Privacy-Aware Device Identifier through a Trusted Web Service

Marcelo da Cruz Pinto, Ricardo Morin,
Maria Emilia Torino, and Danny Varner

Intel Corporation
{juan.m.da.cruz.pinto,ricardo.a.morin,maria.e.torino,
danny.varner}@intel.com

Abstract. Device identifiers can be used to enhance authentication mechanisms for conducting online business. However, personal computers (PC) today are not equipped with standardized, privacy-aware hardware-based device identifiers for general use. This paper describes the implementation of privacy-aware device identifiers using the capabilities of the Trusted Platform Module (TPM) and extending the trust boundary of the device using a Web Service. It also describes a case study based on a device reputation service.

Keywords: device identifiers, identity, privacy, protected execution, reputation systems, TPM, Web Services.

1 Introduction

Internet usage is permeating many aspects of our life. E-Commerce, online communities, media services and many other sectors are growing at a rapid pace. At the same time, issues such as fraud, identity theft and spoofing are real problems [7]. For example, in 2009 the online revenue loss due to fraud has been estimated at USD 3.3 billion [5].

To help address these issues, multifactor authentication can be used to enhance the ability of identifying parties engaged in online business, but it usually requires the issuance of tokens or other devices as a supplemental mechanism for identity proof [9]. One way of enhancing authentication is to use platform-specific device identifiers in lieu of tokens, but these are not standardized in commodity personal computers (PC). This leads to the usage of proprietary software-based methods that attempt to generate device identifiers through multiple sources such as IP address, MAC address, and browser cookies, all of which can be easily spoofed. It is therefore desirable to have the ability to establish and use device identifiers that are hard to tamper with by strongly rooting them in the device's hardware.

One of the key challenges with unique device identifiers is how to preserve the user's privacy, i.e., ensuring complete control over the usage of artifacts that may expose their identity. Users should be able to opt-in and opt-out of requests to release device identifiers for authentication or tracking applications. In addition,

P.P. Maglio et al. (Eds.): ICSOC 2010, LNCS 6470, pp. 649–656, 2010.

device identifiers should be protected from being used to link interactions across multiple sites or domains without the explicit consent of the user.

Today, an increasing number of PCs are being furnished with a Trusted Platform Module (TPM) [6]. The TPM provides hardware assisted key generation, secure storage of keys and hashes, and basic cryptographic functions. However, the TPM does not provide a general purpose protected execution environment for deploying algorithms needed to deliver privacy aware device identifiers. While it is possible to establish multiple, unlinkable device identifiers using the TPM through multiple attestation keys [4], the system is vulnerable to malware software extracting various identifiers or keys and sending them to a remote server for linking, thus compromising the users privacy.

In this paper, we describe the implementation of a strong device identifier (Device ID) that could be used by service providers to make decisions on the trustworthiness of transactions emanating from a device, while protecting the privacy of the user. We explore extending the trust boundary of a TPM-equipped device out to a trusted Web Service hosted in the cloud thus leveraging an execution environment that is protected from local attacks.

Our approach is privacy-aware because it a) requires user opt-in, b) protects device unique keys generated by the TPM, c) ensures that Device IDs are application, site and/or domain specific under the control of the user, and d) users can opt-out by removing Device IDs if desired.

To illustrate the proposed solution we describe a use case based on a hypothetical device reputation system that uses a tamper-proof persistent Device ID as the basis for its identity system.

2 Requirements

Access control: The user shall be in control of the Device ID feature (e.g. opt-int/opt-out and access control for third parties).

Unlinkability: The Device ID shall be site and/or domain specific in that no two (or more) third parties should be able to link users from their respective domains by solely using the Device ID feature. This prevents unscrupulous third parties from damaging the privacy of the user.

Malware protection: Unprotected Device IDs could lead to identity theft and damaging the user's reputation. The solution needs to mitigate these attacks.

Context: The Device ID itself is of no use without some context information such as longevity (i.e., the time elapsed since the creation of identity). This can help a third party better infer trust based on historical records.

Trust & Integrity: In order for a third party to make a trustworthy decision based on the Device ID, the solution needs to provide a proof of origin and tamper evidence in the form of a digital signature.

Confidentiality: The Device ID information shall be protected against eavesdropping and replay attacks while transferred from the device to a third party.

Integration and ease of deployment: The proposed solution shall be easy to integrate to existing Web applications via standards such as XML and HTTP.

3 Architecture

The proposed architecture is depicted in Figure 1. The *Device* that requires the identity feature contains a Web browser that acts as a nexus between the Security Hardware (SH), the *Device Identity Provider* (DIP) and the *Service Provider* (SP).

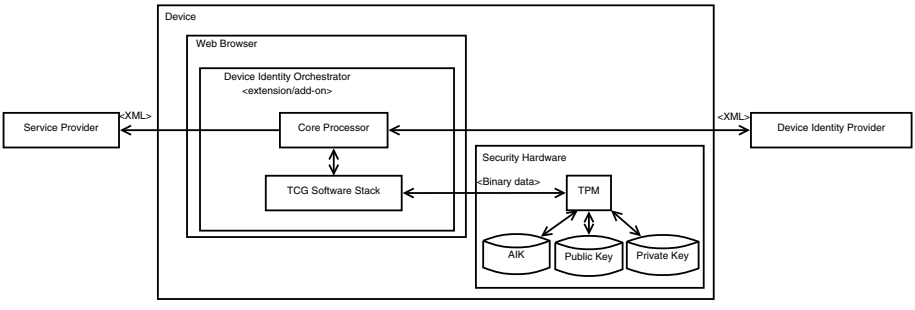

Fig. 1. Proposed architecture

The SH is a component that provides support for digital signature, protection of the involved private keys, and a proof of the hardware support behind it. A good example of SH is the TPM [2].

The DIP is a Web Service that takes the digital signature value generated by the Device (using the SH) and binds this information to some calculation performed in the protected environment of the Web Service. In our solution, the DIP calculates a "pseudonym" identity for the device to meet the *Unlinkability* and *Context* requirements listed in section 2.

The SP is a Web entity that is interested in using the Device ID produced by the DIP for weighing some decision. This entity could be an e-Commerce Web site, social network, banking Web site, etc. that requires strong authentication or an intermediate entity which manages device reputation for the aforementioned (in section 6 we present this as a case study).

The Web browser orchestrates these components while also providing an overall smooth user experience. This configuration effectively extends the trust boundary of the device to the cloud, more specifically to the DIP. The DIP acts as an extension of the device by performing some calculation that would otherwise require a protected execution environment in the device.

4 Security Protocol

As mentioned in section 3, we separate the DIP from the actual platform by using a Web Service. Therefore, we need a secure communication protocol which ensures the correct binding between these components. We achieve this binding by using TPM's support for attestation, and creating a pseudonym identity for the device in the context of the requesting SP. In this section we describe this protocol using a simplified version of the notation from [8].

Declarations

- Let D be the device
- Let SP be the Service Provider
- Let DIP be the Device Identity Provider
- Let T_A be the timestamp for the identity request
- Let N_A be the nonce for the identity request
- Let T_E be the timestamp in which the device was first seen by the DIP ("Enrollment" timestamp)
- Let T_R be the timestamp of the last identity reset
- Let T_S be the current time on the DIP server
- Let RC be the reset count (times the user has requested an identity reset)
- Let $DevSeed$ be a unique number which represents the universal device identity of D for a given DIP. This unique number is randomly generated the first time D accesses the services of the DIP
- Let $pub(X)$ be the public key and $priv(X)$ the private (secret) key of entity X

Pre-conditions

- At least one TPM AIK has been provisioned on D, and a signature key has been generated which is rooted on said AIK. $(pub(D), priv(D))$ refers to the aforementioned signature key pair.
- SP has generated a private/public key pair, and has shared the public portion (or certificate) with the DIP service. $(pub(SP), priv(SP))$ refers to the aforementioned key pair.
- DIP has generated a private/public key pair, and has shared the public portion (or certificate) with the SP. $(pub(DIP), priv(DIP))$ refers to the aforementioned key pair.
- Given the two previous pre-conditions, SP trusts DIP and vice-versa

Protocol

1. $SP \rightarrow D : \{N_A, T_A\}_{priv(SP)}, pub(SP)$
2. $D \rightarrow DIP : \{N_A, T_A\}_{priv(SP)}, \{N_A, T_A\}_{priv(D)}, \{pub(D)\}_{pub(DIP)}$
3. $DIP \rightarrow D :$
   $\{\{N_A, T_E, T_R, RC, hash(DevSeed|pub(SP)), T_S\}_{pub(SP)}\}_{priv(DIP)}$

In the above flow, the Device ID of device D requested by service provider SP in the context of Device Identity Provider DIP is denoted as $hash(DevSeed|pub(SP))$: This enforces the *Unlinkability* requirement from section 2, since two SPs who share information about device identities will not be able to correlate any transactions solely based on this identity. The $DevSeed$ value will be generated by the DIP in step 2, only for devices that are new to the DIP. Once generated, this value will be stored in the DIP's database using D's public key as the index, and recalled every time the same device requests the identity service. This protocol also ensures the requirements of *Trust & Integrity* and *Confidentiality* by using digital signatures and encryption, and the *Context* requirement is met by attaching the timestamps and reset count information corresponding to the device.

5 Implementation

Our implementation uses the TPM as the SH component. We are particularly interested in the Attestation Identity Key (AIK)/Direct Anonymous Attestation (DAA) feature of the TPM specification, which provides support for remote attestation. We use this capability as the proof that the SH component is protecting the signature keys used by the DIP to derive trust in the device.

We assume that the TPM chip inside a device will be provisioned with at least one attestation key. For our implementation, we use the AIK scheme and a Privacy CA (Certification Authority) to attest the TPM hardware, but our architecture does not impose this model. Since the AIK cannot be used to sign artifacts generated outside the TPM hardware, the TPM is instructed to create a generic signing key and sign it with the AIK. The key pair $(AIK, SigningKey)$ is our effective trust anchor for the device.

The proposed implementation uses a Web browser add-on for orchestrating the information flow between the SP and the DIP, including the communication with the TPM for the digital signature primitive.

The DIP exposes a Web Service interface, which provides a TPM AIK signature consumption service and creates unlinkable identifiers for each SP.

TCG Software Stack. A TCG Software Stack (TSS) [1] implementation was used as the interface to the TPM for creating and protecting the Device IDs and represents the SH as described in section 3. The TSS can create a complete PKI (Public Key Infrastructure) tree using an AIK (Attestation Identity Key) as the trust anchor, but for the purpose of this paper and to fulfill the requirement of *Malware Protection* by having the Device ID protected, we only require a single RSA key, issued by an AIK. This TPM public key, which will be pseudonymized by the DIP, forms the basis of the Device ID. It is important to note that this key is never exposed to an SP.

Web browser extension. The Web browser extension is responsible for orchestrating the device identity generation process through the *Device Identity Orchestrator*(DIO) and for providing device identity management abilities through a graphical interface. This allows for standard Web protocols to be used fulfilling the requirement of *Integration* as described in section 2.

This extension also includes improvements in user experience, facilitating the opt-in and opt-out from the service as desired, meeting the *Access control* requirement also described in section 2.

The DIO is made up of two main components that manage the entire device identity generation process: A core communication protocol stack, which ensures that the messages from the SP and DIP are exchanged correctly (see message list in section 4), and the TSS wrapper which allows the interaction with the SH to access the device credentials.

Device Identity Provider (DIP). It provides the implementation of the message exchange protocol described in section 4 as a RESTful Web Service,

mapping all the messages to XML structures. All of the cryptography described in section 4 is implemented with XML Security standards (see [3]).

The DIP verifies the identity of the SP and of the Device. SPs participating in the device identification service must enroll with the DIP through a prior exchange of certificates in order to be considered trusted. The certificate embedded in the XML Digital Signature is then extracted and looked up in the list of trusted SPs to ensure communication was initiated by a valid SP.

The DIP authenticates the identity of the device itself using embedded TPM based credentials rooted into the Privacy CA. The TPM public key is used as a primary key for accessing the DIP database and storing data associated to the device (timestamps and reset count as described in section 4).

A possible extension of this work would be to maintain a list of all identity requests by Service Providers to give the user full transparency in how their Device ID is being accessed.

6 Case Study: A Device Reputation Example

As online fraud costs to online vendors is significant [5] there is a vested community interest in the prevention of processing fraudulent transactions through secure data sharing without impeding on user privacy. To put our proposed system in context, we developed a case study providing an end-to-end scenario demonstrating the strength and value of the solution coupled with a device reputation service. The result is a reputation service that is unique in two ways. The first is user privacy, as the solution is designed to be opt-in only through collecting and rewarding positive behavior. The second is device trust, since the online SP can guarantee the device identity is legitimate.

The case study consists of four actors as seen in Fig. 2: a reputation service, an online vendor subscribed to the reputation service, a client device operated by a user wishing to execute a transaction with the online vendor, and the DIP.

The device reputation service collects data on the Device ID longevity and transaction history of the device to determine the device's reputation in online commerce. The user is in effect relinquishing control of the longevity of their Device ID tied to online transactions to provide evidence the device is not involved in fraudulent activities. The benefit for the user in relinquishing this control can simply be gained confidence in initiating transactions within a trusted web of

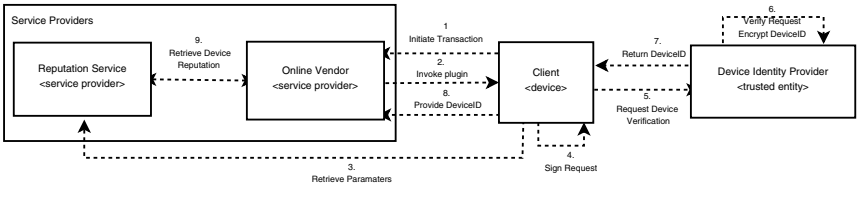

Fig. 2. Case study

SPs. This trusted web of the DIP, SPs subscribed to the DIP, and the user will be defined as a 'trusted computing community.'

The user wishing to participate in the trusted computing community installs a browser plug-in provided by the DIP, the user consents to the opt-in policy and initiates the process of obtaining device identity trust. The wizard walks the user through provisioning of the TPM and provides the user the ability to set an access control list on their Device ID. The plug-in will also expose a menu option for the user to fully reset their device identity, with the associated consequence of resetting any reputation they have built up. The user is now ready to initiate a transaction with an online vendor participating in the trusted computing community. It is worth noting the reputation service and the online vendor can be a single entity or separate SPs. By separating the reputation service from the online vendor, it provides the benefit that a single online vendor can access the reputation history of the device in question for all the device's transactions within the trusted computing community.

The user initiates an online transaction with the online vendor and the device trust process is activated. A nonce from the reputation service is generated, the TPM performs the signature and attestation process, the payload is signed by the reputation service certificate and sent to the DIP. The DIP performs the device verification and then hashes and encrypts the Device ID with the reputation service certificate in order to send it back to the client to be embedded in the response to the online vendor. At this point the online vendor has the verified Device ID in possession however is unable to access any details as it is encrypted by the reputation service key. The online vendor forwards this data to the reputation service to receive a user consented feedback report on the trusted Device ID. The reputation report is generated by the number of successful transactions the device has completed and the longevity of the Device ID. After the transaction is complete the online vendor provides feedback on the results of the transaction to the reputation service, affecting the device's overall reputation.

This case study demonstrates how: a) an online vendor can achieve a trusted network based reputation report on a device without obtaining any additional personally identifiable information; b) the user is able to maintain full control over their privacy through an opt-in only approach and the ability to reset the identity/reputation of the device; and c) the reputation service maintains data integrity by only tracking data on verified devices.

7 Conclusions

We introduced a privacy-aware Device ID system that can be implemented using off-the-shelf TPM capabilities present in many commodity PCs, enhanced with a Web Service that extends the trust boundary of the device. The Web Service is used to protect the user's privacy in addition to providing additional security to prevent tampering attacks. We described an end-to-end implementation of a fully functional proof-of-concept in the context of a device reputation application that can be deployed with today's available technologies. Our contribution is

innovative in two ways. First, through a security protocol, we extend the trust boundary of a device to a Web Service that can be hosted in the cloud. Second, using cryptographic techniques we generate unlinkable identities that are fully controllable by the user.

Future work opportunities would involve the replacement of the trusted external Web Service by utilizing a local secure execution environment. This capability is not available in commodity PC today. However, the system we describe would provide a valuable application of such capability. By describing this solution and its applications, we hope to encourage PC manufacturers to consider the inclusion of secure execution environments in their product offerings in the future.

References

1. Trusted computing group software stack (2010), http://www.trustedcomputinggroup.org/developers/software_stack
2. Trusted platform module (2010), http://www.trustedcomputinggroup.org/developers/trusted_platform_module
3. Xml digital signature (2010), http://www.xml.com/pub/a/2001/08/08/xmldsig.html
4. Balfe, S., Lakhani, A.D., Paterson, K.G.: Trusted computing: Providing security for peer-to-peer networks. In: IEEE International Conference on Peer-to-Peer Computing, pp. 117–124 (2005)
5. CyberSource: 2010 fraud report (2010), http://www.cybersource.com/cgi-bin/resource_center/resources.cgi
6. McFadden, T.: Tpm matrix (2006), http://www.tonymcfadden.net/tpmvendors_arc.html
7. Ahamad, M., et al: Emerging cyber threats report for 2009 (2008), http://www.gtisc.gatech.edu/pdf/CyberThreatsReport2009.pdf
8. Needham, R.M., Schroeder, M.D.: Using encryption for authentication in large networks of computers. Commun. ACM 21(12), 993–999 (1978)
9. O'Gorman, L.: Comparing passwords, tokens, and biometrics for user authentication. Proceedings of the IEEE 91(12), 2021–2040 (2003)