9

Tamper-Resistance Network:
an Inforastructure for moving electronic tokens

Kimio Kuramitsu and Ken Sakamura
Interfaculty Initiative in Information Studies, University of Tokyo, 7-3-1 Hongo Bunkyo-ku, Tokyo 113-0033. Japan, Phone: (+81) 3 5841 2484 / Facsimile: (+81) 3 5841 8459, {kuramitsu, sakamura} @iii.u-tokyo.ac.jp

Abstract: Moving electronic tokens or tickets over networks makes business processes more effective. At the same time, such a movement will sometimes produce disturbed results, such as illegal copies, unauthorized disclosures, and accidental destructions. Tamper Resistance Network (TRN) is a secure distribution infrastructure networked among smartcards and other tamper-proof devices. Over TRN, we can move tokens electronically, preventing from being duplicated or lost. In order to enhance the security of online token trading, the control of atomicity and the trace of the movement are discussed. In addition, we will present an initial experimentation, where electronic tickets are sold, exchanged, and examined over the prototyped TRN implementation.

1. INTRODUCTION

A token, or a ticket, is information with *value.* Information technology today opens up great possibilities for digitalizing tokens and its applications. In particular, a tamper-proof personal device such as a smartcard can protect data from illegal copies and unauthorized modification. Using such a device, we can secure the value of electronic tokens.

Furthermore, a token is essentially a moving object; many kinds of tickets allow a buyer to transfer a ticket to others. Even a nonnegotiable token such as an airline ticket could be distributed through an intermediary, like a travel agency. Accordingly, the *transferability* is very important for implementing electronic token applications. The objective of this study is to secure the transfer of tokens among tamper-proof devices.

The starting point of our approach is, say, a virtual private network among tamper-proof devices. An encrypted session directly opened between two devices

can extend tamper-resistance to a token moving over the network. Furthermore, we will discuss two characteristics that make the encrypted network more reliable and more secure.

1. **Atomicity.** The transfer of electronic token is an all-or-nothing processing between two devices. The token should neither be doubled nor lost, even if the communication is interrupted with malicious intents.
2. **Traceability.** The movement of token should be traceable. While ticketing companies are forced to take good care of users' privacy, they must also prevent, say, "bad money drives out good."

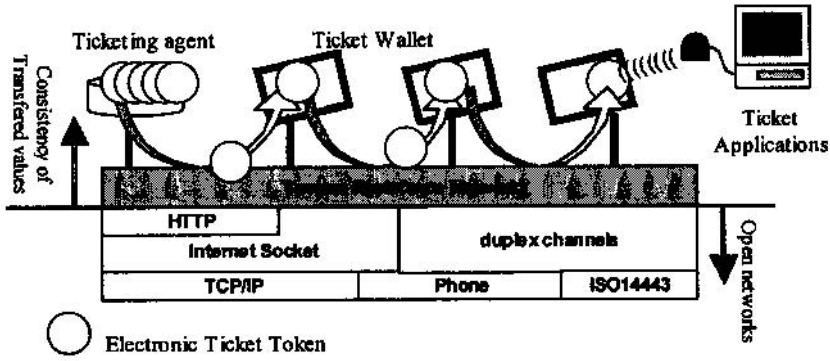

Figure1. The Layered Architecture of Tamper Resistance Network

Tamper Resistance Network (TRN), we will propose here, is a networked environment that enables to ensure the consistency of tokens, distributed among all of the personal devices. The number of tokens distributed over TRN is controlled, and illegally duplicated tokens are detectable. Furthermore, we design TRN as an infrastructure for multiple ticketing vendors; that is, the TRN provides every company with the common secure channel, on which each can setup ticketing applications originally.

This paper presents the design of the TRN infrastructure and its initial implementation. The remainder of the paper is organized as follows. Section 2 overviews the concept of TRN and summarizes its requirements. Section 3 designs the TRN scheme. Section 4 discusses the atomicity of TRN and the security enhancement of the traceability. Section 5 describes a prototyped implementation in the experimental project. Section 6 concludes the paper.

2. ELECTRONIC TRANSFERABLE TOKEN

2.1 Motivation

First of all, let us imagine a hard ticket to a popular event, such as the World Cup soccer and an Olympic game. Vacant seats are not desirable for both the organizer and people who want to attend the game. We consider that electronic transferable tickets will reduce unused tickets and vacant seats; for example, a person who suddenly cancels the game can send his or her ticket to a remote buyer. Also, a secondary marketplace that an authorized broker organizes will inject liquidity of tokens over the Internet.

The movement of tokens, however, results in heightened risks of destroying value, for example, illegal copies, unauthorized disclosures and accidental destructions. Thus, many ticketing systems restrict online transfer. The smartcard-based ticket is regarded as a semi-digitalized solution, where a ticket is exchanged physically although stored electronically. More recently, NTT's Digital Ticket system extends the trading of tickets on smartcard through trusted third parties. [12] However, the transfer through a mediator seems to be inefficient, too complex and less scalable. In addition, some people dislikes even trusted parties in terms of trading privacy.

Our fundamental approach to Tamper Resistance Network is based on two-party communication. A ticket wallet (depicted in the next section) directly sends an electronic token to another wallet. In this context, the TRN could be said to be a virtual private network, tunneling over open networks, authenticating with each other, and encrypting traffic. Indeed, there are not a few standards proposed for secure communication between smartcards. (The typical example is Modex's Value Transfer Protocol [13]) The uniqueness is that the security of TRN is designed to ensure the consistency of tokens distributed among the set of personal wallets. The atomicity control makes the basis of transfer more reliable. The traceability of the transfer, furthermore, enables the recovery of lost tokens and the detection of illegal tokens distributed on the public.

2.2 Ticket Wallet

The user must store an electronic token on a personal *ticket wallet* device. The tamper proof hardware is essential for protecting stored objects from unauthorized disclosure and modification. Its core is a secure integrated circuit computer, consists of multiple components – CPU, a cryptographic coprocessor, dynamic RAM and nonvolatile storage such as E2PROM and FeRAM and communication peripherals – combined with a single chip. The software that controls the transfer of token is installable on the chip.

Smartcard is one of the typical tamper resistance hardware, which embeds a secure chip in a plastic case [5]. However, the smartcard is not necessary suitable to store transferable tokens, because the plastic case gives no help to know the changing content of the card. The user will need some user interfaces to browse the content of token or send a token to another. We have therefore envisioned a smart phone-based ticket wallet. Using user interface and communication capability that the smart-phone supports, the user can easily exchange tokens. Figure 2 illustrates a portable phone, on which a secure chip is installed.

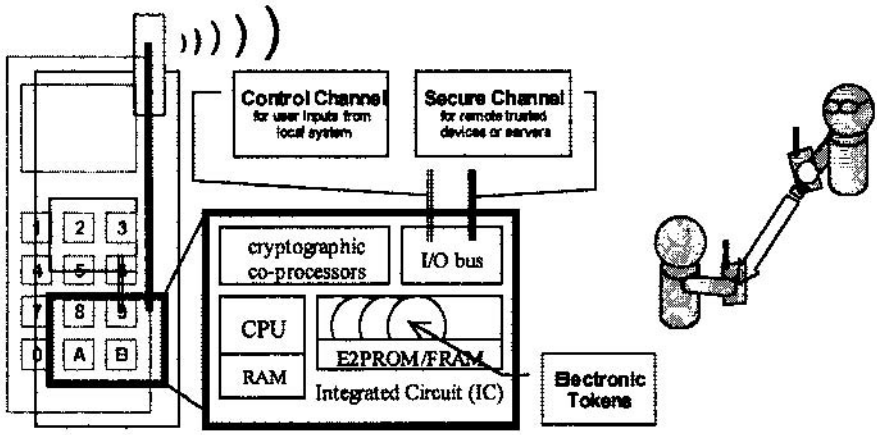

Figure 2. The Phone-based Ticket Wallet

Note that the TRN infrastructure should be designed as device-independent. In particular, inexpensive smartcards are useful for disposal ticket wallet devices. Accordingly, important is the compatibility with the smartcard technology that is limited in memory and processing resources. For example, the state of the art contact-less smartcard only supports an 8~16bit CPU, a 16~32K ROM for program module, 512bytes of RAM and 4~16K EEPROM for data storage. Although higher performance chips are under development, we should pay attention to its compactness.

2.3 Scope and Requirements

Tamper Resistance Network is a secure distribution environment, where a token can move from one wallet to another. In each wallet, a TRN driver will be installed, which authenticates a communicating entity, opens an encrypted session, and control the transfer of tokens. We premise that the TRN driver is certificated and protected from illegal disclosures and modifications, as well as a token object.

For the authentication and encryption, many techniques are now available on smartcard. [5] In the section 3.2, we will present one of the authentication schemes

that satisfy the TRN session. Over them, we will add two features: *transaction control* and *movement trace.* Here we define the "ACID" properties that the TRN driver should ensure.

- **Atomicity.** The transfer of token is an all-or-nothing processing between two devices. The interruption must restart to complete the transferred or untransferred status (without any external coordinators).
- **Consistency.** The restart timing of the interrupted transfer is unpredictable, because the token is moved in suspicious circumstances. The token should neither be doubled nor eliminated from start to end transaction.
- **Isolation.** A token owner must have the ability to retransfer the received token to another wallet as a new transaction.
- **Durability.** The wallet must record logging information to restart and synchronize the interrupted transaction.

Ideally, the control of atomicity enables us to ensure the consistency of distributed tokens over the TRN environment. However, unexpected accidents and troubles might occur. The record of movements also helps detect such accidents and recover (or at least minimize) system troubles. We therefore need to trace moving tokens over the TRN.

Note that the TRN infrastructure proposed in this paper only discusses the scheme of transferring and tracing a data object. We will not specify any format of token, or the semantics of each token. Many applications today encode a ticket object using ASN. 1 BER TLV's (type-length-value) record. [5] In addition, ISO7816-4 defines Application Protocol Data Unit to interact commands to access the record stored on smartcard. The TRN is defined only with the management of token movement, independent of a specific format and access method.

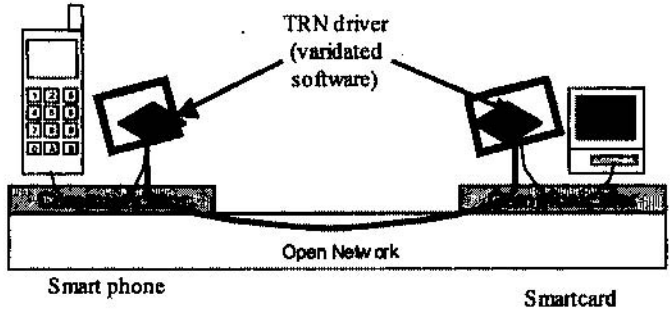

Figure 3. The TRN Driver is Installed in the Secure Chip

3. TAMPER REGISTANCE NETWORK

Tamper Resistance Network is a networked environment, where we can move electronic tokens from one tamper-proof device to another. To secure the movements on the TRN, we design the two protocol schemes, *token atomicity* and *transfer traceabilitiy,* over an *authenticated* session. Note that we premise that the TRN scheme designed in this section will work on a half duplex channel.

3.1 Overview

The TRN environment is comprised of several kinds of entities: a certification authority, ticketing agents, token objects and token devices. To start, we first define a small amount of preliminary notations. Let **C** be a finite set of ticketing agent C_i, **D** be a finite set of the devices D_i , and **O** be a finite set of token object O_i . Throughout the paper, we use the notation shown in Table 1.

A➔B: m	Entity A send a message to another entity
m1, m2	The concatenation of messages m1 and m2
k(m)	Message m encrypted using a cryptographic key k
pk_A, sk_A	A pair of public and private keys of entity A.
$[m]_A$	A digital certificate (signature) for message m by entity A
h(m)	A cryptographic digest of message m

Table 1. Notation used for Protocol and Cryptographic Operation

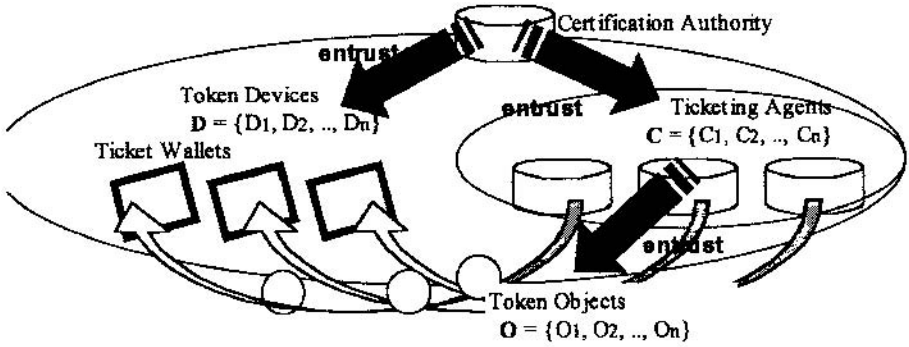

Figure 4. The Certification Relationship Between Entities in TRN

Definiton1. A *certification authority* is a primary trusted entity that controls the whole authenticity of the remainder of all entities (represented as $\mathbf{C} \times \mathbf{D} \times \mathbf{O}$) on

TRN. The authority has a pair of *root asymmetric key* $\{pk_{CA}, sk_{CA}\}$, and can issue digital certificate $[m]_{CA}$ for any messages.

Definition2. A *ticketing agent* C_i is a creator of ticket objects. To issue a certificate for the token object, C_i has a pair of asymmetric key $\{pk_{Ci}, sk_{Ci}\}$. The certification authority registers the certificate $[pk_{Ci}]_{CA}$ for the public-key pk_{Ci}.

Definition3. A *token device* D_i is a communicating entity, including service servers by the certification authority and each ticketing agent as well as personal ticket wallets. It has a tuple of $[id_i, pk_{Di}, sk_{Di}, [id_i, pk_{Di}]_{CA}, pk_{CA}]$, where id_i is a unique identifier of D_i, and a set of $\{pk_{Di}, sk_{Di}\}$ is an asymmetric key to authenticate an encrypted session. The certificate $[id_i, pk_{Di}]_{CA}$ ensures the authenticity of device D_i under the trust of the authority, and pk_{CA} is used to verify the certificate.

Definition4. A *token object* O_i is a tuple of $[oid_i, [oid_i]_{Cj}, body_i]$, where oid_i is a unique object identifier of O_i and $[oid_i]_{Cj}$ is a certificate for O_i by C_j, that is, the creator of O_i is a ticketing agent C_j. The $body_i$ is the body of the object, including a variety of token's properties.

Figure 4 shows the confidential relationship of these entities over the TRN environment. The certification authority gives one's trust to all ticketing agents and token devices. Each entrusted ticketing agent creates token objects over *certificate hierarchy.* The token objects can be moved between entrusted devices.

3.2 Authenticated Session

To start, we will define the TRN session, where two devices communicate together through an authenticated and encrypted stream. Generally, we can generally use a challenge-response authentication or a shared secret key to authenticate trust of smartcard. Also, these techniques are regarded as enough secure over smartcards. [5] However, the important thing to trace moving tokens is to authenticate the identification of communicating entity, as well as trust of them. Accordingly, we introduce a PKI-based authentication [1] that can identify each device. Following shows that the device D_A starts to authenticate D_B.

Step1. The device D_A initializes a session, and then sends the device identifier and its certificate to another device.

$$D_A \rightarrow D_B: id_A, pk_A, [id_A, pk_A]_{CA}$$

The device D_B, after receiving $[id_A, pk_A]_{CA}$, verifies the certificate using the public-key pk_{CA}. If verified, D_B can trust id_A and pk_A on the basis of the confidence of the certification authority. Note that since the certificate $[id_A, pk_A]_{CA}$ is public information, D_B should not at this step believe in a device at the other end of the line.

Step2. The device D_B generates a random number rnd_B and then encrypts it using the received public-key pk_A. In addition, the encrypted number is signed with the secret-key sk_B for avoiding *man-in-the-middle- attack.* D_B sends those of data, following the identification information of D_B.

$$D_B \rightarrow D_A: id_B, pk_B [id_B, pk_B]_{CA}, pk_A(rnd_B), [pk_A(rnd_B)]_B$$

Step3. D_A decrypts the number $pk_A(rnd_B)$ using sk_A, and at the same time verifies the signature $[pk_A(rnd_B)]_B$ using pk_B. If verified, D_A also generates a random number and sends it to D_B as well.

$$D_A \rightarrow D_B: pk_B(\underline{rnd_A}), [pk_B(\underline{rnd_A})]_A$$

Ultimately, both of D_A and D_B finish initializing a new session by exchanging random numbers through the encryptions by one another's public keys. Therefore, D_A and D_B can create a common symmetric key $k_{\underline{AB}}$ at each end.

$$h(rnd_A, rnd_B) \rightarrow k_{AB}; \quad \textit{a session key}$$

Note that D_A and D_B may need to send a greeting message to each other for testing the session key (for example, $D_A \rightarrow D_B : k_{AB}$("hello"))

Definition5. A *TRN session* is denoted as a relation *s* of [id_i , id_j , k_{ij}], where id_i and id_j is a pair of authenticated identifiers of communicating entities D_i and D_j , and k_{ij} is a session key dynamically generated at the current session. (Notice that we must implement the session that can detect *reply attack*[24])

3.3 Token Atomicity

If an electronic product were transferred as a simple monolithic object, the sender and the receiver will repeat ACK for ACK endlessly. Our approach begins with the extension of an object O_i to an electronic transferable token.

3.3.1 Status of Moving token

The movement of token could be thought of a status transition between two devices. To represent the status of a moving token, we define the following two parameters, which adds to O_i .

1. **value** parameter is a flag that determines whether the corresponding token is available, and whether the token is trustable.
2. **from-to** parameter is a record that represents where the corresponding token came from, or where the token will go to.

Definition 6. A *transferable token* or a status of moving O_i is denoted as a relation *t* of [oid_i $value_i$, from-to_i], where oid_i is an identifier of O_i , The $value_i$, is an element of {enabled, disabled, trustless, trustless_disabled}, and from-to_i is an element of the union set of {any, no} and **D.** Table 2 summarizes the semantics of *t* used in this paper.

[24] Some malicious users may attempt to defraud the TRN system reusing encryption messages. A unique identifier for each packet helps detect the replay of duplicated messages in the same session.

Status	Semantics
t[oid_i enabled, any]	Complete token
t[oid_i disabled, id_B]	Disabled token, except for moving to the device id_B.
t[oid_i trustless_disabled, id_A]	Trustless and disabled token, moved from the device id_A
t[oid_i trustless, id_A]	Trustless token, moved from the device id_A
t[null]	Deleted ticket

Table2: The Semantics of Moving Ticket t

3.3.2 Applied 2PC protocol

A two-phase commit (2PC) protocol is a basic scheme that synchronizes the status of transactions among multiple distributed systems. [3] We apply 2PC scheme to the status transition of t between two entities. Let M_{2PC} = {"copy", "copied", "commit", "committed", "true", "false"} be a set of protocol messages. Over a session s[id_A , id_B , k_{AB}], the ticket O_i is transferred from D_A to D_B as follows.

Step1. D_A first disables O_i (complete ticket) and sets the destination id_B at the from-to field (that is, t[oid_i enabled, any] ➔t[oid_i disabled, id_B]), and then starts to copy O_i onto D_B .

$$D_A \rightarrow D_B : k_{AB}(\text{"copy"}, O_i)$$

Step2. D_B stores the received Oi as t[oid_i trustless_disabled, id_A], and then returns an ACK message.

$$D_B \rightarrow D_A : k_{AB}(\text{"copied"}, \text{true})$$

Step3. D_A deletes Oi from the storage, and then commits the transfer of oidi

$$D_A \rightarrow D_B : k_{AB}(\text{"commit"}, oid_i)$$

Step4. D_B switches the value parameter from trustless_disable to trustless, (that is, t[oid_i trustless_disabled, id_A] ➔t[oid_i trustless, id_A]), and then returns an ACK for the commitment.

$$D_B \rightarrow D_A : k_{AB}(\text{"committed"}, \text{true})$$

Finally, D_A deletes the ticket status t[oid_i disabled, id_A] and finally closes the transaction

Sometimes, the receiver entity (D_B above) cannot store O_i, because of owner's rejection or out of memory. In such cases, D_B must respond the k_{AB} ("copied", false) message at Step2. Then, D_A aborts the transaction, roll-backing the status from t[oid_i disabled, id_B] to t[oid_i enabled, any]. Figure 5 illustrates full sequence diagrams at the committed and aborted transfer.

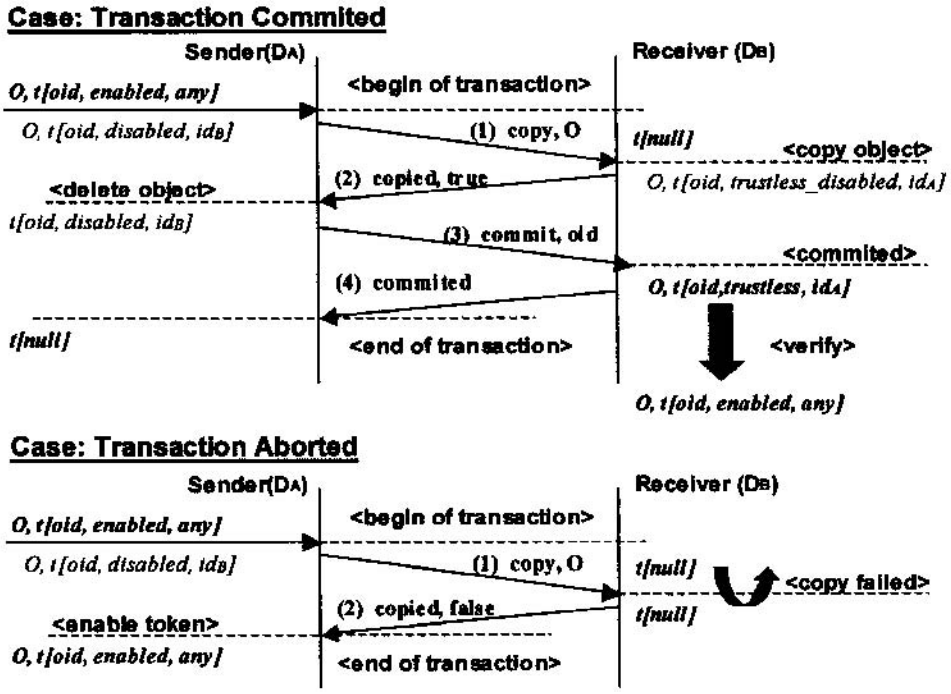

Figure 5. Protocol Sequence in the Transfer Transaction

Note that we already reported an atomic 2PC transfer scheme for an untraceable token [7]. This section extends the 2PC scheme to a traceable token. In short, it is possible to control whether we should trace the transfer or not, at Step4. If an untraceable token is committed, D_B only have to set the complete ticket as t[oid$_i$ enabled, any]. For furthermore information, we will refer to our earlier work [7].

3.4 Traceability

A relative problem with the transferability is the *credibility* of received tokens. Under environment where multiple venders can issue tickets, can you really trust an electronic ticket just entitled "Olympic Baseball Game"? Thus, the token O_i just after received should be restricted as trustless. In order to use the trustless token (or move to another devices), the owner first has to check the authenticity represented by $[\text{oid}_i]_{Cj}$. In this case, the verification of $[\text{oid}_i]_{Cj}$ requires the public-key of the creator C_j, but a resource limited device cannot store public-keys of all agents on **C.** Thus, the ticket wallet has to retrieve the public-key for $[\text{oid}_i]_{Cj}$ from the certification authority. Let M_{Auth} = {"seekfor", "seeked") be a set of protocol messages. Following shows the authentication of O^i over a new session $s[id_B, id_{CA}, k_{BCA}]$.

Step1. D_B seeks for the public key of the creator of O_i. At this time, the status of O_i is represented as t[oid$_i$ trustless, id_A], where id_A represents the identifier of the sender.

$$D_B \rightarrow D_{CA}: k_{BCA}(\text{“seekfor”}, oid_i, id_A)$$

Step2. The certification authority records the movement as a tuple of [oid$_i$, id_A, id_B] (note that, id_B stems from the present session), and then returns the public key of the creator of O_i.

$$D_{CA} \rightarrow D_B: k_{BCA}(\text{“seeked”}, pk_{Cj})$$

D_B verifies the certificate [oid$_i$]$_{Cj}$, and, if verified, switches the status from t[oid$_i$ trustless, id_A] to t[oid$_i$ enabled, any].

Due to resource constraints, we cannot record a chain of all movements on each token or device. However, this scheme can notify the certification authority of each movement of O_i, before the token owner uses or moves the token. The authority on the other hand can combine the notifications and complete the chain of all movements among devices.

Definition 7. A *token movement* is recorded as a tuple of [oid$_i$ id$_l$, id$_k$], which means that the object O_i was moved from id$_l$ to id$_k$.

4. DISCUSSION

4.1 ACID properties

In the transfer of a token object between particular two devices, the status of t is finite. Let the trustless status t[oid, trustless, id$_A$] on D_B be regarded as the same as the complete ticket t[oid, trustless, id$_A$], because the process of ticket verification in Section 3.4 is independent of that of the transfer. Table 3 addresses all pairs of the statuses (I) ~ (IV), transferred between D_A and D_B. Figure 6 illustrates the status transition diagram in the transfer of O_i. Here we will show a brief proof of the ACID properties in TRN.

- **Consistency.** Whenever the transfer is interrupted, the status of t is determined as one of (II) ~ (V). At each status of (II) ~ (V), the value of t on D_A and D_B is not simultaneously enabled. At the any status, the object O_i remains upon at least one device. Therefore, TRN never doubles the value nor lost the object completely.
- **Atomicity.** In TRN, the sender plays a role in a transaction coordinator that resumes interrupted transactions. From viewpoints of D_A, the discriminable interrupted status is limited; D_A can only differ (II) and (III) from (IV) and (V), in terms of the presence of O_i. Thus, D_A restarts the transaction as follows. If the transfer is interrupted at (II) or (III), D_A resends the message k_{AB}(copy, O_i). On the other hand, at (IV) or (V), D_A resends the message k_{AB} (commit, oid$_i$).

- **Durability.** Both of the restarting messages (copy O_i and commit oid_i) are generatable from the status of token respectively. Furthermore, the recipient (id_B) in the ongoing transfer is recorded in the from-to field.
- **Isolation.** At (V), the owner of D_B can move O_i to other device, because the message committed is generatable from t[*null*].

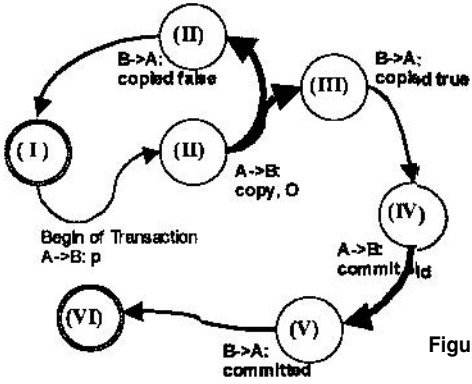

Figure 6. The Status Transition Diagram in transfer *p*

	D_A	D_B
(I)	O, t[*oid, enabled, any*]	t[null]
(II)	O, t[*oid, disabled, ids*]	t[null]
(III)	O, t[*oid, disabled, ids*]	O, t[*oid, trustless, ids*]
(IV)	t[*oid, disabled, ids*]	O, t[*oid, trustless ids*]
(V)	t[*oid, disabled, ids*]	O, t[*oid, enabled, any*]
(VI)	t[null]	O, t[*oid, enabled, any*]

Table 3: The Status of Object O and *t*

4.2 Traceability, Security, and Privacy

Here we will discuss how the traceability affects security, reliability, and privacy in the TRN infrastructure.

- **Detection.** A remarkable advantage with the traceability is that it enables the certification authority to detect device disclosures. The certification authority always analyzes the flow of tokens, and if finding the branch that means that the illegally duplicated tokens are distributed, the authority can stop the flow and block suspicious devices. That is, the TRN infrastructure has the ability to keep the consistency (and the number) of distributed tokens on its running.
- **Recovery.** An electronic token stored on the device might be broken by hardware longevity or natural disturbance such as radial ray and electromagnetic wave. (In order to enhance tamper resistance against a physical analysis, such breakdowns are also desirable) The tracing of the movement help detect the device, where a token will be lost for the last time. A ticketing agent will (if possible) reissue a lost token for the right owner.
- **Privacy.** A unique identifier is assigned to each personal device, but it doesn't associate with a particular personal identification. If there is no map from device to person, privacy could be preserved. But this is ideally a hope. To bring it all down to earth, someone will make the mapping. In order for the users to control privacy matter, the TRN infrastructure should permit the use of disposable (or recyclable) devices, as the need arises.

5. PROTOTYPED IMPLEMENTATION

In the JIPDEC[25] project, we experimented the transferability of electronic tickets over TRN. This section presents our initial implemented prototypes in the experimental project.

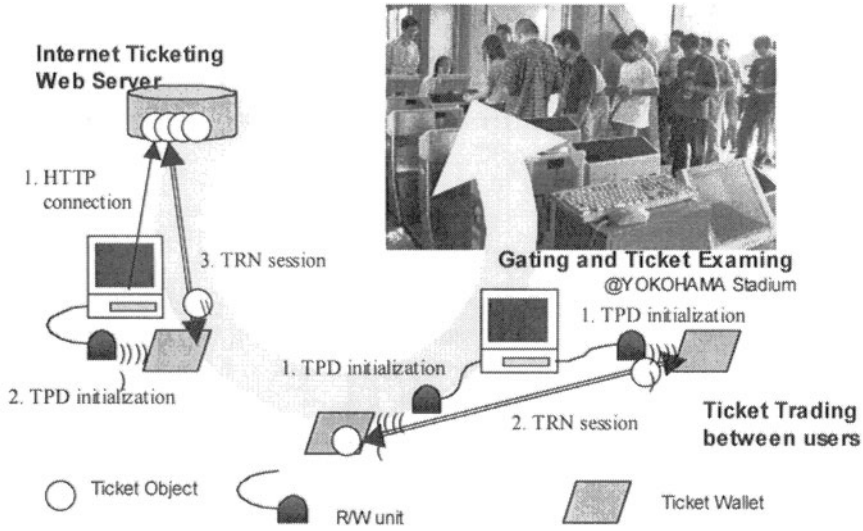

Figure 7. The Overview of Experimental Environment for Digital Ticketing.

5.1 Experimental Environments

The JIPDEC project assumes three ticket applications; (1) a user buying tickets online can receive electronic tickets directly from an Internet ticketing server to his or her personal ticket wallet; (2) the users also can freely exchange the tickets between ticket wallets; (3) ultimately, ticket examiners can check a ticket correctly at gating systems. Figure 7 shows the overview of experimental environment.

- **Ticketing Server.** We implemented the Internet ticketing service on a Servlet-based web server. Here we regard the server as one of the token devices on **D.** Our implemented servlet can switch a HTTP/1.1 session to a TRN session with Switching-Protocol ("Protocol-Upgrade: TRN").
- **Ticket Wallet.** Initially, we developed a personal ticket wallet on the small Java device, named PFU BossaNova™ [9]. Although BossaNova has no proper tamper resistances for its memory, it supports GUI-based touch panel and IrDA communication. After the initial implementation, we developed more secure

[25] JIPDEC (Japan Information Processing DEvelopment Center) is an EC promotion institute sponsored by Japanese government. Available at http://www.jipdec.or.jp/.

wallet using a tamper proof security box that supports elliptic curve cryptographic engine and ISO14443 based communication capability. (This security box is originally developed as an emulation test-bed of contact-less smartcard.) Both types of wallet can connect to a PC through a R/W unit.

– **Ticket browser.** The token devices are by nature passive. To browse and operate a personal wallet device, we used a PC as a user interface. For example, when you want to buy a ticket, you first connect the server with a HTTP request, and then turn to the TRN mode. At the same time, you start polling your wallet on the end of R/W unit. After that, the PC exchange authentication and succeeding encrypted tokens over TRN. From viewpoints of TRN, the server and the wallet never trust mediated PC.

5.2 Summary

More than 100 persons tested to use our prototype system for purchasing tickets, exchanging them, and checking their entrance to the stadium electronically. The size of messages to transfer n byte-length token is estimated as follows: sent size (copy and commit) is about $n + 40$ bytes; received size (copied and committed) is 48 bytes. Moreover, the required size of logging field in each transfer is *value* (1 byte) and *from-to* (4 bytes) with a token object itself. The logging size is small enough to implement the TRN broker on the limited recourses.

6. CONCLUSION

The electronic transferability of tokens is a double-edged sword for organizations that conduct electronic business on the Internet. The main contribution of Tamper Resistance Network (TRN) is to provide us with a secure distribution environment that allows us to move tokens electronically without risk of duplicating or destroying them. The control of atomicity and the trace of movement are provided to enhance the security of trading between smartcards or other tamper-proof device.

This paper addressed the experimental prototype for the distribution of electronic tickets on TRN. The finding of the experimentation is that the TRN framework is useful to setup online business selling electronic tickets through the Web. Future directions we will investigate include additional security evaluation, fair-exchange, and PKI design based on business models.

7. REFERENCE

1. C. Adams and S. Lloyd. Understanding the Public-Key Infrastructure, Macmillan Technology Series, 1999.
2. S. Araki. The Memory Stick. IEEE Micro, pages40-46, July-August 2000.
3. P. Bernstein and E. Newcomer. Principles of Transac-tion Processing. Morgan Kaufmann Publishers, 1997.
4. W. Diffie and M Hellman. New Direction in Cryptog-raphy, IEEE Trans. on Information Theory, pages 644-654, IT-22, 6, 1976.
5. U. Hansmann, M. S. Nicklous, T. Schack. F. Seliger. Smart Card Application Development Using Java. Springer, 2000.
6. S. Kent and R. Atkinson. Security Architecture for the Internet Protocol. IETF RFC 2401, Nov. 1998. Avail-able at http://www.rfc-editor.org/rfc/rfc2401.txt
7. K. Kuramitsu et al. TTP: Secure ACID Transfer Protocol for Electronic Ticket between Personal Tamper-Proof Devices. In *Proceedings of the 24th Annual International Computer Software & Applications Conference (IEEE COMPSAC2000),* Oct. 2000.
8. K. Kuramitsu and K. Sakamura. PCO: EC Content Description Language Supporting Distributed Schema across the Internet. Journal of IPSJ, pages 110-122. Vol. 41 No.1, Jan.2000.
9. PFU BossaNova Homepage, available at http://www.pfu.co.jp/BossaNova/.
10. W. Townsley et al., Layer Two Tunneling Protocol (L2TP), IETF RFC 2661, Aug. 1999. available at http://www.rfc-editor.org/rfc/rfc2661.txt
11. J.D. Tygar. Atomicity in Electronic Commerce. In Proceeding of the 5th Annual ACM Symposium on Principles of Distributed Computing. May 1996.
12. Matsuyama and K. Fujimura. Distributed Digital-Ticket Management for Rights Trading System, In Proceedings of the ACM EC'99, November 1999.
13. Mondex Electronic Cash. Available at http://www.mondex.com/