of the attack described on the previous protocol and a persistent attacker may not compromise the security by making any entity run out of secret keys.

However, the disadvantages of the DiMA protocol compared to the DeMA protocol are the following: the one-way chains are quickly exhausted and many communication sessions are needed for sending only small amounts of authenticated information.

4. The Delayed Message Authentication / Direct Chain Authentication Protocol (DeMA/DiCA)

We conclude that the DeMA protocol is more effective than the DiMA protocol when exchanging authentic information; however, it succumbs when the chain is exhausted and needs to be re-initialized. On the other hand the DiMa protocol does not succumb when one needs to re-initialize the chains but exhausts very quickly the chains when exchanging authentic information. Therefore a hybrid approach, which combines the strong points from both of them without inheriting their weaknesses, is certainly more effective. The hybrid approach uses the DeMA protocol to exchange authentic messages and the DiMA protocol to authenticate the tips of the new one-way chains. The principles on which the protocol works are the following:

1) The protocol uses a one-way function which outputs δ bits.

2) Two one-way chains are generated by each entity: one of length $\eta+4\cdot\delta$ and the other of length $4\cdot\delta$, where η is an integer fixed by common agreement such that it would be feasible for any of the entities to manipulate a chain with the respective length. Let σ_A be the one way chain of length

$\eta + 4 \cdot \delta$ from A's side and ρ_B be the one-way chain of length $\eta + 4 \cdot \delta$ from B's side. Also, let $\omega_{\sigma,A}$ be the one-way chain of length $4 \cdot \delta$ from A's side and $\omega_{\rho,B}$ be the one-way chain of length $4 \cdot \delta$ from B's side.

3) The first element from each chain is used as an initialization value in an off-line initialization stage.

4) The next η elements from the chains σ_A and ρ_B are used to exchange authentic information between A and B with the DeMA Protocol.

5) The last $4 \cdot \delta$ elements from the chains σ_A and ρ_B along with the last $4 \cdot \delta$ elements from the chains $\omega_{\sigma,A}$ and $\omega_{\rho,B}$ are used to exchange the authentic information between A and B necessary to reinitialize the two one-way chains with the DiMA protocol. Obviously, in this case $\lambda = 4$ and in each of the δ sessions 1 bit from each of the 2 new tips of the new one-way chains is sent from one entity to the other (different values of λ may be used to improve the flexibility of the communication).

As we have already discussed for the DeMA and DiMA protocol, in the DeMA/DiCA protocol only one of the entities may start and stop the communication. Since the protocol is based on the application of the previously described protocols a more detailed description is not necessary.

5. Conclusions

Two protocols, DeMA and DiMA, to provide authenticity by using one-way chains and without any shared secret were proposed and analyzed. A final hybrid protocol between these two, which we call DeMA/DiCA, appears to be more useful and effective. This protocol can be used to exchange authentic information between two entities by using only one way chains and without using any shared secret. The solutions may appear complex; however, we note that they can be constructed on simple one-way functions which are easy to compute. Therefore, we expect that the protocols proposed in this paper are suitable especially in constrained environments where computational power is limited. As future work we consider constructing practical implementations of these protocols in order to obtain concrete results and estimate their computational efficiency, we are concerned with the use of both hash functions and functions from public key encryption, such as the discrete squaring function, for the generation of the one-way chains.

6. References

[1] Bleichenbacher, D., Maurer, U.M., On the efficiency of one-time digital signatures, Proc. ASIACRYPT'96, Springer LNCS 1163, 1996, pp. 145-158.

[2] Bergadano, F., Cavagnino, D., Crispo, B., "Individual Authentication in Multiparty Communications". Computer & Security, Elsevier Science, vol. 21 n. 8, 2002, pp.719-735.

[3] Fischlin, M., "Fast Verification of Hash Chains", RSA Security Cryptographer's Track 2004, Springer LNCS 2964, 2004, pp. 339-352.

[4] Groza, B., Petrica, D., Dragomir, T.L., "A time-memory trade solution to generate one-time passwords using quadratic residues in Zn", Studies in Informatics and Control, Vol. 14, No. 3, 2005, pp. 201-212.

[5] Haller, N., Metz, C., Nesser, P., Straw, M., "A One-Time Password System" RFC 2289, Bellcore, Kaman Sciences Corporation, Nesser and Nesser Consulting, 1998.

[6] Jakobsson, M., "Fractal hash sequence representation and traversal", Proceedings of IEEE International Symposium on Information Theory, 2002, pp. 437-444.

[7] Lamport, L., "Password Authentication with Insecure Communication". Comm. ACM, 24, 1981, pp. 770-772.

[8] Menezes, A.J., van Oorschot, P.C., Vanstone, S.A., "Handbook of Applied Cryptography", CRC Press. 1996.

[9] Perrig, A., Szewczyk, R., Wen, V., Culler D., Tygar, J.D., "SPINS: Security Protocols for Sensor Network", Proceedings of Seventh Annual International Conference on Mobile Computing and Networks MOBICOM, 2001.

[10] Perrig, A., "The BiBa one-time signature and broadcast authentication protocol", Proc. of ACM Conference on Computer and Communications Security, 2001, pp.28-37.

[11] Perrig, A., Canetti, R., Tygar, J. D., Song, D., "The TESLA Broadcast Authentication Protocol", In CryptoBytes, 5:2, Summer/Fall 2002, pp. 2-13.

[12] Reyzin, L., Reyzin, N., "Better than BiBa: Short One-Time Signatures with Fast Signing and Verifying", Information Security and Privacy - 7th Australasian Conference, Springer LNCS 2384, 2002, pp.144-153.

[13] Rivest, R., Shamir, A., "Payword and Micromint: Two simple micropayment schemes". CryptoBytes, volume 2, no. 1, RSA Laboratories, 1996.

[14] Sella, Y., "On the Computation-Storage Trade-Offs of Hash Chain Traversal", Proceedings of Financial Cryptography, Springer LNCS 2742, 2003, pp.270–285.

[15] Zhang, K., “Efficient Protocols for Signing Routing Messages”, Symposium on Network and Distributed Systems Security, San Diego, California, 1998.