

What is the best way to prove a cryptographic protocol correct? (- Position Paper -)

Sreekanth Malladi
College of Business and Information Systems
Dakota State University
Madison, SD - 57042
Sreekanth.Malladi@dsu.edu

Gurdeep S. Hura
Dept of Math and Computer Science
University of Maryland-Eastern Shore
Princess Anne, MD - 21853
gshura@umes.edu

Abstract

In this paper, we identify that protocol verification using invariants have significant limitations such as inapplicability to some protocols, non-standard attacker inferences and non-free term algebras. We argue that constraint solving for bounded process analysis can be used in conjunction with decidability of context-explicit protocols as a verification tool and can overcome those limitations. However, this is possible only when new decidability results are obtained for protocol security, especially in presence of non-standard inferences and non-free term algebras. Those results are in progress and will be soon submitted for publication.

Keywords – *Cryptographic protocols, Formal methods, Theorem proving, Invariants, Constraint solving, Decidability.*

1 Introduction

Protocol verification with unbounded number of participants were carried out in the last decade, largely through the use of automated theorem provers (e.g. Isabelle [1], PVS [2], Interrogator [3], TAPS [4]) and manual proof techniques (e.g. Rank functions [5] and Strand spaces [6]). The central concept is similar in these techniques:

1. Create a term set which is to be unchanged by the attacker in every step of the protocol execu-

tion. E.g. Strand space ideals, which are an infinite set of terms closed under concatenation of arbitrary terms with a (purportedly) secret term and encryption with a set of terms that an attacker wouldn't have access to (such as the agents' private keys);

2. Prove that uttering a term in the set is impossible without possessing certain secret and private keys before the execution of the protocol;
3. Prove that every step in the protocol preserves this condition (called an *invariant*) i.e. an attacker never has access to a term in the set (not protected by encryption).

Another way is to create a set of terms using attacker actions but one into which honest agents never utter secrets. This idea of using invariants across many techniques has been nicely described in [7]. Although this technique has been used to prove a number of protocols correct, it has some limitations. In this paper, we identify those limitations and advocate another approach that overcomes those limitations and thus should be considered a better alternative to offer protocol security proofs.

The paper is organized as follows: In Section 2 we explain the limitations of invariant based techniques in detail which include identification of some new attacks on tagging schemes. In Section 3 we explain how the constraint solving technique by Millen-Shmatikov [8] can be used to prove protocol security and argue that it overcomes the limitations of invariant based techniques. In Section 4 we explain some limitations

of constraint based technique and its similarity with strand spaces technique. We sum up with a Conclusion.

2 Limitations of invariant-based methods

2.1 Limited applicability

It has been identified in literature that some invariant techniques (perhaps not all) are applicable to only some protocols. Others cannot be verified using the techniques. For example, arguably the most commonly analyzed Needham-Schroeder-Lowe (NSL) protocol [9] below itself cannot be verified using the strand space ideals technique:

$$\begin{aligned} \text{Msg 1. } A \rightarrow B &: [A, N_A]_{pk(B)}^{\rightarrow} \\ \text{Msg 2. } B \rightarrow A &: [N_B, N_A, B]_{pk(A)}^{\rightarrow} \\ \text{Msg 3. } A \rightarrow B &: [N_B]_{pk(B)}^{\rightarrow} \end{aligned}$$

(N_A, N_B are nonces; A, B are agent identities; $[t]_k^{\rightarrow}$ denotes t encrypted with k using an asymmetric key algorithm; $pk(a)$ is the public-key of a).

In the seminal strand spaces paper [6], Guttman et al. explain that this protocol cannot be verified in the standard way that some protocols such as Otway-Rees were verified — by considering a single Ideal containing secret terms closed under concatenation with arbitrary terms and encryption with secret keys. They also explain that in the case of the NSL protocol, reasoning has to be made into the difference of ideals $I_K[N_B]$ and $I_K[[N_A, N_B, B]_{pk(A)}]^1$, unlike the standard approach.

Note that Guttman et al. do prove the NSL protocol correct in the same paper using standard Dolev-Yao penetrator actions and without using Ideals.

2.2 Limited to standard attacker inferences

Traditionally, invariant techniques always operated under standard attacker inferences wherein a perfect encryption assumption was adopted. i.e. the attacker was assumed to be incapable of breaking encryptions by exploiting cryptographic weaknesses. There are some non-standard inferences which were shown to

¹Notation: $I_K[t]$ represents the set of terms where each term is encrypted with a key in K and contains t as a subterm

be difficult, if not impossible, to be incorporated into invariant-based techniques.

For example, the low-exponent RSA weakness pointed by Coppersmith et al. [10] allows the inference rule:

$$\{a, b, c, d, [a, x, b]_k^{\rightarrow}, [c, x, d]_k^{\rightarrow}\} \vdash x$$

where $a \neq b \wedge c \neq d$.

Here k is a RSA public-key. \vdash denotes the relation “infer”.

Heather and Schneider explain in [5] that this rule presents a problem with not only the data independence framework by Roscoe and Broadfoot [11], but also the Rank functions method and the Strand space ideals. For example, they explain that in order to extend the strand space technique, a strand,

$$\text{LowExp } \langle -[a, x, b]_k^{\rightarrow}, -[c, x, d]_k^{\rightarrow}, -a, -b, -c, -d, +x \rangle$$

has to be added to the penetrator strands. Now, constructing an ideal with x and k^{-1} would mean both $[a, x, b]_k^{\rightarrow}$ and $[c, x, d]_k^{\rightarrow}$ will be contained in the ideal and this set will first have to be shown as honest; it has to be subsequently shown that *only* honest agents can provide entry points to the ideal (i.e. utter a term in the ideal) at every step of the protocol. However, the rule presents a problem by providing the attacker as well, with an opportunity to utter a term in the ideal (by releasing $+x$), contradicting the definition of honesty and spoiling the main supposition that only honest agents can utter terms in the ideal.

With this difficulty, it is hard to prove protocols such as the modified version of the NSL protocol below, correct:

$$\begin{aligned} \text{Msg 1. } A \rightarrow B &: [A, N_A]_{pk(B)}^{\rightarrow} \\ \text{Msg 2. } B \rightarrow A &: [N_A, N_B, B]_{pk(A)}^{\rightarrow}, [B, N_B, N_A]_{pk(A)}^{\rightarrow} \\ \text{Msg 3. } A \rightarrow B &: [N_B]_{pk(B)}^{\rightarrow} \end{aligned}$$

Notice that this protocol is really not much different from the original NSL protocol except that terms inside Message 2 have been slightly rearranged and re-sent. Now it is easy to prove that this protocol is correct under standard attacker inferences with almost

similar reasoning provided for the original NSL protocol (e.g. [9, 6]). It can also be seen that in presence of the RSA weakness, the protocol is no longer secure:

$$\begin{aligned} \text{Msg 1. } & i(a) \rightarrow b : [a, n_i]_{pk(b)}^{\rightarrow} \\ \text{Msg 2. } & b \rightarrow i(a) : [n_i, n_b, b]_{pk(a)}^{\rightarrow}, [b, n_b, n_i]_{pk(a)}^{\rightarrow} \end{aligned}$$

$i(a)$ denotes i spoofing as a . The attacker can now extract n_b from Message 2 since it satisfies the conditions of the RSA weakness:

$$\{[n_i, n_b, b]_{pk(a)}^{\rightarrow}, [b, n_b, n_i]_{pk(a)}^{\rightarrow}, n_i, b, b, n_i\} \vdash n_b$$

with $n_i \neq b$ and $b \neq n_i$. The final step, Message 3 can then be completed:

$$\text{Msg 3. } i(a) \rightarrow b : [n_b]_{pk(b)}^{\rightarrow}$$

However, since invariant techniques do not allow us to reason without breaking the ground rules of the framework, we cannot positively ascertain using those techniques that the protocol is correct when it is correct and incorrect when it is not, if the RSA weakness were to be present. In fact, we can never be sure about the protocol's security, if the perfect encryption assumption was to be broken, allowing for tricky non-standard attacker rules such as the RSA inference rule above.

2.3 Limited to Free term algebras

Almost all the invariant based techniques make a strong typing assumption. i.e. messages are always well-typed and that there will be no type-flaws in any protocol execution. Such an unrealistic assumption was justified by Heather et al. who proved that type-tagging messages eliminates all type-flaw attacks [12]. However, type-tagging prevents type-flaw attacks only under a free term algebra with no algebraic properties for message term operators. For example, consider the following adapted NSL protocol by Chevalier et al. [13] that uses the Exclusive-OR operator (\oplus). We further modify it with Heather et al.'s tagging:

$$\begin{aligned} \text{Msg 1. } & A \rightarrow B : [[\text{agent}, A], [\text{nonce}, N_A]]_{pk(B)}^{\rightarrow} \\ \text{Msg 2. } & B \rightarrow A : [[\text{nonce} \oplus \text{agent}, N_A \oplus B], \\ & \quad [\text{nonce}, N_B]]_{pk(A)}^{\rightarrow} \\ \text{Msg 3. } & A \rightarrow B : [\text{nonce}, N_B]_{pk(B)}^{\rightarrow} \end{aligned}$$

$\text{nonce} \oplus \text{agent}$ is a tag that we introduce in this paper to denote that the type of the term that it is associated with is an XOR of a nonce and an agent. This is in conformance with Heather et al.'s tagging since they denoted functions and operators on messages similarly, although they only advocated that the tag for a function application only represent the function, not necessarily its arguments.

A type-flaw attack is possible on this protocol, despite the tagging:

$$\begin{aligned} \text{Msg } \alpha.1. & a \rightarrow i : [[\text{agent}, a], [\text{nonce}, n_a]]_{pk(i)}^{\rightarrow} \\ \text{Msg } \beta.1. & i(a) \rightarrow b : [[\text{agent}, a], [\text{nonce}, n_a \oplus b \oplus i]]_{pk(b)}^{\rightarrow} \\ \text{Msg } \beta.2. & b \rightarrow i(a) : [[\text{nonce} \oplus \text{agent}, n_a \oplus b \oplus i \oplus b], \\ & \quad [\text{nonce}, n_b]]_{pk(a)}^{\rightarrow} \\ \text{Msg } \alpha.2. & i \rightarrow a : [[\text{nonce} \oplus \text{agent}, n_a \oplus i], [\text{nonce}, n_b]]_{pk(a)}^{\rightarrow} \\ \text{Msg } \alpha.3. & a \rightarrow i : [\text{nonce}, n_b]_{pk(b)}^{\rightarrow} \\ \text{Msg } \beta.3. & i(a) \rightarrow b : [\text{nonce}, n_b]_{pk(b)}^{\rightarrow} \end{aligned}$$

Observe that messages $\beta.2$ and $\alpha.2$ are really the same except that agent identity ' b ' present twice in Message $\alpha.2$ can be canceled by exploiting the cancellation property of XOR. Notice also the type-flaw in $\text{Msg } \alpha.1$ by $i(a)$ sending $\text{nonce} \oplus \text{agent} \oplus \text{agent}$ claiming it as a nonce. This attack cannot exist without this type-flaw. Hence, it is strictly a type-flaw attack.

To date, little to no work has been reported on relaxing the free term algebra model and allowing for operators such as XOR by invariant based techniques. Even if they are extended, a challenge they are going to face is type-flaw attacks, which cannot be ignored in a non-free term algebra setting. Indeed, there is not a tagging scheme proven to prevent all type-flaw attacks in these extended models (at least so far).

3 Overcoming the limitations

Having been exposed to the limitations of invariant based techniques, it is natural to wonder if there are ways to overcome the limitations. Specifically, we need a verification technique or tool that satisfies the following requirements:

Requirement 1. *General enough to accept any protocol and not necessarily restrict itself to certain classes of protocols;*

Requirement 2. Operates in a non-basic environment that allows non-standard attacker inferences; and

Requirement 3. Takes as input a protocol that uses operators with algebraic properties such as XOR.

Of all the tools described in protocol analysis literature thus far, only one tool seems to be capable of satisfying all these requirements. It is the *Constraint Solver* tool based on the constraint solving technique introduced by Millen and Shmatikov [8]. The tool’s technique is well-known but a simple description is below:

1. This tool models protocol roles as strands and calls them collectively as semi-bundles [14].
2. All the messages (both transmissions and receptions) are mixed into finite number of interleavings.
3. Each interleaving is converted into a constraint sequence (C) where every constraint represents that a message m to be received by an agent must be constructed by an attacker using all transmitted messages prior to it (T), denoted as $m : T$.
4. This sequence is solved by a reduction procedure **P** that applies symbolic reduction rules to each constraint in order.

For more details on the technique, the reader is referred to [8]. Let us now look at how the constraint solver can be used to satisfy our requirements on a tool to prove protocol security.

Satisfying requirement 1. The constraint solver can be used to analyze or verify any protocol (whose messages fit into the term algebra considered by the tool) and does not restrict itself to certain classes of protocols only, satisfying our Requirement (1) above.

This tool was invented originally to be applied on scenarios with bounded number of participants and processes played by them. However, we can use the decidability result of Ramanujam et al. [15] to conclude that analyzing protocols with one agent per role of the protocol is sufficient to conclude the protocol’s security in unbounded scenarios as well.

Ramanujam et al.’s result applies to context-explicit protocols wherein encrypted subterms are

non-unifiable. This can be easily achieved by the use of distinct component numbers inside each encryption. For example, the NSL protocol modified with such numbers appears as below:

$$\begin{aligned} \text{Msg 1. } A \rightarrow B &: [1, A, N_A]_{pk(B)}^{\rightarrow} \\ \text{Msg 2. } B \rightarrow A &: [2, N_B, N_A, B]_{pk(A)}^{\rightarrow} \\ \text{Msg 3. } A \rightarrow B &: [3, N_B]_{pk(B)}^{\rightarrow} \end{aligned}$$

It is noteworthy to mention that Ramanujam et al.’s decidability result only considers atomic keys while the constraint solver allows for composed (or constructed) keys as well. An extension of Ramanujam et al.’s result to composed keys is possible and efforts are underway to formally prove it [16].

Satisfying requirement 2. The constraint solver can also successfully analyze or verify protocols in presence of the RSA LowExp inference rule. An extension for this rule to the solver was described by Malladi and Rosenberg [17] through the addition of a simple reduction rule (*LowExp*):

$$\frac{C_{<}, m : T \cup [a, [x, b]]_k^{\rightarrow} \cup [c, [x, d]]_k^{\rightarrow}, C_{>}; \sigma}{C_{<}, a : T, b : T, c : T, d : T, m : T \cup x, C_{>}; \sigma}$$

where $a \neq c \vee b \neq d$.

($C_{<}, C_{>}$ are the constraints before and after the active constraint under consideration; σ is the attacker substitution of ground terms to variables). Protocols were successfully analyzed assuming the presence of this rule.

It is also possible to prove protocol security with this extension using the decidability result whose proof is also underway [16]. According to this result, context-explicit protocol security is always decidable as long as attacker inferences produce only subterms of the initial attacker knowledge. This is not surprising since the requirement for a protocol to qualify as “context-explicit” is that encrypted subterms should be non-unifiable. Obviously, a subterm producing inference will retain the validity of the requirement throughout the protocol execution. Using the decidability result we can again conclude protocol security after analyzing a protocol with one process per role thereby satisfying Requirement (2).

Satisfying requirement 3. The third requirement concerns the XOR operator. The constraint solving technique was extended with this operator [18]. As in satisfying the previous two requirements, we want to depend on the decidability of context-explicit protocols, but there is a problem. Ramanujam et al.’s results assumed a free-term algebra. Further, they some times proved decidability by showing that component numbering can be used to restrict analysis to well-typed runs only [19]. In other words, that component numbering prevents type-flaw attacks and thus ensures decidability. But it turns out that component numbering is also ineffective against type-flaw attacks. For example, consider Chevalier at al.’s adapted NSL protocol again, modified by including component numbers inside all encryptions:

- Msg 1. $A \rightarrow B : [1, N_A, A]_{pk(B)}^{\rightarrow}$
- Msg 2. $B \rightarrow A : [2, N_B, N_A \oplus B]_{pk(A)}^{\rightarrow}$
- Msg 3. $A \rightarrow B : [3, N_B]_{pk(B)}^{\rightarrow}$

A type-flaw attack is possible despite the component numbers:

- Msg α .1. $a \rightarrow i : [1, n_a, a]_{pk(i)}$
- Msg β .1. $i(a) \rightarrow b : [1, n_a \oplus b \oplus i, a]_{pk(b)}$
- Msg β .2. $b \rightarrow i(a) : [2, n_a \oplus b \oplus i \oplus b, n_b]_{pk(a)}$
- Msg α .2. $i \rightarrow a : [2, n_a \oplus i, n_b]_{pk(a)}$
- Msg α .3. $a \rightarrow i : [3, n_b]_{pk(i)}$
- Msg β .3. $i(a) \rightarrow b : [3, n_b]_{pk(b)}$

Although component numbering cannot prevent type-flaw attacks in presence of XOR, a decidability result is still possible. Basically, decidability has to be proven without consideration of type-flaw attacks on a small-system with one process per role of the protocol in question. We can then analyze a protocol using the constraint solver extended with XOR on a small-system. Absence of attacks then serves as a proof of the protocol’s security. Such a result is also under pursuit much like the results to satisfy the previous two requirements [16].

4 More on constraint solving

4.1 Limitations of proofs with constraint solving.

It is worthy to mention that not all non-standard inferences guarantee decidability of context-explicit pro-

ocol security. For example, the inference,

$$[m, n]_k \vdash [m]_k$$

would hold when using Cipher Block Chaining (CBC) for block ciphering. This rule produces a non sub-term of the initial attacker knowledge and thereby potentially violates the non-unifiability requirement for context-explicit protocols. It is an open question as to under what set of attacker inference rules can decidability be guaranteed or proved undecidable.

It is also worthy to mention that regardless of decidability of protocol security under the CBC rule, the constraint solving technique has its own difficulties with the rule. As explained in [17], the procedure is not guaranteed to terminate because the CBC rule potentially introduces terms such as $[A]_k$ where A is a variable. This results in the unification with a pair $[X, Y]$ producing $[X]_k$ and thus more variables. This blocks termination. It is also an open question if other techniques (including invariant based and model checkers) that could use decidability results suffer from this termination problem.

4.2 Similarity of strand space analysis and constraint solving.

A final note on constraint solving is to explain that the technique is quite similar to the proof style adopted by the seminal strand spaces work [6] to prove the NSL protocol correct (without the use of invariants/ideals). For example, to prove the secrecy of the nonce ‘ n_b ’, the authors in [6] prove Proposition 5.10, namely that no bundle can have a node with n_b sent in plain. The constraint solving technique would also proceed along similar lines: as per tradition, to prove secrecy, we would add a test node which receives ‘ n_b ’ to the initial semi-bundle. We would then prove that none of the paths in the reduction procedure \mathbf{P} leads to a solution (which is effectively the same as trying to find out if there is a combination of penetrator strands that can complete the semi-bundle to a bundle).

Both techniques are the same in terms of attacker capabilities (as Millen-Shmatikov point out in [8]). For unbounded scenario security, the techniques depend on two different results in order to be content with considering only one process per role of the

protocol. While the strand spaces depends on type-tagging to prevent type-flaw attacks [12] (indirectly assuming that parallel-run attacks are always type-flaw attacks which was subsequently proven in [19]), the constraint solving will have to depend on the decidability of context-explicit protocols [15].

The strand spaces method will be equally effective in proving security in the presence of the RSA weakness. Indeed, the corresponding strand given in Section 1 for this weakness could be added to the penetrator strands and reasoning could proceed in much the same fashion as the proofs in [6] proceeded for the NSL protocol (without using ideals). Whether strand space method has the termination problem with the CBC weakness and whether they are as capable as constraint solving in adapting to various non-free term algebras such as **Products**, **Inverses**, and **Diffie-Hellman exponentiation** (apart from **XOR**) are yet other open questions.

5 Conclusion

The main intention of the paper is to give an accurate picture of the current state of protocol security proof techniques to protocol designers who might want to verify their designed protocols in a deterministic and assured way, hopefully using automated tools. These are important especially in the wake of protocol security being treated as “solved”, at least in basic models, thanks to decidability results [15, 20]. However, currently protocol designers have little direction on choosing the best approach to conclude the security of their protocols.

The weaknesses in invariant based verification techniques have been pointed out and how constraint solving may be used to overcome those limitations has been explained. The advantage that constraint solving is readily and freely available as an automated tool is well-known. We reiterate that the claims are valid only in the context of new decidability results for protocols with composed keys, non-standard inferences and **XOR** operator [16].

The advantages of constraint solving have been described. Yet, some questions remain:

- Do invariant based techniques (or any other techniques) have advantages over constraint solving?

- Can they prove correctness in contexts where constraint solving cannot? This appears unlikely, but can anyone prove it otherwise?

Acknowledgements. This work was supported in part by the state government of South Dakota under the 2007 Individual Research Seed Grant Program.

References

- [1] Paulson, L.C.: The inductive approach to verifying cryptographic protocols. *Journal of Computer Security* **6** (1998) 85128
- [2] Dutertre, B., Schneider, S.: Using a PVS embedding of CSP to verify authentication protocols. In Gunter, E.L., Felty, A., eds.: *Theorem Proving in Higher Order Logics: 10th International Conference. TPHOLs’97*. Number 1275 in *Lecture Notes in Computer Science*, Murray Hill, NJ, Springer-Verlag (1997) 121–136
- [3] Millen, J.: On the Freedom of Decryption. *Information Processing Letters* **86(6)** (2003) 329–333
- [4] Cohen, E.: First-order verification of cryptographic protocols. *Journal of Computer Security* **11(2)** (2003) 189–216
- [5] Heather, J., Schneider, S.: Equal to the task? In: *Proc. 7th European Symposium on Research in Computer Security (ESORICS)*. (2002) 162–177
- [6] Thayer, F.J., Herzog, J.C., Guttman, J.D.: Strand spaces: Why is a security protocol correct? In: *Proc. IEEE Symposium on Research in Security and Privacy*, IEEE Computer Society Press (1998) 160–171
- [7] Meadows, C.: Invariant generation techniques in cryptographic protocol analysis. In: *Proceedings of 13th Computer Security Foundations Workshop*, IEEE (2000) 159–167
- [8] Millen, J., Shmatikov, V.: Constraint solving for bounded-process cryptographic protocol analysis. In: *Proc. ACM Conference on Computer and Communication Security*, ACM press (2001) 166–175

- [9] Lowe, G.: Breaking and fixing the Needham-Schroeder public-key protocol using FDR. In: Proceedings of TACAS. Volume 1055., Springer-Verlag (1996) 147–166 Also in Software Concepts and Tools, 17:93-102, 1996.
- [10] Coppersmith, D., Franklin, M., Patarin, J., Reiter, M.: Low-exponent RSA with related messages. Lecture notes in computer science **1070** (1996)
- [11] Roscoe, A.W., Broadfoot, P.J.: Proving security protocols with model checkers by data independence techniques. Journal of Computer Security **7**(2-3) (1999) 147–190
- [12] Heather, J., Lowe, G., Schneider, S.: How to prevent type flaw attacks on security protocols. In: Proc. 13th Computer Security Foundations Workshop, IEEE Computer Society Press (2000) 255–268
- [13] Chevalier, Y., Küsters, R., Rusinowitch, M., Turuani, M.: An NP decision procedure for protocol insecurity with XOR. In: Proc. 18th Annual IEEE Symposium on Logic in Computer Science (LICS'03), IEEE Computer Society Press (2003) 261–270
- [14] Song, D.X.: Athena: a new efficient automatic checker for security protocol analysis. In: Proc. 12th IEEE Computer Security Foundations Workshop, IEEE Computer Society Press (1999) 192–22
- [15] Ramanujam, R., Suresh, S.: Decidability of context-explicit security protocols. Journal of Computer Security **13** (2005) 135–165
- [16] Malladi, S.: Decidability of real-world context-explicit security protocols (2007) Work in progress.
- [17] Malladi, S., Rosenberg, S.: Extending constraint solving for cryptographic protocol analysis with non-standard attacker inference rules. In: IASTED International Conference on Communication, Network and Information Security (CNIS 2005). (2005)
- [18] Chevalier, Y.: A simple constraint solving procedure for protocols with exclusive-or (2004) available at <http://www.lsv.ens-cachan.fr/unif/past/unif04/program.html>.
- [19] Ramanujam, R., Suresh, S.: Tagging makes secrecy decidable for unbounded nonces as well. In: 23rd FST&TCS, Lecture Notes in Computer Science. Volume 2914. (2003) 323–374
- [20] Guttman, J., Thayer, F.J.: The sizes of skeletons: security goals are decidable. MITRE Technical report 05B09 (2006) Available at http://www.ccs.neu.edu/home/guttman/sizes_of_skeletons.pdf.