Information Processing Letters 115 (2015) 731–737

Contents lists available at ScienceDirect

Information Processing Letters

www.elsevier.com/locate/ipl

# Locating modifications in signed data for partial data integrity

Thaís Bardini Idalino [a,*], Lucia Moura [b], Ricardo Felipe Custódio [a], Daniel Panario [c]

[a] *Federal University of Santa Catarina, Florianópolis, Brazil*
[b] *University of Ottawa, Ottawa, Canada*
[c] *Carleton University, Ottawa, Canada*

ARTICLE INFO

*Article history:*
Received 2 September 2014
Received in revised form 24 December 2014
Accepted 25 February 2015
Available online 7 March 2015
Communicated by M. Yamashita

*Keywords:*
Digital signatures
Partial data integrity
Modification localization
Combinatorial group testing
Cover-free families
Cryptography

ABSTRACT

We consider the problem of detecting and locating modifications in signed data to ensure partial data integrity. We assume that the data is divided into $n$ blocks (not necessarily of the same size) and that a threshold $d$ is given for the maximum amount of modified blocks that the scheme can support. We propose efficient algorithms for signature and verification steps which provide a reasonably compact signature size, for controlled sizes of $d$ with respect to $n$. For instance, for fixed $d$ the standard signature size gets multiplied by a factor of $O(\log n)$, while allowing the identification of up to $d$ modified blocks. Our scheme is based on nonadaptive combinatorial group testing and cover-free families.

## 1. Introduction

Digital signature schemes can detect if modifications were done in a signed document, but do not offer information on where exactly those modifications occurred. In this context, even a single bit change would invalidate the whole document. In the present paper, we provide a general Modification Location Signature Scheme which determines which parts of the document were modified, thus ensuring partial data integrity.

Partial data integrity is useful in several scenarios. First, we may need to ensure the integrity of specific parts of a document. For example, in fillable forms the owner may need to assure that the document is official, while some parts are expected to be modified. Second, in a data forensics investigation of a crime, the investigator could have more clues on who is the attacker by knowing what exactly was modified [2]. Third, assuring that part of the data is intact can improve the efficiency of a computer system. For example, in a large database, the modification of some of its records would not invalidate the whole database, avoiding total disruption of service.

One can also see partial data integrity as a solution for guaranteeing privacy protection, where the extraction of selected portions of a signed document is to be shared with another party (content extraction signature [7], redactable signature [3]). Our signature scheme capable of locating modifications can be used in this application by substituting the removed parts by “blank” symbols. The original signature can be used to guarantee the integrity of the non-removed parts.

The Modification Location Signature Scheme (MLSS) proposed in this paper employs combinatorial group testing to determine which blocks of a document contain modifications and which ones are intact. This work is closely related to the work of Zaverucha and Stinson [8] who propose the use of group testing to identify modified

[*] Corresponding author.
*E-mail addresses:* thais.idalino@inf.ufsc.br (T. Bardini Idalino), lucia@eecs.uottawa.ca (L. Moura), custodio@inf.ufsc.br (R.F. Custódio), daniel@math.carleton.ca (D. Panario).

http://dx.doi.org/10.1016/j.ipl.2015.02.014
0020-0190/© 2015 Elsevier B.V. All rights reserved.

documents in batch. However, while in [8] group testing is used on the verifier's end to speed up the batch verification algorithm, in our approach it is used both at the signer's and verifier's end, which greatly improves the signature size over the trivial idea of treating each block of a document as an independent signed document. This trivial idea would require $n$ signatures for a document divided into $n$ blocks, which would not be efficient, while for the cases of interest here we would have the size of a signature multiplied by a factor of $O(\log n)$ instead (see Theorem 1 and the discussion that follows it). While MLSS is applicable to any type of document (text, pictures, videos or a mix), the type of document may influence the way one divides it (see Section 3.4).

Definition of the problem and related work in digital signatures and group testing are presented in Section 2; the algorithms for the proposed Modification Location Signature Scheme and analysis are provided in Section 3; conclusions are given in Section 4.

## 2. Definition of the problem and related work

### 2.1. Digital signatures

Following a general definition [8], a signature scheme is specified by algorithms (Gen, Sign, Verify). Gen($k$) receives a security parameter $k$ and outputs a pair of keys $(s_k, p_k)$, a secret key used for signing and a public key used for verification, respectively. Sign($s_k, m$) outputs a signature $\sigma$ on the message $m$ using the secret key $s_k$. Verify($p_k, \sigma, m$) outputs 1, using the public key $p_k$, if $\sigma$ is a valid signature of $m$, and 0 otherwise.

We propose a general digital signature scheme for signing a document divided into blocks providing, in the case of modifications on the document after signing, the extra capability of locating which blocks have been modified.

**Definition 1.** Modification Location Signature Scheme (MLSS): Let $B = (B_1, \ldots, B_n)$ be a document divided into $n$ blocks. MLSS-Gen($k$) receives a security parameter $k$ and outputs a pair of keys $(s_k, p_k)$. MLSS-Sign($s_k, B$) outputs a signature $\sigma$ on $B$ using the secret key $s_k$. MLSS-Verify($p_k, \sigma, B$) outputs 1 if, using the public key $p_k$, $\sigma$ is a valid signature of $B$, it outputs 0 if $\sigma$ has been modified or is not authentic, and otherwise ($B$ has been modified) outputs extra information on the location of the modifications in $B$.

In this paper, we present an approach, which is based on combinatorial group testing, to solve the challenge stated in Definition 1. Zaverucha and Stinson [8] observe that finding invalid signatures in a batch is a group testing problem, and propose the use of group testing methods to improve the efficiency of the batch verification algorithm. In [8], by exploring the best known group testing algorithms, they show how to run $t$ signature verifications to verify a batch of $n$ signed documents, where $t$ is substantially smaller than $n$; in their case, adaptive and nonadaptive group testing can be used. We use a similar idea to improve the verification algorithm, but we also suggest applying group testing at the signer's end in order to minimize the signature size. We produce $t$ digests, each one involving a subset of the $n$ blocks of the document ($t$ much smaller than $n$). This tuple of digests is signed and sent with the document, allowing the verifier to determine the blocks that were modified. This approach requires nonadaptive group testing, since the digests must be prepared at the signer's end independently of where modifications may occur. In this case, we need an upper bound $d$ on the number of modified blocks; this threshold value $d$ needs to be chosen carefully to keep control on the size $t$.

The presented method is not specific for asymmetric key encryption since one can choose any digital signature algorithm as Sign and Verify. In this paper, our presentation is based on public key digital signatures.

### 2.2. Group testing

The purpose of group testing is to identify $d$ defective elements from a set of $n$ elements pooled into $t$ groups where $t < n$. The groups are tested, instead of all elements individually. For a subset of elements (pool), if at least one of the elements is defective, we return the result for the test as a "fail"; if no element is defective we return the result of the test as a "pass". In *adaptive* group testing, the results of the previous tests are used to determine subsequent tests; in *nonadaptive* group testing, all the tests are specified ahead of time, which allows them to be run in parallel (see the book by Du and Hwang [1]). In our method, we need to use nonadaptive group testing, since the organization of blocks into groups must be done at the signer's end, and thus before the modifications ("defects") are introduced. In this case, the most effective way to detect up to $d$ defectives is to use a cover-free family (CFF).

**Definition 2.** A *d-cover-free family*, denoted $d$-CFF$(t, n)$ is a $t \times n$ binary matrix $M$ with $n \geq d+1$, such that for any set of column indexes $C$ with $|C| = d$ and column $c \notin C$, the following property holds: there exists a row $i$ satisfying $M_{i,c} = 1$ and $M_{i,j} = 0$ for all $j \in C$.

We form the tests according to the rows of matrix $M$, i.e. for each $1 \leq i \leq t$, test $i$ consists of exactly the items $j$ for which $M_{i,j} = 1$. The properties of CFFs assure that if the number of defectives is at most $d$ then it is enough to determine the non-defective items from the passing tests. Then, we can conclude that all other items are defective.

Given $d$ and $n$, we wish to find a $d$-CFF$(t, n)$ for the smallest possible $t$, which we call $t(d, n)$. We mention a few useful explicit constructions found in the literature. When $d = 1$, we can use Sperner theorem [6] to show that the smallest number of tests possible is $t(1, n) = \min\{t : \binom{t}{\lfloor t/2 \rfloor} \geq n\}$. We observe that as $n \to \infty$, $t(1, n) \sim \log_2 n$. The top-left of Fig. 1 gives an example with $n = 6$ and $t = 4$. For arbitrary $d$ and $n$, we consider the constructions of Porat and Rothschild [5] (with $t \leq (d+1)^2 \ln n$) and Pastuszak et al. [4] (with $t \leq (d+1)\sqrt{n}$). Some of these constructions are surveyed in [8]. We note that for specific small $d$ one can find more efficient constructions than the general ones listed here; for example, for $d = 2$ a smaller $t$ can be achieved (see [1], Section 7.5). This more specific analysis is out of the scope of this paper.