OPTIMAL ANALYSIS OF A HYBRID AUTHENTICATION SYSTEM: WEB SECURITY VERSUS SYSTEM PERFORMANCE

Ben Soh, Aaron Joy
Applied Computing Research Institute
Department of Computer Science & Computer Engineering
La Trobe University
Bundoora, VIC, Australia 3083

Keywords: Authentication, Web Security

Abstract: A web authentication system uses a number of functions to provide integrity of messages sent between the client and the web server. These include hash functions, public key encryption, and nonce length. In this paper, we present an optimal analysis of investigating the effects of some different parameters on the web security and performance of the authentication system. Our main aim is to devise a technique to achieve the best of both worlds: optimal web security and system performance. To the best of our knowledge, such approach is the first attempt to combine the two dependability attributes in a quantitative study.

1 INTRODUCTION

Figure 1 shows the web server framework that we have developed at La Trobe University (A Joy et al., 2002). A secure channel established between a client and the web server is used to combat interception and modification attacks. However, to protect the client and web server against fabrication attacks (C. Kaufman et al., 1995), we need to incorporate into the secure web server framework two forms of authentication: (i) user-level authentication and (ii) machine-level authentication. User-level authentication is provided by one-time passwords (OTP) (A. Jones, 1981)(N.Haller et al., 1998), while machine-level authentication is provided by a hybrid authentication system, which is the subject of this paper.

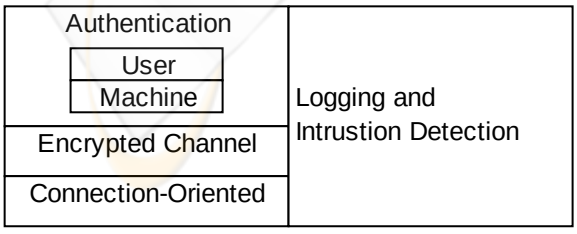

Figure 1: Full web server framework

The hybrid authentication system uses a number of functions to provide integrity of messages sent between the client and the web server. These include hash functions, public key encryption, and nonces (see Section 2). In this paper, we present a quantitative analysis of investigating the effects of various parameters on the security and performance of the hybrid authentication system. Our main aim is to devise a technique to achieve the best of both worlds: optimal web security and optimal system performance. To the best of our knowledge, this type of quantitative analysis in terms of the two dependability attributes is still lacking in the research community.

2 THE HYBRID AUTHENTICATION SYSTEM: BRIEF DESCRIPTION

The following is a semantic representation of the hybrid authentication system based on X.509 (C. I'Anson et al., 1990), where A is a client and B is an web server. Firstly, A generates a non-repeating number, r_A, which is used to detect replay attacks and to prevent forgery. A then sends the following message to B:

A → B: B $\{r_A$, Ap, sgnData$\}$,

where

Soh B. and Joy A. (2004).
OPTIMAL ANALYSIS OF A HYBRID AUTHENTICATION SYSTEM: WEB SECURITY VERSUS SYSTEM PERFORMANCE.
In *Proceedings of the First International Conference on E-Business and Telecommunication Networks*, pages 281-286
DOI: 10.5220/0001383502810286
Copyright © SciTePress

B {} = the encryption of {} using B's public key
r_A = non-repeating number
Ap = A's public key
SgnData = Digital signature.

The sgnData is for data origin authentication and is provided by a digital signature that involves the hash value of the data and is then encrypted using the user's private key. This provides verification that A and B are in possession of their appropriate private key. Once B receives the message it carries out the following actions:

- Obtains Ap
- Verifies the signature, and thus the integrity of the information.
- Checks that B itself is the intended recipient.
- Checks if r_A has not been replayed.

Once the above verification process is completed, B generates r_B, a non-repeating number similar to r_A. B then sends the following message to A :

B→A: A{r_B, Bp, r_A, sgnData}.

B can also provide data origin authentication by using sgnData which is included in the message. Once receiving the message, A will:

- Verify the signature, and thus the integrity of the information.
- Check that B itself is the intended recipient.
- Check if rB has not been replayed.
- Check that rA is the same rA it has sent.

If all the above criteria are satisfied, A sends B the final message

A→B: B{r_B,Bp}

On receiving this message, B checks that the same r_B was sent back and the message is meant for B. The final message is crucial for preventing a man-in-the-middle attack on the web server, by enabling the client and the web server to authenticate each other without the use of a timeserver.

3 PERFORMANCE AND SECURITY EVALUATION PARAMETERS OF THE HYBRID AUTHENTICATION SYSTEM

The dependability, in terms of system performance and security of our hybrid authentication system, depends on four main parameters: the type of hash function, the nonce length, the length of the random seed, and the number of random seeds used. These four parameters are discussed in the following subsections.

3.1 Hash Functions

A hash function is a "computationally efficient function mapping binary strings of arbitrary length to binary strings of some fixed length, called hash values" (A. Menezes et al.,1996). For a hash function to be of cryptographically secure, (i) the hash function *h* must be chosen so that it is computationally infeasible to find two distinct inputs which will hash to a common value, and (ii) given a specific hash-value say *y* it is computationally infeasible to find an input *x* such that *h(x)=y*.

The hash function is used in the hybrid authentication system to provide data integrity for the messages sent by the web server and the client. The information in the message is hashed and the resultant hash value is encrypted with the public key of the recipient. Our hybrid authentication system can accept any of the three types of hash functions: MD5 (R.Rivest,1992), RIPEMD-160 (B. Preneel et., 1997), and SHA-1 (SECURE HASH STANDARD, 1995). Each of these hash functions has its own specifications but the same underlying principles. They can all provide the hash value for the creation of a digital signature in the hybrid authentication system.

3.2 Nonce Length

A nonce is a set length of random bytes that the client and server send to each other. It contains a value used no more than once as an identifier to prevent (undetectable) replay attacks (A. Menezes et al.,1996). The nonce is a major part of the hybrid authentication system, by enabling the system to carry out its authentication function without the use of timeservers. To achieve this, the clients include a nonce in an outgoing message sent to the web server. Thus, any recipient of these messages requires the knowledge of this nonce. This enables the client and the web server to achieve a fixed point as their reference time during the message exchanges. This is analogous to a shared clock. The maximum time permitted for a message exchange is constrained by the timeout period, which somewhat enforces the use of local, independent clocks (A. Menezes et al.,1996). The three nonce sizes used in our hybrid authentication system are 100, 50, and 20 (bytes), thus giving sample spaces of 6.668×10^{240}, 2.582×10^{120}, and 1.461×10^{48}, respectively.

3.3 Random-Seed Length

The random-seed length is the size of the random data that will be placed into the array in the function *RAND_pseudo_bytes()* of the hybrid authentication system. This random data is then used by another function *RAND_seed()* to carry out the public-key data encryption. The three sample spaces used are 1000, 500, and 100 (bytes).

3.4 The Number of Random Seeds Used

The number of times the public–key encryption is seeded is given by the number of random seeds generated by the following code segment :

```
for(int j=0; j < NUMBER_OF_TIMES_SEEDED; j++)
{
        RAND_pseudo_bytes(rand_seed,
        KEYGEN_NUMBER_RAND_SEED);
        RAND_seed(rand_seed, sizeof rand_seed);
}
```

It is noteworthy that the code uses the *for()* loop so that the function *RAND_pseudo_bytes()* and *RAND_seed()* can be run more than once.

4 PERFORMANCE EVALUATION: EXPERIMENTAL SETUP AND RESULTS

The performance evaluation of the hybrid authentication system is based on the speed at which the client and server authenticate each other with the variation of the four parameters discussed in the previous section. The simulations were run on a Pentium (200MMX with 64mb of EDO ram) using Redhat Linux 7.0. The UNIX time command was used to obtain the time required for the hybrid authentication system to complete the full authentication task. The evaluation was based on the end-to-end authentication in that messages were passed from the server functions to the client functions without going through the communicational channel. This was done so that the results would not be affected by the channel delays between the client and the server.

Tables 1-4 are snapshots of the results obtained and respectively show the effects on the system performance of the hybrid authentication system caused by:

hash functions
the nonce length
the random-seed length
the number of random seeds used.

Table 1: Performance wrt hash functions

No of seeds used	Nonce length (bytes)	Seed length (bytes)	Average Performance Time (s) wrt hash functions		
			SHA-1	RIPEND-160	MD5
	100	1000	.3973	.3823	.3863
	50	500	.2999	.3010	.2980
	20	100	.2677	.2720	.2683
	100	100	.2707	.2637	.2647
	50	500	.2870	.2843	.2837
	20	1000	.3123	.3220	.3127
	100	500	.2737	.2750	.2727
	50	1000	.2883	.2847	.2850
	20	100	.2627	.2690	.2653

Table 2: Performance wrt the nonce length

No of seeds used	Hash Function	Seed length (bytes)	Average Performance Time (s) wrt nonce length		
			100	50	20
	SHA-1	1000	.3973	.3317	.3323
	RIPEND-160	500	.3127	.3010	.2987
	MD5	100	.2690	.2713	.2683
	SHA-1	100	.2707	.2667	.2657
	RIPEND-160	500	.2857	.2843	.2860
	MD5	1000	.3083	.3083	.3127
	SHA-1	500	.2737	.2790	.2730
	RIPEND-160	1000	.2850	.2847	.2983
	MD5	100	.2627	.2677	.2653

Table 3: Performance wrt the seed length

No of seeds used	Nonce length (bytes)	Hash function	Average Performance Time (s) wrt seed length		
			1000	500	100
	100	SHA-1	.3973	.2970	.2687
	50	RIPEND-160	.3473	.3010	.2677
	20	MD5	.3330	.2970	.2683
	100	SHA-1	.3110	.2840	.2707
	50	RIPEND-160	.3127	.2843	.2700
	20	MD5	.3127	.2853	.2727
	100	SHA-1	.2877	.2737	.2707
	50	RIPEND-160	.2847	.2777	.2627
	20	MD5	.2843	.2777	.2653

Table 4: Performance wrt the number of seeds

Hash functions	Nonce length (bytes)	Seed length (bytes)	Average Performance Time (s) wrt the number of seeds		
			3	2	1
SHA-1	100	1000	.3973	.3110	.2877
RIPEND-160	50	500	.3010	.2843	.2777
MD5	20	100	.2683	.2727	.2653
SHA-1	100	100	.2687	.2707	.2707
RIPEND-160	50	1000	.3473	.3127	.2847
MD5	20	500	.2970	.2a853	.2777
SHA-1	100	500	.2970	.2840	.2737
MD5	50	1000	.3323	.3083	.2850
RIPEND-160	20	100	.2720	.2713	.2690

Based on the results of the performance evaluation, we define for our hybrid authentication system a measure called *Performance Index (PI)* as the inverse of the average performance time. For example, for the hybrid authentication system using SHA-1 as the hash function, the nonce length of 100 bytes, the seed length of 1000 bytes and 3 random seeds, the *PI* is 1/0.3573 = 2.7988.

5 WEB SECURITY EVALUATION

As a matter of fact, quantitative measures for web security evaluation are yet to be agreed upon in the research community. Consequently we need some mechanisms to calibrate our hybrid authentication system from the security point of view. To this end, we utilize the industrial data and standards, expert knowledge and research outcomes in the field, within the sample space of the four parameters described in Section 3. In the sample space of the parameter concerned, a parameter value deemed to offer the maximum security will be given a weight of 1; the weight is otherwise less than 1 (which we will discuss below). For our hybrid authentication system, we define a measure called *Security Index (SI)* as the average weight of the 4 parameters:

SI = (hash_function_weight + nonce_length_weight + seed_length_weight + number_of_seeds_weight) / 4

where hash_function_weight etc are self-explanatory. This calibration would not lead to an absolute measure for security evaluation but would be useful as a tool for comparative and sensitivity studies by varying the values of those parameters of the hybrid authentication system.

For our hybrid authentication system, the sample space of the hash function parameter is {MD5, RIPEND-160, SHA-1}. Since the collision method found by Paul Oorschot and Michael Wiener (R. M. Needham, 1993), MD5 and RIPEMD-160 have been found to contain weaknesses. In particular, the CryptoBytes newsletter (B. Preneel et., 1997) best described MD5's situation: "The presented attack does not yet threaten practical applications using MD5, but it comes rather close". Also, Paul Van Oorschot and Mike Wiener (C. Kaufman et al., 1995) showed that a brute-force job on MD5 can be done in less than a month…" To date, SHA-1 has been found to be the most secure hash function among the three. Based on the expert knowledge, we consider within the sample space that SHA-1 has a *hash_funciton_weight* of 1. Using the 30/70 rule for a comparative study, we assign 0.43 to the *hash_function_weight* of RIPEND-160 and 0.18 to MD5, which is deemed the weakest among the three.

The nonce is an important factor for authentication in that the larger the nonce the less chance will the nonce be predicted and compromised by an attacker. Our *nonce_length* space is {100, 50, 20}. On this basis, 100 bytes of nonce is given a weight of 1. Using the rule of proportionality for a comparative study, 50 bytes of nonce is given a weight of 0.5, and 20 bytes of nonce a weight of 0.2.

For the random-seed length, the sample space used is {1000, 500, 100}. From a security point of view, the larger the random data seeded the more random is the public-key encryption. This implies that it will be harder for an attacker to find a pattern (National Institute Of Standards and Technology, 2001). Hence, 1000 bytes of seed length has a weight of 1, 500 bytes a weight of 0.5, and 100 bytes a weight of 0.1 (by proportioning).

The sample space for the number of random seeds used is {3, 2, 1}. The number of seeds determines the number of times the public-key encryption is seeded. If data are seeded more frequently, they will tend to be more random, which would be harder for an attacker to compromise. Thus, a weight of 1 is given if the number of seeds used is 3. By proportioning, a weight of 0.67 and 0.33 is given if the number of seeds used is 2 and 1, respectively.

6 OPTIMAL WEB SECURITY-AND-PERFORMANCE INDEX FOR THE HYBRID AUTHENTICATION SYSTEM

From the above discussions, it is obvious that performance and security are two opposing dependability attributes. For example, if the random seed length is excessive (which is good for security reasons), system performance will be reduced tremendously, as mentioned in (D. Eastlake et al.,). This points to the need for a compromise between performance and security for our hybrid authentication system. To this end, using *Performance Index (PI)* and *Security Index (SI)* defined in Sections 4 and 5 respectively, we propose a composite index called *Authentication System Index (ASI)*, which is defined as follows:

$$ASI = PI \times SI.$$

Using the *ASI*, we are able to find an optimal set of the parameters used in our hybrid authentication system, in terms of both security and system performance. Table 5 gives a snapshot of the results involving *ASI*. By analysing all the data (not included here because of space constraints), we are able to list down the following observations for our hybrid authentication system:

1. The random-seed size of 1000 bytes is too large, which reduces the *ASI* value. A seed size of 500 (or 100) bytes may give optimal *ASI*.
2. The nonce size of 20 bytes is too small. A size of 100 (or 50) bytes may produce optimal *ASI*.
3. For optimal *ASI*, the number of random seeds used can be just one.

7 CONCLUSION

In this paper, we have presented a quantitative analysis on the web security and system performance evaluation of an actual hybrid authentication system. We have proposed a composite index called Authentication System Index (*ASI*), which can be used to evaluate at the same time both the security and performance of our hybrid authentication system.

From the results obtained, we have found that: (i) a nonce size of either 50 or 100 bytes is fine for optimal security and system performance; this concurs with the findings presented in (D. Eastlake et al.,); (ii) it is claimed in (D. Eastlake et al.,) that 1000 bytes for the size of the seeds used seem too large, and our study in this paper proves this point; our results show 100 (or 500) bytes is a good seed size to be used for our authentication system, and (iii) the number of seeds used (i.e. frequency of seeding the public-key encryption) can be just once, which would give sufficient randomness.

REFERENCES

A Joy and B Soh, "A proposed secure TCP connection-oriented model for e-commerce systems," *Proceedings of International Conference on Internet and Multimedia Systems and Applications*, Hawaii, Aug 12-14, pp 68-73, 2002.

C. Kaufman, R. Perlman, and M. Speciner. *Network Security Private Communications in a Public World*, Prentice Hall 1995.

A. Jones, "Password authentication with insecure communication," *ACM Communications*, vol. 24, number 11, 1981, pp 12-21.

N.Haller, C. Metz, P. Nesser, and M. Straw . "A One-Time Password System," *RFC2289*, February 1998, http://www.ietf.org.

C. I'Anson and C. Mitchell. "Security defects in CCITT recommendation X.509 – The directory authentication framework," *Computer Communications Review*, pp 45-53, April 1990.

A. Menezes, P. van Oorschot and S. Vanstone, *Handbook of Applied Cryptography*, CRC Press, 1996.

R.Rivest, "The MD5 Message-Digest Algorithm", *Internet RFC 1321*, April 1992.

B. Preneel, and A. Bosselaers, "The Cryptographic Hash Function RIPEMD-160", *CryptoBytes*, vol 3, no 2, Autumn 1997.

National Institute of Standards and Technology, *SECURE HASH STANDARD, FIPS PUB 180-1*, 1995. February 13 2001 http://www.itl.nist.gov/fipspubs/fip180-1.htm.

R. M. Needham , "Denial Of Service," *Proceedings of the 1st ACM conference on Computer and Communications Security*, pp 151-153, November 3-5, 1993, Fairfax, VA USA.

National Institute Of Standards and Technology, *FIPS 112 – Password Usage*. 1995. September 2001, http://www.itl.nist.gov/fipspubs/fip112.htm.

D. Eastlake, S Crocker, and J Schiller, "Randomness Recommendations for Security", *rfc1750*, http://www.i

APPENDIX

Table 5: Performance versus Security

No of seeds used	Nonce length (bytes)	Seed length (bytes)	Hash functn	PI	SI	ASI
3	100	1000	SHA-1	2.5170	1.0000	2.5170
3	50	500	SHA-1	3.3344	0.7500	2.5008
3	20	100	SHA-1	3.7355	0.5750	2.1479
2	100	100	SHA-1	3.6941	0.6925	2.5582
2	50	500	SHA-1	3.4843	0.6675	2.3258
2	20	1000	SHA-1	3.2020	0.7175	2.2974
1	100	500	SHA-1	3.6536	0.7075	2.5849
1	50	1000	SHA-1	3.4686	0.7075	2.4469
1	20	100	SHA-1	3.8066	0.4075	1.5512
3	100	1000	RIPEND-160	2.6157	0.8575	2.2430
3	50	500	RIPEND-160	3.3223	0.6075	2.0183
3	20	100	RIPEND-160	3.6765	0.4325	1.5901
2	100	100	RIPEND-160	3.7922	0.5500	2.0857
2	50	500	RIPEND-160	3.5174	0.5250	1.8467
2	20	1000	RIPEND-160	3.1056	0.5750	1.7857
1	100	500	RIPEND-160	3.6363	0.5650	2.0545
1	50	1000	RIPEND-160	3.5125	0.5650	1.9846
1	20	100	RIPEND-160	3.7175	0.2650	0.9851
3	100	1000	MD5	2.5887	0.7950	2.0580
3	50	500	MD5	3.3557	0.5450	1.8289
3	20	100	MD5	3.7272	0.3700	1.3791
2	100	100	MD5	3.7779	0.4875	1.8417
2	50	500	MD5	3.5249	0.4625	1.6302
2	20	1000	MD5	3.1980	0.5125	1.6390
1	100	500	MD5	3.6670	0.5025	1.8427
1	50	1000	MD5	3.5088	0.5025	1.7631
1	20	100	MD5	3.7693	0.2025	0.7632