비밀키 교환이 없는 One Time Password

[1]정윤화, [*2]박용범
[1] 단국대학교, jinshill99@gmail.com
[*2] 단국대학교, ybpark@dankook.ac.kr

OTP without a Secret key Exchange

[1]Yunhwa Chong, [*2]Young B. Park
[1]Dankook University, jinshill99@gmail.com
[*2]Dankook University, ybpark@dankook.ac.kr

요 약

제도의 변화와 다양한 네트웍 애플리케이션의 등장으로 간편하면서도 안전한 보안 제품에 대한 요구가 증가 하였다. 공인 인증서는 정기적으로 비밀번호의 변경등 갱신이 요하였고 OTP 단말기는 소지가 불편하였다. 이런 사용성의 문제를 해결 하기 위해 소프트웨어적으로 머클트리를 이용한 One time password 를 제안한다. 서버와 클라이언트에 각각 메시지나 거래데이터를 머클트리에 저장하고 그 authentication path 를 OTP 의 비밀키로하여 one time password 를 만들어서 서버와 클라이언트의 OTP 를 비교하여 사용자를 인증한다. 소프트웨어적으로 구현하여서 단말기가 따로 필요없고, 머클트리의 authentication path 를 이용하며 거래의 무결성을 해결하였다. 따라서 키의 교환이나 입력이 필요없어서 사용성이 높아졌다.

Abstract

There have been demands on simple and secure security as the advent of various network applications after regulation change of Certified Certificate and OTP terminals. Carrying terminals makes OTP usage difficult. Regular updates of password and other management issues prevent the wide use of Certified Certificates. This paper proposes Software type of One Time Password based on Merkle Tree authentication path to resolve such issues. The authentication path of Merkle Tree is used as an input of OTP in both server and client. This way enables Keyless authentication thus key management issues are resolved, also usability is enhanced.

Keywords: *Secure hash function, Merkle tree, authentication path, OTP, authentication.*

1. 서론

2015 년 금융위원회 전자금융 감독 규정[1] 이 개정되어서 전자상거래에서 공인인증서 사용을 의무화한 것을 폐지 하게 되었고 일회용 비밀번호(OTP)의 구현 방법을 규제하였으나, 이 부분도 개선되어 다양한 OTP 기술을 도입할 수 있게 되었다. 또한 방화벽, 키보드보안, 백신 등 보안 모듈 등의 설치 의무도 폐지되었다. 보안규제의 폐지로 위의 기능들을 대체하지만 네트웍의 발달과 강화된 컴퓨터환경, 이에 비례해 더욱 커지는

* Corresponding Author
Received: Aug. 30, 2017, Revised: Sep. 22, 2017, Accepted: Sep. 28, 2017

정보누출의 위험으로부터 정보와 자산을 보호할 수 있는 안전한 보안 기능에 대한 요구도 증가 하였다. 특히, 핀테크 활성화와 맞물려 간편 송금도 도입 되었으며 기존 금융권에서 쓰는 전통적인 보안 모듈을 대체하는 안전한 거래에 대한 다양한 보안 요구사항도도 늘어났다 단말기용 OTP 의 경우 가지고 다니기가 불편한 점뿐만 아니라 금융거래에서 단말기형 OTP 로 제한 하므로써 다양한 방식의 OTP 솔류션이 이용되지 못하고 있었다.

최근들어 일회용 비밀번호를 이용한 다양한 보안 솔류션이 시도되고있다.[13,15,16]. 본 논문에서는 서버와 단말기 간, 주로 스마트폰에서 안전한 해쉬 함수 (SecureHash Function)를 전제로 하고 있는 머클 트리를 각각 빌드 하여 고유한 메시지나 트랜잭션의 Authentication Path 를 입력 값으로 하는 소프트웨어 타입의 OTP(One Time Password)값을 일회용 비밀번호로 생성하고 거래의 인증여부를 결정하는 방식을 제안 하게 되었다. 스마트폰등 단말기와 서버간에 OTP 값으로 동기를 맞춤으로서 승인 여부를 결정 하게 되고 OTP 단말기 없이도 비밀번호를 동기화하고 사용`자 입장에서는 별도로 키를 입력하지 않고도 간편하게 거래를 인증 할 수 있게 되었다. 본 논문에서는 간편 송금에 적용하여 예를 들었다.

2. 관련 연구

2.1. 안전한 해쉬함수

안전한 해시 함수(secure hash function)[2][3]는 임의의 길이의 데이터를 고정된 길이의 데이터로 축약하는 단 방향 함수이며, preimage 를 추론 할 수 없어야 한다. 안전한 함수는 다음과 같은 특징을 지닌다. Preimage-resistance -최초 해수함수 아웃풋에 대하여 인풋값을 추론할 수 없어야 한다. h(x')= Y, 해쉬값 Y 에 대해서 x' 값을 추론 할 수 없어야 한다. 2nd-Preimage resitance-특정 해쉬 결과값에 대해서 입력 값이 다른 두 번째 입력 값을 계산 할 수 없어야 한다. h(x')=h(x) 일때, x 에 대하여 $x \neq x'$ 인 x'를 계산할 수 없어야 한다. Collision resistance(충돌회피성)-서로 다른 입력 데이터에 대해 같은 해쉬 값이 나와서 충돌하면 안되는 충돌 회피성(collision resistance)를 가져야한다. 해쉬 함수는 데이터 무결성과 디지털 서명이나 트랜잭션(메시지) 인증, 사용자 인증 등 다양한 분야에서 사용되고 있으며[14], SHA 256, SHA 384, SHA 512 를 쓰며 이외에도 여러 종류가 있다.

2.2. TOTP (Time-Based One Time Password)

일회용 비밀번호로써 사용자인증에 사용되며 민감하고 높은 수준의 보안을 필요로할 때 이용된다. 한번 생성된 키는 다시 사용하지 않으며 이미 사용된 키를 재사용하지 못하므로 공격영역을 줄여준다 또한 주어진 시간 간격마다 번호가 계속 바뀌므로 번호를 가로채기하기 어렵다 현재 시간과 서버와 클라언트만 아는 비밀키의 해쉬를 이용하여 일회용 비밀번호를 계산한다. 네트웍 지연이나 시간이 동기화가 되지 않아서 검증하는데 어려움이 있었으나 시간 스탬프를 30 초 정도로 증가 시킴 으로서 효율적으로 검증 할 수 있게 하였다.

TOTP[4][5]는 다음과 같이 정의한다.

$$TC = floor((current\ unix\ time) - unixtime(T0)) / X$$

$$TOTP = HOTP(secretKey, TC)$$

$$TOTP\text{-}Value = TOTP \bmod 10^d,$$

d 는 일회용 비밀번호, TOTP의 결과값 자리의수

T0 는 time step을 카운트하기 시작하는 시간

X 는 time step 의 단위, 초 단위 (디폴트는 30)

Secret key는 임의의 수면서 클라이언트와 서버만 아는 비밀키 값이 된다.

2.3. 머클 트리 [6][7][8][9][10] [Mer79]

머클 트리는 머클 서명 체계에서 제안된 해쉬 트리로 안전한 해쉬 함수에 그 기반을 둔다. 안전한 해쉬 함수는 역상공격(preimage attack)에 안전 해야 하며 충돌 회피성(collision resistance)을 가지고 있어야 한다. 머클트리는 트랜잭션(메시지)등의 무결성을 검증하는데 사용 된다. 머클 트리는 바이너리 트리이며, 이 트리를 MT 라고하고 높이가 H 라고 했을때 2^H의 리프를 가진다. 이너노드(inner node)는 2^H -1 개가 된다. 각 이너노드는 “0” (왼쪽)과 “1” (오른쪽)의 두 Children nodes 를 갖는다. K 개의 스트링을 갖는 각 노드 n 은, n{0,1}k 를 나타내며 부모 노드 값은 자식 노드의 일 방향 함수값이 된다. P(nparent) = hash (P(nleft) ||P(nright)) 해쉬 함수에는 SHA 256, SHA 384,SHA 512 등 여러 해쉬함수가 있다. 리프의 프리이미지는 이번 제안 에서는 트랜잭션(메시지)의 해쉬 값이 된다. 이 리프의 형제의 해쉬 값을 합쳐서 해쉬 하고 해쉬를 계속해나가면서 트리의 꼭대기에 혼자 위치하는 노드가 루트가되며 루트에 도달하면 이 루트가 모든 리프(트랜잭션의 해쉬값) 의 무결성을 검증하는 해쉬 값이 된다.

2.4. Authentication Path [11][12]

특정 리프에서 루트로 가는 path 중에, 높이 h 노드의 형제 노드의 값이 Authentication path 가 되고 $Auth_h$ 로 놓는다. 모든 리프는 높이가 0 이고, 두 리프의 해쉬 값은 높이 1 을 가지게 된다. 만약 루트의 높이가 H 라면 트리는 $2^H = N$ 개의 리프를 가지게 된다. 모든 리프의 Authentication path 는 set{ $Auth_h$ | 0 <= h < H } 이다. 그림 [1]에서처럼 만약 검증하려는 leaf 가 $A_{0,5}$ 라하면 그것의 형제인 $A_{0,4}$ 가 $Auth_0$ 이되고 이들을 해쉬한 값을 $A_{1,2}$ 이라하면 $A_{1,2}$ 의 형제는 $A_{1,3}$ 가되고 $Auth_1$ 이 된다. $A_{2,1}$ 의 형제는 A_{20} 이되고 $Auth_3$ 이된다. 이렇게 루트까지 해쉬를 반복하고 계산된 루트값과 공개된 루트값이 같으면 리프가 정당함을 검증받을수있다. 특정 리프의 무결성을 검증 받기위해 리프로부터 루트 해쉬값을 다시 생성하는데 필요한 노드인 Authentication Path 를 구성 하는 데는 N 개의 리프로 이루어진 바이너리 트리에 대하여 $Log_2(N)$개의 해쉬 계산이 필요하다. 즉 리프가 8 이면 3 개의 해쉬 계산이 필요하다.

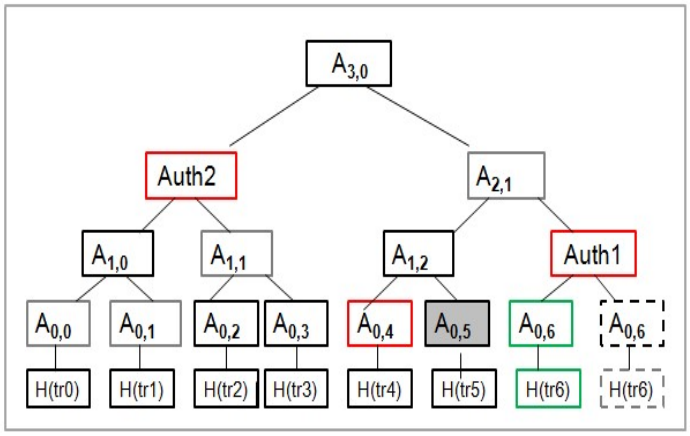

Figure 1. Authentication Path.

3. 제안 방안

거래단말기에서 메시지나 트랜잭션을 생성하고 무결성을 유지할수있는 머클트리에 저장한후 이전거래의 authentication path 를 구하고 이를 현재 시간과 함께 OTP 의 비밀키값으로 사용한다. OTP 를 생성하고 이 OTP 와 현재 트랜잭션과함께 서버에 보내면 서버는 트랜잭션과 OTP 를 받고 자신의 머클트리의 이전 트랜잭션의 authentication path 를 구하여 OTP 를 생성하고 클라이언트에서온 OTP 값과 비교하여 거래의 승인이나 거부 메시지를 보낸다. OTP 값이 동일하면 거래 승인 메시지를 내보내고 자신의 머클트리에 트랜잭션을 추가하고 루트를 갱신한다. 새로운 거래 생성시 매번 고유한 머클트리의 이전거래의 authentication path 를 이용하므로 사용자를 확인할수있는 요소가 되며 이를 OTP 키값으로 사용한다. 서버는 클라이언트에서 보내온 트랜잭션을 서버의 머클트리에 저장하게되므로 클라이언트나 서버나 같은 트랜잭션을 각각의 머클트리에 저장하게 되므로 같은 루트값과, 같은 authentication path 를 가지게된다. 이전 거래의 authentication path 를 입력값으로하는 서버와 클라이언트의 OTP 는 역시 같은 값을 가지게 된다. 구체적으로 트랜잭션(메세지) 생성과 머클트리 빌드, 이전거래의 authentication path 생성, OTP 생성, OTP verification 은 다음과 같다.

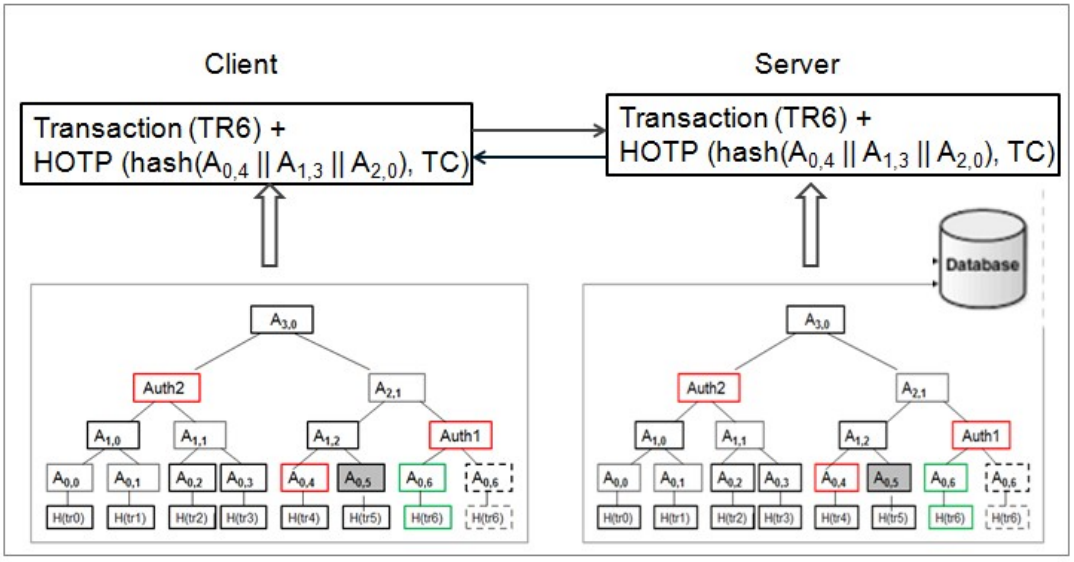

Figure 2. Authentication path generation in client and server.

3.1. 트랜잭션(메시지) 생성

트랜잭션이나 메시지를 생성해야하나 여기서는 PRNG 를 이용하여 트랜잭션을 생성한다. 이렇게 생성된 트랜잭션은 머클트리의 리프의 pre-image 가된다.

3.2. 머클트리 빌드 와 이전거래의 authentication path 생성

클라이언트에서 트랜잭션이 생성되면 클라이언트는 머클트리를 빌드하는데 머클 트리 MT 의 리프를 포함한 전체 노드수는 2^{H+1} -1= 2N 개의 노드를 가진 트리를 빌드할수있다. 높이 H 에 트리 루트가 존재한다. 트랜잭션이 추가될때마다 기존의 트리에 리프로 추가되며 새로운 루트가 계산된다. 그림 [2] 는 트랜잭션 6 번이 생성될 때 트랜잭션 5 번의

Authentication Path 를 참조하는 것을 보여준다. 트랜잭션 5 번 이 생성될 때는 $A_{0,4}$ 가 형제가 되고 $Auth_0$ 가되며, Authentication Path 를 구성할 $A_{1,2}$.의 형제가 없어서 트랜잭션 5 번을 복사해서 $A_{0,6}$, $A_{0,7}$ 리프를 만들고 해쉬한 $A_{1,3}$ 을 형제로, $Auth_1$ 만든후 이들의 해쉬값인 $A_{2,1}$ 의 형제인 $A_{2,0}$, 이 $Auth_2$ 가되고 이들의 해쉬 값인 $A_{3,0}$ 은 루트가 된다. 5 번 트랜잭션의 $A_{0,4}$, $A_{1,3}$, $A_{2,0}$ 이 Authentication path 가 6 번 트랜잭션의 참조 Authentication path 가된다. Authentication Path 중 어느 노드에서라도 값이 달라진 상황에서 계산된 OTP 값을 클라이언트로부터 받으면, 서버에서는 클라이언트와 같은 root 값을 계산할 수 없고 OTP 생성 값도 달라지게 되며 서버는 입력된 트랜잭션에 대하여 거절 메세지를 보낸다.

3.3. OTP 생성

머클트리 MT 에서 높이를 H 라고하면. 높이 h 인 authentication 노드 $auth_h$={$auth_0$, $auth_1$, ... $auth_h$}, h ∈ {0,1,...H-1} 이다. 여기서 secret key = hash($auth_0$|| $auth_1$||.. $auth_h$) 가된다. [그림 2]에서 6 번 트랜잭션의 참조 authentication path 는 5 번 트랜잭션의 authentication path 인 $A_{0,4}$, $A_{1,3}$, $A_{2,0}$ 가 되고 secret key = hash($A_{0,4}$ || $A_{1,3}$ || $A_{2,0}$) 가된다. TOTP = HOTP(secretKey, TC), TOTP-Value= TOTP mod 10d, 이므로 secretkey 대신에 hash($A_{0,4}$ || $A_{1,3}$ || $A_{2,0}$)를 입력하면 TOTP=HOTP(hash($A_{0,4}$ || $A_{1,3}$ || $A_{2,0}$), TC) 가되고 TOTP-Value= TOTP mod 10^d 로 생성될 비밀번호의 자릿수를 계산한다. d 는 TOTP 생성값의 자리수를 나타낸다.

3.4. OTP 체크

Client 는 현재트랜잭션과 직전 트랜잭션의 authentication path 로 생성된 OTP 비밀번호를 서버로 보낸다. 서버는 트랜잭션과 OTP 를 받고 자신이 가지고 있는 머클트리의 직전 authentication path 를 계산하고 OTP 를 생성하여 클라이언트에서 보내온 OTP 와 비교하여 트랜잭션의 거래 여부를 결정한다, 같으면 거래인증 메시지를 클라이언트에게 보내고 트랜잭션을 머클트리에 추가하고 루트를 갱신한다. 다르면 거절 메시지를 클라이언트에 보내고 서버의 트리는 갱신하지 않는다.

4. 결론

본 논문은 안전한 해쉬함수를 기반으로하는 머클 트리의 직전 Authentication Path 를 사용하여 현재 트랜잭션의 OTP 의 입력 값으로 하여 일회용 비밀번호를 생성하는 것을 보여주었다. OTP 의 비밀번호가 일치하기 위해서는 OTP 입력 값으로 변하는 시간외에 secret key 도 같아야한다. 여기서는 이 secret key 를 생성하는 알고리즘으로 머클트리의 authentication path 를 이용하였고 매번 새로운 거래나 메시지가 일어날때마다 머클트리에 추가하여 루트값을 갱신하였다 . 서버와 클라이언트는 각각 같은 머클 트리를 빌드하고 같은 트랜잭션을 추가하므로써 트리의 동기를 유지할때에야 authentication path 가같다. 트랜잭션이나 메시지를 사용하여 생성한 authentication 값이므로 OTP 의 고유하고 무작위 수여야 한다는 조건을 충족한다. 데이터가 추가될수록 서버와 클라이언트는 데이터 트리의 동기를 유지하기 쉽지 않으며 더욱 강한 OTP 비밀번호를 생성할수 있으며 서버와 클라이언트는 특별한 비밀키의 교환 없이도 서로를 확인할수 있는 트랜잭션으로 이루어진 머클트리를 이용하므로써 데이터의 무결성을 유지하고 각각의 시스템에 내재함으로써 사용자는 단말기를 가지고 다닐 필요도 없고 데이터 외에 다른 비밀번호를 입력하지 않아도 됨으로써 사용자의 사용성이 향상 되었다.

5. 감사의 글

"본 연구는 미래창조과학부 및 정보통신기술진흥센터의 고용계약형 SW 석사과정 지원사업의 연구결과로 수행되었음"(H0116-16-1015)

6. 참고문헌

[1] Financial supervisory service laws and regulations, 2015
[2] Alfred J. Menezes, Paul C. Van Ooschot, Scott,A Vanstone. <handbook of Applied Crypotography>(2001)
[3] P. Rogaway, T. Shrimpton, Cryptographic Hash-Function Basics: Definitions, Implications, and Separations for Preimage Resistance, Second-Preimage Resistance, and Collision Resistance, 2004
[4] D. M'Raihi , S. .Machani, M. P ei, J. Rydell ,TOTP: Time Based one time Password Algorithm , Mar. 2011
[5] N. Haller , Bellcore and others A One-Time Password System , Feb. 1998
[6] Georg Becker, Merkle Signature Schemes, Merkle Trees and Their Cryptanalysis 2008
[7] [Ede07] Boris Ederov. Merkle tree traversal techniques. Bachelor Thesis, Cryptography and Computer Algebra, 2007.
[8] Johannes Buchmann Luis Carlos Coronado ,Garcia,Errik Dahmen and others ,CMSS-Animproved Merkle Signature Scheme
[9] [SZY04]Michael Szydlo,, "Merkle Tree Traversal in Log Space and Time", Eurocrypt 2004.
[10] [Mer79]R. Merkle., Secrecy, Authentication and public key systems/A certified digital signature.
[11] Markus Jakobsson, Tom Leighton, Silvio ,Micali, and Michael Szydlo FractalMerkle Tree Representation and Traversal
[12] [SZY03]Michael Szydlo, Merkle Tree traversal in og space and time.(preprintversion, 2003), 2003. Merkle Tree Representation and Traversal
[13] Kwang-Cheul Shin, E-payment Authentication System Using QR_code and Mobile OTP,2015.7
[14] Jae-Young Lee, Do-Eun Cho, Jie-Young Lee, An Integrity Verification Method for Secure Application on the Smartphone, 2011,10
[15] Nam-Ho Kim, Voice-based OTP Generation Techniques for Mobile Banking, The Journal of Korean Institute of Information Technology, vol. 11, no. 5, 2013
[16] Yoon-Su Jeong, Yong-Tae Kim, Gil-Cheol Park, Smartphone User Authentication Mechanism using OTP based on iPIN, The Journal of Korean Institute of Information Technology, vol. 10, no. 9, 2012.