Virtualizing Smart Cards: a Step Toward Ubiquitous Cryptography

Alfonso De Gregorio, Thomas Fossati, Giovanni Frustaci

Abstract— **In this paper we analyze and extend the concepts of holding, management and ownership of credentials in traditional smart cards system from a distributed point of view, where, in fact, distinct entities may play the roles of Data Owner, Credential Holder and Terminal Owner.**
We show that is possible to build in a fairly straight way a virtual smart card system, which retains functional equivalence towards a traditional smart card system and at the same time brings several improvements from a security, versatility, interoperability and deployment point of view.
In order to achieve this functional equivalence, it is necessary to permit a secure interaction between legitimate users and their remote credential holding systems via untrusted terminals and over untrusted networks. We provide a cryptographic protocol which has been carefully build to make the Terminal Owner unable to abuse the Data Owner's credentials.

Keywords— **Key-management, trust splits, virtual smart-cards, untrusted proxies, ubiquitous computing, authentication protocols, one-time passwords.**

I. Introduction

We start from a simple observation concerning a traditional key storage smart card: in it, the roles of Data Owner (i.e., the party that has the control over the credential that has been associated with her), Credential Holder (i.e., the party that really possesses and manages the credentials) and Terminal Owner (i.e., the entity that has control over the device through which the data owner can interact with the credential holding system) must be played by the same entity.
This peculiarity implies some advantages (a definite simplification of active relationships) but also a certain rigidity that has not - together with other factors, such as the many interoperability issues, vulnerabilities and costs - helped in the adoption of smart cards as the natural and ubiquitous means of interaction between users and their surrounding Public Key Infrastructure (PKI).

C&A S.r.l., V.le Fulvio Testi, 126
20092 – Cinisello Balsamo (MI), Italy.
E-Mail: {adg,tho,giovanni}@com-and.com

We, instead, will try to analyze a wider scenario in which the aforementioned roles are generically assigned to distinct entities, confident that a greater flexibility permits, once the newly surfacing threats are countered, to retain functional equivalence towards traditional smart cards system and at the same time to reach a larger spectrum of applications.
In the treatment of this subject we will make use of the experience that we have gained in the past two years while engineering a family of products that actually implement these concepts.

Our paper is organized into three main parts:

a) We first consider the implications hidden in the possible separation of the Credential Holder and Data Owner entities: the issues - related especially to trust propagation - and also some clear advantages achieved by virtualizing the ownership of users' credentials (versatility, wide deployment and better interoperability of PKI dependent applications) (Section III).

b) Given the assignment of different roles to different entities, we introduce a convenient Trust/Threat model that takes into consideration the new interactions and the new feasible attacks to/from the playing parties (Section IV and V.B)

c) We then look at the adequate countermeasures needed to confine the most critical attacks that the Terminal Owner can carry against the Data Owner. Within this analysis, we provide a challenge/response cryptographic protocol which has been carefully built to make the Terminal Owner unable to abuse the Data Owner's credentials (Section V). We analyze it with the BAN logic of authentication [28, 27].
It should be noted that our goal is to propose an effective countermeasure based upon standardized mechanism and not to innovate the underlying cryptographic methods.

II. Related Work

In the following two subsections we review respectively the related work about cryptographic mobility solutions and authentication in presence of untrusted terminals.

A. Cryptographic Mobility Solutions

Gupta makes in [1] an analysis of security characteristics of some representative cryptographic mobility solutions. The proposed approaches to cryptographic mobility are basically four. The first enables the data owner to retrieve a temporary copy of the credentials that he wishes to use locally. The second typology includes solutions that enables data owners to make use of their credentials by instructing the required operations to their own remote credential holding systems (this kind of systems include also network-enabled smart cards systems [2,3,4]). Perrin et al. propose in [20] a third typology of solutions that enable a generic user to request a TTP the execution of specific encryption tasks through credentials associated to the TTP itself. Solutions belonging to the fourth typology, instead, let a user to delegate to a TTP only the management of his own credentials and to instruct their use remotely [5]. The TTP will take care of implementing the confidentiality and integrity of the managed credentials, enabling the use of these credentials only to their legitimate owners, and also publishing possible compromises. The last part of the paper will take into consideration the trustworthiness and security aspects related to either the second or the fourth typologies.

B. Authentication in Presence of Untrusted Terminals

Many authentication mechanisms require the user to place total trust in terminals used to send credentials to the authentication servers. To address the attacks that an unfaithful terminal may perform, many works [23, 24, 25, 26] propose to replace authentication protocols subjected to replay attacks with challenge-response mechanisms, in which the user use a secret she knows to silently answer the challenge that she is given. However, in the application domain we are referring to, the data owner needs an external trusted device to authenticate the data he is sending to the credential holder to be processed. We take advantage of the presence of this device to build an authentication system with strong non-repudiation of origin guarantees. In [27] a smart-card based system is described that allows identification of the user in presence of untrusted terminals. The mechanism we propose enables the data owner to successfully use their credentials stored on remote system through untrusted networks and untrusted terminals, without requiring public terminals to support additional hardware (such as smart card readers or applications). Therefore facilitating the deployment and integrability of the proposed system.

III. VIRTUAL SMARTCARDS VS. SMARTCARDS

Virtualized tokens are not commonly considered a real alternative to smart cards. They are regarded instead as transitory systems, that may be used until smart cards and their respective readers become ubiquitous. We believe that a convincing solution to architectural problems and a careful system engineering can make the remote approaches to credentials management more attractive compared to the traditional smart card solutions from many standpoints. In the rest of this section the major characteristic of either virtual or traditional smart cards are identified and contrasted.

A. Security

A.1 Security Threats Evolution

Traditional tokens are unavoidably subjected to the risks coming from the heterogeneous working environments. The evolution of cryptanalytic techniques, for example, adds extemporaneously, in each environment, the possibility to bring new attacks (e.g., such as those based on side channels [12,13]). A virtual smart-card can be hosted in a trusted location, characterized by a unique environment that can be more quickly adapted to the evolution of security threats.

A.2 Resource Limits and Implications in Feasible Countermeasures

If new attacks show the ineffectiveness of countermeasures adopted in the past, the implementation of new ones will be doubly difficult. The current computation and storage limits of some tokens could make impossible to carry out some countermeasures. The absence of a synchronous internal clock, for example, can preclude the implementation of some time-based authentication scheme, certificate path validation and digital signatures verification algorithms, or accounting mechanisms.

Secondly, upgrading every token affected by a new vulnerability is a formidable and costly task.

Virtual smart-cards, while used to managed the credentials in a multi-user environment, can make the evolution of credential management technologies easier. Inevitably, a centralized approach may create also a single point of failure for an entire set of users and become a high-valuable target for attacks.

Virtual smart-cards should be definitively engineered to reduce the risks of a general security fault (e.g., using split key systems, and threshold and proactive signature schemes [29, 30, 31], while managing asymmetric credentials).

A.3 Competent Responses to Security Threats

No system is secure against all attacks. Traditional credential holding systems and virtualized tokens makes no exceptions. By delegating the storing and management of credential to a professional credential holder, the user can be relieved from keeping herself abreast of the evolution of security threats that impacts on credential holding systems. The typical user should not be required to realize, for example, if a new side channel attack could compromise her credentials. A trusted and professional credential holder can place all his competence in responding to security threats, as well as systems and infrastructures, at the data owner disposal. If a new relevant vulnerability get found, the TTP should inform the data owner and may request the revocation of either all the possible certificates relative to compromised credentials, or the trust associated to his identity. The possibility to delegate the credentials management is made possible by the delocalization of legitimate users from their credential holding systems.

*B. V*ersatility

The high level of abstraction offered by virtual smart-cards allow to achieve an higher grade of versatility in every model and implementation aspects. By evolving a single centralized credential holding system towards new technologies, the whole set of users would benefit of introduced features. New security countermeasures, characteristics of accessed PKIs (e.g., revised data structures and algorithms) can be made available to the users without upgrading every single smart-card.

C. Interoperability

To enable ubiquitous utilization of credential stored in traditional smart cards, it's necessary the presence of interoperable smart card enabled applications. However PKCS #11 [8] incompatible implementations make the development effort of the applications more demanding.
The solution to this problem is a centralized application (Credential Holding System) that permits to an authorized user (Data Owner) to perform all the operation through a secure communication protocol.
In Section V we propose a cryptographic protocol based on a two factor authentication scheme, that is functionally equivalent to the smart-card one.
In our scheme the Data Owner corroborate her own willingness by sending to the Credential Holding System a one-time authentication data computed throw an hand-held passcode generator. Yet, the implementation of our solution does not require public terminals to support neither additional hardware (such as smart card readers) nor specialized API. Therefore facilitating the deployment and integrability of the solution that we propose.

D. Integrability

The standardization of authenticated application protocols used by remote credential holding systems will enable to integrate easily these technologies in wider systems, releasing the latter systems from the burdens of implementing a cryptographic engine. Virtual smart cards, in fact, can be used not only in ubiquitous and pervasive context, but also in private and enterprise ones, where they can acts as cryptographic engines (e.g., for TSAs or CAs).

E. Key-Recovery

The risk of key-recovery exist either in traditional tokens, or in virtual smart-cards. Manufacturers may be asked by government agencies to get the user's secrets in an unnoticeable fashion. Manufacturers can implement key-recovery designing appropriately the access control system or using cryptographic expedients (e.g., kleptography [11, 32] enable a manufacturer not only to recover the user's secret in an unnoticeable fashion, yet protects against attacks by others and against reverse engineering).
To make sure that no key-recovery has been implemented for the credential holding systems in use, both smart-cards and virtualized tokens should be certified by the certification bodies in accordance of their schemes. Obviously, designing and building these technologies could require different levels of difficulty and expertise.

F. Efficiency

With remote credential holding systems, the certificate revocation process can benefit from the improved timeliness in the production of the certificate revocation request.

IV. TRUST MODEL

In this section we analyze the trust model for remote credential management systems. By examining the dependency and trust relationships between the system parties, we try to highlight problems which have not been taken into consideration until now.

A. Model Trust Environment of a Virtual Smart Card

Roles in virtual smart card systems are the same as those found in traditional smart cards [6]. Yet, de-locating the credential management system from its user, together with the need to maximize access opportunities for the service, imposes that the data owner, card holder and terminal owner roles are not necessarily played by the same actor.

Before we proceed with the analysis of the trust model we need to outline each role's profile.

- The **credential holder** is the party that really possesses and manages the credentials. The entity who plays as credential holder may be the same one that plays as data owner. Other parties can delegate her the custody and management of their credentials. She will accomplish her responsibilities in the way she consider the most suitable (e.g., placing the management system in an appropriate physical location) but never forgetting the need for availability of the services he furnishes. She must guarantee the integrity and confidentiality of the managed credentials. Furthermore, if managing a multi-user or multi-credential environment she must guarantee the correct binding between the delegating party and the intended credential. Finally he must publish every compromise of the credential management system to the intended parties. This role is similar to the card holder in traditional smart card based systems.
- The **data owner** is the party that has the control over the credential that has been associated to him. She is responsible for the use of his private credential. In a X.509 based PKI [15] the data owner could be associated with the certificate subject (i.e., the entity associated with the Subject Public Key field in a PKC or with the entityName in the Holder field of an AC). In fully virtualized smart cards, such as with traditional tokens, the data owner and the card holder are played by same actor.
- The **terminal owner** is the entity that has control over the device through which the data owner can interact with the credential holding system. She is responsible for the behavior of the device. In the distributed scenario we are considering this entity is usually distinct from the data owner and the credential holder.
- **card issuer**, **card manufacturer** and **software manufacturer** are the same as in traditional smart card based credential storage systems, so that no new consideration is needed.

B. Dependencies

In Figure 2 and Figure 3 we depict the strategic dependencies respectively for traditional and virtual smart card storage systems (typographical conventions in Figure 1).
We concentrate our analysis on the credential holder, data owner and terminal owner roles since they are affected by the separation of trust driven by the de-localization of the participants.

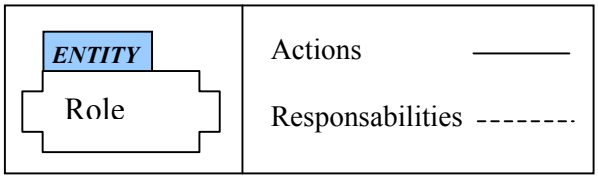

Figure 1: Legend

B.1 Dependencies in traditional smart cards

In traditional systems the same user plays three different roles: card holder, data owner and terminal owner, as a consequence social relationships are rarefied and many attacks are avoided.

B.2 Dependencies in virtual smart cards

Ist Trust Separation: data owner / credential holder
In centralized environments (e.g. mobile scenario, enterprise), where virtual tokens can actually be used, the remote user plays as data owner and optionally as credential holder. The user, in fact, can decide to store her credentials in a secure system accessible exclusively by her. In this case the data owner and credential holder roles are coalesced. But there can be also a situation where the credential holder role is played by a trusted third party, delegated by the data owner to manage his credentials. In that case the credential holder becomes an active party in the PKI with respect to the trust propagation and in some PKI models it could be necessary to make the interaction between the data owner and the credential holder explicit to the other (relying) parties, depending on the liability model in use.
We will discuss and try to formalize the introduction of such TTPs in a PKI in another paper [16].

IInd Trust Separation: data owner / terminal owner
Before they can interact through untrusted channels, the data owner and credential holder need to authenticate each other and build a secure channel for the subsequent communication. As you can see in Figure 3 the data owner and credential holder both rely on the mutual authentication mechanism which has a central role in the security, ease of deployment and versatility of the system.
The choice of a suitable authentication mechanism is a critical point with respect to the overall architecture since it heavily interacts with the trustworthiness assumptions we can do onto the terminal devices.
In fact the terminal holds a position that makes it suitable for a variety of attacks on data and credentials (e.g., reply, modification, stealing) that the terminal owner could be interested in exploiting for its own profit.

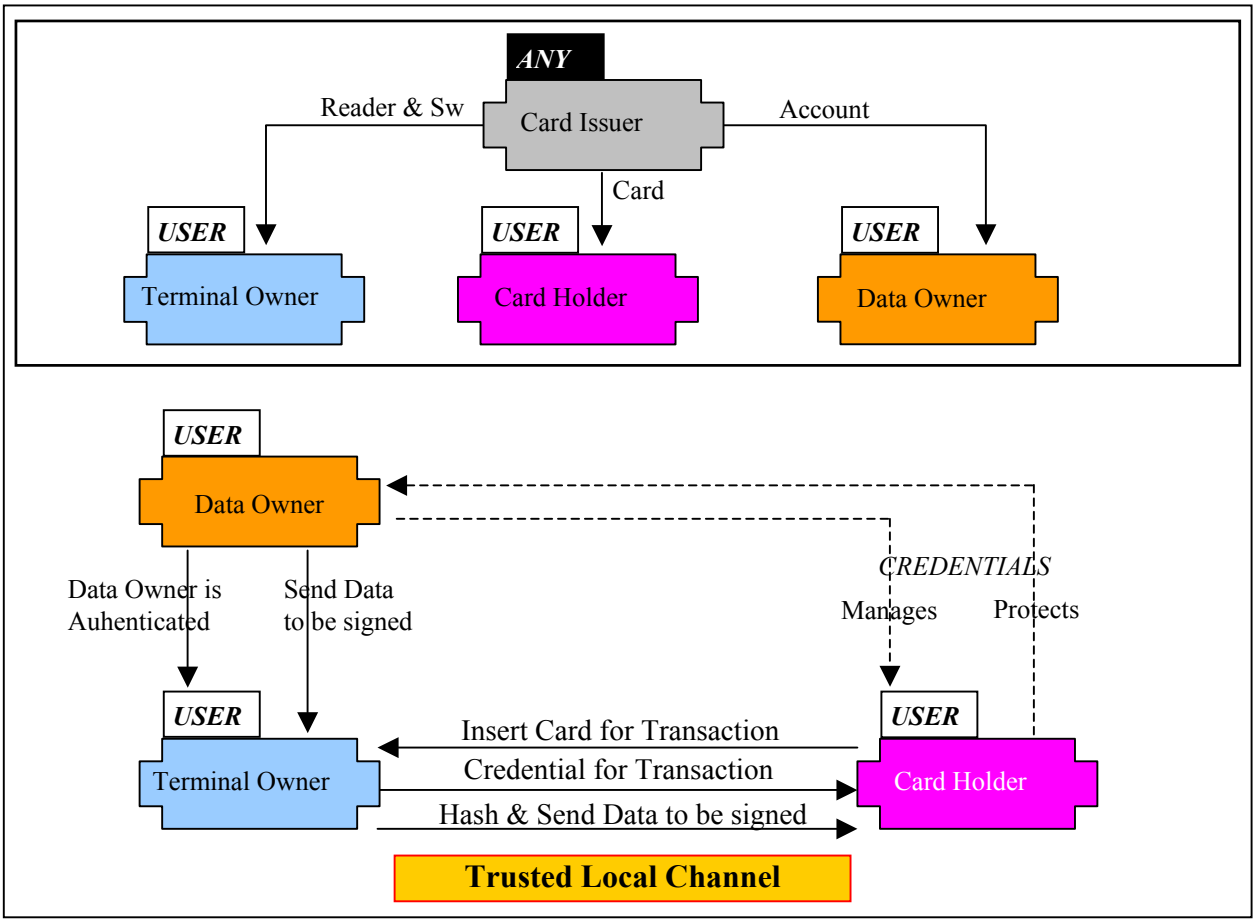

Figure 2: Traditional Smart Card System

Hence, in choosing the authentication mechanism we must take all the possible countermeasures needed to neutralize the most critical classes of attack that the terminal owner can direct to both the data owner and the credential holder.

V. Distrusting Terminals

In the following section, after outlining the context, we will analyze the threat model characterized by unfaithful terminal owners that may act as attackers against data owners. We will propose a countermeasure to the most critical of these attacks. The countermeasure, consisting of a challenge-response authentication mechanism, will enable the data owner to successfully use her own credentials without establishing a total trust relationship with terminals. It will also offer an high grade of non-repudiation of origin and non-repudiation of sending, in the acceptation given in ISO/IEC 13888 [9].

A. Context

The threat model and the countermeasure, discussed in the rest of this section, will refer to the context described here.

In particular, we assume that:

- A secure channel[1] has been setup between the terminal and the remote Authentication Server (AS). The secure channel must guarantee the integrity, confidentiality and authenticity of data exchanged by the terminal and remote credential holding system over untrusted networks.
- The user owns a personal trusted device (hand-held passcode generator). The device will be used to communicate with the Card Holder and optionally may be used to calculate the message digest of the data to be processed.

[1] *Mechanisms for cryptographically strong password-based key agreement and mutual authentication could make mobility easier. These technologies are now object of standardization initiatives that proceed in synergy in IEEE 1363.2 [23] and IETF [22, 24].*

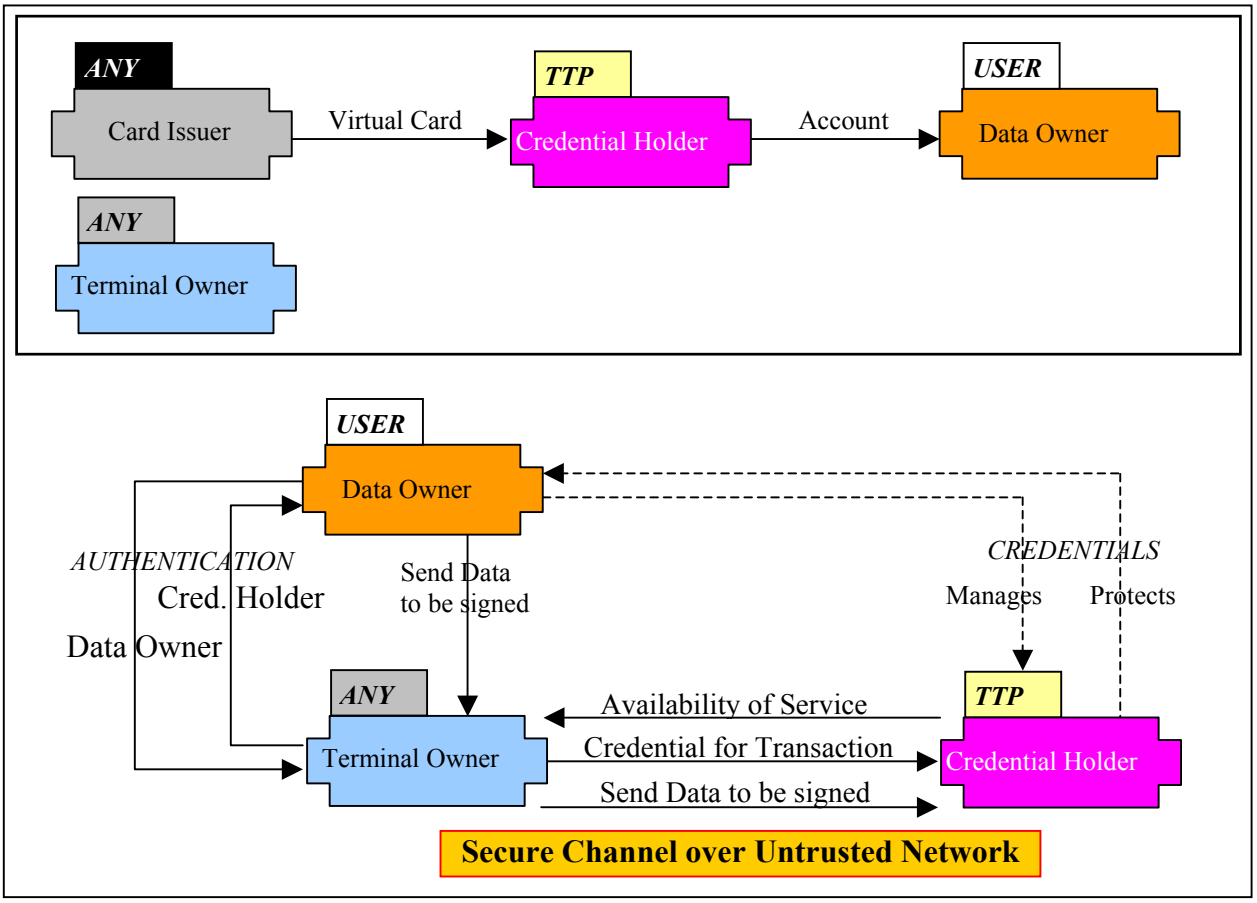

Figure 3: Virtual Smart Card System

- The infrastructure accessed through the AS is to be considered trusted, either in case data owner and credential holder roles are coalesced (e.g., the remote system is controlled by the data owner), or in the presence of a TTP delegated to store and manage credentials.

B. The threat model

The end-to-end security offered by the secure channel, although counteracting Man in the Middle (M.I.M) attacks, do not address the whole set of attacks against the data owner. In fact, the two parties that have the control of the channel can obviously analyze, tamper, reply, store and disclose data they route.
But, while one of the two parties (AS) is trusted, the other (terminal) is not: the considerations made throughout (Section IV.B) do not permit data owners to trust terminal owners (or their devices). So, the terminal owner is generally capable to perform the following attacks:

1. Message Substitution;
2. Credentials Theft;
3. Impersonation;
4. Replay Attack;
5. Interleaving Attack;
6. Cheating on Output;
7. Forced Delay Attack;
8. Sensible-data Disclosure;
9. D.o.S.

However, in the PKI domain, the most critical attacks for a given entity (data owner) are those that let the attacker (terminal owner) abuse of the entity's credentials. These attacks are message substitution, credentials theft, impersonation, replay attack and interleaving attack.

> "The most serious integrity problem for ubiquitous computing, therefore, is once again not with the messages in transit but with the device itself" [22].

C. A Countermeasure

To counteract the attacks mentioned above it is necessary to guarantee the integrity and authenticity of the interaction between lawful data owners and respective credential holders.
These guarantees must be provided for each credential utilization request. Therefore, the claimant must corroborate each command he instructs with information that authenticate himself univocally.
This is achievable using a time-based challenge-response identification protocol: the claimant (data owner), answering a time-variable challenge, provides the proof of his identity to another entity (the verifier) and corroborates his willingness to instruct specific commands.

A (claimant) → *B* (credential holding system):

$$E_K(t_A, B^*) \quad (1)$$

Our protocol is based on a mechanism standardized in ISO/IEC 9798-2 [10]. Regarding notation: t_A, denotes a timestamp produced by *A*; E_K denotes a symmetric encryption algorithm, with a key *K* shared by *A* and *B*. The asterisk (*) denotes the optional message fields, while a comma (,) within the scope of E_K denotes concatenation.
Modeling the optional message fields it is possible to bind, in a not-tamperable way, critical data to the user's identity.
In the following subsections we will describe the proposed protocol and we will analyze it with the logic of authentication of Burrows, Abadi and Needham [28] extended to handle secure and timely channels [27].

C.1 Notations and assumptions

The notations used in the reminder of this section are as follows:

- *credHoldSys* is the credential holding system, *dataOwn* is the data owner, *termOwn* is the terminal owner, *HPG* is the hand-held passcode generator, *P* is a generic third party public untrusted terminal;
- *K* is a shared secret key shared by *dataOwn*'s *HPG* and *credHoldSys*;
- *L* is a secure channel that has been build between *P* and *credHoldSys*;
- I_{HPG} is the passcode generator input keypad (input to *HPG*);
- O_{HPG} is the passcode generator output display (output from *HPG*);
- I_P is the input system of a generic terminal (input to *P*);
- O_P is the output device of a generic terminal (output from *P*).

In the rest of this subsection, we state the assumption made respectively about keys and secrets, channels and trust relationships.

Assumptions about keys and secrets:

1. *HPG* **believes** $\rightleftharpoons^{PIN}$ *dataOwn*: the hand-held passcode generator believes that PIN is a secret shared with the user;
2. *credHoldSys* **believes** $\rightleftharpoons^{K}$ *HPG*: the credential holder believes that *K* is a secret shared between the data owner's hand-held passcode generator and him.

Assumptions about channels:

1. *HPG* **believes** $\prec^{O_{HPG}}$ *HPG*: the hand-held passcode generator believes that its display is a secure channel from it;
2. *dataOwn* **believes** $\prec^{O_{HPG}}$ *HPG*, *dataOwn* **believes timely**(O_{HPG}): the data owner believes that the hand-held passcode generator's display is a secure and timely channel from the *HPG*;
3. *dataOwn* **believes** $\prec^{I_{HPG}}$ *HPG*: the data owner believes that the hand-held passcode generator's keypad is a secure channel from him;
4. *HPG* **believes** $\prec^{I_{HPG}}$ *HPG,*

   HPG **believes timely**(I_{HPG})*:* the hand-held passcode generator believes that its keypad is a secure and timely channel;
5. *credHoldSys* **believes timely**(*L*), *dataOwn* **believes timely**(*L*): the credential holder and the data owner believes *L* to be a timely channel;
6. *credHoldSys* **believes** $\prec^{L}$ *P*, *dataOwn* **believes** $\prec^{L}$ *P*: the credential holder and the data owner believes the link *L* to be a secure channel from *P*: all messages on link *L* are known to have been sent by *P*.

Assumptions about trust:

1. *dataOwn* **believes** *P* **controls** $\prec^{O_P}$ *P*: the data owner trust the terminal when it says that the display O_P is a channel from it;
2. *dataOwn* **believes** *HPG* **controls** *K*: the data owner trust the hand-held passcode generator when it says it protects the shared secret *K* from exposure;

3. *credHoldSys* **believes** *HPG* **controls** *K*: the credential holder trust the hand-held passcode generator when it says it protects the shared secret *K* from exposure;
4. *credHoldSys* **believes** *dataOwn* **controls** PIN: the credential holder trust the Data Owner protects PIN from critical exposure.
5. *dataOwn* **believes** *credHoldSys* **controls** *K*: the data owner trusts the credential holder when he says he protects the shared secret *K* from exposure;
6. *credHoldSys* **believes** $\forall X$. (*HPG* **controls** *dataOwn* **believes** *X*): the credential holder trusts the hand-held passcode generator to relay the data owner's beliefs;
7. *dataOwn* **believes** $\forall X$. (*credHoldSys* **controls** *dataOwn* **believes** *X*): the data owner trusts the credential holder to operate on behalf of the data owner.

Moreover, we assume the encryption scheme is IND-CPA.

C.2 The protocol

Now we discuss the protocol in detail, and show that it offers a high grade of non-repudiation of origin and non-repudiation of sending.

$\forall \langle X \rangle_{\gamma}$ *credHold* **believes** *dataOwn* $\mathbf{said}_{\mathbf{L}}$ *X*

Where γ is a one-time passcode generated by the HPG. For the sake of concreteness, let us imagine that a user want to use his cryptographic credentials stored in a remote and trusted credential holding system.
Moreover, the user wishes to distrust a generic third party public terminal used to get access to the service.

1. The user authenticates himself to his hand-held passcode generator (HPG) by entering a PIN into the keypad of the HPG.

   dataOwn $\rightarrow$ *HPG: PIN* on I_{HPG},
2. The entry of the correct PIN indicates to the HPG that the genuine user is present, rather some HPG thief (*HPG* **believes** *dataOwn* **controls** *HPG)*. Hence, the HPG ask the user to enter into its keypad the required time variable parameters

   HPG **believes timely**(I_{HPG}),

   HPG **believes** $\prec^{I_{HPG}}$ *HPG,*

   HPG **sees** $_{I_{HPG}}$ *PIN*,

   HPG **believes** $\rightleftharpoons^{PIN}$ *dataOwn*
3. The user selects the operation mode (*mode*), then specify the operation ID (*OpID*), the remote credential ID (*credID*) and the message digest of the document he wants to process (π*(intputData)*).

   dataOwn $\rightarrow$ HPG: ⟨mode, OpID, credID, π(inputData)⟩ on I_{HPG},

   HPG **sees** $_{I_{HPG}}$ ⟨mode, OpID, credID, π(input Data)⟩$_{PIN}$,

   HPG **believes** $\rightleftharpoons^{PIN}$ *dataOwn* $\Rightarrow$

   HPG **believes** dataOwn **believes** *mode, OpID, credID,* π *(inputData)*.

   In this way the data owner indicates that he wants a one-time code that allows him to perform the operation identified by *OpID*, on some data which message digest is π*(intputData)*, using the credential identified by *credID* and using the operational mode referenced by *mode*.
4. The HPG responds by displaying a one-time passcode (γ) generated using a general timestamp-based mechanism for unilateral authentication. In particular, the HPG computes:

   $\text{DataOwn} \rightarrow P : \langle \text{userID}, \text{OpID}, \text{credID}, \text{inputData} \rangle_{\gamma}$ *on* I_p

   $E_K(t_{HPG}$*, mode, s,* $\pi(t_{HPG}$*, mode, s,* π*(inputData), OpId, credID, userID))*

   Where t_{HPG} denotes a timestamp produced by HPG and allows either to achieve timeliness and uniqueness guarantees, or to detect replay attacks and forced delays.
   s denotes an auto-produced random seed. In this context it acts as a confounder necessary to provide unpredictability, and to preclude some chosen-text attacks.
   π is a collision-resistant one-way hash function. π(inputData) denotes the hash value computed applying π to the input data that should be processed. Binding this value to the authentication code makes message substitution attack impossible for malicious terminals (i.e., terminals becomes unable to instruct unlawfully the sign or decryption of arbitrary documents).
   OpID identifies the operation the data owner wishes to instruct to the credential holder. By binding this ID to the response, the terminal becomes unable to instruct operations different from those intended by the data owner.
   Analogously, *credID* identifies the remote credential the data owner wishes to use. Binding this value to the response make impossible for attackers to request the use of other credentials, associated to the same data owner.

HPG → *dataOwn:* $\gamma = E_K(t_{HPG}, mode, s, \pi(t_{HPG}, mode, s, \pi(inputData), OpId, credID, userID))$ on O_{HPG}

dataOwn **believes** $\prec^{O_{HPG}}$ *HPG*, *dataOwn* **believes timely**(O_{HPG})

dataOwn **sees** $_{O_{HPG}}$ $\gamma \Rightarrow$

dataOwn **believes** *HPG* **said** $_{O_{HPG}}$ γ,

dataOwn **believes fresh**(γ),

γ is necessary to guarantee, in one pass, the remote trusted credential holding system either on the integrity of data forwarded using the public untrusted terminal, or on the data owner willingness to instruct a specific operation.
The data owner believes that γ is a fresh one-time code generated for him to permit him to perform the requested operation.

5. The user, using the a public terminal input systems, specify his user identifier (*userID*), the operation he wants to perform (*OpID*), the remote credential ID he wants to use to perform the given operation, the one-time passcode (γ) generated using his HPG, and the input data he want to be processed.

6. The terminal sends the 5-tuple *(userID, γ, OpID, credID, inputData)* to the remote credential holding system. For an analysis of attacks that the terminal may try to perform please refer to Section 4.4.

$$P \rightarrow CredHold : \langle \text{userID}, \text{OpID}, \text{credID}, \text{inputData} \rangle_{\gamma} \text{ on } L$$

7. Upon the reception of the 5-tuple, the remote credential holding system decrypts the corroboration data (γ) using the secret key (*K*) shared with the user's HPG identified by *userID*. If the decryption succeeds, the timeliness, uniqueness and integrity of the request get validated according to the operation modality specified in *mode*. The MAC-then-Encrypt composition method used in the proposed challenge-response mechanism provides INT-PTXT (integrity of plaintext) [33]. If the request gets successfully validated, the credential holding system performs the requested operation and sends the results to the terminal.

credHoldSys **believes timely**(*L*), *credHoldSys* **believes** $\prec^{L}$ *P, credHoldSys* **believes** *HPG* **controls** *K, credHoldSys* **believes** ∀*X*. (*HPG* **controls** *dataOwn* **believes** *X*), credHoldSys **sees**$_{\mathbf{L}}$ $\langle userID, OpID, credID, inputData \rangle_{\gamma}$

if the authentication succeed the credential holder believes that data owner said (userID, OpID, credID, inputData) on L.

credHoldSys **believes** *dataOwn* **said**$_{\mathbf{L}}$ *(userID,OpID,credID,inputData)*

Hence, the trusted credential holder perform the requested operation and send back to the terminal the response.

CredHoldSys → *P*: *(response, exitCode)* on *L*

Moreover, the credential holding system should produce and preserve the evidence of fulfilled operations. Alternatively he may choose to notify the result of performed tasks to the user through a trusted channel or in authenticated manner.

8. Finally the terminal returns the response to the data owner.

P → *dataOwn*: *(response, exitCode)* on O_P

All that an unfaithful terminal can do are D.o.S. attacks. Credentials get not compromised.

D. Protocol Security and (Un)Feasibility of Introduced Attacks

The security of the mechanism relies on the quality of the symmetric encryption algorithm associated to E_K and to π. The selection of key sizes for a generic symmetric cryptosystem should be made according to actual guidelines to the selection (e.g., see [14]).
2nd-preimage resistance of π will be also determinant. A malicious terminal, in fact, upon reception of the document that should be processed, may try to find a *2nd-preimage inputData′* where

$inputData' \neq inputData$

such that

$\pi(inputData) = \pi(inputData')$.

A successful search may enable the terminal to perform a message substitution attack, without tampering the token specified by the data owner. However, to succeed in this attack, the search must end within the moment in time determined by the timestamp included in γ and the acceptance window (Δt), used by the remote AS to validate the response timeliness. Answering the challenge immediately before submitting to the terminal the document in input, minimize the time interval available to perform the attack. In all cases the encryption scheme is assumed to provide IND-CPA.

Hand-held Passcode Generator
In a ubiquitous context, answering the challenge in the proposed challenge-response protocol requires some type of portable and trusted computing device.

Observing the precepts that characterize an application model for pervasive computing [7] we can tight again the set of devices adequate to implement the challenge-response mechanism.
Hand-held passcode generators are the natural choice. The generator asks in input the external time-variable parameters (*mode*, π(*inputData*), *OpID*, *credID*), and displays the relative passcode, computed as a function of either the secret key K, timestamp t_A or random seed s.
Hand-held passcode generators can facilitate the deployment, since they do not require additional local readers on each terminal. Yet already widespread devices (e.g., mobile phones) are the natural choice to implement hand-held passcode generators.

Implementation Issues
A special care should be given in selecting the amplitude of (Δt): the choice must be made in consideration of either the length limits that each passcode generator has on input fields, or the format used to display data. These factors limit the maximum length of the hash value computed on *inputData*. For example, and hand-help passcode generator that accepts inputs up to 24 digits only in base10 will limit the length of the hash value to 96 bits. Δt should be chosen accordingly.
Moreover, to prevent typing errors, a check digit, based on ISO/IEC FCD 7964 [21], may be applied on the response.

E. Other Attacks

While the use of the proposed countermeasure guarantees the integrity and authenticity of exchanged data, other attacks remain feasible against data owners. The terminal, for example, can cheat about the output received from the credential holder. To counteract this attack, the credential holder can produce the evidence of fulfilled tasks and preserve it through an audit trail. Alternatively the credential holder can authenticate output data or notify the lawful user through a trusted channel about the real tasks performed with his credentials.

VI. Further Work

We see the following opportunities for further work:

- Producing a Protection Profile for credential holding systems used by TTPs.
- Formalizing the introduction in todays PKI Trust Models of TTPs for credential holding.
- Standardizing the interface to credential holding systems owned by third parties.

VII. Conclusions

We have explained why remote credential storage systems such as Virtual smart cards can be an attractive alternative to traditional smart cards.
We have also shown that the full equivalence of these Virtual smart card systems towards traditional smart cards can be achieved by enabling a safe -with respect to the exposure of user's credentials- interaction of the Data Owner and the Credential Holder through an untrustworthy Terminal.
To fit the above purpose we have supplied a cryptographic protocol that uses a time dependent challenge-response in order to obtain a high grade of non-repudiation of origin and non-repudiation of sending.
Finally, we have analyzed the implications of the separation of the user from its credential storage system showing that, depending on the liability model in use in a given PKI, there may exist the need to make explicit to a relying party the interaction between the Data Owner and its correspondent Credential Holder.

Acknowledgments

The authors would like to thank Romano Pedroli and Hector Martinez for their useful suggestions and determinant help in drafting the English version of this paper. Thanks are also due to Raffaello Galli, Giacomo Sesto and Trevor Perrin for their helpful feedbacks.

References

[1] S. Gupta, "Security Characteristics of Cryptographic Mobility Solutions", 1st Annual PKI Research Workshop, April 24-25, 2002. NIST, Gaithersburg MD, USA

[2] Naomaru Itoi, Tumoko Fukuzawa and Peter Honeyman, "Secure Internet Smartcards", CITI Technical Report 00-6, Program in Smartcard Technology, <http://www.citi.umich.edu/projects/smartcard/>

[3] Thierry Lamotte, "IP Smart Cards in the (Not So) Distant Future", ETSI Project Smart Card Platform Meeting #5, Palm Springs, USA, March 20-23, 2001

[4] Jim Rees and Peter Honeyman, "Webcard: a Java Card web server", CITI Technical Report 99-3, Program in Smartcard Technology, <http://www.citi.umich.edu/projects/smartcard/>

[5] R. Sandhu, M. Bellare and R. Ganesan, "Password-Enabled PKI: Virtual Smart Cards versus Virtual Soft Tokens", 1st Annual PKI Research Workshop, April 24-25, 2002. NIST, Gaithersburg MD, USA

[6] Bruce Schneier and Adam Shostack, "Breaking Up is Hard To Do: Modeling Security Threats for Smart Cards", October 19, 1999, USENIX Workshop on Smart Card Technology, USENIX Press, 1999, pp. 175-185, <http://www.counterpane.com/smart-card-threats.pdf>

[7] G. Banavar, J. Beck, E. Gluzberg, J. Munson, J. Sussman, and D. Zukowski, "Challenges: An Application Model for Pervasive Computing", 6th Annual ACM/IEEE International Conference on Mobile Computing and Networking (Mobicom 2000)

[8] RSA Laboratories, "PKCS#11 v2.11: Cryptographic Token Interface Standard", Revision 1a, November 2001, <ftp://ftp.rsasecurity.com/pub/pkcs/pkcs-11/v211/pkcs-11v2-11r1.pdf>

[9] "ISO/IEC 13888, Information Technology, Security techniques, Non-repudiation", Internation Organization for Standardization, Technical commitee JTC 1/SC 27

[10] "ISO/IEC 9798-2: 1997, Information Technology, Security techniques, Entity Authentication, Part 2: Mechanisms using symmetric encipherment algorithms", Internation Organization for Standardization, Technical commitee JTC 1/SC 27

[11] Adam Young and Moti Yung, "Kleptography: Using Cryptography Against Cryptography", Eurocrypt 1997, IACR, Springer-Verlag

[12] Sergei Skorobogatov and Ross Anderson, "Optical Fault Induction Attacks", University of Cambridge, 2002

[13] S. Moore, R. Anderson, P. Cunningham, R. Mullins and G. Taylor, "Improving Smart Card Security using Self-Timed Circuits", Async 2000

[14] Arjen K. Lenstra and Eric R. Verheul, "Selecting Cryptographic Key Sizes", Journal of Cryptology, IACR, Vol. 14 N. 4 Autumn 2001, Springer-Verlag

[15] "X.509 (03/00) Information technology - Open Systems Interconnection - The Directory: Public-Key and attribute certificate formats"

[16] Alfonso De Gregorio, Thomas Fossati and Giovanni Frustaci, "xplitPKI", to-appear

[17] D. Jablon, "The SPEKE Password-based Key Agreement Methods", Internet Draft, April 15, 2002, <http://www.ietf.org/internet-drafts/draft-jablon-speke-01.txt>

[18] IEEE P1363 Working Group, ", Ed. D. Jablon, "IEEE P1363-2: Password-Based Public-Key Cryptography, <http://grouper.ieee.org/groups/1363/passwdPK/index.html>

[19] T. Wu, "The SRP Authentication and Key Exchange System", RFC 2945, IETF, September 2000, <http://www.ietf.org/rfc/rfc2945.txt>

[20] Trevor Perrin, Logan Bruns, Jahan Moreh, and Terry Olkin, "Delegated Cryptography, Online Trusted Third Parties, and PKI", 1st Annual PKI Research Workshop, April 24-25, 2002. NIST, Gaithersburg MD, USA

[21] "ISO/IEC FCD 7064 Information technology, Security techniques, Data Processing, Check character systems", International Organisation for Standardization, Technical programme JTC1/SC27

[22] Frank Stajano and Ross Anderson, "The Resurrecting Duckling: Security Issues for Ubiquitous Computing", First Security & Privacy supplement to IEEE Computer, April 2002

[23] Tsutomu Matsumoto, “Human identification through insecure channel”, Theory and Application of Cryptographic Techniques, pp.409-421, 1991.

[24] Tsutomu Matsumoto, “Human-computer cryptography: An Attempt”, ACM Conference on Computer and Communication Security, pp. 68-75, 1996.

[25] Nicholas J. Hopper and Manuel Blum, “A Secure Human-Computer Authentication Scheme”

[26] Rachna Dhamija and Adrian Perrig, “Dejà vu: A user study, using images for authentication”, Proceedings of the 9th USENIX Security Symposium, 2000.

[27] Martin Abadi, Michael Burrows, C. Kaufman and Butler W. Lampson, “Authentication and Delegation with Smart-cards”, In Theoretical Aspects of Computer Software, pp. 326-345, 1991.

[28] Michael Burrows, Martin Abadi and Roger M. Needham, “A Logic of Authentication”, Proceedings of the Royal Society of London, A Vol. 426, 1989, pp. 233-271. A preliminary version appeared as Digital Equipment Corporation System Research Centre report No. 39, February 1989.

[29] Tal Rabin, “A Simplified Approach to Threshold and Proactive RSA”, In H. Krawczyk, editor, Advances in Cryptology, CRYPTO'98, Lecture Notes in Computer Science Vol. 1462, pp. 89--104, Springer-Verlag, 1998.

[30] H. Krawczyk, “Simple Forward-Secure Signatures From any Signature Scheme”, In Sushil Jajodia and Pierangela Samarati editors, Proceedings of ACM CCS '00, pp. 108-115, 2000

[31] Victor Shoup, “Practical Threshold Signatures", Lecture Notes in Computer Science, Vol. 1807, 2000. A preliminary version of this paper also appeared as IBM Research Report RZ 3121, April 19, 1999.

[32] A. Young, M. Yung, "The Dark Side of Black-Box Cryptography -or- Should We Trust Capstone?", CRYPTO '96, pp 89-103, Springer-Verlag

[33] M. Bellare and C. Namprempre, “Authenticated Encryption: Relations among notions and analysis of the generic composition paradigm”, Extended abstract in Advances in Cryptology - Asiacrypt 2000 Proceedings, Lecture Notes in Computer Science Vol. 1976, T. Okamoto ed., Springer-Verlag, 2000. Full version available at <http://www.cs.ucsd.edu/users/mihir/papers/oem.ps>