Available online at www.sciencedirect.com

SCIENCE DIRECT®

Electronic Notes in Theoretical Computer Science 154 (2006) 45–57

Electronic Notes in Theoretical Computer Science

www.elsevier.com/locate/entcs

Static Equivalence *is* Harder than Knowledge

Johannes Borgström[1],[2]

School of Computer and Communication Sciences, EPFL, Switzerland

Abstract

There are two main ways of defining secrecy of cryptographic protocols. The first version checks if the adversary can learn the value of a secret parameter. In the second version, one checks if the adversary can notice any difference between protocol runs with different values of the secret parameter.
We give a new proof that when considering more complex equational theories than partially invertible functions, these two kinds of secrecy are not equally difficult to verify. More precisely, we identify a message language equipped with a convergent rewrite system such that after a completed protocol run, the first problem mentioned above (adversary *knowledge*) is decidable but the second problem (*static equivalence*) is not. The proof is by reduction of the ambiguity problem for context-free grammars.

Keywords: Security protocol analysis, Term rewriting, Decidability.

1 Introduction

There are two main ways of specifying secrecy for a cryptographic protocol.

(1) One common approach is to see if the attacker can deduce the value of a secret parameter of the protocol, after some interaction with the protocol participants. This *disclosure*-based approach is taken in, e.g., [15,17,13].

(2) The other approach is to check whether the attacker can notice any difference between protocol runs with different values of the secret parameter. This *indistinguishability*-based approach fits naturally into the process calculus framework [5,8], is a standard notion of secrecy of cryptographic primitives [12], and is thus often used for protocol analysis in the probabilistic polynomial-time tradition [16]. This approach can also be used for other properties than secrecy, by comparing an implementation of the protocol with an executable specification.

[1] Email: `Johannes.Borgstroem@EPFL.ch`

[2] Supported by the Swiss National Science Foundation, grant No. 21-65180.01.

doi:10.1016/j.entcs.2006.05.006

Independently of the particular security properties to be verified, the formal cryptography tradition [11] is moving towards a more complete treatment of algebraic properties of cryptographic primitives [4] as well as a more fine-grained treatment of "compound primitives" such as block encryption algorithms used in electronic code book or cipher block chaining mode, or message authentication codes [14]. However, algorithms treating such more complex message algebras are often defined ad-hoc [9] and/or without termination guarantees (e.g., naive additions to ProVerif [6]). Recent work [1,3] aims at finding a sufficiently large class of message algebras, where the relevant properties still are decidable.

In this paper, we prove that there exist message algebras in which after a protocol run, disclosure is decidable but indistinguishability is not. The proof is by reducing the ambiguity problem for context-free grammars to an indistinguishability problem. Previously, a proof sketch for this separation result, based on another undecidable problem relating two pairs of Turing machines, appeared in [1,2]. The present paper is, to the knowledge of the author, the first published instance of a full proof.

2 Formal Cryptography

The basic idea behind formal cryptography is to abstract from the actual encryption algorithms used, and instead work with some suitable message algebra. The reason for this is that cryptographic primitives are often in themselves fairly complex algorithms, and the guarantees that they provide are usually based on probabilities and computation time. Taken together, this makes for a complicated model for the verification.

Formal cryptography, on the other hand, works with algebraic relationships between cryptographic primitives. Implicit in this approach is that the only possible operations on messages are the ones defined by the algebra. Thus, formal cryptography is the study of protocols under assumptions of perfect cryptography.

2.1 Message Algebras

Definition 2.1 We assume countably infinite sets of names $n \in \mathcal{N}$, variables $x \in \mathcal{V}$ and function symbols $\mathtt{f} \in \mathcal{F}$, and a finite signature $\Sigma : \mathcal{F} \rightharpoonup \mathbb{N}$ taking function symbols to their arity (which may be 0). The set of terms $\mathcal{T}_\Sigma$ is then defined by $t, u ::= \ n \mid x \mid \mathtt{f}(t_1, \ldots, t_n)$ where $\Sigma(\mathtt{f}) = n$. Let $|t|_u$ be the number of occurrences of u in t. We let $\mathrm{n}(t)$ be the names and $\mathrm{v}(t)$ be the variables of a term t. The concrete terms $\mathcal{T}^c_\Sigma$ are those that do not contain any variables.

In algebras for cryptography, message equality is typically induced by some rewrite system. In the case of symmetric cryptography, this may be as simple as the single rule $\mathtt{dec}(\mathtt{enc}(x, k), k) \to x$, stating that a message x encrypted ($\mathtt{enc}$) under the key k can be decrypted ($\mathtt{dec}$) using the same key.

In order to more accurately model the behavior of particular implementations of cryptographic primitives, one can add to and modify this rule [10]. One drawback with such refinements is that the rewrite system might no longer be convergent,