2005

EVALUATION TECHNIQUE IN THE SPI-CALCULUS FOR CRYPTOGRAPHIC PROTOCOLS

Philadelphia University, *Philadelphia University*

EVALUATION TECHNIQUE IN THE SPI-CALCULUS FOR CRYPTOGRAPHIC PROTOCOLS

Hasan M. Al-Refai
halrefai@philadelphia.edu.jo
Hasan_alrefai@yahoo.com
Mobile Phone #: 00962-777357755

Abstract

Bisimulation as a technique could be well invested for proving authenticity and secrecy properties of cryptographic protocols to gain the legality of protocol optimization. In this paper, we will do some changes in the spi-calculus after the original work of M.Abadi and A.Gordon. Then we will introduce evade bisimulation following Abadi and Gordon's framed bisimulation proposal, in which a convenient proof technique is presented. It will impose minimality requirements on the environment and detect the limit beyond which the bisimilarity is kept valid and furthermore it will avoid quantification over contexts. Also, it will give a solution for input transitions for the case of finite processes. Based on the revised spi-calculus would be used to prove that evade bisimilarity, an equivalence relation, is decidable for main security properties: Authenticity and Secrecy.

Keywords: *Cryptographic protocols, testing equivalence, Bisimulation , authenticity and secrecy*

1. Introduction

The present era of modern developments is denoted by mobile communication. In this field a wide range of proposals have been adopted as a result of the increased utilization of internet applications for business transactions, E-commerce, exchanges of sensitive government information and enumerable military implementation tasks. A considerable development in telecommunication networks has contributed to the success of these applications. As all of these services take advantage of communication in distributed systems in which computers are no longer considered devices that operate in isolation. Conversely, they are now part of a global environment where local and remote resources can be shared.

The development of telecommunication networks has therefore encouraged the development of services that take advantage from the communication among different distributed environments .It is clear that most of these applications would not be applied without verifying the security decision.

The *spi-calculus* introduced by Abadi and Gordon [1] as an extension of the π-calculus [20, 19] with cryptographic primitives has been used for cryptographic protocols modeling. By the implementation of these tools, cryptographic protocols design is drawn for possible study and investigations to prove the main security properties of the authentication and secrecy. The major attempts focused on in this thesis are testing equivalence and bisimulation techniques in spi-calculus.

In general, through the bisimulation technique, two different systems could be decided as equivalent when a correspondence between their steps is established or not equivalent when no correspondence is approved. The correspondence involves system behavior steps and also the steps taken by its environment as reactions. In the bisimulation modeling, it is not necessary to model an environment explicitly neither to analyze its possible internal computations but all what is needed, is to forward the modeling aspect towards the overall interaction between the system from one side and the environment from the reaction side.

M.Abadi and A.Gordon defined the *Framed bisimilarity* method [2] as a trend to prove process equivalence based process-environment interactions. Their work represented a restricted attempt that necessitated the existence levels of quantification over infinite domains. Our work follows Abadi and Gordon's framed bisimulation proposal. It introduces *evade* bisimulation in which a convenient proof technique is presented. This technique is driven; it will detect the limit beyond which the bisimilarity is kept valid and avoid quantification over contexts. In this paper, we will also introduce a new variant of the spi-calculus suggested by [7]. Based on the revised spi-$_{H}$-calculus would be used to prove that evade bisimilarity, an equivalence relation, is decidable for main security properties: *Authenticity and Secrecy*.

2. THE Spi-$_{H}$-CALCULUS

This section gives the syntax and informal semantics of the spi-$_{H}$-calculus used by [7] with some changes, after the original work of Abadi and Gordon [1]. The assumptions of this work are built on shared-key cryptography.

2.1 Syntax

Names and operators construct the syntax of the spi-$_H$-calculus, primitively, and other structures are built on them. Protocol model, is a structure of representative spi-$_H$-calculus syntax. In this structure messages, expressions, logical formulae and processes define the attributes needed to express all the objects and the activities driven in establishing protocols. Table 1 summarizes the calculus.

Names $\mathcal{N}$ range over communication channels, data (variables or clear texts and a message) or keys. Along the declaration, names are used alternatively through the syntax regardless of representations. Any name would be tagged to the type by its appearance order.

Notation 1: We reserve the lower case letters *a, b, ch* to denote channels and *k, l* to denote keys and *m, n* to denote messages.

Expressions are those descriptions that are obtained by applying encryption, decryption, paring and projection operators to names and ciphertext. For example, the expression ζ is a cleartext, when it is encrypted using the value η as a key, the overall actions would be given by $^{Enc\ \eta}(\zeta)$, yielding that ζ as a cleartext is encrypted under the key η, conversely the decryption action $Dec_\eta\{\zeta\}$ stands to decrypt the ciphertext using the value of η as a key. We assume that a key should be a name and the encryption action uses shared key in modes simple and nested modes.

Logical formulae generalize the usual equality operator of the π-calculus by conjunction and negation. Moreover we introduce two new predicates $[\zeta : \mathcal{N}]$ and $[\zeta : \mathcal{M}]$. The predicate $[\zeta : \mathcal{N}]$ which tests for the format of the argument ζ, whether it evaluates to a name or not, and the predicate $[\zeta : \mathcal{M}]$ which test for the argument ζ, whether it evaluates to a compound ciphertext or not, and with "*Let"* construct that binds the value of some expression ζ to a name *z.*

A finite set of terms, $T = \{t_1, t_2, \ldots, t_n\}$. The difference from Abadi and Gordon [1] that in this one assume the set *T* to be defined, so that we can associate each state to a term *t*. In order to define the notion of *state* we have to introduce the definition of finite multisets.

Definition 2: A fi*nite multiset* ϖ over a set *L* is a map $\varpi : L \rightarrow M$ such that $\varpi^{-1}(M_1)$ is finite. We define the following operations on finite multisets:

a) The *difference* of the multisets ϖ and ϖ' is the multiset $\varpi \setminus \varpi'$ where$(\varpi \setminus \varpi')(l) = max(0, \varpi(l) - \varpi'(l))$;

b) The *union* of two multisets ϖ and ϖ' is the multiset $\varpi \bigcup \varpi'$ where $(\varpi \bigcup \varpi')(l) = \varpi(l) + \varpi'(l)$;

c) We say that $l \in \varpi$ *iff* $\varpi(l) > 0$.

We define formally our notion of *state* as follows:

Definition 3: We define a *state*, $\sigma = \{\sigma_t\}_{t \in T}$, to be a family of multisets indexed by the terms, where each σ_t represents the local state of the term *t*. We denote the set of all states by $\sum$.

A first approach to the definition of state would be a family of sets. If we consider each σ_t as a set, we were restricting the possibility of a principal to have many copies of the same term. We want to deal with the possibility of existing several inputs of the terms and verify that our system has the desired properties for input transitions.

Processes are the different sequences of activities conducted along a protocol function. There are diverse forms of processes that have a distinct function for each. A process could be built using a set of operators that include standard *π-calculus* [20] and two new ones; Boolean guards and encryption/decryption. However, process forms are used to explain the following:

- **0**, is a null process that dose nothing.
- An input process; $\eta(x).P$ represents input of a generic message *x* along η : the only useful case is when η is a name , otherwise the whole process is stuck .
- An output process $\bar{\eta}\zeta.P$ represents out of ζ along the channel η .The only useful case is when η is a name and ζ is a message, otherwise the whole process is stuck.
- Non-deterministic choice $P + Q$: can behave either as *P* or *Q*; the choice might either be triggereazd by the environment, or by internal computations of *P* or *Q*.
- Parallel composition $P \mid Q$; is the parallel execution of *P* and *Q*.
- Restriction $(\nu a)P$: creates a new name *a* which is only known to *P*.
- Replication $!P$ behaves like many unbounded copies of *P* running in parallel, i.e. $P \mid P \mid P \mid$ **….**

- Boolean Guard ϕP behaves like *P* if the formula is logically true, otherwise is stuck.
- Encryption/Decryption *let z = ζ in P*: attempts evaluation of *ζ*: if the evaluation succeeds, the result bound to *z* within *P*, otherwise the whole process is stuck.

Usual calculus abbreviate *(ν a)(ν b) P* into *(ν a, b) P*, and $\overline{M}(N).0$ into $\overline{M}(N)$. In the spi-$_{H}$-calculus, processes are identified up to renaming of bound names. Bound names are the entities that are enclosed within a process definition *P*, and not those that have explicitly been tagged with any other outside the process. So the closed process will then be defined as the process that has no free variables; *Proc is used to define a set of closed processes.*

Let *fn(P)* denote the set of free names in *P* and *fv(P)* is the set of free variables in *P* , and alpha-equivalence arises as expected , *n(P) is fn(P)* $\cup$ *bn(P)*. In this context, similar notations are used for formulae, expressions and messages.

If two processes can be made equal by conflict-free renaming of bound names then they are alpha-equivalent. Substitutions σ are mappings $\{\zeta/x\}$ from names *x* to messages ζ, following the usual assumption that name-capture is avoided through implicit alpha conversion. Substitutions are applied to processes, expressions and guards very simply as for example, $P\{\zeta/x\}$ replace all free occurrence of *x* in *P* by ζ , possibly renaming bound names in *P* avoiding name capture.

Definition 4: A substitution σ is a finite partial map from the set of names *N* to the set of messages *M*; *{M/x}*.

The definition shows the effect of applying a substitution σ to a process *P*. This is essentially to replace each free occurrence of each name (i.e.; *x*) in *P* by σ(*x*) = *M,* for some *x* and *M* .The mapping must, however, be done in such a way that unintended capture of names by binders is avoided Substitutions are applied to processes, expressions and guards in straightly, i.e. $P\{M/x\}$ replace all free occurrence of x in *P* by *M*, possibly renaming bound names in *P* avoiding name capture.

Domain and its co-domain of σ are denoted as *dom*(σ) and *range*(σ) respectively .Let *n(σ)*= $dom(\sigma) \cup \left(\bigcup_{M \in range(\sigma)} n(M) \right)$ When a tuple of distinct names $\vec{x} = (x_1, x_2, ..., x_n)$ and a tuple of messages $\vec{M} = (M_1, M_2, ..., M_n)$ are given , the substitutions mapping of each x_i to M_i will be convenient . Usually a tuple is a set of its component. $\sigma[\vec{M}/\vec{x}]$ is the substitution σ' which represents union of σ and $[\vec{M}/\vec{x}]$. Such case is referred to as σ' extends σ.

In the current syntax of spi-calculus, the assumptions on the cryptographic systems are recalled thus:

1– For perfect encryption the key *k* used for encrypting a message *M* in the form $Enc_k(M)$ should be used for decrypting that message. Encrypting M under k can only produce the ciphertext. Thus, a hacker cannot decrypt the message *M* without knowing the encryption key *k*.

2- The assumptions necessitate that there is enough redundancy in the structure of messages to ensure whether the decryption of a message on a given key has actually succeeded or not and additionally, to verify the role (either a key or a compound ciphertext).

3– Strictly, a new key is created by using fresh names from a primitive set of names.

TABLE 1 Syntax of the Calculus

$a,b,c...,k,l,m,n...,x,y,z$		names N
$\zeta,\eta ::= a$	$\mid Enc_\eta(\zeta) \mid Dec_\eta(\zeta)$	expression s $\Im$
	$\mid \langle\zeta_1,\zeta_2\rangle \mid \pi_1(\zeta) \mid \pi_2(\zeta)$	
$M,N ::= a$	$\mid \langle M_1,M_2\rangle \mid Enc_k\{M\}$	messages M
$\phi,\psi ::= tt$	$\mid \phi \wedge \phi \mid \neg\phi$	guards Φ
	$\mid [\zeta = \eta]$	(equality)
	$\mid [\zeta : N]$	(is a name)
	$\mid [\zeta : M]$	(is a decomposab ility)
	$\mid$ let $z=\zeta$ in ϕ	(decryption)
$P,Q,R ::=$		processes P
	$\mid 0$	(null)
	$\mid \eta(x).P$	(input prefix)
	$\mid \overline{\eta}\langle\zeta\rangle.P$	(output prefix)
	$\mid P \mid Q$	(parallel compositio n)
	$\mid P+Q$	(choice)
	$\mid !P$	(replicatio n)
	$\mid (\nu n)P$	(restrictio n)
	$\mid \phi P$	(boolean guard)
	$\mid$ let $z=\zeta$ in P	(encryption / decryption)

It is assumed that $Dec_{(.)}$ does not occur in $[\zeta : N]$, $[\zeta = \eta]$, $\eta(x)$. and $\overline{\eta}\langle\zeta\rangle$.Operators $a(x).$, $(\nu a).$ and let $z=\zeta$ in . are binders , with the obvious scope , for names x, a and z, respective ly . In let $z=\zeta$ in . , it is assumed that $z \notin n(\zeta)$

2.2 Operational Semantics

Two main evaluation modes are used in the operational semantics evaluation function. The first is utilized for expressions while the second is used for Boolean Guards. These two evaluations are denoted as follows:

- For an expression $\overline{\overline{\cdot}} : \varepsilon \to M \cup \{\perp\}$ *where* $\perp$ *is a distinct symbol* ($\perp \notin M$) .
- For an evaluation of a Guard $\underline{\underline{\cdot}} : \Phi \to \{tt, ff\}$, *is defined by induction on* Φ.

The evaluation function is defined recursively according to table 2. In this table, it is obvious that expression evaluations rely on the implementation of *let* and *guard* only. Hence, the decryption is bounded along this evaluation scheme.

As table 3 shows, the operational semantics of the spi-$_H$-calculus used in our work. *Let* and guard items are used as primitive rules driven to decrypt messages. A process ΦP behaves like P provided that Φ evaluates to true, otherwise, ΦP is stuck. A process *let* $z = \zeta$ *in* P *behaves like* $P\left\{\underline{\underline{\zeta}}/z\right\}$ provided that the evaluation of ζ succeeds; otherwise, *let* $z = \zeta$ *in* P *behaves like* $P\left\{\underline{\underline{\zeta}}/z\right\}$ is stuck. Rule (E-OUT) details the case when the environment receives a message M and updates its knowledge accordingly, and for the sake of a transition to occur, all channels are supposed to be well announced to the environment.

Rule (E-INP) details the case, when the environment sends a message M to the process. Message m is not arbitrary and the expression ζ describes how this message is built out of the environment and of the names $\vec{b}$ to define M. Creation of new names $\vec{b}$ is recorded by $\left[\vec{b}/\vec{x}\right]$, and in this case a must belong to the knowledge of the environment to explain its announcement.

3. Evade Bisimulation

Based on the earlier works of framed bisimulation[7, 2], we concluded that their definition suffers from quantification over all possible contexts, and furthermore it does not overcome the general problem for the input transitions. Hence, the present work will solves these conducts to give a proper outcome overcoming universal quantification.

TABLE 2 Evaluation in the spi -Calculus

$\overline{\overline{a}}$	$=$	a
$\overline{\overline{Enc_{\zeta}(\eta)}}$	$=$	$\begin{cases} Enc_k(M) & \text{if } \overline{\overline{\eta}} = M \in \mathcal{M} \text{ and } \overline{\overline{\zeta}} = k \in \mathcal{N} \\ \perp & \text{otherwise .} \end{cases}$
$\overline{\overline{Dec_{\zeta}(\eta)}}$	$=$	$\begin{cases} M & \text{if } \overline{\overline{\eta}} = Enc_k(M) \in \mathcal{M} \text{ and } \overline{\overline{\zeta}} = k \in \mathcal{N} \\ \perp & \text{otherwise} \end{cases}$
$\overline{\overline{\langle \zeta_1, \zeta_2 \rangle}}$	$=$	$\begin{cases} \langle M_1, M_2 \rangle & \text{if } \overline{\overline{\zeta_1}} = M_1 \text{ and } \overline{\overline{\zeta_2}} = M_2, \\ & \text{for some } M_1 \text{ and } M_2 \\ \perp & \text{otherwise} \end{cases}$
for $i=1,2$, $\overline{\overline{\pi_i(\zeta)}}$	$=$	$\begin{cases} M_i & \text{if } \overline{\overline{\zeta}} = \langle M_1, M_2 \rangle, \text{for some } M_1 \text{ and } M_2 \\ \perp & \text{otherwise} \end{cases}$
$\overline{\overline{tt}}$	$=$	tt
$\overline{\overline{\phi \wedge \psi}}$	$=$	$\overline{\overline{\phi}} \wedge \overline{\overline{\psi}}$
$\overline{\overline{\neg \psi}}$	$=$	$\neg \overline{\overline{\psi}}$
$\overline{\overline{let\ z = \zeta\ in\ \phi}}$	$=$	$\begin{cases} \overline{\overline{\phi\{M/x\}}} & \text{if } \overline{\overline{\zeta}} = M \in \mathcal{M} \\ ff & \text{otherwise} \end{cases}$
$\overline{\overline{[\zeta : \mathcal{N}]}}$	$=$	$\begin{cases} tt & \text{if } \zeta \in \mathcal{N} \\ ff & \text{otherwise} \end{cases}$
$\overline{\overline{[\zeta : \mathcal{M}]}}$	$=$	$\begin{cases} tt & \text{if } \overline{\overline{\zeta}} \neq \perp \\ ff & \text{otherwise} \end{cases}$
$\overline{\overline{[\zeta = \eta]}}$	$=$	$\begin{cases} tt & \text{if } \zeta = \eta \in \mathcal{M} \\ ff & \text{otherwise} \end{cases}$

TABLE .3 The Operational Semantics of the spi-Calculus

$$(OUT)\ \frac{\overline{\overline{[\eta : \mathcal{N}]}} = a \quad \overline{\overline{[\zeta : \mathcal{M}]}} = M}{\overline{\eta}\langle\zeta\rangle.P \xrightarrow{\overline{a}M} P} \qquad (INP)\ \frac{\overline{\overline{[\eta : \mathcal{N}]}} = a}{\eta(x).P \xrightarrow{aM} P\{M/x\}}$$

$$(SUM)\ \frac{P \xrightarrow{\mu} P'}{P + Q \xrightarrow{\mu} P'}$$

$$(REP)\ \frac{P \mid !P \xrightarrow{\mu} P'}{!P \xrightarrow{\mu} P'} \qquad (ALPH)\ \frac{Q \xrightarrow{\mu} Q' \quad P \equiv_{\alpha} Q}{P \xrightarrow{\mu} Q'}$$

$$(PAR)\ \frac{P \xrightarrow{\mu} P'}{P \mid Q \xrightarrow{\mu} P' \mid Q} \text{ if } bn(\mu) \cap fn(Q) = \emptyset$$

$$(RES)\ \frac{P \xrightarrow{\mu} P'}{(\nu c)P \xrightarrow{\mu} (\nu c)P'} \text{ if } c \notin n(\mu)$$

$$(OPEN)\ \frac{P \xrightarrow{(\upsilon \vec{b})\overline{a}M} P'}{(\nu c)P \xrightarrow{(\nu c \vec{b})\overline{a}M} P'} \text{ if } n(M) \ni c \notin \{a, \vec{b}\}$$

$$(COM)\ \frac{P \xrightarrow{aM} P' \quad Q \xrightarrow{(\nu \vec{b})\overline{a}M} Q'}{P \mid Q \xrightarrow{T} (\nu \vec{b})(P'\left[M/x\right] \mid Q')} \text{ if } \{\vec{b}\} \cap fn(P) = \emptyset$$

$$(GUARD)\ \frac{P \xrightarrow{\mu} P'}{\phi P \xrightarrow{\mu} P'} \text{ if } \overline{\overline{\phi}} = tt \qquad (LET)\ \frac{P\{\overline{\overline{\zeta}}/z\} \xrightarrow{\mu} P'}{let\ z = \zeta\ in\ P \xrightarrow{\mu} P'} \text{ if } \overline{\overline{\zeta}} \neq \perp$$

Sensitive-environment bisimulation is like a game of interactions between a process steps and the current environment's knowledge about names and keys. The moves of the process are constrained by this knowledge. This interaction can be ruled to be a proof technique for such systems under such environments. We need to model the steps taken by the environment as a reaction to corresponding steps triggered by the process. This can be implemented by building a function to map the reactions of the environment to corresponding actions of the process.

In order to define evade bisimulation, we have to precisely define when two environments represented by a pair of frame-theory (*fr,th)* with a pair of substitutions (σ, ρ) -(considered to have the same domain and constructed in the structure of (*fr,th*) pair)- can be considered as equivalent . Informally,

two environments $(fr, th)_{(\sigma,\rho)}$ and $(fr', th')_{(\sigma',\rho')}$ are equivalent whenever they are logically indistinguishable under this pair of substitutions. In other words, in order to compare the knowledge of environments, we need the environment behavior to be built into the frame-theory pair implemented by profiting from a pair of equivalent substitutions. Here, the frame is defined as a set of pairs of names (not like the original definition of Abadi and Gordon) considered to be known to the environments.

Definition 5: *Let evade* $e = (fr, th)_{(\sigma,\rho)}$ *be a frame-theory pair with a pair of equivalent substitutions* (σ, ρ) *such that* $\sigma \approx \rho$ (see Definition 7). *A theory (th) is a finite subset of messages M X M indexed by using the projections* $\pi_i(th)$ *and* $\pi_i(th)$ *for i = 1, 2 .A frame (fr) is a finite subset of names N X N known to the environments. We denote by E the set of all evades.*

Defining a finite partial function that is used for mapping the set of names (considered as a domain) to the set of messages in the theory (considered as a proper co-domain), where the whole function is called substitution σ, i.e. $\{M/x\}$, for some $M \in \mathcal{M} \in (th)_\sigma$ and $x \in \mathcal{N}$ *(variables)*.

Definition 6: *Let the partial functions be as* $\sigma : x \to \mathcal{M}$, $\rho : x \to \mathcal{M}$, *where* $dom(\sigma) = dom(\rho)$ *then* :

1 – f(x) : $x \to M\ X\ M$ *and*

2 – $f(x)$: $\sigma(x) \leftrightarrow \rho(x)$, *when* $\sigma(x) = \zeta \in M$ *and* $\rho(x) = \eta \in M$, *then* $f(x) : \zeta \leftrightarrow \eta$.

To extend this definition to be defined in evade frame-theory pair with two equivalent substitutions, we have to determine the left and right position in *th* as a string $\{l, r\} \in I$, that is, a path through the nested pairs of $\mathcal{M}$.

Definition 7: *let e = (fr, th)* $_{(\sigma,\rho)}$ *be evade framed-theory pairs is consistent iff for all* ζ and η *are messages, such that* $\left\{ \left(\underline{\underline{[\zeta_{l_i} : M]}} \wedge \underline{\underline{[\eta_{r_i} : M]}} \right) \mid (\zeta_{l_i}, \eta_{r_i}) \in (th)_{(\sigma,\rho)},\ i = 1, 2 \right\}$ *and using the projections with indexes i = 1, 2 such that* $\pi_i^l \left(th_{(\sigma_i^l, \rho_i^r)} \right) = \zeta$ *and* $\pi_i^r \left(th_{(\sigma_i^l, \rho_i^r)} \right) = \eta$ *to denote the left and right position of the substitution pair corresponding to th. A frame is defined as a pair of names such that* $\left\{ \left(\underline{\underline{[a_{l_i} : N]}} \wedge \underline{\underline{[b_{r_l} : N]}} \right) \mid (a, b) \in fr,\ i = 1, 2 \right\}$, *and a pair of substitutions* (σ, ρ) *considered equivalent such that* $\sigma \approx \rho$, *when we have the following conditions*:

i - $\forall x_j$, σ *and* ρ , *where* $dom(\sigma) \ni x_j \in dom(\sigma) \Rightarrow dom(\sigma) = dom(\rho)$ *for* $j = \{1, 2, ..., n\}$ *and for* $\sigma : x_j \to \zeta$ *and* $\rho : x_j \to \eta$ *there is a partial function* $f(x_j) : x_j \to \zeta, \eta$, *such that :* $f(x_j) = (th)_{(\sigma^l, \rho^r)} =$

$$\left\{ \left(\sigma^l(x_j), \rho^r(x_j) \right) \middle| \sigma^l(x_j) = \zeta \in \pi_i^l(th)_{(\sigma^l, \rho^r)} \wedge \rho^r(x_j) = \eta \in \pi_i^r(th)_{(\sigma^l, \rho^r)},\ for\ i = 1, 2 \right\}$$

ii – *if* $(\zeta', \eta') \in (th)_{(\sigma^l, \rho^r)}$ *then* $\zeta = \zeta' \Leftrightarrow \eta = \eta'$

iii – if $\sigma(x) \in N \Leftrightarrow \rho(x) \in N$, *where* σ *and* ρ *have the same domain* $x_j = (x_1, ..., x_n)$

iv - if $\zeta = Enc_k(\zeta')$ *and* $\eta = Enc_l(\eta')$ *then* $fr \cap \{k, l\} = \emptyset$

Definition 8: The synthesis *S* (.) of evade is defined as:
$S(e)_{(\sigma,\rho)} = S((fr, \emptyset), e)$. We write $e \succ \zeta \leftrightarrow \eta$ *for* $(\zeta, \eta) \in S(e)$, $e \not\succ \zeta \leftrightarrow \eta$ otherwise.

The next definition formalizes the synthesis *S*(.) of evade in the concept of maximal decryption depth of environments *e,* written *dp*(*e*). Elements in *dp*(*e*) represent the basic knowledge derivable from *e*, i.e. the "building blocks" the environment can use to synthesize more complex messages.

Definition 9: *let (th)* $_{(\sigma^l, \rho^r)}$ *be a subset of M X M of messages,* Definition 7**.** *The maximal decryption depth of* $(th)_{(\sigma^l, \rho^r)}$, *written* $dp(th)_{(\sigma^l, \rho^r)}$ *can be derived by:*

$$(Max - Dec - Depth)\ \frac{(Enc_{\mu_1}(\zeta_1), Enc_{\mu_2}(\zeta_2)) \in dp(th) \qquad (\mu_1, \mu_2) \in dp(th)}{(\zeta_1, \zeta_2) \in dp(th)}.$$

Definition 10: *Let the knowledge of th, written as kn(th)* $\left(\sigma^l,\rho^r\right)$, *be the set of names in dp(th)* $\left(\sigma^l,\rho^r\right)$, *i,e.:* $kn(th)_{(\sigma^l,\rho^r)} := dp(th) \cap N$ *and* $\left\{ kn\left(\sigma^l(x_j)\right) = kn\left(\rho^r(x_j)\right) \mid \left(\sigma^l(x_j),\rho^r(x_j)\right) \in th\left(\sigma^l_j,\rho^r_j\right)\right\}$ *and the knowledge of the environment e, written* $kn(e)_{(\sigma,\rho)}$ *is defined as a set of names in kn(th)* $\left(\sigma^l,\rho^r\right)$ *and fr, i,e.:* $kn(e)_{(\sigma,\rho)} := \left(dp\ (th) \cup fr\right)_{(\sigma,\rho)} \cap N$ *. In other word we can define the knowledge of the as:*

$$kn\ (th)_{\left(\sigma^l,\rho^r\right)} := dp\ (th)_{\left(\sigma^l,\rho^r\right)} \setminus \left\{ \begin{array}{l} (Enc_{\mu 1}(\zeta_1),\ Enc_{\mu 2}(\zeta_2)) \mid \\ (\mu_1,\mu_2) \in dp(th)_{\left(\sigma^l,\rho^r\right)} \ \wedge\ (\zeta_1,\zeta_2) \in M \end{array} \right\}$$

and the knowledge of e is : $kn(e) := kn(th)_{\left(\sigma^l,\rho^r\right)} \cup fr.$

Pairs of messages are decrypted using pairs of keys (names) considered equivalent by the environment This being the maximal decryption depth of evade theory. The resulting notion of knowledge *kn*(.) in fact preserves a minimal extension of the environment.

Note: From now, we will use the term *non-decreasable function* instead of knowledge of the environment *kn*(*e*), which means, no knowledge is disclosed any more to the environment.

Definition 11: *From Definition 3, we can define a state,* $\left\{\sigma = \left\{\sigma_{\zeta_i^l}\right\}_{\zeta_i^l \in th} \text{ and } \rho = \left\{\rho_{\zeta_i^r}\right\}_{\zeta_i^r \in th} \mid i = 1,2\right\}$, *to be a family of multisets indexed by the terms* ζ_i^l *and* ζ_i^r, *where each* σ_t *represents the local state of the term t in* σ *and same for* ρ, *where* $(\zeta_i^l,\zeta_i^r) \in th_{\left(\sigma^l,\rho^r\right)}$.

Definition 12 : *Let* $dp\left(e^{\sigma}_{\pi_i^l(th)}\right) := dp(range\ (\sigma))$ *and* $kn\left(e^{\sigma}_{\pi_i^l(th)}\right) := kn(range(\sigma))$ *same case for* ρ,

$dp\left(e^{\rho}_{\pi_i^l(th)}\right) := dp\ (range\ (\rho)$ *and* $kn\left(e^{\rho}_{\pi_i^l(th)}\right) := kn\ (range(\rho)$ *for all* $i = 1,2.$

We introduce a function that decrypts a pair of messages (ζ_1,ζ_2) as far as possible, i.e., $(\zeta_1,\zeta_2) \in th$ assumed as a range of a pair of substitutions (σ,ρ), using the knowledge of *e* and the pair of substitutions (σ,ρ).

Definition 13: *Let e = (fr,th)* $\left(\sigma^l,\rho^r\right)$ *be evade frame-theory pairs,* $(\zeta_i,\eta_i) \in th$ *and given any equivalent pairs of substitutions* (σ,ρ), *such that* $\sigma_{\pi_i^l(th)} \sim \rho_{\pi_i^r(th)}$, *if* $\sigma_{\pi_i^l(th)} = \left[\zeta_j/x_j\right], \rho_{\pi_i^r(th)} = \left[\eta_j/x_j\right]$ *and* $i,j \in I$, *we denote by crux* $\left((\sigma,x_j),(\rho,x_j)\right)$ *what are left of* (ζ_j,η_j) *after decrypting as much as possible using the keys in* $kn\left(\sigma^l(x_j) \cup \rho^r(x_j)\right)$. *Formally, we define crux as:*

$$crux_{\left(\sigma_{\pi_i^{jl}(th)},\rho_{\pi_i^{jr}(th)}\right)}(\zeta_j,\eta_j) := \begin{cases} crux_{\left(\sigma_{\pi_i^{jl}(th)},\rho_{\pi_i^{jr}(th)}\right)}(\zeta'_j,\eta'_j) \ if\ \zeta_j = Enc_k(\zeta'_j),\ \eta_j = Enc_l(\eta'_j) \\ \qquad\qquad and\ \ fr \nvdash (k,l) \in kn\left(\left(\sigma_{\pi_i^{jl}(th)}\right) \cup \left(\rho_{\pi_i^{jr}(th)}\right)\right) \\ (\zeta_j,\eta_j) \qquad otherwise. \end{cases}$$

Definition 14: We decompose any pair of messages $(\zeta,\eta) \in th$ with a pair of equivalent substitutions (σ,ρ) such that $\sigma \sim \rho$ into:

$\underline{\underline{[(\zeta\sigma),(\eta\rho)]}} = \left(Enc_{k_n}\left(\ldots\left(Enc_{k_2}\left(Enc_{k_1}(crux\ (\zeta))\right)\right)\ldots\right), Enc_{l_n}\left(\ldots\left(Enc_{l_2}\left(Enc_{l_1}(crux\ (\eta))\right)\ldots\right)\right)\right)$ for $(\zeta,\eta) \in th_{\left(\sigma^l,\rho^r\right)}$ and any equivalent pair of substitutions (σ,ρ) with $\{k_1,\ldots,k_n\} \subseteq kn\left(e^{\sigma}_{\pi_i^l(th)}\right)$ and $\{l_1,\ldots,l_n\} \subseteq kn\left(e^{\rho}_{\pi_i^r(th)}\right)$, if $crux_{\left(\sigma_{\pi_i^l(th)},\rho_{\pi_i^r(th)}\right)}(\zeta,\eta) = \left(Enc_k(\zeta'),Enc_l(\eta')\right)$, then we have $fr \nvdash (k,l) \in kn\left(\left(e^{\rho}_{\pi_i^r(th)}\right) \cup \left(e^{\rho}_{\pi_i^r(th)}\right)\right)$. As a special case,

$crux\left(\left(\sigma_{\pi_i^l(th)}, x_j\right), \left(\rho_{\pi_i^r(th)}, x_j\right)\right) = crux\left(\sigma^l(x_j), \rho^r(x_j)\right)$. Therefore,

$kn\left(\left(\sigma_{\pi_i^l(th)}\right) \cup \left(\rho_{\pi_i^r(th)}\right)\right) = \left\{crux\left(\left(\sigma_{\pi_i^l(th)}, x_j\right), \left(\rho_{\pi_i^r(th)}, x_j\right)\right) \mid dom(\sigma) \ni x_j \notin dom(\rho)\right\}$

The environment $e = (fr,th)_{(\sigma^l, \rho^r)}$ pairs that only contains pairs of messages, is consistent if the attacker cannot distinguish between the messages in $\pi_i^l(th): \sigma^l(x_j)$ *and their counterparts in* $\pi_i^r(th): \rho^r(x_j)$, *for all* $i = 1, 2$ *and* $j \in I$.

The attacker can:

1- distinguish between names and encrypted messages,
2- ascertain equality of messages and
3- attempt decryption of messages with a key of his own making.

To show the equivalence of the consistent *evade* pairs with consistent pairs of substitutions. We can use the crux to get an *evade* as follows:

Note, we will use a short-hand $C(.)$ for *crux*(˙).

Definition 15: let $e = (fr,th)_{(\sigma^l, \rho^r)} \succ ok$ *be an evade frame-theory pair considered consistent with the equivalent pairs of substitutions* (σ,ρ), *such that* $\sigma \sim \rho$, *whenever* σ *and* ρ *have the same domain* $x_j = \{x_1,...,x_n\}$ *and for all i =1, 2. The following conditions hold:*

1 – $\forall x_j, dom(\sigma) \ni x_j \in dom(\rho), \exists \zeta, \eta : [\zeta^l : M] \wedge [\eta^r : M]$; *where* $n(\zeta^l) \in dom(\sigma)$ *then* $\left[\zeta^l \sigma_j^l\right] = C\left(\sigma_j^l, x_j\right)$ *and* $\left[\eta^r \rho_j^r\right] = C\left(\rho_j^r, x_j\right)$

2 – $\left(\sigma_j^l(x_j), \rho_j^r(x_j)\right) \in th$, *if* $\sigma_j^l(x_j) = \zeta$ *and* $\rho_j^r(x_j) = \eta$ *then* $e \succ \zeta \leftrightarrow \eta \Rightarrow e \succ \sigma_{\pi_i^l(th)} \sim \rho_{\pi_i^r(th)}$

3 – $C\left(\sigma_j^l(x_j)\right) \in N \Leftrightarrow C\left(\rho_j^r(x_j)\right) \in N$

4 – $C\left(\sigma_j^l(x_j)\right) = C\left(\sigma_n^l(x_n)\right) \Leftrightarrow C\left(\rho_j^r(x_j)\right) = C\left(\rho_n^r(x_n)\right)$

5 – *for* $\sigma_{\pi_i^l(th)} \sim \rho_{\pi_i^r(th)}$ *where* $dom(\sigma) = x_j = dom(\rho)$, *we let* $\left(\zeta_j^l, \eta_j^r\right) \in th_{\left(\sigma_j^l, \rho_j^r\right)}: \zeta_j^l = C\left(\sigma_j^l, x_j\right)$ *and* $\eta_j^r = C\left(\rho_j^r, x_j\right)$ *Then form that and 1, 2, we have either :*

a) $\left[\zeta_j^l \sigma_j^l\right] = \left[\eta_j^r \rho_j^r\right] = \perp$ *or*

b) There are j and a tuple $\tilde{\imath}$ *of indices from I such that :*

$$\left[\zeta_j^l \sigma_j^l\right] = Enc_{\zeta_{j_m}^l}\left(....Enc_{\zeta_{j_2}^l}\left(Enc_{\zeta_{j_1}^l}\left(\zeta_j^l\right)\right).....\right)$$

$$\left[\eta_j^r \rho_j^r\right] = Enc_{\eta_{j_m}^r}\left(....Enc_{\eta_{j_2}^r}\left(Enc_{\eta_{j_1}^r}\left(\eta_j^r\right)\right).....\right)$$

6 – *from the above, let* $\left(\sigma_j^l(x_j), \rho_j^r(x_j)\right) \in th$ *then we have*

$$\sigma_j^l(x_j) = Enc_{C\left(\sigma_j^l, x_{j_m}\right)}\left(.....Enc_{C\left(\sigma_j^l, x_{j_2}\right)}\left(Enc_{C\left(\sigma_j^l, x_j\right)}\left(C\left(\sigma_j^l, x_j\right)\right)\right)....\right),$$

$$\rho_j^r(x_j) = Enc_{C\left(\rho_j^r, x_{j_m}\right)}\left(.....Enc_{C\left(\rho_j^r, x_{j_2}\right)}\left(Enc_{C\left(\rho_j^r, x_j\right)}\left(C\left(\rho_j^r, x_j\right)\right)\right)....\right)$$

Example 1: *let* $e = (fr,th)_{(\sigma^l, \rho^r)} \succ ok$ *be an evade frame-theory pair considered consistent with an equivalent pair of substitutions* (σ,ρ), *as in the previous Definition, and we have the following:*

$\pi_i^l(th)^\sigma = \{\left(\left(Enc_k\{\zeta_1\}/x_1\right), \left(Enc_g\{\eta_1\}/x_2\right)\right)\}$ *Then we have:*

$\pi_i^r(th)^\rho = \{\left(\left(Enc_k\{\zeta_2\}/x_1\right), \left(Enc_g\{\eta_2\}/x_2\right)\right)\}$

$(th)_{(\sigma^l, \rho^r)} = \{\left(\left(Enc_k\{\zeta_1\}/x_1\right), \left(Enc_k\{\zeta_2\}/x_1\right)\right), \left(\left(Enc_g\{\eta_1\}/x_2\right), \left(Enc_g\{\eta_2\}/x_2\right)\right)\}$ *and* $\pi_i^l(th)^\sigma \approx \pi_i^l(th)^\rho$.

To compare the knowledge of environments we use the following pre-order:

Definition 16: let e = $(fr,\ th)_{(\sigma^l,\rho^r)}$ and $e' = (fr',th')_{(\sigma^l,\rho^r)}$, e' extends e written as $e \leq e'$ if and only if $dp\left(e_{(\sigma^l,\rho^r)}\right) \subseteq dp\left(e'_{(\sigma^l,\rho^r)}\right)$. Two evade frame-theory pairs e and e' with consistent pairs of substitution such that $\sigma \approx \rho$ are assumed $\mathcal{M}$-equivalent, written $(fr,th)_{(\sigma^l,\rho^r)} \lhd\rhd (fr',th')_{(\sigma^l,\rho^r)}$ if $e \lhd\rhd e'$, $e' \lhd\rhd e$ and when $kn(e) = kn(e')$. Then $\lhd\rhd$ is transitive and reflexive by the same properties for $\subseteq$. For results on anti-symmetry, the following example and Corollary 21 are used:

Example 2: let $e_1 = \{\{(ch,ch),(\eta,\mu,g)\}, (Enc_\mu(\zeta), Enc_g(\zeta'))\}$ and $e_2 = \{\{(ch,ch),(\eta,\mu,g)\}, (Enc_\mu(\zeta), Enc_\zeta(Enc_g(\zeta')))\}$ then $e_1 \lhd\rhd e_2$. Generally, if e is evade, $e_1 \leq e$ and $e_2 \leq e$ *then* $e \cup e_1 \lhd\rhd e \cup e_2$.

Lemma 17: $e' \leq e$ *iff* $e' \subseteq S(e)$

Proof: $e' \leq e \Rightarrow e' \subseteq dp(e)$, since $e' \subseteq dp(e') \subseteq dp(e)$.

To show the other implication, assume that $e' \subseteq dp(e)$ and take any $(\zeta,\eta) \in th'_{(\sigma^l,\rho^r)}$ such that $e'_{(\sigma^l,\rho^r)} \succ \sigma^l(x_j) \leftrightarrow \rho^r(x_j)$ where $\pi_i^l(th') = \zeta = \sigma^l(x_j)$ and $\pi_i^r(th') = \eta = \rho^r(x_j)$ for all i = 1,2 and x $_j$ = $\{x_1$,...,$x_n\}$ we have $\sigma_{(th)} \approx \sigma_{(th')}$, $\rho_{(th)} \approx \rho_{(th')}$, $\sigma_{(th)} \approx \rho_{(th)} \Rightarrow \sigma_{(th')} \approx \rho_{(th')}$ where all have the same domain x_j. We show that $e_{(\sigma,\rho)} \succ \sigma(x) \leftrightarrow \rho(x)$ by induction on the derivation of $e'_{(\sigma',\rho')} \succ \sigma'(x) \leftrightarrow \rho'(x)$.

If $(\zeta,\eta) \in th'_{(\sigma^l,\rho^r)}$ then $e'_{(\sigma^l,\rho^r)} \succ \sigma^l(x_j) \leftrightarrow \rho^r(x_j)$ by the assumption. Else, by the assumption of *Max-Dec-Depth*, there are ζ',η',μ,g such that $\zeta = \mathrm{Enc}_\mu(\zeta')$ and $\eta = \mathrm{Enc}_g(\eta')$. By the induction hypothesis $e \succ \zeta' \leftrightarrow \eta'$ and $e \succ \mu \leftrightarrow g$.

An *evade* process pair is a triple $((e),P,Q)$ where $e = (fr,th)_{(\sigma^l,\rho^r)}$, fr is a frame, th is a theory evaluated under a pair of substitutions (σ^l,ρ^r) indexed by $\{l, r\}$where $dom(\sigma)=dom(\rho)$ and P and Q are processes. An *evade* relation R is a set of *evade* process pairs. We write $e \succ P\ R\ Q$ *if* $((e),P,Q) \in R$. R is consistent if e is consistent whenever $e \succ P\ R\ Q$.

Now we have got enough notations to define evade bisimilarity.

Definition 18: *A consistent evade relation R is an evade simulation if whenever:*

$e \succ P\ R\ Q$ *and* $P \xrightarrow{\mu_1} P' \Rightarrow Q \xrightarrow{\mu_2} Q'$ *with*

a) $bn(\mu_1) \cap n\left(\pi_i^l(th)\left(\sigma^l,\rho^r\right)\right) \cap n\left(\pi_i^l(fr)\right) = \emptyset = bn(\mu_2) n\left(\pi_i^r(th)\left(\sigma^l,\rho^r\right)\right) \cap n\left(\pi_i^r(fr)\right)$, *for* i=1,2 *(bound names are fresh)*

b) $\underline{\underline{[\sigma(ch(\mu_1))]}} \in \pi_i^l(fr) \wedge \underline{\underline{[\rho(ch(\mu_2))]}} \in \pi_i^r(fr)$ *for* $dom(\sigma) = dom(\rho)$, $fn(\mu_1,\mu_2) \subseteq dom(\sigma)$ *(the transitions are detected)*

then

1 – if $\mu_1 = \tau$ *then* $\mu_2 = \tau$ *and* $e \succ P'\ R\ Q'$.

2 – if $\mu_1 = ch_1(x_1)$ *and there are* ζ_1,η_1 *such that* $\underline{\underline{(\eta_1\sigma)}} = ch_1 \in \pi_i^l(fr)$, $\underline{\underline{(\zeta_1\sigma)}} = M_1$ *and* $\underline{\underline{[\eta_1 : \mathcal{N}]}}$, $\underline{\underline{[\zeta_1 : \mathcal{M}]}} \in \pi_i^l(th)\left(\sigma^l,\rho^r\right)$, *then there exists* $\mu_2 = ch_2(x_2)$ *with* ζ_2,η_2 *such that* $\underline{\underline{(\eta_2\rho)}} = ch_2 \in \pi_i^r(fr)$, $\underline{\underline{(\zeta_2\rho)}} = M_2$ *and* $\underline{\underline{[\eta_2 : N]}}$,

$[\zeta_2 : M] \in \pi_i^r(th)(\sigma^l, \rho^r)$ and for all B, ζ_1, ζ_2 with $B \subset N \times N$ consistent and

– $\pi_1(B) \setminus n(\zeta_1) = \emptyset = \pi_2(B) \setminus n(\zeta_2)$ (all new names are needed)

– $\pi_1(B) \cap (fn(P) \cup n(\pi_i^l(th)(\sigma^l, \rho^r)) \cup \pi_i^l(fr) = \emptyset = \pi_2(B) \cap (fn(Q) \cup n(\pi_i^r(th)(\sigma^l, \rho^r)) \cup \pi_i^r(fr))$
(new names are fresh)

– $(fr \cup B, th) \succ \zeta_1 \leftrightarrow \zeta_2$ (ζ_1 and ζ_2 are indistinguishable) we have $((fr \cup B, th)(\sigma, \rho)) \succ P'\left[\zeta_1/x_1\right] R\, Q'\left[\zeta_2/x_2\right]$.

3 – if $\mu_1 = (\upsilon \tilde{c}_1)\overline{ch}_1 \zeta_1$ and there is η_1 such that $fn(\eta_1) \subseteq dom(\sigma)$ and $(\eta_1 \sigma) = ch_1 \in \pi_i^l(fr)$ and $[\eta_1 : \mathcal{N}]$, $[\zeta_1 : \mathcal{M}] \in \pi_i^l(th)(\sigma^l, \rho^r)$ and $\{\tilde{c}\}_1 \cap (fn(P) \cup n(\pi_i^l(th)(\sigma^l, \rho^r)) \cup \pi_i^l(fr)) = \emptyset$ then there exists $\mu_2 = (\upsilon \tilde{c}_2)\overline{ch}\ 2\ \zeta_2$ and there is η_2 such that $fn(\eta_2) \subseteq dom(\sigma)$ and $(\eta_2 \rho) = \overline{ch}_2 \in \pi_i^r(fr)$, $[\eta_2 : \mathcal{N}]$, $[\zeta_2 : \mathcal{M}] \in \pi_i^r(th)(\sigma^l, \rho^r)$ and $\{\tilde{c}_2\} \cap (fn(Q) \cup n(\pi_i^r(th)(\sigma^l, \rho^r)) \cup \pi_i^r(fr)) = \emptyset$. Then we have :
$kn\{e \cup (fr, th)(\sigma^l, \rho^r)\} \succ P' \; R \; Q'$.

In the above definition, channel correspondence is checked by adding the channels to the *fr* considered to be known by the environment. If there is no correspondence between ($ch(\pi_1(fr)) \leftrightarrow ch(\pi_2(fr))$), the resulting environment will not be consistent.

On process output we use $kn(\cdot)$ to construct the new environment after the transitions. This necessitates applying all decryptions with keys that are known by the environment, producing the minimal extension of the environment *e* with $(fr,th)(fr,th)(\sigma^l, \rho^r)$ for $(\zeta_1,\zeta_2) \in th$ and $dom(\sigma) = dom(\rho)$.

On process input, any input that the environment can construct (i.e., satisfying $(fr \cup B, th) \succ \zeta_1 \leftrightarrow \zeta_2$ must be considered. Automating bisimilarity checks is thus made difficult, since the set of potential inputs is infinite. However, due to the large number of inputs to consider, this method is practical only for finite processes. But here we showed that evade bisimilarity is decidable for finite processes (large number of inputs, but not infinite).

Definition 19: *S* is an evade bisimulation if both *S* and S^{-1} are evade simulations.

Definition 20: An evade bisimilarity is the greatest evade bisimulation, written $\approx_e$, which is the union of all evade bisimulations ($\approx_e$ is symmetric).

7.3 PROVES AND PROPERTIES OF EVADE BISIMULATION:

Here, we will give some Lemmas proving our technique, and the main properties of preorder relation.

Corollary 21: Some properties relating $\leq$ with some operations:

1– *If* $e' \subseteq e$ *then* $e' \leq e$.

2 – *If* $e'_1 \leq e$ *and* $e'_2 \leq e$ *then* $(e'_1 \cup e'_2) \leq e$.

3 – *If* $e'_1 \leq e_1$ *and* $e'_2 \leq e_2$ *then* $(e'_1 \cup e'_2) \leq (e_1 \cup e_2)$.

The two $\mathcal{M}$-equivalent *e* and e' preserved by *Kn*(.) are equal, so the preorder $\leq$ is an ordering relation on the set of them.

Lemma 22: *If* $e \lhd\rhd e'$, then $Kn(e) \leq Kn(e')$.

From the Definitions 16 and 17 we have:

Definition 23: *Evade is non-decreasable if e = kn* (*e*). *e is decreasable if e is not non-decreasable.*

An alternative definition is as follows:

Lemma 24: *An evade e is non-decreasable iff the following condition holds:*
If $(Enc_k(\zeta), Enc_g(\eta)) \in th(\sigma^l, \rho^r)$ *then* $(k,g) \notin th(\sigma^l, \rho^r)$, *where* $\pi_i^l(th)(\sigma^l, \rho^r) = \sigma^l(x), \pi_i^r(th)(\sigma^l, \rho^r) = \rho^r(x)$ *and* $dom(\sigma^1) = dom(\rho^r)$, for *i* = 1, 2.

Proof: if this holds then we cannot apply *Max-Dec-Depth* to any pair in $th_{(\sigma^l, \rho^r)}$ with an equivalent pair of substitutions, so $dp(th)_{(\sigma^l, \rho^r)} = th_{(\sigma^l, \rho^r)}$. By the definition of $kn(th)_{(\sigma^l, \rho^r)}$ we then have that $kn(th)_{(\sigma^l, \rho^r)} = th_{(\sigma^l, \rho^r)}$. If $th_{(\sigma^l, \rho^r)}$ is non-decreasable then the disjointedness holds by the definition of $kn(th)_{(\sigma^l, \rho^r)}$, using that $dp(th)_{(\sigma^l, \rho^r)} \supseteq kn(th)_{(\sigma^l, \rho^r)}$.

Corollary 25: $kn(th)_{(\sigma^l, \rho^r)}$ is non-decreasable for all evades *th*.

As might be expected, the non-decreasable function of evade can be used to generate any message that can be generated by the hedge.

Lemma 26: *For any evade e,* $e \le dp(e)_{(\sigma^l,\rho^r)} \lhd\rhd kn(e)_{(\sigma^l,\rho^r)}$.

Proof: As $e \subseteq dp(e)_{(\sigma^l,\rho^r)} \supseteq kn(e)_{(\sigma^l,\rho^r)}$ Corollary 21 gives that $e \le dp(e)_{(\sigma^l,\rho^r)} \ge kn(e)_{(\sigma^l,\rho^r)}$. What remains to be proved is $dp(e)_{(\sigma^l,\rho^r)} \ge kn(e)_{(\sigma^l,\rho^r)}$. By Lemma 17, it suffices to show $dp(e)_{(\sigma^l,\rho^r)} \subseteq S(kn(e)_{(\sigma^l,\rho^r)})$. Assume that $(\zeta,\eta) \in kn(e)_{(\sigma^l,\rho^r)}$, we show that $kn(e) \succ \zeta \leftrightarrow \eta$ by structural induction on ζ. If $\zeta \in N$ then $(\zeta,\eta) \in kn(e)_{(\sigma,\rho)}$. Otherwise there are ζ' and k such that $\zeta = Enc_k(\zeta')$. If $(\zeta,\eta) \in kn(e)_{(\sigma^l,\rho^r)}$. Otherwise, by definition of *kn* (.) there are η' and g such that $\eta = Enc_g(\eta'), (\zeta',\eta') \in dp(e)_{(\sigma^l,\rho^r)}$ *and* $(k,g) \in dp(e)_{(\sigma^l,\rho^r)}$. By the induction hypothesis we have $kn(e)_{(\sigma^l,\rho^r)} \succ k \leftrightarrow g$ and $kn(e)_{(\sigma^l,\rho^r)} \succ \zeta' \leftrightarrow \eta'$, so by *Max-Dec-Depth* we have $kn(e)_{(\sigma,\rho)} \succ \zeta \leftrightarrow \eta$.

Non-decreasable evade is a subset of any M –equivalent evade.

Lemma 27: *if* $e' \lhd\rhd e$ *and* e' *is non – decreasable then* $e' \subseteq e$.

Proof: Having any $(\zeta,\eta) \in (\mathrm{th}')_{(\sigma'^l,\rho'^r)}$. As $e' \lhd\rhd e$, $e \succ \zeta \leftrightarrow \eta$. We have two cases:

- If $\zeta \in \mathcal{N}$ *or* $\eta \in \mathcal{N}$ then $(\zeta,\eta) \in (\mathrm{th}')_{(\sigma'^l,\rho'^r)}$ by Definition 7- (iii).

- Else, $\zeta = Enc_k(\zeta')$ and $\eta = Enc_g(\eta')$. Since e' is non-decreasable $(k,g) \notin e'$ by Lemma 24. By Definition 7- (iii) $e' \not\succ k \leftrightarrow g$, *so* $e \not\succ k \leftrightarrow g$ and *Max-Dec-Depth* cannot derive $e \succ \zeta \leftrightarrow \eta$. This shows that $(\zeta,\eta) \in e$.

Two M-equivalent non-decreasable evades are equal, so the preorder is $\le$ an ordering relation in the set of non-decreasable evades.

Corollary 28: If $(e)_{(\sigma^l,\rho^r)} \lhd\rhd (e')_{(\sigma'^l,\rho'^r)}$ and both *e* and e' are non-decreasable preserved by *Kn*(.), then $e = e'$.

The ordering of evades is preserved by *kn* (.).

Lemma 29: *If* $e' \le e$, *then* $kn(e')_{(\sigma'^l,\rho'^r)} \le kn(e)_{(\sigma^l,\rho^r)}$.

Proof: from Lemma 26 $kn(e')_{(\sigma'^l,\rho'^r)} \le dp(e)_{(\sigma^l,\rho^r)}$, so by the transitivity of $\le$ we only need to show $dp(e')_{(\sigma'^l,\rho'^r)} \le kn(e)_{(\sigma^l,\rho^r)}$. By Lemma 17 this holds iff $dp(e') \subseteq S(kn(e))$, which we show by induction on the derivation of $dp(e')_{(\sigma'^l,\rho'^r)}$. Take any $(\zeta,\eta) \in dp(e')_{(\sigma'^l,\rho'^r)}$, the base case is that $(\zeta,\eta) \in (e')_{(\sigma'^l,\rho'^r)}$. By Lemma 26 $(e)_{(\sigma^l,\rho^r)} \le kn(e)_{(\sigma^l,\rho^r)}$, so $(e')_{(\sigma'^l,\rho'^r)} \le kn(e)_{(\sigma^l,\rho^r)}$ by the transitive of $\le$. In particular, $kn(e) \succ \zeta \leftrightarrow \eta$. Otherwise we have to use *Max-Dec-Depth* to derive $(\zeta,\eta) \in dp(e')_{(\sigma'^l,\rho'^r)}$ which means that there are *k* and *g* such that $(Enc_k(\zeta), Enc_g(\eta)) \in dp(th')_{(\sigma'^l,\rho'^r)}$ and $(k,g) \in dp(e')_{(\sigma'^l,\rho'^r)}$. By induction $kn(e)_{(\sigma^l,\rho^r)} \succ Enc_k(\zeta) \leftrightarrow Enc_g(\eta)$ and $kn(e)_{(\sigma^l,\rho^r)} \succ k \leftrightarrow g$. By Definition 7- (iii) $(k,g) \in kn(e)_{(\sigma^l,\rho^r)}$, so $(Enc_k(\zeta), Enc_g(\eta)) \notin kn(th')_{(\sigma'^l,\rho'^r)}$ by the definition of *kn*(.). Then *Max-Dec-Depth* must have been used to derive $kn(e)_{(\sigma^l,\rho^r)} \succ Enc_k(\zeta) \leftrightarrow Enc_g(\eta)$, which gives that $kn(e)_{(\sigma^l,\rho^r)} \succ \zeta \leftrightarrow \eta$.

The non-decreasable of two M-equivalent evades are equal.

Lemma 30: *if* $(e')_{(\sigma'^l,\rho'^r)} \lhd\rhd (e)_{(\sigma^l,\rho^r)}$, *then* $kn(e')_{(\sigma'^l,\rho'^r)} = kn(e)_{(\sigma^l,\rho^r)}$.

Proof: $kn(e')_{(\sigma'^l,\rho'^r)} \lhd\rhd kn(e)_{(\sigma^l,\rho^r)}$ by Lemma 7.3.9. $kn(e')_{(\sigma'^l,\rho'^r)}$ and $kn(e)_{(\sigma^l,\rho^r)}$ are both non-decreasable by Corollary 25. The equality then follows from Corollary 28.

Lemma 31: *If* e' *and* e *are evades, then* $kn\left(kn(e)_{(\sigma^l,\rho^r)} \cup (e')_{(\sigma'^l,\rho'^r)}\right) = kn\left((e)_{(\sigma^l,\rho^r)} \cup (e')_{(\sigma'^l,\rho'^r)}\right)$.

Proof: By Corollary 21-(1) $(e')_{(\sigma'^l,\rho'^r)} \leq \left((e)_{(\sigma^l,\rho^r)} \cup (e')_{(\sigma'^l,\rho'^r)}\right)$, so using Lemma 29 we have that $kn\left((e')_{(\sigma,\rho)}\right) \leq kn\left((e)_{(\sigma^l,\rho^r)} \cup (e')_{(\sigma,\rho)}\right)$. By Lemma 26 $(e')_{(\sigma'^l,\rho'^r)} \leq kn\left((e')_{(\sigma'^l,\rho'^r)}\right)$, so by transitivity $(e')_{(\sigma'^l,\rho'^r)} \leq kn\left((e)_{(\sigma^l,\rho^r)} \cup (e')_{(\sigma'^l,\rho'^r)}\right)$.Concerning $kn\left((e)_{(\sigma^l,\rho^r)}\right)$, note that $kn\left((e)_{(\sigma^l,\rho^r)}\right) \subseteq dp\left((e)_{(\sigma^l,\rho^r)}\right) \subseteq dp\left((e)_{(\sigma^l,\rho^r)} \cup (e')_{(\sigma'^l,\rho'^r)}\right) \leq kn\left((e)_{(\sigma^l,\rho^r)} \cup (e')_{(\sigma'^l,\rho'^r)}\right)$, where the last relation is due to Lemma 17. By Corollary 21-(2) $\left(kn\left((e)_{(\sigma^l,\rho^r)}\right) \cup (e')_{(\sigma'^l,\rho'^r)}\right) \leq kn\left((e)_{(\sigma^l,\rho^r)} \cup (e')_{(\sigma'^l,\rho'^r)}\right)$, so $kn\left(kn\left((e)_{(\sigma^l,\rho^r)}\right) \cup (e')_{(\sigma'^l,\rho'^r)}\right) \leq kn\left((e)_{(\sigma^l,\rho^r)} \cup (e')_{(\sigma'^l,\rho'^r)}\right)$. $(e)_{(\sigma^l,\rho^r)} \leq kn\left((e)_{(\sigma^l,\rho^r)}\right)$ by Lemma 26, so by Corollary 21-(3) we have that $\left((e)_{(\sigma^l,\rho^r)} \cup (e')_{(\sigma'^l,\rho'^r)}\right) \lhd\rhd \left(kn\left((e)_{(\sigma^l,\rho^r)}\right) \cup (e')_{(\sigma'^l,\rho'^r)}\right)$. By Lemma 29 we have that $kn\left((e)_{(\sigma^l,\rho^r)} \cup (e')_{(\sigma'^l,\rho'^r)}\right) \leq kn\left(kn\left((e)_{(\sigma^l,\rho^r)}\right) \cup (e')_{(\sigma'^l,\rho'^r)}\right)$, so $kn\left((e)_{(\sigma^l,\rho^r)} \cup (e')_{(\sigma'^l,\rho'^r)}\right) \lhd\rhd kn\left(kn\left((e)_{(\sigma^l,\rho^r)}\right) \cup (e')_{(\sigma'^l,\rho'^r)}\right)$. The equality follows from Corollary 28.

From what we have seen so far, an environment mapping should preserve consistency. To prove that this holds for the environment mappings we will define, we then investigate consistent evades and their properties.

Lemma 32: If *e* is consistent then *e* is non-decreasable.

***Proof*:** By Definition 7 for consistency, we have that $((Enc_k(\zeta)), (Enc_g(\eta))) \in (th)_{(\sigma^l,\rho^r)}$ implies $(k,g) \notin (fr)_{(\sigma^l,\rho^r)}$. By Lemma 24 this means that *e* is non-decreasable.

Note that we have only used a special case of Definition 7 in the proof of Lemma 32. The all conditions for consistency are pairwise disjointed, so consistency is a much stronger constraint than non-decreasability.

For consistency a generalized version of Definition 7 is that a consistent *evades* cannot generate two message pairs that only differ in one component.

Lemma 33: Let *e* be consistent *evade* with $e \succ \zeta \leftrightarrow \eta$ and $e \succ \zeta' \leftrightarrow \eta'$. Then $\zeta = \zeta'$ *iff* $\eta = \eta'$.

Proof: By symmetry we need only to argue the case $\zeta = \zeta'$. The proof is by induction on the derivation of $e \succ \zeta \leftrightarrow \eta$. If $(\zeta,\eta) \in (th)_{(\sigma^l,\rho^r)}$ we will first show that $(\zeta,\eta') \in (th)_{(\sigma^l,\rho^r)}$. If ζ is a name this follows from Definition 7. Otherwise $\zeta = Enc_k(M)$, but since $k \notin \pi_1^l(th)_{(\sigma^l,\rho^r)}$ by Definition 7 of consistency we cannot use *Max-Dec-Depth* to derive $e \succ \zeta \leftrightarrow \eta'$. Now, we know that $(\zeta,\eta) \in (th)_{(\sigma^l,\rho^r)}$ and $(\zeta,\eta') \in (th)_{(\sigma^l,\rho^r)}$, when $dom(\sigma^l) = dom(\rho^r)$ then $\eta = \eta'$ by Definition 7 for consistency. If $\zeta = Enc_k(M)$, $\eta = Enc_g(N)$, $e \succ M \leftrightarrow N$ and $e \succ k \leftrightarrow g$ then $k \in \pi_1^l(th)_{(\sigma^l,\rho^r)}$ by Definition 7. As *e* is consistent, $\zeta \notin \pi_1^l(th)_{(\sigma^l,\rho^r)}$, so we must have used *Max-Dec-Depth* to derive $e \succ \zeta \leftrightarrow \eta'$. This gives that $\eta' = Enc_a(N')$ for some *a, N'* such that $e \succ M \leftrightarrow N'$ and $e \succ k \leftrightarrow a$. By induction $N = N'$ and $g = a$.

Two M-equivalent consistent evades are always equal.

Lemma 34: *If* $e' \lhd\rhd e$ *and both* e' *and e are consistent, then* $e' = e$.

Proof: e' and *e* are non-decreasable by Lemma 32. The equality follows from Corollary 28.

Any non-decreasable of a consistent *evade* is consistent.

Lemma 35: *If e is consistent,* e' *is non-decreasable and* $e' \leq e$ *then* e' *is consistent.*

Proof: Assume that $(\zeta,\eta) \in (th')_{(\sigma'^l,\rho'^r)}$ and note that $e \succ \zeta \leftrightarrow \eta$ when $\zeta = \sigma'^l(x)$ *and* $\eta = \rho'^r(x)$. We only need to show one direction of the symmetric conditions.

1. If $\zeta \in N$ then $(\zeta,\eta) \in (th)_{(\sigma^l,\rho^r)}$ by Definition 7, so $\eta \in \mathcal{N}$ as *e* is consistent.

2. See Lemma 33.

3. Assume that $\zeta = Enc_k(M)$. If $(\zeta,\eta) \in (th)_{(\sigma^l,\rho^r)}$ then $k \notin \pi_1^l(th)_{(\sigma^l,\rho^r)}$ by Definition 7 for consistency, so $k \notin \pi_1^l(th')_{(\sigma'^l,\rho'^r)}$. Else *Max-Dec-Depth* has been used to derive $e \succ \zeta \leftrightarrow \eta$ when $\zeta = \sigma'^l(x)$ *and* $\eta = \rho'^r(x)$, so $\eta = Enc_g(N)$ where $e \succ k \leftrightarrow g$ and $e \succ M \leftrightarrow N$. By Lemma 24 $(k,g) \in e'$ would contradict that e' is un-decreasable. If $\eta \neq g$ we have by Lemma 33 $e \not\succ k \leftrightarrow \eta$. This gives that $(k,\eta) \notin e'$ so $k \notin \pi_1^l(th')_{(\sigma'^l,\rho'^r)}$.

Disjointed consistent evades may be directly combined.

Lemma 36: *If e' and e are consistent and n (e') ⋂ n (e) = ∅, then $e' \cup e$ is consistent.*

Proof: Take any message pair such that $(\zeta,\eta) \in \left((th')_{(\sigma'^l,\rho'^r)} \cup (th)_{(\sigma^l,\rho^r)} \right)$. By symmetry we may assume that $(\zeta,\eta) \in (th')_{(\sigma'^l,\rho'^r)}$.

1. $\zeta \in N \Leftrightarrow \eta \in N$ is clear, since e' is consistent.

2. Take any $(\zeta',\eta') \in \left((th')_{(\sigma'^l,\rho'^r)} \cup (th)_{(\sigma^l,\rho^r)} \right)$. As $n(\zeta) \neq \emptyset \neq n(\eta)$ we have that $(\zeta',\eta') \in (th')_{(\sigma'^l,\rho'^r)}$ whenever $\zeta = \zeta'$ or $\eta = \eta'$. As e' is consistent, $\zeta = \zeta'$ iff $\eta = \eta'$.

3. If $\zeta = Enc_k(\zeta')$ and $\eta = Enc_g(\eta')$ then $k \notin \pi_1(th')_{\sigma'}$ and $g \notin \pi_i^r(th')_{(\sigma'^l,\rho'^r)}$ as e' is consistent. As $\{k,g\} \subseteq n(e')$ we have that $\{k,g\} \cap n(e) \cap fr = \emptyset$ and as a special case of this, $k \notin \pi_i^l(th)_{(\sigma^l,\rho^r)}$ and $g \notin \pi_i^r(th)_{(\sigma^l,\rho^r)}$.

Conclusion And Future works

The importance of reasoning about knowledge for understanding distributed computations in the field of process algebras has been given some emphasis in recent literature [3, 4, 5, 6, 9, 10, 11, 13]. Furthermore, there have been some formal descriptions of cryptographic protocols [2, 7, 8, 21, 22, 23, 24, 25, 26]. These works have suggested some useful proof techniques. Framed bisimulation is one of the techniques introduced by Abadi and Gordon [2], in which they represent the knowledge of the environment with which the protocol interacts in the form of a frame-theory pair.

Abadi and Gordon defined the *Framed bisimilarity* method as a trend to prove process equivalence based process-environment interactions. Their work represented a restricted attempt that necessitated the existence levels of quantification over infinite domains

Our work follows Abadi and Gordon's framed bisimulation proposal. It introduces a new convenient proof technique called *evade bisimulation*. This technique is proved to detect the limit beyond which the bisimilarity is kept valid and avoids quantification over contexts for output transitions and a finite number of input transitions.

In addition, we have shown that our new bisimilarity method based on spi-$_H$-calculus coincides with testing equivalence and we proved that evade bisimulation is a decidable technique for main cryptographic protocols properties, namely: *authenticity* and *confidentiality*. Also, we deem that our technique is simply can be a step toward automation, because of its strong structure in spi-$_H$-calculus.

For further work, we believe that it will not be difficult to build evade bisimulation on public-key cryptography. This is something we may look into in the future. We are greatly interested in developing our method toward a symbolic evades bisimulation technique that may enable the automatic verification of cryptographic protocols.

References:

[1] M. Abadi and A. D. Gordon. A Calculus for Cryptographic Protocols: The Spi Calculus. *Journal of Information and. Computation,* 148(1): 1—70, 1999. An extended abstract appeared in the *Proceedings of the Fourth ACM Conference on Computer and Communications Security (Zurich, April 1997).* An extended version of this paper appears as Research Report 149, Digital Equipment Corporation Systems Research Center[A] January 1998, and, in preliminary form, as Technical Report 414, University of Cambridge Computer Laboratory, January 1997.

[2] M. Abadi and A. D. Gordon. A Bisimulation Method for Cryptographic Protocols. *Nordic Journal of Computing,* 5(4):267-303, Winter 1998. An extended abstract appeared in the *Proceedings of ESOP '98,* LNCS 1381, pages 12-26.

[3] M. Abadi and A. D. Gordon. Reasoning about cryptographic protocols in the spi calculus. In *Proc. of CONCUR '97.* pages 59-73. LNCS 1243, 1997.

[4] M. Abadi. Two facets of authentication. In *Proceedings of IIth IEEE Computer Security Foundations Workshop,* pages 25 32. IEEE press. 1998.

[5] M. Burrows, M. Abadi, and R. Needham. A Logic of Authentication. *ACM Transactions on Computer Systems,* pages 18 36, February 1990.

[6] Robin Milner. The polyadic π-calculus: a tutorial. Technical Report ECS-LFCS-91-180, Laboratory for Foundations of Computer Science, Department of Computer Science, University of Edinburgh, UK, October 1991. Appeared in *Proceedings of the International Summer School on Logic and Algebra of Specification,* Marktoberdorf, August 1991. Reprinted in *Logic and Algebra of Specification,* ed. F. L. Bauer, W. Brauer, and H. Schwichtenberg, Springer-Verlag, 1993.

[7] Hasan M. Al-refai, Tengku M. T. Sembok & Mohammed Yusoff. 2004. Decidablity of Cryptographic Protocol Properties through Framed bisimulation. *Proc. International conference on Informatics.* **2**(1): 779-796.

[8] M. Boreale, R. De Nicola and R. Pugliese. Proof Techniques for Cryptographic Processes. *SIAMJournal on Computing,* 2002.

[9] Rocco De Nicola and Matthew C. B. Hennessy. Testing equivalence for processes. In Josep Diaz, editor, *Automata, Languages and Programming, 10th Colloquium,* volume 154 of *Lecture Notes in Computer Science,* pages 548-560, Barcelona, Spain, 18-22 July 1983. Springer-Verlag.

[10] D. Gollmann. What do we mean by Entity Authentication. In *Proceedings of the 1996 IEEE Symposium on Security and Privacy,* pages 46-54. IEEE Computer Society Press, 1996.

[11] E.M. dark and J. Jacob. A survey of authentication protocol literature. 1997.

[12] R. Focardi, A. Ghelli, and R. Gorrieri. Using non interference for the analysis of security protocols. In *Proceeding of the DIMACS Workshop on Design and Formal Verification of Security Protocols,* DIMACS Center, Rutgers University, September 1997.

[13] J. C. Mitchell, M. Mitchell, and U. Stem. Automated analysis of cryptographic protocols using Murϕ. In *Proceedings of the 1997 IEEE Symposium on Research in Security and Privacy,* pages 141- 153. IEEE Computer Society Press, 1997.

[14] R. Kemmerer, C. Meadows, and J. Millen. "Three systems for cryptographic protocol analysis". *J. Cryptology,* 7(2):79-130, 1994.

[15] A. Durante, R. Focardi, and R. Gorrieri. CVS: A compiler for the analysis of cryptographic protocols". In *Proceedings of 12th Computer Security Foundations Workshop.* IEEE CS Press, 1999.

[16] R. A. Kemmerer. Analyzing encryption protocols using formal verification techniques. *IEEE Journal on Selected Areas in Communications, 7,* 1989.

[17] C. Meadows. Applying formal methods to the analysis of a key management protocol. *Journal of Computer Security,* 1(1):5-36, 1992.

[18] L. Paulson. Proving properties of security protocols by induction. In *Proceedings of the 10th IEEE Computer Security Foundations Workshop,* pages 70-83, 1997.

[19] R. Milner. *Communicating and Mobile Systems: the π-Calculus.* Cambridge University Press, May 1999.

[20] R. Milner, J. Farrow, and D. Walker. A calculus of mobile processes, parts I and n. *Information and Computation,* pages 1-40 and 41-77, September 1992.

[21] D. Park. Concurrency and automata on infinite sequences. In P. Deussen, editor, *Theoretical Computer Science: 5th G I-Conference, Karlsruhe,* volume 104 *of Lecture Notes in Computer Science,* pages 167-183. Springer-Verlag, March 1981.

[22] M. Boreale, R. De Nicola and R. Pugliese. Proof Techniques for Cryptographic Processes. *In Proceedings ofLICS '99,* pages 157-166. IEEE, Computer Society Press, July 1999.

[23] A. S. Elkjaer, M. Hohle, H. Huttel and K. Overgard. Towards Automatic Bisimilarity Checking in the Spi Calculus. In C. S. Calude and M. J. Dinneen, eds, *Combinatorics, Computation & Logic,* volume 21(3) of *Australian Computer Science Communications,* pages 175-189. Springer-Verlag Singapore Pte. Ltd., Jan. 1999. As part of the *Australian Computer Science Week,* January 18-21, 1999.

[24] U. Frendrup, H. Huttel and J. N. Jensen. Two Notions of Environment Sensitive Bisimilarity for Spi-Calculus recesses, http://www.cs.auc.dk/research/FS/ny/PR-pi/ESB/twoNotionsOfESB.ps, 2001.

[25] Johannes Borgstrom and Uwe Nestmann. On bisimulations for the spi calculus. In Helene Kirchner and Christophe Ringeissen, editors, *Proceedings of AMAST g002,* volume 2422 of *LNCS,* pages 287-303. Springer, 2002.

[26] Hans Huttel. Deciding framed bisimilarity. In Antonin Kucera and Richard Mayr, editors, *Proceedings of INFINITY 2002,* volume 68 of *ENTCS.* Elsevier Science Publishers, 2002.

[27] R. Focardi and R. Gorrieri. A classification of security properties. *Journal of Computer Security,* 3(1), 1995.