Compositional verification of integrity for digital stream signature protocols *

Roberto Gorrieri
Dipartimento di Scienze dell'Informazione
Università di Bologna, Italy
gorrieri@cs.unibo.it

Fabio Martinelli, Marinella Petrocchi, Anna Vaccarelli
Istituto di Informatica e Telematica – C.N.R.
Area della Ricerca di Pisa, Italy
{fabio.martinelli,marinella.petrocchi,anna.vaccarelli}@iit.cnr.it

Abstract

We investigate the application of concurrency theory notions as simulation relations and compositional proof rules for verifying digital stream signature protocols. In particular, we formally prove the integrity of the Gennaro-Rohatgi protocols in [7]. As a peculiarity, our technique is able to check a protocol with an unbounded number of parallel processes. We argue also that our approach may be applied to a wider class of stream signature protocols.

1. Introduction

One of the main challenges of securing multicast/broadcast communication is integrity, or enabling receivers of multicast data to verify that the received data was not modified en-route. The problem becomes more complex in common settings where other receivers of the data are not trusted and where lost packets are not re-transmitted. Dealing with multicast/broadcast information means, in the terminology currently present in the literature, dealing with digital streams, i.e. long (potentially infinite) sequence of bits. Applications that deal with streams are typically live-broadcasts, digitized audio and video, data feeds, applets, multi-party video games, multi-party video conferences. For successful developments, these applications could require security properties to be verified (i.e., confidentiality and integrity).

The use of formal techniques to analyze stream signature protocols represents an interesting challenge because of the diversity of such protocols from standard cryptographic schemes. Indeed, two peculiarities are: *i*) a sender broadcasts a continuous (and possibly unbounded) stream of messages to a possibly unbounded set of receivers; *ii*) receivers use information retrieved in earlier packets to legitimate later packets or vice-versa. Some particular scenarios may include mobility, i.e participants through the protocol can move and change dynamically. Currently, the successful outcome of formal techniques applied to these classes of protocols is in question. In [2] Archer states a formal analysis based on model checking techniques is not feasible and exploits theorem proving techniques to analyze a well known stream authentication protocol (the TESLA protocol [14]). On the other hand, Broadfoot and Lowe show their successful results derived applying model checking techniques on [14], motivating, even though informally, several steps of the analysis, [3].

The aim of this paper is to apply compositional proof rules to verify integrity in digital streams, where integrity means, informally, that the information accepted by a receiver is exactly what the sender has intended. The compositional rules were first discussed in [10] for the GNDC schema of properties, defined in [5, 6]. In turn, the schema is based on the notion of non-interference, [8].

A compositional proof rule for checking a property f works as follows: in order to check if a system $P \mid Q$ satisfies f it is enough to check whether both P and Q satisfy f. (Notation $P \mid Q$ represents the parallel composition of subsystems P and Q). The compositional reasoning is useful in many cases. For instance, the state-space of the system $P \mid Q$ is usually considerably bigger than those for P and Q. Hence, formal verification techniques as model checking benefit of such compositional rules. Another field of application is the analysis of systems with an unbounded number of equal components. Let us consider the parallel

*Work partially supported by a grant of Microsoft Research Europe (Cambridge); by MIUR project "Formal Methods for Security and Time (MEFISTO)" and by a CSP grant for the project "SeTAPS II".

composition of equal processes P:

$$\overbrace{P \mid \ldots \mid P}^{n}$$

To prove that this system enjoys f (for whatever n) it is sufficient to prove that P enjoys f.

The main target of our (compositional) analysis is the Gennaro-Rohatgi protocol, developed to sign digital streams, [7]. It is our opinion that the protocol should be considered paradigmatic, being essentially, in its 1997 version, one of the first proposals to efficiently solve the problem to sign digital streams. Efficient cryptographic solutions (i.e. fast to be computed and verified) have been adopted in the protocol to allow the entities at stage to minimize their communication and computation overhead. Protocols dealing with the problem of securing streamed data over channels with packet loss have been recently proposed, [9, 13, 15]. They all can be basically considered as valuable extensions of the Gennaro-Rohatgi constructions.

The main contributions of this paper are the following: i) the Gennaro-Rohatgi stream signature protocol has been formally analyzed with compositional proof rules and the results are reported in this paper. To the best of our knowledge this is the first attempt to prove the integrity of digital streams by means of compositional rules; ii) our analysis aims in allowing the modeling and formal validation of a set of multicast/broadcast protocols, embracing the basic schemes of [7] as well as further extensions regarding a scheme of stream signature dealing with packet loss; iii) contrary to previous work in the area, [2, 3], we are able to check a system with an unbounded number of receivers. (The target of our analysis however being different from TESLA.)

The paper is organized as follows. In Section 2, we present the formal language we use for the description of cryptographic protocols. In Section 3, we describe the Gennaro-Rohatgi stream signature protocol in more detail. Section 4 recalls two general schemes for defining security properties, first introduced in [5, 6] and illustrates some compositional results to establish if a system enjoys security properties defined by means of the general schemes. Section 5 shows how to apply the compositional results to successfully prove the correctness of the Gennaro-Rohatgi off-line solution in terms of packets' integrity. The compositional analysis can be efficiently extended to the case of a scenario with an unbounded number of receivers, as Subsection 5.1 shows. In Section 6 we outline our ongoing research regarding the extension of this kind of analysis to a class of protocols dealing with packet loss.

2. A formal language for the description of protocols: Crypto-CCS

A language, a slight modification of CCS process algebra [12], is adopted for the description of cryptographic protocols. It makes use of cryptographic-modeling constructs and deals with confidential values, hence the name Crypto-CCS [6]. Other languages have been developed in the past for the description of cryptographic protocols, e.g. [1]. The Crypto-CCS model consists of a set of sequential agents able to communicate by exchanging messages. The data handling part of the language consists of messages and inference systems.

Messages are the data manipulated by agents, they form a set *Msgs* of terms possibly containing variables. The set *Msgs* is defined by the grammar:

$$m ::= x \mid b \mid F^1(m_1, \ldots, m_{k_1}) \mid \ldots \mid F^l(m_1, \ldots, m_{k_l})$$

where F^i (for $1 \leq i \leq l$) are the constructors for messages, $x \in V$, a countable set of variables, $b \in B$, a collection of basic messages, and k_i, for $1 \leq i \leq l$, gives the number of arguments of the constructor F^i. Messages without variables are *closed* messages.

Inference systems model the possible operations on messages. These systems consist of a set of rules r:

$$r = \frac{m_1 \quad \ldots \quad m_n}{m_0}$$

where $m_1, \ldots, m_n$ is a set of premises (possibly empty) and m_0 is the conclusion. An instance of the application of the rule r to closed messages m_i is denoted as $m_1 \quad \ldots \quad m_n \vdash_r m_0$. Given an inference system, we can define a *deduction function* $\mathcal{D}$ s.t. if ϕ is a finite set of closed messages, then $\mathcal{D}(\phi)$ is the set of closed messages that can be deduced starting from ϕ by applying instances of the rules in the system.

In Table 1 we show a suitable inference system we are going to use for the formalization of digital streams' protocols. Rule $(pair)$ builds the pair of two messages x and y; rules (fst) and (snd) are used to obtain the components of a pair; rule $(sign)$ allows message x to be digitally signed by applying the secret key $sk(y)$ of agent y (talking about digital signatures, we assume the use of asymmetric algorithms like the one proposed in [16]); rule (ver) allows a digital signature $\{x\}_{sk(y)}$ to be verified by applying the public key of signer y, $pk(y)$; rule $(hash)$ allows an agent to apply a one-way hash function to message x and obtain digest $h(x)$. Throughout the paper, hash functions enjoy the following assumptions: i) they map arbitrarily long binary strings into strings of a fixed length; ii) they are "collision resistant", i.e. only with negligible probability it is possible to obtain the same output from two different inputs; iii) they are not reversible (with high probability).

$\frac{x \quad y}{Pair(x,y)}(pair)$	$\frac{Pair(x,y)}{x}(fst)$	$\frac{Pair(x,y)}{y}(snd)$
$\frac{x \quad sk(y)}{\{x\}_{sk(y)}}(sign)$	$\frac{\{x\}_{sk(y)} \quad pk(y)}{x}(ver)$	$\frac{x}{h(x)}(hash)$

Table 1. A simple inference system.

The control part of our language consists of compound systems, basically sequential agents running in parallel. The terms of our language are generated by the following grammar:

COMPOUND SYSTEMS:	$S ::= S \setminus C \mid (S_1 \mid S_2) \mid A_\phi$
SEQUENTIAL AGENTS:	$A ::= \mathbf{0} \mid p.A \mid [m_1 \dots m_n \vdash_r x]A$
	$\mid [m = m']A \mid A(x_1, \dots, x_n)$
PREFIX CONSTRUCTS:	$p ::= c!m \mid c?x$

where $m, m', m_1, \dots, m_n$ are *closed* messages or variables, $x, x_1, \dots, x_n$ are message variables, $c \in Ch$, a finite set of channels, ϕ is a finite set of *closed* messages, C is a subset of Ch. The informal semantics of sequential agents and compound systems is as follows:

- $\mathbf{0}$ is the process that does nothing.
- $p.A$ is the process that can perform an action according to the particular prefix construct p and then behaves as A:
  - $c!m$ denotes a message m sent on channel c;
  - $c?x$ denotes the receiving of a message m on channel c. The received message replaces the variable x.
- $[m_1 \dots m_n \vdash_r x]A$ is the inference construct[1]. If, by applying an instance of rule r with premises $m_1 \dots m_n$, a message m can be inferred then the process behaves as A (where m replaces x). This is the message-manipulating construct of the language. For instance,

  $$[m \quad sk(y) \vdash_{sign} x]A$$

  is the process that uses the rule $sign$ to obtain a digitally signed message from m and $sk(y)$ and then behaves like A.
- $[m = m']A$ is the match construct to check message equality, i.e. the if $m = m'$ then it behaves as A otherwise it gets stuck.
- $A(m_1, \dots, m_n)$ is a constant process whose behavior is defined through a constant definition $A(x_1, \dots, x_n) =_{def} P$ where all the free variables of P are in $\{x_1, \dots, x_n\}$.

[1]The symbol $\vdash_r$ is syntactic sugar for recording which rule is used.

- A compound system $S \setminus C$ allows only visible actions whose channel is not in C.
- A compound system $S_1 \mid S_2$ denotes the parallel execution of S_1 and S_2. $S_1 \mid S_2$ performs an action p if one of its sub-components performs p. A synchronization or internal action, denoted by the symbol τ, may take place whenever S_1 and S_2 are able to perform two complementary actions, i.e. send-receive actions on the same channel.
- The term A_ϕ is a single sequential agent whose knowledge, i.e. the set of messages which occur in its term, is described by ϕ. The agent's knowledge increases either when it receives messages (see rule (?) in Tab. 2) or infers new messages from the messages it knows (see rule $\mathcal{D}$ in Tab. 2). For every sequential agent A_ϕ, we require that all the closed messages that appear in A belong to its knowledge ϕ.

2.1 Operational semantics and auxiliary notions.

The agents' activities are described by the actions they can perform. The set Act of actions which may be performed by a compound system is defined as: $Act = \{c?m, c!m, \tau \mid c \in C, m \in Msgs, m \text{ closed}\}$. As usual, we use a Labeled Transition System (LTS) to assign semantics to our language. The semantics of *closed* terms is given by the least set of action relations induced by the rules shown in Tab. 2. For notational convenience, we sometimes use $S\|_C X$ instead of $(S \mid X)\setminus C$. As a notation we also use $S \Longrightarrow S'$ for denoting that S and S' belong to the reflexive and transitive closure of $\xrightarrow{\tau}$; $S \stackrel{\gamma}{\Longrightarrow} S'$ if γ is a finite sequence of actions $\alpha_i, 1 \leq i \leq n$ s.t. $\alpha_i \neq \tau$ and $S \stackrel{\tau}{\Longrightarrow}\xrightarrow{\alpha_1}\Longrightarrow \dots \Longrightarrow\xrightarrow{\alpha_n}\stackrel{\tau}{\Longrightarrow} S'$.

As behavioral relations among Crypto-CCS terms, in the following we will be mainly interested in trace inclusion (equivalence) and (weak) simulation.

Definition 1 *We say that the traces of P are included in the traces of Q ($P \leq_{trace} Q$) whenever, if $P \stackrel{\gamma}{\Longrightarrow} P_1$ then $Q \stackrel{\gamma}{\Longrightarrow} Q_1$. We write that $P_{=_{trace}}Q$ iff $P \leq_{trace} Q$ and $Q \leq_{trace} P$.*

Definition 2 *We say that a relation $\mathcal{R}$ among Crypto-CCS processes is a (weak) simulation whenever if $(P, Q) \in \mathcal{R}$ and $P \xrightarrow{a} P_1$ then $Q \stackrel{a}{\Longrightarrow} Q_1$ and $(P_1, Q_1) \in \mathcal{R}$.*

The union of all weak simulations is a weak simulation and it is denoted by $\prec$.

As usual, it holds that if $P \prec Q$ then $P \leq_{trace} Q$.

$$(!)\frac{}{(c!m.A)_\phi \xrightarrow{c!m} (A)_\phi} \qquad (|_1)\frac{S \xrightarrow{\alpha} S'}{S \mid S_1 \xrightarrow{\alpha} S' \mid S_1}$$

$$(?)\frac{m \in Msgs}{(c?x.A)_\phi \xrightarrow{c?m} (A[m/x])_{\phi\cup\{m\}}} \qquad (|_2)\frac{S \xrightarrow{c!m} S' \quad S_1 \xrightarrow{c?m} S'_1}{S \mid S_1 \xrightarrow{\tau} S' \mid S'_1}$$

$$(\mathcal{D})\frac{m_1 \ldots m_n \vdash_r m \quad (A_1[m/x])_{\phi\cup\{m\}} \xrightarrow{\alpha} (A'_1)_{\phi'}}{([m_1 \ldots m_n \vdash_r x]A_1)_\phi \xrightarrow{\alpha} (A'_1)_{\phi'}} \qquad (\backslash_1)\frac{S \xrightarrow{c!m} S' \quad c \notin L}{S \setminus L \xrightarrow{c!m} S' \setminus L}$$

$$(=)\frac{m = m' \quad (A_1)_\phi \xrightarrow{\alpha} (A'_1)_{\phi'}}{([m = m']A_1)_\phi \xrightarrow{\alpha} (A'_1)_{\phi'}}$$

$$(Const)\frac{A(x_1, \ldots, x_n) =_{def} P \quad P[m_1/x_1, \ldots, m_n/x_n] \xrightarrow{\alpha} P_1}{A(m_1, \ldots, m_n) \xrightarrow{\alpha} P_1}$$

Table 2. Operational semantics, where the symmetric rules for $|_1, |_2, \backslash_1$ are omitted.

3. The Gennaro-Rohatgi Protocol

In [7], Gennaro and Rohatgi developed a mechanism to sign digital streams, i.e. long (potentially infinite) sequence of bits. They aim at assuring a receiver that the information he received is exactly what the sender has intended. Applications that deal with streams are typically digitized audio and video, data feeds, applets. This kind of applications requires the user to consume the data he receives at almost the input rate, without excessive delay. For this reason, signing digital streams represents a different problem compared with the signature of finite messages. Traditional digital signature schemes do not fit properly because they require the receiver to process the entire message in order to verify the signature.

The authors present two solutions to the problem, distinguishing two cases: i) a finite stream which is entirely known to the sender (e.g. a movie); ii) a potentially infinite stream not known in advance to the sender (e.g. a live broadcast).

We discuss both the proposed schemes below. Both the schemes rely on the basic idea to divide the stream into blocks and to add cryptographic information in each block such that receivers use information retrieved in earlier blocks to legitimate later blocks.

We first use an intuitive notation usually found in literature. We consider a set of agents able to receive messages. Basically we represent the sending and reception of a message msg from a sender A to a receiver B in the following way:

$$label \quad c_j \quad A \to B \quad : \quad msg$$

where msg is the exchanged message, c_j is the j-th communication channel, over which the exchange takes place. A and B are the sender and the receiver of msg and $label$ is the name of msg.

3.1 The off-line solution

We consider a set $\{b_i\} \in Msgs$ [2] of meaningful payloads, $i = 1 \ldots l$. The protocol for the off-line case can be formalized as follows[3] [4]:

Block $b_0^{'}$	c_0	S	$\rightarrow$	R	$: \{h(b_1^{'})\}_{sk(S)}$
Block $b_i^{'}$	c_i	S	$\rightarrow$	R	$: b_i, h(b_{i+1}^{'})\ i = 1..l-1$
Block $b_l^{'}$	c_l	S	$\rightarrow$	R	$: b_l$

The protocol exploits the technique of embedding the hash of the following block in the current block. Bootstrapping integrity of the digital stream is obtained by applying a single traditional signature in combination with hash chaining.

The notation has to be intended in the following way: $h(m)$ is the digest of m after applying the hash function; $\{m\}_{sk(S)}$ is message m digitally signed by the sender's private key $sk(S)$.

The protocol works as follows. The sender S first divides the stream to be sent in l blocks. He then generates the digital signature on the hash of the first block (Block $b_0^{'}$). After verification of the signature the receiver knows what the hash of the first block should be and then starts receiving the full stream (blocks $b_i^{'}$). When the receiver receives the first block $b_1^{'}$, he computes its hash and checks the hash against what the signature was verified upon. The other blocks consist of an authentication chain, in which

[2] We assume that the sender's private key $sk(S)$ does not occur in the set $\{b_i\}$.

[3] In the original paper [7], the first block contains an encoding of the length of the stream. The structure of the first block is here simplified (without however affecting the results of our analysis). We assume the receiver knows in advance the number of blocks in which the stream is divided.

[4] To avoid replay attacks when executing multiple runs of the protocol one can simply include nonces in the digitally signed block.

each block contains the hash of the subsequent block. Embedding the hash of the subsequent block implies that the sender knows the stream in advance, hence the non feasibility of this construction for applications like live broadcasts.

We present the Crypto-CCS specifications of the given protocol. The sender and receiver nodes are modeled as Crypto-CCS processes. They can perform sendings and receptions according to the protocol original specifications. The Crypto-CCS representation is very flexible: checks about the received blocks are explicitly represented.

Each conclusion of an inference construct is a message variable. With notation x_m we mean "variable x should contain message m".

$$\begin{array}{rcl} Sender_0 & \doteq & [b_1^{'} \vdash_{hash} x_{h(b_1^{'})}] \\ & & [x_{h(b_1^{'})} \quad sk(S) \vdash_{sign} x_{b_0^{'}}] \\ & & c_0!x_{b_0^{'}}.Sender_1 \\ Sender_i & \doteq & [b_{i+1}^{'} \vdash_{hash} x_{h(b_{i+1}^{'})}] \\ & & [b_i \quad x_{h(b_{i+1}^{'})} \vdash_{pair} x_{b_i^{'}}] \\ & & c_i!x_{b_i^{'}}.Sender_{i+1} \\ Sender_l & \doteq & c_l!b_l.\mathbf{0} \end{array}$$

The sender initially builds the initialization block $b_0^{'}$ (more precisely, he builds a variable containing $b_0^{'}$) to bootstrap the chain: by means of inference rules $hash$ and $sign$ in Table 1 he computes block $b_0^{'}$, sends it on communication channel c_0 and travels to the next state $Sender_i$. The sender can now send significative payloads b_i together with hashed blocks $h(b_{i+1}^{'})$ until he reaches the last state l.

The receiver process is parameterised by the hashed blocks he receives from the sender (more precisely, variables that should contain the hashed blocks).

$$\begin{array}{rcl} Receiver_0(null) & \doteq & c_0?x_{b_0^{'}}.[x_{b_0^{'}} \quad pk(S) \vdash_{ver} x_{h(b_1^{'})}] \\ & & Receiver_1(x_{h(b_1^{'})}) \\ Receiver_i(x_{h(b_i^{'})}) & \doteq & c_i?x_{b_i^{'}}.[x_{b_i^{'}} \vdash_{hash} x_{h(b_{MYi}^{'})}] \\ & & [x_{h(b_i^{'})} = x_{h(b_{MYi}^{'})}][x_{b_i^{'}} \vdash_{fst} x_{b_i}] \\ & & c_{out_i}!x_{b_i}.[x_{b_i^{'}} \vdash_{snd} x_{h(b_{i+1}^{'})}] \\ & & Receiver_{i+1}(x_{h(b_{i+1}^{'})}) \\ Receiver_l(x_{h(b_l^{'})}) & \doteq & c_l?x_{b_l^{'}}.[x_{b_l^{'}} \vdash_{hash} x_{h(b_{MYl}^{'})}] \\ & & [x_{h(b_l^{'})} = x_{h(b_{MYl}^{'})}]c_{out_l}!x_{b_l^{'}}.\mathbf{0} \end{array}$$

In the initial state the receiver aims at verifying the digital signature (we assume he has previously retrieved the public key $pk(S)$ corresponding to the private key of the supposed sender). After verifying the correctness of the signature he travels to the next state $Receiver_i(x_{h(b_1^{'})})$ maintaining history of the (supposed) next hashed block $h(b_1^{'})$. The receiver has now reached the phase in which he can receive meaningful payloads b_i. Acceptance of the subsequent blocks is conditioned to the successful outcome of the equality tests between the hash he maintains as a parameter and the hash he computes from what he has presently received, respectively $x_{h(b_i^{'})}$ and $x_{h(b_{MYi}^{'})}$. The successful outcome of the equality test is here modeled imaging the receiver sends the meaningful payload contained in x_{b_i} to his application level to consume it. This is modeled by a sending action over channel c_{out_i}. He then extracts the supposed hash of the block to be received immediately later. The mechanism is repeated until the reception of l-th block. Whether the verification of the signature in the initial state or the equality tests in subsequent states do not succeed the receiver should abort. (The inference construct of Section 2 takes implicitly into account this last possibility).

3.2 Extending the model to multiple receivers

The Crypto-CCS specifications in the previous subsection do not reflect exactly a typical multicast/broadcast scenario, as it does not take into account a set of multiple, potentially unbounded receivers. Along with one sender and a set of multiple receivers we could extend the model to the treatment of multicast and broadcast applications by adding a new process MB responsible for potentially sending each block an unbounded number of times in order to simulate a one-to-many (one-to-all) sending typical of a multicast (broadcast) environment. The new process is parameterized by the block the sender is to multicast (or broadcast).

$$MB_i(x_{b_i^{'}}) \quad \doteq \quad c_i!x_{b_i^{'}}.MB_i(x_{b_i^{'}})$$

In the light of this new process, the sender's specifications of subsection 3.1 can be re-written as follows:

$$\begin{array}{rcl} Sender_0 & \doteq & [b_1^{'} \vdash_{hash} x_{h(b_1^{'})}] \\ & & [x_{h(b_1^{'})} \quad sk(S) \vdash_{sign} x_{b_0^{'}}] \\ & & (Sender_1 \mid MB_0(x_{b_0^{'}})) \\ Sender_i & \doteq & [b_{i+1}^{'} \vdash_{hash} x_{h(b_{i+1}^{'})}] \\ & & [b_i \quad x_{h(b_{i+1}^{'})} \vdash_{pair} x_{b_i^{'}}] \\ & & (Sender_{i+1} \mid MB_i(x_{b_i^{'}})) \\ Sender_l & \doteq & MB_l(b_l) \end{array}$$

The receiver specifications remain the same.

3.3 The on-line solution

When the sender does not know the entire content of the stream in advance, the protocol makes use of 1-time signature schemes, a special kind of signature scheme introduced in [11], faster to compute and verify than regular signatures. These schemes can be used to sign only one packet. The protocol formalization follows[5]:

[5]We assume that the sender's private key $sk(S)$ and the sender's 1-time secret keys $1sk_S^i$ do not occur in the set $\{b_i\}$ of meaningful payloads.

$$\begin{array}{llll} \textbf{Block } b_0' & c_0\ S \to R: & \{1pk_S^0\}_{sk(S)} & \\ \textbf{Block } b_i' & c_i\ S \to R: & b_i, 1pk_S^i, & \\ & & \{h(b_i, 1pk_S^i)\}_{1sk_S^{i-1}} & i \geq 1 \end{array}$$

Notation $1pk_S^i$ and $1sk_S^i$ indicates the *i*-th 1-time public and private key of the sender; notation $\{msg\}_{1sk_S^i}$ means: "$m$sg is signed with the private key corresponding to the i-th 1-time public key of the sender.

Bootstrapping integrity of digital streams is obtained by applying a single traditional digital signature in combination with 1-time signatures. The digital stream is divided into blocks and each block carries a 1-time public key, which is used in a 1-time signature scheme to verify the signature over the following block. Only the first block needs to be signed with a traditional signature scheme.

4 Towards compositional analysis within GNDC

In this section we present the general schema *Generalized Non Deducibility on Compositions* (GNDC for short) for the definition of security properties given in [5, 6] and some compositional proof rules for such a schema. The schema is based on the notion of non-interference. The main idea is the following: a system S is $GNDC_{\triangleleft}^{\alpha}$ *iff* for every enemy X the composition of the system with X satisfies a specification $\alpha(S)$. Basically, $\alpha(S)$ denotes the "correct" behavior of the system. $GNDC_{\triangleleft}^{\alpha}$ guarantees that the property identified by α is satisfied (with respect to $\triangleleft$ relation) even when the system is composed with any possibly hostile enemy (or malicious user).

Analyzing cryptographic protocols involves to specify the set of messages known by the enemy at the beginning of the computation. We impose some constrains on this initial knowledge. Given a system S, we call $ID(S)$ the set of closed messages that syntactically appear in S. This set contains all the messages that are initially known by S. Let $\phi \subseteq Msgs$ be the initial knowledge that we would like to give to the enemy.

For a certain enemy X, we want $ID(X)$ being consistent with ϕ. This can be obtained by simply requiring that all the messages in $ID(X)$ are deducible from ϕ.

The set $\mathcal{E}_C$ of processes that can communicate on a subset of C and have an initial knowledge bound by ϕ can be thus defined as follows:

$$\mathcal{E}_C = \{X \in \mathcal{P} \mid sort(X) \subseteq C \text{ and } ID(X) \subseteq \mathcal{D}(\phi)\}$$

where $\mathcal{P}$ is the set of all the $closed$ terms in the language and $sort(X)$ is the set of all the channels that syntactically occur in the term X. We consider as hostile processes only the ones belonging to $\mathcal{E}_C$. We define the property $GNDC_{\triangleleft}^{\alpha}$ as follows:

Definition 3 *S is $GNDC_{\triangleleft}^{\alpha}$* iff $\forall X \in \mathcal{E}_C : S\|_C X \triangleleft \alpha(S)$ *where $\triangleleft$ is a relation between processes and α is a function between processes.* ■

This definition is natural but difficult to be checked due to the universal quantification. A way to avoid universal quantification over all the admissible enemies is to show the equivalence between $GNDC$ and *Strong Nondeterministic Non-Interference* ($SNNI$, for short).

In particular, we introduce the *hiding* operator $P/^{\phi}C$ that is used to model a process in a context where there is an intruder. Let us call this intruder Top, whose knowledge is ϕ and that is willing to receive on each channel he is able to communicate over , i.e. C, each messages he can deduce from his knowledge, i.e. ϕ. Top is also able to receive each message sent over channels C, so his knowledge increases. The defining rules are the following:

$$\frac{P \xrightarrow{a} P'}{P/^{\phi}C \xrightarrow{a} P'/^{\phi}C}(a \notin C)$$

$$\frac{P \xrightarrow{c!m} P' \quad c \in C}{P/^{\phi}C \xrightarrow{\tau} P'/^{\phi \cup \{m\}}C}$$

$$\frac{P \xrightarrow{c?m} P' \quad c \in C \quad m \in \mathcal{D}(\phi)}{P/^{\phi}C \xrightarrow{\tau} P'/^{\phi}C}$$

A Crypto-CCS process P is $SNNI_C^{\phi}$ if $P\backslash C$, i.e. process P where all actions in C are forbidden, behaves like the system P where all the actions in C are *hidden* through the *hiding* operator.

Definition 4 *A process is $SNNI_C^{\phi}$ if $P\backslash C =_{trace} P/^{\phi}C$.*

Basically, the *hiding* operator embodies the so-called most powerful enemy. We are able to prove that by using this operator we can avoid using the universal quantification on checking $GNDC$, at least for properties based on trace equivalence. (For a deeper discussion see [6].)

Proposition 1 $GNDC_{=_{trace}}^{\alpha} = SNNI_C^{\phi}$ where $\alpha = P\backslash C$.

4.1 Compositional results

We illustrate some compositional proof rules for establishing if a system enjoys a $GNDC_{\triangleleft}^{\alpha}$ property and in particular $SNNI_C^{\phi}$.

The property $SNNI$ is compositional, i.e. if $P, Q \in SNNI$ then $(P \mid Q) \in SNNI$. This holds within the usual non-interference theory in a setting without cryptography modeling (e.g., see [4]). The same does not hold when considering enemies with limited knowledge, as for $SNNI_C^{\phi}$. Consider the processes:

$$\begin{array}{lll} P & = & c_1!m_1.c_2?x.[x = m_2]c_3!m_2.\mathbf{0} \\ Q & = & c_1!m_2.c_2?x.[x = m_1]c_3!m_1.\mathbf{0} \end{array}$$

Assuming $C = \{c_1, c_2\}$ and $\phi = \emptyset$, we have that $P, Q \in SNNI_C^{\phi}$. However, $P \,|\, Q \notin SNNI_C^{\phi}$. $(P \,|\, Q)\backslash C$ is equivalent to $\mathbf{0}$, while $(P \,|\, Q)/^{\phi}C$ may perform both $c_3!m_1$ and $c_3!m_2$. The intruder can indeed add to his knowledge messages m_1, m_2 received over channel c_1 and send them later over channel c_2.

To get a compositional rule for establishing that a composition of processes belongs to $SNNI_C^{\phi}$ we strengthen our requirements on the behavior of the processes.

Definition 5 *We say that a process P is stable w.r.t. ϕ, whenever if $P/^{\phi}C \Longrightarrow P'/^{\phi'}C$ then $\mathcal{D}(\phi) = \mathcal{D}(\phi')$.*

Basically, a process P is stable when an enemy with a certain knowledge ϕ does not increase significantly ϕ during the execution of P.
The following proposition holds.

Proposition 2 *Assume that $P, Q \in SNNI_C^{\phi}$ and that P, Q are stable w.r.t. ϕ. Then $(P \,|\, Q) \in SNNI_C^{\phi}$ and $P \,|\, Q$ is stable w.r.t. ϕ.*

Using the previous result, it is possible to prove that a similar compositional rule holds also for the $GNDC_{\leq_{trace}}^{\alpha}$ schema (under the assumption that the involved processes are stable).

Proposition 3 *Given the set of the intruder initial knowledge ϕ and the set of public channels C, assume $P_i \in GNDC_{\leq_{trace}}^{\alpha_i(P_i)}$, with $i = 1, 2$, and P_1, P_2 are stable w.r.t. ϕ. It follows that $(P_1 \,|\, P_2) \in GNDC_{\leq_{trace}}^{\alpha_1(P_1) \,|\, \alpha_2(P_2)}$ and $P_1 \,|\, P_2$ is stable w.r.t. ϕ.*

Compositional proof rules may be used to check the correctness of specifications with an unbounded number of equal processes. (As notation we use $\Pi_{i=1..n}P$ for the parallel composition $\overbrace{P \,|\, \ldots \,|\, P}^{n}$.)

Proposition 4 *If P is stable w.r.t. ϕ and $P \in GNDC_{\leq_{trace}}^{\alpha}$ then for all n we have:*

$$\Pi_{i=1..n}P \in GNDC_{\leq trace}^{\Pi_{1..n}\alpha}$$

5 Compositional analysis of the Gennaro-Rohatgi protocol

For checking integrity the above compositional proof rules can be successfully applied. In particular, we show how to apply the compositional analysis to a class of stream signature protocols. We start by analyzing the Gennaro-Rohatgi off-line solution and we prove the correctness of its construction by using the previous compositional rules. The goal is to exploit the compositional rules to prove the integrity of the received blocks when the presence of an intruder is considered.

We formally define integrity in the GNDC schema as the ability to accept only the message b_i by a receiver as the $i-th$ message sent by the sender. Assume that a receiver signals the acceptance of a block as a legitimate one, by issuing it on a special channel c_{out}. Thus, let α_{int} be $Spec_1$ where $Spec_i = c_{out_i}!b_i.Spec_{i+1}$ with $1 \leq i \leq l-1$, and $Spec_l = c_{out_l}!b_l.\mathbf{0}$.

Definition 6 *A system P consisting of a sender of a stream of messages b_i and a receiver, enjoys the integrity property whenever $P \in GNDC_{\leq_{trace}}^{\alpha_{int}}$.*

Basically, it means that the receiver accepts exactly the blocks b_i in the correct order even in presence of an intruder.

We are able to prove that $Sender_0$ and $Receiver_0$ (Subsection 3.1) are stable w.r.t. the following initial knowledge ϕ:

$$\phi = \{pk(S), b_0'\} \cup \{b_i', b_i, h(b_i') \mid i = 1, \ldots, l-1\} \cup \{b_l', b_l\}$$

Actually, we include in the initial knowledge ϕ the messages an intruder would be able to add to his knowledge by eavesdropping on a run of the protocol. This implies we arrange an intruder to have the most powerful means to act since the beginning of the computation. If the protocol satisfies the integrity property in this very hostile environment then it means that it will satisfy this property in a less powerful one (this may be formally justified).

The following set of channels is considered as channels over which an intruder is able to communicate: $C = \{c_i \mid i = 0, \ldots, l\}$.

Lemma 1 *$Sender_0$ and $Receiver_0$ are stable w.r.t. ϕ.*

We are able to prove that $Sender_0$ enjoys $GNDC_{\leq_{trace}}^{\mathbf{0}}$ and $Receiver_0$ enjoys $GNDC_{\leq_{trace}}^{\alpha_{int}}$, that is to say for all X with $ID(X) \subseteq \mathcal{D}(\phi_X)$ we have $(Sender_0 \,|\, X)\backslash C \leq_{trace} \mathbf{0}$ and $(Receiver_0 \,|\, X)\backslash C \leq_{trace} \alpha_{int}$. This may be done by finding a suitable weak simulation relation between $(Sender_0 \,|\, X) \setminus C$ and $\mathbf{0}$ and between $(Receiver_0 \,|\, X) \setminus C$ and $Spec_1$, respectively.

Lemma 2 $Sender_0 \in GNDC_{\leq_{trace}}^{\mathbf{0}}$.

Lemma 3 $Receiver_0 \in GNDC_{\leq_{trace}}^{\alpha_{int}}$.

Proposition 5 $Sender_0 \,|\, Receiver_0 \in GNDC_{\leq trace}^{\alpha_{int}}$.

5.1 Compositional analysis of the multiple receivers version

Compositional reasoning is powerful. The correctness of the multiple receivers version in Subsection 3.2 can be proved by only checking that the sender is stable w.r.t. ϕ. The rest follows by re-using the previous results.

Lemma 4 *$Sender_0$ is stable w.r.t. ϕ and $Sender_0 \in GNDC^{\mathbf{0}}_{\leq trace}$.*

Note that in a multi-receiver environment with one sender, a protocol guarantees integrity whenever each receiver accepts only the stream of messages that sender wishes to deliver. In our case, the specification for n receivers is simply the parallel composition of α_{int} n-times. The following proposition holds.

Proposition 6 *The Gennaro-Rohatgi off-line solution enjoys integrity for whatever number of receivers.*

6 Further extensions

The compositional analysis can be successfully applied to formally verify integrity properties in a multicast/broadcast environment. (We remind that integrity means, from our point of view, assurance that multicast data are not modified *en-route*.) In particular, we have showed the efficiency of the analysis applied to the scheme of [7] in its off-line solution variant. We are able to formally check the system with an unbounded number of receivers.

We applied the compositional rules to the Gennaro-Rohatgi on-line solution and it follows that it enjoys integrity for whatever number of receivers. With reference to a formal Crypto-CCS specifications of the scheme in Subsection 3.3, (not reported in the paper), we tested the stability condition and the fulfillment of $GNDC$ of a sender and an unbounded number of receivers w.r.t. the intruder initial knowledge: $\phi_X = \{pk(S), b'_0, 1pk^0_S\} \cup \{b'_i, b_i, 1pk^i_S) \mid i \geq 1\}$. The proof proceeds essentially as in the off-line case, apart considering a formal reasoning concerning the verifications of 1-time signatures instead of comparing the hashes.

One of the problems in the approach of [7] is that if a block is missing, the authentication chain is broken and the integrity of subsequent packets can not be verified. Efficient constructions to solve the problem of authenticating and proving the integrity of streamed data over channels with packet loss have been recently proposed, [9, 13, 15]. In [15] the authors propose EMSS, whose basic scheme is the following: block b_i includes a hash $h(b_{i-1})$ of the previous block b_{i-1}. Further, a signature block is included at the end of the stream, which contains the hash of the final block along with a standard digital signature. To solve (in part) the problem of packets loss each block contains multiple hashes of previous blocks and the signature block signs hashes of multiple blocks. [9] proposes a variant of EMSS, in which hashes of previous and subsequent blocks are included in each packet. The distribution of the hashes and the signature blocks is deterministically chosen in [9], while in [15] the authors also propose a random distribution.

From some preliminary investigations we argue that our approach may be successfully extended to correctly verify the property of integrity in [9] and in the deterministic schemes suggested in [15]. As future work, we plan to i) complete the formal analysis of EMSS when a random distribution of hashes and signatures packets is considered; ii) apply our compositional proof rules to (erasure codes)-based solutions like the one proposed in [13]; iii) enhance the compositional analysis technique in order to check other classes of protocols.

References

[1] M. Abadi and A. Gordon. A Calculus for Cryptographic Protocols: the Spi Calculus. *Information and Computation*, 148(1), 1999.

[2] M. Archer. Proving Correctness of the Basic TESLA Multicast Stream Authentication Protocol with TAME. In *Proc. of WITS'02*, Portland, USA, January 2002.

[3] P. Broadfoot and G. Lowe. Analysing a Stream Authentication Protocol using Model Checking. In *Proc. of ESORICS'02*, volume LNCS 2502, pages 146–161. Springer, October 2002.

[4] R. Focardi and R. Gorrieri. A classification of security properties. *Journal of Computer Security*, 3(1):5–33, 1995.

[5] R. Focardi, R. Gorrieri, and F. Martinelli. Non Interference for the Analysis of Cryptographic Protocols. In *Proc. of ICALP'00*, volume LNCS 1853, pages 354–372. Springer, 2000.

[6] R. Focardi and F. Martinelli. A uniform approach for the definition of security properties. In *Proc. of FM'99*, volume LNCS 1708, pages 794–813. Springer, 1999.

[7] R. Gennaro and P. Rohatgi. How to Sign Digital Streams. *Information and Computation*, 165(1):100–116, 2001.

[8] J. A. Goguen and J. Meseguer. Security Policies and Security Models. In *Proc. of IEEE S&P'82*, pages 11–20, 1982.

[9] P. Golle and N. Modadugu. Authenticating Streamed Data in the Presence of Random Packet Loss. In *Proc. of NDSS'01*. The Internet Society, San Diego, CA, February 2001.

[10] R. Gorrieri, E. Locatelli, and F. Martinelli. A Simple Language for Real-time Cryptographic Protocol Analysis. In *Proc. of ESOP'03*, April 2003.

[11] L. Lamport. Constructing Digital Signatures from a One-Way Function. Technical Report CSL 98, SRI Intl, 1979.

[12] R. Milner. *Communication and Concurrency*. Prentice Hall, 1989.

[13] J. M. Park, E. K. P. Chong, and H. J. Siegel. Efficient Multicast Packet Authentication using Signature Amortization. In *Proc. of IEEE S&P'02*, pages 227–240. IEEE, Berkeley, CA, May 2002.

[14] A. Perrig, R. Canetti, D. X. Song, and D. Tygar. Efficient and Secure Source Authentication for Multicast. In *Proc. of NDSS'01*. The Internet Society, San Diego, CA, February 2001.

[15] A. Perrig, R. Canetti, J. D. Tygar, and D. X. Song. Efficient Authentication and Signing of Multicast Streams over Lossy Channels. In *Proc. of IEEE S&P'00*, pages 56–73, 2000.

[16] R. Rivest, A. Shamir, and L. Adleman. A Method for Obtaining Digital Signatures and Public Key Cryptosystems. *Comm. of the ACM*, 21(2):120–126, 1978.