# An Overview of Certificate Request Message Format

Spyridon Antakis

Eindhoven University of Technology
Department of Mathematics & Computer Science
Den Dolech 2, P.O Box 513, 5600 MB Eindhoven
Email: {s.antakis}@student.tue.nl

February 20, 2009

**Abstract**

A Public Key Infrastructure enables users to communicate to an unsecured public network, such as the Internet. Thus, users must somehow, secure the exchanged data that are transmitted using this network. A popular way for accomplishing security and privacy is to obtain and share a certificate, through a trusted authority. Certificate Request Message Format (CRMF) is one of the specifications that define a format for certificate requests. This paper, presents a general overview of CRMF with regard to security issues and it makes a comparison with the PKCS#10.

## 1 Introduction

A *Public Key Infrastructure (PKI)*, is a system that aims to create an environment that supports the exchanging of *keys* and *certificates*. Inside this system, many different components which are closely related, are combined. Each component, performs a specific operation and contributes into the entire infrastructure. *Certificate Request Message Format (CRMF)* is a specification that was presented in 2005, by J .Schaad [1]. This specification describes the format that must be used in the certificate and key exchange, inside a PKI. More precisely, it defines the *syntax* and the *semantics* of a requested certificate in a X.509 Public Key Infrastructure.

This document focuses on the usage of CRMF. In section 2, the security requirements and concerns of CRMF are introduced and the importance of *proof-of-possession* is discussed. Furthermore, in section 3, two different kind of threats are described and possible countermeasures are proposed. Moreover, a comparison between *CRMF* and the *Public Key Cryptography Standard 10 (PCKS#10)* is performed. Finally, the document concludes about the importance of CRMF specification inside the X.509 PKI.

### 1.1 Usage of the Certificate Request Message Format

A certificate request message, which is constructed using the CRMF specification, contains 3 main parts: i) *a certificate request*, *an optional proof of possession field* and also *an optional registration information field*. Certificate request, includes the important details for the requested certificate. Proof of possession is a calculated value, that proves the binding of the entity's private key with the public key included in the requested certificate. Registration information, is a field that includes additional useful information for the fulfillment of the request. [1]

On the submission of a certificate request, PKI components such as *Certification Authority (CA)*, *Registration Authority (RA)* and *End Entity (EE)* cooperate in order to produce a X.509 certificate. In practice, an *EE* submits a certificate request, possibly via a *RA*, to a *CA* and it waits the issued certificate as a return. The problem that arises is that CA must determine whether this EE does really own the corresponding private key or not. Therefore, the EE should provide a proof-of-possession for this private key. While for signature keys the straightforward solution is to provide a signature during registration, the situation is not so easy for encryption keys. CRMF focus on this problem and proposes different solutions, on how a certificate request exchange must be implemented.

Nevertheless, the reader must keep in mind that the created CRMF certificate request messages, do not constitute an independent protocol. Thus, an external *Certificate Request Protocol (CRP)* is also needed, as a transporter between the PKI entities. The choice of CRP is really crucial and vital for the request exchange, as is expected to define what algorithms will be used.

The *core* of CRMF specification, lies on the provided *Proof-Of-Possession (POP)* mechanism. POP is a procedure that performs a check, on whether the certificate requester is in possession of the private key that corresponds to the public key included in the requested certificate. The CRP is responsible for enforcing the POP algorithm, but this is not always possible, as it is not supported by every operational protocol (e.g. several mail protocols). There are many different methods to perform POP, but this is something that depends on how the key is used (signature, encipherment or key agreement).

The policy of the system sets the requirements and decides who must validate the certificates. However, the specification of CRMF, does not restrict who must perform the certificate validation (CA, RA, both). Therefore, the following cases are possible, depending on the system's policy: i) RA forwards the certificates to CA without validating them and then CA verifies the certificates during issuance, ii) RA first verifies the certificates, rejects the failing certificate requests and forwards the valid ones to CA. Then, CA verifies them also and proceeds with their certificate issuance, iii) RA verifies the certificate requests and simple informs CA through an element for the success of their validation, iv) CA and RA use out-band-methods in order to verify proof of possession.

## 2 Security Requirements

The enrollment protocols used by several implementations and also the CRP, must provide security mechanisms which will ensure a safe communication between EEs, RAs and CAs. A possible leakage of the private key, will allow to third parties to attack the system with many different ways, depending on type used for the key [1] :

- Compromise the signer's private key and be able to sign messages, by pretending the signer.
- Compromise the decryption private key and intercept sensitive messages.

Therefore, if an implementation fails to secure the communication, then all the private information and the sensitive data exchanged will be an easy target for a potential attacker. Thus, it is really necessary for all protocols to provide reliable security mechanisms, such as masking via encryption and high entropy for private key generators.

### 2.1 Security Concerns

A number of security concerns appear, depending on the implementation and the usage of CRMF. *Repudiation* and *key escrow* are two of the main concerns that may lead to a successful attack. The provided feature of the remotely generation of the key, must be not used for creating signing keys because it can cause a repudiation attack. Any user, that captures the generated key during its transportation from the generator to the EE, will be able to sign also messages with this key. The same vulnerability can cause an interception of an encryption key by a third party. A proposed solution, is to apply a cryptographic algorithm that will perform a masking into the transported generated key, therefore only the proper receiver will be able to unmask it.

During a *key escrow*, the user must be sure for the identity of the entity that is going to handle the private keys. For this reason, the CRP is responsible for providing an identification mechanism. Furthermore, in order to increase security measures, the key should be masked during any transportation and stored in a secure way (i.e. re-encrypted). Nevertheless, the major concern is that an escrow agent, needs to check the requests for escrowed keys and only when the requester is allowed to have access it should return the private key. However, if this check does not take place, an attacker could intercept the encrypted private key of an entity, build a certificate request around it and then ask for a recovery operation on the private key. Therefore, a proof of identity is needed for avoiding this kind of threats.

## 2.2 Importance of Proof-Of-Possession mechanism

Proof-Of-Possession, is really important for implementing security countermeasures for the certificate exchange between the involved entities. A POP algorithm must provide mechanisms that can prevent security weaknesses that appear in this exchange. In fact, POP was designed and included into CRMF specification, to ensure two basic things [1] :

- That the private key is tied to a specific user.
- That the user has use of the key in question.

## 3 Defending against possible attacks

As we already described in subsection 2.2, Proof-Of-Possesion is an efficient countermeasure for avoiding some of the security problems. Below, we present two possible threats and how we can defend against them, by applying POP.

### 3.1 Threat A - Signing Keys

Suppose that Alice (EE), which is in possession of a *private key X* and its corresponding *public key Y*, sends to Charlie (CA) a certificate request containing Y. Then, Charlie returns to Alice the issued certificate that contains also the Y, but without checking the binding between the two keys (No - POP). Imagine now, that Mal (EE) which is in possession of a different *private key Z*, which does not correspond to the public key Y, sends to Charlie also a certificate request that contains key Y (Fig: 1). In this way, as there is no binding key check, Mal can also receive an issued certificate that contains key Y. [1]

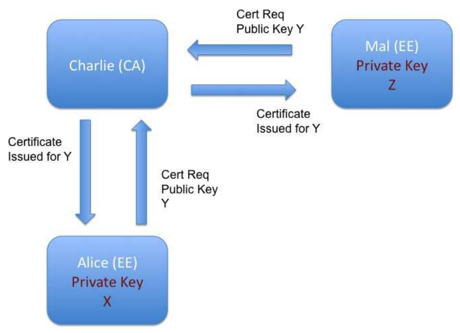

Figure 1: *Attack 1- No Proof-Of-Possesion for signing keys*

This lack of POP check, permits a repudiation attack. First of all, Mal can now claim that he was the *real signer* of a transaction, which was signed by Alice, with her private key X. Moreover, Alice can deny that she was the signer of a transaction and she can claim that it was Mal, the one that signed the transaction. Thus, in order to avoid and stop this security attack, Charlie (CA) must request from Mal (EE) to prove that he really possesses the private key X , before issue the requested certificate.

### 3.2 Threat B - Key Management

Suppose that Al is a professor in a University Department and Dorothy is the Department Head. Al wants to send a copy of a draft final exam that he created, to Dorothy for a review. This exam will be encrypted, as several students have access to the system. However, inside the certificate repository, many students have certificates containing the same public key management key as Dorothy. [1]

Thus, if no POP is performed by the CA, Al has no way of knowing whether all of the students have simply created these certificates without knowing the corresponding private key or whether the students have somehow acquired Dorothy's *private key D* (Fig: 2). So, if CA provides POP, then either no students will have such certificates or Al can know for sure that the students have intercepted Dorothy's *private key D*.

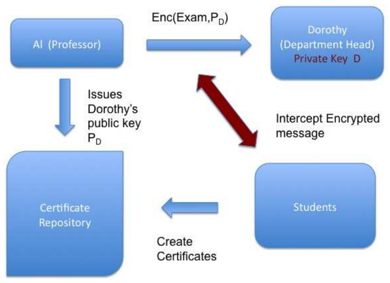
Enc(Exam,P_D)
Al (Professor)
Dorothy (Department Head) Private Key D
Issues Dorothy's public key P_D
Intercept Encrypted message
Certificate Repository
Students
Create Certificates

Figure 2: *Attack 2 - No Proof-Of-Possesion for key management*

## 4 CRMF & PKCS#10

*Public Key Cryptographic Standard 10 (PKCS#10)* is a popular certificate request syntax standard, which was presented in 2000, by RSA Laboratories. A PKCS#10 certificate request consists of three parts: i) *certfication request information*, ii) *a signature algorithm identifier* and iii) *a digital signature on the certification request information.* [2]

PKCS#10 defines a syntax in which entities can provide information needed for creating a certificate from the CA. Moreover, it allows them to include other data that can be used during and after the certification process. PKCS#10 syntax, requires the message to be secured with a digital signature. One of the purposes that this is done, is to provide a *proof-of-possesion* for the corresponding private key, since the signature can be verified from the receiving party (RA, CA). PKCS#10 is one of the most commonly used certificate format, especially in the message exchange of web-based applications. However, this syntax does not support keys used for encryption only. Encryption keys, can not be treated the same as the signing keys due to limitations of the cryptosystem and they have different security requirements which PKCS#10 seems not able to handle.

On the other hand, CRMF format is able to handle encryption keys and it is designed to overcome PKCS#10 obstacles. While handling signature keys identically to PKCS#10, it provides three methods for *proof-or-possesion* of encryption keys. The first is to reveal the private key to the CA. The second one, called direct method, requires exchange of challenge-response messages and thus it needs extra messages. The third one, is the indirect method, in which the certificate issued is encrypted with the contained public key and then EE is demanded to send the decrypted certificate , as a confirmation message. Through this procedure, EE is able to demonstrate ownership of the corresponding private key.

In general, both mechanisms are used for accomplishing the same purpose. PKCS#10 has been in use for some time and covers a large part of the used applications, but it is missing the important cryptographic features. As a comparison, CRMF has been developed to cover all of the used applications and support the cryptographic features, but is not widely implemented. Therefore, as long as the industry find it useful to support both standards in their products, the current PKCS#10-based applications will be able to participate in the PKI system.

## 5 Conclusions

Certificate Request Message Format (CRMF) is a major and promising contributor inside the X.509 Public Key Infrastructure. It introduces extended cryptographic mechanisms for securing the certificate exchange and points out the most severe possible threats that the implementers of a system would probable have to face. Furthermore, CRMF presents and describes why there is a need for applying *proof-of-possesion* algorithms, as a countermeasure in the exchange of certificate requests. It shows, the importance of verifying that a private key, indeed corresponds to the public key inside a certificate request. Moreover, the specification clarifies that a constructed certification request message, is not a stand-alone protocol, but it needs to closely cooperate with a *certificate request protocol*, which must enforce security mechanisms. Therefore, it becomes clear to the reader that for the success of a safe certificate request message transportation between the PKI entities, CRMF specification is not enough. Several other components (e.g. enrollment protocols) are involved and must be taken into account.

In a way, we could say that CRMF is the expected improved version of PKCS#10 format. Until now, both of them are used in commercial applications, but the intention is CRMF to substitute the PKCS#10, as it constitutes a complete solution for certificate request format. At the moment, the majority of the applications are supporting both certificate request formats and very often PKCS#10 is used for compatibility issues. Nevertheless, everything indicate that in the future, one commonly acceptable specification for certificate requests will be adopted, aiming to provide fundamental security countermeasures.

## References

[1] *J. Schaad,* "Internet X.509 Public Key Infrastructure Certificate Request Message Format (CRMF)", RFC 4211, Soaring Hawk Consulting, (September 2005).

[2] *M. Nystrom and B. Kaliski,* "PKCS#10: Certification Request Syntax Specification Version 1.7", RFC 2986, RSA Laboratories, (November 2000).

[3] *B.Weger* "Cryptography II - Cryptographic Systems", Eindhoven University of Technology, Department of Mathematics & Computer Science, v1.2, (December 2008).