Cryptography 2, Assignment 1: Traceable Anonymous Certificate (RFC5636)

Rens van der Heijden
University of Twente
TU/e student number 0762495
r.w.vanderheijden@student.utwente.nl

May 20, 2011

1 Introduction

This paper provides an overview and discussion of the cryptography used for Traceable Anonymous Certificates (TACs) in the Request For Comment (RFC). RFCs are published by the Internet Engineering Task Force (IETF) for documenting standards, recommendations, and informational and experimental documents. The discussed RFC [5] has experimental status and provides the details for a technical implementation of TACs within the X.509 Public Key Infrastructure (PKI). Procedural details are not defined within this RFC, though some recommendations are made with respect to the separation of the two introduced entities. The technical implementation is provided by means of two protocols, which apply cryptographic means to protect the user of the certificate from systematic linking of his identity to the certificate. The applied cryptographic concepts are blind signatures and threshold cryptography.

The layout of this paper is as follows: Section 2 explains how TACs fit into the PKI architecture, Section 3 discusses the requirements for this RFC, while Section 4 describes the solution in detal and Section 5 explains where TACs may be used in practice.

2 TACs in the PKI

In RFC 5636, a scheme for issuing anonymous X.509 certificates is described. Usually, X.509 certificates are issued using the real name of the end user within the certificate, in the Subject field. Instead, TACs will contain a pseudonym in this field, thereby creating a certificate that is not linked to the real name of the user.

These certificates are expected to be used by general end users, who can use them for the same purposes as normal certificates, without explicitly linking their real name to this certificate. This works because for an end user certificate, the name of this end user is usually not important for the relying party to trust the certificate, unlike certificates used by companies. Conversely, company names are public knowledge and are generally not linked to a single person, which limits the benefit of TACs for corporate entities. Notice that the TAC cannot offer the user a means to be completely anonymous; a relying party might still obtain identifying information about the user, through the use of a cookie or other uniquely identifying information, such as browser information [2].

Using TACs in place of a normal certificate provides the user with additional protection, by hiding his real name, without any additional disadvantages. To obtain a certificate, the user initiates the procedure to obtain a certificate as usual. However, instead of a certificate, he will receive a token from the Blind Issuer (BI), which is passed on to the Anonymous Issuer (AI) for the signing procedure. The AI and BI communicate to verify the uniqueness and correctness of the token, after which both use their part of the threshold signing key to sign the certificate.

To achieve this, the Anonymous Issuer (AI) and the Blind Issuer (BI) are introduced. Here, the BI maintains the relation between pseudonyms for traceability (see also Sections 3 and 4), while the AI performs the actual signing. Thus, the identity of the user is known to the BI, but not to the AI. In this context, the BI corresponds to a normal RA, while the AI performs the tasks of a CA. For this model to provide pseudonymity also against these two parties, a strong separation between RA and CA is necessary.

Another important note is that in the TAC scheme, one cannot obtain several certificates with the same subject name and different purposes. This limits the applicability of the TACs (eg, it cannot be used for S/MIME, as noted in the RFC), but protects against abuse. The author of this paper thinks it is conceivable to issue multiple certificates for distinct, non-overlapping purposes. For example, a user could ask for multiple certificates, say k, in the registration procedure; the BI can then maintain a counter, initialized to k, and sign the blinded hashes only when k tokens have been received from the AI. The BI should include this number to the token, in order to prevent abuse; notice also that this requires stricter authentication between user and AI, to prevent an attacker from obtaining part of these certificates. If the user authenticates with a separate TAC, this process will not cost him anonymity, since linking the certificates for different purposes should already be possible. However, there will be some additional cost in terms of implementation, processing overhed and revocation overhead to implement a scheme like this.

For similar reasons, the RFC also does not allow the user to refresh the TACs. On the other hand, a user is allowed to obtain an unlimited amount of certificates from the CA, so obtaining sufficient key material should not be a problem. When the TAC approaches the end of its lifetime, its user can request a new certificate and use the current certificate to link the two TACs with respect to the relying parties the user is interacting with.

3 Security Requirements

The following security requirements are the core requirements of TACs. Requirements defined by the protocol, which mostly considers authenticated secure channels between communicating parties, are not considered here, as they are set by the RFC.

Pseudonymity Pseudonymity or conditional anonymity requires messages to be linkable to a pseudonym, while only under strict conditions allow linking of the pseudonym to the real identity of a user. This requirement is the purpose of TACs; if it is not met, using TACs is not beneficial. The requirement is met by providing the user with a certificate that contains a pseudonym chosen by the user, unless this pseudonym has already been issued by this CA, in which case the AI will select a random one or stop the procedure.

Fraud Prevention Fraud Prevention means that the CA can prevent repeated abuse of its certificates by a user. Notice that certificate revocation is not an adequate solution here; the user may request many certificates in a short period of time in order to increase his anonymity. The RFC solves this requirement by requiring linkability and providing a protocol that allows the CA to link the TAC involved in fraud to the user to whom it was issued. The protocol will be described in Section 4.3.

Non-repudiation Non-repudiation means that the user of the certificate cannot deny that he participated in the certification protocol. This requirement is closely related to the fraud prevention requirement. However, fraud prevention could also be achieved in another fashion, while non-repudiation requires that it is not possible for a user to request a TAC for another user.

4 Solution

The main challenge solved by this RFC is deciding how to provide some anonymity, while still being able to use the standard X.509 certificates.

4.1 Cryptography

To provide proper signatures on the certificates without revealing the real name of the end user, blind signatures are used. To prevent attackers from requesting signatures for arbitrary domains, and to be able to trace the certificate back to the end user, the link between the pseudonym in the certificate and the name of the user is stored by a separate entity.

Blind signatures work as follows:

Threshold signatures are an application of threshold cryptography, which offers the following:

4.2 Certification Protocol

The certification protocol is initiated when a user wants to obtain a certificate. The entire protocol is shown in Figure 1.

Step 1 In the first step, the user that requests the certificate authenticates himself to the BI. The BI, verifies the users identity similar to an RA and stores

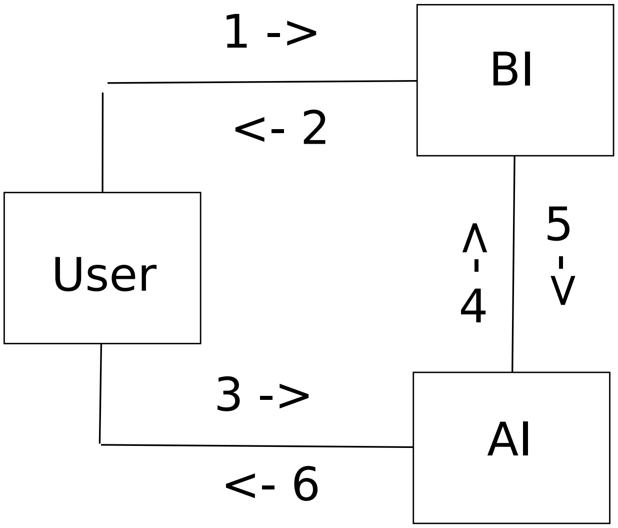

Figure 1: The Certification Protocol from section in [5]

the users identity linked to a unique[1] user key and a session expiration time. This user key must not be linkable to the users real identity; for example, if a hash is used, this hash should be salted to protect against guessing attacks. The BI creates a token containing user key and the expiration time, signed with its own key material.

Step 2 The BI sends the signed token to the user over a secure channel, after which the user should verify the signature before proceeding to the next step. The token will be used for later linking the TAC back to the user and to prevent a user from requesting certificates without registering with the BI.

Step 3 The user creates a certificate request in some standard format; PKCS10 is required to be supported, but other formats like CMC may be used. In this request, the subject field should contain the pseudonym chosen by the user. The pseudonym may also be generated by software provided by the CA, in order to limit collisions on the subject field. The user then sends its request to the AI over a secure channel that authenticates the AI. This request will contain the token the user received from the BI as an attribute. Notice this channel must not authenticate the user, as this could reveal his identity, or at least allow the AI to link the new TAC to another, existing certificate, reducing anonymity.

Step 4 The AI checks the format of the received message and verifies the signature on the token. After this, the expiration time defined in the token is checked and if valid, checked against a cache of recently validated tokens to prevent replay attacks. If the subject name is already used for another certificate, the AI will either reject the request or select a random other pseudonym. The AI sets all the fields that are included in the signed data of the certificate, computes a hash over this data and blinds the hash value. Notice that it is essential that the keypair used for blinding remain secret from the BI, if anonymity from the BI is to be maintained. Subsequently, the AI signs the blinded hash and sends it over a bidirectionally authenticated and secure channel, together with the token provided by the user, who obtained it from the BI. The certificates for the signature from AI are required to be included.

Step 5 The BI checks the signature of AI on the blinded hash and its own signature on the token. It then checks the token to verify it has not previously been used to create a certificate. The token is permanently associated with the user data in the database, to allow for the traceability component of the TAC. The BI then signs the blinded hash with his part of the threshold signing key, after which it signs the result with his own signing key and returns it to the AI.

Step 6 The AI verifies the signature of BI on the blinded, partially signed hash and matches it against a list of outstanding requests to the BI. The AI then unblinds the result, providing the partially signed hash, and uses its part of the threshold signing key to complete the signature on the TAC. The relation between the TAC and token is stored so that the user may later be traced. Finally, the AI returns the TAC to the user using the protocol used by the user in step 3. The user can now use the TAC for the purpose it requested it for.

Issues in the certification protocol Notice that in steps 4 and 5 is a small flaw in the system. If an attacker inside the AI decides to break the system, he can send a different token with the blinded hash, which will cause the final certificate to be linked to someone other than the user that requested the TAC. This can happen only when there is a token that has not been used previously and will cause the token to be invalided. Also, the attacker cannot perform this

[1] unique to this specific BI

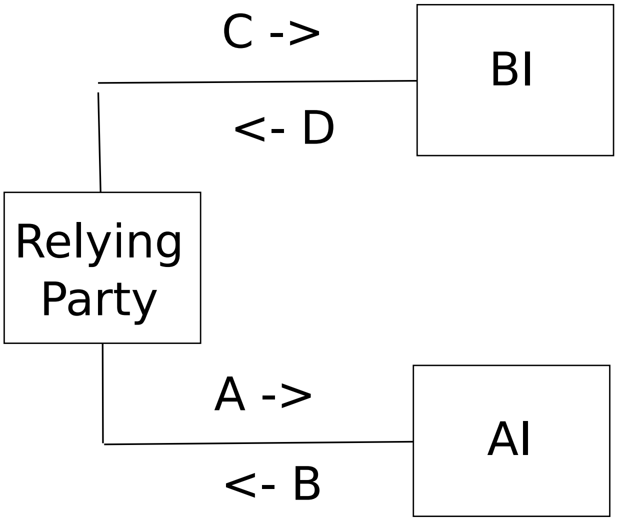

Figure 2: The Linking Protocol from section in [5]

task selectively unless he colludes with either someone in the BI (to link token to user data) or with the user (to give this user a certificate that is linked to some random other user). If the attacker is also the user, he can obtain such a certificate for himself. For this reason, the author of this paper recommends that employees from the AI should not be allowed to register at the associated BI. Considering that the AI acts as CA, which is considered to be a trusted party, and the limited gain from this type of attack, the risk of this attack occuring in practice should be limited.

4.3 Linking Protocol

To obtain the identity of a fraudulent user, a relying party can start the linking protocol by communicating separately with the AI and BI. The protocol is shown in Figure 2.

Step A The AI first verifies that the relying party indeed has shown that the user has abused the anonymity of the certificate. Procedures for this are not defined in the RFC. The author considers it to be good practice that a judge should resolve this dispute, which is somewhat stronger than the current system where the user of a specific IP address at a specific time. The current system for IP addresses depends largely on how well the service provider protects the rights of its customers. For CAs, the rules should be stricter, because releasing anonymity of a certificate may reveal more information than revealing information about the IP address.

After this, the TAC is provided to the AI by the relying party. The AI then revokes the TAC by adding it to the next Certificate Revocation List (CRL), released by the AI. This CRL is issued by a second CA certificate, directly below the CA issuing the TAC and managed by the AI, because according to RFC 5280 [1], processing indirect CRLs is not required. Notice that providing a direct CRL would require interaction with the BI, because the CA certificate used for signing is managed by both.

Step B Next, the AI searches for the token corresponding to the TAC and transmits it to the relying party through a bidirectionally authenticated and secure channel.

Step C The relying party forwards the token to the BI and requests the identity of the user. The BI may independently verify the correctness of the complaints of the relying party, depending on the certificate policy.

Step D BI verifies its signature on the token, retrieves the identity of the user to whom the TAC was issued and sends the information to the relying party.

Issues in the linking protocol As noted in the section about security considerations in the RFC, during this protocol the AI can return an incorrect user key (the random value in the token originally generated by the BI during registration). This will trace the certificate to a random other user. While the RFC considers it an unavoidable issue, the author of this paper thinks that it should be possible to extend the certificate to contain a commitment to the user key, in order to protect against this. The commitment scheme used for this should be information-theoretically hiding and computationally binding, such that the AI cannot cheat within reasonable time and no information can be deduced from the commitment within the certificate. This can be achieved by using a Pedersen commitment [6].

5 Applications

Pseudonymous certificates as proposed in this RFC are interesting for numerous applications where temporary or permanent identities are necessary. In particular, in the near future, the introduction of Vehicular Ad-hoc Networks (VANETs) will require frequent switching between certificates in order to preserve location privacy of the driver [4, 3]. Although TACs do not address all the requirements for VANETs, they can be considered a good first step towards a PKI that is appropriate for this application.

6 Conclusion

The TACs proposed in this RFC adequately address the pseudonymity goals for the external attacker. However, from an inside attacker from within the AI, the user of a TAC is not sufficently protected, as explained by the various flaws discussed in Sections 4.2 and 4.3.

References

[1] D. Cooper, S. Santesson, S. Farrell, S. Boeyen, R. Housley, and W. Polk. Internet X.509 Public Key Infrastructure Certificate and Certificate Revocation List (CRL) Profile. RFC 5280 (Proposed Standard), May 2008.

[2] P. Eckersley. How unique is your web browser? In M. Atallah and N. Hopper, editors, *Privacy Enhancing Technologies*, volume 6205 of *Lecture Notes in Computer Science*, pages 1–18. Springer Berlin / Heidelberg, 2010.

[3] F. Kargl, P. Papadimitratos, L. Buttyan, M. Mter, E. Schoch, B. Wiedersheim, T.-V. Thong, G. Calandriello, A. Held, A. Kung, and J.-P. Hubaux. Secure vehicular communication systems: Implementation, performance, and research challenges. *IEEE Communications Magazine*, 46(11):110–118, November 2008.

[4] P. Papadimitratos, L. Buttyan, T. Holczer, E. Schoch, J. Freudiger, M. Raya, Z. Ma, F. Kargl, A. Kung, and J.-P. Hubaux. Secure vehicular communication systems: design and architecture. *IEEE Communications Magazine*, 46(11):100–109, November 2008.

[5] S. Park, H. Park, Y. Won, J. Lee, and S. Kent. Traceable Anonymous Certificate. RFC 5636 (Experimental), Aug. 2009.

[6] B. Schoenmakers. Cryptography 2 / cryptographic protocols: Lecture notes. `http://www.win.tue.nl/~berry/2WC13/LectureNotes.pdf`, 2011.