Trusted Computing Enhanced OpenID

Andreas Leicher and Andreas U. Schmidt
Novalyst IT AG
Robert-Bosch-Straße 38,
61184 Karben, Germany
Email: {andreas.schmidt,
andreas.leicher}@novalyst.de

Yogendra Shah and Inhyok Cha
InterDigital Communications, LLC
781 Third Avenue
King of Prussia, PA 19406, USA
Email: {yogendra.shah,
inhyok.cha}@interdigital.com

Abstract

Trusted Computing, used as a security technology, can establish trust between multiple parties. One implementation of Trusted Computing Technology standardized by the Trusted Computing Group is the Trusted Platform Module (TPM). We build on the security provided by the TPM to create a trusted variant of Identity Management Systems based on the popular OpenID protocol. We show that it is feasible to bind OpenID identities to the trustworthiness of the device. Our concept and implementation builds on previous work which showed that Trusted Computing can be used to create tickets. In this work, we use such tickets as a building block to establish trust between the identity provider and the device.

1. Introduction

Trusted Computing (TC) is generally regarded as a protection and security technology centered on single devices. Used as a platform-neutral security infrastructure, TC offers ways to establish trust between entities that are separated by technical boundaries, e.g., different access technologies. As some concepts of TC have similarities to Identity management (IdM), we increase the security of a common lightweight IdM protocol for the internet, namely OpenID, by the use of TC technology. The main contribution is not only to bind an OpenID identifier to a single platform, and hence provide protection from remote phishing attacks, but also to provide protection from identity theft by malware or Man-In-The-Browser attacks by enabling the use of the Trusted OpenID identifier only after a successful integrity verification of the client platform.

In previous work we have presented a concept to use TC within Kerberos [1], and have also shown that identity federation between different provider domains can be supported by TC [2]. In this paper we combine integrity validation of a client system's trustworthiness with user authentication in the widely used OpenID protocol. We further demonstrate that this combination can be done efficiently in a generic demonstration environment for TC-based applications.

1.1. Trusted Computing technology

With the growing presence of computer systems in ubiquitous environments such as mobile phones, machine-to-machine communication, and sensor networks, the need for an increase in security arises. On the other hand, the enormous increase in system complexity inhibits a formal verification of the whole system. As a consequence, other means have to be established to encounter the risks and dangers to which every single system gets exposed. In a networked scenario, where multiple systems communicate, TC is a core technology to determine whether a communications partner can be trusted.

As a root of trust for the secure operation of the system, a hardware security anchor is the key to the protection of the system behavior. Establishment of the trust boundary is associated to the boot cycle of the platform and extending trust from the root to further loaded components is a central concept of TC. This means that components that are started later on, are measured by a protected entity on the platform before they are executed, for example, digest values are generated and stored protected by the root of trust. Specified by the Trusted Computing Group (TCG) this process is called authenticated boot [3]. Another embodiment called secure boot [4] adds a local enforcement engine which denies loading components whose measurement values do not match trusted reference values.

To prove trustworthiness of a system to an external party acting as a verifier, attestation mechanisms and protocols have been envisaged. They transport measurement values and verification data necessary to retrace the post-boot system state to the verifier. The trust anchor, irremovably bound

to a particular platform, is represented by the Trusted Platform Module (TPM) [5]. Together, the TPM and its platform form a trusted platform (TP). Through the TPM the TP gains a unique identity, a cryptographic engine and protected storage. A more detailed overview can be found in [1], [5]–[7].
Attestation protocols as defined by the TCG [8] rest on 1024 bit RSA Attestation Identity Keys (AIKs) acting as placeholders for the TPM identity. The Remote Attestation (RA) protocol offers pseudonymity by the use of a trusted third party, the Privacy CA (PCA), which issues an AIK certificate stating that the AIK is generated by a sound TPM within a valid platform.

Instead of using a PCA to obtain AIK certificates, the TCG has also defined Direct Anonymous Attestation [5], [8]–[11] which uses the Camenisch-Lysyanskaya signature scheme [12]. However, DAA cannot always fulfill the promise of increased privacy [13]–[15] and according to [11], [16] the level of privacy provided by DAA can be even lower than that offered by a PCA based AIK certification, since verifiers in DAA get to know the identity of the issuer that attested to a platform's conformity, whereas in the PCA scheme only the PCA's identity is revealed to the verifier. The method to equalise the privacy levels proposed in [11] introduces additional complexity and a third party, which makes practical adoption of DAA questionable. While DAA is a means to obtain a certificate for an AIK, RA is used for platform integrity verification by a remote entity.

The system state is measured using the TPM to calculate hash digest values and storing them securely in the Platform Configuration Registers (PCR). The Stored Measurement Log (SML) keeps a more extensive record of the state and can be used together with the PCR values for the validation of the platform, where the AIK signed PCR values are used to provide implicit integrity protection for the SML. This technique is based on initial work by Schneier and Kelsey for securing audit logs [17] and implemented as Integrity Measurement Architecture (IMA) [18].

1.2. Related work

IdM and Single Sign-On (SSO) are technologies which can benefit from TC. A general analysis of how TC concepts can be applied to support SSO is discussed in [16], [19]. The concept described in [19] implements the identity provider (IdP) on the TP of the user, and the IdP proves its integrity to all service providers. The scheme requires all service providers to keep a database of known reference values for the reported integrity measurements. Our approach is different in that we employ integrity verification of the user's platform centrally by the IdP which in turn authenticates users and can issue statements on the platform's trustworthiness to the service providers.

In a previous paper [1] we demonstrated how TC technology can increase the security of a ticket system, namely Kerberos, by binding the issued tickets to the client device TPM and the trustworthiness of the state of the platform to which the TPM is bound. The paper [1] also reviews further related work with regard to secure IdM. Kerberos, however, has more architectural components which makes it less efficient than other IdM solutions. Specifically, the two interactions with Authentication Server (AS) and Ticket Granting Server (TGS) add to complexity. Therefore, we looked at natural extensions of TC concepts to more compact IdM systems such as OpenID. A specific shortcoming of Trusted Kerberos is that verification of the Trusted Tickets must be implemented by every service provider, introducing changes to their applications. While being feasible for small environments with few services, this concept might not be feasible for internet services which are otherwise not in a trust relationship. Furthermore, Trusted Kerberos requires all participants, i.e. service providers, AS and TGS to maintain a shared key database for the encryption of the tickets. Clients that are registered in another realm cannot access this service provider unless the service provider registers with multiple realms.

2. Trusted OpenID protocol

The OpenID protocol does not specify any user authentication method to be used, and thus allows for different methods for user authentication. To claim an identity at the OpenID provider, several methods can be used, where the most common is the use of logon forms, where the user provides a password. In our Trusted OpenID (TOID) protocol, we replace this logon with a TPM based logon process. The user once registers an identity that is tightly bound to his/her specific platform and TPM. If he later decides to login using this identity, the OpenID provider challenges the platform to provide the correct credentials. In this case the credentials consist of a TPM generated ticket. This allows the user to login without the need for a password at the OpenID provider. A password at the user's computer can still be used to protect the identity from local attacks.

The logon is combined with an integrity verification of the specific platform. Using TPM signed system configuration values, the OpenID provider can compare the reported system state to previously generated reference values, allowing only trustworthy clients to login and claim an identity. This combined authentication and attestation allows for a fine grained access control by not only binding the authentication to a specific platform but also to a trustworthy system state. This enables OpenID to be applied to new scenarios requiring enhanced security and detection of unauthorized changes to the system.

2.1. OpenID protocol

As an open, decentralized IdM framework, OpenID [20] was developed to provide a SSO experience to users across services on the Internet. With OpenID (see Figure 1), it is possible to sign on to different services, with a single identity, called OpenID identifier, eliminating the need to create separate logins and passwords for the services the user wants to access. OpenID is supported by major companies, including AOL, Facebook, Google, Microsoft, Yahoo, etc. Reports of OpenID usage [21] count over 1 billion OpenID enabled accounts and over 9 million websites utilizing OpenID for registration and login. Recently, efforts were made by the OpenID Foundation and the US government [22] to deploy OpenID on federal websites.

The websites supporting OpenID login are referred to as Relying Parties (RP).

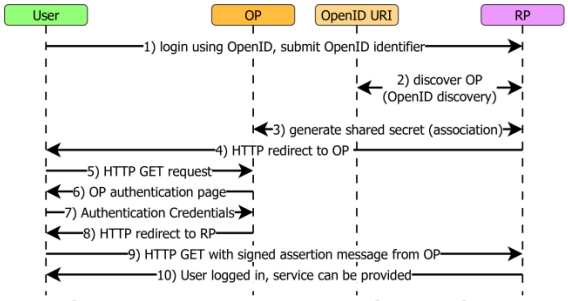

Figure 1. OpenID protocol overview.

In 2007, Eugene and Vlad Tsyrklevich [23] examined security issues of OpenID. One of the concerns when dealing with OpenID is phishing. If an attacker is able to trick users into giving away their credentials to a fake OP, owned by the attacker, he can access a broad range of RPs in the name of the legitimate user. Since OPs host the identifiers for multiple users they are a rewarding target for the attacker as they can collect multiple identifiers. Redirecting users to the fake OP can be achieved by setting up a RP which redirects the user to the fake OP instead of the original one without attacking the OP directly. Other attacks include Man-in-the-middle (MITM) attacks during the association between the RP and the OP, replay attacks, involving sniffing the session identifier of an authenticated session as well as Cross-Site-Request-Forgery (CSRF) attacks which silently log the user on to other sites once they have logged in into another OpenID site and perform actions in the user's name. Such a CSRF attack mainly relies on the fact that the OP and not the RP decides on the user login security.

2.2. Integration of TC concepts

The integration of TC concepts into the OpenID protocol allows countering some of the threats and leverage of the overall security of OpenID. Four entities are involved in the TOID authentication process: (1) the user accessing a service, (2) the OP supporting trust validation, (3) a PCA to certify AIKs from the user's TPM and (4) a RP using OpenID authentication.

We will assume that the following requirements hold on these entities: A PCA, chosen by the user has to issue an AIK certificate, i.e. a X.509 certificate with additional extensions defined by the TCG [8]. As AIKs can only be used to sign data originating from the TPM, we use the following indirection, as described in more detail in [1], [24] to obtain a Certified Signing Key (CSK): The TPM generates a new RSA key pair, whose private part is secured by the TPM. After signing it with the AIK using the internal operation TPM_CertifyKey, this key is referred to as a CSK and can subsequently be used to sign arbitrary data. The AIK certificate provides a binding of the AIK to the platform since the PCA checks the relevant TPM certificates during the AIK certification and states that this AIK is generated and stored securely in a sound TPM. The verifier, upon receipt of the AIK certificate and AIK signed CSK, is able to verify that the CSK is a TPM-secured key by deriving the trust from the PCA issued AIK certificate. After certifying an AIK at the PCA, the user can choose an OP supporting the TOID protocol to host his OpenID identity. In order to register the identity, he must provide a valid AIK certificate to the OP. It is important to note that the user does not establish any shared credentials with the PCA during AIK certification. The user generates the AIK locally inside the TPM and is able to establish a local secret, which is checked by the TPM and never shared with an outside entity, to protect the AIK from unauthorized use. The PCA is the only instance that will be able to resolve the identity to a real platform. The RP only has to enable OpenID login for his site and has to accept assertions from this OP. Different AIKs can be used by different users, where each AIK is protected by the TPM. Hence, each user in a multi-user system environment can create an own AIK and associate his OpenID identity with this AIK in the registration process with his Trusted OpenID enabled OP.

The Trusted Ticket Server: We implemented the trusted client and the enhanced OP server side of our concept in Java, based on the OpenID4Java project [25]. It is invoked from a JSP page and can be used to associate a RP and to authenticate a user. Classes and methods for integrity validation and authentication with the Trusted Ticket Server (TTS) component running on the client machine were added to enable TPM based authentication. The TTS runs as a service application on the user's machine and is responsible for the authentication towards a TOID OP. It is a trusted functional entity deployed in a Trusted Execution Environment (TEE).

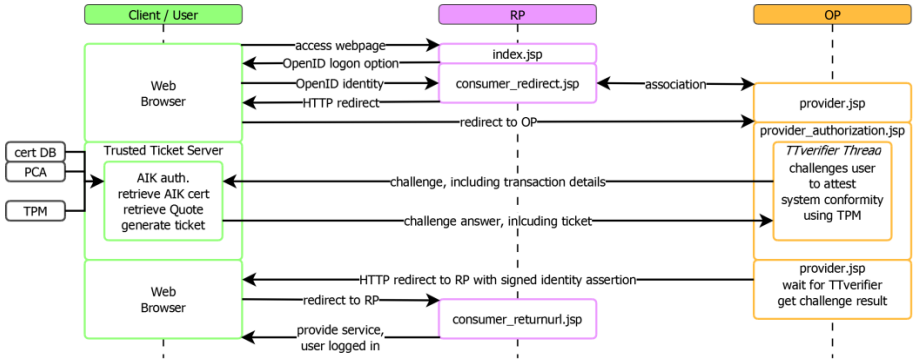

Figure 2. Overview of the protocol flow for TOID.

The TEE provides an isolated, integrity protected and secured execution environment for the TTS. In general, we can assume that such a TEE is available, provided mainly by the TPM and OS functions. Another approach, based on virtualization is presented by Gajek et. al. [26]. Our protocol consists of the following general steps, shown in Figure 2.

1) **Initial Connection to the RP:** The user accesses the website of the service provider (index.jsp). If the user wants to login using his OpenID URI, the consumer_redirect.jsp page at the RP connects to the given URI and thus retrieves the address of the OP hosting the claimed identity.

2) **Association of Service Provider to OpenID Provider:** According to the OpenID protocol, the RP associates with the OP. This includes a secure exchange of the request, the claimed identity and a return URL to which the client will be redirected by the OP if authentication is successful. These steps are performed on the server provider side using consumer_redirect.jsp and on the OpenID provider side using provider.jsp. After the association is established, the client is redirected to the webpage of the OP. The page checks if the user is already logged in, and if not redirects the user's browser to the OP login page provider_authorization.jsp, as retrieved from the user supplied identifier.

3) **Authentication of the Client:** The provider_authorization.jsp page of the OP, requests authentication and user authorization to log in to the RP. After the user accepts the login request, a new background thread starts which challenges the TTS on the client side. The provider_authorization.jsp redirects the user back to the provider.jsp page, which waits for the thread to finish and evaluates the result of the challenge.

4) **Redirection to the Service Provider:** The OP (provider.jsp) redirects the user to the consumer_returnurl.jsp page at the RP which checks that the redirect comes from the associated OP and grants access to the user.

In our implementation, the TTS must be trusted to handle AIK certificates and CSKs properly, protect the creation process of the tickets and collect and report platform validation data to the OP. Upon receipt of an authentication challenge, containing the user's OpenID identity and the service request he issued at the RP, the user is asked to explicitly allow the challenge which is shown to the user along with the transaction details in the TTS screen. In the current proof of concept implementation the user can visually compare the challenges shown in the browser window and in the TTS screen. This separate, secure user interface (UI) can protect the user from input phishing on the client side, e.g. by using the rudimentary feature of a color-coding for the secure UI as described in [27]. Another option is that the OP calculates a one time password (OTP) after successful integrity verification which is then cryptographically bound to the integrity state using the TPM sealing functions. The TTS then decrypts the OTP and shows it to the user who provides it to the OP. In principle it is possible to implement additional protection methods, such as OP authentication performed by the TTS, which could be done by the use of TLS/SSL certificates. Denial of Service (DOS) attacks against the TTS could be prevented by integrating the TTS functionality as a browser extension, which is able to communicate with the browser such that the TTS only accepts incoming requests if an authentication session is currently taking place. Depending on the desired level of security, such a trusted browser extension could also store the user decision per session, to further increase the SSO experience. If the challenge is accepted, the user is prompted to enter the password for the AIK corresponding to the given identity and to authenticate for TPM usage by giving the SRK password. The TTS then tries to retrieve a previously acquired certificate for this identity from the local certificate storage. If no certificate can be found in the local database, the user can decide to perform an AIK certification process and obtain a certificate for the AIK. Therefore, he must supply the correct owner password of the TPM. This prevents creation of rogue identities by other persons than the owner of the TPM. Especially in corporate environments, where the owner of the TPM is not the user, this allows to control the enrollment process for new OpenID identifiers in the corporate network.

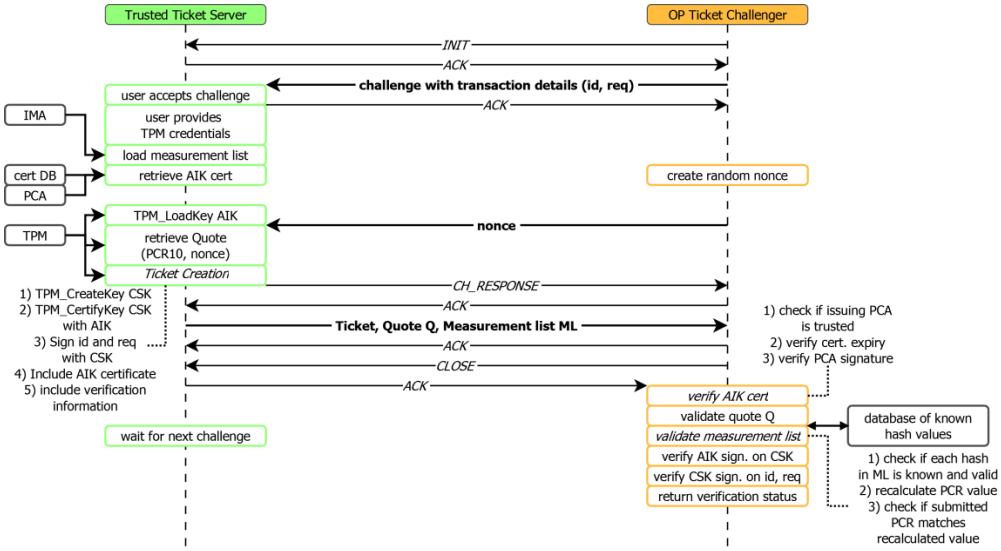

Figure 3. Detailed protocol flow for TTS - OP communication.

The TTS receives a random nonce from the challenger. An AIK-signed quote, including the nonce is retrieved from the TPM, providing a statement about the system's state. Next, the TTS creates a signed ticket which involves the creation of a CSK that can be used to sign the request and the identity. The information needed to verify the signatures is included in the ticket, so that the receiving party can easily verify the ticket. Together with the SML the ticket is sent back to the challenger. Figure 3 shows the authentication flow between the TTS and the OP.

In order to develop and implement our concepts for TOID, we set up a Trusted Demonstration Environment called ethemba [28]. The goal was to design a system in which it is possible, without the need of a physical TPM, to access all desired TPM functions. We used the TPM emulator [29] as base and to simulate a complete system we established a connection between the TPM emulator and QEMU, a virtualization emulation environment, enabling our virtual machines to execute TPM applications. Our virtual machine uses a patched kernel to support IMA [18], enabling measurement and logging of every component the kernel loads. The ethemba framework also includes a PCA implementation and support for the TCG remote attestation protocols, as described in [1]. We extended it with the implementation of all necessary OpenID functions on both client and server sides. Figure 4 shows screenshots of our OP and the TTS.

2.3. Device authentication and validation by the OP

During the TOID protocol, the OP receives the following information from the TTS: (1) the signed quote from the TPM, including the nonce as anti-replay protection, (2) the plaintext measurement file and (3) the ticket, including the signed identity and request string, the public key portion of the CSK, the AIK signature on the CSK and the AIK certificate issued by the PCA. In order to authenticate the client, the challenger first checks the validity of the AIK certificate. This validation includes the verification of the PCA signature on the AIK certificate. The challenger then has to verify the ticket, i.e. the credential chain incorporated into it. Therefore, the AIK signature on the CSK public key hash in the ticket and the CSK signature on the service request and identity in the ticket are validated. To validate the trustworthiness of the system, the challenger then checks the entries in the received SML against a database of known good values. This step is accompanied by recalculating the expected PCR value which is, in the final step, compared to the reported signed PCR value. If at any step in this process verification fails, the client will not be authenticated. In this protocol, the OP receives the platform configuration in order to verify the platform integrity. Therefore the OP must be trusted by the user not to reveal this configuration to unauthorized third parties. Since the user in OpenID already trusts the OP not to misuse his OpenID identifier, this is an additional requirement on an entity the user already has an established trust relationship with. If the RP should however get some information about the authentication mechanism, e.g. TPM based vs. non-TPM based or additional assurance of the platform integrity, additional mechanisms can apply. The OP might be offering this authentication as additional service and guarantees integrity checking for RPs by contractual agreements, and hence does explicitly disclose a platform's properties to any RP. In security demanding applications however, it can be desirable to signal the outcome of the integrity verification to RPs, which can be achieved by including this information in the OP-signed assertion

message to the RP. Such information could for example include details on the platform, the type of integrity checks performed, or along the lines of a Property-based attestation [30], by reporting the platform conformance to a certain set of properties.

TOID achieves user authentication and device trust attestation at the same time. User authentication is achieved because the user must already have pre-registered the certificates for the AIK with the OP, and the device will send data that is signed with the AIK and the CSK, which is verified by the OP. Device trust attestation is achieved because the OP can verify the signed PCR values, the SML, and the part of the ticket (identifier and request) which are all signed using the verified AIK/CSK. If both match, then the OP verifies that the client used to send the OpenID request is in fact trustworthy. If only the verification of the user and the request is achieved but the comparison of the PCR values and the measurement logs fail, then the OP knows that the device may be compromised and is in a different configuration state than expected.

2.4. Analysis of Trusted OpenID

Our solution does not require any changes to the OpenID protocol standards. The OpenID specifications do not foresee a single method for user authentication. We therefore developed an enhanced authentication concept which basically would allow OPs to differentiate amongst themselves, with one function being the assertion of enhanced security features. Such assertions are interesting for the users since they can be assured, that their OpenID identity is well protected, especially if it is HW-bound and even more important, such OPs enable RPs to rely on the information received from them, e.g. enabling banks, government and other security-demanding services to use OpenID with a whitelist of 'security-aware and certified' OPs.

Some of the mentioned vulnerabilities of the standard OpenID protocol are addressed by the OpenID Security Best Practices [31]. Phishing however remains one of the best known attacks and is a main problem for the OpenID protocol. Since the identifier is used with all RPs, the security of this identifier should be of special concern. Replacing multiple insecure passwords or a single password which is used across multiple RPs with a strongly authenticated identity reduces the spread of secret data, which is especially important if the same password is used across all RPs. Recent attacks, gathering credentials from social networks have shown that proliferation of credentials is an important security issue. Centralisation of credentials at a trustworthy OP also allows the user to control the spread of personal information in a privacy protecting way. With the use of a single point of authentication, namely the OP, it is easier and cheaper to build strong, e.g. multifactor, authentication between the user and the OP, which then translates to a strong authentication between the user and all RPs that he uses. We follow this direction with the TPM providing a secure storage for the credentials, paired with a local authentication where no credentials are released directly to the OP, and binding the authentication to the platform the user initially registered the identifier with. No password or username is sent to the OP in our approach, which in turn renders phishing attacks as described in section 2.1 on the OP side useless. The phishing attacker's target is to steal the user's credentials using a fake OP and then re-use the login credentials with the real OP later on. In TOID an attacker would not be able to retrieve re-usable credentials with a fake OP. The attacker's fake OP is assumed to use another website than the legitimate OP, and hence cannot impersonate the legitimate user at a RP even if the user is tricked into approving a challenge from the fake OP, since the user's OpenID identifier is hosted at the web address specified in the OpenID identifier, which is the real OP. Hence the attacker would have to perform an online attack in which he is able to trick the user into approving the challenge and at the same time control the URL of the real OP, to be able to establish associations between his fake OP and arbitrary RPs. Such an online attack is possible with the normal OpenID protocol as well. The attacker is required to control the OP to RP communication channel as well as the OP to user communication channel at the same time, resulting in a very limited and targeted attack scenario. TOID can increase the security even in this scenario if security demanding RPs are requesting signed platform integrity information from the OP which cannot be provided by the fake OP. Replaying fake challenge responses to the real OP is not possible since the OpenID protocol uses a nonce for every session. Faking challenge responses would require the attacker to gain access to the TPM protected AIK and CSK respectively for the creation of the response signature. By the inclusion of a replay protection inside the ticket, the problem of a phishing OP is mitigated. Additionally, a MITM attack, where the attacker is able to sit between the TTS and OP, can be prevented by a mutual authentication between OP and TTS, in which the TTS securely stores and verifies the OP certificates. To support multiuser environments we are able to establish multiple identities, represented by AIKs stored in the TPM, on a single platform and are able to protect them with a local password, secured by the TPM. Each user creates a new unique AIK and performs the AIK certification protocol. AIK certification is integrated in the AIK creation process, can be done before running an OpenID authentication session and is needed only once per AIK.

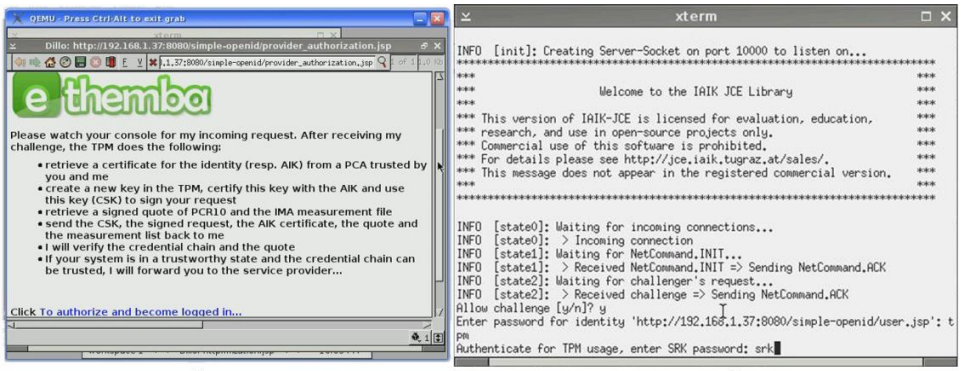

Figure 3. Screenshots showing: (a) the OP login page, (b) the TTS after receiving the challenge from the OP prompting the user for the local passwords.

Compared to simple TLS authentication with a TPM protected key, TOpenID combines attestation and authentication, whereas TLS alone only provides authentication. Using TLS extensions with integrity attestation is an option to integrate platform information. However, using TLS every RP will have to support the verification of the TLS credentials as well as the integrity verification of the reported measurement values which is highly impractical. Hence using an established and widely adopted lightweight IdM solution such as OpenID and bridging it with additional security and trust information bears an increased benefit, since a secure web-SSO can be provided to a large variety of RPs without any modifications to the existing OpenID authentication mechanisms implemented at the RPs. The presented scheme binds the use of an OpenID identifier to a specific TPM and hence a platform and the platform's integrity. By this binding an increased level of security can be achieved, as well as the use of OpenID for security demanding applications could be enabled. The goal of the current contribution is to enhance the security and bind identity authentication to a single platform. This effectively blocks attackers from transferring authentication credentials to a different machine, as it is done in typical phishing attacks. This contribution does not yet address the issue of identifier migration, e.g. using the same identifier with different devices, such as a laptop, smartphone and PC. Several mechanisms for migration are discussed in the context of virtualisation architectures [32] and could be applied to the presented solution.

3. Conclusions

OpenID is a lightweight protocol for federated identity management and is rapidly being adopted by the industry. We have shown that by incorporating Trusted Computing technology into the protocol, it is possible to create a more robust and secure OpenID implementation, which subsequently enables the protocol to be used for secure transactions such as financial payments etc. We have shown that by binding OpenID identities to the trustworthiness of the device, assured by the device's TPM, and then relaying the trust information to the OpenID provider, we can establish trust between the OpenID provider and the device. We have also implemented our concepts on a virtualization environment, faithfully implementing all functions required of the TPM, the device, and the OpenID provider in order to implement the TOID protocol. Many laptops and desktops now incorporate a TPM and trusted software stacks, facilitating the introduction of this improved OpenID protocol.

4. Acknowledgements

This work was funded by InterDigital, Inc. Special thanks go to Lawrence Case, Bob DiFazio David Greiner, Louis Guccione, Dolores Howry, and Michael V. Meyerstein, for many useful discussions and comments.

5. References

[1] A. Leicher, N. Kuntze, and A. U. Schmidt, "Implementation of a Trusted Ticket System," in Proc. IFIP SEC 2009, Pafos, Cyprus, May 18-20, 2009. Springer-Verlag, 2009, pp. 152–163.

[2] B. Fichtinger, E. Herrmann, N. Kuntze, and A. U. Schmidt, "Trusted Infrastructures for Identities," in Virtual Goods. Proc. 5th Intl. Workshop for Technical, Economic and Legal Aspects of Business Models for Virtual Goods, Koblenz, October 11-13, 2007. Nova Publishers, 2008.

[3] Trusted Computing Group, "TCG PC Client Specific Implementation Specification for Conventional BIOS," v. 1.20 rev. 1.00, July 2005.

[4] Trusted Computing Group, "TCG Mobile Reference Architecture," v. 1.0 rev. 5.

[5] Trusted Computing Group, "TPM Main," Specification Version 1.2 Level 2 rev. 103.

[6] E. Gallery, Trusted Computing. IEE, London, 2005, ch. An overview of trusted computing technology, pp. 29–114.

[7] N. Kuntze and A. U. Schmidt, "Trusted Ticket Systems and Applications," in Proc. IFIP SEC 2007. May 14-16, 2007, Sandton, South Africa. Springer-Verlag, 2007, pp. 49–60.

[8] Trusted Computing Group, "TCG Infrastructure Working Group Reference Architecture for Interoperability (Part I)," Specification Version 1.0 Revision 1

[9] E. Brickell, J. Camenisch, and L. Chen, *Trusted Computing*. IEE, London, 2005, ch. The DAA scheme in context, pp. 143–174.

[10] E. Brickell, J. Camenisch, and L. Chen, "Direct anonymous attestation," in Proceedings of the 11th ACM conference on Computer and communications security. ACM, 2004, pp. 132–145.

[11] J. Camenisch, "Better privacy for trusted computing platforms," Proc. ESORICS 2004, pp. 73–88.

[12] J. Camenisch and A. Lysyanskaya, "A signature scheme with efficient protocols," *Security and communication networks*, pp. 268–289, 2003.

[13] B. Smyth, M. Ryan, and L. Chen, "Direct Anonymous Attestation (DAA): Ensuring Privacy with Corrupt Administrators," *Security and Privacy in Ad-hoc and Sensor Networks*, vol. 4572/2007, pp. 218–231, 2007.

[14] C. Rudolph, "Covert Identity Information in Direct Anonymous Attestation,". in Proc. IFIP SEC 2007. May 14-16, 2007, Sandton, South Africa. Springer-Verlag, 2007, pp. 443–448.

[15] A. Leung, L. Chen, and C. J. Mitchell, "On a Possible Privacy Flaw in Direct Anonymous Attestation (DAA)," in Proc. TRUST 2008, Villach, Austria, March 11-12, 2008 Springer, 2008, pp. 179–190.

[16] A. Pashalidis and C. J. Mitchell, *Trusted Computing*. IEE, London, 2005, ch. Single Sign-On using TCG-conformant platforms, pp. 175–193.

[17] B. Schneier and J. Kelsey, "Secure audit logs to support computer forensics," *ACM Trans. Inf. Syst. Secur.*, vol. 2, no. 2, pp. 159–176, 1999.

[18] R. Sailer, X. Zhang, T. Jaeger, and L. Van Doorn, "Design and implementation of a TCG-based integrity measurement architecture," in Proc. 13th USENIX Security Symposium, 2004, pp. 223–238.

[19] A. Pashalidis and C. J. Mitchell, "Single sign-on using trusted platforms," in Proc. ISC 2003. Springer-Verlag, 2003, pp. 54–68.

[20] OpenID.net, "OpenID Specifications." Available: http://openid.net/developers/specs/. Access date: 18 August 2010.

[21] OpenID.net, "OpenID - Year 2009 in Review." Available: http://openid.net/2009/12/16/openid-2009-year-in-review//. Access date: 18 August 2010.

[22] OpenID Foundation (OIDF), "Open Trust Frameworks for Open Government" Available: http://openid.net/docs/Open_Trust_Frameworks_for_Govts.pdf. /. Access date: 18 August 2010.

[23] E. Tsyrklevich and V. Tsyrklevich, "Single Sign-On for the Internet: A Security Story," BlackHat Conference Las Vegas 2007, 2007.

[24] N. Kuntze, D. Mähler, and A. U. Schmidt, "Employing trusted computing for the forward pricing of pseudonyms in reputation systems," in Axmedis 2006 : Proc. 2nd Intl. Conference on automated production of cross media content for multi-channel distribution. Firenze University Press, 2006, pp. 145–149.

[25] OpenID4Java. Available: http://code.google.com/p/openid4java//. Access date: 18 August 2010.

[26] S. Gajek, A.-R. Sadeghi, and M. Winandy, "TruWallet: Trustworthy and Migratable Wallet-Based Web Authentication," in Prco. 4th ACM workshop on Scalable Trusted Computing. ACM, 2009, pp. 19–28.

[27] S. Gajek, A. Sadeghi, C. Stuble, and M. Winandy, "Compartmented security for browsers-or how to thwart a phisher with trusted computing," in Proc. Availability, Reliability and Security, 2007. IEEE, 2007, pp. 120–127.

[28] A. Leicher and A. Brett, "Ethemba, a Trusted Computing Demonstration and Experimentation Environment." Available: http://ethemba.novalyst.de. Acess date: 18 August 2010.

[29] M. Strasser, "A Software-based TPM Emulator for Linux," Available: http://www.infsec.ethz.ch/people/psevinc/TPMEmulatorReport.pdf. Access date: 18 August 2010.

[30] A. Sadeghi and C. Stüble, "Property-based attestation for computing platforms: caring about properties, not mechanisms," in Proc. Workshop on New security paradigms. New York, NY, USA, 2004. ACM, pp. 67–77.

[31] OpenID.net, "OpenID security best practices." Available: http://wiki.openid.net/OpenID-Security-Best-Practices. Access date: 18 August 2010.

[32] D. Plaquin, S. Cabuk, C. Dalton, D. Kuhlmann, P. Grete, C. Weinhold, A. Böttcher, D. Murray, T. Hong, and M. Winandy. (2009, 5) TPM Virtualisation Architecture document. IST-027635 / D04.7 FINAL 1.0_Update. Available: http://www.opentc.net/deliverables2008_2009/OpenTC_04.7_TPM_Virtualisation_Architecture_document_v2_M42.pdf. Access date: 18 August 2010.