Logical Formalisation and Analysis of the Mifare Classic Card in PVS

Bart Jacobs and Ronny Wichers Schreur

Institute for Computing and Information Sciences, Radboud University Nijmegen
Heijendaalseweg 135, 6525AJ Nijmegen, The Netherlands
{bart,ronny}@cs.ru.nl

Abstract. The way that Mifare Classic smart cards work has been uncovered recently [6,8] and several vulnerabilities and exploits have emerged. This paper gives a precise logical formalisation of the essentials of the Mifare Classic card, in the language of a theorem prover (PVS). The formalisation covers the LFSR, the filter function and (parts of) the authentication protocol, thus serving as precise documentation of the card's ingredients and their properties. Additionally, the mathematics is described that makes two key-retrieval attacks from [6] work.

1 Introduction

Computer security is hard. Any small oversight during the design and implementation of a system can render it insecure. A determined attacker can, and will, use any weakness in the system to get access. Formal methods can be a great help for designers and implementers to obtain precise descriptions of systems and rigorous proofs of their security properties.

The benefits of formal methods already become apparent in making a precise mathematical description of the system. Experience shows that many errors are found during this description phase, even before any verification is attempted.

In the area of protocol analysis, formal methods have become an important tool. Here the proofs are normally *symbolic*: the cryptographic primitives such as encryption and decryption functions are assumed to be sound and the analysis focuses on security properties of the protocol itself. The tools for symbolic protocol analysis are typically automated. Examples include ProVerif [1], based on the pi calculus, and other tools based on model checking (see [5] for a survey). Paulson [10] is a prominent example of interactive symbolic verification, using Isabelle.

Apart from the symbolic approach, assuming perfect cryptography, there is the computational one, which is more realistic but makes verification more difficult. In this approach cryptographic primitives act on strings of bits and security is defined in terms of low probability of success for an attacker. Both protocols and attackers are modelled as probabilistic polynomial-time Turing machines.

More recently cryptographic schemes are verified via a formalisation of game-based probabilistic programs with an associated logic, so that standard patterns

M. Van Eekelen et al. (Eds.): ITP 2011, LNCS 6898, pp. 3–17, 2011.

in provable security can be captured. This approach is well-developed in the CertiCrypt tool, integrated with the theorem prover Coq, see *e.g.* [3,2] (or [16] for an overview).

The security of a software system also depends on the correctness of its implementation. This analysis is part of the well-established field of *program correctness*. Many programming errors do not only limit the functionality of the system, but also lead to exploits. So, all the advances that have been made in the area of program verification are also relevant for security. Here both automated and interactive proof tools are employed.

It is important to realise the limitations of formal methods in the area of computer security. By nature, a formal system describes an abstraction of the actual system and the environment in which it operates. For example, there may be a formal proof of the correctness of a program, but an attacker can try to randomly flip bits by shooting a laser at the chip that runs the program. This will result in behaviour that the formal model does not capture, but that may allow the attacker to break the system.

Another disadvantage of applying formal methods are the costs. It is labour intensive and thus expensive to formalise a system and to prove its correctness. This is why formal verification is only required at the highest Evaluation Assurance Level (EAL7) in the Common Criteria certification[1].

The current paper gives another twist to the use of formal methods in computer security. Rather than proving a system secure, we describe and analyse attacks on an insecure system. To describe these attacks we give a formalisation of the system under attack. The formalisation of the attacks gives a direct connection between the attacks and properties of the system that enables them. In doing so, it clearly demonstrates the system's design failures. The formalisation also gives precise boundaries for the conditions under which the attacks apply.

The flawed system that we consider is the Mifare Classic. This is a contactless (RFID) smart card, sold by NXP (formerly Philips Semiconductors), that is heavily used in access control and public transport (like in London's Oyster card, Boston's Charlie card, and the Dutch OV-chipkaart). It is estimated that over 1 billion copies have been sold worldwide. The design goes back to the early nineties, predating Common Criteria evaluation practices in the smart-card area. The card (and Mifare readers) contains a proprietary encryption algorithm, called Crypto1. It uses a 48-bit linear feedback shift register (LFSR), a device that is well-studied in the literature (see *e.g.*, [13,15]), together with a special filter function that produces the keystream bits.

The security of the card relies partly on the secrecy of this algorithm. Details of Crypto1 have emerged, first after hardware analysis [8][2], and a bit later after a cryptanalysis [6]. The latter reference presents a mathematical model of the card, together with several attack scenarios for key retrieval. The current paper builds on [6] and elaborates certain mathematical (logical) details of this model and these attacks. It does not add new (cryptographic) results, but provides

[1] See `commoncriteriaportal.org`

[2] Made public in a presentation at the Chaos Computer Club, Berlin, 27/12/07.