

An Open Approach for the Applied Pi Calculus

Han Zhu

Basics Lab, Shanghai Jiao Tong University, Shanghai, China

Email: zhu-h@sjtu.edu.cn

Yonggen Gu, Xiaohong Wu

Department of Computer Science and Technology, Huzhou Teachers College, Huzhou, China

Email: {gyg68, xhww}@hutczj.cn

Abstract—A notion of open bisimulation is proposed for the Applied Pi Calculus, which extends π -calculus in order to facilitate analyzing security protocols. Our notion is based on the labeled transition system, and takes a knowledge aware open approach to model knowledge in security protocols. It is shown to be sound to labeled bisimilarity and is a congruent relation. As a running example, we analyze two e-commerce protocol, namely iKP and Ferguson's electronic cash protocol, by Applied Pi and open bisimilarity.

Index Terms—Applied Pi, Open Bisimulation, Security Protocol, Anonymity, iKP, Electronic Cash Protocol

I. INTRODUCTION

The Applied Pi Calculus is introduced by Abadi and Fournet in [1]. As an extension of π -calculus [2], [3] and the Spi Calculus [4], Applied Pi shifts the focus away from encodings and substantial theories to an applied usefulness, from which its name is derived. It has a more general setting of terms to model most security primitives nowadays, and this setting provides a mechanism to model unknown new primitives which would appear in future. The calculus itself is decoupled from terms, so its algebraic property is not affected by which primitives its terms are modeling. In the field of analyzing security protocols, where knowledge and term equations play an important rule in guaranteeing security properties of protocols (secrecy, authentication, etc), Applied Pi is the most suitable process calculus. It has been used to model Just Fast Keying [5], certified email [6], private authentication [7], etc.

Labeled bisimilarity [1] is defined upon labeled transition system. However, labeled bisimilarity does not seem to be a suitable equivalence relation between processes for security protocols. One reason is that it does not take account of knowledge. Secondly it is not a congruent relation. It is also coarser than we want. For example,

let us take the following two processes:

$$\begin{aligned} A &\equiv \nu m.(\bar{a}\langle \text{enc}(m, k) \rangle.b(x).(\tau.\bar{c}\langle c \rangle + \tau.\bar{d}\langle d \rangle) \\ &\quad + \tau.\text{if } \text{dec}(x, k) = m \text{ then } \bar{c}\langle c \rangle \text{ else } \bar{d}\langle d \rangle)), \\ B &\equiv \nu m.(\bar{a}\langle \text{enc}(m, k) \rangle.b(x).(\tau.\bar{c}\langle c \rangle + \tau.\bar{d}\langle d \rangle)). \end{aligned}$$

One can easily check that A is labeled bisimilar to B since

$$\text{dec}(\text{enc}(m, k), k) = k$$

is one of the equations in the equation theory. But from the viewpoint of a practical observer, these two processes should not be bisimilar. Because the third summand of A ,

$$\tau.\text{if } \text{dec}(x, k) = m \text{ then } \bar{c}\langle c \rangle \text{ else } \bar{d}\langle d \rangle,$$

could tell apart the content it receives on channel b by sending something on channel c or d . While B does not have this ability.

To remedy the situation, we turn to open bisimilarity, a much finer equivalence relation than early and late labeled bisimilarity. Introduced by Sangiorgi [8] on π -calculus, open bisimilarity is quite attractive due to both its reasonable full congruence property and simple axiomatization (for finite terms). A full congruence means the equivalence relation is preserved by all operators including the input prefix. Simple axiomatization makes it rather straightforward to build tools to check the equivalence relation.

The approach Sangiorgi developed is to use sets of distinction to keep track of the inequalities of names. Accordingly, open bisimulation is a set of distinction-indexed relations. The distinction of each relation restricts the set of substitutions in the bisimulation game to those reasonable. Applied Pi has compound terms which π -calculus does not have. Since compound terms may be made up of bounded names, simply extending the set of distinctions to all terms does not work for us. For example, we cannot put $(x, \nu k.(\text{enc}(m, k)))$ into the distinction set, because m may belong to the domain of σ , then $x\sigma \neq \nu k.(\text{enc}(m, k))\sigma$ will make only a mess.

Active substitution and statically equivalence are other additions to the pure π -calculus. Active substitutions model the messages and the knowledge of the environment. Statically equivalence means the messages output

This work is supported by the National 973 Project (2003CB317005), the National Nature Science Foundation of China (60573002, 60703033), the Nature Science Foundation of Huzhou City Zhejiang Province (2007YZ10) and the Sci-Tech Brainstorm Stress Project of Huzhou City Zhejiang Province (2007G22).

by two processes are indistinguishable from the viewpoint of the environment. These facts suggest us for a knowledge aware definition of open bisimulation on Applied Pi.

Briaes and Nestmann revisited open bisimulation in [9], where they bring up a same relation as open bisimilarity but through a knowledge aware approach. In their open bisimulation, a knowledge based environment with time precedence relation is used to deal with the bounded name. The environment records the history of a process when running. The time precedence relation records those pairs of names, where the first part is a bounded name and the second part is a free substitutable name. (n, a) is recorded in the time precedence relation means the bounded name n is emitted before a becomes a free substitutable name. The respectfulness of a substitution and the environment plays the role in inhibiting from substituting a later emitted bounded name for a free substitutable name.

The basic idea is straightforward. A knowledge based environment with time precedence relation is used to deal with the bounded names. The environment records the history of a process when it is running. The time precedence relation records those pairs of names and terms. Its role is to inhibit from substituting a later emitted term for a free substitutable name.

Contributions. Our contributions are threefold. i) We develop a knowledge aware open style bisimulation equivalence relation for the Applied Pi Calculus. ii) The open bisimilarity is shown to be sound to labeled bisimilarity and is a congruent relation. iii) We analyze the anonymity of iKP and an electronic cash protocol to show the effectiveness of our approach.

Related Work. Process calculi are a family of related approaches for formally modeling concurrent systems and providing tools for the high-level description of interactions and communications between parallel agents or processes. They also provide algebraic laws to aid us in analyzing process descriptions, and make it possible to formally reason about certain properties of processes (bisimilarity, etc). Leading examples of process calculi include CSP [10], CCS [11], and ACP [12], [13]. More recent additions to the family include π -calculus [14], χ -calculus [15], and ambient calculi [16], [17].

Using process calculi to analyze network security protocols is firstly studied by Lowe [18], wherein a flaw of Needham-Schroeder key distribution protocol is found by modeling the protocol in CSP and checking it with FDR. Later on, some *ad hoc* calculi are proposed, among whom the Spi Calculus [4] and the Applied Pi Calculus [1] are the most widely studied. Equipped with cryptographic primitives and simplified in expressiveness compared to π -calculus, Spi specializes in modeling security protocol, but it still suffers from its lack of generality. Applied Pi has a more general setting of terms, therefore it can model a greater variety of security protocols and security properties [5], [6], [7], [19].

Briaes and Nestmann revisited open bisimulation of

π -calculus in [9], where they bring up a same relation as open bisimilarity but through a knowledge aware approach. Many knowledge aware equivalence relations for Spi have been proposed to extend standard bisimulation equivalence which is lack of knowledge representation. These knowledge aware relations include framed bisimulation [20], hedged bisimulation [21], etc.

Anonymity, also called untraceability, was first proposed by Chaum [22] to solve the Dining Cryptographer Problem. After that, a great deal of research has been carried out on this topic and various formal definitions and frameworks for analyzing anonymity have been developed in the literature. For example, Schneider and Sidiropoulos analyzed anonymity with CSP [23]. They used substitution and observable equivalence to define anonymity in CSP. In their framework, the automatic tool FDR [18] was used to check the equivalence of two processes. In [24] Kremer and Ryan analyzed the FOO92 voting protocol with the applied pi calculus and proved that it satisfies anonymity. Chothia [25] used bisimulation in the pi calculus to test the anonymity of an anonymous file-sharing system. Chothia *et al* [26] proposed a general framework based on the process algebraic verification tool μ CRL for checking anonymity and applied it to several protocols, including the Dining Cryptographer Problem and the FOO92 voting protocol. Our framework is similar to [24], but our proofs are partly done by ProVerif when in [24] all the proofs are done manually.

Outline of the Paper. The rest of the paper is organized as follows. Section II briefly introduces the Applied Pi Calculus and its labeled bisimilarity. Section III develops knowledge aware open bisimilarity and proves its soundness to labeled bisimilarity and the congruence property. Running examples of analyzing iKP and electronic cash protocol are shown in Section IV. Finally Section V concludes and points out some directions for future research. All technical details can be found in [27].

II. APPLIED PI AND LABELED BISIMULATION

In this section, we will briefly recall the syntax and semantics of the Applied Pi Calculus. Readers are referred to [28] for full details.

We presume a countable set of names \mathcal{N}_c , ranged over by a, b, \dots and their decorated forms, and a countable set of variables \mathcal{N}_v , ranged over by x, y, \dots and their decorated forms. \tilde{n} denotes some finite set of names $\{n_1, \dots, n_k\}$. A signature Σ is the set of all function symbols needed to model some protocols.

Terms are defined as follows:

$L, M, N, T ::=$	terms
$a, b, c, \dots, k, \dots, m, n$	names
$ x, y, z$	variables
$ f(M_1, \dots, M_l)$	function applications

They are constructed from names, variables and function application on terms. A *substitution* σ is a map from variables to terms. As in π -calculus, names are used to

express channels, and set of channels is denoted by $Chan$. Names can represent atomic data such as keys, nonces, random numbers as well. Function applications on terms are proposed to model all kinds of cryptographic operations such as *encryption* and *decryption*. The relationship of cryptographic primitives is described by an equation theory E . For example, when we are dealing with symmetric cryptography, we could let $\Sigma = \{\text{enc}, \text{dec}\}$ and E has the following equation $\text{dec}(\text{enc}(x, y), y) = x$.

Plain processes are constructed by commonly seen operators:

$P, Q, R ::=$	plain processes
$\mathbf{0}$	null process
$ P Q$	parallel composition
$! P$	replication
$ \nu n. P$	name restriction
$ \text{if } M = N \text{ then } P \text{ else } Q$	conditional
$ u(x). P$	message input
$ \bar{u}(N). P$	message output

Extended processes are plain processes paralleled by active substitutions and the restrict operator also has effect on active substitutions:

$A, B, C ::=$	extended processes
P	plain processes
$ A B$	parallel composition
$ \nu n. A$	name restriction
$ \nu x. A$	variable restriction
$ \{M/x\}$	active substitution

The differences between plain processes and extended processes are *active substitutions*. The notation $\{M/x\}$ is an active substitution which replaces the variable x with the term M . The active substitution $\{M/x\}$ typically appears when the term M has been sent out, and the environment could receive it. The *frame* of an extended process is the active substitution part of an extended process. It can be written in the form of $\phi_A = \nu \tilde{n}. \{\tilde{M}/\tilde{x}\}$ and $\tilde{n} \subseteq n(\tilde{M})$. Every extended process can be mapped into its frame by removing plain processes.

We write $fn(A)$ and $bn(A)$ for free and bound names of A . The set of names that occur in A is denoted as $n(A) = bn(A) \cup fn(A)$.

If A is an extended process, the frame of A is the active substitution part of it. We write $dom(\phi)$ for the domain of ϕ , a set of variables which appear in ϕ but not under a variable restriction. The set of frames are denoted by $Frame$.

Frames can be viewed as the static knowledge exposed by A to the environment, so there comes the *deduction* [29] which can be done by the environment. We write $\phi \vdash M$ to mean that M can be deduced from ϕ . The deduction rules are listed in Table I.

An *evaluation context*, denoted by $C[_]$, is a context whose hole is not under a replication, a conditional, an

input, or an output. An evaluation context $C[_]$ closes A when $C[A]$ is closed, and $C[_]$ is called a *closing evaluation context*.

Definition 1. We say that two terms M and N are *equal* in the frame ϕ , and write $(M = N)\phi$, if and only if $\phi \equiv \nu \tilde{n}. \sigma$, $M\sigma = N\sigma$ and $\{\tilde{n}\} \cap (fn(M) \cup fn(N)) = \emptyset$ for some names \tilde{n} and substitution σ .

Definition 2. We say that two closed frame ϕ and ψ are *statically equivalent*, denoted by $\phi \approx_s \psi$ if $dom(\phi) = dom(\psi)$ and for all terms M, N , we have $(M = N)\phi$ if and only if $(M = N)\psi$.

Definition 3 (Static Equivalence \approx_s). We call two closed extended processes A, B *statically equivalent*, and write $A \approx_s B$, if their frames are statically equivalent.

The structural rules and labeled transition rules are also listed in Table I.

Definition 4. Labeled bisimilarity \approx_l is the largest symmetric relation \mathcal{R} on closed extended processes such that $A \mathcal{R} B$ implies,

- 1) $A \approx_s B$;
- 2) if $A \xrightarrow{\tau} A'$, then $B \xrightarrow{\tau} B'$ and $A' \mathcal{R} B'$ for some B' ;
- 3) if $A \xrightarrow{\alpha} A'$ and $fv(\alpha) \subseteq dom(A)$ and $bn(\alpha) \cap fn(B) = \emptyset$, then $B \xrightarrow{\tau} B' \xrightarrow{\alpha} B'' \xrightarrow{\tau} B''$ and $A' \mathcal{R} B''$ for some B' .

The equational theory plays an important rule in the Applied Pi. Some positive results have been reported in [29]. For example, deducibility is decidable in polynomial time for a large class of equational theories including the theories for encryption, decryption and digital signatures. In this paper, we assume that equational theories are decidable.

III. OPEN BISIMULATION

The active substitution is already a good candidate for the knowledge aware environment. However it will be absorbed by an input action. This causes some information lost after an interaction, so we reserve the time precedence constraint on substitutions. Since a term can substitute for a free name, our precedence relation is defined on $Frame \times (\mathcal{N}_c \cup \mathcal{N}_v)$. The frame of a process is obtained by mapping all plain processes to $\mathbf{0}$, and all active substitutions remained.

The environment will be the form of $e = \langle V, \prec \rangle$, plus the implicit frames of extended processes. V is the set of substitutable names and variables, and \prec is time precedence relation. The constraint on substitution is achieved by the so-called respectiveness of a substitution to an environment $\sigma \triangleright e$. As the bisimulation game is playing, the environment should be updated and extended accordingly.

When talking about open bisimulation, substitutions play a crucial part of role in its distinguishability. Carefully defined concept of respectfulness confines substitution to a reasonable domain in order to capture the intu-

TABLE I
STRUCTURAL, LABELED TRANSITION AND DEDUCTION RULES

(Par-0) $A \equiv A 0$	(Par-A) $A (B C) \equiv (A B) C$	(Par-C) $A B \equiv B A$	(Repl) $!P \equiv P !P$
(New-0) $\nu n.0 \equiv 0$	(New-C) $\nu u.\nu v.A \equiv \nu v.\nu u.A$	(New-Par) $A \nu u.B \equiv \nu u.(A B)$ if $u \notin fv(A) \cup fn(A)$	
(Alias) $\nu x.\{M/x\} \equiv 0$	(Subst) $\{M/x\} A \equiv \{M/x\} A\{M/x\}$	(Rewrite) $\{M/x\} \equiv \{N/x\}$ if $\Sigma \vdash M = N$	
(In) $a(x).P \xrightarrow{a(M)} P\{M/x\}$	(Out-Atom) $\bar{a}\langle u \rangle.P \xrightarrow{\bar{a}\langle u \rangle} P$	(Par) $\frac{A \xrightarrow{\alpha} A', \quad bv(\alpha) \cap fv(B) = bn(\alpha) \cap fn(B) = \emptyset}{A B \xrightarrow{\alpha} A' B}$	
(Scope) $\frac{A \xrightarrow{\alpha} A', \quad u \notin n(\alpha) \cup v(\alpha)}{\nu u.A \xrightarrow{\alpha} \nu u.A'}$	(Open-Atom) $\frac{A \xrightarrow{\bar{a}\langle u \rangle} A', \quad u \neq a}{\nu u.A \xrightarrow{\nu u.\bar{a}\langle u \rangle} A'}$	(Struct) $\frac{A \equiv B, \quad B \xrightarrow{\alpha} B', \quad B' \equiv A'}{A \xrightarrow{\alpha} A'}$	
Sub $\frac{x \in dom(\sigma) \wedge x\sigma = M}{\nu \tilde{n}.\sigma \vdash M}$	Eq $\frac{\phi \vdash M \wedge M = N}{\phi \vdash N}$	Fun $\frac{\phi \vdash M_1, \dots, M_k, \quad f \in \Sigma}{\phi \vdash f(M_1, \dots, M_k)}$	Nonce $\frac{s \notin \tilde{n}}{\nu \tilde{n}.\sigma \vdash s}$

ition of substitution. In open bisimulation for π -calculus, the sets of distinctions record what should be thought different in every substitution up to this time. Here we use an alternative auxiliary entity, namely environment, to record what can be substituted and what can be substituted for up to this time.

In Applied Pi, it is not required that the same σ applies to both P and Q , since two key protected cyphertexts are thought to be equal when the key is unknown to observers. We use a tuple $\sigma = \langle \sigma_1, \sigma_2 \rangle$, whose first part σ_1 applies to P and σ_2 applies to Q . From now on, we write bold face A to denote a tuple in the form of $\langle A_1, A_2 \rangle$.

Definition 5. A tuple $\langle V, \prec \rangle$ is an *environment*, if $V \subseteq (\mathcal{N}_c \cup \mathcal{N}_v)$ is a finite set of substitutable names and variables from both two sides, $\prec = \langle \prec_1, \prec_2 \rangle$ and $\prec_i \subseteq Frame \times V (i \in \{1, 2\})$ is the precedence relation between frames and substitutable names and variables. The set of all environments is denoted by E .

The intuition behind an environment $e = \langle V, \prec \rangle$ is to make a clear notion of what can be substituted for and what can be substituted when applying substitution. Relation \prec_i stores the time precedence between the frame and the input names, thus avoiding substituting a later emitted bounded name or term for a currently substitutable name.

Definition 6. We say a pair of substitution $\sigma = \langle \sigma_1, \sigma_2 \rangle$ *respects* e , denoted by $\sigma \triangleright e$ if the following holds,

- $dom(\sigma_i) \subseteq V$ for $i \in \{1, 2\}$;
- $\forall x \in V : x \in dom(\sigma_1) \iff x \in dom(\sigma_2)$;
- $\forall x \in V : x\sigma_1 = x\sigma_2$, where $\phi_i \prec_i x$ for $i \in \{1, 2\}$;
- if $x \in Chan$, then $x\sigma_i \in N$ for $i \in \{1, 2\}$.

The environment should be updated when a pair of substitution is applied, reflecting the changes of names.

Definition 7. Let $\sigma = \langle \sigma_1, \sigma_2 \rangle$ be a pair of substitutions, $e = \langle V, \prec \rangle$ be an environment, the *updated* environment $e^\sigma = \langle V', \prec' \rangle$ is defined as follows,

- $V' = V \setminus dom(\sigma)$;

- $\phi_i \sigma_i \prec'_i x \sigma_i$ if $\phi_i \prec_i x$.

The environment should also be extended when an input bounded name becomes free, e.g. the x in P when $a(x).P \xrightarrow{a(x)} P$. So we have the following definition.

Definition 8. Let $e = \langle V, \prec \rangle$ be an environment. The *extension* of e , denoted by $e \oplus_V(x) = \langle V', \prec' \rangle$ will be defined as,

- $V' = V \cup \{x\}$;
- $\prec'_i = \prec_i \cup \{\langle \phi_i, x \rangle\}$.

We define $e^{-1} = \langle V^{-1}, \prec^{-1} \rangle$ where \prec^{-1} is the reverse relation of \prec . An *open* relation R is a subset of $E \times P \times P$, such that $\forall \langle e, P_1, P_2 \rangle \in R : fn(P_i) \subseteq V$. It is called *symmetric* if $\forall \langle e, P_1, P_2 \rangle \in R : \langle e^{-1}, P_2, P_1 \rangle \in R$.

Definition 9. An open relation R is an *open bisimulation* if $\langle e, P_1, P_2 \rangle \in R$ implies $P_1 \approx_s P_2$, and for all $\sigma \triangleright e$, if $P_1 \sigma_1 \xrightarrow{\mu_1} P'_1$ with $bn(\mu_1) \cap fn(P_1) = \emptyset$, then there exists P'_2, μ_2 such that $P_2 \sigma_2 \xrightarrow{\tau}^* \xrightarrow{\mu_2}^* \xrightarrow{\tau}^* P'_2$ with $bn(\mu_2) \cap fn(P_2) = \emptyset$ and,

- if $\mu_1 = \tau$, then $\mu_2 = \tau$ and $\langle e^\sigma, P'_1, P'_2 \rangle \in R$;
- if $\mu_1 = \bar{a}\langle x \rangle$, then $\mu_2 = \bar{a}\langle x \rangle$ and $\langle e^\sigma, P'_1, P'_2 \rangle \in R$;
- if $\mu_1 = a(M_1)$, then $\mu_2 = a(M_2)$ where $M_1 = M_2$ and $\langle e^\sigma \oplus_V(fn(M_1, M_2)), P'_1, P'_2 \rangle \in R$.

We say that P and Q are open bisimilar, denoted by $P \approx_o^e Q$, if there exists an open bisimulation R such that $\langle e, P, Q \rangle \in R$. We call $\approx_o^{(fn(P,Q), \langle \emptyset, \emptyset \rangle)}$ open bisimilarity.

Our definition of open bisimilarity is sound to labeled bisimilarity.

Theorem 1. If $P \approx_o^{(fn(P,Q), \langle \emptyset, \emptyset \rangle)} Q$ then $P \approx_l Q$.

Proof sketch: This is done by showing that the relation

$$\mathcal{R} = \{(P, Q) \mid P \approx_o^e Q \wedge e \in E\}$$

is a labeled bisimulation. ■

The example stated in Section I shows that there exist two labeled bisimilar processes which are not open bisimilar. So we have the following corollary.

Corollary 1. *Open bisimilarity is strictly included in labeled bisimilarity.*

Congruence is an important property of equivalence relation in process algebra. It requires an equivalent relation closed under every operator. Congruence makes it possible to manipulate process expressions like terms in algebra without breaking up their equivalent relation. In the fields of analyzing security protocols, a congruent equivalence relation means we can verify security properties of a large system by break up to small parts, then each small part satisfy a security property implies the whole system also satisfy it. Fortunately, we have the following theorem.

Theorem 2. *Open bisimilarity is a congruence.*

IV. ANALYZING PROTOCOLS

We take two e-commerce protocols as our running examples. The first is iKP and the second is Ferguson's electronic cash protocol.

A. iKP

iKP (i-Key-Protocol, $i = 1, 2, 3$) is a family of secure electronic payment protocols [30], [31]. It involves three parties: the buyer B , the seller S and the acquirer A . We are concerned in the anonymity of 1KP.

We write SK_X for the secret key of party X ($X \in \{B, S, A\}$), and $PK(SK_X)$ for its corresponding public key. H is an ideal hash function. E_X means encryption and S_X means signature. We will use the following notations for messages transmitted in the protocol:

- $Desc$: Description of purchase and delivery address.
- $Salt_B$: Random number to salt $Desc$.
- $Authprice$: Amount and currency.
- $Date$: Time stamp.
- $Nonce_S$: Nonce of S .
- ID_S : ID of S .
- TID_S : Transaction ID chosen by the seller.
- Ban_B : Buyer's account number.
- R_B : Random number chosen by B .
- ID_B : Pseudo-ID of B , which is equal to $H(R_B, Ban_B)$.
- $Code$: Authorization code.

Composite messages are defined as follows:

- $Common = (Authprice, ID_S, TID_S, Date, Nonce_S, ID_B, H(Salt_B, Desc))$.
- $Clear = (ID_S, TID_S, Date, Nonce_S, H(Common))$.
- $Slip = (Authprice, H(Common), Ban_B, R_B)$.
- $EncSlip = E_A(Slip)$.
- $Sig_A = S_A(Code, H(Common))$.

The operation flow of this protocol consists of six steps:

- 1) **Initiate** $B \rightarrow S: Salt_B, ID_B$. The buyer sends init, which is composed of the hash code of his/her account number, a random number, etc.
- 2) **Invoice** $S \rightarrow B: Clear$. The seller answers by the hash code of price and other information. The message is composed of common and clear.
- 3) **Payment** $B \rightarrow S: EncSlip$. The buyer checks the invoice and sends the payment slip encrypted by the acquirer's public key.
- 4) **Auth-Request** $S \rightarrow A: Clear, H(Salt_B, Desc), EncSlip$. The seller requests the acquirer to authorize the payment. The message is composed of clear and req.
- 5) **Auth-Response** $A \rightarrow S: Code, Sig_A$. The acquirer checks the request, extracts information from it, authorizes it and then sends response resp to the seller.
- 6) **Confirm** $S \rightarrow B: Code, Sig_A$. The seller extracts the response and the signature of the acquirer, verifies it and then sends confirm to the buyer.

We now model this protocol in Applied Pi. The signature Σ of it is:

$$\Sigma = \{\text{init, clear, common, slip, req, resp, confirm, proj.i, hash, dec, enc, checksig, pk, true, false}\}.$$

The function symbols $\text{init, clear, common, slip, req, resp,}$ and confirm represent the message flows transmitted in the protocol. Their parts can be extracted. The extraction operator here we use is proj.i . The rest of function symbols in Σ is self-evident. They are common cryptographic primitives.

The equation system is:

$$\begin{aligned} \text{dec}(\text{enc}(x, \text{pk}(z)), z) &= x, \\ \text{checksig}(\text{enc}(x, z), \text{pk}(z)) &= \text{true}, \\ \text{proj.i}(\text{tuple}(x_1, \dots, x_l)) &= x_i \quad \text{if } 1 \leq i \leq l. \end{aligned}$$

The last equation is a short form of a family of equations with similar forms, that is

$$\text{tuple} \in \{\text{init, clear, common, slip, req, resp, confirm}\}.$$

We use substitutions to define the messages transmitted in the protocol:

$$\begin{aligned} \sigma_1 &= \{\text{init}(Salt_B, ID_B)/x_1\}, \\ \sigma_2 &= \{\text{clear}(ID_S, TID_S, Date, Nonce_S, \\ &\quad \text{hash}(Authprice, ID_S, TID_S, Date, Nonce_S, \\ &\quad \text{proj.2}(x_1, \text{hash}(\text{proj.1}(x_1), Desc))))/x_2\}, \\ \sigma_3 &= \{\text{enc}(\text{slip}, \text{pk}(SK_A))/x_3\}, \\ \sigma_4 &= \{\text{req}(x_2, \text{hash}(\text{proj.1}(x_1), Desc), x_3)/x_4\}, \\ \sigma_5 &= \{\text{resp}(Code, \text{enc}(Code, \text{hash}(\text{common}), SK_A))/x_5\}, \\ \sigma_6 &= \{\text{confirm}(\text{proj.1}(x_5), \text{proj.2}(x_5))/x_6\}. \end{aligned}$$

Three parties are defined as follows: P_B stands for the buyer, P_S for the seller and P_A for the acquirer. In addition, $F_B(x), F_S(x), F_A(x)$ represent respectively the actions which the buyer, the seller and the acquirer are

required to perform after finishing the protocol flows. The processes are:

$$\begin{aligned}
 P &\equiv (\nu Authprice, Desc).P_B | P_S | P_A, \\
 P_B &\equiv (\nu Salt_B, R_B).(\overline{C_{BS}}\langle x_1\sigma_1 \rangle | !(C_{SB}(x_2). \\
 &\quad \text{if proj.5}(x_2) = \text{hash}(Authprice, \text{proj.1}(x_2), \text{proj.2}(x_2), \\
 &\quad \text{proj.3}(x_2), \text{hash}(R_B, Ban_B), \text{hash}(Salt_B, Desc)) \\
 &\quad \text{then } (\overline{C_{BS}}\langle x_3\sigma_3 \rangle | !(C_{SB}(x_6).\text{if checksig}(\text{proj.2}(x_6), \\
 &\quad \text{pk}(SK_A)) \text{ then } F_B(x_6))))), \\
 P_S &\equiv (\nu ID_S, TID_S, Date, Nonce_S).(C_{BS}(x_1). \\
 &\quad (\overline{C_{SB}}\langle x_2\sigma_2 \rangle | C_{BS}(x_3).(\overline{C_{SA}}\langle x_4\sigma_4 \rangle | C_{AS}(x_5). \\
 &\quad \overline{C_{SB}}\langle x_6\sigma_6 \rangle.F_S(x_5))))), \\
 P_A &\equiv (\nu SK_A).!(C_{SA}(x_4).\text{if proj.2}(\text{dec}(\text{proj.3}(x_4), \\
 &\quad SK_A)) = \text{hash}(\text{proj.1}(\text{dec}(\text{proj.3}(x_4), SK_A)), \\
 &\quad \text{proj.1}(\text{proj.1}(x_4)), \text{proj.2}(\text{proj.1}(x_4)), \\
 &\quad \text{proj.3}(\text{proj.1}(x_4)), \text{proj.4}(\text{proj.1}(x_4)), \\
 &\quad \text{hash}(\text{proj.3}(\text{dec}(\text{proj.3}(x_4), SK_A)), \\
 &\quad \text{proj.4}(\text{dec}(\text{proj.3}(x_4), SK_A))), \text{proj.1}(x_4)) \\
 &\quad \text{then } (\nu Code).(\overline{C_{AS}}\langle x_5\sigma_5 \rangle.F_A(x_4))).
 \end{aligned}$$

Buyers want to keep anonymous from eavesdroppers and sellers. Furthermore, buyers may even want to keep anonymous from the payment system provider (acquirers). We will show that iKP does not offer anonymity with respect to the payment system provider. It does minimize the exposure of buyers' identities to sellers and eavesdroppers. The crucial point is whether the Ban_B (Buyer B 's Account Number) in P_B can be exposed to the environment. The results are formally stated in the following theorem.

Theorem 3 (Buyers' Anonymity). *Suppose $F_A(x_1) \approx_o^e F_A(x_2)$ for every e if and only if $x_1 = x_2$ and the same property holds for $F_B(x)$ and $F_S(x)$. Suppose $P_B(x)$ be the process P_B having x substituted for Ban_B and $B_1 \neq B_2$. Let $v = \{C_{SB}, C_{BS}\}$, $v' = v \cup \{C_{AS}, C_{SA}\}$ and $p = \langle \emptyset, \emptyset \rangle$. If we abbreviate $(\nu Authprice, Desc).P_B(x) | P_S$ to $P_{BS}(x)$, then we have*

$$P_{BS}(B_1) \approx_o^{(v,p)} P_{BS}(B_2),$$

and

$$P_{BS}(B_1) | P_A \not\approx_o^{(v',p)} P_{BS}(B_2) | P_A.$$

sketch: Let us play the bisimulation game using our open approach. In each step, we update or extend the environment accordingly in order to record the knowledge exposed to the environment faithfully. We find that Ban_B only occurs in the frame of processes. After $P_{BS}(B_1)$ and $P_{BS}(B_2)$ have fired their respective output actions $\overline{C_{BS}}\langle x_3\sigma_3 \rangle$,

$$\begin{aligned}
 \phi(P_{BS}(B_1)) = \\
 \{ \text{enc}((Authprice, \text{hash}(\text{common}), B_1, R_B), \text{pk}(SK_A))/x_3 \} \\
 | \{ (Salt_B, \text{hash}(R_B, B_1))/x_1 \} | \dots,
 \end{aligned}$$

and

$$\begin{aligned}
 \phi(P_{BS}(B_2)) = \\
 \{ \text{enc}((Authprice, \text{hash}(\text{common}), B_2, R_B), \text{pk}(SK_A))/x_3 \} \\
 | \{ (Salt_B, \text{hash}(R_B, B_2))/x_1 \} | \dots.
 \end{aligned}$$

It is obvious that

$$\phi(P_{BS}(B_1)) \approx_s \phi(P_{BS}(B_2)),$$

then

$$P_{BS}(B_1) \approx_o^{(v,p)} P_{BS}(B_2)$$

follows easily.

For the case in which acquirers are taken account of, $\text{proj.3}(\text{dec}(x_3, K_A)) = B_1$ is deducible in $\phi(P_{SB}(B_1) | P_A)$, but is not deducible in $\phi(P_{SB}(B_2) | P_A)$. So we have

$$\phi(P_{BS}(B_1) | P_A) \not\approx_s \phi(P_{BS}(B_2) | P_A).$$

This suggests us that the attack hidden in the environment can tell the difference of two account numbers. By the assumption that

$$F_A(x_1) \not\approx_o^e F_A(x_2)$$

for all e if $x_1 \neq x_2$, we have

$$P_{SB}(B_1) | P_A \not\approx_o^{(v',p)} P_{BS}(B_2) | P_A. \quad \blacksquare$$

B. Electronic Cash

Ferguson's electronic cash protocol [32] consists of three types of participants: the bank B , the payer P and the shop S . This protocol employs a randomized blind signatures scheme to ensure that the payer can obtain the bank's signature on the coin, while the bank has no knowledge of the coin. It has three stages:

- **Withdraw Stage:** The payer blinds the coin c and then sends this blinded coin together with its identity U to the bank. The bank signs on the blinded coin, generates two blinded signatures s_1 and s_2 and sends them back to the payer. The payer unblinds these two signatures.
- **Payment Stage:** The payer sends the coin c to the shop. The shop sends the payer a randomly chosen (non-zero) challenge x . The payer computes a response and a signature, sends them back. The shop verifies the signature and then completes the trade with the payer.
- **Deposit Stage:** The shop sends the coin c , the challenge x and the response to the bank. The bank checks whether the coin is valid, informs the shop if it is valid.

We now model this protocol in Applied Pi. The signature Σ of it is:

$$\begin{aligned}
 \Sigma = \{ \text{blind}, \text{unblind}, \text{pk}, \text{blindsignA}, \text{blindsignB}, \\
 \text{sign}, \text{checksign}, \text{hash1}, \text{hash2}, \text{hash3}, \text{resp}_r, \\
 \text{resp}_s, \text{reveal}, \text{proj.i} \}.
 \end{aligned}$$

Most of the function symbols in Σ are self-evident. Others can be explained by the equation system. The following equation system reveals their relationship:

$$\begin{aligned}
& \text{proj.}i(x_1, \dots, x_l) = x_i, \quad 1 \leq i \leq l \\
& \text{checksign}(M, \text{sign}(M, sk), \text{pk}(sk)) = \text{true} \\
& \text{unblind}(\text{blindsignA}(\text{blind}(coin, k, g), U, sk), g) \\
& = \text{sign}(\text{hash1}(coin, k, U), sk) \\
& \text{unblind}(\text{blindsignB}(\text{blind}(coin, k, g), U, sk), g) \\
& = \text{sign}(\text{hash2}(coin, k, U), sk) \\
& \text{reveal}(\text{resp}_r(x_1, k, U), \text{resp}_r(x_2, k, U)) \\
& = U, \text{ where } x_1 = x_2 \\
& \text{resp}_s(x, \text{sign}(\text{hash1}(coin, k, U), sk), \text{sign}(\text{hash2}(coin, \\
& k, U), sk)) = \text{sign}(\text{hash3}(c, x, \text{resp}_r(x, k, U)), sk)
\end{aligned}$$

Here $\text{resp}_s(x, s_1, s_2)$ denotes a response over challenge x and two signature s_1 and s_2 which are obtained from the bank. Function $\text{resp}_r(x, k, U)$ stands for a response over challenge x , and reveal is a special function to facilitate detecting spender's identity.

The system consists of three participants: the payer P , the shop S and the bank B . Let $F_P(x), F_S(x), F_B(x)$ be respectively the actions which the payer, the shop and the bank are required to perform after finishing the protocol flows. We give the processes one after another.

$$\begin{aligned}
P \equiv & \nu k. \nu g. \nu c. \bar{c}_1 \langle (\text{blind}(c, k, g), U) \rangle. c_2(x_1). \\
& \text{if checksign}(\text{hash1}(c, k, U), \text{unblind}(\text{proj.1}(x_1), g), \\
& Pub_B) = \text{true then} \\
& \text{if checksign}(\text{hash2}(c, k, U), \text{unblind}(\text{proj.2}(x_1), g), \\
& Pub_B) = \text{true then} \\
& \bar{c}_{pay} \langle c \rangle. c_3(x_2). \\
& \bar{c}_4 \langle (\text{resp}_r(x_2, k, U), \text{resp}_s(x_2, \text{unblind}(\text{proj.1}(x_1), g), \\
& \text{unblind}(\text{proj.2}(x_1), g))) \rangle. F_P(x_2)
\end{aligned}$$

The payer P generates a fresh random number k , a fresh blind factor g , and a coin c . Next, the payer blinds the coin c with k and g , and then sends this blinded coin together with its identity U to the bank, expecting two corresponding signatures. Finally, the payer sends the coin c to the shop and accomplishes a challenge-response procedure with the shop.

$$\begin{aligned}
S \equiv & \nu x. c_{pay}(x_1). \bar{c}_3 \langle x \rangle. c_4(x_2). \\
& \text{if checksign}(\text{hash3}(x_1, x, \text{proj.1}(x_2)), \text{proj.2}(x_2), \\
& Pub_B) = \text{true then} \\
& \bar{c}_5 \langle (x_1, x, \text{proj.1}(x_2), \text{proj.2}(x_2)) \rangle. c_{payOK}(x_3). \\
& F_S(x_3)
\end{aligned}$$

In order to determine whether a paid coin is valid, the shop S initiates the challenge-response procedure with the payer. Then the shop sends the coin and the result of the challenge-response procedure to the bank to deposit this coin.

$$\begin{aligned}
B \equiv & \nu Prv_B. (\{\text{pk}(Prv_B)/Pub_B\} \mid ! (c_1(x_1). \\
& \bar{c}_2 \langle (\text{blindsignA}(\text{proj.1}(x_1), \text{proj.2}(x_1), Prv_B), \\
& \text{blindsignB}(\text{proj.1}(x_1), \text{proj.2}(x_1), Prv_B)) \rangle \\
& \mid (c_5(x_2). \text{if checksign}(\text{hash3}(\text{proj.1}(x_2), \text{proj.2}(x_2), \\
& \text{proj.3}(x_2)), \text{proj.4}(x_2), Pub_B) = \text{true} \\
& \text{then } \bar{c}_{payOK} \langle OK \rangle \rangle. F_B(x_2))
\end{aligned}$$

After receiving a blinded coin from the payer, the bank B sends back two blind signatures, and debits one dollar from the payers account at the same time. When the shop requests to verify the validity of a coin, the bank first verifies the correctness of signature, and then deposits one dollar to the shops account.

The whole system is

$$Sys \equiv P \mid S \mid B.$$

In this protocol, the anonymity we are concerned with is that eavesdroppers cannot tell apart those by whom a specific coin is spent. This is captured formally by the following theorem.

Theorem 4. *Suppose $F_P(x_1) \approx_o^e F_P(x_2)$ for every e if and only if $x_1 = x_2$ and the same property holds for $F_S(x)$ and $F_B(x)$. Let $P(x, y)$ be the process P having x substituted for c and y for U , and $coin_1 \neq coin_2$. Let $v = \{\bar{c}\}$, $p = \langle \emptyset, \emptyset \rangle$ and*

$$\begin{aligned}
P_1 & \equiv P(coin_1, payer_1), \\
P_2 & \equiv P(coin_2, payer_2).
\end{aligned}$$

We have

$$P_1 \mid P_2 \approx_o^{(v,p)} P_1 \{coin_2/coin_1\} \mid P_2 \{coin_1/coin_2\}.$$

Proof sketch: The crucial observation of it is that

$$P(coin_1, U) \approx_o^{(v,p)} P(coin_2, U).$$

Therefore we have both

$$P_1 \approx_o^{(v,p)} P_1 \{coin_2/coin_1\}$$

and

$$P_2 \approx_o^{(v,p)} P_2 \{coin_1/coin_2\}.$$

By Theorem 2, the following holds:

$$P_1 \mid P_2 \approx_o^{(v,p)} P_1 \{coin_2/coin_1\} \mid P_2 \{coin_1/coin_2\}. \quad \blacksquare$$

V. CONCLUSION

In this paper we propose a knowledge aware open bisimulation relation for the Applied Pi Calculus. Our relation is suitable for analyzing security protocols. It also has many other advantages, which include the congruent property and a finer distinguishability. By analyzing the anonymity of iKP and electronic cash protocol, it shows

that the knowledge aware bisimulation game can effectively analyze anonymity, which is hard to obtain by many other methods.

As for the future work, we would like to apply this approach to check other security properties, such as authentication and non-repudiation. We are also working on developing automatically equivalence checking tools.

ACKNOWLEDGEMENT

The authors would like to express their gratitude to Prof. Yuxi Fu for his inspiration, guidance and many fruitful discussions on this topic. The authors are also indebted to members of Basics Lab for their proofreading of this paper and many useful suggestions on improvements. The authors also thank anonymous referees for their comments and suggestions.

REFERENCES

- [1] M. Abadi and C. Fournet, "Mobile Values, New Names, and Secure Communication," in *Proceedings of POPL'01*. ACM Press New York, NY, USA, 2001, pp. 104–115.
- [2] R. Milner, J. Parrow, and D. Walker, "A Calculus of Mobile Processes, part I," *Information and Computation*, vol. 100, no. 1, pp. 1–40, 1992.
- [3] —, "A Calculus of Mobile Processes, part II," *Information and Computation*, vol. 100, no. 1, pp. 41–77, 1992.
- [4] M. Abadi and A. Gordon, "A Calculus for Cryptographic Protocols: The Spi Calculus," *Information and Computation*, pp. 1–70, 1999.
- [5] M. Abadi, B. Blanchet, and C. Fournet, "Just Fast Keying in the Pi Calculus," in *Proceedings of ESOP'04*, ser. LNCS, vol. 2986. Springer, 2004, pp. 340–354.
- [6] M. Abadi and B. Blanchet, "Computer-assisted verification of a protocol for certified email," *Proceedings of SAS'03, LNCS 2694*, pp. 316–335, 2003.
- [7] C. Fournet and M. Abadi, "Hiding Names: Private Authentication in the Applied pi Calculus," in *Proceedings of ISSS'02*, ser. LNCS, vol. 2609. Springer, 2002, pp. 317–338.
- [8] D. Sangiorgi, "A Theory of Bisimulation for the π -Calculus," *Proceedings of CONCUR'93*, pp. 127–142, 1993.
- [9] S. Briaies and U. Nestmann, "Open Bisimulation, Revisited," *Proceedings of the 12th Expressiveness in Concurrency (EXPRESS'05)*, vol. 154, no. 3, pp. 109–123, 2006.
- [10] C. Hoare, "Communicating Sequential Processes," *Communications of the ACM*, vol. 21, no. 8, pp. 666–677, 1978.
- [11] R. Milner, "A Calculus of Communicating Systems," *LNCS 92*, 1980.
- [12] J. Baeten and W. Weijland, *Process Algebra*. Cambridge University Press New York, NY, USA, 1991.
- [13] J. Bergstra and J. Klop, "Process Algebra for Synchronous Communication," *Information and Control*, vol. 60, pp. 109–137, 1984.
- [14] R. Milner, *Communicating and Mobile Systems: The π -Calculus*. Cambridge University Press, 1999.
- [15] Y. Fu, "The χ -Calculus," in *Proceedings of the 1997 Conference on Advances in Parallel and Distributed Computing*, 1997, pp. 74–81.
- [16] L. Cardelli and A. Gordon, "Mobile Ambients," *Theoretical Computer Science*, vol. 240, no. 1, pp. 177–213, 2000.
- [17] Y. Fu, "Fair Ambients," *Acta Informatica*, vol. 43, no. 8, pp. 535–594, 2007.
- [18] G. Lowe, "Breaking and Fixing the Needham-Schroeder Public-Key Protocol Using FDR," *Software - Concepts and Tools*, vol. 17, no. 3, pp. 93–102, 1996.
- [19] Z. Luo, X. Cai, J. Pang, and Y. Deng, "Analyzing Electronic Cash Protocols Using the Applied Pi Calculus," in *Proceedings of ACNS'07*, ser. LNCS, vol. 4521. Springer, 2007, pp. 87–103.
- [20] M. Abadi and A. Gordon, "A Bisimulation Method for Cryptographic Protocols," *Nordic Journal of Computing*, vol. 5, no. 4, pp. 267–303, 1998.
- [21] J. Borgström, S. Briaies, and U. Nestmann, "Symbolic Bisimulation in the Spi Calculus," in *Proceedings of CONCUR'04*, ser. LNCS, vol. 3170. Springer, 2004, pp. 161–176.
- [22] D. Chaum, "The dining cryptographers problem: Unconditional sender and recipient untraceability," *Journal of Cryptology*, pp. 65–75, 1988.
- [23] S. Schneider and A. Sidiropoulos, "CSP and anonymity," *Proceedings of ESORICS 1996*, 1996.
- [24] S. Kremer and M. Ryan, "Analysis of an Electronic Voting Protocol in the Applied Pi-calculus," in *Proceedings of ESOP'05*, ser. Lecture Notes in Computer Science, vol. 3444. Springer, 2005, pp. 186–200.
- [25] T. Chothia, "Analysing the MUTE anonymous file-sharing system using the pi-calculus," *Proceedings of FORTE 2006*, 2006.
- [26] T. Chothia, S. Orzan, J. Pang, and M. Dashti, "A framework for automatically checking anonymity with μ CRL," *Proceedings of TGC 2006*, 2006.
- [27] H. Zhu and X. Cai, "An open approach for the applied pi calculus," Shanghai Jiao Tong University, Tech. Rep., 2008.
- [28] X. Cai, Y. Fu, Y. Deng, and Z. Luo, "The Applied Pi Calculus Revisited," 2008, submitted.
- [29] M. Abadi and V. Cortier, "Deciding Knowledge in Security Protocols under Equational Theories," in *Proceedings of ICALP'04*, ser. LNCS 3442. Springer, 2004, pp. 46–58.
- [30] M. Bellare, J. Garay, R. Hauser *et al.*, "Design, implementation, and deployment of the iKP secure electronic payment system," *IEEE Journal of Selected Areas in Communications*, vol. 18, pp. 611–627, 2000.
- [31] —, "iKP — a family of secure electronic payment protocols," *Proceedings of 1st USENIX Workshop on Electronic Commerce*, 1995.
- [32] N. Ferguson, *Single Term Off-line Coins*. Springer, 1993.

Han Zhu was born in 1981, Shanghai, China.

He got his Bachelor of Science from Shanghai Jiao Tong University in 2003. He now is a candidate of Ph.D in Department of CS in Shanghai Jiao Tong University, supervised by Prof. Yuxi Fu. His main research interests include process algebra, security protocols and formal methods.

Yonggen Gu was born in 1968, Huzhou, China.

He got his Bachelor of Science from Hangzhou University, and his master and doctor degree in Computer Science from Shanghai Jiao Tong University in 1996 and 2006 respectively. He now is a professor of Huzhou Teachers College. His main research interests include process algebra, security protocols, zero-knowledge proof and formal methods.

Xiaohong Wu was born in 1974, Huzhou, China.

She is a professor of Huzhou Teachers College. Her main research interests include process algebra, security protocols, zero-knowledge proof and formal methods.