Client-Side Biometric Verification Based on Trusted Computing

Jan Vossaert, Jorn Lapon, Bart Decker, Vincent Naessens

Client-Side Biometric Verification Based on Trusted Computing

Jan Vossaert[1], Jorn Lapon[1], Bart De Decker[2], and Vincent Naessens[1]

[1] Katholieke Hogeschool Sint-Lieven, Department of Industrial Engineering
Gebroeders Desmetstraat 1, 9000 Ghent, Belgium
firstname.lastname@kahosl.be
[2] KU Leuven, Department of Computer Science, iMinds-DistriNet
Celestijnenlaan 200A, 3001 Heverlee, Belgium
Bart.DeDecker@cs.kuleuven.be

Abstract. Traditionally, a user requires substantial trust in a workstation for correctly handling her credentials (e.g. password/login). Unfortunately, malware and compromised software makes them unsuitable for secure credential management. Credentials are easily stolen and the user cannot trust what is being displayed on her workstation, obstructing informed consent.
This paper presents a new solution that addresses these issues. Credentials are bound to the owner using biometrics, effectively impeding abuse through credential sharing and theft. The biometric verification is performed on the client side, preserving the privacy of the user. The solution ensures that the user is correctly informed about the pending authentication, preventing abuse by malware. To demonstrate the feasibility of our approach, a prototype was implemented.

1 Introduction

Many companies and governments are digitalizing their services. This allows users to access these services remotely. To prevent unauthorized users from gaining access and protect the integrity of these services, access control measures are enforced. These access control measures are typically enforced using digital credential technologies such as X.509 certificates, attribute based credentials or simply a username/password combination.

However, merely using digital credential technologies is not sufficient to fulfill the complex security and privacy requirements that apply in these settings. Credential technologies themselves, for instance, do not prevent users from *sharing* their credentials (e.g. digital credentials can be copied and distributed among users). Moreover, credentials can also be *abused by malicious software*. For instance, malicious software can use the credentials of the user to access personalized services, potentially without consent of the user. This impedes abuse detection and consequently credential revocation.

These issues can be tackled by binding the credentials to the owner. One way to bind a credential to its owner is by using biometric authentication to activate

the credential. Existing solutions typically use a tamperproof device such as a smart card on which the user's biometric template and credentials are stored. The user can activate these credentials by transferring her biometric scan to the tamperproof device. This binds the user to the credentials stored on the tamperproof device.

These biometric-based solutions, however, still have a number of disadvantages. They often require a significant amount of trust in the workstation. For instance, to inform the user about the pending authentication or to handle the user's biometric data. Moreover, besides a biometric scanner, these solutions require dedicated tamperproof hardware. Updating and patching the software running on these tamperproof devices is often difficult or even impossible. This hinders software security updates from being rolled-out and decreases flexibility with respect to using new biometric algorithms or credential technologies.

This paper presents a new solution for activating credentials bound to the owner by means of biometrics. In contrast to existing solutions based on tamperproof hardware, the verification is performed on the workstation in a trusted application. It, therefore, applies *Secure Virtualization Technologies* embedded in modern commodity workstations and laptops for building a Secure Execution Environment (SEE). In the prototype solution the user's credentials are stored on her mobile device and are bound to her biometrics. Both the biometric scan and its binding to the credential are verified in the SEE at the client side, trusted by both the user and the service provider. This strategy avoids leaking biometric information to, for instance, the service provider and requires no additional hardware infrastructure to be rolled out. Moreover, this solution is designed to be generic and, hence, supports the use of public, potentially untrusted workstations.

The *contribution* of this paper is threefold. First, it presents a solution for the secure verification of biometric traits on a workstation by applying Secure Virtualization Technologies. Second, access to remote services is controlled by credentials that can only be used after a successful biometric verification on the workstation. A mobile device is used to carry the user's credentials. Third, a prototype implementation of the system was realized, validating our solution. For the implementation an UHCI USB stack and a biometric scanner driver and algorithm were added to an existing framework for building SEE applications.

The rest of this paper is structured as follows. Section 2 points to related work. The used technologies are described in Section 3 and are followed by the design in Section 4. In Section 5, more information about the realization is presented. Subsequently, the system is evaluated in Section 6. Finally, we draw some conclusions in Section 7.

2 Related work

Some credential systems such as the Identity Mixer library [8] provide all-or-nothing non-transferability to discourage users from sharing their credentials. All the credentials of the user are tied together. Hence, assuming that the user owns

at least one valuable credential, he will not be willing to share her credentials with other users. A similar approach is PKI-assured non-transferability where the credentials are bound to a valuable secret outside the system (e.g. credit card information). Whereas these systems focus on the discouragement of credential sharing, the system proposed in this paper also addresses abuse (after theft) prevention and informed consent.

Another approach to prevent digital credentials from easily being copied or shared is by embedding them inside tamperproof hardware. The system proposed in [7] leverages the DAA [6] protocol, available in Trusted Platform Modules (TPMs) of modern workstations, to bind the user's anonymous credentials to a TPM. Similarly, smart cards are used to implement anonymous credential systems [2, 21, 1]. Although smart cards prevent the credentials from being copied, they do not fully prevent sharing of the credentials. Moreover, anonymous credential systems have other unsolved issues, such as their performance and correctly informing which information is being disclosed.

To tackle this, biometric authentication can be used to bind credentials to the owner. For instance, in [17, 4] the *wallet with observer* architecture [9] is extended to include biometric authentication towards the observer. The user is issued a tamperproof card containing her credential and biometric template. To use the credential in the card, the holder is required to scan her biometric data. Only if the scanned data matches the template stored in the card, the credential is activated. As an example, a privacy preserving identity card has been designed [11] taking advantage of this approach. Another approach [3] uses *fuzzy extractors* [14, 16] to generate a cryptographic key based on the retrieved biometric features. This key is never stored and the tamperproof device is trusted to erase the value after authentication. Hence, fresh biometric readings are required to reconstruct the cryptographic key. These systems focus on the prevention of abuse through theft and sharing, but do not fully realize the aspect of informed consent. This is especially important in case anonymous credentials are used. Tamperproof devices are also less flexible with respect to software updates and the support of complex biometric systems or credential technologies.

3 Background

TCG Trusted Computing. Nowadays, modern commodity computers are equipped with a Trusted Platform Module (TPM) [15]. This is a hardware module physically attached to the computer's motherboard, extending the system with a set of security related features. One these features is the measurement of the state of the system. To this end, the TPM contains several Platform Configuration Registers (PCRs). These are cleared upon power up and can only be modified using the extend operation, performed inside the TPM. The result of this operation for a specific PCR is a new PCR value, being the hash of the current value concatenated with a new value (i.e. $PCR_n := SHA1(PCR_n \mid\mid value)$).

A transitive trust model is employed: each software component, starting from the Core Root of Trust for Measurement in the BIOS, is responsible for measur-

ing the following component in the chain before passing control. Hence, before loading subsequent software components, the preceding component hashes the binaries of the components to be loaded and extends the result in a specific PCR. As a result, the PCRs represent the state of the system, or in other words, the loaded software configuration.

Based on this state, the TPM also supports a number of additional features. Data can be sealed with the seal operation and only if the system resides in the state specified during the seal operation can the data be unsealed (unseal). Additionally, the quote command returns a proof of the state (i.e. a *quote*) which a third party can verify (verifyQuote) asserting that the (remote) system runs in a specific (trusted) state. This functionality uses the private key and certificate of the TPM (i.e. sk_{tpm} and $cert_{tpm}$) to assert that the operation is performed by a genuine TPM. These credentials can either be generated by the hardware manufacturer or during an enroll phase.

Secure Execution Environment. While TPMs are being embedded in workstations for several years now, a more recent evolution is the adoption of SEE technologies such as Intel's Trusted Execution Technology (TXT) [10] and AMD's Secure Virtual Machine (SVM) [13]. These technologies allow the execution of measured code independently of previously executed software. The TPM specification has been extended with additional capabilities to support these new technologies.

Recent work [20] presents a framework that uses these technologies to allow developers to isolate security critical code from applications and run it in a secure environment. The main, possibly untrusted, OS is temporarily suspended after which the sensitive Piece of Application Logic (*PAL*) is securely executed. When the execution of the sensitive code is completed, the OS resumes execution. Typically, the *PAL* extends its state in the TPM with a fixed known value before releasing control back to the OS. This prevents the OS from gaining access to secrets from the *PAL* or from asserting data on behalf of the *PAL*. The framework supports data transfer between the main OS and the *PAL*. The TPM operations can be used to assert to a remote party that certain data was generated by a trusted *PAL*. The framework supports both Intel TXT and AMD's SVM technology on Windows and Linux based systems. In [5] this framework is extended to allow secure user interaction (i.e. input via the keyboard and output via the monitor).

Biometric Authentication. Biometry can be used to uniquely identify a person [19, 18]. Commonly used biometric traits include a fingerprint, iris, face and voice. A special purpose sensor device is used to read the biometric trait of the user. During enrollment, a distinguishing feature set is extracted from the biometric data and stored as a *biometric template* (bt_u) of the user. During authentication, the user scans her biometric trait (bioScan) and the resulting feature set is *matched* to the feature set contained in the template of the user. Based on the similarity of the two sets, the authentication is either accepted or rejected (bioVerify).

These biometric templates can be bound to the credentials of the user. The credential based authentication is then combined with the biometric authentication. A verifier can, hereby, check that the user of the credentials is also the owner. Binding the user's biometrics to an X.509 certificate can, for instance, be done by including a cryptographic hash of the biometric template in the certificate. For other types of credentials such as passwords or anonymous credentials similar principles can be applied. The verification of the biometric binding is further denoted as verifyBinding.

4 Design

This section first lists the different actors in the system. Followed by the requirements and a general description of the system. Finally, a detailed description of the protocols is presented.

4.1 Roles

We assume a user U, carrying a mobile device M. The mobile device stores the user's credentials and is used as a credential vault for accessing remote services from the workstation. The workstation runs a legacy operating system (WS) and supports SEE technologies for running a trusted application (PAL). A biometric scanner is attached to the workstation.

4.2 Requirements and adversary model

Functional requirements

- F_1 The system requires commodity hardware only.
- F_2 The solution does not require authentication to be bound to a particular workstation.
- F_3 The system is extensible and modular, allowing for new biometric systems or algorithms and credential technologies to be included.
- F_4 In case vulnerabilities are found in the system, software updates are easily applied.

Security and privacy requirements

- S_1 A credential for authenticating to a remote service can only be used by its owner.
- S_2 Malicious software cannot mislead the user into approving malicious signing transactions or authentication attempts.
- P_1 The system protects the biometric information of the user.

Adversary model

A_1 Regarding the secure execution environment, the same assumptions as the Trusted Computing Group [15] are made.
A_2 The trusted application (PAL) is assumed to be formally verified.
A_3 The biometric system is assumed to be secure (i.e. sufficiently low false acceptance rate).
A_4 The mobile device is assumed to securely handle the user's credentials and biometric information. To this end, an embedded secure element can be used for managing this data.
A_5 The service providers correctly verify the received attestations.

4.3 General approach

A user authenticates on the workstation towards a remote service provider. The service provider requires that the user proves ownership of the used credentials, before allowing access to its services. To his end, the user's credentials are bound to her biometrics. The verification of the biometric binding between the credentials and the user is performed by a dedicated trusted application (i.e. the PAL). In addition, the PAL informs the user about the details of the pending authentication. This ensures that malware cannot mislead the user into approving malicious transactions. This is especially important in case anonymous credentials are used, as the user should give consent on the selective disclosure of attributes. The *quote* functionality of the TPM is used to assert to service provider that a trusted PAL properly executed the verification. The credentials and biometric data are stored on the mobile device of the user and are only released towards the trusted PAL, running on the workstation. This ensures that the user does not release her biometric data to malicious applications.

The PAL is trusted by both the user and the service provider. The user trusts the PAL not to release her biometric information to third parties and for informed consent regarding the pending authentication. The service provider trusts the PAL to correctly verify the biometric binding between the user and the used credential(s). This trust is supported by the attestation of the PAL towards the service provider and mobile. The state of the trusted PAL can be certified by a trusted third party. The PAL should also be open source so that independent developers can verify that the certified state correctly represents the desired functionality.

4.4 Protocols

Prerequisites The SEE technologies on the workstation are enabled in the BIOS and the TPM has been certified (i.e. sk_{tpm} and $cert_{tpm}$ are generated). The PAL running on the workstation of the user has been initialized. During initialization the PAL generates a keypair (pk_{pal} and sk_{pal}) and *seals* it to its state resulting in the sealed object denoted as $keyStore$. The mobile device and the service provider obtain the certified PCR state of the trusted PAL. Hence,

these parties can verify that the application running on the workstation is indeed the intended trusted application.

A credential issuer issues credentials bound to the user's biometrics, which are stored on the user's mobile device. The mobile requires the user to choose a unique authentication image (img_u) that will allow the user to visually verify that the software running on the workstation is indeed trusted.

Pairing Before authentication, the mobile device verifies that a valid trusted application is running on the workstation. As part of this protocol, the mobile retrieves the *PAL*'s public key (pk_{pal}), which is used to encrypt data addressed to the *PAL*. The mobile device stores this public key for future authentications. We denote this protocol as the *pairing protocol*. Note that this pairing can be performed immediately before the authentication in case a public workstations is used.

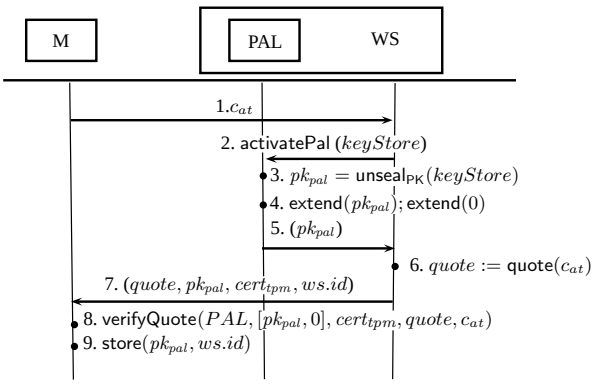

Fig. 1. The pairing protocol.

Figure 1 illustrates the steps of the pairing protocol. First, the mobile sends a random attestation challenge to the workstation (1). The workstation activates the trusted *PAL* with the sealed *keystore* (2) and the *PAL* retrieves his public key from the sealed *keystore* (3). To allow attestation that this public key is indeed managed by the trusted application, the state is extended with this key (4). The *PAL* returns its public key to the workstation which resumes its execution (5). A quote operation on the state resulting from the *PAL* execution is performed using the attestation challenge (6). As the public key was extended in the state, this ensures the authenticity of the public key sent to the mobile. The challenge ensures freshness of the quote. The quote, public key, certificate and an identifier of the workstation is sent back to the mobile, which then verifies the quote (7-8). If the quote verification was successful, the public key is stored together with the workstation's identifier (9).

Authentication Figure 2 presents the protocol for authenticating the user towards a remote service provider. First, the user requests access to a protected resource from the remote service provider (1). The provider responds with an authentication request containing its certificate, an authentication and an attestation challenge (2). The mobile now receives the authentication challenge, certificate of the service provider and a unique identifier of the workstation (3). The mobile informs the user about the workstation on which the authentication will be performed and towards which service provider (4). If the user acknowledges, the mobile signs the authentication challenge with the user's credential and encrypts the signature, authentication certificate, biometric template and the user's unique image using the public key of the *PAL* (5-7). The encryption is sent to the workstation where it is then passed as a parameter to the *PAL*, together with the *PAL*'s sealed keys in *keyStore* (8-9). The *PAL* now unseals its private key to decrypt the encrypted data *enc* (10-11). Subsequently, the binding between the biometric template and the authentication credential is verified (12). If the verification succeeds, the user is informed about the pending authentication and requested to scan his biometric data. The user's unique image is shown to indicate to the user that the trusted environment is running (13). The user can, hence, trust all information shown on the monitor. To acknowledge the authentication, the user scans his biometric data using the biometric scanner attached to the workstation (14). As the *PAL* is in complete control of the workstation, it can directly interact with the hardware. This ensures that the data shown on the monitor and read from the biometric scanner cannot be tampered with. The *PAL* can now verify if the biometric template matches with the scanned biometric data (15). Upon successful verification, the *PAL* is assured that the user is the owner of the authentication credentials and *extends* its state with the signature, authentication certificate and the certificate of the service provider (16). The *PAL* ends its execution and returns the user's signature and certificate back to the regular OS, that resumes its operation (17). The OS now performs a *quote* operation on the state resulting from the *PAL* execution (18). This quote attests towards the service provider that the trusted *PAL* indeed verified the biometric binding (i.e. the *PAL* state is extended with the user's certificate and signature) and that the user was shown the correct information about the service provider (i.e. the *PAL* state is extended with the service provider's certificate). The resulting quote is sent to the service provider along with sig and $cert_u$ and the certificate of the TPM (19). The service provider now verifies the quote, the user's certificate and signature (20-21). Upon success, the service provider grants access to the requested resource (22).

5 Realization

For the realization of a prototype, an off-the-shelf USB fingerprint scanner (i.e. Eikon UPEK fingerprint reader) was used as biometric reader. The user's credential is a X.509 certificate with a 1024 bits RSA key. The user's fingerprint template is bound to her authentication credential by including a cryptographic

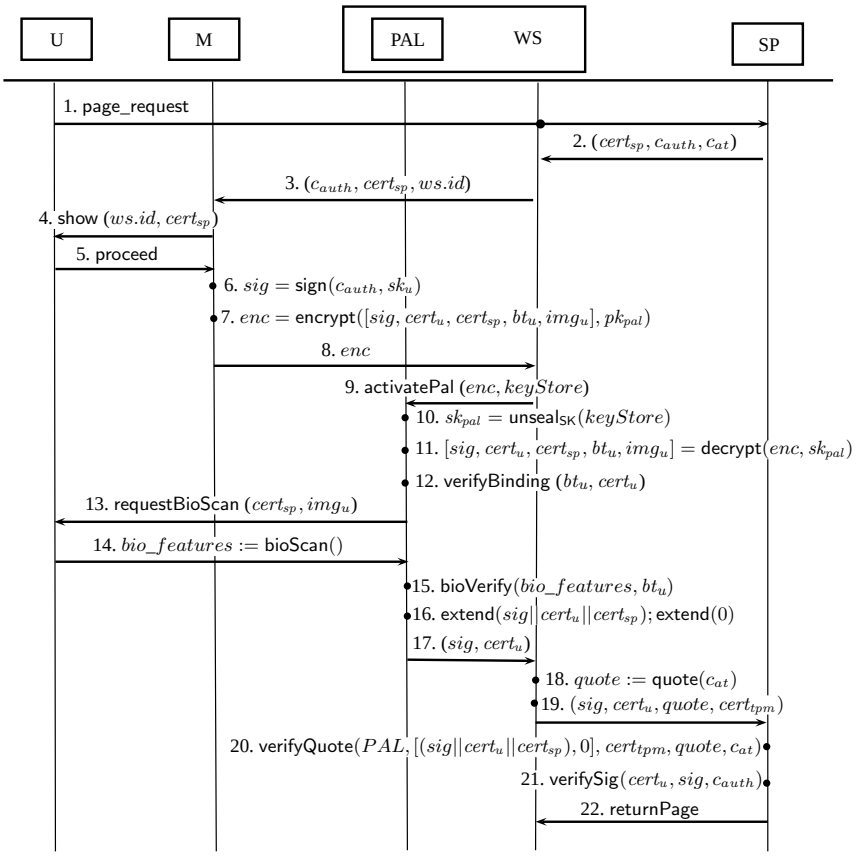

Fig. 2. The authentication protocol.

hash of the fingerprint template in the X.509 certificate. Table 1 shows the hardware used for the realization of the different entities.

Entity	**Realization**
M	Samsung i9000 Galaxy S: 1GHz ARM Cortex-A8, 512MB RAM The smartphone runs Android 2.3.3
SP, *PAL*, *WS*	DELL E4200: Intel Core2 Duo U9400 @ 1.4GHz, 4GB RAM The laptop runs Ubuntu 12.04 32-bit, 3.2.0 kernel

Table 1. The hardware platforms for the realization of the different entities.

For the functionality of the mobile device, an Android app has been developed. The service provider is implemented on an Apache Tomcat server. Spring Security is used to enforce its access control policy. A Spring authentication

module was added to handle the custom authentication protocol. Nevertheless, the focus of the prototype lies in the development of the software components on the workstation. The user accesses the service provider via the browser on the workstation. When authentication is required, the authentication request (*bioauth:authRequestData*) is forwarded to a local application on the workstation. For demonstration purposes a bidirectional network connection is setup between the local application and the mobile device. To this end, a QR code, containing the workstation's IP address, is displayed by the local application and scanned with the mobile device. Using this IP address, the mobile connects to the workstation and uses this channel for further communication. The local application also informs the user about the progress of the authentication protocol. Note that the actual authentication is delegated to a background daemon that accepts incoming TCP/IP connections from the mobile of the user and has the required privileges to suspend the OS and start the *PAL*.

For the biometric verification, a basic USB UHCI stack and fingerprint driver for the Eikon reader were implemented. For parsing X.509 certificates and other cryptographic operation, parts of the PolarSSL library were used. When the *PAL* has finished execution, the daemon uses the TrouSerS TCG software stack to implement the *quote* operation which is used to attest the *PAL* towards the service provider.

Our *PAL* application consists of three main components. The Eikon fingerprint driver (447 lines of code) running on top of the USB UHCI stack (1001 lines of code). The implementation of the *PAL* protocol itself consists of 1040 lines of code. This adds a total of 2488 lines of code to the flicker/soft cards framework, preserving a minimal Trusted Computing Base (TCB).

Table 2 illustrates the performance of the prototype. The computations on the mobile and server have negligible impact on the overall system performance. The *PAL* and the *WS* introduce the largest overhead as they use the constrained resources of the TPM. The workstation uses the TPM for the quote operation, the *PAL* for the unseal functionality and the establishment of the SEE. The actual scanning of the fingerprint is not included in the measurement to avoid impact of the user interaction. The measurements of the *PAL*, however, do include the initialization of the USB stack and fingerprint driver. Once, the reader is initialized, the actual swiping is only a fraction of the total time. Although the authentication phase of the *PAL* is the most time consuming operation, the user experience isn't degraded as some operations can be performed while the user is reviewing the authentication details on the monitor.

Performance	**PAL**	**WS**	**M**	**SP**	**Total**
Pairing	1900	720	30		2650
Authentication	4550	740	40	8	5338

Table 2. The performance of the pairing and the authentication (ms).

6 Evaluation

6.1 Requirements review.

This section discusses how the requirements presented in section 4.2 are realized in the design.

Functional requirements. As the prototype illustrates, the system only requires commodity hardware. The mobile component was implemented on a smartphone and the secure virtualization technologies required on the workstation are being embedded in off-the-shelf workstations, satisfying requirement F_1.

The system works with any workstation supporting the required SEE prerequisites. Hence, a user can use any such workstation to securely authenticate towards a remote service provider, satisfying requirement F_2.

The *PAL* can easily be updated by distributing a new binary to the workstation and certifying the new state of the application with the mobile device of the users and the service providers. This can be managed by a trusted third party that certifies and revokes these states. The software on the mobile device and server can be updated using traditional mechanisms. This allows the integration of additional biometric and/or credential technologies and security updates to be installed, satisfying requirements F_3 and F_4.

Security and privacy requirements. The authentication credentials of the user are bound to her biometrics. The *PAL* ensures the correct verification of the user's biometrics before asserting the authentication. The service provider can verify the assertion and, hence, check that the actual owner authorized the authentication, satisfying requirement S_1.

The mobile device verifies integrity of the *PAL* running on the workstation, before sending any personal data. Moreover, the user is assured that the trusted application is running, as her personal image img_u is shown on the workstation. The user is, hence, assured that the provided information about the pending authentication is correct. Furthermore, the *PAL* binds the verification process to the service provider presented to the user. This satisfies security requirement S_2.

In the prototype, the biometric template of the user is bound to the user's certificate by including a cryptographic hash of the template in the certificate. This prevents biometric information of the user from being leaked when the certificate is released. The mobile device only discloses the biometric template to a trusted *PAL* (i.e. the *PAL* does not reveal the fingerprint to a third party) over a secure channel. Moreover, the user can verify that a trusted *PAL* is running when scanning her fingerprint. This prevents freshly scanned biometric information of the user from being leaked, satisfying requirement P_1.

6.2 Security and privacy considerations.

The main focus of the system is protecting the user from software attacks, as these are the most common and scalable types of attacks. Moreover, TPMs are not designed to be secure against hardware attacks. To mitigate the impact of hardware attacks on a TPM, its credentials can be revoked.

The *PAL* is trusted by both the user and the service provider to correctly execute the specified protocol. To limit implementation flaws that could be exploited, the functionality of the *PAL* is kept to the minimum (i.e. informed consent, USB communication and biometric verification). The small TCB decreases the chance of bugs and suggests that formal verification is possible. Moreover, as mentioned in the functional requirements review section, the *PAL* can be easily updated after which the previous version can be revoked by blacklisting its state.

Currently, the system assumes that all service providers require biometric verification. If a service provider does not require this proof of ownership, the protocol requires some minor modifications. Otherwise, malware could trick the user in signing an authentication challenge obtained from another service provider, which does not require this biometric verification. One simple solution could be to have the mobile verify the service provider's certificate and encrypt the user's signature with the contained public key.

Ideally, the biometric reader used for obtaining a biometric scan of the user is bound to the workstation on which the user is working. However, most biometric scanners are plug-and-play and can easily be removed and replaced with other hardware devices. If a fingerprint scan is eavesdropped, it can be replayed as a fresh reading. This risk can be mitigated by implementing a cryptographic protocol between the trusted application and the reader to ensure freshness of the scan. Although hard to achieve, to prevent relay attacks, the *PAL* should also be able to verify that the used biometric reader is actually attached to the workstation on which the *PAL* is running.

The system presented in this paper increases the user's privacy with respect to the biometric authentication. Nevertheless, this system is easily extended to further increase privacy by supporting anonymous credentials in which no linkable information should be released to the service provider. In the prototype, the *quote* generation requires a certificate bound to the TPM. This makes all transactions performed on a single workstation linkable. Therefore, modern TPMs also support the *DAA* protocol. This protocol allows anonymous attestation of the platform. As such, the system only leaks the state of the *PAL* and the data disclosed during the user authentication.

Note that it is even possible to support password based authentication with our system. In that case the certification authority should bind the biometric data to the login or username.

6.3 Applicability

This section discusses two possible application domains in which the system presented in this paper can be used to realize increased security compared to currently deployed systems, namely *eID systems* and *online banking*.

eID Systems typically allow the user to access a wide range of personal services. This can go from very privacy sensitive services such as online tax submission to less privacy sensitive services such as pay-per-view news site. While these privacy sensitive services will typically only be accessed from a trusted home computer, other services might be accessed on workstations not fully trusted by the user. However, when using the same authentication credentials for the privacy sensitive and the other services, malware on an untrusted workstation could use the authentication credentials to access other services then requested by the user. The system presented in this paper prevents this type of abuse by correctly informing the user about the service which will be accessed.

Online banking enables a wide variety of banking services (e.g. viewing the status of your bank accounts, loans and execute bank transactions) via a workstation connected to the Internet. The user owns a credential with which he can log in and authorize transactions (e.g. wire transfer).

In current eBanking systems this secret is stored in an smart card (i.e. the bank card of the user). To authorize transactions, the details of the transaction are transferred to the bank card that subsequently generates an authorization response. This approach protects the credentials of the user but is vulnerable to phishing attacks. Our approach also tackles the latter as the secure execution environment application correctly informs the user about the pending transaction. It, moreover, replaces PIN authentication of the user with stronger biometric authentication.

6.4 Comparison with existing systems

As discussed in related work, several systems for misuse protection of credentials exist. For this comparison, the existing systems are categorized as follows. *Hardware based protection (HBP)* systems [7, 2, 21] rely on tamperproof hardware to prevent credentials from being digitally copied and, hence, easily abused. *Software based protection (SBP)* systems [8] rely on binding the credentials of the user to a valuable secret of the user, discouraging users to share their credentials. Finally, *biometry-based protection (BBP)* systems [17, 4, 9, 12, 11, 3] embed the user's credentials in a tamperproof module that requires biometric authentication of the user before the credentials can be used. The results of the comparison are summarized in Table 3.

Informed consent allows user to (dis)approve transactions based on correct information about the transaction. The tamperproof hardware components used in HBP and BBP systems typically do not allow direct communication with the user. Therefore, other, potentially compromised, devices are required to inform the user about the transaction. In SBP systems, the credential operations are typically executed on a regular workstation. In the system presented in this paper, the trusted application has full control over the hardware of the workstation and can, therefore, use the monitor to reliably inform the user about the transaction.

The HBP and BBP both rely on tamperproof hardware components to store the user's credentials and execute the credential and biometric operations. These devices are typically resource constrained limiting the usage of computationally intensive credential technologies such as anonymous credentials and the number of authentication credentials that can be stored inside those devices. The SBP and the system proposed in this paper are implemented on a general purpose workstation.

The HBP system implements misuse protection by embedding the credentials in a tamperproof module preventing them from being digitally copied. The SBP system discourages the user from sharing her credentials. The BBP and the approach discussed in this paper both implement misuse protection by binding the credentials to a specific user using biometrics. The solution presented in this paper, however, could easily be extended to support these credentials without checking the biometric binding.

The HBP and BBP both rely on tamperproof modules for executing credential and biometry related operations. The software installed on these modules is typically difficult to update providing less flexibility compared to the SBP and the system presented in this paper. These systems run on general purpose hardware in which updates can be part of the update infrastructure of the operating system.

	HBP	**SBP**	**BBP**	**Our approach**
Informed consent	No	No	No	Yes
Hardware resources	Constrained	Powerful	Constrained	Powerful
Protection	Copy prevention	Discourage sharing	Bio. binding	Bio. binding
Flexibility	Low	High	Low	High

Table 3. Comparison between the system proposed in this paper and existing misuse protection systems.

7 Conclusion

This paper presents a new solution for activating credentials bound to its owner by means of biometrics. It assures that users are physically present when their credentials are used, effectively impeding credential sharing and abuse by theft. Moreover, credential abuse by malware is prevented by isolating the credential operations in a secure environment on a workstation. Apart from the hardware support available in modern commodity workstations, no additional infrastructure is required.

The system can be applied to general client-server authentication use-cases or dedicated use-cases such as electronic identity systems or eBanking. A prototype implementation demonstrates the feasibility of our system. In future research,

this approach can be extended to include the verification of contextual information such as geographical data. A service provider could require that the workstation to be located in a certain country. This could be used to impede relay attacks.

Acknowledgements

This work is made possible through funding from the MobCom project, by the Flemish agency for Innovation by Science and Technology (IWT).

References

1. Patrik Bichsel, Jan Camenisch, Bart De Decker, Jorn Lapon, Vincent Naessens, and Dieter Sommer. Data-minimizing authentication goes mobile. In Bart De Decker and David W. Chadwick, editors, 13th Joint IFIP TC6 and TC11 Conference on Communications and Multimedia Security - CMS 2012 September 3th - September 5th, 2012, Canterbury, UK,, pages 55–71. Springer, September 2012.
2. Patrik Bichsel, Jan Camenisch, Thomas Groß, and Victor Shoup. Anonymous credentials on a standard java card. In Proceedings of the 16th ACM conference on Computer and communications security, CCS '09, pages 600–610, New York, NY, USA, 2009. ACM.
3. Marina Blanton and William M. P. Hudelson. Biometric-based non-transferable anonymous credentials. In Proceedings of the 11th international conference on Information and Communications Security, ICICS'09, pages 165–180, Berlin, Heidelberg, 2009. Springer-Verlag.
4. Gerrit Bleumer. Biometric yet privacy protecting person authentication. In In Proceedings of 1998 Information Hiding Workshop (IHW 98, pages 101–112. Springer-Verlag, 1998.
5. Franz Ferdinand Brasser, Sven Bugiel, Atanas Filyanov, Ahmad-Reza Sadeghi, and Steffen Schulz. Softer smartcards - usable cryptographic tokens with secure execution. In Angelos D. Keromytis, editor, Financial Cryptography, volume 7397 of Lecture Notes in Computer Science, pages 329–343. Springer, 2012.
6. Ernie Brickell, Jan Camenisch, and Liqun Chen. Direct anonymous attestation. In Proceedings of the 11th ACM conference on Computer and communications security, CCS '04, pages 132–145, New York, NY, USA, 2004. ACM.
7. Jan Camenisch. Protecting (anonymous) credentials with the trusted computing groups tpm v1.2. In Simone Fischer-Hbner, Kai Rannenberg, Louise Yngstrm, and Stefan Lindskog, editors, Security and Privacy in Dynamic Environments, volume 201 of IFIP International Federation for Information Processing, pages 135–147. Springer US, 2006.
8. Jan Camenisch and Els Van Herreweghen. Design and implementation of the idemix anonymous credential system. In Proceedings of the 9th ACM conference on Computer and communications security, CCS '02, pages 21–30, New York, NY, USA, 2002. ACM.
9. David Chaum and Torben P. Pedersen. Wallet databases with observers. In Proceedings of the 12th Annual International Cryptology Conference on Advances in Cryptology, CRYPTO '92, pages 89–105, London, UK, UK, 1993. Springer-Verlag.

10. Intel Corporation. LaGrande technology preliminary architecture specification. Intel Publication no. D52212, May 2006.
11. Yves Deswarte and Sébastien Gambs. A proposal for a privacy-preserving national identity card. Trans. Data Privacy, 3(3):253–276, December 2010.
12. Yves Deswarte and Sébastien Gambs. Cryptography and security. chapter The challenges raised by the privacy-preserving identity card, pages 383–404. Springer-Verlag, Berlin, Heidelberg, 2012.
13. Advanced Micro Devices. AMD64 architecture programmer's manual: Volume 2: System programming. AMD Publication no. 24594 rev. 3.11, Dec 2005.
14. Yevgeniy Dodis, Leonid Reyzin, and Adam Smith. Fuzzy extractors: How to generate strong keys from biometrics and other noisy data. In Christian Cachin and Jan Camenisch, editors, Advances in Cryptology - EUROCRYPT 2004, volume 3027 of Lecture Notes in Computer Science, pages 523–540. Springer Berlin / Heidelberg, 2004.
15. Trusted Computing Group. TCG TPM specification. `http://www.trustedcomputinggroup.org/resources/tpm_main_specification`.
16. Feng Hao, Ross Anderson, and John Daugman. Combining crypto with biometrics effectively. IEEE Trans. Comput., 55(9):1081–1088, September 2006.
17. Russell Impagliazzo and Sara Miner More. Anonymous credentials with biometrically-enforced non-transferability. In Proceedings of the 2003 ACM workshop on Privacy in the electronic society, WPES '03, pages 60–71, New York, NY, USA, 2003. ACM.
18. Anil K. Jain, Patrick Flynn, and Arun A. Ross. Handbook of Biometrics. Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2007.
19. Anil K. Jain, Karthik Nandakumar, and Abhishek Nagar. Biometric template security. EURASIP J. Adv. Signal Process, 2008:113:1–113:17, January 2008.
20. Jonathan M. McCune, Bryan J. Parno, Adrian Perrig, Michael K. Reiter, and Hiroshi Isozaki. Flicker: an execution infrastructure for tcb minimization. SIGOPS Oper. Syst. Rev., 42(4):315–328, April 2008.
21. Wojciech Mostowski and Pim Vullers. Efficient U-Prove implementation for anonymous credentials on smart cards. In Muttukrishnon Rajarajan, Fred Piper, Haining Wang, and George Kesidis, editors, 7th International ICST Conference on Security and Privacy in Communication Networks, SecureComm 2011, London, UK, September 7-9, 2011. Proceedings, volume 96 of Lecture Notes of the Institute for Computer Sciences, Social-Informatics and Tele-communications Engineering (LNICST), pages 243–260. Springer-Verlag, August 2012.