Mechanized Analysis of Multi-Secret Sharing Based on Lagrange Interpolating Polynomial in the Applied Pi-calculus

Zhong-hai Zhou, Jian Yuan, Hui Zhao

The Institute of Oceanographic Instrumentation, SDAS, Qingdao, 266001, P. R. China
Ocean University of China, Qingdao, 266001, P. R. China

Abstract

In this paper, we give an abstraction of multi-secret sharing schemes based on lagrange interpolating polynomial that is accessible to a fully mechanized analysis. The abstraction is formalized within the applied pi-calculus using an equational theory that abstractly characterizes the cryptographic semantics of secret share. Based on that, we verify the threshold certificate protocol in a convergent rewriting system suitable for the automated protocol verifier ProVerif.

Keywords: Protocol Verifier, Pi-calculus, Multi-Secret Sharing

1. Introduction

The mechanization of language-based security proofs has recently enjoyed substantial improvements that have further strengthened the position of language-based technologies as a promising approach for the analysis of complex and industrial-size cryptographic protocol. Owning to the fact that 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, 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. Secret share constitute such a prominent primitive.

Due to the complexity of multi-secret sharing schemes, it is very difficult to realize the mechanization of secret-sharing proof. In this paper, we give an abstraction of multi-secret sharing schemes based on lagrange interpolating polynomial that is accessible to a fully mechanized analysis. The abstraction is formalized within the applied pi-calculus using an equational theory that abstractly characterizes the cryptographic semantics of secret share. Based on that, we verify the threshold certificate protocol [4] in a convergent rewriting system suitable for the automated protocol verifier ProVerif [3].

2. Review of the Applied Pi-calculus.

The syntax of the applied pi-calculus [4] is given as follows. Terms are defined by means of a signature $\Sigma$, which consists of a set of function symbols, each with an arity. The set of terms $T_\Sigma$ is the free algebra built from names, variables, and function symbols in $\Sigma$ 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 $E \mapsto M = N$ and $E \mapsto M \not\equiv N$ for an equality and an inequality, respectively, modulo $E$.

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


3.1. An Abstraction Theory of Secret-sharing

Our abstraction of secret-sharing $\Sigma_{SS}$ is explained in the following. Secret-sharing process with threshold $(l, i, t)$ is represented as a term $SSP_{l,i,t}(\tau)$, name $\tau$ is used to identify specified secret-sharing process, we abuse notation by writing $\tau_{l,i}$ which represents $SSP_{l,i}(\tau)$; The secret key for secret share is represented as a term $SSK_{i,j,k}(\tilde{M}, m, \tau, \tilde{F})$, where $\tilde{M}$, called dealer parameters, denote sequence $M_1 \ldots M_l$ of terms; while $m$, called the proof’s identity $Id$, can be used to identify different secret key in same secret-sharing process and we have $m \leq 1$. Similarly, the corresponding verification key for secret share is represented as a term of form $SVK_{i,j,k}(\tilde{M}, m, \tau, \tilde{F})$ and we have $m \leq 1$. Further, the secret share is represented as a term of form $SSS_{i,j,k}(\tilde{M}, m, \tau, \tilde{F})$, where $\tilde{N}$, called player parameters, denote sequences $N_1 \ldots N_j$ of terms. We also include in $\Sigma_{SS}$ functions $SVer_{i,j,k}$ and $SCombin_{i,j,k,r}$ which represent verification and combination of secret shares respectively. In the definitions above, $\tilde{F}$ denote sequence $F_1 \ldots F_k$ of $(i,j)$-functions, see below.

The $(i,j)$-function $F$ constitutes a constant without names and variables, which is built upon distinguished nullary functions $\alpha$ and $\beta$ with $i \in N$.

Definition 3.1. We call a term an $(i,j)$-function if the term contains neither names nor variables, and if for every $\alpha_n$ and $\beta_q$, occurring therein, we have $m \in [1,i]$ and $n \in [1,j]$.

The values $\alpha$ and $\beta$ in $F$ constitute placeholders for the terms $M_i$ and $N_j$. In our abstraction model, the relationship between secret and dealer parameters, player parameters can be defined through $(i,j)$-functions.

3.2. A Finite Specification of Secret-sharing

In this section, we specify a finite equivalent theory $F^{TR,k}_{SS}$ in terms of a convergent rewriting system. This theory turns out to be suitable for mechanized security protocol analysis. The central idea of our finite equivalent theory is to focus on the secret-sharing proofs used within the process specification and to abstract each from the additional ones that are possibly generated by the environment. This makes finite the specification of the equational theory.

First, we track each secret share generated, verified 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 $k (i,j)$-functions of a 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 $term\{\}$ denote the set of terms in $P$. We can now formally define the notion of $(TR, h)$-validity of terms and processes.

Definition 3.2. A term $Z$ is $(TR, h)$-valid if and only if the following conditions hold:

1. For every $SSK_{i,j,k}(\tilde{M}, M, N, \tilde{F})$, $SFK_{i,j,k}(\tilde{M}, M, N, \tilde{F})$, $SSS_{i,j,k}(\tilde{M}, M, \tilde{F})$, $SVer_{i,j,k}(M, N, \tilde{F})$, 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 $E \mapsto \tilde{F} = \tilde{F}'$, we have $\tilde{F} = \tilde{F}'$.

2. For every $l \in N$, $\alpha$ and $\beta$ occur in $Z$ only inside of $(i,j)$-function of $Z$.

3. For every $SSP_{l,i}(M) \in terms(Z)$, we have $l \in [1,h]$.

A process $P$ is $(TR, h)$-valid if and only if $M$ is $(TR, h)$-valid for every $M \in terms(P)$.

We check that each secret share generation, verification and combination is tracked in $TR$ (condition1). We also check that for all secret-sharing proofs used in the process specification, the arity of dealer parameters, player parameters and $(i, j)$-functions is less or equal than $h$, respectively (condition3).

We now define the static compilation of term and process.

Definition 3.3. the $(TR, h)$-static compilation is the partial function $s : T_{eq} \otimes T_{SS}^{eq}$ recursively defined as follows:

150
SSK\textsubscript{\textit{i},j,k}(\vec{M},M,N,F)\sigma = SSK\textsubscript{\textit{i},j,k}(\vec{M},M,N); \hspace{1cm} (1)

SVK\textsubscript{\textit{i},j,k}(\vec{M},M,N,F)\sigma = SVK\textsubscript{\textit{i},j,k}(\vec{M},M,N); \hspace{1cm} (2)

SS\textsubscript{\textit{i},j,k}(\vec{M},M,F)\sigma = SS\textsubscript{\textit{i},j,k}(\vec{M},M); \hspace{1cm} (3)

SVer\textsubscript{\textit{i},j,k}(M,N,F)\sigma = SVer\textsubscript{\textit{i},j,k}(Mo,Na); \hspace{1cm} (4)

$S\text{Combin}_{i,j,k,r}(\vec{M},F)\sigma = S\text{Combin}^p_{i,j,k,r}(\vec{M},\sigma)$. \hspace{1cm} (5)

Since $\vec{F}$ are uniquely determined by $SSK_{\textit{i},j,k}^p$, $SVK_{\textit{i},j,k}^p$, $SS_{\textit{i},j,k}$ and $S\text{Combin}_{\textit{i},j,k,r}^p$, it can be omitted from the protocol specification.

Verification of secret shares with respect to verification key for secret shares is modeled by the following equational rule:

$S\text{Ver}_{\textit{i},j,k}^p(SVK_{\textit{i},j,k}^p(M,m,\tau,l),SS_{\textit{i},j,k}(\vec{N},SSK_{\textit{i},j,k}^p(\vec{M},m,\tau,l))) = true$. \hspace{1cm} (6)

For combination of different secret shares with same secret in the same $(l,t)$-threshold secret-sharing process, it suffice to check whether the arity of secret shares is more that $t$. Thus, we include in $E_{SS}^{TR,h}$ the functions $P\text{Combin}_{\textit{i},j,k,r}$ and $S\text{Ver}_{\textit{i},j,k,r}$.

$S\text{Ver}_{\textit{i},j,k,r}$ is used to determine if secret can be computable from $r$ different secret shares in a secret-sharing scheme with threshold $(l,t)$ and can be modeled as follow.

$S\text{Ver}_{\textit{i},j,k}^p(SS_{\textit{i},j,k}^p(\vec{N},SSK_{\textit{i},j,k}^p(\vec{M},i_1,\tau,l_1)),...,SS_{\textit{i},j,k}^p(\vec{N},SSK_{\textit{i},j,k}^p(\vec{M},i_t,\tau,l_t))) = eq(t,r)\lor \hspace{.5cm} (7)$

$S\text{Ver}_{\textit{i},j,k,t}^p(SS_{\textit{i},j,k}^p(\vec{N},SSK_{\textit{i},j,k}^p(\vec{M},i_1,\tau,l_1)),...,SS_{\textit{i},j,k}^p(\vec{N},SSK_{\textit{i},j,k}^p(\vec{M},i_{t-1},\tau,l_{t-1}))) for r>1$. \hspace{1cm} (7)

Thus, combination of $r$ different secret shares with same secret in the same $(l,t)$-threshold secret-sharing process is modeled by the following equational rules:

$S\text{Combin}_{\textit{i},j,k,r}^p(\vec{M}) = P\text{Combin}_{\textit{i},j,k,r}^p(\vec{M},S\text{Ver}_{\textit{i},j,k,r}^p(\vec{M})). \hspace{1cm} (9)$

$P\text{Combin}_{\textit{i},j,k,r}^p(SS_{\textit{i},j,k}^p(\vec{N},SSK_{\textit{i},j,k}^p(\vec{M},i_1,\tau,l_1)),...,SS_{\textit{i},j,k}^p(\vec{N},SSK_{\textit{i},j,k}^p(\vec{M},i_t,\tau,l_t))),true)$

$= F / \{ M / \alpha \} \{ N / \beta \}. \hspace{1cm} (10)$

The $P\text{Combin}_{\textit{i},j,k,r}$ and $S\text{Ver}_{\textit{i},j,k,r}$ functions are private, hence they cannot be used by the adversary.


We then analyze the security properties of $(l,t)$-threshold certificate protocol [4] with $E_{SS}^{TR,h}$.

The goal of threshold certificate protocol is to enable secret-sharing schemes to resist player’s cheating. The threshold certificate protocol is composed of three subprotocols: the secret distributing protocol, the secret reconstruction protocol and the secret recovering protocol. The secret distributing protocol allow players and off-line TTP to get secret share and $(l,t)$-threshold agreement certificate from dealer. The secret reconstruction protocol enable more than $t$ players to reconstruct the secret. The secret recovering protocol enable more than $t$ players and off-line TTP to recover the secret.

We assume dealer has a key-pair called endorsement key (EK) for each secret-sharing scheme as well as a publicly known identity $bsn_{d}$. 

---

Zhong-hai Zhou, Jian Yuan, Hui Zhao
4.1. Secret distributing protocol

Suppose that the dealer wants to share the secret sign(h(N₀), SK) among players whose public IDs are ID₁, ..., IDₙ according to the (l, t)-threshold secret-sharing policy, where IDᵢ ⊆ ID if i = 1, ..., n. For convenience, player IDᵢ denotes a player whose ID is IDᵢ. Let the special IDs of the off-line TTP be ID₁ where IDᵢ ⊆ ID, i = 1, ..., n, and i = 1, 2, ..., l − 1.

(1) The dealer uses an (2l − t + 1, l + 1)-threshold secret-sharing scheme to partition the secret sign(h(N₀), SK) into 2l − t + 1 shares (ID₁, tpss₁), (ID₂, tpss₂), ..., (IDₙ, tpssₙ), and (IDᵢ, tpssᵢ), where IDᵢ ⊆ ID, i = 1, ..., n, and i = 1, 2, ..., l.

(2) The dealer generates the signatures sign(tpss₁, SK), sign(tpss₂, SK), ..., sign(tpssₙ, SK).

(3) The dealer uses a (l, t)-threshold secret-sharing scheme to partition secret sign(h(Nᵢ), SK) into t shares (ID₁, ss₁), (ID₂, ss₂), ..., (IDₙ, ssₙ).

(4) The dealer sends (IDᵢ, tpssᵢ), (IDⱼ, tpssⱼ), ..., (IDᵢ-ᵢ, tpssᵢ-ᵢ) to the offline TTP over a secure channel.

(5) The dealer sends (tpssᵢ, sign(tpssᵢ, SK)) to player IDᵢ over a secure channel, for each i = 1, 2, ..., n.

It is noted that the part (tpssᵢ, sign(tpssᵢ, SK)) is called the (l, t)-threshold agreement certificate of the player IDᵢ.

In our calculus, we can model the dealer in the secret distributing protocol as follows.

define ssproof = sign(beta₁, alpha₁).
dea₁ = \Pi^{ID₁, ID₉} \cdot 
let tpssk = SSK₂₁,₁(SK, pk(SK), i, SSP₂₁−l−₁, l+₁(pi), ssproof) in
let tpssvk = SVK₂₁,₁(SK, pk(SK), i, SSP₂₁−l−₁, l+₁(pi), ssproof) in
let tpss = SS₂₁,₁ (h(N), tpssk, ssproof) in
let ssCERT = sign(tpss, SK) in
let ssK = SSK₂₁,₁ (SK, pk(SK), i, SSP (pi), ssproof) in
let ss = SS₂₁,₁ (h(N), ssK, ssproof) in
\text{oeb} (ID₁, tpss, ssCERT, ss)|\text{oeb}(= ID₁).oeb(ID₀, tpssvk).
dea₂ = \Pi^{ID₀, ID₉} \cdot 
let tpss = SSK₂₁,₁ (SK, pk(SK), l + j, SSP₂₁−l−₁, l+₁(pi), ssproof) in
let tpss = SS₂₁,₁ (h(N), tpssk, ssproof) in
\text{oeb} (ID₉, tpss).
dea = vpi. vSK. vN. !(let bsn . pk(SK)) | dea₁ | dea₂).

Here the define statement defines an abbreviation ssproof for the (i, j)-function we use in all secret-sharing proofs.

4.2. Secret reconstruction protocol

After successfully executing the secret distributing protocol, each player get secret share and a (l, t)-threshold agreement certificate from dealer. Players ID₁, ID₂, ..., IDₙ ∈ ID₉ will reconstruct the secret where ID₀ ⊆ ID₉, they may perform the following procedure:

(1) Each player IDᵢ submits her/his (2l − t + 1, l + 1)-threshold share (tpssᵢ, sign(tpssᵢ, SK)), which we call the player IDᵢ 's (l, t)-threshold agreement certificate ssCERTᵢ.

(2) Each player verifies the (l, t)-threshold agreement certificate (tpssᵢ, ssCERTᵢ) by the equation check(ssCERTᵢ, pk(SK)) = true or not, for each j = 1, 2, ..., n. k. If the number of the certificates which pass the verification is less than t, then they stop the procedure according to the (l, t)-threshold access structure. Otherwise, they perform the following steps.
Mechanized Analysis of Multi-Secret Sharing Based on Lagrange Interpolating Polynomial in the Applied Pi-calculus
Zhong-hai Zhou, Jian Yuan, Hui Zhao

(3) Each player submits her/his secret share \((ID_j, ss_j)\) to reconstruct the secret \(sign(h(N_\Omega), SK)\).

In our calculus, we can model the player in the secret reconstruction protocol as follow:

\[
\text{recon}_1^i = \text{c}(ttppss, ttpcert). cp^{k-2}(= ID_1). c(ID, ss).
\]

\[
\text{recon}_2^i = c(ttppss, ttpcert). \text{if check(ttpcert, PK) = true then}
\]

\[
\text{authenticate}^i (ttppss). cp^{k-2} (ttppss, ttpcert). \text{cp}^{k-3} (ID_j, ttppss, ttpcert).
\]

\[
c (= ID, ss). cp^{k-2} (ss).
\]

\[
\text{recon}_3^i = \text{!init}^i (ID_i). (\text{recon}_1^i | \text{recon}_2^i).
\]

\[
\text{recon}_4^i = \text{cp}^{i-1}(ss_1). \text{cp}^{i-1}(ss_2). \ldots \text{cp}^{i-1}(ss_l).
\]

\[
\text{let s = 1h}_i(S\text{Combine}_{1,1,1}(ss_1, ss_2, ..., ss_l, ssproof)) \text{in}
\]

\[
\text{obtain}^i (s).
\]

\[
\text{recon}_5^i = \text{!cp}^{i-1}(ID_\Omega, ttppss, ttpcert). \text{oeb} (= ID_\Omega, ttppss) | \text{oeb} (= ID_\Omega, ID_\Omega, ttppss) | \text{oeb}(=ID_\Omega, ss, obtain^i (s)).
\]

\[
\text{player} = \text{ct} (= \text{bsn}_\Omega, PK). \text{cp}^{i-1}(ss). \text{cp}^{i-1} (ss).
\]

\[
\text{tpdistribute}^i (ttppss, ttpcert). (\text{recon}_1^i | \text{recon}_2^i | \text{recon}_3^i).
\]

To simplify our model, here the restricted channel \(cp^{k-2}\) is used to decide if at least \(t\) different \((l, t)\)-threshold agreement certificates pass the verification of \(ID_\Omega\).

4.3. Secret recovering protocol

Suppose that some of the players don’t submit their \((l, t)\)-threshold secret shares. The remaining players may send \(\{(ID_j, ttppss_j)\}_j^{k-1}\) to the offline TTP and request her/him to reconstruct the secret. In the following, we present the procedure the off-line TTP performs in response to the request:

(1) The off-line TTP verifies the received \((2l-t+1, l+1)\)-threshold secret shares. If \(k < t\), he rejects the request and stops the procedure. Otherwise, she/he performs the following steps.

(2) The off-line TTP uses the \((2l-t+1, l+1)\)-threshold secret shares \(\{(ID_j, ttppss_j)\}_j^{k-1} \cup \{(ID_j, ttppss_j)\}_j^{k-1}\) to reconstruct the secret \(sign(h(N), SK)\).

(3) The off-line TTP sends the secret \(sign(h(N), SK)\) to the player \(ID_j\), for each \(j = 1, 2, ..., k\).

In our calculus, we can model the TTP in the secret recovering protocol as follow:

\[
\text{recov}_1^i = \text{P}_{\text{ob}(ID_\Omega, ttppss)}. \text{oeb}(=ID_\Omega, ttppsvk).
\]

\[
\text{if SVer}_{\Omega, l}(ttppsvk, ttppss, ssproof) = \text{true then}
\]

\[
\text{cp}^{i-1} (ID_\Omega, ttppss). \text{cp}^{i-1} (= ID_\Omega, ss). \text{distribute}^{i-1} (ID_\Omega, ss). \text{oeb}(=ID_\Omega, ss) | \text{cp}^{i-1} (= ID_\Omega, ttppss). \text{cp}^{i-1} (= ID_\Omega, ttppss). \ldots \text{cp}^{i-1} (= ID_\Omega, ttppss).
\]

\[
\text{cp}^{i-2} (ttppss_1). \text{cp}^{i-2} (ttppss_2). \ldots \text{cp}^{i-2} (ttppss_{l+1}).
\]

\[
\text{let s = 1h}_i(S\text{Combine}_{1,1,1,1}(ttppss_1, ttppss_2, ..., ttppss_{l+1}, ssproof)) \text{in}
\]

\[
\text{!cp}^{i-2} (ID_\Omega, ss).
\]

\[
\text{tp} = \text{recov}_1^i | \text{recov}_2^i.
\]
4.4. Authenticity of the protocol

We will now discuss the main security property of the threshold certificate protocol, the authenticity property, and how to model it in our calculus. Firstly, we can define this protocol as follow:

\[ \text{System} = \text{dealer} \mid \text{player} \mid \text{ttp}. \]

The security goal of threshold certificate protocol is to enable secret-sharing schemes to resist player’s cheating. Thus, any player who submits a false \((l, t)\)-threshold agreement certificate will be detected. And, no information about the secret shares can be computed from the cheating.

\[ \text{GS}(\text{ID}_G, \text{ID}_j, t) \] is defined to include all the sets which contain \(t\) elements different from \(\text{ID}_j\) in \(\text{ID}_G\). Thus, the authenticity property of threshold certificate protocol is defined as the fulfillment of the following trace properties:

\[
\forall \text{ID}_i \in \text{ID}_G. \quad \text{obtain} (\text{obtain}_i, \text{sign}(h(N), SK) \rightarrow (\text{GS}(\text{ID}_G, \text{ID}_j, t + 1) \mid \text{authenticate} (\text{authenticate}_i, SSK_{2, 1, 1}(h(N), SSK_{2, 1, 1}(SK, pk(SK), k, ssp_{2s-l-t+1,l-t+1}(pi), ssp_{proof}, ssp_{proof})) \vee \text{distribute}(\text{ID}_i, \text{sign}(h(N), SK))).
\]

Which means that if a player obtains the secret \(\text{sign}(h(N), SK)\), then either there exists at least \(t-1\) other players who verify at least \(t-1\) players’ certificates in the same run or the secret \(\text{sign}(h(N), SK)\) is from TTP to resist player’s cheating.

Trace properties such as above can be verified with the mechanized prover ProVerif. In ProVerif script, the parallel compositions are replaced with replicated inputs, for example, \(\text{ID}_i \mid \text{ID}_j\) are replaced with ! \(\text{exp}_{\text{ID}_i}(\text{ID}_i, i)\) on the restricted channel \(\text{exp}_{\text{ID}_i}\). We also add the events \(\text{authenticate}(\text{authenticate}_i, \text{ttpss}), \text{obtain}(\text{obtain}_i, s), \) and \(\text{distribute}(\text{ID}_i, s)\) just before \(\text{authenticate}(\text{ttpp}, \text{obtain}_i, s), \) and after \(\text{distribute}_2(\text{ID}_i, s)\), respectively.

ProVerif shows that the threshold certificate protocol satisfies the authenticity property.

5. Conclusion

In this paper, we apply an abstraction of multi-secret sharing schemes based on lagrange interpolating polynomial to the threshold certificate protocol, yielding its first mechanized security proof within the applied pi-calculus. On the basis of this, we realize the security proof with the mechanized prover ProVerif and our results shows that the threshold certificate protocol satisfies the authenticity property.

References


