

Version of July 1998

**Authenticated Public Key Encryption Schemes using Universal Hashing  
(a submission to IEEE P1363a)**

By Yuliang Zheng (Email: yzheng@fcit.monash.edu.au)

This proposal describes a technique for strengthening public key encryption so that it is secure against adaptively chosen ciphertext attacks. It uses universal hash functions in two different ways: (1) to extract randomness hidden in a longer string, (2) to authenticate a message for the purpose of non-malleability of a ciphertext. No one-way hashing is involved in the technique.

# Universal Hash Function in Randomness Extraction and Message Authentication

## As a Randomness Extractor

It has been known for some time that an *almost strongly universal class of hash functions*, or *universal hash functions* for short, can act as an efficient randomness extractor [ImpagliazzoZuckerman89], and hence can be used in cryptographically strong pseudo-random number generation [HstadEtal].

## As a Message Authentication Code

A *message authentication code* is a method by which two parties who share a common secret key can exchanged messages in an authenticated manner, namely, they can detect modifications or fabrications by an un-authorized third party. The shared common key between the two parties is usually chosen uniformly at random from the set of all possible keys. It is well known that an almost strongly universal class of hash functions is an excellent candidate for message authentication codes. More discussions on the use of universal hash functions in message authentication can be found in [Wegman&Carter], while information on constructing computationally efficient universal hash functions can be found in [den Boer93, Shoup96, Atici&Stinson96].

It is assumed that an instance  $S$  of an almost strongly universal class of hash functions is specified by an octet string of length  $sLen$ . In addition, it is required that  $S$  can take an input message  $M$  of an arbitrary length. The output of  $S$  is a string  $H$  of a fixed length  $hLen$ .

**Input:** an octet string  $M$ , the message.

**Output:** an octet string  $H$  of length  $hLen$ , the hash value.

## Evaluation Universal Hash

An evaluation universal hash is an almost strongly universal class of hash functions based on the evaluation of a polynomial over a finite field. It was first proposed by den Boer in [den Boer93], and was further studied by Shoup in his Crypto'96 paper. An instance  $S$  of an evaluation hash is specified by a string of  $2 * hLen$  octets, i.e., twice the length of its output.

The application of the evaluation universal hash  $S$  to an input message  $M$  consists of the following steps:

1. Break up  $M$  into  $n$  blocks  $M_{n-1}, M_{n-2}, \dots, M_1, M_0$ , each containing  $hLen$  octets. If the last block  $M_{n-1}$  has less than  $hLen$  octets, it should be appended with the required number of all-zero octets.
2. For each  $0 \leq i \leq n-1$ , use OS2FEP to convert  $M_i$  to an element  $m_i$  in  $GF(2^{8 \cdot hLen})$ .
3. View  $m_{n-1}, m_{n-2}, \dots, m_1, m_0$  as the coefficients of a polynomial  $m(x)$  of degree less than  $n$  over  $GF(2^{8 \cdot hLen})$ , namely
 
$$m(x) = m_{n-1}x^{n-1} + m_{n-2}x^{n-2} + \dots + m_1x + m_0$$
4. Denote the first  $hLen$  octets of  $S$  by  $S_1$  and the remaining  $hLen$  octets by  $S_2$ . Use OS2FEP to convert  $S_1$  and  $S_2$  to  $\alpha$  and  $\beta$  in  $GF(2^{8 \cdot hLen})$  respectively.
5. Evaluate the following polynomial
 
$$h = m(\alpha) \cdot \alpha + \beta = (m_{n-1}\alpha^{n-1} + m_{n-2}\alpha^{n-2} + \dots + m_1\alpha + m_0) \cdot \alpha + \beta.$$
6. Convert  $h$  to an octet string  $H$  with FE2OSP and output  $H$  as the hash value of  $M$  under  $S$ .

Notes:

The finite field  $GF(2^{8 \cdot hLen})$  can be specified by an irreducible polynomial  $p(x)$  of degree  $8 \cdot hLen$  on  $GF(2)$ . Once the field is specified, it is fixed during all subsequent communications.

## Rationale for selecting an almost strongly universal class of hash functions

Here are the criteria used:

1. All instances in the class can be computed efficiently.
2. The specification of an instance in the class should be short, ideally by a *constant* number of bits or octets.
3. An instance in the class should be able to hash a message of an arbitrary length.
4. Every string of an appropriate length should correspond to an instance in the class. This means no random bits will be wasted.
5. When the output of the function is long enough, say of 64 or more bits, the failure probability  $P_{d_1}$  should be low enough for messages of all sizes that may arise in practice. Note that  $P_{d_1}$  is also called the success probability of a spoofing attacker after seeing a valid pair of message and its corresponding authentication tag.

When all these criteria are taken into account, the best candidate boils down to the evaluation universal hash function. For this hash function, we have  $sLen = 2 \cdot hLen$

and  $P_{d_i} \approx \frac{n}{2^{8*Len}}$ . Experiments by Shoup show that when  $p(x)$  is a sparse irreducible polynomial, the efficiency of the evaluation universal hash function is comparable to that of MD5 one-way hash function.

## DLAES-UHF

DLAES-UHF is Discrete Logarithm Authenticated Encryption Scheme using Universal Hash Function. It consists of an encryption operation and a decryption operation.

### DLAES-UHF encryption operation

DLAES-UHF encryption operation is similar to DLES encryption operation. The differences are outlined below.

**Additional input:** octet string *controlInfo*, the optional control information.

**Additional auxiliary functions:** two instances of an almost strongly universal class of hash functions (or a universal hash function, for simplicity). The number of octets required to specify an instance function is *sLen*, and the output of the function is a string of *hLen* Octets.

**Output:** same as in DLES encryption operation, except by *hLen* octets longer.

**Primitives:** same as in DLES encryption operation.

The encryption operation shall proceed as follows:

1. Generate a one-time key pair  $(x, u)$  from the DL parameters with primitive DLKGP. Convert  $x$  to an octet string  $X$  of length  $l$  with primitive I2OSP.
2. Derive a field element  $t$  from the public key  $y$  and the one-time private key  $u$  with primitive DLSVDP.
3. Generate uniformly at random an instance  $F$  of the almost strongly universal class of hash functions.  $F$  is a string of  $sLen$  octets. Apply  $F$  on  $t$  to "extract" a random secret value  $k$  of  $hLen$  Octets.
4. Convert  $k$  to an octet string  $K$  of length  $hLen$  with primitive I2OSP.
5. Generate a string  $Z$  of length  $mLen+sLen$  from the string  $K$  with the mask generation function. The first  $mLen$  octets of  $Z$  are a mask string and denoted by  $Y$ , and the remaining  $sLen$  octets of  $Z$  specify a universal hash function and are denoted by  $S$ .
6. Combine  $Y$  with the message  $M$  by bitwise exclusive-or to produce an octet string  $C = Y \oplus M$ .
7. Concatenate  $M$  and *controlInfo* to produce  $M' = M \parallel controlInfo$ . Apply the universal hash function  $S$  to  $M'$  to produce the hash value  $H$ .  $H$  is an octet string of length  $hLen$ .
8. Concatenate the octet strings  $X$ ,  $F$ ,  $H$ , and  $C$  to form an octet string  $EM = X \parallel F \parallel H \parallel C$ . The string  $EM$  has length  $l + sLen + hLen + mLen$ .
9. Output the octet string  $EM$  as the encrypted message.

**Remarks:** For simplicity, the two almost strongly universal classes of hash functions (one used to "extract" a random seed  $K$  for the mask generation function, and the other to generate an authentication tag  $H$ ) have been assumed to have the same parameters ( $sLen$  and  $hLen$ ). In practice this does not have to be the case. One may even choose to use two almost strongly universal classes of hash functions that are based on different mathematical techniques.

## DLAES-UHF decryption operation

DLAES-UHF encryption operation is similar to DLES encryption operation. The differences are outlined below:

**Additional input:** an octet string *controlInfo*, the optional control information.

**Additional auxiliary function:** two instances of an almost strongly universal class of hash functions (or a universal hash function, for simplicity). The number of octets required to specify an instance function is  $sLen$ , and the output of the function is a string of  $hLen$  Octets.

**Output:** the decrypted message, an octet string  $M$  of length  $emLen - l - hLen - sLen$ , or "invalid."

**Primitives:** same as in DLES decryption operation.

The message  $M$  shall be computed by the following or an equivalent sequence of steps:

1. Parse  $EM$  into four octet strings:  $X$ , the first  $l$  octets,  $F$ , the next  $sLen$  octets,  $H$ , the next  $hLen$  octets, and  $C$ , the remaining octets. Let  $mLen$  be the length of  $C$ .
2. Convert  $X$  to an integer  $x$  with primitive OS2IP.
3. Derive a field element  $t$  from the one-time public key  $x$  and the private key  $v$  with primitive DLSVDP.
4. Apply  $F$  on  $t$  to "extract" a random secret value  $k$  of  $hLen$  octets.
5. Convert  $k$  to an octet string  $K$  of length  $hLen$  with primitive I2OSP.
6. Generate a string  $Z$  of length  $mLen+sLen$  from the string  $K$  with the mask generation function. The first  $mLen$  octets of  $Z$  are a mask string and denoted by  $Y$ , and the remaining  $sLen$  octets of  $Z$  specify a universal hash function and are denoted by  $S$ .
7. Compute the bitwise exclusive-or of  $Y$  and the octet string  $C$  to produce the message  $M = Y \oplus C$ .
8. Concatenate  $M$  and *controlInfo* to produce  $M' = M || controlInfo$ . Apply the universal hash function  $S$  to  $M'$  to produce the hash value  $H'$ .  $H'$  is an octet string of length  $hLen$ .
9. If  $H' \neq H$ , output "invalid", and stop.
10. Output  $M$  as the decrypted message.

## Security Consideration

To argue the security of the scheme, we need to assume *that the Diffie-Hellman key in a sub-group of order  $q$  in  $Z_p^*$  hides an adequate number of simultaneously hard bits*. For the above scheme, this number is  $8 * hLen$  which typically takes a value between 64 and 128. It is currently unclear as to whether this assumption is stronger than the standard Diffie-Hellman assumption (in the group  $Z_p^*$ ) or not. The best result known so far appears to be that Diffie-Hellman (in the group  $Z_p^*$ ) hides at least  $O(\log \log p)$  simultaneously hard bits (see Boneh and Venkatesan's Crypto'96 paper).

Under the above assumption, we can argue the scheme is "non-malleable" and hence secure against adaptively chosen ciphertext attacks.

First we note that if Alice and Bob share a truly random one-time secret key of  $mLen + sLen$  octets, they can achieve truly secure and "non-malleable" communications. What they have to do is as follows:

1. to use the first  $mLen$  octets to encrypt a message (say via XOR) and,
2. to use the next  $sLen$  octets as an instance of the hash function to generate an authentication tag.

The communications are secure and "non-malleable" in the information theoretical sense --- even an all powerful attacker will not be able to learn any information about the message, not can the attacker create a valid message-tag pair between Alice and Bob even if the attacker has access to Bob's deciphering algorithm.

When the truly random secret string between Alice and Bob is replaced by a pseudo-random string, the communications between them are still secure and "non-malleable", albeit in a polynomial time sense. Thus Alice and Bob can share a relatively short string, and use a pseudo-random number generator to extend it into the required length, without sacrificing security in all practical applications.

In the our scheme, the Diffie-Hellman key generation method, together with the randomness extractor, acts as a session key generation scheme, creating a random and secure one-time key between Alice and Bob. This shared one-time key is then used in conjunction with the above secure and non-malleable communication scheme. Therefore we can say that the resulting scheme is secure and non-malleable.

## Extension

The technique can be extended to elliptic curve based schemes.

## Advantages

1. The technique is simple and easy to understand.
2. As no one-way hash function is involved, the technique does NOT rely on the security of a one-way hash function.
3. The universal hash functions used by these schemes are NOT based on any unproved *cryptographic* assumption. The past few years have witnessed an explosive interest in using universal hashing in message authentication.
4. For a suitably chosen universal hash function, such as the evaluation universal hash proposed den Boer in 1993 and further investigated by Shoup in his Crypto'96 paper, the speed of our scheme is comparable to that of an authenticated encryption using a one-way hashing such as MD5 and SHA.

## Known Limitations

None.

## Copyright Issues

The author has not applied for patenting for the technique.

## References:

M. Atici and D. Stinson, "Universal hashing and multiple authentication", Advances in Cryptology - CRYPTO'96, Lecture Notes in Computer Science, 1109 (1996), Springer-Verlag, 16-30.

B. den Boer, "A simple and key-economical unconditional authentication scheme", Journal of Computer Security, 2 (1993), 65-71.

J. Hastad, A. W. Schrift and A. Shamir: "The discrete logarithm modulo a composite hides  $O(n)$  bits".

R. Impagliazzo and D. Zuckerman: "How to recycle random bits", Proceedings of 30<sup>th</sup> FOCS, 1989, pp.248-254.

V. Shoup, "On fast and provably secure message authentication based on universal hashing", Advances in Cryptology - CRYPTO'96, Lecture Notes in Computer Science, 1109 (1996), Springer-Verlag, 313-328. (see also an Erratum available from the author at [shoup@bellcore.com](mailto:shoup@bellcore.com)).

M. Wegman and L. Carter, "New hash functions and their use in authentication and set equality", Journal of Computer and Systems Science, 22, 1981, 265-279.

Y. Zheng and J. Seberry, "Immunizing public key cryptosystems against chosen ciphertext attacks", IEEE Journal on selected areas in communications, 11 (5), 1993, 715-724.