Attestation Protocols: A Tutorial Introduction

Perry Alexander and Brigid Halling
Information and Telecommunication Technology Center
The University of Kansas
{palexand,bhalling}@ku.edu

November 12, 2012

Contents

List of Figures

List of Tables

Abstract

This document is intended to provide a tutorial overview of the basic attestation protocols used by a TPM.

1 Introduction

2 Privacy Certificate Authority Based Attestation

There are two versions of the Privacy CA attestation process, one documented by Ryan [2009] and the other documented in the TCG specification [—, 2007]. They are not dramatically different, but should be reconciled.

2.1 The Ryan Approach

The Ryan protocol, shown in figure 1, is documented in his invaluable technical report Ryan [2009]. We have enhanced it here to include more explicit interaction between the appraiser and the user software.

1. An appraiser sends an attestation request indicating what PCRs are needed using a PCR mask along with a nonce, n.[1]
2. The user software receives the request and requests a fresh AIK from the TPM, wrapped with the TPM's SRK using the `TPM_MakeIdentity` command. Parameters to the `TPM_MakeIdentity` command describe properties of the desired AIK pair and include a digest of the CA and Label identifying the target CA (CA_d).
3. The TPM responds to the `TPM_MakeIdentity` command by generating a new AIK wrapped by SRK and returns the new wrapped AIK along with *idContents* containing CA_d and AIK signed by AIK^{-1}. *The new AIK is a key wrapped by SRK and can only be installed on the TPM associated with SRK. Thus, the only way that $\{|CA_d, AIK|\}_{AIK^{-1}}$ can be created is in the presence of the TPM that generated AIK.*
4. The user software sends a certified public EK ($\{|EK|\}_{AIK^{-1}}$) and the *idContents* ($\{|CA_d, AIK|\}_{AIK^{-1}}$) obtained from the TPM to the Privacy CA and requests that it certify AIK.
   It is not clear where the certified public EK is obtained, but the public EK is available to the CA in many different fashions.
5. The Privacy CA uses AIK to check the signature on EK and then determines if the EK has been revoked. It then uses AIK to check the signature on *idContents*. *If idContents signature check and EK signature check succeed, the Privacy CA knows the AIK, CA_d and EK came from the same TPM.* It then uses its own public key and name to regenerate CA_d and compare with the signed CA_d sent to it. *If the generated digest is the same as the received digest, the CA knows the certification request is intended for it.* If all three checks are successful – EK signature, *idContents* signature, and CA_d generation – the Privacy CA signs the public AIK and encrypts the AIK certificate with a fresh session key. It they encrypts both the session key and the certified AIK with EK and returns both to the user software. *Only the TPM with EK^{-1} can decrypt this package, ensuring that only the TPM that generated the request can decrypt the response.*
6. The user software uses the `TPM_ActivateIdentity` command to decyrpt the session key ($\{K\}_{EK}$). *Only the TPM associated with EK^{-1} can decrypt the session key.* The user software then in turn decrypts the signed AIK ($\{\{|AIK|\}_{CA^{-1}}\}_K$) using K. *The (AIK) signed by the Privacy CA ($\{|AIK|\}_{CA^{-1}}$) can only be obtained in the presence of the TPM associated with EK.*
7. The user software requests a quote signed with AIK^{-1} from the TPM using the `TPM_Quote` command and the nonce, n, sent by the appraiser. The TPM produces the quote, signed using AIK^{-1}. *This ensures the quote comes from the holder of AIK^{-1}.* $\{|AIK|\}_{CA^{-1}}$ is also signed with AIK^{-1}. *This ensures the certified public AIK comes from the only TPM that could decrypt it.* The user software sends the signed, certified AIK and signed quote back to the appraiser.[2]

[1] This request is not formally documented, but its specifics are not critical for this discussion.
[2] The details of this communication are not specified.

8. The appraiser analyzes the signed blobs received from the user software as follows:
   (a) AIK is used to check the signature of $\{|\{|AIK|\}_{CA^{-1}}|\}_{AIK^{-1}}$ — The AIK was signed by the TPM where the AIK is installed.
   (b) CA is used to check the signature of $\{|AIK|\}_{CA^{-1}}$ — The Privacy CA has certified the AIK is installed in a legitimate TPM. The session key encrypting the certificate can only be decrypted by the TPM making the certification request.
   (c) AIK is used to check the quote signature — The quote sent was signed by the TPM where the AIK is installed and the same TPM that generated the signature on $\{|\{|AIK|\}_{CA^{-1}}|\}_{AIK^{-1}}$
   (d) n is checked against the nonce sent with the original request — The TPM quote is fresh, avoiding a replay attack using a cached quote.
   (e) PCRs from the quote are compared with expected values — The remote system is configured as expected.

2.2 TCG Documentation Approach

The TCG protocol is documented in the TPM technical documentation by way of describing the TPM command set provided [—, 2007]. At this time, the distinction is the `TPM_MakeIdentity` command returning a signed public EK in addition to the signed AIK. If we can determine that this is the same as the certification request (CR), then the two protocols are basically the same.

- $CA_d \equiv$ CA public key and label digest
- $SRK_h \equiv$ SRK key handle
- $AIK_h \equiv$ AIK key handle
- $AIK_d \equiv$ public AIK key digest
- $PCR_m \equiv$ mask specifying target PCRs for a quote
- $PCR_d \equiv$ digest of PCR values

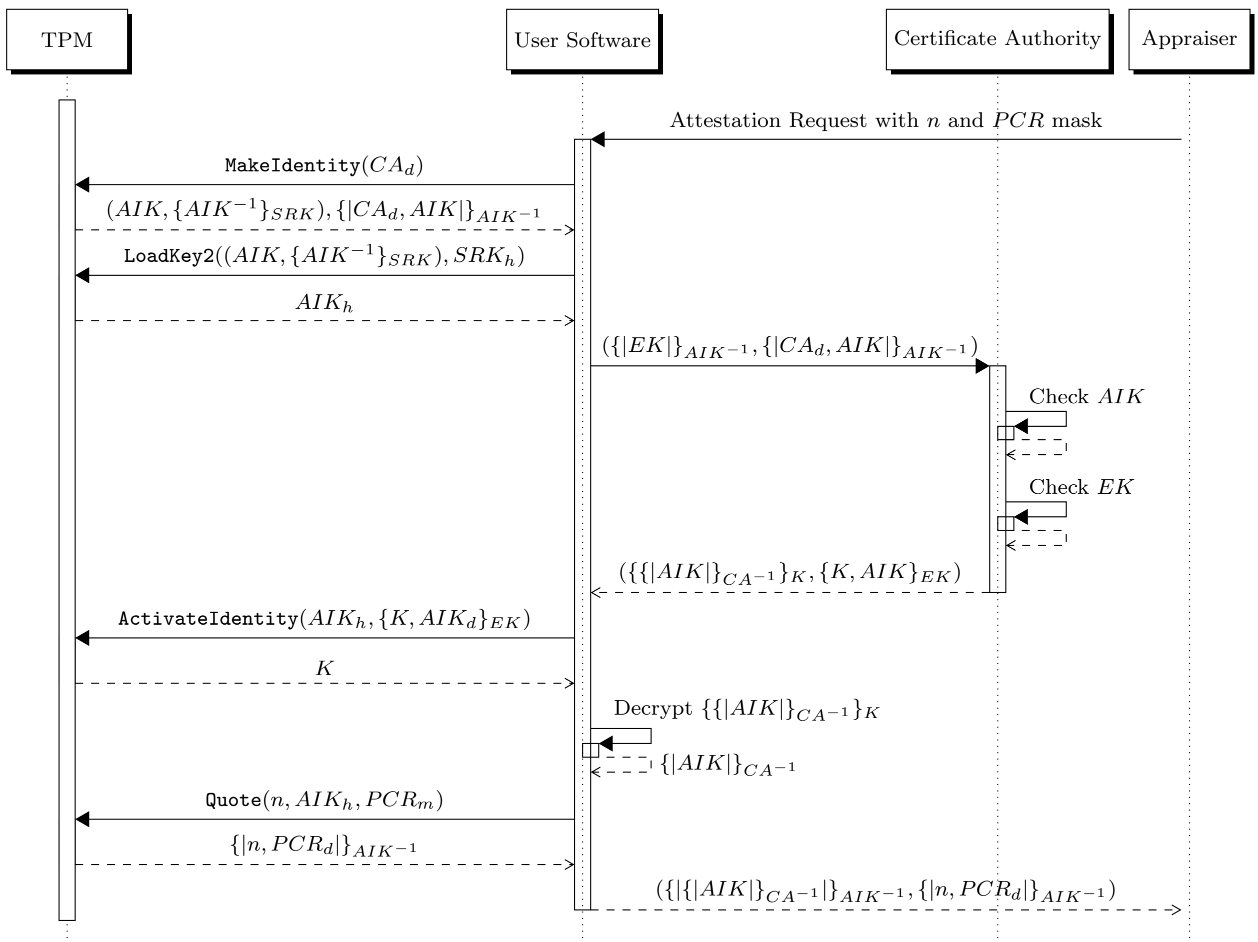

Figure 1: Sequence Diagram for the Privacy CA protocol as described by Ryan

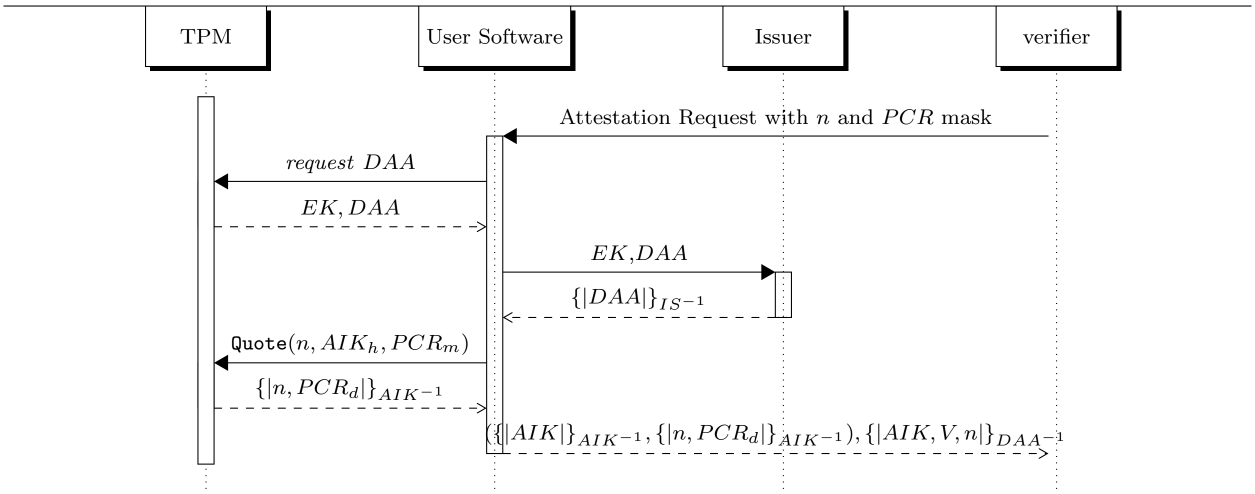

Figure 2: Basic DAA protocol execution **DRAFT**.

3 Direct Anonymous Attestation

Don't actually deliver the DAA certificate signed by the issuer. Instead, a proof that:

- User generated signature by DAA on AIK, V, and n ($\{|AIK, V, n|\}_{DAA^{-1}}$). V identifiers the target verifier. AIK is the identity being verified as coming from a legitimate TPM and used for the quote. n is a nonce to ensure freshness.
- User possess the certified DAA from the issuer ($\{|DAA|\}_{IS^{-1}}$), but does not send it to the verifier.

3.1 Camenisch-Lysyanskaya Signature Scheme

Define the public key of the DAA signature issuer as:

$$IS \equiv (n, a, b, d)$$

where n is an RSA modulus. Then define the DAA signature on message x:

$$\{|x|\}_{DAA^{-1}} \equiv (c, e, s)$$

such that:

$$c^e = a^x b^s d \bmod n \tag{1}$$

The DAA signature gives c, e, and s while the DAA public key gives n, a, b, and d. Knowing x, its DAA signature, and the DAA public key for the signer allows checking equation 1. However, we don't want to hand out the signature on x. Can we instead prove that we have it without handing it out?

For DAA, the public key associated with a TPM is

$$DAA \equiv a^x \bmod n$$

where x is the secret key of the TPM. Is x equal to AIK^{-1} or EK^{-1}?

The protocol convinces the verifier of possession of certificate on secret message. Specifically, Prover wants to convince Verifier that they know x such that $y = a^x$ where the verifier learns only y and a:

- r is random
- c is random
- Verifier checks $t = y^c a^s$ (I think)

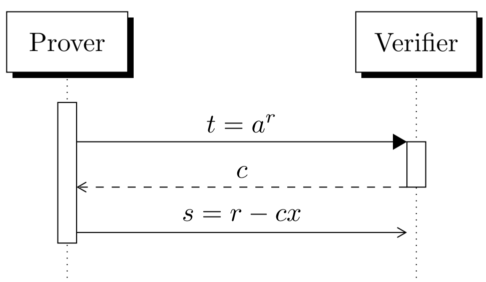

Figure 3: Proving knowledge of certificate on secret message

Prover wants to convince verifier that they know x_1 and x_2 such that $y = a^{x_1} b^{x_2}$ where the verifier learns only y, a and b:

- r_1, r_2 are random
- c is random
- Verifier checks $t = y^c a^{s_1} b^{s_2}$ (I think)

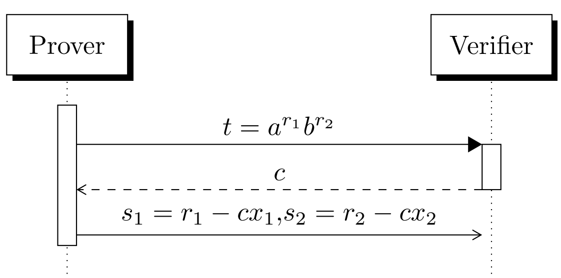

Figure 4: Proving knowledge of certificate on secret message

Recall x and (c, e, s) such that $c^e = a^x b^s d \bmod n$.

- blind cert: compute $c' = cb^{s'} \bmod n$ with random s'. $c'^e = a^x b^{s^*} d \bmod n$
- send c' to verifier
- prove knowledge of x, e, s^* such that $d = c'^e a^{-x} b^{-s^*} \bmod n$

4 Glossary

$\{|M|\}_{K^{-1}}$ — M signed with private K.

$\{M\}_K$ — M encrypted with public K.

References

—. *TCG TPM Specification*. Trusted Computing Group, 3885 SW 153rd Drive, Beaverton, OR 97006, version 1.2 revision 103 edition, July 2007. URL https://www.trustedcomputinggroup.org/resources/tpm_main_specification/.

J. Camenisch and A. Lysyanskaya. A signature scheme with efficient protocols. In S. Cimato, G. Persiano, and C. Galdi, editors, *Security in Communication Networks*, volume 2576 of *Lecture Notes in Computer Science*, pages 268–289. Springer Berlin / Heidelberg, 2003. ISBN 978-3-540-00420-2.

M. Ryan. Introduction to the tpm 1.2. Draft Report, March 2009.