Digital-Ticket-Controlled Digital Ticket Circulation

Ko Fujimura, Hiroshi Kuno, Masayuki Terada,
Kazuo Matsuyama, Yasunao Mizuno, and Jun Sekine

NTT Information Sharing Platform Laboratories
{fujimura, kuno, terada, matsuyama, mizuno, sekine}@isl.ntt.co.jp

Abstract

This paper presents a new digital-ticket circulating scheme and trust management scheme for a digital ticket. A digital ticket is a digital medium that guarantees certain rights of the owner and it includes software licenses, resource access tickets, event tickets, and plane tickets.

The circulation of digital tickets comprises three types of principal transactions: issuance, transfer, and redemption. Depending on the application, various conditions must be satisfied to execute these transactions, e.g., only qualified shops can issue the tickets and only a certain agent can transfer the tickets. This paper introduces circulation control tickets, which are required to issue, transfer, redeem a ticket, and proposes specifying the required control ticket types in the ticket to be circulated itself using the Generalized Ticket Definition Language. The ticket circulating system issues, transfers, or redeems a ticket only if the control tickets are owned by the participants of the transaction. The circulation control tickets themselves can be any type of digital ticket, e.g., a driver's license or a membership certificate to certain group, and these tickets can be recursively circulated in the ticket circulating system. This scheme provides the ticket circulating system with both the flexibility needed to match the business scheme of interest and application independence.

This paper also proposes a ticket-type-based trust management scheme that enables users to mechanically verify the trust of a ticket by the presented ticket type verification procedure.

1. Introduction

A digital ticket is a digital medium that guarantees certain rights of the owner and it includes software licenses, resource access tickets, event tickets, and plane tickets. A digital ticket covers a wide range of digital rights from a digital certificate [8][20], in which transferability is not required and where there are no restrictions on the number of times it can be consumed, to digital cash [2][14], in which transferability is required and restrictions on consumption apply.

We are developing a system that can circulate all tickets with various rights in a common manner. This system enables service providers to reduce the development costs of the ticketing software/hardware and also enables users to view and manage various tickets using a common "ticket wallet", which greatly improves usability.

The trust management scheme [1][3][6][7] developed for digital certificates and the double-spending protection scheme [2][14] developed for digital cash can also be applied to digital tickets as base technologies, since digital tickets have aspects of both digital certificates and digital cash. These technologies, however, do not provide solutions for the specific requirements of digital tickets, i.e., diversity in circulating requirements and business schemes.

To circulate various types of digital tickets using a common ticket processing system, we proposed a general-purpose digital ticket framework, in which a ticket is circulated by interpreting the ticket properties of anonymity, transferability, and divisibility, specified in the ticket itself using the Generalized Ticket Definition Language [9]. No circulation control scheme or trust management scheme for digital tickets have, however, been presented up to now. These issues are especially important since the requirements depend on the ticket's business scheme, and this makes generalization difficult. This paper focuses on these issues and presents a new approach that we implemented in a prototype system.

The following were taken as our design goals:

No centralized organization. Designs that rely on deference to a global, centralized organization should be avoided since a single failure in the organization may impair the entire system. No centralized broker who sells all types of tickets, or centralized authority that authenticates all issuers or other participants, should be assumed.

Business scheme flexibility. Unlike digital cash, various requirements must be satisfied when a ticket is circulated depending on the application. Examples include "only qualified agents can transfer the tickets", or "only a certain member of a group can redeem the tickets". To satisfy these business requirements, flexible control of ticket circulation is required.

Management autonomy. Responsibility for the ticket and for settlement when a task or service is not satisfactorily rendered should be application dependent. These management policies should be defined freely by the ticket issuer. For example, a ticket issuer should ask a certificate authority (CA) to endorse the contents of their tickets only if the issuer desires it.

Trust manageability. Diverse types of digital tickets will be circulated in the future. This makes it difficult for users to judge whether a ticket can be trusted or not. To support such judgement, a trust management system that mechanically verifies the trust of a ticket is thus required.

Simplicity. Simplicity is important in understanding the system, which is necessary for people to trust the system. It also minimizes the probability of a security hole resulting from an implementation error.

Prevention of duplicated redemption is also an important issue of the digital-ticket circulation system. This exceeds the scope of this paper since several double-spending protection schemes invented for digital cash can be applied to digital tickets. The digital ticket storage system, i.e., how digital tickets are stored in smart cards, PCs, or network, is also a major issue when implementing a digital-ticket circulation system. This point is also beyond the scope of this paper. We will address this issue in other papers.

To achieve these design goals, this paper introduces circulation control tickets, which are required for issuing, transferring, and redeeming a ticket, and specifies, using the Generalized Ticket Definition Language, the required control ticket types in the ticket to be circulated. The ticket circulating system issues, transfers, or redeems a ticket only if the control tickets are owned by the participants in the transaction. The circulation control tickets themselves can be any type of digital ticket, e.g., a driver's license or certain membership certificate, or other certificates issued by a certificate authority (CA), and these tickets can be recursively circulated using the ticket circulating system. This scheme provides flexibility to the ticket circulation schemes since various conditions can be defined. This scheme also provides management autonomy since any type of certificate can be used for endorsing the tickets without introducing complexity or a centralized organization.

This paper also proposes a new trust management scheme based on ticket type; the scheme defines the format and restrictions placed on ticket properties. In this scheme, the trust of a ticket can be verified mechanically by the proposed ticket type verification procedure, which checks whether the ticket meets the corresponding ticket type definition managed by the user.

The following section presents a ticket circulation model. The third section describes the circulation control scheme in detail. We then discuss the trust management scheme that is the basis of security in Section 4. Section 5 overviews an implementation of the ticket circulation system. Finally, we draw a comparison to related work in Section 6.

2. Ticket Circulation Model

This section presents the basic ticket circulation model assumed herein. The design goal of *no centralized organization* is the only one related to the basic model and we address it in this section. Approaches to the other design goals are described in Section 3 and 4.

2.1 Participants

The participants in our ticket circulation model and the assumed ticket flow are shown in Figure 1. There are three types of participants in the ticket circulation model: an *issuer* creates, signs, and issues a ticket; a *user* redeems the ticket; and a *service provider* fulfills the service or task represented by the ticket. The issuer and service provider can be the same physical organization. Additionally, a *shop*, *broker*, or other participants exist in real paper ticket circulation but they are not included in the settlement because they are treated as users who buy tickets from issuers (or users) and transfer the tickets to other users with payment.

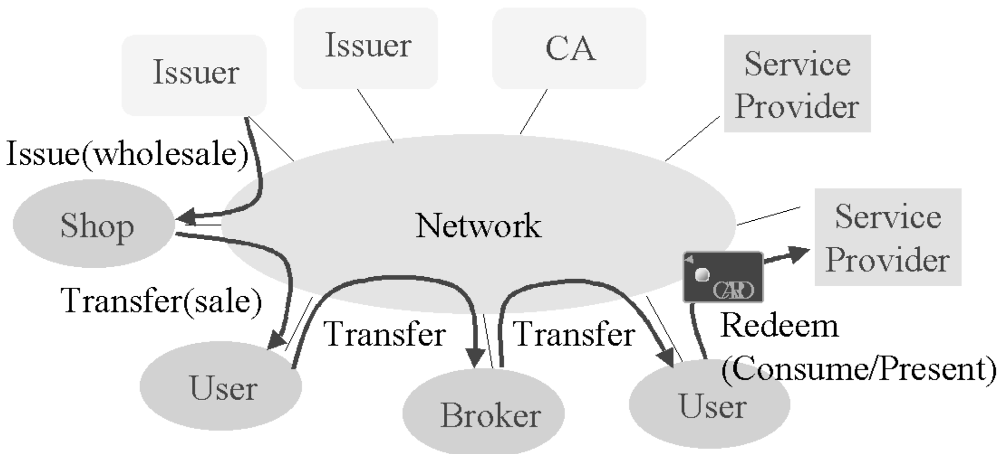

Figure 1. Ticket circulation model

2.2 Digital ticket

In this paper, a *digital ticket* (or *ticket*) is defined as $\text{Signed}_I(I, P, O)$, where I is the ticket issuer, O is the ticket owner, and P is a *promise* to the ticket owner. The phrase "Signed_I" means that the entire block is signed by the issuer's digital signature. Promise P has several sub-properties that represent various rights depending on the application.

2.3 Transaction

Circulation of a digital ticket comprises three types of principal transactions:

Issuance is an action in which issuer I gives ownership of ticket T to user U. In our model, we assume that this transaction is implemented by issuer I creating ticket $T = \text{Signed}_I(I, P, U)$ and sending it to user U.

Transfer is an action in which user U_0 transfers ownership of ticket T to user U_1. In our model, we assume that this transaction is implemented by attaching a *transfer certificate* to the ticket to be transferred, i.e., user U_0 creates transfer certificate $T_{t1} = \text{Signed}_{U0}(U_0, \text{transfer}(T), U_1)$ and sends it to user U_1 with T, where transfer(T) is a promise that T was transferred.

Redemption is an action in which user U redeems the rights represented by ticket T to service provider S. In our model, we assume that this transaction is implemented by attaching a *redeem certificate* to the ticket, i.e., user U creates redeem certificate $T_r = \text{Signed}_U(U, \text{redeem}(T), S)$ and sends it to service provider S with T, where redeem(T) is a promise that T was redeemed. The situation in which ownership of the ticket is retained when the ticket is redeemed, e.g., redemption of licenses or passports, is termed *presentation*. The situation in which ownership of the ticket is voided or the number of times it is valid is reduced when the ticket is redeemed, e.g., redemption of event tickets or telephone cards is termed *consumption*.

Assume that a ticket was circulated between participants $I \to U_0 \to U_1 \to ... \to U_n \to S$, using the transactions of issuance, transfer, and redemption. A set of tickets T, $T_{t1}, ..., T_{tn}$, T_r, called a *transfer list*, is sent to the service provider as a result of the circulation. We use the transfer list to detect who transferred or redeemed a ticket more than once after fraud is detected. Our prototype system also offers an offline fraud prevention scheme using a smart card, but this scheme is beyond the scope of this paper.

Generally speaking, money, services, or products are circulated against the flow of the tickets. There are issues on how the atomicity between these two delivery flows is guaranteed [6]. This paper, however, focuses only on the ticket flow and makes payment methods independent because this approach enables easy integration with legacy application systems that use existing payment systems. Of course, integration with payment methods is an important issue to be studied.

3. Circulation Control Scheme

In this section, we present an approach to satisfy the design goals described in Section 1: *business scheme flexibility*, *management autonomy*, and *simplicity*.

3.1 Requirements

Business scheme flexibility is realized by establishing a general circulation control scheme that can handle various circulation requirements some of which are described below:

(1) Only qualified shops can issue tickets

(2) Tickets may be circulated only between the registered members of a group

(3) Only certain agents are allowed to transfer tickets, the general public is prohibited from transferring tickets to anybody else, e.g., plane tickets and event tickets

(4) Only qualified people can redeem tickets, e.g., student discount tickets

(5) Only a certain shop or agent is allowed to examine (punch) the ticket. Most tickets have such a condition

Management autonomy can not be achieved if the circulation system forces every participant to meet specific requirements, e.g., each participant must be registered with a specific CA. In this case, the management policies would be fixed by the CA's certificate policy. Which certificates are required for a transactions should depend only on the application. The management policy of an application must be application (or ticket) specific and not circulation-system specific.

3.2 Onion ticket accumulation model

We found that the above requirements can be satisfied by checking if the participants of a transactions have certificates confirming their qualifications before conducting the transaction. To represent these qualifications, we introduce *circulation control tickets*. The circulation requirements are specified by what type of circulation control tickets are required for each type of ticket to be circulated, and separate requirements are specified for the sender and receiver. The ticket circulating system issues, transfers, or redeems a ticket only if all circulation requirements are satisfied. The circulation control tickets themselves can be any type of digital ticket, e.g., a driver's license, certain membership certificate, or other certificates issued by a CA, and these tickets can be recursively circulated using the ticket circulating system. We call this scheme the *onion ticket accumulation model* (Figure 2).

In this model, the identity of a participant forms the onion core, and the rights (tickets) given to the identity form the layers beyond the core. A transaction is conducted between two onions (participants). The model illustrates that an outer layer (tickets) cannot be peeled or moved to another onion unless inner layers (tickets) exist.

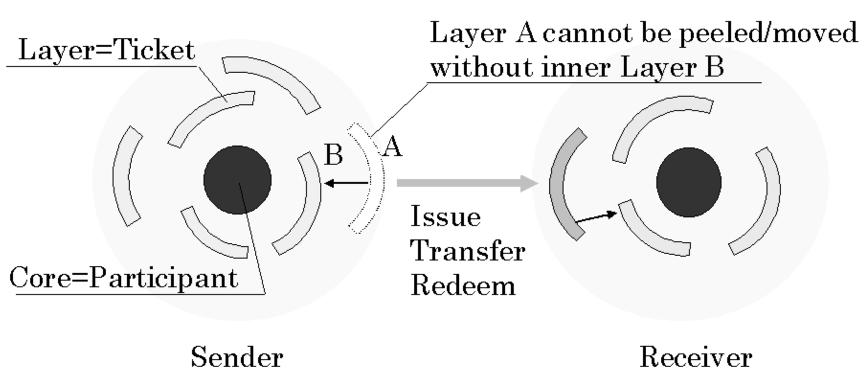

Figure 2. Onion Ticket Accumulation Model

Transactions	Sender conditions	Receiver conditions
Issue	Airline certificate	(None)
Transfer	Travel agent certificate	(None)
Redeem	(None)	Airline certificate

Table 1. Example of Circulation Condition Definition

For example, a plane ticket can be modeled using circulation conditions as shown in Table 1. A plane ticket can be issued and punched by the airlines that have an airline certificate issued by the International Air Transport Association (IATA). A plane ticket can be transferred by the travel agencies who have a travel agent certificate issued by a government, while the public is prohibited from transferring the tickets to anybody else.

The next issue is how and where these circulation requirements are defined in the system. We describe this issue in the following sections.

3.3 Ticket type definition

To make the ticket-circulation system application independent, we propose specifying the circulation requirements for a ticket in the ticket itself using the Generalized Ticket Definition Language. However, efficiently and securely specifying the “type” of control tickets, e.g., airline certificates or travel agent certificates, in the circulation requirements is still an issue.

We found that there are two classes of information in a ticket. One is common to each type of ticket. The other is different for each ticket instance. For example, the circulation requirements presented in Section 3.2 are an instance of the former class of information and do not have to be defined for each ticket. We, therefore, introduce two types of definitions: a *ticket type definition* and *ticket definition*. The common information within tickets of the same ticket type is defined in the ticket type definition. The specific information on the ticket instance is defined in the ticket definition. To bind a ticket definition to its ticket type definition securely, we propose setting the hash value of the ticket type definition in the ticket definition, which is digitally signed by the ticket issuer. This scheme reduces communication cost and improves efficiency of ticket circulation since the ticket type definition can be pre-distributed to the participants who may use the tickets of a particular ticket type. Details of the distribution of the ticket type definitions are described in the following section.

A *ticket type definition* is the following tuple:

TypeInfo: The ticket type definition information. It includes the contact address of the person defining the ticket type, the name of the ticket type, version number, or other information.

TypeValidity: The validity condition of the ticket type. The valid period, start date and end date, are specified.

IssueConditions: Pre-conditions of the issue transaction. This is a tuple of *SenderConditions* and *ReceiverConditions* described below.

TransferConditions: Pre-conditions of the transfer requirement. This is a tuple of *SenderConditions* and *ReceiverConditions* described below.

RedeemConditions: Pre-conditions of the redeem transaction. This is a tuple of *SenderConditions* and *ReceiverConditions* described below.

TicketSchema: Syntax definition for the Promise property of the tickets of this ticket type. This defines sub-properties of the *Promise* property and the necessity value, i.e., mandatory or optional, and default value if any. There are several specifications for this purpose. XML DCD [16] or SOX [19] can be used [12]. Details of the definition method are thus outside the scope of this paper.

SenderConditions and *ReceiverConditions* are one of the following:

(1) $TypeID_1 \ldots TypeID_n$: A list of the ticket type identifiers that must be held by the sender or receiver, where $TypeID_X$ = hash(*the ticket type definition X*).

(2) *Identity*$_X$: The identity of the sender or receiver X. It can be implemented using a public key or other names with the scope rule but this paper assumes *Identity*$_X$ = hash (PK_X) for simplicity.

(3) *Not specified*: This means no restrictions apply to the sender or receiver.

Condition (1) specifies what types of tickets the participants must possess and condition (2) specifies the participants directly. Conditions are application dependent and specified by the business scheme or legislation.

In the plane ticket example shown in Section 3.2, an airline certificate is specified as the sender condition within the issue conditions. This is an example of condition (1) above. This condition can be defined by specifying the ticket type identifier of the airline certificate in the type definition of the plane ticket. This scheme enables new airlines to issue plane tickets after getting their airline certificates from IATA. It is not necessary to redistribute ticket type definition to users in this case. As an example of condition (2) above, we assume a plane ticket, the type of which is defined by each airline and issued without IATA involvement. In this case, the sender conditions within the issue conditions can be defined by specifying the airline's identity in the type definition of the plane ticket. This scheme does not require new airlines to get their airline certificate from IATA but they must distribute the ticket type definition to the users.

There are several ways in which sender and receiver conditions can be specified other than (1) and (2) above; disjunction, conjunction, and threshold [3][8] of the ticket types or ticket instances, restrictions on property values of a ticket, etc. can be used. Our analysis of paper tickets shows that implementing conditions (1) and (2) achieves sufficient flexibility for describing the conditions of most tickets.

3.4 Ticket definition

Basically a ticket has the structure Signed$_I$ (I, P, O) as described in Section 2. To establish a binding between a ticket and its ticket type, we introduce a *ticket type identifier* in the ticket definition. Additionally, two other properties, *ticket instance identifier* and *ticket validity*, are also introduced for implementation practicality. The ticket instance identifier is useful for efficiently detecting duplicate redemption.

A *ticket definition* is the following tuple:

TypeID: The ticket type identifier.

TicketID: The ticket instance identifier. It must be unique for each ticket with the same TypeID.

TicketValidity: The validity conditions of the ticket. At least the valid period, start date and end date, are specified.

IssuerID: The ticket issuer's identity.

Promise: Various properties are specified depending on the application.

OwnerID: The ticket owner's identity.

Signature: Issuer's digital signature on the above.

4. Trust Management Scheme

In this section, we present an approach to satisfy the design goal of *trust manageability* described in Section 1.

4.1 Requirements

The proposed ticket circulation system enables various tickets to be issued very easily by using Generalized Ticket Definition Language based definitions without developing software for ticketing and ticket examination servers. As a result, a wide variety of digital tickets may be circulated freely. This makes it difficult for users to judge whether a ticket can be trusted or not. For example, even if the digital signature of the ticket is valid, this is no guarantee of the rights described in the ticket since it might be signed by a person who has no right to issue the ticket. There are two approaches to preventing or detecting forgery:

Trusted broker: A ticket bought directly from a trusted broker can be trusted even if the ticket issuer cannot be trusted. In this scheme, however, several problems exist: no offline capability, centralized organization, and additional payments to the broker.

Authorization by a CA: If a CA who authorizes all issuers for all types of tickets exists, users can ask the CA if the ticket can be trusted. This scheme, unfortunately, destroys management autonomy. If a different CA exists for each specific type of ticket, management autonomy might be achieved but it is difficult for users to find a CA who can judge the ticket. No practical way of managing the binding between a specific type of ticket and its CA has be proposed yet.

The trust management scheme for digital tickets should, therefore, provides some means of verifying if a ticket can be trusted regardless of its circulation route. It also should provide a tool for managing the basis of trust, which might differ with the ticket type.

4.2 Ticket type based trust management

To achieve the above requirements, this paper proposes a new trust management scheme based on the ticket type. In this scheme, once a user establishes the binding between the conceptual rights in the real world and the ticket type definition (or its identifier), the user can mechanically verify the trust of the same type tickets.

We assume that a user establishes a binding between conceptual rights recognized in the real world and a ticket type identifier by some means. We use this binding as the basis of trust desired by the user. Examples of binding are as follows:

Conceptual rights	Ticket type identifier
Plane ticket	F796452E753FFDEE4379BB2C883C2CAA
Lottery ticket	AAE1DC379BB2C883C2CAAF796452E753
Ticket for car wash	85579BB2C883C2CAAF796452E7536651

This scheme similar to PGP [15] in which a set of bindings between user IDs and public keys (or fingerprints) is the basis of trust.

There are several ways to distribute the ticket type identifiers. For example:

- CD-ROMs sold in bookshops.
- Physical ticketing machines managed by service providers.
- A trusted web site with a secure communication channel.

To manage ticket types, which are key elements of trust, this paper introduces a *ticket type book*. A ticket type book is managed by each user and it stores information about the ticket types trusted by the user. In the ticket type book, a type is managed as the tuple (*TypeName*, *TypeID*, *TicketTypeDefinition*). When a new type is given, the tuple is stored in the ticket type book only if it is *trusted* by the user. A *TypeName* represents conceptual rights recognized by the user and any name can be replaced as he/she likes.

Assuming that a user has the ticket type book described above, the user can use the following type verification procedure to verify if the issuer has the right to issue the ticket:

Ticket type verification procedure: Let *T* be a ticket definition and *TT* be a ticket type definition. We say that *T meets TT* if and only if the following procedure is completed.

(1) Verify the digital signature of *T*.

(2) Verify that the ticket type identifier *TypeID* in *T* and hash(*TT*) are the same.

(3) Verify that the ticket type identifier *TypeID* is in the ticket type book.

(4) Retrieve *TT* from the ticket type book.

(5) If identity $Identity_X$ (onion core) is specified as the issuer conditions in *TT*, verify that $Identity_X$ and the issuer identity *IssuerID in T* are the same.

(6) If ticket type identifiers $TypeID_1 \ldots TypeID_n$ (onion layers) are specified as the issuer conditions in *TT*, verify that the issuer owns all tickets $T_1 \ldots T_n$ corresponding to $TypeID_1 \ldots TypeID_n$ (getting $T_1 \ldots T_n$ from the issuer directly or from the circulation history attached to the transferred ticket, and verify that *OwnerIDs* in $T_1 \ldots T_n$ and *IssuerID* in *T* are the same), and verify that for each ticket definition $T_1 \ldots T_n$ *meet* the corresponding type definitions $TT_1 \ldots TT_n$ by applying this procedure recursively.

Note that verification of *TicketValidity* in *T* and *TypeValidity* in *TT*, and verification that the *Promise* defined in *T* meets the *TicketSchema* in *TT* are not included in the above procedure for readability. The verification of the series of transfer certificates, which must be verified for a transferred ticket, is also omitted for readability.

This trust management scheme enables users to detect a forged ticket regardless of the circulation route, and can be performed mechanically by the proposed type verification procedure. This scheme also provides easy management of the basis of trust since the user need manage only one type definition for each type of ticket, even if complicated issuer conditions are set.

5. Overview of Implementation

5.1 Generalized ticket definition language

We proposed to implement the Generalized Ticket Definition Language on top of RDF [18], which, in our previous paper [9], is layered on XML [17]. Our current implementation, however, uses XML directly [12]. The reason for this is because RDF is too rich and redundant to describe circulation conditions or other properties for controlling ticket circulation, which are common to all tickets, and the semantics are not important except with regard to the Promise property.

The standardization of XML signed documents [5] is now being actively discussed in W3C and IETF. Our current specification does not comply with these specifications but integration with any future standard is an important issue to be studied.

5.2 Protocols

The ticket circulation protocols are overviewed in Figure 3. First, a definition for the ticket to be circulated is sent to the receiver in order to send the circulation conditions. In this phase, ownership is not transferred. Second, the receiver checks if the ticket type of the ticket definition is in the receiver's ticket type book. If not, the receiver obtains the type definition in some way, e.g., from the sender or network. Third, the receiver and sender check if the circulation condition is satisfied. In this phase, tickets that must be owned by the sender or receiver are checked against each other. Finally, ownership is transferred from sender to receiver by sending a transfer or redemption certificate.

We have implemented a prototype of the above protocols using Java and confirmed its feasibility. We also implemented three common components on top of the protocol handling system: ticketing server, ticket examination server, and ticket wallet.

5.3 Application example

The GUI of the ticket wallet is shown in Figure 4. In this example, four types of tickets are issued and stored in the ticket wallet after conducting several transactions in the following scenario: First, a wallet certificate was issued when a user installed his/her ticket wallet. Next, a customer certificate for a loyalty program was issued when he/she registered at the shop. After several points were earned at the shop, an award ticket, which can be exchanged for a prize, was issued with the redemption of some of the earned points.

Based on the onion ticket accumulation model, several requirements can be given in this example such as; the customer certificate requires users to have a wallet certificate and award ticket transfer requires users to have the customer certificate. Such flexible circulation control can be achieved without any application-specific software in the ticket wallet.

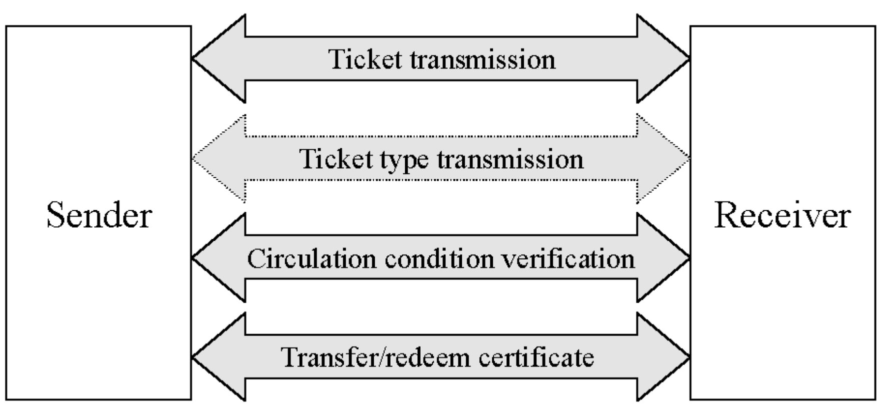

Figure 3. Ticket Circulation Protocol Overview

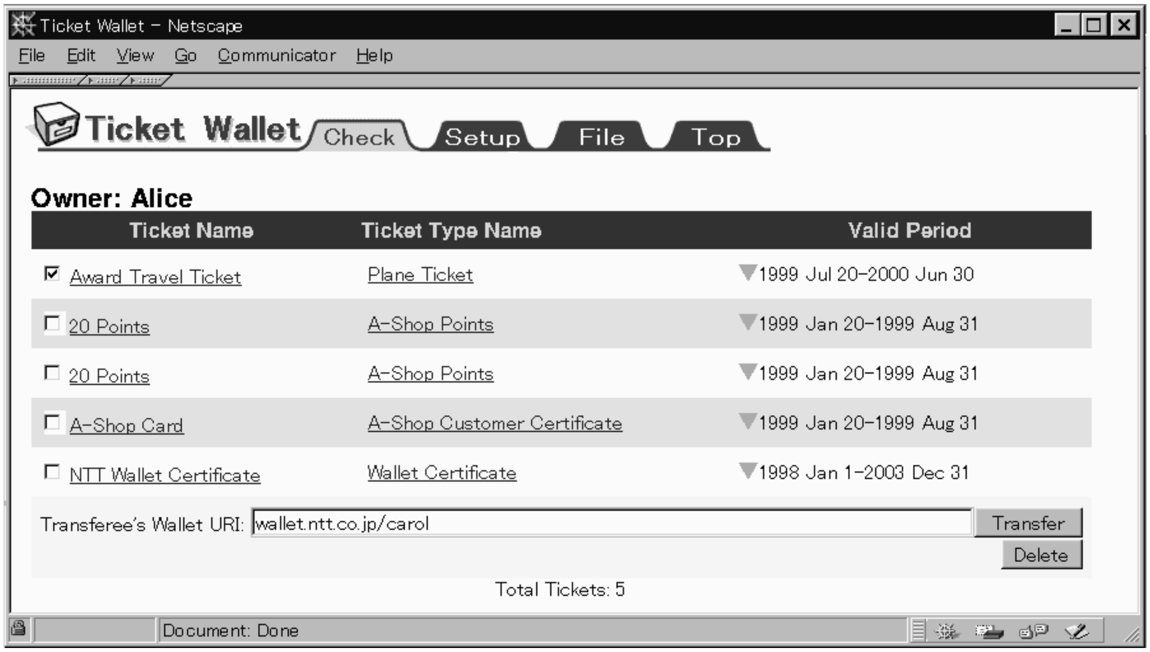

Figure 4. Ticket wallet displayed in a Web browser

6. Related Work

KeyNote [3][4] presents an authorization control system that determines whether actions are consistent with policies, which are a collection of certificates called assertions. We present here a ticket circulation system that determines whether issuance, transfer, and redemption requirements are consistent with the sender or receiver qualification conditions. In this sense, our approach is similar to but more specific than KeyNote and is focused on ticket circulation. The KeyNote system uses a general programmable language to define assertions, whereas our system uses a ticket type based specific language to allow high-level control descriptions.

SPKI [8] and PGPticket [11] present an authorization control system that provides a mechanism for deriving authorization decisions from a collection of certificates. They provide the ability to delegate fine-grained authorization from one person to another. These systems, however, do not introduce certificate transferability, in which the transferor loses the rights when the certificate is transferred. As a result, it is difficult to realize event tickets or other tickets that can be consumed only once, although we note that it can be applied to license or pass-type tickets.

The Eternal Resource Locator [1] presents a scheme to establish trust without relying on any PKI. This scheme uses the hash value of the root hypertext document as the basis of trust. Our scheme has similarities to this in that the hash value of the type definition is used as the basis of trust.

NetBill [6] presents an authorization scheme, in which a customer's authority is represented by an identity ticket, which is a pseudonym of the customer, and one or more *credentials*, each of which represents proof of group membership. This model has some similarities to our onion ticket accumulation model. The NetBill system introduced this model mainly for flexible price control, whereas our model is used to increase the flexibility of circulation control.

Capability cards [13] represent a digital media that can circulate various types of digital objects including digital rights using a card metaphor. However, they do not provide flexible circulation control or trust management, both of which are offered by the schemes proposed in this paper.

7. Conclusion

This paper described issues and design goals for a generalized ticket circulation system that can circulate the various types of digital tickets required in diverse business schemes. To enable flexible control of ticket circulation, we specify circulation control tickets. Before issuing, transferring or redeeming a ticket, the sender or receiver must have the appropriate circulation control tickets. We define a ticket in two parts, ticket type and individual ticket information, and use the type identifier to specify the circulation control tickets required. These schemes make it possible for any type of ticket, e.g., driver's license or social security certificate, to be used as the PKI for ticket circulation in addition to identity certificates issued by a CA. We also proposed a new trust management scheme based on the trust of ticket type definitions. This scheme enables users to mechanically verify the trust of a ticket by executing the proposed ticket type verification procedure.

Acknowledgement

The authors wish to thank Yoshiaki Nakajima, Yoshihito Oshima, Masayuki Hanadate, Nobuyuki Chiwata, Tomoji Takehisa, and Takaaki Matsumoto for useful comments and discussions.

References

[1] R. J. Anderson and V. Matyáš Jr., "The Eternal Resource Locator: An Alternative Means of Establishing Trust on the World Wide Web", *3rd USENIX Workshop on Electronic Commerce*, August 1998, pp. 141-153.

[2] N. Asokan, P. A. Janson, Michael Steiner, and M. Waidner, "The State of the Art in Electronic Payment Systems", *IEEE Computer*, September 1997, pp. 28-35.

[3] M. Blaze, J. Feigenbaum, and M. Strauss, "Compliance Checking in the PolicyMaker Trust-Management System", In *Proceedings of Financial Cryptography '98*, LNCS 1465, pp. 254-274.

[4] M. Blaze, J. Feigenbaum, A. D. Keromytis, and J. Ioannidis, "The KeyNote Trust-Management System", IETF Internet Draft, August 1998.

[5] R. D. Brown, "Digital Signatures for XML" IETF Internet Draft, January 1999.

[6] B. Cox, D. Tyger, and M. Sirbu, "NetBill Security and Transaction Protocol", *1st USENIX Workshop on Electronic Commerce*, July 1995.

[7] C. M. Ellison, "Establishing Identity Without Certification Authorities", *6th USENIX Security Symposium*, July 1996.

[8] C. M. Ellison, B. Frantz, B. Lampson, R. Rivest, B. M. Thomas, and T. Ylonen, "SPKI Certificate Theory", IETF Internet Draft, November 1998.

[9] K. Fujimura and Y. Nakajima, "General-purpose Digital Ticket Framework", *3rd USENIX Workshop on Electronic Commerce*, August 1998, pp. 177-186.

[10] S. Glassman, M. Manasse, M. Abadi, P. Gauthier, and P. Sobalvarro, "The Millicent Protocol for Inexpensive Electronic Commerce," In *Proceedings of WWW4*, December 1995.

[11] V. Moscaritolo and A. N. Mione, "PGPticket", IETF Internet Draft, November 1998.

[12] Y. Nakajima and K. Fujimura, "XML Ticket Model and Syntax Specification," IETF Internet Draft, to appear.

[13] K. Otani, H. Sugano, and M. Mitsuoka, "Capability Card: An Attribute Certificate in XML", IETF Internet Draft, November 1998.

[14] Peter Wayner, "Digital Cash", Academic Press Ltd., 1997.

[15] P. Zimmermann, The Official PGP User's Guide, MIT Press, 1995.

[16] Document Content Description for XML (DCD), The World Wide Web Consortium, Note, 1998, http://www.w3.org/TR/NOTE-dcd

[17] Extensible Markup Language (XML) 1.0, The World Wide Web Consortium, Recommendation, 1998, http://www.w3.org/TR/REC-xml

[18] Resource Description Framework (RDF) Model and Syntax Specification, The World Wide Web Consortium, Recommendation, 1998, http://www.w3.org/TR/REC-rdf-syntax/

[19] Schema for Object-oriented XML (SOX), Note, 1998, http://www.w3.org/TR/NOTE-SOX/

[20] ISO/IEC 9594-8 (X.509), Information Technology - Open System Interconnection - The Directory: Authentication Framework