ISSN:2231-0711

Nikita Padmanabhuni et al | IJCSET |September 2013 | Vol 3, Issue 9, 329-333

A Study on Efficient Authentication Protocol with User Anonymity for Wireless Networks

Nikita Padmanabhuni, Mylavarapu.Prasanna, Vangala Himaja

Department of CSE,
JNTUK University, A.P. INDIA.

***Abstract*—In this paper, a new anonymous authentication protocol based on anonymous proxy signature for wireless Communications is proposed. The protocol involves only two parties including mobile user and visited server, without the participation of home server. Then the security and performance of the protocol are analyzed and compared with existing protocols. It is shown that the proposed protocol is efficient and power-saving with low time delay, which is appropriate for practical application.**

***Keywords*— Authentication; Anonymity; Proxy signature; Key Exchange.**

I. INTRODUCTION

With the wide spread of wireless network, more and more users are requiring anonymous authentications while roaming among different networks. Often a user does not like to be identified and tracked by anyone else including foreign/visited servers, except its own home server. But in wireless networks, wireless devices are limited in computation and storage, and bandwidths are also limited, so it is hard now for a user to get anonymity while impulse little burden on a wireless device. Five important properties have been proposed in [1] for strong anonymity:

(1) (Server Authentication) The user is sure about the identity (ID) of the visited server.

(2) (Subscription Validation) The visited server is sure about the ID of the home server of the user.

(3) (Key Establishment) The user and the visited server establish a random session key which is known only to them and is derived from contributions of both of them. In particular, the home server should not obtain the session key.

(4) (User Anonymity) Besides the user and the home server, no one can tell the ID of the user.

(5) (User Untraceability) Besides the user and the home server, no one including the visited server is able to identify any previous protocol runs which have the same user involved.

The above five rules must be strictly obeyed to get strong anonymity. If not only the users and the home server, but also the visited server can identifies or tracks the user, then we can only get weak anonymity.

This paper is organized as follows. In Section 2, some related work is introduced including conventional protocols based on three parties and new methods based on two parties. Then in Section 3, a protocol based on proxy signatures is proposed. We also want to reduce computation,so we instantiate an anonymous proxy signature based on elliptic curve. Its security and performance is analyzed and compared with existing protocols in Section 4. Finally we conclude it in Section 5.

II. RELATED WORK

There are two kinds of anonymous authentication protocols based on two parties and three parties respectively.

2.1 Three-party anonymous authentication protocols:

Conventional anonymous authentication protocols are involving all three parties including the user, the visited server and the home server, such as [2], [3] and [4]. In these protocols, the visited server has to communicate with the home server to authenticate the user. Obviously, all above protocols need many interactive rounds and long time delay. Moreover, the home server has to be always online or the authentication will not continue.

2.2 Two-party anonymous authentication protocols:

In recent years, authentications based on two parties making use of signatures have made some progress ([5], [6] and [9]). The visited server authenticates the user by signature of the user. The signature is not simply signed by the user, but also by its home server, which means the visited server does not have to communicate with the home server to authenticate the user. The visited server verifies the signature to ensure that the user is one of the clients of the home server, but it identifies who the user exactly is. The advantages are as follows:

(1) The authentication is involving only two parties, so the interactive round is less than previous protocols based on three parties, often less than 5 rounds.

(2) The home server does not need to participate in the authentication directly, so it can be offline so as to save money.

Most current researches focus on group signatures [7] to realize two-party anonymous authentication.

In this method, the home server is regarded as a group manager and the user is regarded as a member of this group. The user generates a group signature representing the group. This method realizes strong anonymity and its charging and user revocation mechanism are reasonable, but its only fault is its computation [6]. It needs much time and power to sign and verify a group signature, thus makes it impractical in mobile communication. One variant of group signature called Direct Anonymous Attestation (DAA) [8] is proposed in order to reduce time delay. The signature in this protocol is not generated by software but hardware called Trusted Platform Module (TPM). So it is safer and of course much faster. But with the extra TPM, it takes users much more money and is still impractical to massive applications. Another variant is k-times anonymous authentication [9]. It originates from group signature but is different in that the group manager cannot identify the user in a permitted number of authentication times. It realizes stronger anonymity because in k times even the home server does not know the ID of the user. Of course this scheme is more complex and harder to implement in mobile devices.

III. Anonymous Authentication Protocols Based On Proxy Signature

From the above we can see that, though group signature is a hotspot for two-party anonymous authentications, it is complicated to compute and hard to implement in ordinary and cheap mobile limited-resource devices. So we focus on other methods and find a new way called anonymous proxy signature to achieve our target.

3.1 Definition of anonymous proxy signature:

Definition: a proxy signature [10] is a signature that is authorized by original signer to proxy signers to generate a valid signature on behalf of the original signer. It is composed of four parts:

(1) Initialization: Parameters and key pairs are chosen for signature scheme.

(2) The delegation of signature right: The home server delegates its signature right to the user.

(3) The generation of proxy signature: The user generates a proxy signature on behalf of the home server.

(4) The verification of proxy signature: The visited server verifies the validity of the proxy signature.

If a proxy signer hides its ID in the proxy signature, then the proxy signature is called anonymous proxy signature.

3.2 The design of anonymous authentication protocols based on proxy signature:

In our design, the home server first authorizes the user the right of proxy signature; when the user roams to a visited network, it computes a proxy signature anonymously on behalf of the home server. The visited server verifies the signature to ensure that the user is one valid member of the home server. While the authentication is processing, the messages signed by them is used for key exchange to get a new session key. After the authentication, the user communicates with the visited server with the new ID and new session key.

3.3 Description of the protocol

3.3.1 Symbol definition:

H : ID of the home server.

B : ID of the user.

V : ID of the visited server.

mB : A message that is sent by B for key exchange.

mV : A message that is sent by V for key exchange.

$m\varpi$: A warrant obtained from H to B, which includes the ID of H, proxy expiration time of B,
the message types that are delegated.

Sig() : Signing algorithm of V .

Verify() : Verifying algorithm of V .

PKG() : Proxy key pair generation algorithm.

PSig() : Signing algorithm of B with its proxy private key.

PVerify() : Verifying algorithm of B with its proxy public key.

ECDSA : Elliptic curve digital signature algorithm.

3.3.2 Initialization

The key pairs of H and V are (xH, yH) and (xV , yV) respectively, with the public keys yH and yV known to all.

3.3.3 The delegation of signature right

H generates a temporary ID *alias* of B, records (*alias, B*) to its database for later privacy revocation and charging, and then replaces B with *alias* in $m\varpi$. Finally, H computes a signature sH on $m\varpi$ and sends to B, B gets *alias* from $m\varpi$ and computes a proxy key pair (xp, yp) by running PKG().

3.3.4 Authentication process of the protocol

The authentication process is illustrated in Fig. 1.

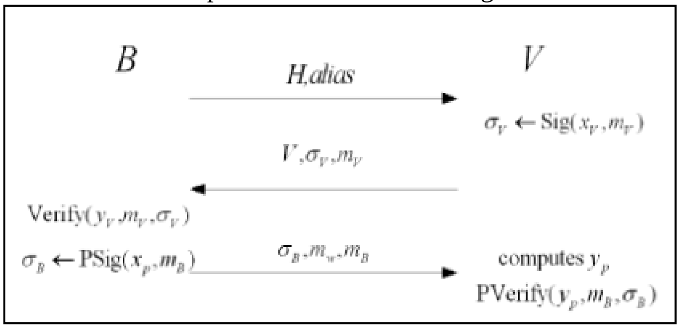

Fig. 1: Authentication process based on proxy signature

We describe it in detail as follows:

(1) When B roams to V , it first sends H and its temporary ID *alias* to V .

(2) V computes a signature σV with xV on mV , then sends (V, σV ,mV) to B.

(3) B verifies σV , and computes a signature σB with its proxy private key xp on message mB,
then sends (σB,mB,$m\varpi$) to V .

(4) V gets *alias'* from $m\varpi$, compares it with *alias* to see if *alias = alias'*. If the equation is
right, it then computes the proxy public key yp of B and verifies σB by PVerify(yp,mB, σB),
otherwise it terminates the authentication process.

(5) During the authentication, a session key is generated between B and V from mB and mV .

3.3.5 Revealing ID of B:

When it is necessary such as revoking or charging B, V submits *alias'* to H. H gets (*alias,B*) from its database and judges if *alias = alias'*. If the equation is right, then V charges B for its service.

3.4 An instantiation of the protocol

There are many anonymous proxy signature schemes which can be used in our protocol ([11], [12], and [13]). We choose a scheme [12] based on elliptic curve proxy signature which is efficient and fast in computation. Compared with [11] and [13], it is easier to implement. In terms of key exchange, conventional Diffie-Hellman key exchange is complex and low-efficient, so we choose Elliptic Curve Diffie-Hellman (ECDH) exchange to compute the session key.

3.4.1 Initialization

Let Fq be a finite field with an elliptic curve E in it, and G be a base point of E with prime $n$as its order. H has a key pair (xH, yH) with 1 _ xH _ n □ 1 and $yH = xHG$, B and V have similar key pairs (xB, yB) and (xV , yV). h() is a secure hash function.

3.4.2 The delegation of signature right

B selects a random 1 _ *kB* _ *n* □ 1, computes *rB* = *kBG* and *sB* = *xB* + *kBrB* mod *n*, then sends (*rB, yB,B*) to *H*. *H* computes *Yp* = *r*2 .*B* + *yB* mod *n* as the temporary ID *alias*, records
(*alias,B*) to its database, then replaces the ID *B* of *mϖ* with *Yp*; after that, *H* selects a random 1 <=*Kh*<= *n*-1, computes *rH* = *kHG*, then computes a signature *sH* = *xHh*(*mϖ, rH*)+*kH* mod *n* on *mϖ* and sends (*rH, sH*) to *B*. Finally *B* generates a proxy key pair (*xp, yp*), with the proxy private key *xp* = *sH*+*rHsB* mod *n* and the proxy public key *yp* = *yHh*(*mϖ, rH*)+*rH*+*rHYp* mod *n*.

3.4.3 Authentication and key exchange process

The protocol is illustrated in Fig. 2, which is described in detail as follows:

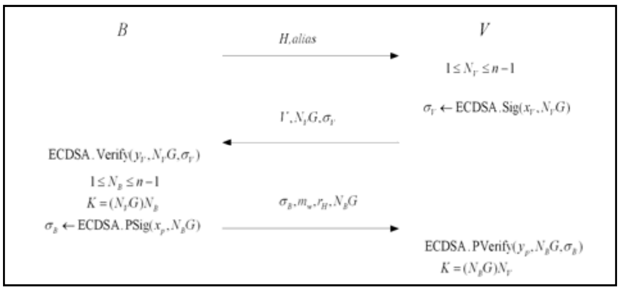

Fig. 2: Authentication and key exchange process

(1) When *B* roams to *V* , it first sends *H* and *alias* = *Yp* to *V* .

(2) *V* selects a random *NV* , computes *mV* = *NV G*, then computes a ECDSA signature *σV* on *mV* and sends (*V,mV ,σV*) to *B*.

(3) *B* verifies *σV* , selects a random *NB*, computes *K* = (*NV G*)*NB* and *mB* = *NBG*, then computes a ECDSA signature *σB* on *mB* with *xp* and sends (*σB,mϖ, rH,mB*) to *V* .

(4) *V* gets *Yp* from *mϖ*, compares it with *alias* to see if *alias* = *Yp*. Then it computes *yp* = *yHh*(*mϖ, rH*) + *rH* + *rHYp* mod *n*, verifies *σB* with *yp*, and computes *K* = (*NBG*)*NV* as the session key.

After the process, *B* communicates with *V* with *alias* and *K*.

3.4.4 Revealing ID of B

V submits *Yp* to *H*. *H* gets the record (*alias,B*) from its database and judges if *alias* = *Yp*. If the equation is right, then *V* charges *B* for its service.

IV. A More Efficient Protocol Using Elliptical Curve

4.1 Description:

We combine group signature and ACTP to propose an efficient and secure anonymous authentication protocol which is composed of two authentication processes:

- When *M* connects to an authenticator *F*1, *M* first authenticates *F*1 by verifying *F*1's signature, then *F*1 authenticates *M* by verifying *M*'s group signature. This process is similar to Yang and Huang's protocol [10].
- When *M* roams to authenticator *F*2, in order to reduce authentication delay, first authentication state information about *M* is transferred from *F*1 to *F*2, then *M* simplifies authentication process with *F*2. This process is more efficient than Yang and Huang's protocol.

4.2 System Parameters

Table 2 gives some notations.

4.3 Initialization

Fq is a finite field with an elliptic curve *E* in it, and *G* is a base point of *E* with prime *n* as its order; *H* has a master key pair (*mpkH*, *mskH*) of GSA, with public key *mpkH* known to all; *M* has a user signing key *uskM* of GSA; *F*1 has a key pair (*pkF*1 , *skF*1) of ECDSA, with verifying key *pkF*1 known to all. *H* generates a big and secret random number *N*, and computes an alias *aliasM* = *h*(*N*) ⊕ *IDM* for *M*, then gives it to *M* secretly.

4.4 Authentication

4.4.1 Process When M Connects to F1

Fig. 3 is the authentication process.

Table 1: Notations

ID_H	ID of H
ID_M	ID of M
ID_{F_1}	ID of F_1
ID_{F_2}	ID of F_2
$E_K()$	Encryption algorithm using a symmetric key K
$h()$	A secure one-way hash function
$PRNG()$	A pseudo-random number generator
ECDSA	The elliptic curve digital signature algorithm described in [10]
ECDSA.Sig()	The generation algorithm of ECDSA
ECDSA.Ver()	The verification algorithm of ECDSA
GSA	The group signature algorithm described in [10]
GSA.Sig()	The generation algorithm of GSA
GSA.Ver()	The verification algorithm of GSA

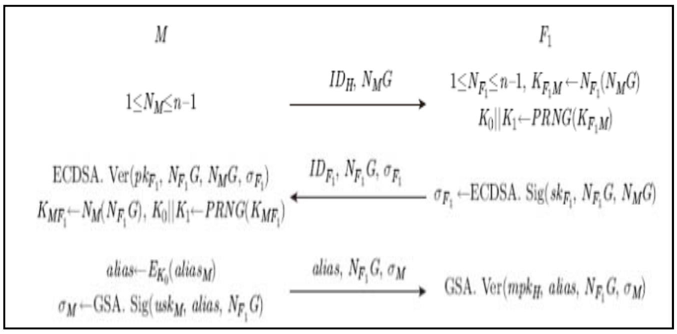

Fig. 3: Authentication process when *M* connects to *F*1

The authentication process is illustrated as follows:

(1) *M* selects a random number *NM*, and sends (*IDH,NMG*) to *F*1.

(2) *F*1 selects a random number *NF*1 , computes a ECDSA signature *σF*1 using *skF*1 , and then sends (*IDF*1,*NF*1*G*, *σF*1) to *M*. *F*1 then computes *KF*1*M* ← *NF*1(*NMG*) and derives two keys (*K*0,*K*1) by computing *K*0 ∥ *K*1 ← *PRNG*(*KF*1*M*).

(3) *M* verifies *σF*1 using *pk F*1 . If the signature is valid, it computes *KUV*1 and (*K*0,*K*1) as shown above, and then takes *K*1 as the session key; it then computes a temporary ID *alias* and a group signature *σM*, and then sends (*alias,NF*1*G, σM*) to *F*1. Otherwise, *M* rejects the connection.

(4) *F*1 verifies *σM* with *mpkH*. If the signature is valid, it then takes *K*1 as the session key. Otherwise, *F*1 rejects the connection.

4.4.2 Process When M Roams to F2

When M roams to $F2$, first $F1$ passes information (*alias, IDH,KF1M*) of M to $F2$. Then $F2$ computes ($K0,K1$) as shown above and gets *alias M* by decrypting *alias*. Finally M simplifies authentication process with $F2$ and updates the temporary ID and the session key. Fig. 3.1 is the authentication process.

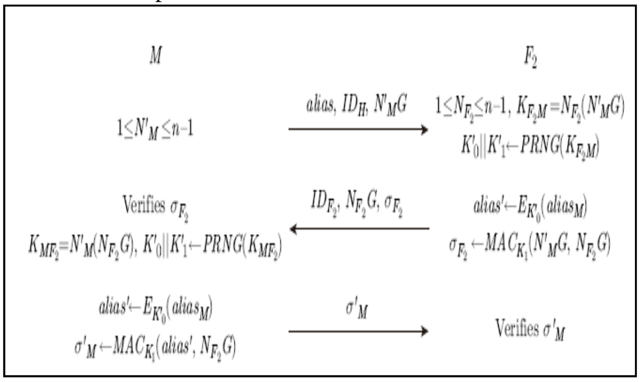

Fig. 3.1: Authentication process when M roams to $F2$

The authentication process is illustrated as follows:

(1) M selects a random number $N'M$, and sends (*alias, IDH,N'M G*) to $F2$.

(2) $F2$ selects a random number $NF2$, computes a message authentication (MAC [16]) value $\sigma F2$ using $K1$, and then sends ($IDF2$,$NF2G$, $\sigma F2$) to M. After that, $F2$ computes $KF2M = NF2(N'M\ G)$ and derives two keys ($K'0$,$K'1$) by computing $K'0 \mathbin{/\!/} K'1 \leftarrow PRNG(KF2M)$, and then updates M's temporary ID $alias' \leftarrow EK'0\ (alias\ M)$.

(3) M verifies $\sigma F2$ by computing $MACK1(N'M\ G,NF2G)$ and comparing it with $\sigma F2$. If $\sigma F2$ is valid, M computes $KUV2$, ($K'0$,$K'1$) and $alias'$ as shown above, and then computes a MAC $\sigma'M$ and sends it to $F2$. Otherwise, M rejects the connection.

(4) $F2$ verifies $\sigma'M$ by computing $MACK1(alias',NF2G)$ and comparing it with $\sigma'M$. If $\sigma'M = MACK1(alias',NF2G)$, then $F2$ takes $alias'$ as M's new temporary ID and $K'1$ as new session key. Otherwise $F2$ rejects the connection.

4.5 Reveal M's ID:

$F1$ sends M's message-signature pair ($alias \mathbin{/\!/} NF1G$, σM) to H; as shown in Ref. [10], H can recover IDM by some trapdoor. $F2$ only sends *alias M* to H, H takes out the secret random number N and recovers IDM by computing $IDM = h(N) \oplus alias\ M$.

V. Security And Anonymity

5.1 Security

The security is assured because the session key is generated based on Elliptic Curve Diffie-Hellman (ECDH) exchange which is secure according to Decisional Diffie-Hellman (DDH) assumption.

5.2 Anonymity

When M connects to $F1$, it does not send IDM in plaintext but a temporary ID *alias* instead. Anyone else including $F1$ cannot get IDM because only H knows the trapdoor.

When M roams to $F2$, $F2$ only gets *alias M* and cannot recover IDM because only H knows N. Besides, anyone else including $F1$ does not know $KF2M$, and thus cannot get M's new temporary ID $alias'$ which is encrypted using $K'0$, so M will not be identified and traced.

VI. Performance

We compare our protocol with Yang and Huang's protocol [10] in terms of terminal public key operations and time delay on a terminal with a 200MHz processor. When M first connects to $F1$, both protocols need 8.75 Elliptic Curve Scalar Multiplication (ECSM) plus 3Pairing operations on M.

When M roams to $F2$, 8.75ECSM plus 3Pairing operations are still needed in Yang and Huang's protocol, but only 2ECSM operations are needed in our protocol. Performance comparison is shown in Table 3. Fig. 4 is the latency comparison which shows that time delay in our protocol is nearly half of that in Yang and Huang's protocol.

Schemes	Number of rounds	Transmission time (ms)	Terminal public key operations	Terminal computation time (ms)	Time delay (ms)	Time delay Percentage (Our/ Yang-Huang)
Yang-Huang	3	300	8.75ECSM+3P	315.25	615.25	56.24%
Our protocol	3	300	2ECSM	46	346	

Table 3: Performance comparison when M roams to $F2$

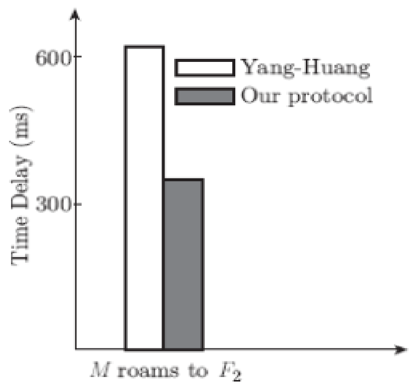

Fig. 4: The latency comparison

Conclusion

Our new protocol needs only three rounds to complete anonymous authentication with low delay, and involves only two parties without the participation of home server, so it is appropriate for band-limited wireless network. Though it cannot change its temporary ID at random and can only get weak anonymity, it is still a practical protocol because of its excellent performance. We will focus on user untraceability as our future work to enhance its anonymity.

Group signature imposes high computational load on terminals, so it is not proper to use group signature very often, especially when users roam among heterogeneous networks frequently. ACTP is a good way to improve the efficiency, and its security is also assured. Using ACTP in our protocol reduces nearly half of the roaming latency. So it is practical for heterogeneous wireless networks.

REFERENCES

[1] J. M. Zhu, J. F. Ma, An efficient authentication protocol with anonymity for wireless IP networks, Journal of China Institute of Communication, 25 (2004), 12-18

[2] H. X. Peng, D. G. Feng, Security flaws and improvement to a wireless authentication protocol with anonymity, Journal on Communications, 27 (2006), 78-85

[3] L. J. Dang, W. D. Kou, X. F. Cao, Mobile IP registration protocol with user anonymity, Journal of Xidian University, 35 (2008), 281-287

[4] L. Yang, J. F. Ma, J M Zhu, Trusted and anonymous authentication scheme for wireless networks, Journal on Communications, 30 (2009), 29-35

[5] R. F. Wan, F. W. Li, J. Zhu, An Efficient anonymity mutual authentication protocol, Journal of University of Electronic Science and Technology of China, 34 (2005).

[6] F. Zhang, J. F. Ma, S. J. Moon, Universally compassable anonymous Hash certification model, Science in China Series F – Information Sciences, 50 (2007), 440-455

[7] Y. Lee, I. Lee, H. Lee, New identity escrow scheme for anonymity authentication, in: Proc. INDOCRYPT.

[8] W. Jiang, C. Clifton, Privacy-preserving distributed k-anonymity, in: Proc. DBSeic, 2005, 166-177

[9] H. Ge, An anonymous authentication scheme for identification card, in: Proc. ICICS, 2006, 238-248

[10] G. Yang, Q. Huang, D. S. Wong, X. Deng, Universal authentication protocols for anonymous wireless communications, IEEE Transactions on Wireless Communications, 9 (2010), 168-174

[11] J. Camenisch, S. Hohenberger, M. Kohlweiss, A. Lysyanskaya, M. Meyerovich, How to win the clonewarsi: Efficient periodic n-times anonymous authentication, in: Proc. ACM Conference on Computer and Communications Security, 2006, 201-210

[12] C. Politis, K. A. Chew, N. Akhtar, M. Georgiades, R. Tafazolli, T. Dagiuklas, Hybrid multilayer mobility management with AAA context transfer capabilities for all-IP networks, IEEE Wireless Communications Magazine, 11 (2004), 76-88

[13] M. Georgiadesii, H. Wang, R. Tafazollii, Security of context transfer in future wireless communications, in: Proc. IWCMC, 2006, 389-394

[14] F. Allard, J. M. Combes, R. Marin, A. F. Gomez, Security analysis and security optimizations for the context transfer protocol, in: Proc. NTMS, 2008, 1-5

[15] G. Karopoulosi, G. Kambourakis, S. Gritzalis, Privacy protection in context transfer protocol, in: Proc. 16th Euromicro Conference on PDP, 2008, 590-596

[16] J. Black, S. Halevi, H. Krawczyk, T. Krovetz, P. Rogaway, UMAC: Fast and secure message authentication, in: Proc. CRYPTO, 1999, 216-233