Relational-based Trust Management in a Generic Payment System

Lamia Chaffai-Sghaier[1] and Sihem Guemara-El Fatmi

SUPCOM, University of Carthage,2083 Tunisia

Abstract. Trust Management represents a vital component for the protection of business transactions. This paper considers the application of a relational-based model for Trust Management in Electronic Payment Systems. We introduce a generic payment model that provides a good framework to validate our trust model. We use the special features, new extensions and relational techniques provided by the Trust Management model to specify entities, actions and security policy axioms and rules in the generic payment system. We also discuss compliance correctness issues such as security policy specification correctness and validation, certificate chain discovery and revocation as well as performance issues. Finally, we consider some implementation issues.

1 Introduction

The main issues in access control of distributed networks are authentication, authorization and enforcement. Identification of principals is handled by authentication. Authorization answers the question: should a request r submitted by a principal K be allowed? Enforcement addresses the problem of implementing the authorization during an execution. Trust Management (TM) systems resolve these issues by defining languages for authorizations and access control policies, and by providing a TM engine for deciding when a request is authorized.

In [1], Boudriga and al. present a TM model which brings the novelty of using relational techniques for authentication and compliance checking processes. The model is based on (1) a relational language allowing constructs for the description of TM entities such as actions, certificates, requests and security policies, (2) a relational calculus for performing proofs, verifying features, and computing systems answers,(3) a mechanism for identifying entities requesting to perform actions based on public key certificates and (4) a compliance engine, which provides a service to applications for determining how an action that is requested by a principal can be granted based on relational certificates chaining.

Electronic commerce services, which basically rely on Payment Systems (PS) and use public-key cryptography on a mass-market scale, require sophisticated mechanisms for managing trust. Furthermore, authorization constitutes the most important relationship in a PS. In this paper, we introduce a generic payment model to demonstrate the possible generalization of our TM model to all payment schemes and protocols.

The paper is organized as follows. Section 2 provides an overview of payment systems' characteristics and security requirements. Section 3 introduces a generic payment

Chaffai Sghaier L. and Guemara-El Fatmi S. (2004).
Relational-based Trust Management in a Generic Payment System.
In *Proceedings of the 1st International Workshop on Electronic Government and Commerce: Design, Modeling, Analysis and Security*, pages 116-125
DOI: 10.5220/0001402801160125

model providing an abstraction of payment services. Section 4 presents the TM model components and new features relational compliance check processes. In section 5, we apply the TM model to the generic payment model to demonstrate its flexibility and its capacity to encompass all types of PSs. Section 6, addresses the issues of the security policy specification correctness and compliance correctness. Section 7, presents an implementation methodology. Section 8 concludes this paper.

2 Electronic Payment Systems

2.1 Models and characteristics

Commerce always involves a customer and a merchant who exchange money for goods or services and at least one financial institution (i.e. bank). PSs can be classified based on the following features [2]: *(a) Pre-paid vs post-paid,(b) Account based vs token based*, (c) *On-line vs off-line,(d) Macropayments vs micropayments,(e) Deterministic vs probabilistic.*

2.2 Security properties and requirements

In general, PSs must exhibit the following security properties[2]: *authentication, integrity of data*, *confidentiality of information, non-repudiation, availability and reliability (atomicity)*. Additional properties like *anonymity* and *multiple-spending prevention* are sometimes required (i.e. cash-like systems*)*. Moreover, PSs should fulfill mandatory security requirements [3] (1)Bank requirements: proof of transaction authorization by customer and by merchant, (2)Customer requirements: proof of transaction authorization by bank, authentication and certification of merchant and payment receipt from merchant,(3)Merchant requirements: proof of transaction authorization by bank and by customer. All these security properties and requirements can be achieved by using Public-Key Cryptographic tools. While confidentiality is attained by enciphering each message, using a private key known only to the sender and recipient, the authenticity features are attained via *key management* carried out by a *certification authority.* Anonymity is attained via blind-signature and zero-knowledge proofs.

3 Generic payment model

In this section, we describe the architecture of the generic payment model and its registration, payment, and electronic token purchase protocols. Players are the customer, the merchant and the bank. The bank plays several roles such as issuer, acquirer, registration (RA) and certification (CA) authority. Customer and merchant have accounts in the bank. We use the concept of a wallet linked to a single account and supporting multiple PSs [5]and we adopt the GPSF [4] primitive value transfer services constituting the core of any PS.

3.1 Registration Procedure

Before issuing a name certificate, the bank's RA may perform an *in-person authentication* and ask the customer/merchant to provide physical credentials. We assume that upon receiving the required credentials, the RA provides the customer/merchant with an *authenticator* that will be included in the certificate request. The customer/merchant generates a key pair and sends a certificate request to the CA using the authenticator. The CA generates and delivers the certificate directly to the customer/merchant.

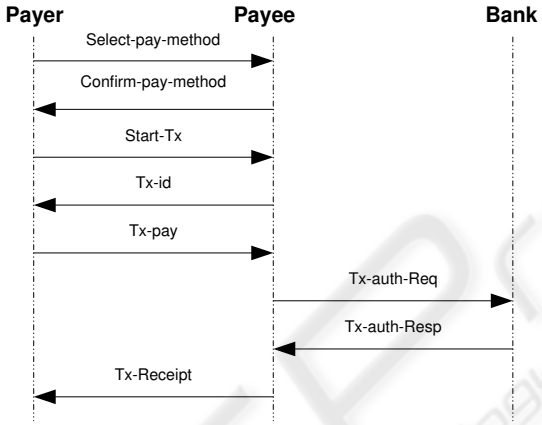

Fig. 1. Generic Payment Protocol

3.2 Generic Payment Protocol

We generalize the payment protocol presented by Bolignano [6] to cover account based and token based PSs. We also add messages for the negotiation of a payment method (see figure 1). This protocol corresponds to the purchase and validation phases. The customer is supposed to have already selected the products or services to purchase and completed the order form. The customer has a valid order description (*Ordd*) and a valid payment description (*Payd*) and is supposed to initiate the purchase. The transaction data *Tx-data,* composed and signed by the customer, includes *Tx-Amount,* customer and merchant identities and time of transaction. It is sent together with *Ordd and Payd*. *Ordd* should only be known by the merchant and is encrypted with *Km* (merchant's key) while *Payd* (account number and PIN, credit card number or a set of tokens {e1..en}) should only be known by the bank and is encrypted with *Kb* (bank's key).

3.3 Protocol for electronic token purchase (withdrawal)

The customer sends a request to the bank (issuer) to purchase/withdraw a certain value of tokens. A generic token purchase protocol is as follows : The bank and customer form a random transaction identifier *Tx-Id*. The customer sends a token purchase request including *Tx-Id* and the value of tokens and signs the request. The bank checks that the

customer's account has enough provision. It debits the customer's account and forms the tokens *e1..en,* encrypts them with the customer's public key Kc, signs the envelope and sends it to the customer.

4 Relational-based Trust Management model

Over the last ten years, several TM systems [8,7] have been developed, some focusing on authentication like X.509 and PGP, others for general purpose authorization like SPKI and KeyNote and others based on logics. Keynote [8]uses credentials that directly authorize actions. It requires that credentials and policies be written in a specific assertion language, designed to work smoothly with its compliance checker. The Keynote engine is provided by calling a list of credentials, policies, along with the public keys of the requester and a list of attribute-value pairs generated by the application and defining an action environment. However, it [9]does not support revocation, external function calls, parametrized action attribute and verification of credential ownership.

SPKI [10]uses cryptographic keys to represent identities, local names to designate users, and authorizations to help achieving compliance checking. Authorization certificates grant authorizations, or delegate the ability to grant authorizations. Further more, certificate chain discovery and certificate reduction mechanisms are used for proof retrieval and validity.

4.1 Definition and Representation of the Trust Management model

As stated in [1], our relational-based TM model is defined by (a) a *set X*, called the certificate/request space (requests for on-line checks and operations accompanied by appropriate certificates), (b) a *set Y*, called the checking outputs (all signed responses) and (c) a *binary relation R* from $X^+$to Y ($X^+$is the set of finite non empty chains of elements in X).

Entities and actions. The model uses SPKI certificates but still supports other certificate formats (i.e. X.509). The certificate set includes CRLs. The authorization and delegation expressions are extended. An action description is a pair <action (act-name input parameters), π>, where act-name is an action identifier and π is a predicate defining requirements for the action execution. Delegation can include a time dependent predicate.

Requests and on-line checks. The TM language specifies and classifies requests and on-line checks in the authorization, delegation and validity checks.

Axiomatic representation of the model. The model is described as a deductive system $\triangle$= *(W, Axiom, Rule) where*

W is the set of well formed formula (formula of the first order logic interpreted on X^+ and Y and formula of the form "(C,y) $\in$R";

Axioms of the form : *(C,y)*$\in$*R*$\wedge$ π*(c)*$\wedge$ *p(y)* where π and *p* are predicates interpreted on X^+ and Y,

Rules define equivalences between certificate chains.

4.2 Main features

Our relational based TM system has the following features:

(1) *A relational language* providing a simple and mathematically defined semantics. This language is monotonic but it still supports negative credentials. It also allows to specify entities by naming certificates for authentication and authorization certificates for actions and authorization delegation.

(2) *A compliance engine* addressing various tasks including: certificate validation, request authorization, and enforcement of local security policies. The compliance checking process is specified by a simple and compact relational calculus using fix-point approach and internal state derivation.

(3) *An axiomatic representation of the security policy* which helps capturing all the properties that the security manager needs to address.

(4) *A mechanism for identifying entities* requesting to perform actions based on public key certificates.

5 Trusted Generic payment model

5.1 Specification of entities and actions

Entities in our PS are (1)the customer identified by his public key Kc and a name certificate associated to his account,(2) the merchant identified by his public key Km and a name certificate associated to his account and the bank identified by its public key Kb and its name certificate.

Name certificates. A name certificate is structured as follows:

<*cert(issuer (name k n)(subject p), val*> where *k* and *n* are the public key and the name of the bank while *p(k' n')* and *k'* is the public key of the account holder and *n'* is the name of the account holder and account identifier; In case of anonymous PSs, the customer may be identified by a *pseudonym.* *Val* is the validity period {not-before, not-after} of the certificate.

Authorization certificates. We are mostly concerned by the authorization and validation tasks of the bank and its compliance engine as well as the security policy that protects bank's, customer's and merchant's assets. Authorization certificates have the form :

<*cert (issuer (name k' n')(subject p) Act Del Val*> where *k'* and *n'* are the public key and the name (or pseudonym) of the certificate owner and subject p(*k" n"*) represents the public key and name of the principal taking benefit from actions in Act.

Act (action field) has the form <action(act-name in-parameters), μ> where μ is a predicate defining requirements for the action execution. In our case, Act $\in$ {withdraw, deposit, pay, multi capture}

Del = 0 which means that delegation is not allowed in our system.

Val has the form <Val (not-before, not-after, options) > where <options (online-test)> and <online-test (online-type uris issuer(p)>. Val contains a predicate now(t) so that the certificate becomes invalid as soon as the execution of the requested action ends.

Certificate revocation list CRL. CRLs have the form:

<Crl(issuer (name k n)) (revoked c1 ... cn) Val> where cj=(kj,hj) for j≤n, cj represents the public key and the hash value of a revoked certificate.

Requests and on-line checks. the customer and merchant may submit requests for naming certificate generation, renewal and revocation as well as validity and revocation online-checks and having the form:

<Req (Checker check-type) (k p Act Del Val) opt.> where Checker corresponds to uniform resource identifiers (uris) to be used for on-line check. check-type ∈ {val-ch, aut-ch}; val-ch corresponds to validity checks aut-ch corresponds to authorization checks.

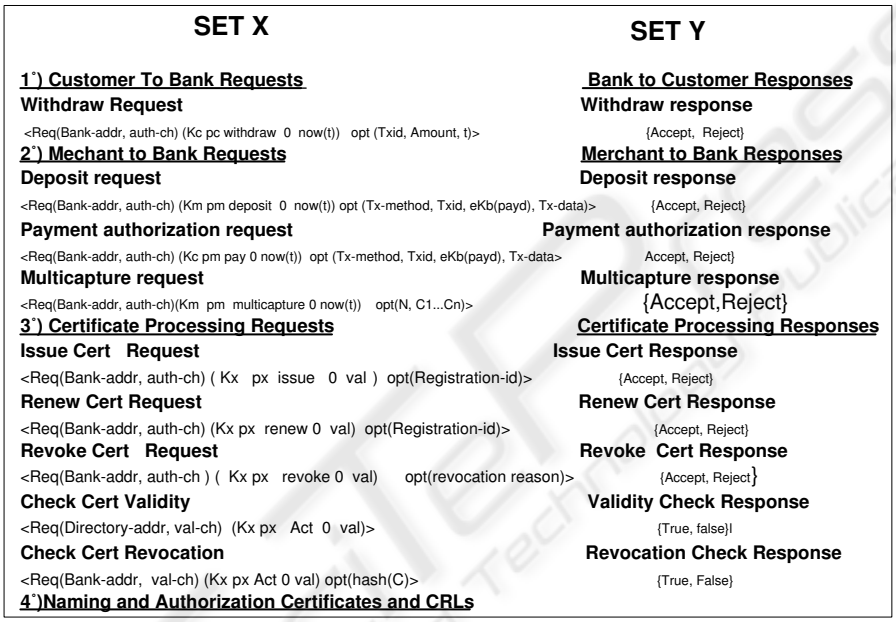

Fig. 2. Set X (certificates and requests) and Set Y(responses)

5.2 Specification of the model

We need to define sets X and Y and the binary relation R. Set X contains name, authorization and revocation certificates and requests for on-line checks and operations. Set Y contains all signed responses returned by the bank. R contains axioms of the form (xo..xn; y) where xn is a request received by b and y is a signed response of b based on all messages prior to xn (see figure 2).

5.3 Specification of the security policy

The specification of the security policy describes the relation R to be used in the compliance check processes. Axioms and rules are deduced semi-automatically from the PSs' security requirements. We introduce some functions to be used in the expression of axioms and rules:

owner (c) returns the owner of certificate c
signer (c) returns the signer of certificate c
val(c) returns the content of the validity field of c
multi-spend(c) is a boolean function showing if the tokens have been spent
valid-tokens(c) is a boolean function indicating if the tokens are valid
Max-Tx is the maximum allowed amount for an account-based transaction
owner-key(c) returns the public key of the naming certificate c

Axioms. As defined in the relational-based trust model[1], axioms have the form : *(C,y)* $\in R \wedge \pi(c) \wedge p(y)$, where $\pi$and *p* are predicates interpreted on $X^+$and Y, respectively, and y typically defines the desired response, at time t, of the model for a trivial input chain of certificates, denoted C. Due to space constraints, we only present a sample of axioms and rules to demonstrate the expressiveness of our TM language.

A1-Payment axiom (c1.c2.c3; auth) ∈ R if:
Tx-method (C3) = account
∧ c1 is a valid naming certificate
∧ c1 is not revoked
∧signer(c3) = owner (c2)
∧owner(c3) = owner (c1)
∧ c2 is a valid naming certificate
∧ c2 is not revoked
∧ c3 is a payment certificate
∧ Tx-amount(c3) < Max-Tx

Where c1 is the merchant name certificate, c2 is the customer name certificate, c3 is the customer one-time payment authorization certificate.

The payment axiom states that an account-based payment transaction is authorized if the merchant and the customer have valid and not revoked certificates at the time of the transaction and the authorization certificate is generated and signed by the customer to the benefit of the merchant. The axiom also states that Tx-amount should not exceed a maximum value Max-Tx fixed by the bank.

A2- Deposit axiom (c1.c2.c3; auth) ∈ R if:
Tx-method (c3) = cash
∧ c1 is a valid naming certificate
∧ c1 is not revoked
∧signer(c3) = owner (c2)
∧owner(c3) = owner (c1)
∧ c2 is a valid naming certificate
∧ c2 is not revoked
∧ c3 is a deposit certificate
∧ multi-spend(c3) = false
∧ valid-tokens(c3) = true

The deposit axiom states that a deposit of electronic coins is authorized if the merchant and the customer have valid and not revoked certificates at the time of the transaction and the authorization certificate is issued and signed by the customer to the benefit

of the merchant. The axiom also states that the submitted tokens should be valid and not spent before.

Deduction rules. Deduction rules typically define equivalences between certificate chains or more precisely, request histories and have the following form:

(C',y) ∈ R
π(C,C')
————————-
(C,y) ∈R

R1-Closed-world Rule (C.c2;v)∈ R,
there is a naming certificate c1 ∈ C;
signer (c2) = owner(c1),
C' does not contain any request related to c1
————————————————————-
(C.c1.C'.c2;v) ∈ R

The closed-world Rule states that the system response to a request related to a given account should remain independent from any other request or operation that does not involve this account.

R2- Certificate Revocation Rule (C.x.C'.x.C"; y) ∈ R
x: request for the issuance of name certificate c
x: request for the revocation of name certificate c
————————————————————————
(C.C"; y) ∈ R

Where C is a request submitted before the issuance of c and C" is a request submitted after the revocation of C .

The *certificate revocation rule* insures that as soon as a certificate is revoked it has no effect on the responses generated for the following requests.

6 Correctness issues

As declared in [1], the compliance checking relation is a solution of the system : $A \cup D \circ R \subseteq R$, where A is a subset of all pairs (C,y) in $X^+ \times Y$ for which an axiom is satisfied and D is a subset of all pairs (C,C') of lists in $X^+ \times Y$ for which a rule is applicable. By using the least fix-point approach, R is given by :

$$R = A \cup D \circ A \cup D^2 \circ A \cup D^3 \circ A \cup ... = (\cup_{I \geq 0} D^i) \circ A$$

Specification completeness. Specification completeness is achieved when the system requirements are entirely covered by the specification of entities, actions and particularly the input set X, the output set Y and the relation R (axioms and rules). A special care should be dedicated to define a minimal, consistent and non redundant specification of axioms and rules.

Specification validation. Verification and Validation (V&V) techniques [11]help detecting requirements and specification errors at an early stage.

Compliance correctness. Compliance correctness directly depends on security policy specification correctness. The compliance checking process involves axioms and rules declarations by using a reduction process and computing the appropriate output.

Certificate Chain discovery. Several certificate chain discovery solutions have been proposed [12] for TM systems. In our TM system, certificate chains can be directly derived from requests and requests histories.

Handling revocation. Monotonicity requirement can be a problem for revocation handling in certain TM systems[9]. Our TM language supports negative assertions and revocation is handled by the revocation rule which includes the account history $\underline{c}$ after certificate revocation in replacement of certificate c. Hence, a revoked certificate has no effect on future requests.

Performance issues. Our TM system enhances the relational calculus performance by virtue of reduction algorithms provided by equivalence classes. Equivalence classes deduced from rules reduce the compliance checking process run-time by providing the possibility to reduce certificate chains supporting a request, i.e., the certificate chain C.c0.c1 is reduced to C.c0 for validity checking providing that signer(c1)=owner(c0).

7 Implementation Issues

The following methodology is suggested to implement the relational based TM system in electronic PSs (1) *Establishing the needs :* Several approaches [13]have been suggested for the specification of policy and requirements, particularly in electronic commerce systems. This step leads to the definition of the security policy, (2) *Customizing the relational model:* The building blocks of our TM model are the compliance engine, the security policy module and the request histories database. The model requires a relatively small coding effort, (3) *Using the appropriate relational engine :* relational calculus algorithms for compliance checking and system state derivation have been defined in [1] and constitute the core of the compliance engine, (4) *Building useful interfaces :* the compliance engine needs interfaces to interact with several components: policy block, CA, RA,..., (5) *Testing the implementation:* several testing techniques are provided for the V&V purposes such as white-box testing and black-box testing.

8 Conclusion

We have presented the novel techniques and special features of our relational based TM system through its application to a generic payment model. The elaboration of the security policy axioms and rules demonstrates the expressiveness of the TM relational language. Moreover, we explained how compliance correctness can be demonstrated through completeness and minimality of security policy. Finally, an implementation methodology is proposed. Future work will focus on delegation issues as well as formal verification of the TM model.

References

1. Guemara-El Fatmi, S., Boudriga, N., Obaidat, M.S.: Relational-based calculus for Trust Management in network services. To appear in Computer Communication Journal (2004)
2. Asokan, N., Janson, P.A., Steiner, M., Waidner, M.: The state of the art in electronic payment systems. IBM Zurich (1997)
3. Bellare, M., Garay, J., Herzberg, A., Krawczky, H., Steiner, M., Tsudik, G., Waidner.M.: Design implementation and deployment of the iKP secure electronic payment system. IEEE Journal of Selected Areas in Communications (2000)
4. Abad Peiro, J.L., Asokan, N., Steiner, M., Waidner., M.: Designing a generic payment service. IBM Systems Journal, vol. 37, N°1 (1998)
5. Dasweni, N., Boneh, D., Garcia-Molina, H., Ketchpel, Paepcke, S.A.: SWAPEROO: a simple wallet architecture for payments, exchanges, refunds and other operations. 3rd Usenix workshop on electronic commerce (1998)
6. Bolignano, D.: Towards the Formal Verification of Electronic Commerce Protocols. 10th Computer Security Foundations Workshop, IEEE Computer Society Press (1997) 113–147
7. Weeks, S.: Understanding Trust Management systems. In Symp.on. Res.in Sec and Privacy .IEEE Computer Society Press (2001)
8. Blaze, M., Ioannidis, J., Keromytis, D.: Experience with the Keynote Trust Management System: Applications and Future Directions. In Proceedings of the 1st International Conference on Trust Management (2003) 283–300
9. Seamons, K., Winslett, M., Yu, T., Smith, B., Child, E., Jacobson, J., Mills, H., Yu, L.: Requirements for Policy Languages for Trust Negotiation (2002)
10. Ellison, C., Frantz, B., Lampson, B., Rivest, R., Thomas, B., Ylonen, T.: SPKI Certificate Theory. RFC 2693 (1999)
11. IEEE Standard 1012 for Software Verification and Validation (1998)
12. Li, N., Mitchell, J., Winsborough, W.: Distributed Credential Chain Discovery in Trust Management. Stanford Security Workshop (2002)
13. Anton, A., Earp, J.: Strategies for developing Policies and Requirements for Secure Electronic Commerce Systems. CCS2000 (2000)