Hierarchical-CPK-Based Trusted Computing Cryptography Scheme*

Fajiang Yu[1,2], Tong Li[2], Yang Lin[2], and Huanguo Zhang[1,2]

[1] School of Computer, Wuhan University, Wuhan, Hubei, 430072, P.R. China
fjyu@whu.edu.cn
[2] Key Laboratory of Aerospace Information Security and Trusted Computing,
Ministry of Education in China

Abstract. PKI-based trusted computing platform (TCP) requires platform users to apply for multiple Platform Identity Key (PIK) certificates to provide remote attestation, users must pay the fee of digital certificates, which increases users' economic burdens and leads there is hardly any TCP has really performed the core function of trusted computing, platform remote attestation, so the application of TCP is not very wide. This paper presents a trusted computing cryptography scheme based on Hierarchical Combined Public Key (HCPK), which can reduce the risk of single Private Key Generator (PKG), and let the verifier authenticate TCP directly without third party, so platform users do not need to apply additional digital certificates. This scheme can reduce users' cost of using TCP, and encourage the development of TCP application.

Keywords: Trusted Computing, Combined Public Key (CPK), Hierarchical Combined Public Key (HCPK), Trusted Cryptography Module (TCM).

1 Introduction

Platform remote attestation is one of core functions of trusted computing [1,2]. Before platform providing remote attestation to a verifier, platform users must apply for Attestation Identity Key (AIK) certificate from privacy CA based on Endorsement Key (EK) in Trusted Platform Module (TPM). Because of high cost of PKI CA construction and operation, users should pay some administration fees of AIK certificates. For protecting the privacy of platform identity, users need apply for multiple AIK certificates for different applications, which further increases users' economic burdens. So there is hardly any TCP has really performed the core function of trusted computing, platform remote attestation, and the application of TCP is not very wide.

* This work is supported by the National Natural Science Foundation of China, Grant No: 60673071, 60970115, 91018008, and the Fundamental Research Funds for the Central Universities in China, Grant No: 3101044.

J.M. Alcaraz Calero et al. (Eds.): ATC 2011, LNCS 6906, pp. 149–163, 2011.

For reducing the dependence on privacy CA, Trusted Computing Group (TCG) added one method named Direct Anonymous Attestation (DAA) in TPM 1.2 specifications. DAA employs the methods including Camenisch Lysyanskaya (CL) signature, zero knowledge proof based on discrete logarithm, Fiat Shamir heuristics, group signature and etc. When using DAA, a verifier can affirm that one requesting platform is a host of one real TPM, the verifier can not obtain real identity information about TPM. DAA needs to provide zero knowledge proof at least for three times during one process of identity authentication, which leads low efficiency and implementation complexity. There is few practical applications of DAA.

Being aware of the importance of trusted computing which is a basic security solution for computing platform, as early as in 2006, China Cryptography Administration began to collect correlative institutions for writing *Trusted Computing Platform Cryptography Scheme* and *Technology Specification of Cryptographic Support Platform for Trusted Computing.* China Cryptography Administration released *Functionality and Interface Specification of Cryptographic Support Platform for Trusted Computing* in 2007, which requires Trusted Cryptography Module (TCM, corresponding to TPM) to use *state public key cryptographic algorithm SM2 based on elliptic curves (ECC)* of China. There is also some difference between the subscription process of Platform Identity Key (PIK, corresponding to AIK) certificate and normal certificate, so users can not apply for PIK certificate from current CAs which has supported SM2, and we never see that one privacy CA for TCM is in operation.

In 2003, Identity-Based Combined Public Key (IBCPK, CPK called for short) is presented by one famous cryptography expert of China, Nan Xianghao, in *a profile to network security techniques* [3] for the first time. CPK suffered conspiracy attack and private key collision [4,5], some solutions has been given out [6,7], and CPK also has been developed from version 1.0 to 5.0 [8,9,10,11]. In CPK, Entity's identity name just is public key, there is no need of online database for managing public keys. CPK can form very large key space based on small scale of key seed matrix, directly distribute public key seed matrix to entities, and the entity can be authenticated directly without third party. Comparing with Identity-Based Encryption (IBE) [12,13] based on bilinear map, CPK has high efficiency performance.

This paper presents a trusted computing cryptography scheme based on CPK, which can let the verifier authenticate TCP directly without third party, platform users do not need to apply additional digital certificates. This scheme can reduce users' cost of using TCP, and encourage the development of TCP application. ECC has been implemented in TCM, and CPK is presented by Chinese expert for the first time, which have created good conditions for our research.

2 PKI-Based Trusted Computing Cryptography Scheme

PKI-based TCM key architecture is shown as Figure 1. The manufacture generates an EK during manufacturing stage of TCM. EK is an asymmetric key pair which is stored in the non-volatile protected storage area in the TCM. EK also can be regenerated by the user before obtaining platform ownership. One TCM only have just one EK during its life cycle. Before using TCP, users must take ownership of the platform at first. When taking ownership, TCM generates a Storage Root Key (SRK), which is used to protect users' storage key, sign key, seal key and etc.

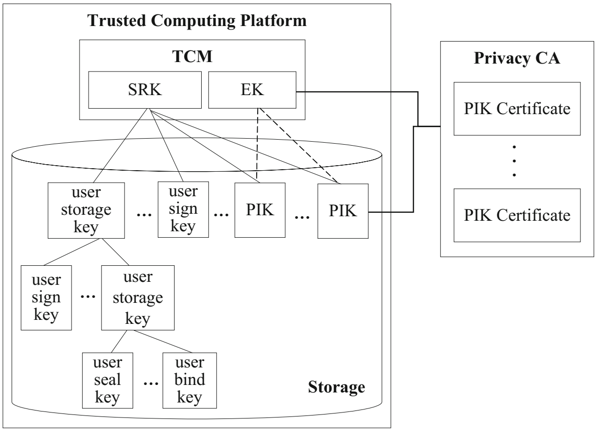

Fig. 1. PKI-based TCM key architecture

Platform remote attestation is one of core functions of trusted computing, at first the platform should prove its identity is trusted. In order to protect privacy, the platform doesn't directly use EK to sign for identity authentication. Users have to request TCM to generate PIK, and apply for PIK certificate from private CA based on EK. The privacy of corresponding relationship between EK and PIK is protected by privacy CA. PIK private key is also protected by SRK. Users should ask TCM to generate different PIKs and apply for different PIK certificates for different applications. Then one TCM and its host platform may have multiple PIKs. Thus, users can use different PIK for identity authentication in different situations, in order to protect the privacy of platform identity.

The procedures of generating of PIK, applying for, signing and activating PIK certificate are shown as Figure 2.

1. Platform user sends a command `MakeIdentity` to TCM via TCM Service Module (TSM). TCM generates a PIK and encrypt private part of PIK with SRK. Then TCM use private key of PIK to sign the digest value of public key of private CA and public part of PIK, the signature is $\mathsf{PIKSign} = \mathrm{Sign}(\mathsf{PIK}_{\mathsf{Pri}}, H(\mathsf{CAK}_{\mathsf{Pub}} \parallel \mathsf{PIK}_{\mathsf{Pub}}))$. TCM returns public part of PIK $\mathsf{PIK}_{\mathsf{Pub}}$ and $\mathsf{PIKSign}$.

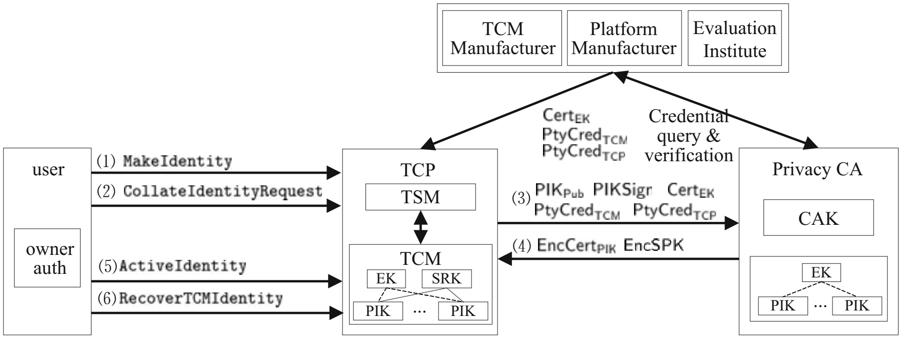

Fig. 2. Procedures of generating of PIK, applying for, signing and activating PIK certificate

2. Platform user sends a request `CollateIdentityRequest` to TSM for getting EK certificate $\mathsf{Cert}_{\mathsf{EK}}$ and property certificate for TCM and its host platform from evaluation institute and manufacturer $\mathsf{PtyCred}_{\mathsf{TCM}}$, $\mathsf{PtyCred}_{\mathsf{TCP}}$.
3. Platform user sends the message including $\mathsf{PIK}_{\mathsf{Pub}}$, $\mathsf{PIKSign}$, $\mathsf{Cert}_{\mathsf{EK}}$, $\mathsf{PtyCred}_{\mathsf{TCM}}$, $\mathsf{PtyCred}_{\mathsf{TCP}}$ to private CA and applies for PIK certificate.
4. Privacy CA verifies $\mathsf{PtyCred}_{\mathsf{TCP}}\mathsf{PtyCred}_{\mathsf{TCM}}\mathsf{Cert}_{\mathsf{EK}}$, then it will use $\mathsf{PIK}_{\mathsf{Pub}}$ to verify $\mathsf{PIKSign}$. After these two verifications, privacy CA generates and signs PIK certificate $\mathsf{Cert}_{\mathsf{PIK}}$, then randomly generates a symmetric key $\mathsf{sessionKey}$ and use it to encrypt $\mathsf{Cert}_{\mathsf{PIK}}$ to get $\mathsf{EncCert}_{\mathsf{PIK}} = \mathrm{AEnc}(\mathsf{sessionKey}, \mathsf{Cert}_{\mathsf{PIK}})$. Finally, it use the public key of EK to encrypt the digest of $\mathsf{sessionKey}$ and public part of PIK to get $\mathsf{EncSPK} = \mathrm{SEnc}(\mathsf{EK}_{\mathsf{Pub}}, \mathsf{sessionKey} \parallel \mathrm{H}(\mathsf{PIK}_{\mathsf{Pub}}))$. Private CA sends $\mathsf{EncCert}_{\mathsf{PIK}}$, EncSPK to TCP.
5. Platform user sends a request `ActiveIdentity` to TCM via TSM. TCM use the private key of EK to decrypt EncSPK, $\mathsf{sessionKey} \parallel \mathrm{H}(\mathsf{PIK}_{\mathsf{Pub}}) = \mathrm{SDec}(\mathsf{EK}_{\mathsf{Pri}}, \mathsf{EncSPK})$. We can determine whether $\mathsf{sessionKey}$ decrypted is right by judging the correctness of $\mathrm{H}(\mathsf{PIK}_{\mathsf{Pub}})$, while only the valid TCP can get the correct $\mathsf{sessionKey}$. TCM returns $\mathsf{sessionKey}$.
6. Platform user sends a request `RecoverTCMIdentity` to TSM, TSM uses $\mathsf{sessionKey}$ to decrypt $\mathsf{EncCred}_{\mathsf{PIK}}$, and gets PIK certificate $\mathsf{Cert}_{\mathsf{PIK}} = \mathrm{ADec}(\mathsf{sessionKey}, \mathsf{EncCert}_{\mathsf{PIK}})$.

3 CPK Introduction

The mathematical base of CPK is the following ECC combination theorem [4]:

Theorem 1 (ECC Combination Theorem). *ECC parameters are $\langle p, a, b, G, n\rangle$, that is given one elliptic curve E based on the selected finite field F_p: $y^2 \equiv (x^3 + ax + b)(\text{mod } p)$. G is the generator of one additive cyclic sub-group of points on E. n is the order of this group. If there are $h(h \in \mathbb{Z}, 1 < h < n)$ ECC key pairs: $(d_1, Q_1), (d_2, Q_2), \ldots, (d_h, Q_h)$, the summary of these h private* *keys is denoted as d, the summary of these h public keys is denoted as Q, that is $d = \left(\Sigma_{i=1,2,\ldots,h} d_i\right) \bmod n, Q = \Sigma_{i=1,2,\ldots,h} Q_i$, then (d, Q) also is one ECC key pair.*

Proof (of ECC Combination Theorem).
Because $(d_1, Q_1), (d_2, Q_2), \ldots, (d_h, Q_h)$ are ECC key pairs, then

$$\begin{aligned}
&Q_1 = d_1 G, Q_2 = d_2 G, \ldots, Q_h = d_h G \\
&Q = d_1 G + d_2 G + \ldots + d_h G \\
&Q = \big((d_1 + d_2 + \ldots + d_h)(\text{mod } n)\big) G = dG
\end{aligned}$$

So (d, Q) is a ECC key pair.

The basic components of CPK include private key seed matrix $(d_{ij})_{m\times h}$, public key seed matrix $(Q_{ij})_{m\times h}$, the mapping function set F, and the algorithm of combining public and private key Alg_{KG} [8]:

1. **Private Key Seed Matrix** $(d_{ij})_{m\times h}$

$$(d_{ij})_{m\times h} = \begin{pmatrix} d_{11} & d_{12} & \ldots & d_{1h} \\ d_{21} & d_{22} & \ldots & d_{2h} \\ \vdots & \vdots & \vdots & \vdots \\ d_{m1} & d_{m2} & \ldots & d_{mh} \end{pmatrix}$$

where $h, d_{ij}, d_{i'j'} \in \mathbb{Z}, 1 < h, d_{ij}, d_{i'j'} < n, i, i' \in \mathbb{Z}_m, j, j' \in \mathbb{Z}_h$. Only under the condition $i = i'$ and $j = j'$, $d_{ij} = d_{i'j'}$, otherwise $d_{ij} \neq d_{i'j'}$. $(d_{ij})_{m\times h}$ is just stored in Private Key Generator (PKG) as a secret.

2. **Public Key Seed Matrix** $(Q_{ij})_{m\times h}$

$$(Q_{ij})_{m\times h} = \begin{pmatrix} Q_{11} & Q_{12} & \ldots & Q_{1h} \\ Q_{21} & Q_{22} & \ldots & Q_{2h} \\ \vdots & \vdots & \vdots & \vdots \\ Q_{m1} & Q_{m2} & \ldots & Q_{mh} \end{pmatrix}$$

where $Q_{ij} = d_{ij} G, i \in \mathbb{Z}_m, j \in \mathbb{Z}_h$. $(Q_{ij})_{m\times h}$ is public to all entities.

3. **Mapping Function Set** $F = \{f_1, f_2, \ldots, f_h\}$

$$f_i : \{0,1\}^l \mapsto \{1, 2, \ldots, m\}, in \in \mathbb{Z}_h$$

where l is the length of entity identity defined by the system. F is public.

4. **Algorithm of Combining Public and Private Key** Alg_{KG}
CPK directly uses entity identity $\text{ID}_{\mathbb{E}}$ as public key, the combined public or private key pair $(d_{\mathbb{E}}, Q_{\mathbb{E}})$ is:

$$d_{\mathbb{E}} = \left(\Sigma_{i=1,2,\ldots,h} d_{r_i i}\right) \bmod n, Q_{\mathbb{E}} = \Sigma_{i=1,2,\ldots,h} Q_{r_i i}, r_i = f_i(\text{ID}_{\mathbb{E}})$$

Alg_{KG} is public.

The CPK component mentioned above is the basic component defined by CPK1.0. In CPK1.0, one entity's private key $d_{\mathbb{E}}$ is a linear combination of elements in private key seed matrix. Because the mapping function set F is public, $\mathbb{E}$ can write a linear equation $d_{\mathbb{E}} = (d_{r_1 1} + d_{r_2 2} + \ldots + d_{r_h h}) \bmod n$ based on its identity. If there are $m \times h$ entities launching a conspiracy attack, then $m \times h$ linear equations can be wrote. If these $m \times h$ equations are linearly independent, then $m \times h$ unknown variants can be worked out, that is the all elements in private key seed matrix $(d_{ij})_{m\times h}$.

The conspiracy attack on CPK1.0 need not work out all seed private key. Suppose there are two entities $\mathbb{E}_1$ and $\mathbb{E}_2$, their private keys are $d_{\mathbb{E}_1}$ and $d_{\mathbb{E}_2}$, their public keys are $Q_{\mathbb{E}_1}$ and $Q_{\mathbb{E}_2}$ respectively. $d_{\mathbb{E}_1}$ conspires with $d_{\mathbb{E}_2}$, that is $d_{\mathbb{E}_1}$ is combined with $d_{\mathbb{E}_2}$ linearly: $d_{\mathrm{Atk}} = (a_{\mathrm{Atk}} d_{\mathbb{E}_1} + b_{\mathrm{Atk}} d_{\mathbb{E}_2}) \bmod n$, $Q_{\mathbb{E}_1}$ is combined with $Q_{\mathbb{E}_2}$ linearly: $Q_{\mathrm{Atk}} = a_{\mathrm{Atk}} Q_{\mathbb{E}_1} + b_{\mathrm{Atk}} Q_{\mathbb{E}_2}$. Try to select $\mathrm{ID}_{\mathrm{Try}} \in \{0,1\}^l$, and compute $Q_{\mathrm{Try}} = \Sigma_{i=1,2,\ldots,h} Q_{r_i i}, r_i = f_i(\mathrm{ID}_{\mathrm{Try}})$. If $Q_{\mathrm{Try}} = Q_{\mathrm{Atk}}$, then this is a successful conspiracy attack. $(d_{\mathrm{Atk}}, Q_{\mathrm{Atk}})$ is a valid key pair, and it can pretend to be a valid entity, but d_{Atk} is not generated by PKG.

In order to resist the conspiracy attack and keep the character of CPK, CPK5.0 [11] adds separation private key sequence $(Sd_i)_k$ and separation public key sequence $(SQ_i)_k$, $i \in \{1,2,\ldots,k\}$, k is the number of separation keys, $SQ_i = Sd_i G$. $(Sd_i)_k$ is stored in PKG as a secret. $(SQ_i)_k$ is public. Some functions also are added into the mapping function set: $F = \{f_1, f_2, \ldots, f_h\} + \{f_{S_1}, f_{S_2}, \ldots, f_{S_t}\}, t \in \mathbb{Z}, 1 < t < k$:

$$f_{Si} : \{0,1\}^l \mapsto \{1,2,\ldots,k\}, i \in \{1,2,\ldots,t\}$$

The combined private and public key in CPK5.0 are:

$$\begin{aligned}&\mathrm{CPriK}_{\mathbb{E}} = \left(d_{\mathbb{E}} + \Sigma_{i=c_1,c_2,\ldots,c_t} Sd_i\right) \bmod n\\ &\mathrm{CPubK}_{\mathbb{E}} = Q_{\mathbb{E}} + \Sigma_{i=c_1,c_2,\ldots,c_t} SQ_i, c_i = f_{S_i}(\mathrm{ID}_{\mathbb{E}})\end{aligned}$$

By using the separation key sequence, CPK5.0 multiplies the difficulty of conspiracy attack.

CPK also has the possibility of key collision. Document [6,7] have presented an optimized scheme of CPK seed matrix to avoid key collision.

4 HCPK-Based Trusted Computing Cryptography Scheme

4.1 Motivation for HCPK

In the CPK cryptosystemall private keys can be generated by PKG. If PKG is attacked, all private keys will be leaked out, PKG is at a high risk. CPK can form very large key space based on small scale of key seed matrix, so in theory the flatting management of CPK keys can be implemented, that is all the private keys can be generated by one PKG. But in practice, PKG should verify the identity of every entity, then generates and distributes private key, when PKG is in a large network application system, the single PKG might became a bottle neck. If there are too many entities, it is hard for PKG to distribute private key into each entity in a security controlled environment.

In order to disperse security risk and work load of single PKG, this paper presents a Hierarchical CPK architecture (HCPK) by referring HIBE [14]. In HCPK, every PKG of each layer has its own private key seed matrix. If some PKG at some level is attacked, its private key seed is leaked out, only the entities belonged to this PKG will be affected, the security of other entities are still guaranteed. Even root PKG is attacked, the entities' private keys won't be leaked out. PKG at every level only needs to verify the identities of the entities belong to the PKG and its next level PKG, generate and distribute the corresponding private keys. In HCPK, the work load of root PKG is distributed to other low level PKGs, the private key distribution can be done locally under a security environment.

4.2 Hierarchical Combined Public Key (HCPK)

To have a clear understanding, we describe HCPK based on CPK1.0, the construction of HCPK based on CPK5.0 can use the method similarly.

1. **Setup**
   (a) Given one elliptic curve E based on the selected finite field F_p: $y^2 \equiv (x^3 + ax + b)(\bmod\ p)$, G is the generator of one additive cyclic subgroup of points on E, n is the order of this group. ECC parameters are $\langle p, a, b, G, n\rangle$, which are public.
   (b) Building mapping function set: $F = \{f_1, f_2, \ldots, f_{h_{\max}}\}, f_i : \{0,1\}^l \mapsto \mathbb{Z}_m, i \in \mathbb{Z}_{h_{\max}}$, where $h_{\max}$ is the maximum number of columns in all PKG public and private key seed matrixes, $1 < h_{\max} < n$, l is the length of entity identity $\mathrm{ID}_{\mathbb{E}}$. F is public.
   (c) Choosing one HASH function: $H : \{0,1\}^* \mapsto \mathbb{Z}_p$. H is public.
   (d) PKG_k builds private key seed matrix. PKG_k represents the PKG at level k, $k \geqslant 0$, PKG_0 represents root PKG. There is only one root PKG in every HCPK system. PKG_k builds private key seed matrix $(d^k_{ij})_{m\times h_k}$, where $h_k \in \mathbb{Z}, 1 < h_k < h_{\max}, d^k_{ij}, d^k_{i'j'} \in \mathbb{Z}, 1 < d^k_{ij}, d^k_{i'j'} < n, i, i' \in \{1, 2, \ldots, m\}, j, j' \in \{1, 2, \ldots, h_k\}$. Only under the condition $i = i'$ and $j = j'$, $d^k_{ij} = d^k_{i'j'}$, otherwise $d^k_{ij} \neq d^k_{i'j'}$. $(d^k_{ij})_{m\times h_k}$ is just stored in PKG_k as a secret.
   (e) PKG_k builds public key seed matrix $(Q^k_{ij})_{m\times h_k}$, where $Q^k_{ij} = d^k_{ij}G, i \in \{1, 2, \ldots, m\}, j \in \{1, 2, \ldots, h_k\}$. $(Q^k_{ij})_{m\times h_k}$ is public.
2. **Extract**
   $\mathbb{E}_t$ represents a entity at level t, $t \geqslant 1$, $\mathbb{E}_t$ may be the PKG at level t. The identity tuple of $\mathbb{E}_t$ is: $(\mathrm{ID}_1, \mathrm{ID}_2, \ldots, \mathrm{ID}_t)$, where ID_1, ID_2, , ID_{t-1} are the identities of $\mathbb{E}_t$'s ancestor PKGs at level 1, 2, , $t-1$ respectively. ID_t is the identity of $\mathbb{E}_t$. $(\mathrm{ID}_1, \mathrm{ID}_2, \ldots, \mathrm{ID}_t)$ is public. The parent PKG of $\mathbb{E}_t$ PKG_{t-1} compute:

$$d_{\mathbb{E}_t} = \left(d_{t-1} + \Sigma_{i=1,2,\ldots,h_{t-1}} d^{t-1}_{r_i i}\right) \bmod n, r_i = f_i(\mathrm{ID}_t)$$

where d_{t-1} is the private key of PKG_{t-1}. If $t = 1$, then PKG_{t-1} is PKG_0 and $d_{t-1} = 0$. $d_{\mathbb{E}_t}$ is stored in $\mathbb{E}_t$ as a secret.

3. **Sign**

   Signature and verification schemes are based on the algorithms in document[15].

   Entity $\mathbb{E}_t$ signs message m with the private key $d_{\mathbb{E}_t}$:

   (a) Compute $h = H(m)$;
   (b) Choose a random $r \in [1, n-1]$;
   (c) Compute $(x_r, y_r) = rG$;
   (d) Compute $u = (h + x_r) \bmod n$, if $u = 0$ or $u + r = 0$, then goto (b);
   (e) Compute $v = \big((1 + d_{\mathbb{E}_t})^{-1} \cdot (r - u \cdot d_{\mathbb{E}_t})\big) \bmod n$, if $v = 0$ then goto (b);
   (f) The signature is $\sigma = (u, v)$.

4. **Verify**

   Verify the signature $\sigma = (u, v)$ of m signed by $\mathbb{E}_t$, the identity tuple of $\mathbb{E}_t$ is $(\mathrm{ID}_1, \mathrm{ID}_2, \ldots, \mathrm{ID}_t)$:

   (a) Compute $Q_{\mathbb{E}_t} = \Sigma_{k=1,2,\ldots,t}\big(\Sigma_{i=1,2,\ldots,h_{k-1}} Q^{k-1}_{r^k_i i}\big), r^k_i = f_i(\mathrm{ID}_k)$;
   (b) Compute $h = H(m)$;
   (c) Compute $t = (u + v) \bmod n$;
   (d) Compute $(x_r, y_r) = vG + tQ_{\mathbb{E}_t}$;
   (e) Compute $u' = (h + x_r) \bmod n$;
   (f) If $u = u'$, then the signature is right.

4.3 Application of HCPK in Trusted Computing

According to TCM production, evaluation and application, a four levels HCPK can be used in trusted computing, as is shown in Figure 3. Under the root PKG authenticated by China Cryptography Administration, every TCM manufacturer and enterprise user build their own PKGs, and individual users can use manufacturer PKG directly.

The scale of public and private key seed matrix in root PKG, manufacturer PKG and enterprise PKG is determined by the amount of their subaltern PKGs and TCMs respectively. Root PKG only needs to generate and distribute private keys for manufacturer PKGs, the security of private keys distribution can be

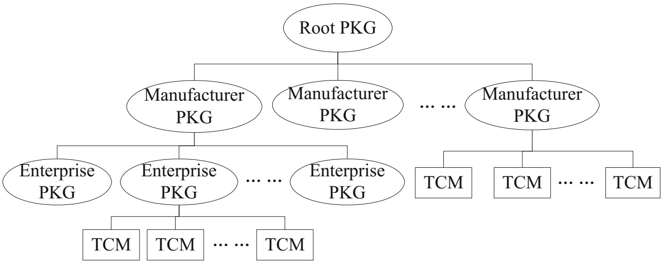

Fig. 3. Trusted Computing HCPK

guaranteed more easily, and the work load also are reduced. Similarly, manufacturer PKG generates and distributes private keys for enterprise PKGs or TCMs, enterprise PKG generates and distributes private keys for TCMs, which also can be done in a local controlled security environment. During identity verification of TCM host platform, identity tuple $(\mathrm{ID}_{\mathrm{Man}}, \mathrm{ID}_{\mathrm{Ent}}, \mathrm{ID}_{\mathrm{TCM}})$ or $(\mathrm{ID}_{\mathrm{Man}}, \mathrm{ID}_{\mathrm{TCM}})$ should be used, where $\mathrm{ID}_{\mathrm{Man}}$, $\mathrm{ID}_{\mathrm{Ent}}$, $\mathrm{ID}_{\mathrm{TCM}}$ are the identity of manufacturer PKG, enterprise PKG and TCM respectively.

If one enterprise PKG is attacked and its private key seed matrix is leaked out, only the TCMs belonged to the enterprise are affected, since other manufacturer PKGs have their own private key seed matrixes. Even if the root PKG has been attacked, it will not completely expose all TCMs' private keys, because manufacturer PKGs, enterprise PKGs have their own private key seed matrixes.

During platform identification, the TCM on Access Request Platform (ARP) uses its own PIK private key to sign given PCR values. The verifier receives the signature, according to identification tuple $(\mathrm{ID}_{\mathrm{Man1}}, \mathrm{ID}_{\mathrm{Ent1}}, \mathrm{ID}_{\mathrm{TCM}})$, it is able to compute the corresponding ECC public key of PIK private key, and complete identity verification consequently.

4.4 TCM Key Architecture Based on HCPK

Within the architecture of HCPK, PIK private key is directly generated by manufacturer PKG or enterprise PKG based on TCM identity $\mathrm{ID}_{\mathrm{TCM}}$. Platform users need not apply for PIK public key certificate based on EK. There is no need for a privacy CA to maintain the database of PIK certificate, because the identity tuple $(\mathrm{ID}_{\mathrm{Man}}, \mathrm{ID}_{\mathrm{Ent}}, \mathrm{ID}_{\mathrm{TCM}})$ or $(\mathrm{ID}_{\mathrm{Man}}, \mathrm{ID}_{\mathrm{TCM}})$ just is the PIK public key.

PKI-based platform can have multiple PIKs and the corresponding certificates, in order to protect the privacy of platform identity. To ensure the security of private key distribution and using, CPK-based PIK private key are usually directly loaded into TCM by PKG in one controlled and security environment. With the limited memory capacity, TCM fails to save multiple PIK private keys. It also is unable to generate PIK dynamically for CPK-based platform, because the security of CPK private key online distribution can not be guaranteed easily. So there is only one PIK on CPK-based platform. Compared with PKI-based platform, CPK-based platform has a more simplified TCM key architecture, which is shown in Figure 4. The anonymity of platform identity can be ensured by TCM ring signature.

In order to authenticate TCP directly without third party, there also are CPK parameters including public key seed matrix stored in TCM.

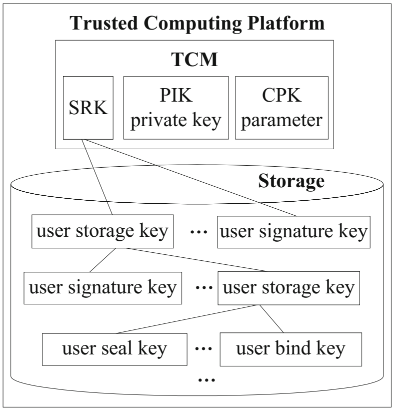

Fig. 4. HCPK-based TCM key architecture

4.5 Cross-Domain Platform Identity Authentication Based on HCPK

CPK usually loads CPK parameters into all entities in one security domain, for authenticate TCP directly without third party and online database. Because PKG is at high security risk, different institutes or organizations will establish their own PKGs. However in some application scenarios, ARP and the verifier may belong to different security domains, which needs cross-domain remote authentication and resource access. Since each security domain has its own PKG, different PKG usually has different CPK parameters, so it is unable to do cross-domain authentication directly.

Within the architecture of HCPK, considering the most typical situation of cross-domain attestation, ARP and verifier belong to different enterprise PKGs, and this two enterprise PKGs belong to different manufacturer PKGs. Even ARP and verifier belong to different security domains, but they belong to a same root PKG, sharing same CPK parameters, which is shown in Figure 5, it is able to do cross-domain attestation directly.

5 Security Analysis

Security analysis of CPK resistance against conspiracy attack and private key collision caused by mapping function are out of the scope of this paper, we just analyse the security of HCPK-based trusted computing cryptography scheme, that is the security of platform identity authentication with PIK private key based on HCPK. This paper also does not care the security problems caused by specific implementations, and just analyses the security of HCPK signature scheme based on Computation Diffie-Hellman Problem (CDHP) in random oracle model.

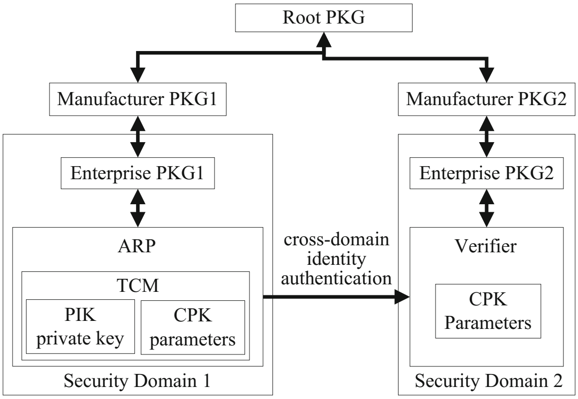

Fig. 5. Cross-domain platform identity authentication based on HCPK

Computation Diffie-Hellman Problem (CDHP): There is an additive cyclic sub-group of points on one elliptic curve in a finite field F_P, G is the generator of the group, n is the order of the group. Given $(G, aG, bG), a, b \in \mathbb{Z}_n$, compute abG.

5.1 Attack Model and Security Definition

The most general known notion of security of an ID-based signature scheme is Existential Forgery on Adaptively Chosen Message and ID Attacks (EF-ACM-IA) presented in document [16]. CPK is a ID-based cryptography scheme, the model of EF-ACM-IA on HCPK is (the adversary algorithm is denoted as $\mathcal{A}$, the challenger playing the following game against $\mathcal{A}$ is denoted as $\mathcal{C}$):

1. $\mathcal{C}$ runs **Setup** of HCPK system, and gives returned system parameters to $\mathcal{A}$;
2. $\mathcal{A}$ issues the following queries as he wants:
   (a) Mapping function query. $\mathcal{A}$ gives an identity name ID, $\mathcal{C}$ computes $f_i(\text{ID}), i = 1, 2, \ldots,$
   $h_{\max}$, and returns the result to $\mathcal{A}$;
   (b) HASH function query. $\mathcal{A}$ gives a message m, $\mathcal{C}$ computes $H(m)$, and returns the result to $\mathcal{A}$;
   (c) **Extract** query. $\mathcal{A}$ gives an identity tuple $(\text{ID}_1, \text{ID}_2, \ldots, \text{ID}_t)$, $\mathcal{C}$ runs **Extract**, and returns the result, a private key d_t, to $\mathcal{A}$;
   (d) **Sign** query. $\mathcal{A}$ gives a private key d_t and a message m, $\mathcal{C}$ runs **Sign**, and returns the result, a signature σ , to $\mathcal{A}$;
3. $\mathcal{A}$ outputs $\big((\text{ID}_1, \text{ID}_2, \ldots, \text{ID}_t), m, \sigma\big)$, where $(\text{ID}_1, \text{ID}_2, \ldots, \text{ID}_t)$ and m are not equal to the inputs of any query to **Extract** and **Sign**, respectively. $\mathcal{A}$ wins the game if σ is a valid signature of m for $(\text{ID}_1, \text{ID}_2, \ldots, \text{ID}_t)$.

Definition 1. *If no polynomial time algorithm $\mathcal{A}$ has non-negligible probability advantage of winning above game, then HCPK signature scheme is secure under EF-ACM-IA.*

5.2 Security Proof

First we modify the above game of EF-ACM-IA on HCPK as Existential Forgery on Adaptively Chosen Message and Given ID Attacks (EF-ACM-GIA): Given an identity tuple $(\mathrm{ID}_1, \mathrm{ID}_2, \ldots, \mathrm{ID}_t)$ in step (1), $\mathcal{C}$ returns system parameters to $\mathcal{A}$ together with $(\mathrm{ID}_1, \mathrm{ID}_2, \ldots, \mathrm{ID}_t)$. In step (3), $\mathcal{A}$ must output the given $(\mathrm{ID}_1, \mathrm{ID}_2, \ldots, \mathrm{ID}_t)$ together with corresponding message m and signature σ as its final result.

Referring Lemma 1 in document [16], the following Lemma 1 can be obtained:

Lemma 1. *If there is an algorithm $\mathcal{A}_0$ which can win the game of EF-ACM-IA to HCPK signature scheme with polynomial running time t_0 and probability advantage ϵ_0, then there is an algorithm $\mathcal{A}_1$ which can win the game of EF-ACM-GIA with polynomial running time $t_1 \leqslant t_0$ and probability advantage $\epsilon_1 \geqslant \epsilon_0 \cdot (1 - 1/n)/q_{_F}$, where $q_{_F}$ is the maximum number of queries to mapping function asked by $\mathcal{A}_0$. The numbers of queries to mapping function, HASH function,* ***Extract*** *and* ***Sign*** *asked by $\mathcal{A}_1$ are the same as those of $\mathcal{A}_0$.*

Lemma 2. *If there is an algorithm $\mathcal{A}_1$ which can win the game of EF-ACM-GIA with polynomial running time t_1 and probability advantage $\epsilon_1 \geqslant 10(q_{_S} + 1)(q_{_S} + q_{_H})/n$, then CDHP can be solved by an algorithm $\mathcal{A}_2$ with polynomial running time $t_2 \leqslant 23q_{_H}t_1/\epsilon_1$ and probability advantage $\epsilon_2 \geqslant 1/9$, where $q_{_H}$, $q_{_S}$ are the maximum number of queries to HASH function and* ***Sign*** *asked by $\mathcal{A}_1$ respectively.*

Proof (Lemma 2).

The algorithm $\mathcal{A}_1$ can be viewed as an adversary with adaptively chosen message attack to the non-ID-based scheme obtained by fixing an ID in HCPK-based signature scheme. So we can refer correlative lemma or theorem in document [17].

> **Lemma 4 in document [17].** Let $\mathcal{A}$ be a Probabilistic Polynomial Time (PPT) Turing machine whose input only consists of public data. The number of queries that $\mathcal{A}$ can ask to the random oracle and the number of queries that $\mathcal{A}$ can ask to the signer are denoted as $q_{_R}$ and $q_{_S}$ respectively. Assume that, within a time bound t, $\mathcal{A}$ produce, with probability $\epsilon \geqslant 10(q_{_S}+1)(q_{_S}+q_{_R})/n$, a valid signature (m, r, h, σ). If the triples (r, h, σ) can be simulated without knowing the secret key, with an indistinguishable distribution probability, then, a replay of the attacker $\mathcal{A}$, where interactions with the singer are simulated, outputs two valid signatures (m, r, h, σ) and (m, r, h', σ'), such that $h \neq h'$, within time $t' \leqslant 23q_{_R}t/\epsilon$ and with probability $\epsilon' \geqslant 1/9$.

The signature value of HCPK signature scheme $\sigma = (u, v)$, so the two valid signatures are $\big(m, r, h, (u, v)\big)$ and $\big(m, r, h', (u', v')\big)$.

Since $u = (h + x_r) \bmod n$ and $v = \big((1 + d_{\mathbb{E}_t})^{-1} \cdot (r - u \cdot d_{\mathbb{E}_t})\big) \bmod n$, so

$$u - h \equiv x_r (\mathrm{mod}\ n), vG + (u + v)Q_{\mathbb{E}_t} = rG$$

Similarly, $u' - h' \equiv x_r (\bmod\ n), v'G + (u' + v')Q_{\mathbb{E}_t} = rG$, then

$$\begin{cases} u - h \equiv u' - h' (\text{mod n}) \Leftrightarrow (u - h)Q_{\mathbb{E}_t} = (u' - h')Q_{\mathbb{E}_t} & (1) \\ vG + (u + v)Q_{\mathbb{E}_t} = v'G + (u' + v')Q_{\mathbb{E}_t} & (2) \end{cases}$$

Compute (2) − (1): $(v - v' + h - h') \cdot d_{\mathbb{E}_t} G = (v' - v)G$

Suppose $a = v - v' + h - h', b = d_{\mathbb{E}_t}$, then $abG = (v' - v)G$. That is we have known G, $aG = (v - v' + h - h')G$ and $bG = Q_{\mathbb{E}_t}$, we can compute abG, and CDHP can be solved, so Lemma 2 has been proved.

Combining Lemma 1 and 2, we can get Theorem 2.

Theorem 2. *If there is an algorithm $\mathcal{A}_0$ which can win the game of EF-ACM-IA to HCPK signature scheme with polynomial running time t_0 and probability advantage $\epsilon_0 \geqslant 10(q_{_S} + 1)(q_{_S} + q_{_H})q_{_F})/(n - 1)$, then CDHP can be solved by an algorithm $\mathcal{A}_2$ with polynomial running time $t_2 \leqslant 23q_{_H}q_{_F}t_0/(\epsilon_0(1 - 1/n))$ and probability advantage $\epsilon_2 \geqslant 1/9$, where $q_{_F}$, $q_{_H}$, $q_{_S}$ are the maximum number of queries to mapping function, HASH function and **Sign** asked by $\mathcal{A}_0$ respectively.*

Because there is no probabilistic polynomial time algorithm which can solve CDHP up to now, there is no algorithm $\mathcal{A}_0$ which can win the game of EF-ACM-IA to HCPK signature scheme with polynomial running time and non-negligible probability advantage, and HCPK signature scheme satisfies the requirements in Definition 1.

6 Performance Analysis

This paper analyses the performance of HCPK-based TCM cryptography scheme compared with PKI-based scheme. Because of limited resources, TCM has high performance requirement of cryptography scheme. This paper mainly analyses the calculation performance on TCM of the two cryptography schemes, and does not analyse the performance of PKI-based TPM cryptography scheme, because TPM uses RSA algorithm, TCM uses ECC algorithm, it is uneasy about comparing the performance of RSA with ECC directly.

The differences between HCPK and PKI-based cryptography scheme mainly are in PIK generation and signature signing. Because signature verification can be done on the host, this paper does not analyze the performance difference in PIK signature verification.

HCPK-based PIK private key is generated by parent PKG of TCM, and is distributed and loaded into TCM directly, there is no need for TCM to do any calculation.

PKI-based PIK creation includes generating of PIK, applying for, signing and activating PIK certificate. PIK is generated by TCM, the time spent is denoted as T_{KeyGen}. TCM uses PIK private key to sign the digest of privacy CA public key and PIK public key, the time spent is denoted as $T_{\text{Hash}} + T_{\text{Sign}}$. TCM also needs to use EK private key for decrypting to get the session key for encrypting

PKI certificate, the time spent is denoted as T_{SDec}. PKI-based TCM may crate multiple PIKs, so it will multiply the time spent.

The time spent by HCPK-based TCM for signing PCR value with PIK private key is the same as PKI-based TCM, denoted as T_{Sign}. For using PIK in PKI-based TCM, PIK must be loaded into TCM by using command TCM_LoadKey firstly, the time spent is denoted as T_{LdKey}.

The comparison of TCM performance between HCPK-based cryptography scheme and PKI-based scheme is shown in Table 1.

Table 1. Comparison of TCM perforamnce between HCPK-based scheme and PKI-based scheme

	time spent on PIK creation	**time spent on PIK signing**
HCPK-based scheme	0	T_{Sign}
PKI-based scheme	$x(T_{\text{KeyGen}} + T_{\text{Hash}} + T_{\text{Sign}} + T_{\text{SDec}})$	$T_{\text{LdKey}} + T_{\text{Sign}}$

We can see from Table 1 that HCPK-based cryptography scheme has a great advantage of TCM performance compared with PKI-based scheme.

In addition, some TCM commands about EK and PIK can be reduced in HCPK-based TCM, such as TCM_CreateEndorsementKeyPair, TCM_CreateRevocableEK, TCM_RevokeTrust, TCM_ReadPubEK, TCM_MakeIdentity, TCM_ActivateIdentity and etc, which also can save TCM storage space, simplify TCM implementation, and improve the performance of TCM.

7 Conclusion and Future Work

PKI-based TCP requires platform users to apply for multi PIK certificates, the annual fee of one certificate is about 8 $, so users of PKI-based TCP must pay $8x$ $ one year. HCPK-based TCP can reduce the risk of single PKG, and let the verifier authenticate TCP directly without third party, platform users do not need pay any fee for applying additional digital certificates. HCPK-based trusted computing cryptography scheme also can be implemented without any modification of current TCM hardware. So HCPK-based TCP can reduce users' cost of using TCP. HCPK-based trusted computing cryptography scheme can simplify TCM key architecture, and authenticate cross-domain platform identity. This paper has proved that HCPK signature scheme is secure under EF-ACM-IA in random oracle model. Comparing with PKI-based scheme, HCPK-based cryptography scheme has obvious advantage in the performance of TCM. HCPK and PKI-based platforms can authenticate each other, and HCPK-based cryptography scheme also can be implemented on TPM.Next.

In future, we will consider how to protect the anonymity of PIK private key and platform identity based on HCPK, and design a platform remote attestation protocol based HCPK, which can directly authenticate cross-domain platform without third party, and protect the privacy of platform component information.

References

1. Shen, C., Zhang, H., Wang, H., et al.: Research and development of trusted computing. Science China: Information Science 40(2), 139–166 (2010) (in chinese)
2. Shen, C., Zhang, H., Feng, D., et al.: Survey of information security. Science China: Information Science 37(2), 1–22 (2007) (in chinese)
3. Nan, X., Chen, Z.: A profile to network security techniques. National Defense Industry Press, Beijing (2003) (in chinese)
4. Chen, H., Guan, Z.: Explanation of some questions about CPK. China Information Security 9, 47–49 (2007) (in chinese)
5. Wang, G., Wang, M., Wu, D., et al.: Analysis of the CPK random collision probability. China Information Security 11, 87–88 (2008) (in chinese)
6. Rong, K., Li, Y.: A optimized scheme of the CPK seed matrix. Journal of Computer Engineering and Applications 42(24), 120–121 (2006) (in chinese)
7. Xing, H.: Research and applications of the key technologies of combined public key. Engineering master dissertation of National University of Defense Technology (2009) (in Chinese)
8. Nan, X.: Identity authentication based on CPK. National Defense Industry Press, Beijing (2006) (in Chinese)
9. Nan, X.: CPK-crypotosystem and cyber security. National Defense Industry Press, Beiing (2008) (in Chinese)
10. Nan, X.: Cyber security technical framework — Trusting system based on identity authentication. Electronic Industry Press, Beijing (2010)
11. Nan, X.: Combined Public Key (CPK) Cryptosystem Standard (v5.0). Network & computer security (2010) (in Chinese)
12. Shamir, A.: Identity-based cryptosystems and signature schemes. In: Blakely, G.R., Chaum, D. (eds.) CRYPTO 1984. LNCS, vol. 196, pp. 47–53. Springer, Heidelberg (1985)
13. Boneh, D., Franklin, M.: Identity-based encryption from the weil pairing. In: Kilian, J. (ed.) CRYPTO 2001. LNCS, vol. 2139, pp. 213–229. Springer, Heidelberg (2001)
14. Gentry, C., Silverberg, A.: Hierarchical ID-based cryptography. In: Zheng, Y. (ed.) ASIACRYPT 2002. LNCS, vol. 2501, pp. 548–566. Springer, Heidelberg (2002)
15. China Cryptography Administration. State Public Key Cryptographic Algorithm SM2 Based on Elliptic Curves (December 2010) (in Chinese), `http://www.oscca.gov.cn/UpFile/2010122214822692.pdf` (March 2011)
16. Cha, J.C., Cheon, J.H.: An identity-based signature from gap diffie-hellman groups. In: Desmedt, Y.G. (ed.) PKC 2003. LNCS, vol. 2567, pp. 18–30. Springer, Heidelberg (2002)
17. Pointcheval, D., Stern, J.: Security Arguments for Digital Signatures and Blind Signatures. Journal of Cryptology 13(3), 361–396 (2000)