Applying a cryptographic scheme in the RPINA protocol

Giannakis Antoniou[1], Ana Jancic[2], Udaya Parampalli[3], Leon Sterling[4]
[1, 3, 4]*Department of Computer Science and Software Engineering*
The University of Melbourne
{[1]gant, [3]udaya, [4]leon}@csse.unimelb.edu.au
[2]*Deakin University*
ana.jancic@deakin.edu.au

Abstract

The Respect Private Information of Non Abusers (RPINA) protocol allows users to communicate anonymously while the corresponding server ensures that the identity of the user will be revealed by a third party, called Directory Service (DS), should the user attempt a malicious attack. However, malicious co-operation between the server and the DS is the major vulnerability of the RPINA protocol. In this paper we introduce a technique, which decrease this vulnerability.

1. Introduction

The desire of Internet users to hide their identity has led to the development of technologies to protect the identity of users. Such technologies are called Privacy Enhancing Technologies (PET). There are PETs offering anonymity at the communication layer (i.e. hiding IP address), and others at the data layer (i.e. hiding email address, credit card and social security number). Some PETs which offer communication anonymity are Anonymizer [13], Crowds [14], and TOR [15]. A user, who wants to protect his /her communication identity, sends the messages to the PET and the PET forwards the messages to the desired server. In this scenario, the server is unable to identify the user should it transpire that the user attempted to attack the server. For this reason a server needs to have an accountability service where the communication identity of the user-attacker can be revealed if necessary.

There are a number of protocols/approaches which are revoking anonymous communications [1, 16, 17 and 18]. The RPINA [1] protocol is one of them. The RPINA (as well as the PPINA [2]) protocol lets a user enjoy communications anonymity through the PET while the server enjoys the prospect of accountability service in case the user attempts to attack. The revocation anonymity of the user takes place through a third entity called Directory Service (DS). The DS is the only entity, apart from the PET entities, that can relate the exchanged messages with the sender of those messages, even though the DS knows neither the messages nor the related Server before the Server contacts with the DS. Therefore, the DS is a single point of failure. If a Server receives messages of an anonymous user (AU) and the Server wants to get the identity of that AU, then a compromised DS can reveal the identity of the user. In this paper, we propose a technique which reduces this vulnerability of the RPINA protocol.

1.1. RPINA protocol

A user requests a ticket from a DS and the DS issues the ticket, which contains the signature of the DS. The DS doesn't know the potential Server which the user is going to communicate with. The user sends the ticket to the Server through a PET. The PET is responsible for preventing the Server identifying the user. Once the Server verifies the validity of the ticket, it accepts communication with that user.

Second International Workshop on Digital Forensics and Incident Analysis (WDFIA 2007)
0-7695-2941-0/07 $25.00 © 2007 IEEE

The user sends messages to the Server, always through the PET. Once the Server identifies that the user attempted to attack, the Server requests from the DS to reveal the identity of the user. The DS will reveal the identity of the user only if the messages are indeed malicious. Otherwise, the DS does not reveal the identity of the user.

However, the DS can be compromised by the Server and reveal the identity of an honest user. It would be desirable to employ a technique where the Server will have a more difficult task before successfully attacking and revealing the identity of the user, rather than attacking to a single entity like the DS.

The fact that the ticket reveals the identity of the corresponding DS (the ticket contains the digital signature of the DS) helps a malicious server to identify the entity (DS) who knows the owner (AU) of the ticket. In case the Server successfully compromises the DS, it can find out the relative AU.

We need to find a technique/scheme which fulfills the following requirements:

a) The ticket should not reveal the identity of the DS which can link the ticket with the corresponding AU, because a Server may find and compromise the DS in order to identify the AU. Therefore, the DS must be able to sign anonymously.

b) One or more entities should be able to prove (by providing evidence which offers non-repudiation) that the ticket has been issued by the specific entity (i.e. DS). Otherwise, these entities may claim than an unrelated DS issues the ticket. Therefore, the scheme must be able to offer non-repudiation of the actions of the DS.

c) Only one entity should be able to link the ticket with the AU. The more entities that can link the ticket with the corresponding AU, the more chances exist to violate the privacy of the AU.

d) The ticket should be linked with the entity/entities, which can link the ticket with the DS. This entity (or entities) should have the obligation to reveal the identity of the DS; otherwise that entity can be considered to be an attacker. Therefore, the scheme must be able to offer non-repudiation of the actions of the central entity (i.e. group manager).

1.2. Signature schemes

There are a number of signature schemes that have been derived from the classical digital signatures, such as group signatures, threshold signatures, and proxy signatures.

Various threshold signature schemes, such as [4, 7 and 8] have been proposed by now in order to increase the security by removing single point of failure. The idea behind threshold signature schemes [4] is to distribute secret information and computation among n parties by trusted dealer or without it in order to remove single point of failure. To sign a message M any subset of more than t parties can use their shares of the secret and execute an interactive signature generation, which output a signature of M that can be verified by anybody using the unique fixed public key [4]. The threshold group signature [4] stays secure even if there are any $t < n/2$ malicious parties in the system.

The threshold signature scheme is compromised if the number of malicious signers is greater or equal to $n/2$, and it would be very hard to identify malicious signers from the group. Thus it cannot offer complete non-repudiation of the actions of members. Based

on the requirements listed in section 1.1 the Threshold scheme is not appropriate because it violates the second requirement.

The group signature is similar concept that has been introduced by Chaum and van Heyst [6]. Several different models have been proposed [3, 5 and 9]. In the groups signature scheme all members of a group sign messages on behalf of the group. Signatures are then verified with respect to a single group public key, but they do not reveal the identity of the signer. It is not possible to decide whether two signatures have been issued by the same group member. However, there is a group manager who can open signatures and reveal the identity of the signer. Compared to threshold signature scheme, the group signature offers higher level of confidentiality by removing the single point of failure. However, the signatures produced by the group members link the group manager with the signature. Therefore, it violates the first requirement where the ticket should not reveal the identity of the entity who knows the AU.

A proxy signature scheme [11, 10 and 12] enables a proxy signer to sign a message on behalf of an original writer. According to delegation rights, Mambo et al. [11] defined three different levels of delegation: full delegation, partial delegation, and delegation by warrant. In full delegation, the original signer gives his private key to the proxy signer and proxy signer uses the key to sign documents.

This scheme does not provide accountability for proxy signers (DSs) because they sign on behalf of the original signer (group manager). For this reason the proxy signature scheme is not appropriate as it violates the second requirement.

None of the above schemes fulfilled all the requirements listed on section 1.1, which need to be satisfied in order to be applied in the RPINA protocol and prevent a DS and a Server to co-operate maliciously and reveal the identity of an honest user. In the next section we proposed a scheme based on classical public digital signatures in order to fulfill the desired requirements listed in section 1.1.

This paper has the following structure: In Section 2 the technique resolving the problem is described in detail. In Section 3, the technique is applied to the RPINA protocol. In Section 4 a couple of scenarios are described, while Section 5 concludes and suggests future work.

2. The proposed technique

In this section we combine a set of classical signature scheme (which offers anonymity) with the RPINA protocol in order to decrease the possibility of the Server attacking the DS and revealing the identity of an honest user.

Here are the functions used in the next sections:

$S_X(M)$: The message M is signed using the private key X. The function returns the value signedMessage (where signedMessage=signature, message)

$\{M\}K_X$: The message M is encrypted by the asymmetric key X.

$\{M\}KS_X$: The message M is encrypted by the symmetric key X.

H(M): The message M is hashed.

$V_X(M)$: The message M is verified by the public key X.

nonce: A random number

GenerateKey(): The function returns a random number appropriate to be used as a secret key

GeneratePairOfKeys(): The function returns a random pair of keys - public and private keys

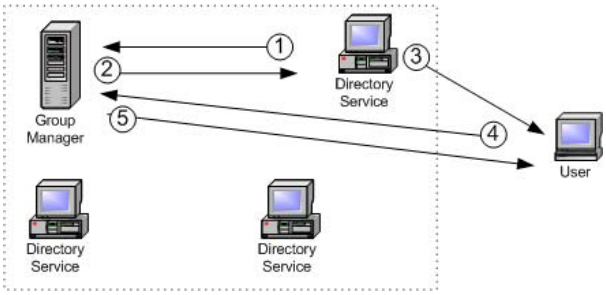

Figure 1 - Anonymous Signature scheme with PKI

We assume that each entity has a pair of keys, where the private key (i.e. GMprivate-key, DSprivate-key) is kept secret and the public key (i.e. GMpublic-key, DSpublic-key) is published to a trusted third party (TTP) and is known to the rest of the participating entities.

We also assume that the communication among the entities is protected against a traffic analysis attack, by using a PET.

With respect to the requirements listed in section 1.1, here is a description of each step (Figure 1) followed by the actual protocol:

Let GMticket= $S_{GMprivate\text{-}key}$ (DSsilent-public-key)

Step 1: The DS generates a pair of keys (DSsilent-public-key/DSsilent-private-key). The DS signs the DSsilent-public-key with the DSsilent-private-key, and finally DS signs everything with the DSprivate-key. The DSpublic-key is publicly known (it is available to a TTP). Any signature verified by the DSpublic-key is considered to belong to DS.

$$DS \rightarrow GM: S_{DSprivate\text{-}key}(S_{DSsilent\text{-}private\text{-}key}(DSsilent\text{-}public\text{-}key))$$

Step 2: The GM verifies that the DSsilent-public-key belongs to the owner of the DSprivate-key because the DSsilent-public-key verifies the signature produced by the DSsilent-private-key. The GM signs the DSsilent-public-key with his GMprivate-key and sends it to the DS. The sent information on step 2 is also called GMticket.

$$GM \rightarrow DS: GMticket$$

Step 3: Although the DS is now ready to sign messages without being identified, the GM has evidence which link the signed messages (which are signed by using the DSsilent-private-key) with the specific DS. The DS signs and sends the desired message (MESSAGE) to the user, including the GMticket. The user is responsible to verify that the data from step 3 contains a valid GMticket where the DSsilent-public-key (found in the GMticket) verifies the signature created by the DSsilent-private-key.

$$DS \rightarrow User: S_{DSsilent\text{-}private\text{-}key}(GMticket, MESSAGE)$$

Step 4: Let us say that the user wants to find out the originator of the message. The user sends the received data from step 3 to the GM. The user can identify the GM based on the signature found in the GMticket, as far as the DSsilent-public-key signed by the GMprivate-key can verify the signed message produced by the DSsilent-private-key.

$$User \rightarrow GM: S_{DSsilent\text{-}private\text{-}key}(GMticket, MESSAGE)$$

Step 5: Let us say that the GM wants to help the user to identify the DS who signs the MESSAGE. The GM is able not only to identify the signer of the MESSAGE but it can also prove to the user that the signature belongs to the specific DS. The GM sends to the user the data from step 1, which shows that the DS was able to sign the DSsilent-public-key by using the DSsilent-private-key. Only the entity who knows the DSsilent-private-key could sign the MESSAGE. Therefore, the user can be sure about the identity of the responsible DS.

$$GM \rightarrow User: S_{DSprivate\text{-}key}(S_{DSsilent\text{-}private\text{-}key}(DSsilent\text{-}public\text{-}key))$$

The DS is able to sign anonymously a message by using the DSsilent-private-key as far as the GMticket is valid and part of the signed message.

Based on step 3, the User can:

a) Verify the authenticity of the message by checking whether the DSsilent-public-key found in the GMticket verifies the signature produced by the DSsilent-private-key.

If $V_{DSsilent\text{-}public\text{-}key}(S_{DSsilent\text{-}private\text{-}key}(\text{GMticket, MESSAGE}))$ ***Then***

The entity who signs the message has been authenticated by the owner of the GM (based on the GMprivate-key found in the GMticket).

Else

The entity who signs the message has not been authenticated by the owner of the GM

End if

b) Identify only the GM because GMticket contains the signature of the GM.

c) Prove to any third party that the GM knows the DS who generates the information of step 3. This can be concluded from the fact that the GM authenticates the entity who can sign with the DSsilent-private-key.

Once the GM reveals the identity of the DS to the user, the DS must request a new GMticket from the GM because that user is able to link the DSsilent-public-key with the specific DS. Moreover, in order to achieve a higher level of anonymity, the DS can use a GMticket only once and each time the DS wants to sign anonymously, the DS should request a new GMticket. However, the technique with the higher level of anonymity is practically inefficient.

In the next section we apply the above scheme to the RPINA protocol.

3. Modified RPINA protocol

The RPINA protocol is well described and analyzed in [1]. In this paper we describe and analyse only the effects of the RPINA protocol after we apply the proposed scheme into it. The RPINA protocol describing in this section differs from the RPINA protocol described in [1] as follows:

a) The GM is introduced in order to authenticate the DS and let the DS sign anonymously.

b) The Ticket reveals the identity of the GM and not the identity of the DS because the DS is able to sign anonymously. Therefore, a malicious Server who wants to compromise the DS and get the identity of the AU cannot do it without knowing the relative DS.

c) A Server victim should first contact the GM (and provide evidence that the messages are malicious) in order to get the identity of the DS. Once the GM accepts that the messages are malicious, it reveals the identity of the DS who has issued the Ticket. Then the Server

contacts the DS and re-provides the evidence which prove that the messages are malicious. Finally, the DS reveals the identity of the AU.

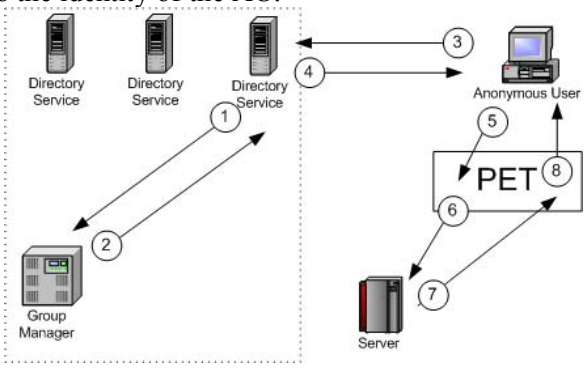

Figure 2 - Initialization Phase

Figure 3 - Main Phase

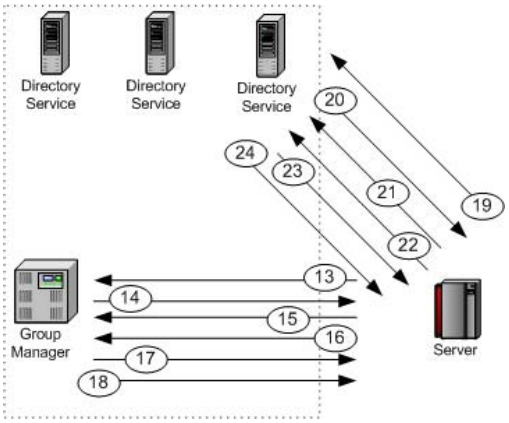

Figure 4 - Forensic Investigation Phase

$Ticket = S_{DSsilent\text{-}private\text{-}key}(GMticket, S_{AUsession\text{-}private\text{-}key}(Token2))$

$RequestForTicket = S_{AUprivate\text{-}key}(\{aKey, Token\}K_{DSpublic\text{-}key}, \{S_{AUsession\text{-}private\text{-}key}(Token2)\}\ KS_{aKey})$

The RPINA protocol is divided into three phases, the Initialization (Figure 2), the Main (Figure 3) and the Forensic Investigation (Figure 4) phases. The three phases are described below (a more detailed description is in [1]).

Initialization Phase

The DS requests the GMticket from the GM (step 1) and the GM issues the GMticket (step 2). The DS is now able to sign Tickets anonymously. Before the AU request a ticket from the DS, the AU needs to generate a pair of keys [AUsession-private-key/AUsession-public-key= GeneratePairOfKeys()], generates a random key [aKey=GenerateKey()], calculates the hash value of the AUsession-public-key [Token=H(AUsession-public-key)], and calculates the

hash value of the Token [Token2=H(Token)]. Once the AU requests a Ticket (step 3) from the DS, the DS issues the Ticket (step 4). The AU sends the Ticket and the AUsession-public-key (also known as "secret key") to the Server through the PET (steps 5 and 6).

DS→GM: $S_{DSprivate\text{-}key}(S_{DSsilent\text{-}private\text{-}key}(DSsilent\text{-}public\text{-}key))$ (Step 1)

GM→DS: GMticket (Step 2)

AU→DS: RequestForTicket (Step 3)

DS→AU: $\{Ticket\}K_{AUpublic\text{-}key}$ (Step 4)

AU→PET: Ticket, $\{S_{AUsession\text{-}private\text{-}key}(secret\ key)\}K_{SERVERpublic\text{-}key}$ (Step 5)

PET→SERVER: Ticket, $\{S_{AUsession\text{-}private\text{-}key}(secret\ key)\}K_{SERVERpublic\text{-}key}$ (Step 6)

SERVER→PET: $\{Ticket\}KS_{secret\ key}$ (Step 7)

PET→AU: $\{Ticket\}KS_{secret\ key}$ (Step 8)

Main Phase

The AU and the Server can communicate through the PET without the Server knowing the identity of the AU. However, the Server can identify and accuse the AU in the case that the AU acts maliciously. Once the Server detects an attack, the Server begins the Forensic Investigation Phase.

AU→PET: $S_{AUsession\text{-}private\text{-}key}(\{Data,nonce\}KS_{secret\ key})$ (Step 9)

PET→SERVER: $S_{AUsession\text{-}private\text{-}key}(\{Data,nonce\}KS_{secret\ key})$ (Step 10)

SERVER→PET: $\{S_{SERVERprivate\text{-}key}(Data)\}KS_{secret\ key}$ (Step 11)

PET→AU: $\{S_{SERVERprivate\text{-}key}(Data)\}KS_{secret\ key}$ (Step 12)

Forensic Investigation Phase

The Server first contacts the GM and gives as evidence the messages received by the AU during the Initialization and Main phases. The Server can identify the responsible GM based on the Ticket because the Ticket contains the GMticket signed by the specific GM. The GM examines the messages and reveals (by providing evidence) the identity of the DS. Then the Server sends the messages again to the DS. DS also examines the messages and finally the DS reveals (by providing evidence) the identity of the AU.

SERVER→GM: $S_{SERVERprivate\text{-}key}(Ticket)$ (Step 13)

GM→SERVER: $S_{GMprivate\text{-}key}(GM\_CaseID)$ (Step 14)

SERVER→GM: $S_{AUsession\text{-}private\text{-}key}(\{Data,\ nonce\}KS_{secret\ key})$ (Step 15)

SERVER→GM: $S_{SERVERprivate\text{-}key}(GM\_CaseID)$ (Step 16)

GM→SERVER: $S_{GMprivate\text{-}key}(ForensicReceipt,\ GM\_CaseID)$ (Step 17)

GM→SERVER: $S_{DSprivate\text{-}key}(S_{DSsilent\text{-}private\text{-}key}(DSsilent\text{-}public\text{-}key))$ (Step 18)

SERVER→DS: $S_{SERVERprivate\text{-}key}(Ticket)$ (Step 19)

DS→SERVER: $S_{DSprivate\text{-}key}(DS\_CaseID)$ (Step 20)

SERVER→DS: $S_{AUsession\text{-}private\text{-}key}(\{Data, nonce\}KS_{secret\ key})$ (Step 21)

SERVER→DS: $S_{SERVERprivate\text{-}key}(DS\_CaseID)$ (Step 22)

DS→SERVER: $S_{DSprivate\text{-}key}(ForensicReceipt, DS\_CaseID)$ (Step 23)

DS→SERVER: $S_{DSprivate\text{-}key}(\{S_{DSprivate\text{-}key}(secretkey, ForensicReceipt, aKey, IP\ Address, RequestForTicket)\}K_{SERVERpublic\text{-}key})$ (Step 24)

The information sent at steps 15, 18 and 21 has already been obtained from the Server, GM and Server respectively during the steps 10 (main phase), 1 (init. phase) and 10 (main phase).

3.2. Analysis of the protocol

The above protocol successfully removes the single point of failure. The Server has to maliciously compromise the GM and the related DS in order to discover the identity of an honest user, because the Server cannot identify directly the related DS based on the Ticket. Before applying the proposed scheme in the RPINA protocol, an AU should rely only on the DS. After we apply the scheme, the AU can rely not only on the DS but also on the GM. Even though one of them (DS and GM) is reliable, the malicious Server cannot identify the AU. However, both entities, the DS and the GM, still examine the evidence provided by the Server (steps 15 and 21) subjectively.

4. Scenarios

The following Scenarios present the outcome of having malicious participating entities (giving emphasis to the Forensic Investigation Phase) in the RPINA protocol. Although relatively similar scenarios have been also described in [1 and 2], the following scenarios present a different perspective.

4.1. Scenario - Dishonest AU:

In this scenario the only dishonest entity is the AU who attacks the Server. However, the Server is not able to identify the AU nor the DS because the Server does not know their identities. Based on the ticket, the Server can identify only the GM. The Server sends the messages, received by the AU (steps 6 and 10), to the GM, who is responsible for deciding whether the messages are malicious or not. The GM decides that the messages are malicious and as a result it reveals the identity of the DS. By proving that the DS knows the DSsilent-private-key, the GM can convince the Server that the ticket has been issued by the specific DS. The message from step 1 proves that the DS knows the DSsilent-private-key.

The Server contacts the DS and re-provides the received messages of the AU (steps 6 and 10) to the DS. Once the DS decides that the messages are considered malicious, it reveals the identity of the AU to the Server. The DS can also prove that the malicious messages have been sent by the specific user because the RequestForTicket contains the [$S_{AUsession\text{-}private\text{-}key}(Token2)$] which is also found in the ticket. As far as the messages are linked with the ticket, and the ticket is linked with the RequestForTicket, we can safely conclude that the user who made the RequestForTicket is also the same entity who has sent the messages to the Server.

4.2. Scenario - Dishonest DS and Server:

In this scenario the DS and the Server are dishonest while the GM and the AU are honest. When the DS issues the ticket, it doesn't know the identity of the Server. Moreover, although

the Server has the ticket, it doesn't reveal the identity of the DS. Therefore, the Server and the DS don't know each other. For that reason, the Server contacts the GM in order to get the identity of the DS. However, the Server doesn't have evidence which proves that the owner of the ticket has acted maliciously. As a result, the GM does not reveal the identity of the DS and the Server is not able to get the identity of the honest user.

5. Conclusions and Future work

A layer to prevent the violation of a user's privacy has been introduced in this paper. The layer has been built by using an anonymous signature scheme and then we applied the scheme into the RPINA protocol in order to let the DS hide its identity when it signs tickets. It is understandable that the introduction of the GM does not completely solve the problem. However, it can prevent malicious co-operation between the DS and the Server revealing the identity of an honest user.

Future research is planned to develop an appropriate scheme where the AU can rely on a number of entities, rather than the DS and GM only, and at the same time the scheme must fulfill the requirements of section 1.1.

6. References

[1] Antoniou, G., Gritzalis, S., "RPINA: Network Forensics protocol embedding Privacy Enhancing Technologies", Proceedings of the ISCIT 2006 International Symposium on Communications and Information Technologies, A. Taguchi et al. (Eds.), October 2006, Bangkok, Thailand, ISBN: 0-7803-9741-X, IEEE Press, pp. 297-302, 2006

[2] Antoniou, G., Wilson, C., Geneiatakis, D.,"PPINA - A Forensic Investigation Protocol for Privacy Enhancing Technologies", Proceedings of the 10th IFIP CMS'06 Communications and Multimedia Security Conference, Iraklion, Greece, H. Leitold and E. Markatos (Eds.): CMS 2006, LNCS 4237, pp. 185–195, 2006

[3] Boneh, D.Boyen, X., and Shacham , H., "Short group signatures", In Advances in Cryptology- CRYPTO 2004 (Santa Barbara, California, USA, August 2004), M. K. Franklin, Ed. Vol. 3152 of Lecture Notes in Computer Science, Springer-Varlag, pp. 45-55, 2004

[4] Boldyreva, A., "Efficient signature, mutlisignature and blind signature schemes based on the Gap-Diffie-Hellman-group signature scheme", Proceedings of PKC, 2003. August 12, 2002. http://mirror.cr.yp.to/eprint.iacr.org/2002/118.pdf

[5] Camenisch, J., and Groth, J. ,"Group signatures: Better efficiency and new theoretical aspects", In Security Communication Networks (SCN2004) (2004), vol. 3352 of Lecture Notes in Computer Science, Springer, pp. 120-133, 2004

[6] D. Chaum, E. van Heyst. "Group signatures", Advanced Cryptology- EuroCrypt '91, pages 257-265, Berlin, 1991. Springer- Varlag. Lecture Notes in Computer Science Volume 547, 1991

[7] Y.Desmedt, "Threshold cryptography", European Transactions on Telecommunications, 5(4), 1994

[8] Y.Desmedt, and Y. Frankel, "Threshold cryptosystems", Advances in Cryptology- Crypto'89, LNCS Vol. 435, G. Brassard ed., Springer-Verlag, 1989.

[9] Furukawa, J., and Imai, H. "An efficient group signature scheme from bilinear maps", In Australasian Conference on Information Security and Privacy (ACISP2005) (2005) vol. 3574 of Lecture Notes in Computer Science, Springer, pp. 455-467, 2005

[10] Kim, S., Park, S., and Won, D. (1997), "Proxy signatures, revisited", Proc. Of ICICS'97, International Conference on Information and Communications Security', LNCS 1334, Springer-Verlag, pp. 223-232, 1997

[11] Mambo, M., Usuda, K. & Okamoto, E. (1996), "Proxy signatures for delegating signing operation", in 'Proc. 3rd ACM Conference on Computer and Communications Security', New Dehli, India , ACM Press New York, pp.48-55, 1996

[12] Petersen, H. and Horster, P..:"Self-certified keys-concepts and applications", in 'Proc. Communications and Multimedia Security'97', Chapman & Hall, pp.102- 116, 1997

[13] Anonymizer (2003), available at http://www.anonymizer.com

[14] Reiter M., Rubin A., "Crowds: Anonymity for web transactions", ACM Transactions on Information and System Security (TISSEC), Vol. 1, Issue 1 (Nov 1998), Pages: 66 - 92 , 1998

[15] Roger Dingledine, Nick Mathewson, and Paul Syverson, “Tor: The Second-Generation Onion Router”, In Proceedings of the 13th USENIX Security, Symposium, August 2004

[16] Claudia Diaz and Bart Preneel, "Accountable Anonymous Communication," Chapter in: Security, Privacy and Trust in Modern Data Management, Springer, (in print), 15 pages, 2006.

[17] Stefan K¨opsell1, Rolf Wendolsky2, Hannes Federrath2, “Revocable Anonymity”, Proc. Emerging Trends in Information and Communication Security: International Conference, ETRICS 2006, Freiburg, Germany, June 6-9, 2006, LNCS 3995, Springer-Verlag, Heidelberg 2006, 206-220, 2006

[18] J. Claessens, C. Diaz, C. Goemans, B. Preneel, J. Vandewalle, and J. Dumortier, "Revocable anonymous access to the Internet," Journal of Internet Research 13(4), pp. 242-258, 2003