http://ansinet.com/itj

ITJ

ISSN 1812-5638

INFORMATION TECHNOLOGY JOURNAL

ANSInet
Asian Network for Scientific Information
308 Lasani Town, Sargodha Road, Faisalabad - Pakistan

Information Technology Journal 12 (3): 522-525, 2013
ISSN 1812-5638 / DOI: 10.3923/itj.2013.522.525

Security Analysis of Practical Anonymous User Authentication Scheme with Security Proof

[1]Chenglian Liu, [2]Changlu Lin and [1]Shuliang Sun
[1]Fuqing Branch of Fujian Normal University, Fuqing, 350300, China
[2]Fujian Normal University, Fuzhou, 350007, China

Abstract: Chien proposed a practical anonymous user authentication scheme with security proof in 2008. Even he used bitwise exclusive-or to against multiplicative attack and the exclusive-or implants easier and faster. But he may have misused order of operation in mathematical precedence. In this article, we would like to point out these errors in the related work and scheme.

Key words: Digital signature, bitwise exclusive-or operation, mathematical precedence

INTRODUCTION

Most bitwise exclusive-or operation used in cryptographic protocol. But the designer does not mention it may a very dangerous if they misused this function. Chien (2008) proposed a practical anonymous user authentication scheme with security proof in 2008. However, Chien might also misuse mathematical precedence properties in computer programming which logical exclusive-or combine string concatenation operation for their scheme. A sample of paper (Zhang and Wang, 2005; Xu *et al*., 2010) where it also misused same situation actually.

BRIEFLY CHIEN'S SCHEME

In Chien (2008) review some hard problems and then propose a practical anonymous user authentication scheme with security. The notation and definitions are same in review of Chien (2008).

In this commonly believed that there is no polynomial-time algorithm to solve FACP, DLP_N, CDHP_N or DDHP_N with non-negligible probability (Girault, 1991). Based on the FAC problem and the CDHP_N problem, we propose the two-party key agreement scheme with client anonymity as follows. Our proposed scheme consists of two phases: the key generation phase and the anonymous user identification phase.

Key generation: The SCPC chooses two large safe prime p and q computes:

$$N = pq \tag{1}$$

selects e and d computes such that:

$$cd \equiv 1 \ (\text{mod } \phi(N)) \tag{2}$$

Where:

$$\phi(N) = (p-1)\,(q-1) \tag{3}$$

It chooses a generator g which is a primitive root for both $\mathbb{Z}_p^*$ and $\mathbb{Z}_q^*$, a symmetric-key cryptosystem $E_k(\ )$ (such as AES), three cryptographic one-way hash functions:

$$H_1: \{0, 1\}^* \rightarrow \mathbb{Z}_N^*,\ H_2: \{0, 1\}^w \rightarrow \{0, 1\}^w$$

and:

$$H_3: \{0, 1\}^{|N|} \rightarrow \{0, 1\}^l$$

where, w is a public parameter such that $w > \phi(N)$ and l is the key length for the symmetric encryption scheme. The encryption function and the decryption function under the secret key K are denoted as $E_k(\ )$ and $D_k(\ )$, respectively. This encryption scheme should satisfy the indistinguishability under chosen plain text attack (IND-CPA) property. The SCPC then publishes parameters $\{N, e, g, w, E(\ ), D(\ ), H_1, H_2, H_3\}$ as public parameters and keeps {d, p} and {q} private. Finally, through a secure channel, the SCPC sends to each registered entity U_i (which could be a client with identity C_i or a server with identity P_i) a secret token:

$$S_{C_i} \equiv H_{C_i} \ (\text{mod } N) \tag{4}$$

Where:

$$H_{C_i} = H_1\,(C_i) \tag{5}$$

Corresponding Author: Chenglian Liu, Department of Mathematics and Computer Science, Fuqing Branch of Fujian Normal University, Fuqing, 250300, China

If U_i is a client C_i; or, a secret token:

$$S_{P_i} \equiv H_{P_i} \pmod N \tag{6}$$

Where:

$$H_{P_i} = H_1 (P_i) \tag{7}$$

If U_i is a server P_i. In the following, sid denotes the unique session identifier of the current session.

Anonymous user identification: To request a service from the server P_i, the client C_i sends the request to P_i. Upon receiving the request, P_i chooses a random number k, compute:

$$z \equiv g^k S_{P_i} \pmod N \tag{8}$$

and sends z to C_i. After receiving z, C_i chooses a random number t and computes the following values:

$$m \equiv ze/H_{P_i} \pmod N \tag{9}$$

$$r \equiv m^t \pmod N \tag{10}$$

$$k_{sess} = H_3 \tag{11}$$

$$x = g^{et} \pmod N \tag{12}$$

$$s \equiv S_{C_i}^{H_2(sid\|x\|T)} \pmod N \tag{13}$$

$$y \equiv E_{k_{sess}}(C_i \oplus r\|s \oplus r) \tag{14}$$

where, T is the current timestamp. C_i then sends (x, y, T) to P_i. Upon receiving the message, P_i first checks the validity of timestamp T by checking whether the timestamp is fresh and is within valid time window (Fig. 1). If so, P_i further computes:

$$r \equiv x^k \pmod N \tag{15}$$

and:

$$k_{sess} = H_3(r) \tag{16}$$

Using the key, decrypts to derive:

$$C_i\|s \leftarrow C_i \oplus r\|s \oplus r \leftarrow d_{k_{sess}}(y) \tag{17}$$

computes:

$$H_{C_i} = H_1 (C_i) \tag{18}$$

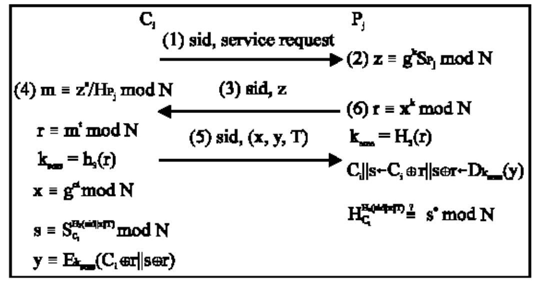

Fig. 1: Chien's Scheme (Chien, 2008)

and verifies whether the equation:

$$H_{C_i}^{H_2(sid\|x\|T)} \stackrel{?}{\equiv} s^e \pmod N \tag{19}$$

If all the verifications succeed, then the request is granted; otherwise, the request is rejected. Finally, the client and the server share one common session key and the arrangement of:

$$E_{k_{sess}} (C_i \oplus r\|s \oplus r) \tag{20}$$

is to ensure the indistinguishability in the formal model. It is easy to show the correctness of the verification equation:

$$H_{C_i}^{H_2(sid\|x\|T)} \stackrel{?}{\equiv} s^e \pmod N \tag{21}$$

as follows:

$$\begin{aligned} s^e &\equiv S_{C_i}^{eH_2(sid\|x\|T)} \\ &\equiv H_{C_i}^{deH_2(sid\|x\|T)} \\ &\equiv H_{C_i}^{H_2(sid\|x\|T)} \pmod N \end{aligned} \tag{22}$$

OUR SECURITY ANALYSIS

Here, we introduce two point views, one is logical bitwise exclusive-or operation, the other one is order of operations in computer system. We introduce the precedence properties in some programming languages.

Order of operation: For example, the bitwise exclusive-or is in level 9. However, there is no string concatenation operator in C/C++ directly. Although, the C/C++ provides some libraries such as strcat(), sprintf() and so on functions to connect string but it still can not combine strings with numbers (Table 1).

For Java programming language, the string concatenation is in level 5 and the bitwise exclusive-or is in level 10 (Table 2). The string concatenation therefore is

Table 1: Operator Precedence in C/C++

Level	Operator	Description
1	() [] - >:	Grouping, scope, array/member access
2	! ~ - + * & size of type cast ++x --x	(most) unary operations, sizeof and type casts
3	* /%	Multiplication, division, modulo
4	+ -	Addition and subtraction
5	<< >>	Bitwise shift left and right
6	< <= > >=	Comparisons: less-than, . . .
7	== =!	Comparisons: equal and not equal
8	&	Bitwise AND
9	^	Bitwise exclusive OR
10	|	Bitwise inclusive (normal) OR
111	&&	Logical AND
12	||	Logical OR
13	?:	Conditional expression (ternary operator)
14	= += -= *= /= %= & = |= ^= <<= >>=	Assignment operators
15		Comma operator

Table 2: Operation precedence in Java (Sedgewick and Wayne, 2008)

Operation	Description	Level	Associativity
[]	Access array element	1	Left to right
.	Access object member		
()	Invoke a method		
++	Post-increment		
-	Post-decrement		
++	Pre-increment	2	Left to right
-	Pre-decrement		
+	Unary plus		
-	Unary minus		
!	Logical NOT		
~	Bitwise NOT		
()	Cast	3	Left to right
New	Object creation		
*	Multiplicative	4	Left to right
/			
%			
+-	Additive	5	Left to right
+	String concatenation		
<< >>	Shift	6	Left to right
>>>>			
< <=	Relational type comparison	7	Left to right
>=			
instance of			
= =	Equality	8	Left to right
!=			
&	Bitwise AND	9	Left to right
^	Bitwise XOR	10	Left to right
|	Bitwise OR	11	Left to right
&&	Conditional AND	12	Left to right
||	Conditional OR	13	Left to right
?:	Conditional	14	Left to right
= += -= *= /=			
%= &= ^= |=	Assignment	15	Left to right
<<= >>= >>>=			

higher precedence than bitwise exclusive-or as known. The Chien's scheme does not appear to be true, as pointed out below. According to Sedgewick and Wayne (2008) and Kruse and Ryba (1999), the string concatenate has higher precedence than bitwise exclusive-or. Therefore the string concatenate should be applied first and then to process bitwise exclusive-or operation in most parts of computer programming such as Java, Visual Basic (Microsoft, 2011) and so on; this rule is known as a precedence rule or order of operation.

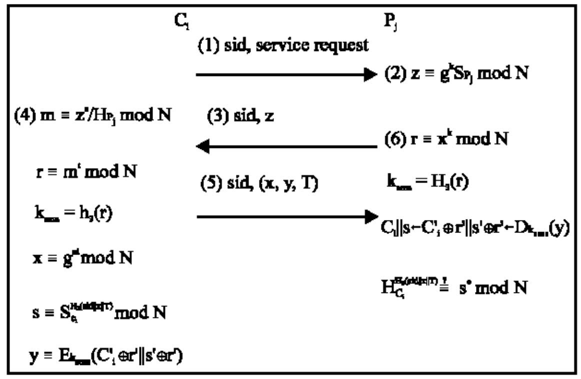

Fig. 2: Our attack

According to Table 1, the Eq. 14 becomes to:

$$y \equiv E_{k_{sess}}[C_i \oplus (r \| s) \oplus r] \tag{23}$$

It therefore:

$$y \equiv E_{k_{sess}}(C_i \oplus r \| s \oplus r) \neq E_{k_{sess}}[C_i \oplus (r \| s) \oplus \tag{24}$$

Correction Eq. 14 is used instead of equation:

$$y \equiv E_{k_{sess}}[(C_i \oplus r) \| (s \oplus r)] \tag{25}$$

Then, the Eq. 17 must change to:

$$C_i \| s \leftarrow [(C_i \oplus r) \| (s \oplus r) \leftarrow d_{k_{sess}}(y) \tag{26}$$

So, the Eq. 20 become to:

$$E_{k_{sess}}[(C_i \oplus r) \| (s \oplus r)] \tag{27}$$

Our attack: The attack (Fig. 2) can forge a valid parameter (r, s) where $r \oplus s \stackrel{?}{=} (-r \oplus -s)$ (Liu *et al*., 2012). He does follow stops:

- **Step 1:** Attack sets r' = 'r
- **Step 2:** Attack sets s' = -s
- **Step 3:** Attack sets $C_i'' = -C_i$

CONCLUSION

In general, the string concatenate operator is always higher than bitwise logical exclusive-or operation and if a designer or developer misused or misunderstood this situation, it may cause a dangerous problem. We clearly described an example of this case in the paper. On the other hand, user used XOR operation in two's complement number system, it may cause others dangerous problem. Thus, the Chie's scheme is insecure. From previous section, we confirm our assumption.

ACKNOWLEDGMENTS

This study is partially supported by the National Natural Science Foundation of China under Grant No. 61103247 and the Natural Science Foundation of Fujian Province under Grant No. 2011J05147.

REFERENCES

Chien, H.Y., 2008. Practical anonymous user authentication scheme with security proof. Comput. Secur., 27: 216-223.

Girault, M., 1991. An identity-based identification scheme based on discrete logarithms modulo a composite number. Adv. Cryptol., 473: 481-486.

Kruse, R.L. and A.J. Ryba, 1999. Data Structures and Program Design in C++. Prentice Hall Inc., Englewood Cliffs, USA., ISBN: 13-9780137689958, Pages: 717.

Liu, C., C. Chen and S. Sun, 2012. Security analysis of mutual authentication and key exchange for low power wireless communications. Proceedings of the International Conference on Future Electrical Power and Energy, April 12-13, 2012, Hong Kong, pp: 644-649.

Microsoft, 2011. Operator precedence in visual basic. Microsoft MSDN. http://msdn.microsoft.com/en-us/library/fw84t893.aspx.

Sedgewick, R. and K.D. Wayne, 2008. Introduction to Programming in Java: An Interdisciplinary Approach. Pearson Addison-Wesley, Boston, MA., USA., ISBN: 13-9780321498052, Pages: 723.

Xu, L., C. Liu and N. Wang, 2010. Comment on an improved signature without using one-way hash functions. Proceedings of the International Conference on Cyber-Enabled Distributed Computing and Knowledge Discovery (CyberC), October 10-12, 2010, Huangshan, pp: 441-443.

Zhang, J. and Y. Wang, 2005. An improved signature scheme without using one-way hash functions. Applied Math. Comput., 170: 905-908.