Verifiable Multi-secret sharing in the applied pi-calculus

Hui Zhao† Mingchu Li† Yizhi Ren‡ Kouichi Sakurai‡

†School of Software, Dalian University of Technology
Dalian, 116621, P.R.China
‡ Dept. of Informatics, Kyushu University
744 Motooka, Nishi-ku, Fukuoka 819-0395, Japan

Abstract In this paper, we define an abstraction of verifiable multi-secret sharing protocols that is accessible to a fully mechanized analysis. The abstraction is formalized within the applied pi-calculus using a novel equational theory that abstractly characterizes the cryptographic semantics of secret share. We present an encoding from the equational theory into a convergent rewriting system that is suitable for the automated protocol verifier ProVerif.

1 Introduction

Pi-calculus are now widely considered a particularly salient approach for formally analyzing security protocols, dating back to Abadi's seminal work on secrecy by typing [1]. One of the central challenges in the analysis of complex and industrial-size protocols is the expressiveness of the formalism used in the formal analysis and its capability to model complex cryptographic operations. While such protocols traditionally relied only on the basic cryptographic operations such as encryption and digital signatures, modern cryptography has invented more sophisticated primitives with unique security features that go far beyond the traditional understanding of cryptography to solely offer secrecy and authenticity of a communication. Secret share constitute a prominent such primitive.

In 1994, Dawson et al. [4] propose multi-secret sharing (MSS) schemes. In such schemes, several secrets can be shared during one secret sharing process. In 2004, Yang et al. (YCH) [5] propose a new MSS, which is based on two-variable one-way function and allows to reconstruct several secrets parallelly. In 2005, Shao and Cao (SC) [6] propose an efficient verifiable multi-secret sharing based on YCH and feldman' schemes. In 2006, Zhao et al. (ZZZ) [7] propose a practical verifiable multi-secret sharing based on YCH and Hwang-Chang (HC) schemes [5].

Due to the complexity of verifiable multi-secret sharing scheme, it is very difficult to devise the abstraction of secret-sharing proof which can hold all the security features above.

Our main contributions are as follows: First, we present an abstraction of verifiable multi-secret sharing schemes within the applied pi-calculus [2] using a novel equational theory that abstractly characterizes the cryptographic semantics of secret-sharing proofs. On the basis of that, we transform our abstraction into an equivalent formalization that is accessible to ProVerif [3], a well-established tool for the mechanized analysis of different security properties.

2 Review of the Pi-calculus

The syntax of the applied pi-calculus [3] is given as follow. Terms are defined by means of a *signature* Σ, which consists of a set of function symbols, each with an arity. The set of terms T_Σ is the free algebra built from names, variables, and function symbols in Σ applied to arguments. We let u range over names and variables. Terms are equipped with an equational theory E, i.e., an equivalence relation on terms that is closed under substitution of terms and under application of term contexts (terms with a hole). We write $E \vdash M = N$ and $E \vdash M \neq N$ for an equality and an inequality, respectively, modulo E.

The grammar of processes (or *plain processes*) is defined as follows. The null process **0** does nothing; $\nu n.P$ generates a fresh name n and then behaves as P; if $M = N$ then P else Q behaves as P if $E \vdash M = N$, and as Q otherwise; $u(x).P$ receives a message N from the channel u and then behaves as $P\{N/x\}$; $\overline{u}\langle N\rangle.P$ outputs the message N on the channel u and then behaves as P; $P \mid Q$ executes P and Q in parallel; $!P$ generates an unbounded number of copies of P.

Extended processes are plain processes extended with *active substitutions*. An active substitution $\{M/x\}$ is a floating substitution that may apply to any processes that it comes in contact with. To control the scope of the active substitutions, we can restrict the variable x. intuitively, $\nu x.(P \mid \{M/x\})$ constrains the scope of $\{M/x\}$ to process P. If the variable x is not restricted, as it is the glasses case in the process $(P \mid \{M/x\})$, then the substitution is exported by the process and the environment has immediate access to M. As usual, the scope of names and variables is delimited by restrictions and by inputs. We write $fv(A)$ and $fn(A)$ to denote the free variables and names in an extended process A and we write $bv(A)$ and $bn(A)$ to denote the bound variables and names in an extended process A, respectively. We let $free(A) := fv(A) \cup fn(A)$ and $bound(A) := bv(A) \cup bn(A)$. For sequence $\widetilde{M} = M_1, \ldots, M_k$ and $\tilde{x} = x_1, \ldots x_k$, we let $\{\widetilde{M}/\tilde{x}\}$ denote $\{M_1/x_1\} \mid \ldots \mid \{M_k/x_k\}$. We always assume that substitutions are cycle-free, that extended processes conation at most one substitution for each variable and that extended processes contain exactly one substitution for each restricted variable.

A *context* is a process or an extended process with a hole. An evaluation *context* is a context without private function symbols whose hole is not under a replication, a conditional, an input, or an output. A context $C[\_]$ closes A if $C[A]$ is closed. A frame is an extended process built up from 0 and active substitutions by parallel composition and restriction. We let ϕ and ψ range over frames. The domain $dom(\phi)$ of a frame ϕ is the set of variables that ϕ exports, i.e., those variables x for which ϕ contains a substitution $\{M/x\}$ not under a restriction on x. Every extended process A can be mapped to a frame $\phi(A)$ by replacing every plain process embedded in A with 0.

Definition 1 (Structural Equivalence) *Structural equivalence* ($\equiv$) *is the smallest* relation *on extended processes that satisfies the rules in Table 2 and that is closed under* α *-renaming, i.e., renaming of bound names and variables, and under application of evaluation contexts.*

Definition 2 (Internal Reduction) *Internal reduction* ($\rightarrow$) *is the smallest relation on extended processes that satisfies the rules in Table 3 and that is closed under structural equivalence and under application of evaluation contexts.*

We write $A \Downarrow \alpha$ to *denote* that A can send a message on α, i.e., $A \rightarrow^* C[\overline{a}\langle M\rangle.p]$ for some evaluation context $C[\_]$ that does not bind a.

Definition 3 (Observational Equivalence) *Observational equivalence* ($\approx$) *is the largest symmetric relation R between* closed *extended processes with the same domain such that ARB implies*:

1) *if* $A \Downarrow \alpha$, *then* $B \Downarrow \alpha$;
2) *if* $A \rightarrow^* A'$, *then* $B \rightarrow^* B'$ *and* $A' R B'$ *for some* B';
3) $C[A] R C[B]$ *for all closing evaluation contexts* C[_].

3 An Equational Theory of Secret-sharing

Our equational theory is explained in the following. Secret-sharing process with threshold (l, t) is represent as a term $ssp(\tau, l, t)$, name τ is used to identify specified secret-sharing process, we abuse notation by writing $\tau_{l,t}$ which represents $ssp(\tau, l, t)$; The secret key for secret share is represent as a term of $SSK_{i,j,k}(\widetilde{M}, m, \tau_{l,t}, \widetilde{F})$, where $\widetilde{M}$, called dealer parameters, denote sequence $M_1 \ldots M_i$ of terms; while m, called the proof's *identity Id*, can be used to identify different secret key in same secret-sharing process and $m \le l$. The secret share is represent as a term of form $SS_{i,j,k}(\widetilde{N}, SSK_{i,j,k}(\widetilde{M}, m, \tau_{l,t}, \widetilde{F}), \widetilde{F})$, where $\widetilde{N}$, called player parameters, denote sequence $N_1 \ldots N_j$ of terms; $\widetilde{F}$ denote sequence $F_1 \ldots F_k$ of (i, j)-formulas which constitute a formula over i dealer parameters and j player parameters, see below; Hence, $SS_{i,j,k}$ is a function of arity j+k+1 and $SSK_{i,j,k}$ is a function of arity i+k+2.

The formula F constitutes a constant without names and variables, which is built upon distinguished nullary functions α_i and β_i with $i \in \mathrm{N}$.

Definition 4 ((i, j)-formulas**)** *We call a* term *an* (i, j)*-formula if the term contains neither names nor variables*, *and* if *for every* α_m *and* β_n *occurring therein*, *we have* $m \in [1, i]$ *and* $n \in [1, j]$.

The values α_i and β_j in F constitute placeholders for the terms M_i and N_j. For instance,

$SS_{1,1,1}(h(m), SSK_{1,1,1}(SK, 1, \tau_{3,2}, F), F)$; $F = sign(\beta_1, \alpha_1)$; denotes a secret share of the term $sign(h(m), SK)$ which is a signature of m with SK in a Secret-sharing scheme with threshold (3, 2). More precisely, the statement reads: "The dealer use $SSK_{1,1,1}(SK, 1, \tau_{3,2}, F)$ to start a Secret-sharing process with threshold (3, 2) according to dealer parameter SK. Then players use $SS_{1,1,1}(h(m), SSK_{1,1,1}(SK, 1, \tau_{3,2}, F), F)$ to create secret share according to player parameter $h(m)$. Finally, $sign(h(m), SK)$ can be computed from 2 secret shares of the Secret-sharing process". Since each player does not own SK, $sign(h(m), SK)$ is kept the secret to him.

The verification key share for a secret share is representing as a term of form $SVK_{i,j,k}(\widetilde{M}, m, \tau_{l,t}, \widetilde{F})$ and we have $m \le l$. Verification of a secret share with respect to a secret share verification key is modelled as a function $SVer$ of arity 3 that is defined by the following equational rule:

$SVer_{i,j,k}(SVK_{i,j,k}(\widetilde{M}, m, \tau_{l,t}, \widetilde{F}), SS_{i,j,k}(\widetilde{N}, SSK_{i,j,k}(\widetilde{M}, m, \tau_{l,t}, \widetilde{F}), \widetilde{F}), \widetilde{F}) = true$.

Combination of secret shares with respect to (i, j)-formulas is modelled as function $SCombin_{i,j,k,r}$ of arity r+k that is defined by the following equational rules:

$SCVer_{i,j,k,r}(SS_{i,j,k}(\widetilde{N}, SSK_{i,j,k}(\widetilde{M}, i_1, \tau_{l,t}, \widetilde{F}), \widetilde{F}), \ldots, SS_{i,j,k}(\widetilde{N}, SSK_{i,j,k}(\widetilde{M}, i_r, \tau_{l,t}, \widetilde{F}), \widetilde{F}), \widetilde{F})) = true$ *iff*

1) $i_m \ne i_n$ *for* $1 \le m, n \le r$ *and* $m \ne n$;
2) $r \ge t$.

$Secret_p(SCombin_{i,j,k,r}(ss_1, \ldots, ss_r, \widetilde{F})) = \widetilde{F}_p\{\widetilde{M} / \alpha\}\{\widetilde{N} / \beta\}$ *iff*

1) $SCVer_{i,j,k,r}(ss_1, \ldots, ss_r, \widetilde{F}) = true$;
2) $p \le k$.

This rules guarantee in the abstract model the soundness and correctness of Secret-sharing protocols with threshold (l, t) that knowledge of any t-1 or fewer secret shares leaves secret completely undetermined and knowledge of any k or more secret shares make secret easily computable. We shall often omit arities and write this statement as $SCombin(\widetilde{M})$.

4 Towards a Mechanized Analysis of Secret-sharing Knowledge

The equational theory E_{SS} defined in the previous section is not suitable for existing tools for mechanized security protocol analysis. The reason is that the number of possible formulas, and thus the number of equational rules in E_{SS}, is infinite. In this section, we specify an equivalent equational theory in terms of a convergent rewriting system. This theory turns out to be suitable for Proverif [3].

4.1 A Finite Specification of Secret-sharing

The central idea of our equivalent finite theory is to focus on the secret shares used within the process specification and to abstract away from the additional ones that are possibly generated by the environment. This makes finite the specification of the equational theory.

Pinning down this conceptually elegant and appealing idea requires to formally characterizing the secret share generated and combined in the process specification. First, we track the secret share generated or combined in the process specification by a set *TR* of triples of the form $(i, j, k, \tilde{F})$, where $\tilde{F}$ is sequence of k (i, j)-formulas of Secret-sharing scheme. Second, we record the arity h, g, p, q of the largest used in the process specification. For terms M and processes P, we let *terms*(M) denote the set of subterms of M and *terms*(P) denote the set of terms in P. We can now formally define the notion of (*TR*, h, g, p, q)-validity of terms and processes.

Definition 5 (**Term Validity**) *A term Z is* (*TR*, h, g, p, q)*-valid if and only if the following* conditions *hold:*

1) *For every* $SSK_{i,j,k}(\tilde{M}, M, N, \tilde{F})$, $SVK_{i,j,k}(\tilde{M}, M, N, \tilde{F})$, $SS_{i,j,k}(\tilde{M}, M, \tilde{F})$, $SVer_{i,j,k}(M, N, \tilde{F})$, $SCVer_{i,j,k,r}(\tilde{M}, \tilde{F})$ *and* $SCombin_{i,j,k,r}(\tilde{M}, \tilde{F}) \in terms(Z)$, *we have*
   a). $(i, j, k, \tilde{F}) \in TR$,
   b). *for every* $(i, j, k, \tilde{F}') \in TR$ *such that* $E_{SS} \to \tilde{F}' = \tilde{F}$, *we have* $\tilde{F}' = \tilde{F}$.
2) *for every* $l \in \mathrm{N}$, α_l *and* β_l *occur in Z only inside of* (i, j)*-formula of* Z.
3) *for every* $(i, j, k, \tilde{F}) \in TR$, *we have* $i \in [0, h]$, $j \in [0, g]$ *and* $k \in [0, p]$.
4) *for every* $SCVer_{i,j,k,r}(SS_{i,j,k}(\tilde{N}, SSK_{i,j,k}(\tilde{M}, i_1, \tau_{l,t}, \tilde{F}), \tilde{F}), \ldots, SS_{i,j,k}(\tilde{N}, SSK_{i,j,k}(\tilde{M}, i_r, \tau_{l,t}, \tilde{F}), \tilde{F}), \tilde{F}) \in terms(Z)$, *we have*
   a). $r \le q$,
   b). $i_m \neq i_n$ *for* $1 \le m, n \le r$ *and* $m \neq n$.
5) *for every* $SCombin_{i,j,k,r}(SS_{i,j,k}(\tilde{N}, SSK_{i,j,k}(\tilde{M}, i_1, \tau_{l,t}, \tilde{F}), \tilde{F}), \ldots, SS_{i,j,k}(\tilde{N}, SSK_{i,j,k}(\tilde{M}, i_r, \tau_{l,t}, \tilde{F}), \tilde{F}), \tilde{F}) \in terms(Z)$, *we have*
   a). $r \le q$,
   b). $i_m \neq i_n$ *for* $1 \le m, n \le r$ *and* $m \neq n$;
6) *For every* $Secret_l(M) \in terms(Z)$, *we have* $l \le q$.

We check that each secret share generation, verification and combination is tracked in *TR* (condition 1). We also check that for all Secret-sharing proofs used in the process specification, the arity of dealer parameters, player parameters and (i, j)-formulas is less or equal than h, g and p, respectively (condition 3). Finally, we check that the arity of $SCVer^{\tilde{F}}_{i,j,k,r}$ and $SCombin^{\tilde{F}}_{i,j,k,r}$ used in the process specification is less or equal that q and only different secret shares in same secret-sharing process can be combined (condition 4, 5).

We now encode the Secret-sharing proof generated by the environment. These proofs are possibly different from the ones specified in the process. We include in the signature $E^{TR,h,g,p,q}_{SS}$ the function symbols $SSK^{\tilde{F}}_{i,j,k}$, $SVK^{\tilde{F}}_{i,j,k}$, $SS^{\tilde{F}}_{i,j,k}$, $SVer^{\tilde{F}}_{i,j,k}$, $SCVer^{\tilde{F}}_{i,j,k,r}$, $SCombin^{\tilde{F}}_{i,j,k,r}$. We then replace every term $SSK_{i,j,k}(\tilde{M}, M, N, \tilde{F})$, $SVK_{i,j,k}(\tilde{M}, M, N, \tilde{F})$, $SS_{i,j,k}(\tilde{M}, M, \tilde{F})$, $SVer_{i,j,k}(M, N, \tilde{F})$, $SCVer_{i,j,k,r}(\tilde{M}, \tilde{F})$, $SCombin_{i,j,k,r}(\tilde{M}, \tilde{F})$ with $SSK^{\tilde{F}}_{i,j,k}(\tilde{M}, M, N)$, $SVK^{\tilde{F}}_{i,j,k}(\tilde{M}, M, N)$, $SS^{\tilde{F}}_{i,j,k}(\tilde{M}, M)$, $SVer^{\tilde{F}}_{i,j,k}(M, N)$, $SCVer^{\tilde{F}}_{i,j,k,r}(\tilde{M})$ and $SCombin^{\tilde{F}}_{i,j,k,r}(\tilde{M})$ respectively. Since $\tilde{F}$ are uniquely determined by $SSK^{\tilde{F}}_{i,j,k}$, $SVK^{\tilde{F}}_{i,j,k}$, $SS^{\tilde{F}}_{i,j,k}$, $SVer^{\tilde{F}}_{i,j,k}$, $SCVer^{\tilde{F}}_{i,j,k,r}$ and $SCombin^{\tilde{F}}_{i,j,k,r}$,

it can be omitted from the protocol specification.

For finitely modeling the combination of secret-sharing, we include in $E_{SS}^{TR,h,g,p,q}$ the function $PCombin_{i,j,k,r}^{\tilde{F}}$.

Combination of r different secret shares is modeled by the following equational rules:

$SCombin_{i,j,k,r}^{\tilde{F}}(\widetilde{M}) = PCombin_{i,j,k,r}^{\tilde{F}}(\widetilde{M}, SCVer_{i,j,k,r}^{\tilde{F}}(\widetilde{M}))$;

$SCVer_{i,j,k,r}^{\tilde{F}}(SS_{i,j,k}^{\tilde{F}}(\widetilde{N}, SSK_{i,j,k}^{\tilde{F}}(\widetilde{M}, i_1, \tau_{l,t})), \ldots, SS_{i,j,k}^{\tilde{F}}(\widetilde{N}, SSK_{i,j,k}^{\tilde{F}}(\widetilde{M}, i_r, \tau_{l,t}))) = (t = r) \vee$

$SCVer_{i,j,k,r-1}^{\tilde{F}}(SS_{i,j,k}^{\tilde{F}}(\widetilde{N}, SSK_{i,j,k}^{\tilde{F}}(\widetilde{M}, i_1, \tau_{l,t})), \ldots, SS_{i,j,k}^{\tilde{F}}(\widetilde{N}, SSK_{i,j,k}^{\tilde{F}}(\widetilde{M}, i_{r-1}, \tau_{l,t})))$;

$SCVer_{i,j,k,1}^{\tilde{F}}(SS_{i,j,k}^{\tilde{F}}(\widetilde{N}, SSK_{i,j,k}^{F}(\widetilde{M}, i, \tau_{l,t}))) = (t = 1)$;

$Secret_p(PCombin_{i,j,k,r}^{\tilde{F}}(SS_{i,j,k}^{\tilde{F}}(\widetilde{N}, SSK_{i,j,k}^{\tilde{F}}(\widetilde{M} i_1, \tau_{l,t})), \ldots, SS_{i,j,k}^{\tilde{F}}(\widetilde{N}, SSK_{i,j,k}^{\tilde{F}}(\widetilde{M}, i_r, \tau_{l,t})), true)) = \tilde{F}_p\{\widetilde{M} / \alpha\}\{\widetilde{N} / \beta\}$;

The $PCombin_{i,j,k,r}^{\tilde{F}}$ functions are private, hence they cannot be used by the adversary.

4.2 Compilation into Finite Form

We now define the static compilation of term and processes. We first review these notions.

Definition 6 (Term Equality in Frame) *Two term M and N are equal in a frame ϕ, written $(M = N)\phi$, if and only if $\phi \equiv \nu\tilde{n}.\sigma$, $M\sigma \equiv N\sigma$, and $\{\tilde{n}\} \cap (fn(M) \cup fn(N)) = \varnothing$ for some name $\tilde{n}$ and substitution σ.*

The next definition introduces a normal form of terms. Intuitively, a term is in (TR, h, g, p, q)-normal form if the subterms generated by the environment cannot be further simplified.

Definition 7 (Normal Form) *A term M is in (TR, h, g, p, q)-normal form with respect to a frame ϕ if and only if the following conditions hold:*

1) *For every $SSK_{i,j,k}(\tilde{Z}, x, y, \tilde{F})$ and $SVK_{i,j,k}(\tilde{Z}, x, y, \tilde{F}) \in terms(M)$, we have that $E_{SS} \rightarrow x\phi = m$, $E_{SS} \rightarrow y\phi = \tau_{l,t}$ and $m \in \mathrm{N}, m \le l$, $(i, j, k, \tilde{F}) \in TR$;*
2) *For every $SS_{i,j,k}(\tilde{Z}, x, \tilde{F}) \in terms(M)$, we have that $E_{SS} \rightarrow x\phi = SSK_{i,j,k}(\widetilde{M}, m, \tau_{l,t}, \tilde{F})$, $(i, j, k, \tilde{F}) \in TR$;*
3) *For every $SVer_{i,j,k}(x, y, \tilde{F}) \in terms(M)$, we have that $E_{SS} \rightarrow x\phi = SS_{i,j,k}(\widetilde{N}, M, \tilde{F})$, $E_{SS} \rightarrow y\phi = SVK_{i,j,k}(\widetilde{M}, m, \tau_{l,t}, \tilde{F})$, $(i, j, k, \tilde{F}) \in TR$;*
4) *For every $SCVer_{i,j,k,r}(z_1, \ldots, z_r, \tilde{F}) \in terms(M)$, we have that $E_{SS} \rightarrow z_m\phi = SS_{i,j,k}(\widetilde{N}, SSK_{i,j,k}(\widetilde{M}, n_m, \tau_{l,t}, \tilde{F}), \tilde{F})$, $m = 1; 2 \ldots; r$, $(i, j, k, \tilde{F}) \in TR$ and $n_e \neq n_f$ iff $1 \le e, f \le r$, $e \neq f$.*
5) *For every $SCombin_{i,j,k,r}(z_1, \ldots, z_r, \tilde{F}) \in terms(M)$, we have that $E_{SS} \rightarrow z_m\phi = SS_{i,j,k}(\widetilde{N}, SSK_{i,j,k}(\widetilde{M}, n_m, \tau_{l,t}, \tilde{F}), \tilde{F})$, $m = 1; 2 \ldots; r$, $(i, j, k, \tilde{F}) \in TR$ and $n_e \neq n_f$ iff $1 \le e, f \le r$, $e \neq f$.*
6) *For every $Secret_l(x) \in terms(Z)$, we have that $E_{SS} \rightarrow x\phi = SCombin_{i,j,k,r}(M_1, \ldots, M_r, \tilde{F})$, $l \le k$.*

For any term there exists an equivalent term in normal form.

We now characterize the notion of validity of extended processes. Intuitively, an extended process is (TR, h, g, p, q)-valid process if it can be separated into an (TR, h, g, p, q)-valid process and a frame where free variables, referring to output messages, are associated to (TR, h, g, p, q)-valid terms, and bound variables, referring to input messages, are associated to terms in (TR, h, g, p, q)-normal form that only contain free names and free variables.

Definition 8 (Extended Process Validity) *A frame ϕ is (TR, h, g, p, q)-valid if and only if there exist $\tilde{n}$, $\tilde{y}$, $\{\tilde{Z} / \tilde{x}\}$, with $\tilde{y} \subseteq \tilde{x}$, such that the following conditions hold:*

1) *$\phi = \nu\tilde{n}.\nu\tilde{y}.\{\tilde{Z} / \tilde{x}\}$;*
2) *for every $x_k \in fv(\phi)$, we have that Z_k is (TR, h, g, p, q)-valid;*
3) *for every $x_k \in bv(\phi)$, we have that Z_k is in (TR, h, g, p, q)-normal form with respect to ϕ and $free(Z_k) \cap bound(\phi) = \varnothing$.*

An extended process A is (TR, h, g, p, q)-valid if and only if there exist $\tilde{n}$, $\tilde{y}$, $\{\tilde{Z} / \tilde{x}\}$, with $\tilde{y} \subseteq \tilde{x}$, such that the following conditions hold:

1) *$A = \nu\tilde{n}.\nu\tilde{y}.(\{\tilde{Z} / \tilde{x}\} \mid P)$*
2) *$\nu\tilde{n}.\nu\tilde{y}.\{\tilde{Z} / \tilde{x}\}$ is (TR, h, g, p, q)-valid.*
3) *P is (TR, h, g, p, q)-valid.*

We now introduce the static compilation of

terms at run-time.

Definition 9 (Static Compilation) *the* (*TR*, *h*, *g*, *p*, *q*)*-static compilation is the partial function* σ: $T_{\Sigma_{SS}} \rightarrow T_{\Sigma_{SS}^{TR,h,g,q}}$ *recursively defined as follows*:

$SSK_{i,j,k}(\widetilde{M},M,N,\widetilde{F})\sigma = SSK^{\widetilde{F}}_{i,j,k}(\widetilde{M}\sigma,M\sigma,N\sigma) \quad \forall(i,j,k,\widetilde{F}) \in TR$

$SVK_{i,j,k}(\widetilde{M},M,N,\widetilde{F})\ \sigma = SVK^{\widetilde{F}}_{i,j,k}(\widetilde{M}\sigma,M\sigma,N\sigma)\ \forall(i,j,k,\widetilde{F}) \in TR$

$SS_{i,j,k}(\widetilde{M},M,\widetilde{F})\ \sigma = SS^{\widetilde{F}}_{i,j,k}(\widetilde{M}\sigma,M\sigma) \quad \forall(i,j,k,\widetilde{F}) \in TR$

$SVer_{i,j,k}(M,N,\widetilde{F})\ \sigma = SVer^{\widetilde{F}}_{i,j,k}(M\sigma,N\sigma)\ \forall(i,j,k,\widetilde{F}) \in TR$

$SCVer_{i,j,k,r}(\widetilde{M},\widetilde{F})\ \sigma = SCVer^{\widetilde{F}}_{i,j,k,r}(\widetilde{M}\sigma) \quad \forall(i,j,k,\widetilde{F}) \in TR$

$SCombin_{i,j,k,r}(\widetilde{M},\widetilde{F})\sigma = SCombin^{\widetilde{F}}_{i,j,k,r}(\widetilde{M}\sigma)\ \forall(i,j,k,\widetilde{F}) \in TR$

$f(M_1,\ldots M_i)\sigma = f(M_1\sigma,\ldots M_i\sigma)$

$x\sigma = x$

$n\sigma = n$

In the following, for every (*TR*, *h*, *g*, *p*, *q*)*-valid* process $A = \nu\tilde{n}.\nu\tilde{y}.(\{\widetilde{M}/\tilde{x}\} \mid P)$, we can write $A\sigma$ to denote $\nu\tilde{n}.\nu\tilde{y}.(\{\widetilde{M}/\tilde{x}\} \mid P)\sigma$.

The next definition introduces the notion of similarity for frames.

Definition 10 (Frame Similarity) *two frame* ϕ *and* ψ *are similar, written* $\phi \sim \psi$, *if and only if the following conditions hold:*

1) *There exist TR, h, g, p and q such that* ϕ *and* ψ *be two* (*TR*, *h*, *g*, *p*, *q*)*-valid frames*;
2) $\phi = \nu\tilde{n}.\nu\tilde{y}.\{\widetilde{M}/\tilde{x}\}$ *and* $\psi = \nu\tilde{m}.\nu\tilde{y}.\{\widetilde{N}/\tilde{x}\}$;
3) *For every* $x_i \in bv(\phi)$, *we have* $M_i = N_i$.

The following theorem finally states that observational equivalence is preserved under static compilation and hence asserts the soundness of the encoding from the infinite specification into the finite specification.

Theorem 1 (Preservation of Observational Equivalence) *Let A and B be extended process such that* $A = \nu\tilde{n}.\nu\tilde{y}.(\{\widetilde{M}/\tilde{x}\} \mid P)$, $B = \nu\tilde{n}'.\nu\tilde{y}.(\{\widetilde{M}'/\tilde{x}\} \mid P')$, *for some* (*TR*, *h*, *g*, *p*, *q*)*-valid processes P and P' and* $\nu\tilde{n}.\nu\tilde{y}.\{\widetilde{M}/\tilde{x}\} \sim \nu\tilde{n}'.\nu\tilde{y}.\{\widetilde{M}'/\tilde{x}\}$. *Let* σ *be the* (*TR*, *h*, *g*, *p*, *q*)*-static compilation. If* $A\sigma \approx_{E_{SS}^{TR,h,g,p,q}} B\sigma$, *then* $A \approx_{E_{SS}} B$.

5 Conclusion

We have designed an abstraction of Secret-sharing protocols in the applied pi-calculus. A novel equational theory for terms characterizes the semantic properties of secret share. Additionally, we propose an encoding into a finite specification in terms of a convergent rewriting system that is accessible to a fully mechanized analysis. The encoding is sound and fully automated.

参考文献

[1] M. Abadi. Secrecy by typing in security protocols. Journal of the ACM, 46(5):749–786, 1999.

[2] M. Abadi, B. Blanchet, and C. Fournet. Just fast keying in the pi calculus. ACM Transactions on Information and System Security, 10(3):9, 2007.

[3] B. Blanchet. An efficient cryptographic protocol verifier based on Prolog rules. In Proc. 14th IEEE Computer Security Foundations Workshop (CSFW), pages 82–96. IEEE Computer Society Press, 2001.

[4] A. J. He, E. Dawson. Multistage secret sharing based on one-way function, Electronics Letters, 30(19):1591–1592, 1994.

[5] C. C. Yang, T. Y. Chang, M. S. Hwang. A (t,n) multi-secret sharing scheme, Applied Mathematics and Computation, 151: 483–490, 2004.

[6] J. Shao, Z. F. Cao. A new efficient (t,n) verifiable multi-secret sharing (VMSS) based on YCH scheme. Applied Mathematics and Computation, 168(1): 135–140, 2005.

[7] J. Zhao, J. Zhang, R. Zhao, A practical verifiable multi-secret sharing scheme, Computer Standards and Interfaces 29 (1): 138–141, 2007.