Remote Credential Management with Mutual Attestation for Trusted Execution Environments

Carlton Shepherd(✉), Raja Naeem Akram, and Konstantinos Markantonakis

Information Security Group, Royal Holloway,
University of London, Egham, UK
{carlton.shepherd.2014,r.n.akram,k.markantonakis}@rhul.ac.uk

Abstract. Trusted Execution Environments (TEEs) are rapidly emerging as a root-of-trust for protecting sensitive applications and data using hardware-backed isolated worlds of execution. TEEs provide robust assurances regarding critical algorithm execution, tamper-resistant credential storage, and platform integrity using remote attestation. However, the challenge of remotely managing credentials *between* TEEs remains largely unaddressed in existing literature. In this work, we present novel protocols using mutual attestation for supporting four aspects of secure remote credential management with TEEs: *backups*, *updates*, *migration*, and *revocation*. The proposed protocols are agnostic to the underlying TEE implementation and subjected to formal verification using Scyther, which found no attacks.

Keywords: Credential management · TEEs · Security protocols

1 Introduction

Trusted computing offers robust, on-device protection of security-critical data and the ability to securely report evidence of platform integrity, which has culminated in efforts such as the Trusted Platform Module (TPM). Until recently, however, such technologies were relatively restricted: neither arbitrary application execution nor secure input/output (I/O) are realisable with TPMs without substantially increasing the hardware-software Trusted Computing Base (TCB), say, through the use of virtual machines [11]. Trusted Execution Environments (TEEs), discussed further in Sect. 2, have emerged as the forerunner for addressing these shortcomings, particularly for constrained devices [19]. Unlike TPMs, TEEs provide hardware-enforced isolated execution of critical applications and data on the same underlying hardware. TEEs aim to thwart sophisticated software adversaries from a conventional Operating System (OS) irrespective of its protection mode, e.g. Rings 0–3. Modern Intel and ARM chipsets offer Intel

Published by Springer Nature Switzerland AG 2019
O. Blazy and C. Y. Yeun (Eds.): WISTP 2018, LNCS 11469, pp. 157–173, 2019.
https://doi.org/10.1007/978-3-030-20074-9_12

Software Guard eXtensions (SGX) and ARM TrustZone respectively for instantiating a TEE from the CPU or System-on-Chip (SoC).

Despite widespread availability, managing TEE data credentials throughout their life-cycle has received little attention by the community. Such credentials, whether derived from a public-key certificate, password or another value, are typically used to authenticate sensitive actions and transmitted data. Challenges arise, however, when credentials require migrating, revoking, updating or backing-up in a secure and trusted manner with bi-directional assurances between both end-points. Firstly, large numbers of TEEs must be administered, thus limiting the feasibility of human intervention, potentially over a multitude of communication mediums. Secondly, heterogeneous TEEs must be accommodated: Intel SGX, for example, is confined to Intel CPUs on more powerful devices, while ARM TrustZone is limited to ARM-based SoCs. Hence, for the first time, we address four key challenges when managing heterogeneous TEE credentials over its lifetime with bi-directional trust assurances for remote *migration* (Sect. 3), *revocation* (Sect. 4), *backups* (Sect. 5), and *updates* (Sect. 6). This paper presents the following contributions:

- An examination of existing smart card and TPM work relating to each credential management challenge and their applicability to TEEs.
- A suite of proposed protocols for facilitating TEE credential management with mutual attestation. The protocols are agnostic of the TEE and communication medium, and employ a Trusted Service Manager (TSM) in line with the GlobalPlatform TEE specifications [10].
- The proposed protocols are subjected to formal symbolic verification using Scyther, which found no attacks. We publicly release the verification specifications for further research[1].

2 Trusted Execution Environments (TEEs)

GlobalPlatform defines a TEE as an isolated execution environment that *"protects from general software attacks, defines rigid safeguards as to the data and functions a program can access, and resists a set of defined threats"* [9]. TEEs aim to isolate applications from integrity and confidentiality attacks from the untrusted OS, e.g. Android, or Rich Execution Environment (REE), by allocating distinct memory regions with accesses controlled by hardware. We summarise the foremost commercial TEEs for Intel and ARM chipsets; the reader is referred to [22] for a detailed survey of secure and trusted execution environments.

Intel Software Guard eXtensions (SGX) is an extension to the x86-64 instruction set that enables the creation of per-application 'enclaves'. Enclaves reside in isolated memory regions within RAM with accesses mediated by the CPU, which is considered trusted [6]. Secure storage is provided via the sealing abstraction, where data is encrypted to the untrusted world using a key derived from a processor-specific Storage Root Key (SRK). Enclave- or author-specific

[1] Available online at: https://cs.gl/extra/wistp18-scripts.zip.

keys can be derived; that is, respectively, binding data to only that enclave, or from an ID string to preserve persistence between enclaves from the same author. Remote attestation enables remote verification of enclaves and secret provisioning using the Enhanced Privacy ID (EPID) scheme [4], which authenticates enclave integrity measurements without revealing the CPU's identity.

GlobalPlatform (GP) TEE with ARM TrustZone maintains two worlds for all trusted and untrusted applications (see Fig. 1). A TEE kernel is used for scheduling, memory management, cryptographic methods and other OS functions, while user-mode TEE Trusted Applications (TAs) access OS functions exposed by the GP TEE Internal API. The GP TEE Client API [9] defines the interfaces for communicating with TAs from the REE. The predominant method for instantiating the GP TEE is with ARM TrustZone, which enables two isolated worlds to co-exist in hardware using two virtual cores for each world per physical CPU core and an extra CPU bit (NS bit) for distinguishing REE/TEE execution modes. TrustZone provides secure I/O with peripherals connected over standard interfaces, e.g. SPI and GPIO, by routing interrupts to the TEE kernel using the TrustZone Protection Controller (TZPC) for securing on-chip peripherals from the REE, and the Address Space Controller (TZASC) for memory-mapped devices. Both TZASC and TZPC utilise the NS bit for access control. The GP TEE implements secure storage using the sealing abstraction described previously, or to TEE-controlled hardware, e.g. Secure Element (SE).

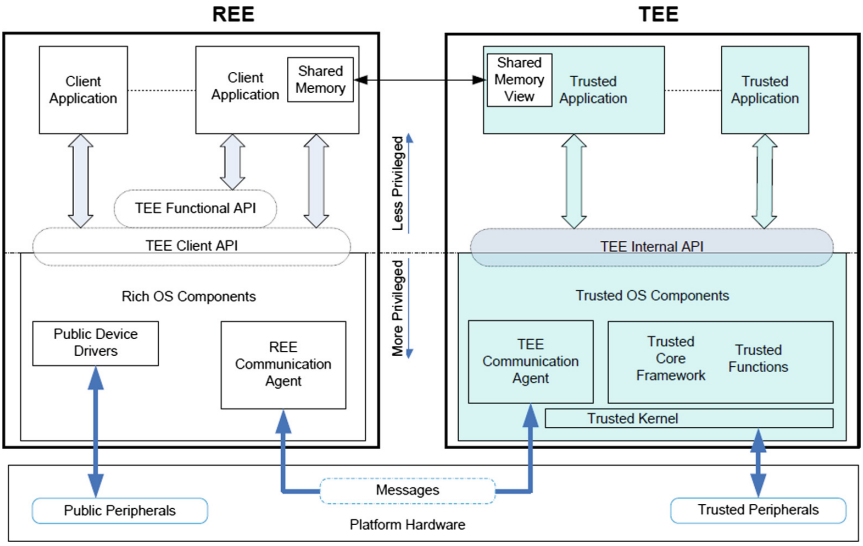

Fig. 1. GlobalPlatform TEE system architecture [22].

Credential Management. Security credentials are the evidence that a communicating party possesses for accessing privileged data and services. Credentials are typically programmed initially into a TEE during the personalisation phase

following the procurement of the SoC and TEE software, but prior to deployment. After this, a Trusted Service Manager (TSM) – incorporated into the device manufacturer or outsourced – is responsible for maintaining the TEE, its TAs and on-board credentials thereafter. We define TEE credentials as *the set, C, of key material, certificates and other authentication data issued by TSM that is provisioned into a TA*. TEE credentials may also comprise a key derived from a password-based key derivation function using a password from an operator, or encapsulated by biometrics, e.g. iris and fingerprint, or a behavioural model that maps device continuous data to authentication states [21].

2.1 Credential Management: Security and Functional Requirements

The GP Trusted Management Framework (TMF) does not stipulate particular secure channel protocols, but only that the TSM and TEE should mutually authenticate over a channel that preserves the *"integrity and the confidentiality of the exchanges,"* and addresses replay attacks against a Dolev-Yao adversary [10]. These basic requirements omit desirable features identified in existing literature [2,12,20], such as assurances that the target TEE is authentic and integral. This is typically realised using Remote Attestation (RAtt) where, firstly, system measurements are taken at boot-time or on-demand, which are collected and signed by a trusted measurer under a device-specific key; RAtt protocols subsequently transmit the measurements over a secure channel to a remote verifier, who evaluates the platform's integrity based on these values.

In sensitive deployments, mutually authenticating *both* end-points is useful during TEE-to-TEE communication; for example, between uploading backups from a GP TEE to a cloud-based backup enclave using Intel SGX. Here, RAtt protocols can be conducted independently for each end-point, or using a *mutual attestation* protocol wherein both parties are attested in a single protocol instance. We refer to such mutual attestation protocols, e.g. [20], as providing a Secure and Trusted Channel Protocol (STCP) in this work. We now formalise the baseline security and functional requirements from issues raised in related work and those stipulated in the GlobalPlatform specifications:

(S1) *Mutual key establishment*: a shared secret key is established for communication between the two entities.
(S2) *Forward secrecy*: the compromise of a particular session key should not affect past or subsequent protocol runs.
(S3) *Trust assurance*: the proposal shall allow third-parties to verify the target platform's integrity prior to credential transmission.
(S4) *Mutual trust verification*: both end-points shall successfully attest the state of the other before permitting the establishment of a secure channel.
(S5) *Mutual entity authentication*: each communicating end-point shall authenticate the other's identity to counter masquerading attempts.
(S6) *Denial of Service (DoS) resilience*: resource allocation shall be minimised at both end-points to prevent DoS conditions from arising.
(S7) *Key freshness*: the shared key shall be fresh to the session in order to prevent replay attacks.

(F1) *Avoidance of additional trust hardware*: the protocol shall avoid the need for additional security hardware, e.g. TPMs and SEs, other than the TEE.
(F2) *TEE agnosticism*: the protocols shall remain agnostic of the underlying TEE architecture to facilitate interoperability.

Setup Assumptions. A public-key infrastructure is assumed in which a Certificate Authority (CA) issues certificates to the TSM, TAs, and backup (BA), revocation (RA) and maintenance (MA) authorities used for managing backups, revoking credentials and physically maintaining devices respectively. The TEEs themselves are assumed to be trusted and to possess certified, device-specific attestation and command keys for signing quotes and requests to the TSM. Quotes are a widely-used remote attestation abstraction for TPMs and TEEs, comprising the TEE's identity and the platform integrity measurements collected by a TEE-resident trusted measurer. The resulting quote is signed using the attestation key and transmitted to the remote verifying authority. The credentials are assumed to be securely stored within the TEE, usually performed by encrypting them under a device-specific SRK, as well as secure means of random number generation and key derivation.

3 Migration

TEE migration is the process of transferring and re-provisioning credential data from TA_A to TA_B in distinct TEEs. Migration is crucial in preserving credentials during a device replacement or relocation, where credentials can be remotely transferred without incurring reinitialisation costs. Migrating credentials across TEEs has already attracted some attention in related literature [3,16]. We summarise these schemes and their contributions.

Arfaoui et al. [3] tackle the challenge of credential migration on GlobalPlatform TEEs. The authors introduce a trusted TEE Admin, which possesses a Security Domain (SD) with a TA and key-pair on the source device, to mediate and authenticate the authorisation and migration procedures. Two PKI-based protocols are developed for performing the migration between the target TA and the service provider's TA. Both protocols are subjected to formal verification using the AVISPA analysis tool. While the authors note the importance of remote attestation to verify the target TEE, it is not presented or verified as part of the protocol; it is also omitted during the credential transfer process between the TEEs. Moreover, *mutual* trust assurances between the service provider and TEE is not discussed.

Kostiainen et al. [16] tackle migration for TEE open credential platforms where service providers can provision arbitrary credentials, say, for virtualised access control cards. The authors propose encrypting and backing-up credentials on a trusted server using a tokenised password known only to the user. The credentials are migrated by re-entering the password, which is re-tokenised on the receiver device, and transmitted and verified by the backup server that releases the encrypted credentials. However, like [3], the proposal lacks trust assurances between the TSM and both TEEs, nor is it subjected to formal analysis.

3.1 Proposed High-Level Migration Procedure

Credentials must be deleted on the device from which they are migrated, while transferring them over a secure channel with mutual trust assumptions. In Fig. 2, we show how migration can be performed between remote TAs accounting for the shortcomings in related work, which comprises the following messages:

1. A mutual remote attestation protocol [2,12,20] is executed between TSM and TA_A to bootstrap a secure and mutually trusted channel (STCP).
2. TSM transmits the begin migration command to TA_A.
3. TA_A unseals credentials, C, from its secure storage for transmission.
4. TA_A acknowledges to TSM that the credentials were unsealed successfully.
5. A separate STCP instance is executed between (TSM, TA_B).
6. TSM instructs TA_B to prepare for credential provisioning.
7. TA_B acknowledges to TSM that it is ready to receive credentials.
8. TSM transmits the ID of TA_B to TA_A, e.g. IP address, to which to transmit the unsealed credentials.
9. An STCP is formed using mutual remote attestation between TA_A and TA_B.
10. The credential transfer occurs between TA_A and TA_B.
11. The transferred credential is provisioned into the secure storage of TA_B.

12–13. TA_B acknowledges its provisioning success to TA_A and TSM.

14. TSM instructs TA_A to delete its migrated credential(s).

15–16. TA_A deletes C and acknowledges its success to TSM.

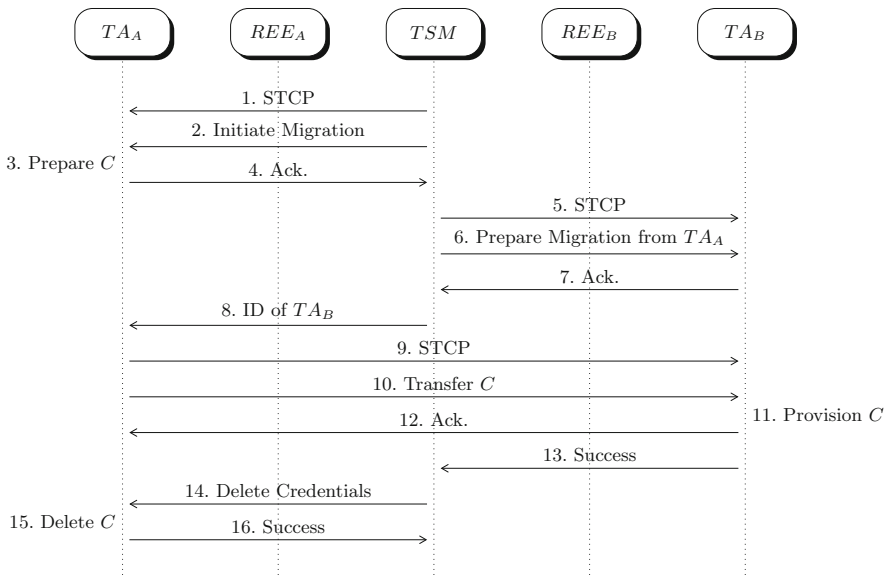

Fig. 2. Proposed TEE credential migration procedure.

The high-level procedure uses three STCPs with mutual attestation between (TSM, TA_A), (TSM, TA_B) and (TA_A, TA_B), thus addressing the absence of bi-directional trust assurances in existing work. The proposal avoids unnecessarily exposing credentials to the TSM by transmitting data directly between the mutually authenticated TAs. Implicitly, the protocol avoids specifying TEE-specific functionalities; rather, for F2 (TEE agnosticism), we abstract the protocol appropriately to allow migrations between heterogeneous TEEs by allowing either a GP TEE application or Intel SGX enclave to act as either TA. For TEE-specific implementation guidance, the reader is referred to existing work such as the GlobalPlatform TMF specifications [10], and the work by Arfaoui et al. [3] for managing and authorising SDs on the GP TEE. In Sect. 7, we specify the protocols and procedures formally, and detail an enhanced mutual attestation protocol for STCP for satisfying the remaining requirements from Sect. 2.1.

4 Revocation

Credentials are typically revoked when they reach the end of their predefined lifespan as part of a key rotation policy; if the OEM discovers a vulnerability in the TA or TEE kernel code, and the credentials were potentially compromised; or the device is retired from service, say, due to obsolescence. Revocation has attracted much attention in related TPM and smart card literature. Chen and Li [5] address credential revocation in TPM 2.0 Direct Anonymous Attestation (DAA). Like conventional group signatures, DAA allows the signer to demonstrate knowledge of its individual private key corresponding to the group's public key; however, this complicates revocation because the signer's identity is not revealed, even to the group manager. The authors review two solutions: *rekey-based*, where the issuer regularly updates its public key (which may or may not include its corresponding secret key), allowing only legitimate non-revoked signers to update their credentials accordingly; and *Verifier-Local Revocation (VLR)*, where the verifier inputs a revocation list (RL), to the DAA's verification function and accepts only signatures from signers $S \notin RL$.

Lueks et al. [18] address revoking attribute-based credentials (ABCs) for smart cards anonymously. Here, the RA possesses an RL of anonymous revocation values, $g^r_{\epsilon,v}$, submitted by the user or verifier (user- and system-instantiated revocation), where r is the revocation value in the user's credential. A revocation 'epoch', ϵ, corresponding to a time period, is used to provide unlinkability by re-computing and re-sending the new valid RLs to the verifiers at each epoch; that is, $RL_{\epsilon,V} = sort(\{g^r_{\epsilon,V} \mid r \in MRL\})$, where MRL is the master revocation list. Using bloom filters, this occupies only 4–8 MB for 2^{21} revoked credentials depending on the chosen probability tolerance.

Katzenbeisser et al. [15] propose revocation for TPM 1.2 using blacklisting and whitelisting. For blacklisting, a list of revoked keys, BL, is ordered into a hash chain and encrypted under the TPM's Storage Root Key (SRK); the final hash chain value is stored in a TPM register to maintain integrity. Before loading a key, k', each $k_i \in BL$ is fed sequentially into the TPM, where it is decrypted,

and $k' \stackrel{?}{=} k_i$ is tested. Whitelisting incrementally creates keyed hashes of each permitted key under the TPM's SRK and internal secure counter representing the whitelist's 'version'. A key is valid *iff* the keyed hash counter value matches the TPM's internal counter. Revocation is performed by incrementing the TPM's counter and updating all non-revoked hashes with the new value.

4.1 Proposed High-Level Revocation Procedure

Privacy-preserving credential schemes, e.g. DAA and anonymous credentials, are beneficial in verifying credentials without divulging or linking users' identities. However, to serve as an initial baseline, we scope the focus of this work is scoped to centralised deployments for applications where the concern of violating credential privacy has far fewer consequences than government electronic ID cards or TPMs on consumer devices. As such, we consider Industrial IoT (IIoT), logistics, and public devices in smart cities to serve as three potential application domains. Relaxing this constraint provides us with headway to pursue simpler, PKI-based solutions as a first step for providing TEE credential revocation with mutual attestation. We suggest two approaches for blacklisting and whitelisting using a trusted RA, the procedure for which illustrated in Fig. 3 and described as follows:

1. TSM and TA form a STCP using mutual remote attestation to verify each platforms' integrity and bootstrap a secure channel.

2–3. TSM instructs TA to reveal the current credentials in use, C, which are then unsealed from storage, e.g. encrypted in untrusted storage or an SE.

4. C is transmitted to the TSM over the STCP by TA.

5. The TSM forms a STCP with the revocation authority, RA, who maintains the master revocation list of white- or blacklisted credentials.

6–8. TSM submits C to RA, who returns a list of the revoked credentials in C, i.e. $RC \subseteq C$, from its master revocation list (MRL).

9–11. TSM instructs TA to revoke RC internally; TA performs the deletion and acknowledges its success.

Note that a malicious device may purposefully fail to update the status of RC internally and attempt to reuse revoked credentials. Consequently, the use of revoked credentials should be reported to MA responsible for decommissioning compromised devices, a simple protocol for which is listed in Steps A–E in Fig. 3 based on mutual attestation involving (MA, RA). Like [18], delegating revocation list management to RA removes the burden of potentially multiple verifiers synchronising a single list; the TSM can submit a lookup request to the RA, who queries the blacklist or whitelist in $O(1)$ using an associative array. Either black- or whitelisting can be performed in this model. For blacklisting, the RA maintains a master revocation list (MRL) of revoked credentials that should not be used in any transaction in which the MA submits credentials it wishes to revoke to RA (*maintainer-instantiated revocation*). Here, RA tests the revocation status of C by verifying $C \notin MRL$. Whitelisting, conversely, comprises a list of

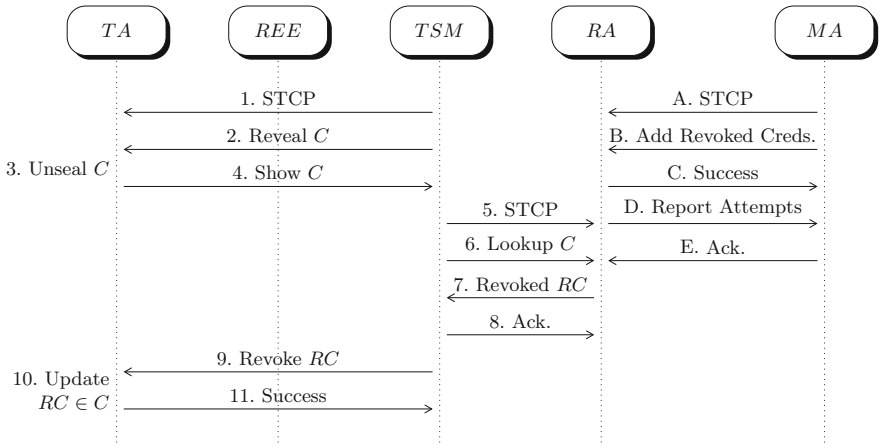

Fig. 3. Proposed credential revocation. *Steps A–E are independent of 1–11.*

only the permitted credentials; a credential is revoked by removing it from the whitelist and, if applicable, updating the list with its replacement. Revocation is tested by verifying $C \in MRL$.

5 Remote Backups

Backup is the process of securely retrieving the set of credentials, C, belonging to a TA for remote storage. In standard practice, backups underpin disaster recovery plans – as stipulated by ISO 27001:2013 [14] – for recovering data from corruption and accidental deletion. Backups may also constitute part of a data retention policy, where device data is used as evidence of regulatory adherence. Secure backup is beneficial when the original credential has non-trivial reinitialisation costs. Next, we examine related work in the backup of remote credentials aboard secure and trusted execution technologies.

Kostiainen et al. [17] address TEE credential backup, restoration and disabling on consumer mobile phones, proposing two solutions. The first uses a SE – a SIM card – in which the TEE credentials are protected under a SIM-specific key provisioned by its provider. This allows the user to uninstall a familiar hardware element, i.e. the SIM, before releasing the device for repairs or to lend to an untrusted user. On reinserting the SIM, an on-board TEE credential manager is used to decrypt and re-initialise the encrypted credentials. The second solution involves the use of a removable microcontroller to counter an honest-but-curious remote server, RS. RS possesses a shared key K_s with the TEE, and stores the backups using a secure counter for rollback protection. To prevent RS reading the credentials, the TEE encrypts them under a separate key, K, derived from a local counter on the microcontroller, and re-encrypts them under K_s.

Akram et al. [1] examine credential restoration for multi-application smart cards on smartphone SEs. The SE's Trusted Environment and Execution Man-

ager (TEM), which dynamically enforces the smart card's run-time security policies, is expanded to facilitate credential backups and restoration. A Backup and Restoration Manager (BRM) is added to the smart card software stack that interfaces with a TEM-resident backup token handler, which stores tokens issued by application service providers. The user first registers the BRM with a backup server (BS) and, when the user wishes to perform the backup, the BRM encrypts the token(s) and communicates them to BS. To restore data, such as to a new card, the user provides the BRM with his/her BS credentials to download the backed-up tokens. A secure channel is formed and the token(s) authenticate the credential restoration from the service provider(s).

5.1 Proposed High-Level Backup Procedure

We introduce a trusted Backup Authority (BA) responsible for storing retrieved credentials. This may be a cloud-based storage provider or Hardware Security Module (HSM) possessed by the credential issuing authority; the precise means by which BA securely stores credentials is out-of-scope in this work. Importantly, we note that credential restoration can be performed by executing the proposed Remote Update procedure in Sect. 6. The proposed procedure between the target TA and BA is shown in Fig. 4, which is described as follows:

1–4. TSM and BA establish a secure and trusted channel with mutual attestation, and TSM requests BA to prepare for backup.
5–6. TSM forms an STCP with TA and commands it prepare the credentials for remote backup; TSM also provides the identity of BA.
7–8. TA unseals the credentials to transmit to BA, and notifies TSM that they were unsealed successfully.

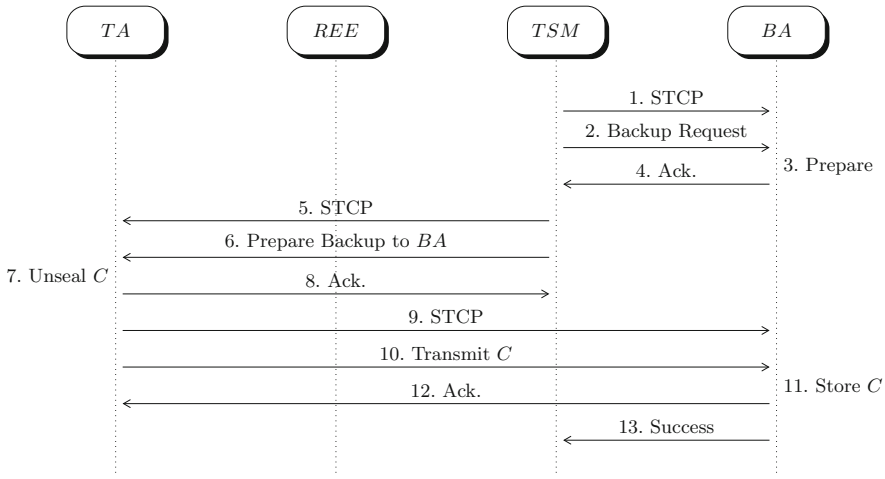

Fig. 4. Proposed high-level remote credential backup protocol.

9–12. TA and BA form a STCP over which C is transmitted to and stored securely by BA. While TSM is considered trusted, a direct connection between TA and BA mitigates the risk of unnecessary credential exposure to TSM.
13. BA notifies TSM that C from TA was backed-up successfully.

6 Remote Updates

Remotely updating credentials is beneficial during routine renewal schedules; for example, with X.509 certificates that reach their validity expiry date, or the device is relocated and the organisational unit to which the credential is issued is no longer valid. Generally speaking, update is the process by which an outdated credential, c_i, is securely replaced by a freshly issued c_i'. Once replaced, c_i should be revoked to prevent reusing obsolete credentials. Little work has been conducted regarding TEE credential updates, which is likely due to the simplicity of a TSM transmitting a new credential over a secure channel or, indeed, the similarity with backup restoration (Sect. 5). Updates can be considered a variation of backup restoration where c_i' is retrieved from an update server; the addition is revoking c_i, achievable using the revocation process proposed in Sect. 4. Next, we describe how this can be achieved with mutual attestation.

6.1 Proposed High-Level Update Procedure

We reintroduce the maintenance authority (MA) from Sect. 4, which issues credential updates as part of a standard rotation policy. If desired, the MA is also responsible for registering obsolete credentials with the revocation authority. The high-level update mechanism is as follows (Fig. 5):

1. MA, who provides the updated credentials, establishes an STCP with TSM.
2–3. MA notifies the TSM of an updated credential. This may include identities of which TEEs need updated or all TEEs.
4–5. An STCP is conducted between (TSM, TA) and an update preparation command is transmitted to TA, along with an optional ID of MA from whom to retrieve the update.
6. TA is locked, i.e. prevented from interacting with the REE, until the update is performed in order to prevent the use of outdated credentials.
7. TA acknowledges to TSM that it is ready to update.
8–10. TA establishes a STCP with MA to receive the credential update; MA transmits the updated credential, c_i', to TA.
11–12. TA seals c_i' to its secure storage for future use; c_i should be deleted internally before unlocking. TA acknowledges that c_i' was initialised successfully.
13–16. (MA, RA) use an STCP to white- or blacklist the obsolete credential, c_i. Lastly, MA acknowledges completion of the update procedure to TSM.

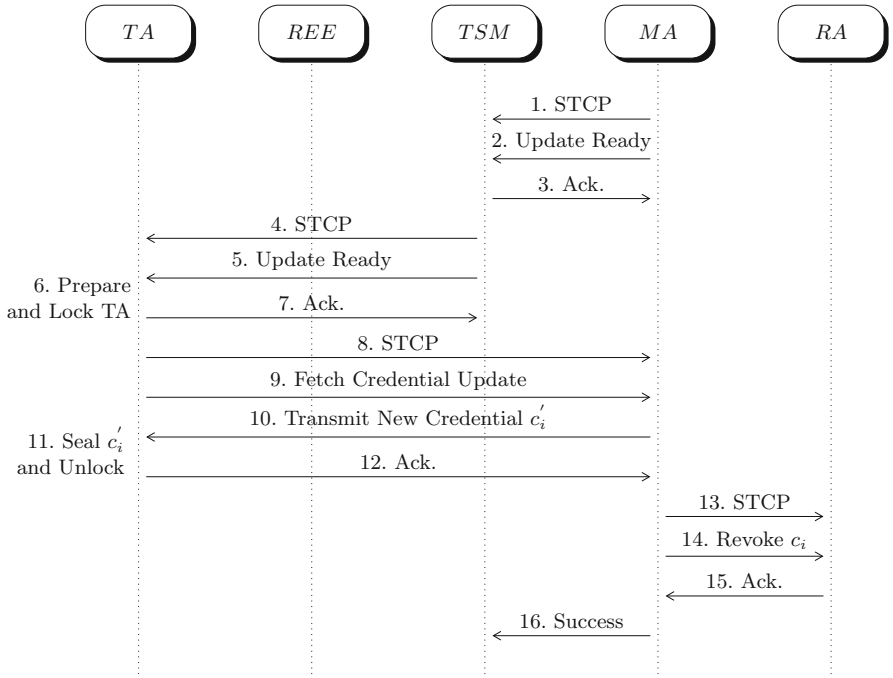

Fig. 5. Proposed credential update procedure.

7 Proposed Protocol Analysis

We now formalise the protocols from the high-level procedures presented previously, which are listed in Protocols 1 to 6 using the notation from Table 1. In Sect. 2.1, we outlined the requirements and assumptions of the protocols, which are referred to throughout. Each proposed protocol is underpinned by an enhancement of the BTP mutual attestation protocol in [20] for establishing the STCP between the TEEs. This protocol (Protocol 6), which establishes a TEE-to-TEE secure channel after exchanging and verifying attestation quotes, is simplified to support authenticated encryption (AE), e.g. AES in GCM mode, rather than a non-AE symmetric scheme with an additional HMAC as in original proposal. This simplification is aimed at reducing protocol implementation complexity and improving performance based on existing benchmarks [13]. The protocol is based on ephemeral Diffie-Hellman key agreement that achieves session forward secrecy (S2), mutual key establishment (S1) and key freshness (S7). Moreover, TEE quotes are mutually exchanged for verifying the integrity of each platform, thus satisfying S3 and S4 (mutual trust verification). The signed attestation values, command instructions, e.g. *Prep_Backup* and *Revoke_Success*, and the shared secret provides mutual entity authentication (S5).

Crucially, the protocols avoid the use of additional trusted hardware, such as TPMs, secure elements and smart cards (F1). The protocols are designed

Protocol 1. Proposed Migration Procedure with BTP (MPBT)

1: Execute BTP (TSM, TA_1)
2: $TSM \rightarrow TA_1 : [(Init\_Migrate \mid\mid X_{TSM})\sigma_{TSM}]_{AE_K}$
3: $TA_1 \rightarrow TSM : [(TA_1\_Ack \mid\mid X_{TA1})\sigma_{TA1}]_{AE_K}$
4: Execute BTP (TSM, TA_2)
5: $TSM \rightarrow TA_2 : [(Prep\_Migrate \mid\mid X'_{TSM})\sigma_{TSM}]_{AE_{K'}}$
6: $TA_2 \rightarrow TSM : [(TA_2\_Ack \mid\mid X'_{TA2})\sigma_{TA2}]_{AE_{K'}}$
7: $TSM \rightarrow TA_1 : [(ID_{TA2} \mid\mid X_{TSM})\sigma_{TSM}]_{AE_K}$
8: Execute BTP (TA_1, TA_2)
9: $TA_1 \rightarrow TA_2 : [(C \mid\mid X''_{TA1})\sigma_{TA1}]_{AE_{K''}}$
10: $TA_2 \rightarrow TA_1 : [(TA_2\_Ack \mid\mid X''_{TA2})\sigma_{TA2}]_{AE_{K''}}$
11: $TA_2 \rightarrow TSM : [(TA_2\_Done \mid\mid X'_{TA2})\sigma_{TA2}]_{AE_{K'}}$
12: $TSM \rightarrow TA_1 : [(Delete\_Creds \mid\mid X_{TSM})\sigma_{TSM}]_{AE_K}$
13: $TA_1 \rightarrow TSM : [(TA_1\_Success \mid\mid X_{TA1})\sigma_{TA1}]_{AE_K}$

Protocol 2. Proposed Revocation Lookup with BTP (RLBT)

1: Execute BTP (TSM, TA)
2: $TSM \rightarrow TA : [(Reveal\_Creds \mid\mid X_{TSM})\sigma_{TSM}]_{AE_K}$
3: $TA \rightarrow TSM : [(C \mid\mid X_{TA})\sigma_{TA}]_{AE_K}$
4: Execute BTP (TSM, RA)
5: $TSM \rightarrow RA : [(Lookup \mid\mid C \mid\mid X'_{TSM})\sigma_{TSM}]_{AE_{K'}}$
6: $RA \rightarrow TSM : [(Revoked \mid\mid RC \mid\mid X'_{RA})\sigma_{RA}]_{AE_{K'}}$
7: $TSM \rightarrow RA : [(TSM\_Ack \mid\mid X'_{TSM})\sigma_{TSM}]_{AE_{K'}}$
If $RC \neq \emptyset$:
8: $TSM \rightarrow TA : [(Revoke \mid\mid RC \mid\mid X_{TSM})\sigma_{TSM}]_{AE_K}$
9: $TA \rightarrow TSM : [(Revoke\_Success \mid\mid X_{TA})\sigma_{TA}]_{AE_K}$

Protocol 3. Proposed Revocation Procedure with BTP (RPBT)

1: Execute BTP (MA, RA)
2: $MA \rightarrow RA : [(Revoke \mid\mid C \mid\mid X_{MA})\sigma_{MA}]_{AE_K}$
3: $RA \rightarrow MA : [(Revoke\_Success \mid\mid X_{RA})\sigma_{RA}]_{AE_K}$
4: If reported credentials ($RepC \neq \emptyset$):
$RA \rightarrow MA : [(Report \mid\mid RepC \mid\mid X_{RA})\sigma_{RA}]_{AE_K}$
$MA \rightarrow RA : [(Report\_Success \mid\mid X_{MA})\sigma_{MA}]_{AE_K}$

Protocol 4. Proposed Backup Procedure with BTP (BPBT)

1: Execute BTP (TSM, BA)
2: $TSM \rightarrow BA : [(Prep\_Backup \mid\mid X_{TSM})\sigma_{TSM}]_{AE_K}$
3: $BA \rightarrow TSM : [(BA\_Ack \mid\mid X_{BA})\sigma_{BA}]_{AE_K}$
4: Execute BTP (TSM, TA)
5: $TSM \rightarrow TA : [(Prep\_Backup \mid\mid ID_{BA} \mid\mid X'_{TSM})\sigma_{TSM}]_{AE_{K'}}$
6: $TA \rightarrow TSM : [(TA\_Ack \mid\mid X'_{TA})\sigma_{TA}]_{AE_{K'}}$
7: Execute BTP (TA, BA)
8: $TA \rightarrow BA : [(C \mid\mid X''_{TA})\sigma_{TA}]_{AE_{K''}}$
9: $BA \rightarrow TA : [(Backup\_Ack \mid\mid X''_{BA})\sigma_{BA}]_{AE_{K''}}$
10: $BA \rightarrow TSM : [(Backup\_Success \mid\mid X_{BA})\sigma_{BA}]_{AE_K}$

Protocol 5. Proposed Update Procedure with BTP (UPBT)

1: Execute BTP (MA, TSM)
2: $MA \rightarrow TSM$: $[(Update\_Ready \mid\mid X_{MA})\sigma_{MA}]_{AE_K}$
3: $TSM \rightarrow MA$: $[(TSM\_Ack \mid\mid X_{TSM})\sigma_{TSM}]_{AE_K}$
4: Execute BTP (TSM, TA)
5: $TSM \rightarrow TA$: $[(Prep\_Update \mid\mid X'_{TSM})\sigma_{TSM}]_{AE_{K'}}$
6: $TA \rightarrow TSM$: $[(TA\_Ack \mid\mid X'_{TA})\sigma_{TA}]_{AE_{K'}}$
7: Execute BTP (TA, MA)
8: $TA \rightarrow MA$: $[(Fetch\_Update \mid\mid X''_{TA})\sigma_{TA}]_{AE_{K''}}$
9: $MA \rightarrow TA$: $[(c'_i \mid\mid X''_{MA})\sigma_{MA}]_{AE_{K''}}$
10: $TA \rightarrow MA$: $[(New\_Cred\_Ack \mid\mid X''_{TA})\sigma_{TA}]_{AE_{K''}}$
11: Execute BTP (MA, RA)
12: $MA \rightarrow RA$: $[(Revoke \mid\mid c_i \mid\mid X'''_{MA})\sigma_{MA}]_{AE_{K'''}}$
13: $RA \rightarrow MA$: $[(Revoke\_Success \mid\mid X'''_{RA})\sigma_{RA}]_{AE_{K'''}}$
14: $MA \rightarrow TSM$: $[(Update\_Success \mid\mid X_{MA})\sigma_{MA}]_{AE_K}$

Protocol 6. Adapted Bi-directional Trust Protocol (BTP) from [20]

1: $A \rightarrow B : ID_A \mid\mid ID_B \mid\mid n_A \mid\mid g^A \mid\mid AR_B$
2: $B \rightarrow A : ID_B \mid\mid ID_A \mid\mid n_B \mid\mid g^B \mid\mid \left[(X_B)\sigma_B \mid\mid (V_B)\sigma_B\right]_{AE_K} \mid\mid AR_A$
$X_B = H(ID_A \mid\mid ID_B \mid\mid g^A \mid\mid g^B \mid\mid n_A \mid\mid n_B)$
$V_B = Q_B \mid\mid n_B \mid\mid n_A$
3: $A \rightarrow B : [\ (X_A)\sigma_A \mid\mid (V_A)\sigma_A\ ]_{AE_K}$
$X_A = H(ID_A \mid\mid ID_B \mid\mid g^A \mid\mid g^B \mid\mid n_A \mid\mid n_B)$
$V_A = Q_A \mid\mid n_A \mid\mid n_B$

to incorporate abstract TAs, which are verified using the quoting abstraction, whether they be Intel SGX enclaves of GP TEE TAs, thus providing TEE agnosticism (F2). Note, however, that this abstracts away the precision of related work, e.g. Arfaoui et al. [3], which addresses migration specifically in the context of the GP TEE. Such work incorporates GP TEE-specific entites, such as security domains (SDs) and root SDs, which do not exist on Intel SGX or earlier TPM-based TEEs like Intel TXT [11]. As such, users of this work should be aware of the implementation specifics when deploying these protocols; we refer users to [3] and [10] for guidance for GP TEEs, and [6] for Intel SGX.

Formal Symbolic Verification. Scyther by Cremers [7] was employed to verify the correctness of the proposed protocols. A protocol is first specified in the Scyther description language, comprising communicating parties (roles), messages and the desired security properties (claims). Scyther verifies whether the protocol specification satisfies those claims under the 'perfect cryptography' assumption, whereby an adversary learns nothing from an encrypted message unless the decryption key is known, against all possible behaviours of a Dolev-Yao adversary. Despite the challenge of security protocol verification being undecidable in general, many practical protocols can be proven correct; notably, Scyther has been used to verify IKEv1 and IKEv2, and the ISO/IEC 9798 authentication

Table 1. Protocol notation.

Notation	Description
TSM	TEE trusted service manager
BA	Credential backup authority
MA	Device maintenance authority
RA	Revocation authority
TA_X	TEE trusted application on device X
n_X	Secure random nonce generated by X
$H(D)$	Secure one-way hash function, H, on D
$X \rightarrow Y$	Message transmission from X to Y
ID_X	Identity of X
$A \mid\mid B$	Concatenation of A and B
g^X	Diffie-Hellman exponentiation of X
AR_X	Attestation request on target entity X
Q_X	Attestation quote from TEE X
$(A)\sigma_X$	Signed message A from X under a private-public key-pair (K, P)
$[m]_{AE_K}$	Message m is encrypted using authenticated encryption under session key K derived from the protocol's shared secret
D'	Data specific to a separate session to D

protocol family [8]. We analyse all protocols using Scyther, testing for the secrecy of transmitted quotes from both parties, e.g. (`Secret, qta1`) and credentials (`Secret, c`); aliveness (`Alive`); replay protection, i.e. non-injective agreement (`Niagree`) and non-injective synchronisation, (`Nisynch`), defined in [7]; session key secrecy (`SKR, K`); and the reachability of all entities, e.g. (`Reachable, TA`). We publicly release the protocol specifications for future research by the community (see Sect. 1). Scyther found no attacks on any protocol.

8 Conclusion

TEEs are emerging as a flexible mechanism for providing a range of assurances regarding the on-device protection of security-critical applications, credentials and related data. In this work, we presented a suite of proposals for remote TEE credential management using mutual attestation for secure *migration*, *revocation*, *backups*, and *updates*. After summarising the features of leading TEE implementations, we formalised the threat model, requirements and assumptions for a typical TEE credential deployment in a centralised setting. Next, we reviewed the state-of-the-art for each credential management challenge, before proposing

procedures and protocols for securely realising these notions. The protocols were formalised and subjected to symbolic verification using Scyther, which found no attacks under the Dolev-Yao adversarial model; the protocol specifications are also published publicly for further research. In future work, we aim to incorporate privacy-preserving attestation into our protocol suite, which we considered out-of-scope in this work for centralised deployments, through the use of techniques like DAA and Blacklistable Anonymous Credentials (BLACs). We also wish to address decentralised deployments, where devices have intermittent or potentially no access to a centralised TSM.

Acknowledgements. Carlton Shepherd is supported by the EPSRC and the British government as part of the Centre for Doctoral Training in Cyber Security at Royal Holloway, University of London (EP/K035584/1).

References

1. Akram, R.N., Markantonakis, K., Mayes, K.: Recovering from a lost digital wallet. In: Embedded and Ubiquitous Computing, pp. 1615–1621. IEEE (2013)
2. Akram, R.N., Markantonakis, K., Mayes, K., Bonnefoi, P.F., Sauveron, D., Chaumette, S.: An efficient, secure and trusted channel protocol for avionics wireless networks. In: 35th Digital Avionics Systems Conference. IEEE (2016)
3. Arfaoui, G., Gharout, S., Lalande, J.-F., Traoré, J.: Practical and privacy-preserving TEE migration. In: Akram, R.N., Jajodia, S. (eds.) WISTP 2015. LNCS, vol. 9311, pp. 153–168. Springer, Cham (2015). https://doi.org/10.1007/978-3-319-24018-3_10
4. Brickell, E., Li, J.: Enhanced privacy ID from bilinear pairing for hardware authentication and attestation. Int. J. Inf. Priv. Secur. Integrity **1**(1), 3–33 (2011)
5. Chen, L., Li, J.: Revocation of direct anonymous attestation. In: Chen, L., Yung, M. (eds.) INTRUST 2010. LNCS, vol. 6802, pp. 128–147. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-25283-9_9
6. Costan, V., Devadas, S.: Intel SGX Explained. IACR Cryptology ePrint (2016). https://eprint.iacr.org/2016/086.pdf
7. Cremers, C.J.F.: The scyther tool: verification, falsification, and analysis of security protocols. In: Gupta, A., Malik, S. (eds.) CAV 2008. LNCS, vol. 5123, pp. 414–418. Springer, Heidelberg (2008). https://doi.org/10.1007/978-3-540-70545-1_38
8. Cremers, C.: Key exchange in IPsec revisited: formal analysis of IKEv1 and IKEv2. In: Atluri, V., Diaz, C. (eds.) ESORICS 2011. LNCS, vol. 6879, pp. 315–334. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-23822-2_18
9. GlobalPlatform: TEE Protection Profile (v1.2) (2014)
10. GlobalPlatform: TEE Management Framework (TMF), v1.0 (2016)
11. Greene, J.: Intel Trusted eXecution Technology (TXT): Hardware-based technology for enhancing server platform security. Intel, Inc., Technical report (2012)
12. Greveler, U., Justus, B., Löhr, D.: Mutual remote attestation: enabling system cloning for TPM based platforms. In: Meadows, C., Fernandez-Gago, C. (eds.) STM 2011. LNCS, vol. 7170, pp. 193–206. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-29963-6_14
13. Gueron, S.: AES-GCM for efficient authenticated encryption - ending the reign of HMAC-SHA-1. In: Real World Cryptography (2013)

14. International Standards Organisation: ISO 27001:2013 - Information Security Management (2013). https://www.iso.org/standard/54534.html
15. Katzenbeisser, S., Kursawe, K., Stumpf, F.: Revocation of TPM keys. In: Chen, L., Mitchell, C.J., Martin, A. (eds.) Trust 2009. LNCS, vol. 5471, pp. 120–132. Springer, Heidelberg (2009). https://doi.org/10.1007/978-3-642-00587-9_8
16. Kostiainen, K., Asokan, N., Afanasyeva, A.: Towards user-friendly credential transfer on open credential platforms. In: Lopez, J., Tsudik, G. (eds.) ACNS 2011. LNCS, vol. 6715, pp. 395–412. Springer, Heidelberg (2011). https://doi.org/10.1007/978-3-642-21554-4_23
17. Kostiainen, K., Asokan, N., Ekberg, J.-E.: Credential disabling from trusted execution environments. In: Aura, T., Järvinen, K., Nyberg, K. (eds.) NordSec 2010. LNCS, vol. 7127, pp. 171–186. Springer, Heidelberg (2012). https://doi.org/10.1007/978-3-642-27937-9_12
18. Lueks, W., Alpár, G., Hoepman, J.H., Vullers, P.: Fast revocation of attribute-based credentials for both users and verifiers. Comput. Secur. **67**, 308–323 (2017)
19. Sadeghi, A.R., Wachsmann, C., Waidner, M.: Security and privacy challenges in industrial Internet of Things. In: 52nd Design Automation Conference. ACM (2015)
20. Shepherd, C., Akram, R.N., Markantonakis, K.: Establishing mutually trusted channels for remote sensing devices with trusted execution environments. In: 12th International Conference on Availability, Reliability and Security. ACM (2017)
21. Shepherd, C., Akram, R.N., Markantonakis, K.: Towards trusted execution of multi-modal continuous authentication schemes. In: Proceedings of the 32nd Symposium on Applied Computing. pp. 1444–1451. ACM (2017)
22. Shepherd, C., et al.: Secure and trusted execution: past, present, and future - a critical review in the context of the Internet of Things and cyber-physical systems. In: 15th IEEE International Conference on Trust, Security and Privacy in Computing and Communications. pp. 168–177. IEEE (2016)