On Strengthening of Weak Algebraic Manipulation Detection Codes

Maksim Alekseev, St. Petersburg State University of Aerospace Instrumentation, Saint Petersburg, Russia

ABSTRACT

Algebraic manipulation detection codes were introduced in 2008 to protect data against a special type of its modification – an algebraic manipulation. There are three classes of codes: weak, strong and stronger ones. Weak codes detect only weak algebraic manipulations, while strong and stronger are capable to detect both weak and strong manipulations. In this paper, a method to transform codes from weak to stronger is described resulting in a construction of generalized robust codes. The proposed construction forms the second known family of stronger AMD codes. Codes provide simple procedures of error detectiona and correction, and allow using multi-code codec design that makes it applicable in fault-tolerant computing, storage and cryptographic devices.

Keywords: *Algebraic Manipulation, AMD Codes, Data Integrity, Fault-Injection Attack, Nonlinear Codes, Robust Codes*

1. INTRODUCTION

An algebraic manipulation is a model of an undesirable data modification (Cramer, Fehr, & Padró, 2013; Jongsma, 2008). It assumes data distortion by some additive error whose value does not depend on the value of data being distorted. In the paper a concept of algebraic manipulations is described in accordance with a design of secure cryptographic devices (Akdemir, Wang, Karpovsky, & Sunar, 2012; Karpovsky & Wang, 2013; Wang, Karpovsky, Sunar, & Joshi, 2009).

It is well known that cryptographic algorithms' implementations are vulnerable to side-channel attacks (Zhou & Feng, 2005). A fault-injection attack is one of the most efficient attacks, based on an analysis of a cryptographic device's functioning in a presence of faults and computational errors. In many cases, this can lead to a cracking of a cipher-under-attack by revealing a secret key to an attacker. Almost all popular ciphers including RSA, DES and AES are recognized as vulnerable to the fault-injection attack (Abuelyaman & Devadoss, 2005; Anderson & Kuhn, 1998; Dusart, Letourneux, & Vivolo, 2003; Karri, Wu, Mishra, & Kim, 2002; Bar-El, Choukri,

DOI: 10.4018/IJERTCS.2015040101

Naccache, Tunstall, & Whelan, 2006; Canivet et al., 2011; Kim & Quisquater). As a model of injected faults, an algebraic manipulation model is considered (Cramer et al., 2013; Jongsma, 2008; Wang & Karpovsky, 2011).

An algebraic manipulation is defined as an attacker being able to modify data being processed or stored (distort it by injecting a fault) by some device without being able to obtain any information on the value of the data (Jongsma, 2008). From here, it is possible to distinguish two algebraic manipulation models: a weak manipulation model and a strong manipulation model.

In the weak manipulation model, the attacker is not able to control an input to the device-under-attack. In other words, an informational message input into the device is considered to be uniformly distributed and there is no dependency between injected error and data to be distorted. The probability of error masking in case of weak manipulations is denoted P^{w}_{undet}.

The strong manipulation model is used in cases when the attacker is able to control the input to the device. This can lead to a more sophisticated choice of an error's value by the attacker. The probability of error masking in case of strong manipulations is denoted P^{s}_{undet}.

Some special data encoding is required in order to protect data against such manipulations. The most widely used method is to encode data using linear error detecting and correcting codes such as duplication codes, parity check codes and a Hamming code. This method is suitable for situations when arising error patterns correspond to the error detecting capability of the code. However, in cases when an attacker is able to inject specific error patterns, linear codes are not effective. Every q-ary linear code of a dimension k has $q^k - 1$ error patterns (corresponding to nonzero codewords) that are undetectable by this code. Therefore, the attacker can successfully inject one of the codewords as an error into the device. For instance, injecting any double error in data protected with parity check code is undetectable.

A special class of error detecting codes called algebraic manipulation detection (AMD) codes were proposed to protect data against such attacks. Nonlinear weak AMD codes were presented as a method of encoding to detect weak manipulations (these codes are also called robust codes) (Jongsma, 2008; K. J. Kulikowski, Wang, & Karpovsky, 2008; Akdemir et al., 2012). The codes are able to detect every error pattern with some nonzero probability. Thus, weak AMD codes provide robust protection against all error patterns even if the attacker is able to inject specific errors. However, in the case of the strong manipulation model these codes are not effective because of the deterministic encoding.

Nonlinear strong AMD codes were proposed to protect against the strong manipulation model. Strong AMD codes have the probabilistic encoding that is controlled by a random number. This number is assumed to be located inside the protected device and not accessible by the attacker (but may be also distorted by him). Strong AMD codes are constructed in such a way that even if the attacker knows the informational message, he is not able to choose an error that is undetectable for all values of the random number. In other words, strong AMD codes have no undetectable errors under the strong algebraic manipulation model (Karpovsky & Wang, 2013; Cramer et al., 2013).

Stronger AMD codes forms a sub-family of strong codes (Wang & Karpovsky, 2011; Cramer et al., 2013). While strong codes are required to detect only special types of errors, stronger codes are capable to detect all nonzero errors (Alekseev, 2015).

Although algebraic manipulations were described in terms of secure hardware, there are several other applications of this error model and, consequently, AMD codes (Cramer et al., 2013). Originally, AMD codes were introduced to provide data integrity in linear