Covert Identity Information in Direct Anonymous Attestation (DAA)

Carsten Rudolph

Fraunhofer Institute for Secure Information Technology – SIT,
Rheinstrasse 75, Darmstadt, Germany, Carsten.Rudolph@sit.fraunhofer.de

Abstract. Direct anonymous attestation (DAA) is a practical and efficient protocol for authenticated attestation with satisfaction of strong privacy requirements. This recently developed protocol is already adopted by the Trusted Computing Group and included in the standardized trusted platform module TPM. This paper shows that the main privacy goal of DAA can be violated by the inclusion of covert identity information. This problem is very relevant, as the privacy attack is both efficient and very difficult to detect.

1 Introduction

Authenticity and strong privacy seem to be obviously contradictory requirements. One cannot remain anonymous and authenticate oneself at the same time. Pseudonymous authentication based on certification authorities require strong trust into these authorities, as they can link pseudonymous identities to real identities. Recently sophisticated approaches have been developed in order to achieve pseudonymous authentication while preserving the privacy of the entity to be authenticated and relying on weaker trust assumptions for the certification authorities involved in the process. One practical and efficient protocol is called *direct anonymous authentication (DAA)* and was proposed by Brickell, Camenisch and Chen [2, 3]. A straightforward application for DAA lies in the area of *trusted computing*, where a platform is said to be trusted when it can attest to comply with a particular standard and runs with a particular configuration. The consequences to the privacy of the platform owner have been widely discussed (e.g. [6, 1]). Among other things, the TPM is criticised as being a tool for collecting personal information and for supporting data miners in generating consumer profiles.

The DAA protocol is based on Camenisch-Lysyanskaya signatures [4]. The main goal of DAA is to provide strong authentication and privacy even if the certification authority (here called *DAA issuer*) and the verifier collude. The issuer and the verifier could even be the same entity. By using DAA a prover can remain anonymous (or pseudonymous), and nevertheless, provide evidence by which is attested that it is using certified trusted hardware (e.g. a TPM protected platform) and pseudonymous identification without the possibility of tracing and linking actions of one particular TPM. DAA seems to provide a reasonable solution for the privacy problems associated with TPMs. DAA was quickly adopted by the Trusted Computing Group (TCG) and included in the TCG standard for the trusted platform module (TPM) [5].

In this paper we show that the issuer can use the join protocol of DAA to include covert identity information into the public key used for certification and DAA signature verification. This inclusion of identity information does not constitute an attack on the cryptographic mechanisms used by DAA. The security proofs for DAA rely on the (implicit) assumption that no data in the DAA protocol contains covert identity information. The presented privacy attack nicely shows that even such a highly sophisticated protocol can be easily misused to successfully undermine the security requirements. Furthermore, the attack cannot be detected by any honest participant of the protocol.

2 Direct anonymous attestation

We give a high level description of DAA. This description contains only those details required to understand the privacy problem explained in the subsequent section. In particular, all details concerned with the detection of rogue TPMs and revocation are not relevant for the attack and therefore omitted. More details of the scheme, explanation of the underlying cryptographic algorithms and security proofs can be found in the original publication [2] and in a full version of [2] available at http://eprint.iacr.org/2004.

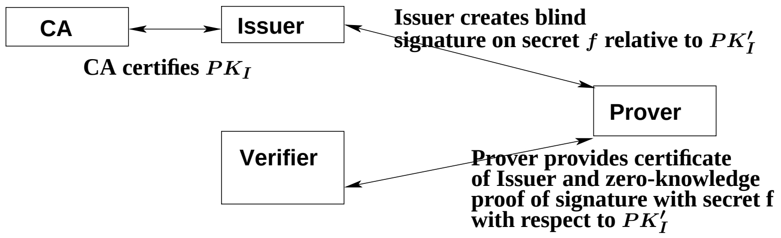
CA
Issuer
Issuer creates blind signature on secret f relative to PK_I'
CA certifies PK_I
Prover
Verifier
Prover provides certificate of Issuer and zero-knowledge proof of signature with secret f with respect to PK_I'

Fig. 1. DAA overview

The scenario for DAA requires four actors as depicted in Figure 1 showing an overview of the DAA protocol. A certification authority CA certifies long-term public keys (PK_I). The role of this CA corresponds to the role of certification authorities in standard public key infrastructures. A second (low-security) certification instance, the issuer, provides blindly signed credentials to be used for DAA signatures with respect to a DAA public key (PK_I'). The main actors in the DAA scheme are prover and verifier. In a trusted computing scenario the verifier might require that a prover provides evidence that its platform is equipped with a trusted platform module TPM and is in conformity with the particular standard. Further, the verifier might request an authentic report on the current configuration of the platform. In this process, it can be in the interest of the prover to stay anonymous or at least to provide only pseudonymous identification. It shall not be possible for different verifiers to link actions by the prover using the same platform, i.e. initiated using a particular TPM. Furthermore, no verifier should learn any identity information from the DAA signature, apart from the pseudonym used for this DAA exchange. Please note that DAA allows for different modes distinguished by the level of privacy for the prover. This level ranges from total anonymity to a pseudonymous authentication that allows the verifier to link different actions. The verifier decides on the level of privacy.

The DAA scheme consists of two main phases, the join protocol and the actual sign/verify protocol.

The goal of the join protocol as shown in Figure 2 is that the issuer provides to the prover a blind signature that can be used to prove that a secret f was generated by the prover (e.g. by the prover's TPM). In the context of trusted computing, this credential is a critical component in the scheme, as with knowledge of the credential one can "impersonate" TCG-compliant hardware. Therefore, the credential must stay inside the TPM, even during the signature verification process.

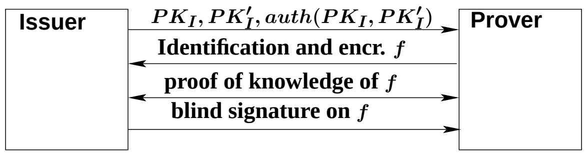
Issuer
$PK_I, PK_I', auth(PK_I, PK_I')$
Prover
Identification and encr. f
proof of knowledge of f
blind signature on f

Fig. 2. DAA join protocol

The main steps of the join protocol can be summarized as follows:

1. Issuer generates a public key PK_I' and authorises this key with its own long-term public key PK_I which is certified by the CA.
2. Issuer proves to the prover that PK_I' is correctly generated to satisfy the conditions for secure use in DAA (see [2] for details).
3. Prover provides identity information to the issuer and authenticates itself. A TPM, for example, uses its endorsement key EK that uniquely identifies a particular TPM.
4. Prover chooses secret f and sends f encrypted for blind signature to issuer.
5. Issuer generates a blind signature on f and sends it to the prover.

In the DAA sign protocol, the prover (i.e. the prover's platform and TPM) signs a message m, (in the trusted computing context this might be an anonymous attestation identity key AIK). The following description shows the principal steps of the sign protocol.

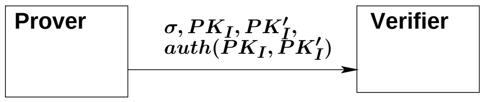

Fig. 3. DAA sign protocol

1. The prover and verifier agree on the parameters to be signed. One of the parameters is the *name* of the platform in the protocol. If this name computed from a random number, the sign is completely anonymous, if the name is computed from a number chosen by the verifier, the sign constitutes a pseudonymous identification and the verifier can link different instances of the sign protocol by the same platform. However, the verifier cannot link these with DAA instances the platform has run with other verifiers.
2. The prover produces a signature of knowledge σ over the name and a few other parameters expressing that σ was computed using the secret f blindly certified by the issuer.
3. The verifier receives PK'_I and σ and can now verify this signature w.r.t PK'_I. In the context of trusted computing the verifier can now conclude that the issuer has checked the endorsement key EK (and conformance credentials), that secret F is stored within this TPM, and that message m belongs to this TPM.

3 Insertion of covert identity information into the issuer's DAA public key

A central security proposition of DAA is that nobody can link DAA signatures to particular instances of the join protocol, i.e. the identity of the TPM is protected and different actions of one TPM with different verifiers cannot be linked even when verifiers and issuer collude. In the remainder of this paper the term *identity information* denotes all information that weakens this property.

Before the start of the join phase, the issuer computes the DAA public key PK'_I. During the join protocol the TPM's endorsement key EK is used to identify the TPM. This endorsement key uniquely identifies a TPM. Obviously, if the issuer is able to create a unique public key for each TPM, every DAA signature of the TPM can be linked back by the issuer to the TPM's EK. As the DAA public keys are self-certified by the issuer using its long-term key PK_I, the issuer can generate these keys without the help of any other entity. Thus, by using different keys PK'_I for different TPMs the issuer has effectively included identity information into the public keys. Every entity taking part in the protocol can now link a PK'_I to a particular TPM. The DAA protocol itself remains totally unchanged and neither the platform nor the TPM is able to tell whether the issuer has associated the public key with any identity information. However, if verifiers are aware that the issuer has generated unique public keys for each TPM they can match this information and tell which transactions where made using the same TPM. By colluding with the issuer a verifier can even match particular actions with a TPM's public endorsement key, thus totally breaking the pseudonymity or anonymity of the protocol.

4 Relevance of the attack

The relevance of the attack depends on several factors, most importantly: plausibility of the attack scenario, difficulty of detection by honest participants, efficiency of the attack, and availability of fixes to prevent the attack.

First, the attack scenario is very plausible. It attacks one of the main advantages of DAA. The issuer is not supposed to be as secure as a "classical" PKI certification authority. The only trust required is that the issuer will not issue wrong certificates, but there are no privacy requirements. Issuer and verifier could even be the same entity. Thus, the issuer can indeed be interested in including covered identity information into the DAA public key. Furthermore, one can imagine a situation where an issuer generates a completely new public key for each DAA join protocol without the intention to provide any identity information. In principal, such a behaviour should increase the security of the protocol. However, by being able to distinguish different instances of DAA join, verifiers are able to identify actions by particular provers even without any malicious behaviour of the issuer or any collusion between issuer and verifier.

Second, the issuer does not act in contradiction to the DAA protocol. Consequently, nobody can tell whether any of the components of the DAA public key is linked to any identity information. Only verifiers with access to many DAA public keys from the same issuer can detect the embedded information. However, verifiers might be interested in collecting identity information or might collude with the issuer. Therefore, detection of the privacy attack is very difficult.

Third, the generation of a large number of public keys can be done very efficiently. During the setup stage the issuer first selects a RSA modulus $n = pq$ with $p = 2p' + 1, q = 2q' + 1$. Then a random generator g' of the group of quadratic residues modulo n is chosen. Next it chooses six random integers $x_0, x_1, x_z, x_s, x_h, x_g \in [1, p'q']$ and computes
$g := {g'}^{x_g} \bmod n,\ h := {g'}^{x_h} \bmod n,\ \ S := h^{x_s} \bmod n,$
$Z := h^{x_z} \bmod n,\ R_0 := S^{x_0} \bmod n,\ R_1 := S^{x_1} \bmod n.$

The values g, h, S, Z, R_0, R_1 are components of the issuer's DAA public key and can therefore be used by the issuer to link the key to the identity information. The issuer only needs to go through the setup process once and then independently compute, for example, a unique value $R_1 := S^{x_1} \bmod n$ for each TPM. Thus, only a minimal amount of additional computation (one modular exponentiation) is required for each join protocol.

Finally, there is no obvious change to DAA that can prevent the attack. One possibility is to require higher security from the issuer (audits, control, etc) losing one big advantage of the protocol. However, even if only a small number of public keys is used, it is still possible for the issuer and the verifiers to distinguish several groups of users by using a specific key for each particular group.

For the trusted computing scenario the consequences have to be considered very problematic. All DAA signatures by a particular TPM could be linked to the identity of this TPM. The TPM has no control over the issuer's public key. Thus, the TPM cannot detect and consequently cannot prevent the inclusion of covert identity information. If verifiers are aware of this covert identity information, they can track TPM actions and several verifiers can tell which actions where executed by the same TPM without actively colluding with the issuer.

5 Conclusions

DAA is a very sophisticated approach to achieve some kind of authenticity and security without violating privacy requirements. However, as the example in this paper shows covert identity information can be easily embedded by the DAA issuer. This breaks the main privacy goals of the protocol. The consequences for a TPM owner's privacy largely depends on the particular applications and circumstances in which the protocol is used. Nevertheless, the relevance of the problem is increased by the fact that the degree of required collusion between verifier and DAA issuer is quite small. As soon as the verifier realises that the issuer uses a unique public key for every EK, all actions by the same TPM can be linked by the verifier without any contribution of the issuer. Only if the platform of the prover shall be identified by its EK, issuer and verifier have to collude as this information can only be provided by the issuer. However, this kind of collusion is realistic in the case of DAA, because DAA was explicitly developed to also support business scenarios where issuer and verifier are the same entity.

References

1. R. Anderson. 'trusted computing' frequently asked questions. http://www.cl.cam.ac.uk/~rja14/tcpa-faq.html, 2003.
2. E. Brickell, J. Camenisch, and L. Chen. Direct anonymous attestation. In *11th ACM Conference on Computer and Communications Security*. ACM Press, 2004.
3. J. Camenisch. Better privacy for trusted computing platforms. In *9th European Symposium On Research in Computer Security (ESORICS 2004)*, 2004.
4. J. Camenisch and A. Lysyanskaya. A signature scheme with efficient protocols. In *Security in Communication Networks, Third International Conference, SCN 2003*, volume 2576 of *Lecture Notes in Computer Science*, pages 268–289. Springer Verlag, 2003.
5. Trusted Computing Group. TCG TPM Specification 1.2 revision 94. www.trustedcomputing.org, 2006.
6. M. Hansen. A double-edged blade - on trusted computing's impact on privacy. In *Datenschutz und Datensicherheit*, pages 525–528, 2004.