International Journal of Computer Networks and Communications Security

VOL. 4, NO. 3, MARCH 2016, 56–62
Available online at: www.ijcncs.org
E-ISSN 2308-9830 (Online) / ISSN 2410-0595 (Print)

A New secure email scheme Using Digital Signature with S/MIME

Mehrdad AhmadZadeh Raji[1], Fatemeh Amiri[2], Mohsen Ahmadian[3]

[1] PhD, [3] Bsc Students at Razi University, Iran

[2] Msc Student at Shiraz University, Iran

E-mail: [1]mraji@razi.ac.ir, [2]fatemehmahsa.amiri@yahoo.com, [3]mohsen.etc@gmail.com

ABSTRACT

Email as one of the most popular application among internet of things, needs more attention in front of potential dangerous attacks. Digital Signature is known as one of the most prominent applications of public key cryptography to resist in front of attacks. To save secure email against forgery, S/MIME is of the best choices in which uses digital signature. However, S/MIME use RSA with a weakness hash function that may be broke by intruders. On the other hand, the speed of implementing S/MIME push some time complexity. Therefore, it need a better scheme to refuse attacks. This paper presents a new scheme of an efficient email application which is uses of a secure public-key encryption – digital signature. We implement our scheme with three most popular digital signature algorithm: RSA that currently uses by S/MIME (for comparison), ELGAMAL and elliptic curve (ECDSA) with different hash functions in PHP as one of the best web languages. As our contribution we discuss and analyze security aspects of proposed digital signature schemes to choose the best scheme which has features of low computational complexity and fast operating speed for using beside S/MIME.

Keywords: *Cryptography, Communication, information, email Security, data encryption, email Security, S/MIME.*

1 INTRODUCTION

1.1 Email Security

The way a client uses to be sure that his message will not be intercepted by someone is the most important and urgent question that security professionals have to answer when dealing with email systems. Many attempts have been made to add security facilities to Internet mail but in fact many of today's users only use authentication (digital signatures) [1]. OpenPGP and S/MIME protocols are the most prominent standardized security solutions that were originally developed for Email. The current work on S/MIME is being done in the IETF's S/MIME Working Group. The charter for the S/MIME WG [2] states clearly that the purpose of the group is to create S/MIME v3 protocols that can become IETF standards.

S/MIME (Secure/Multipurpose Internet Mail Extensions) is a widely accepted protocol, for sending digitally signed and encrypted messages [3].

It adds several security features to the insecure SMTP.

S/MIME was originally developed by RSA Data Security, Inc. It is based on the PKCS #7 data format for the messages, and the X.509v3 format for certificates. PKCS #7, in turn, is based on the ASN.1 DER format for data. However, it suffers from some disadvantages as well. Some of these problems that we interest to discuss about include:

a) privacy concerns of mail senders can pass a digitally signed message to others [7]
b) non-repudiation in situations of lost keys [25]
c) unsigned message headers because of long length of key used in RSA [26]
d) It uses RSA algorithm for signatures and a weak algorithm for encryption (RC2 with 40-bit keys) [2].

Among these weaknesses in this paper as our contribution we just focus on the digital signature

used by S/MIME. Other factor need more attend and are out of the domain of this work. We want to design and implement an email application using digital signature to authenticate users that use mailing application. We use some well-known schemes with different hash function to discuss about their effectiveness and propose a new scheme for S/MIME in authentication step.

1.2 Introduction to Cryptography

A cryptographic primitive that is fundamental in authentication and authorization, is the principal of digital signature which its purpose is to provide a means for an entity to bind its identity to a piece of information [23].

Digital signature schemes allow a signer S who has established a public key pk to "sign" a message in such a way that any other part y who knows pk (and knows that this public key was established by S) can verify that the message originated from S and has not been modified in any way. This method not only guarantee web applications reliability and help to promote the trust, but also seem to be fast and does not waste bandwidth and other resources to compute. Signature schemes can be viewed as the public-key counterpart of message authentication codes, though there are some important points to regard as we will see in the following [4].

This work is organized as follows: We provide mathematic background on digital signature in Section II and details about our implementation in Section III. Performance results, Risk analysis and a comparison are part of Section IV, before we conclude with some remarks in Section V.

2 MATHEMATICAL BACKGROUND

2.1 Digital Signature

A signature scheme is a method of signing a message stored in electronic form [25].
A signature scheme is a five-tuple *(P, A, K, S, V),* where the following conditions are satisfied:

1. *P* is a finite set of possible messages
2. *A* is finite set of possible signatures
3. *K* , the key space, is a finite set of possible keys
4. For each $K \in$ K, there is a signig algorithm $\mathbf{sig}_K \in S$ and a corresponding vrificatin algorithm $\boldsymbol{ver}_K \in V$. Each $\boldsymbol{sig}_K : P \rightarrow A$ and $\boldsymbol{ver}_K\ P \times A \rightarrow \{true\ ,\ false\}$ are functions such that the following equation is satisfied for every message $x \in P$ and for every signature $y \in A$:

$$ver\ k(x) = \begin{cases} true & ,if\ y = \boldsymbol{sig}_k\ (x) \\ false & ,if\ y \neq\ \boldsymbol{sig}_k\ (x) \end{cases}$$

A pair *(x , y)* with $x \in P$ and $y \in A$ is called a signed message. This algorithm is applied in all three schemes that we focus in this experiment, meaning RSA, ElGamal and ECDSA. Stinson in [25] represented a randomized mechanism for ElGamal. RSA was proposed shortly after discovery of public- key cryptography [9] and is so popular for its speed and simplicity and finally the Elliptic Curve Digital Signature Algorithm (ECDSA) is the elliptic curve analogue of the Digital Signature Algorithm (DSA). It is the most widely standardized elliptic curve-based signature scheme, appearing in the ANSI X9.62, FIPS 186-2, IEEE 1363-2000 and ISO/IEC 15946-2 standards as well as several draft standards [9].
Other details of its specification can be found in the NIST Digital Signature Standard [10] [11].

2.2 Hash Function

A cryptographic hash function is a function that takes any data and returns a fixed- size piece of information [26] [24]. Currently the most popular cryptographic hash functions belong in the SHA - 1/256/224/512 family. In this paper we want to analyze our implementation with some of these hash functions to realize the best candidate for each scheme.

3 IMPLEMENTATION

In this section, the implementation of three popular digital signature schemes is presented: ElGamal, RSA (that currently uses by S/MIME) and ECDSA as a security check part for an email application to test and compare the performance and characteristics of these schemes. Most of high level web applications are written with PHP. So, in this paper PHP language is handled to implement these schemes. To analyze performance results simulation tools available on Netbeans software are used. This application is written based on an implementation of DSS using one of the major multi-precision mathematics extensions, GMP available in PHP. GMP stands for GNU Multiple Precision arithmetic library. This library is freely available under the GNU general public license, and is optimized for performance [12]. The implementation leverages the object- oriented syntax of PHP to clearly delineate collections of properties and operations on them and the database used for designing this implementation is MySQL 5.5.

The PHP-application includes a Test Suite, along with a class encapsulating NIST published safe curves at various key-lengths. NIST Curve class is used as an encapsulation of static methods that contains all NIST published elliptic curves that are approved as secure for elliptic curve cryptography.[13][14].

The key generation process generates the public and private keys in pairs – figure 2. This step in S/MIME is one of responsibilities of CA. however this paper just want to implement different schemes with a variety of hashes to compare their effectiveness as a key part of an email security structure. If required, the keys can be viewed after generation. This application uses MD5, SHA1, SHA256 and SHA512 hash algorithms for the digital signature. SHA256 and SHA512 are more secure than the other ones. However, SHA512 needs more computational resources and this paper tried to analyze that is it good to use this secure function or it is not economical [24].

Fig. 1. Key Generation phase

After key generation the Signature Generation module is processed when Alice (sender) want to send a message to Bob (receiver) which is supposed to be signed by the signer and sent to the recipient. See figure 3.

Fig. 2. Signature Generation

Further at the receiving time, the recipient (Bob inbox) verifies the signature by execution of Signature Verification process to authenticate that the message has been send by the authentic sender and to validate that the message has been tampered or not. The figures 4 and 5 shows the execution of the developed application.

If Bob checks his inbox, he can see recent Alice message (figure 4). Now if this message is sent true with no forgery, he can find a verify status in his details of message part, see figure 5.

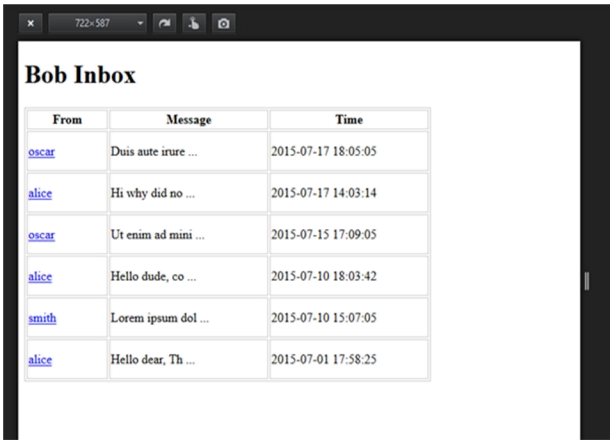

Fig. 3. Bob inbox

Fig. 4. Details of selected message in figure 4

However, if Oscar try to alter Alice message in between, then Bob can find out that this message is not from Alice with an unverified flag, see figure 6.

Fig. 5. details of an altered message by Oscar

4 RESULTS AND ANALYSIS

4.1 Test Environment

To test the efficiency of the secure communication, two nodes in the Local Area Network (LAN) are established. Two nodes running on two computers are set up. These two computers are running 64-bit Windows 8.1 with the hardware environment (Intel(R) Core (TM) i3-2120 CPU 3.30GHz,

RAM: 4.00GB). All applications are running on the Lighttpd, PHP platform.

4.2 Results

The test results of the developed web application for signature generation and signature verification using MD5, SHA1 and SHA256 and SHA512 are tabulated in microseconds as shown in Tables 1 to 3. In table 1 you can see the timed that need except by the algorithm.

Table 1: time needed for hash functions

Hash Function	Time needed
MD5	0.000941
SHA1	0.001264
SHA256	0.002611
SHA512	0.0004491

Table 2 shows the comparison of ECC with RSA, DSA (ElGamal), and DH in terms of key length and time to break on machine running 1 MIPS [15]. Due to this Table we select key length 256 for elliptic curve and 3072 for our RSA and ElGamal schemes to analyze in a same security manner. We select a text file with size 344 KB for all the experiments.

Table 2: Key comparison of Symmetric, RSA/DSA/DH, ECC [16]

Symmetric	RSA/DSA/DH	ECC	Time to break in MIPS years
80	1024	160	10^{12}
112	2048	224	10^{24}
128	3072	256	10^{28}
192	7680	384	10^{47}
256	15360	512	10^{66}

In Table 3and Table 4 we have done some experiments for all the schemes and phases for testing both CPU and memory consumption.

Table 3: Experimental Results of CPU usage- in microseconds

Digital Signature Scheme	MD5		SHA1		SHA256		SHA512	
	Sign. Gen.	Sign. Verify	Sign. Gen.	Sign. Verify	Sign. Gen.	Sign. Verify	Sign. Gen.	Sign. Verify
ElGamal	0.018723	0.009031	0.031243	0.009555	0.036037	0.011188	0.044959	0.013915
RSA	0.009033	0.001248	0.003645	0.001549	0.005032	0.002941	0.006976	0.004818
ECDSA	0.004259	0.517166	0.0004644	0.51590	0.006161	0.520569	0.007959	0.524041

Table 4: Experimental Results of Memory usage - in Byte

Digital Signature Scheme	MD5		SHA1		SHA256		SHA512	
	Sign. Gen.	Sign. Verify	Sign. Gen.	Sign. Verify	Sign. Gen.	Sign. Verify	Sign. Gen.	Sign. Verify
ElGamal	4176	496	4184	504	4216	528	4272	592
RSA	35184	1880	35192	1872	35432	1880	35208	1880
ECDSA	6192	728	6200	736	6232	760	6288	824

5 COMPARISON

It's generally thought that the security of the 256 bits ECC is equal to the 3072 bits RSA which is uses by S/MIME and ElGamal cryptography and the arithmetic speed is faster. Short scale of secret key, fast realizing speed and high security are its main advantages. It is especially caught on in the situation such as weak calculating capacity, limited integrated circuit space, limited broadband and the high-speed-realizing case [17].

However, in this paper we do an implementation with some popular digital signature schemes for web application to recognize the detail difference between them. It can be clearly seen that the amount of CPU consumption in RSA in both signature generation and verification in better that the others schemes. After that ECDSA and ElGamal compete. Since ECDSA is faster in signature generation but in verification ElGamal is better, see figure 7 and 8.

At the same time, when we want to consider another resource, meaning memory consumption, due to our result of analysis, we can see that this time RSA is the worst scheme with a sharp distance with the other two. Now ElGamal performance is the best an ECDSA hast the second grade, figure 9.

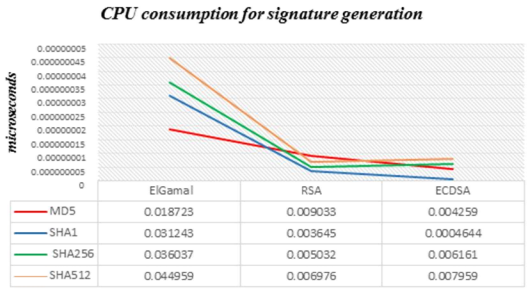

Fig. 6. CPU usage of signature schemes for signature generation

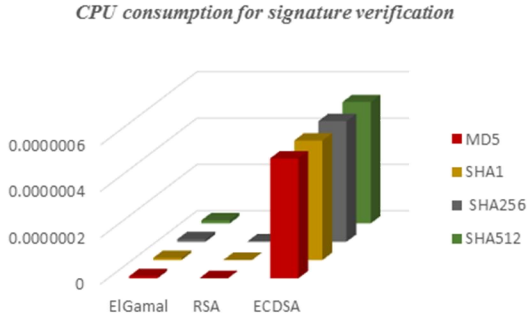

Fig. 7. CPU usage of signature schemes for signature verification

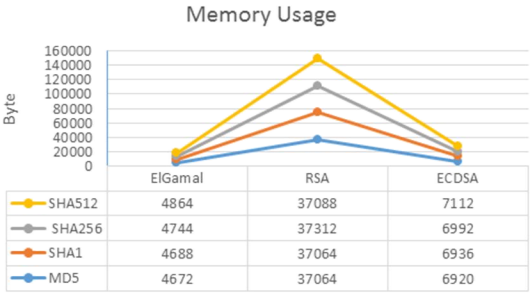

Fig. 8 . Memory usage of signature schemes

Another factor that we can consider for compare among these schemes is the length of public key that ECDSA has the shortest length [18], meaning lowest overload in network traffics which is shown in table 5.

Table 5: Length of public key in different schemes

scheme	Length of public key
ElGamal	1.25 KB
RSA	572 B
ECDSA	127 B

Overall, if we want to decide the best choice of digital signature schemes for web application we should summarize some important key point:

- RSA and ElGamal take sub-exponential time and ECDSA takes full exponential time.
- ECDSA offers same level of security with smaller key sizes [19] but DATA size for RSA is smaller than ECDSA and still faster than for ECC with same strength.
- ECC key size is relatively smaller than RSA key size and ElGamal, thus encrypted message in ECC is smaller.
- At the average computational power is smaller for ECDSA.
- ECDSA provides effective and compact implementations for cryptographic operations requiring smaller chips and Due to smaller chips less heat generation and less power consumption. Therefore ECDSA has easier hardware implementations [21] [22].
- one advantage of RSA is that its mathematics are somewhat simpler than those involved for elliptic curves , so many engineers feel that they "understand" RSA more than elliptic curves.

- One key that need to attend is trust; RSA has been around longer than EC, and people feel they understand it, and they trust it more (and in security, this is important). It's also easier to implement.
- And the last important key is length of public key which in ECDSA is shortest and therefore much shorter time in proceed.

So RSA is faster and easier to implement in front of complicated mathematic rules in ECC but ECC is more secure and the key length is much shorter. However the security level in ECC is quite high and infeasible to break up to know. Email application due to their security needed could attend any of these schemes which is implement and test quite in details.

6 CONCLUSION

This paper presents a security suggestion in email application using S/MIME. So we implement three popular digital signature schemes with different hash functions and measured their performance in time and memory. Our experiments show that Elliptic Curve Digital Signature algorithm (ECDSA) which is one of the variants of Elliptic Curve Cryptography (ECC) proposed as an alternative to established public key systems can be a good choice for this reason. The main point for the attractiveness of ECDSA is the fact that there is no sub exponential algorithm known to solve the elliptic curve discrete logarithm problem on a properly chosen elliptic curve. Hence, it takes full exponential time to solve while the best algorithm known for solving the underlying integer factorization for RSA which is uses by S/MIME and discrete logarithm problem in ElGamal both take sub exponential time. The key generated by the implementation is highly secured and it consumes lesser bandwidth because of small key size used by the elliptic curves 256 in front of 3072. There it seems that if email application wants to be more secure ECDSA is the best choice in unreliable environment like wireless. Combination of S/MIME and elliptic curve would guarantee the security in authentication process.

7 REFERENCES

[1] M. Toorani, "SMEmail-a new protocol for the secure e-mail in mobile environments." Telecommunication Networks and Applications Conference, 2008. ATNAC 2008. Australasian. IEEE, 2008.

[2] IETF S/MIME Working Group, http://www.ietf.org/html.charters/smime-charter.html

[3] RFC 5751: Secure/Multipurpose Internet Mail Extensions (S/MIME) Version 3.2 Message Specification, https://tools.ietf.org/html/rfc5751

[4] C. Moris,S. Smith, 'Towards usefully Secure E-mail,' IEEE technology and Society Magazine, 26(1), 25-34 (2007).

[5] A.Kapadia . 'A Case (Study) For Usability in Secure E-mail Communication,' IEEE Security & Privacy, 5(2), 80-84. 2007

[6] B.Leiba, J. Fento. 'DomainKeys Identified mails (DKIM):Using Digital signatures for Domain Verifcation,' CEAS 2007, Fourth Conference on E-mail and Anti-Spam August 2-3, 2007, Mountain View, California USA. 2007

[7] W. Trappe, L. C. Washington. "Introduction to cryptography with coding theory". Pearson Education India, 2006.

[8] T. ElGamal, "A public key cryptosystem and a signature scheme based on discrete logarithms." Advances in cryptology. Springer Berlin Heidelberg, 1985.

[9] Hankerson, Darrel, Alfred J. Menezes, and Scott Vanstone. Guide to elliptic curve cryptography. Springer Science & Business Media, 2006.

[10] A. Khalique, K. Singh, S. Sood. "Implementation of elliptic curve digital signature algorithm." International Journal of Computer Applications 2.2 (2010): pp- 21-27.

[11] Digital Signature Standard (DSS). Technical Report FIPS PUB 186-4, National Institute of Standards and Technology, July 2013.

[12] M. Zandstra, "PHP Objects, Patterns, and Practice", second edition, Ed. Apress, 2008.

[13] [NIST (2010a)] NIST. (2010) Fips 186-2. Digital signature standard (DSS) . [Online]. Available: http://csrc.nist.gov/publications/fips/archive/fips186-2/fips186-2.pdf

[14] NIST (2010b)]. (2010) Digital signature standard (dss). [Online]. Available:http://csrc.nist.gov/publications/fips/fips186-3/fips_186-3.pdf

[15] J. Bayuk, "Information security metrics–an audit-based approach." Computer System Security and Privacy Advisory Board (CSSPAB) Workshop on Security Metrics. 2000.

[16] V. Gupta, D. Stebila, S. Fung, S. C. Shantz, N. Gura, & H. Eberle, "Speeding up Secure Web Transactions Using Elliptic Curve Cryptography." In Proceedings of the 11th Annual Network and Distributed System Security Symposium. The Internet Society NDSS. 2004, 231-239.

[17] Y. C. Wang, H. X. Fang, Y. Xia. "Research and Application of Digital Signature Based on Elliptic Curve Cryptosystem*." Advances in Biomedical Engineering 8 (2012): 236.

[18] A. Thankachan, R. Ramakrishnan, & M. Kalaiarasi. "A survey and vital analysis of various state of the art solutions for web application security." Information Communication and Embedded Systems (ICICES), 2014 International Conference on. IEEE, 2014.

[19] A. Abidi, B. Bouallegue, F. Kahri. "Implementation of elliptic curve digital signature algorithm (ECDSA)." Computer & Information Technology (GSCIT), 2014 Global Summit on. IEEE, 2014.

[20] T. Ryutov, C. Neuman, K. Dongho, Z. Li, “Integrated access control and intrusion detection for web servers”, IEEE Transactions on Parallel and Distributed Systems 14 (9) (2003) 841–850.

[21] N. Koblitz, "Elliptic curve cryptosystems", Mathematics of Computation vo1.48 , pp.203-209, 1987.

[22] .Miller, "Use of elliptic curves in cryptography." Advances in Cryptology—CRYPTO’85 Proceedings. Springer Berlin/Heidelberg, 1986.

[23] AJ Menezes, PC Van Oorschot, SA Vanstone. “Handbook of applied cryptography”. CRC press, 1996.

[24] K. Jonathan, Y. Lindell. “Introduction to modern cryptography”. CRC Press, 2014.

[25] D. R. Stinson, “Cryptography: theory and practice”. CRC press, 2005.

[26] M. Danter, "An Elliptic Curve Digital Signature Algorithm and Diffie-Hellman Key Agreement Algorithm Implementation." (2010), available at http://www.matyasdanter.com/2010/12/elliptic-curve-php-oop-dsa-and-diffie-hellman/