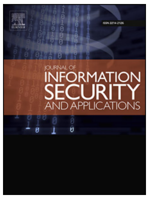
Journal of Information Security and Applications 42 (2018) 118–134

Contents lists available at ScienceDirect

Journal of Information Security and Applications

journal homepage: www.elsevier.com/locate/jisa



Functional Signcryption

Pratish Datta[a,*], Ratna Dutta[b], Sourav Mukhopadhyay[b]

[a] Secure Platform Laboratories, NTT Corporation, Tokyo, 180-8585, Japan
[b] Department of Mathematics, IIT Kharagpur, Kharagpur, 721302, India

ARTICLE INFO

Article history:

2010 MSC:
94A60,

Keywords:
Functional signcryption
Indistinguishability obfuscation
Statistically simulation-sound non-interactive zero-knowledge proof of knowledge
Polynomial-size circuits

ABSTRACT

Functional encryption (FE) allows to restrict decryption in a highly sophisticated fashion, whereas, *functional signature* (FS) enables to enforce arbitrarily complex control on signing capabilities. This paper introduces a new cryptographic primitive, termed as *functional signcryption* (FSC), which *unifies* the functionalities of FE and FS into a *cost-effective* formulation. FSC is a crucial step towards efficient implementation of modern digital communication and storage systems that demand advanced forms of confidentiality and authenticity simultaneously. Precisely, we make the following contributions:

– First, we present a formal definition of FSC and carefully formulate its security requirements.
– Next, we provide a generic construction of FSC supporting signing and decryption functionalities realizable by *general polynomial size circuits*, based on fundamental cryptographic tools, namely, *indistinguishability obfuscation* (IO) for circuits and *statistically simulation-sound non-interactive zero-knowledge proof of knowledge* (SSS-NIZKPoK).
– Finally, we exhibit a number of representative applications of this interesting cryptographic primitive:
(i) We develop the *first ever attribute-based signcryption* (ABSC) scheme for *arbitrary polynomial size circuits* from FSC.
(ii) We show how FSC can be utilized to build SSS-NIZKPoK systems and IO for general circuits. This result in conjunction with our FSC construction can be interpreted as establishing an *equivalence* between FSC and the other two important cryptographic primitives.

1. Introduction

Confidential as well as authenticated message transfer and storage has been one of the central focus of cryptography since years. In the public key setting, a standard approach for achieving this goal has been to utilize digital signature and public key encryption primitives in sequence. However, this strategy amounts to incurring a direct addition of the costs of both primitives. *Digital signcryption*, introduced by Zheng [60] and subsequently explored in a long sequence of works [7,40,41,57,61], is an ambitious cryptographic paradigm that unifies the functionalities of both encryption and authentication in a cost-effective formulation.

However, in the standard notion of digital signcryption, the control over signing and decryption rights is "all or nothing": Only those in possession of the secret signing key corresponding to the system public key can signcrypt a message and the resulting ciphertext can be unsigncrypted by only those having the matching secret decryption key. In the modern era of Internet communication and cloud technology where multiple users are involved, such an "all or nothing" control over signing and decryption capabilities is no longer sufficient, rather highly sophisticated restrictions over signing and decryption rights must be enforced.

In order to realize fine-grained control over decryption capabilities, the concept of *functional encryption* (FE) has been introduced [14,48]. An FE scheme includes a trusted authority which holds a master secret key and publishes system public parameters. An encrypter uses this system public parameters to encrypt a message. A decrypter may obtain a decryption key $\textsc{dk}(g)$ for some decryption function g from the authority if and only if the authority deems that the decrypter is entitled to possess that key. The decrypter can now use the decryption key $\textsc{dk}(g)$ to decrypt a ciphertext encrypting some message m to obtain $g(m)$, and nothing more about m.

Depending on the function family realized, FE schemes are classified into various sub-categories, e.g., *identity-based encryption* (IBE) [12], *attribute-based encryption* (ABE) [32,56], *predicate encryption* (PE) [39], *inner-product* FE (IPFE) [1], *quadratic* FE

[*] Corresponding author.
E-mail addresses: pratish.datta.yg@hco.ntt.co.jp (P. Datta), ratna@maths.iitkgp.ernet.in (R. Dutta), sourav@maths.iitkgp.ernet.in (S. Mukhopadhyay).

https://doi.org/10.1016/j.jisa.2018.08.004
2214-2126/© 2018 Elsevier Ltd. All rights reserved.

(QFE) [8] etc., which are being actively studied by the crypto-community. For instance, in case of a (key-policy) ABE scheme, a ciphertext encrypts a pair of the form (y, M), where y is a string of descriptive attributes and M is the actual payload, while a decryption key is associated with a function $g_\wp$, where $\wp$ is a decryption policy. Decrypting a ciphertext encrypting an attribute-string-payload pair (y, M) using a decryption key associated with a decryption function $g_\wp$ recovers $g_\wp(y, M)$, which is defined to be $g_\wp(y, M) = (y, M)$, if $\wp(y) = 1$, i.e., the decryption policy $\wp$ accepts the attribute string y, and $(y, \perp)$, otherwise. Here, $\perp$ is a distinguished symbol indicating failure. One typical example of a descriptive attribute string y could be $y = (\mathsf{AGE} = 35) \| (\mathsf{DESIGNATION} = \mathsf{PROFESSOR})$, while that of a decryption policy $\wp$ could be $\wp = [(\mathsf{AGE} \geq 40) \bigwedge (\mathsf{DESIGNATION} = \mathsf{ASSOCIATE\ PROFESSOR})] \bigvee [\mathsf{DESIGNATION} = \mathsf{PROFESSOR}]$. Note that $\wp$ accepts y in this example.

On the other hand, *functional signature* (FS), introduced in [11,15], allows managing complex signing credentials. Just like an FE scheme, an FS system also involves a trusted authority that publishes system public parameters and possesses a master signing key which can be used for signing any message and providing a constrained signing key $\textsc{sk}(f)$ for some signing function f to a signer after verification of its signing credentials. This restricted signing key $\textsc{sk}(f)$ can be used for producing signatures, verifiable under the system public parameters, on only those messages that are in the range of the function f.

Similar to FE, FS schemes are also categorized into numerous sub-classes based on the underlying functionality, e.g., *group signatures* (GS) [18], *ring signatures* (RS) [55], *attribute-based signatures* (ABS) [45] etc., which are also being investigated to a great extent. For example, in case of a (key-policy) ABS, a message to be signed is the form of a pair $(\bar{y}, M)$, where $\bar{y}$ is a string of descriptive signing attributes and M is the actual payload, while a signing key is associated with a signing function $f_\wp$, where $\wp$ is a signing policy. An attribute string-payload pair is defined to lie within the range of a signing function $f_\wp$ if and only if $\wp(\bar{y}) = 1$, i.e., $\bar{y}$ is accepted by $\wp$

Besides the steady development in the different sub-classes of FE and FS, in the past few years, a remarkable progress has taken place towards realizing FE and FS schemes supporting *general* functionalities, such as those expressible in terms of *arbitrary polynomial-size circuits* based on advanced cryptographic primitives such as indistinguishability obfuscation, multilinear maps, statistically simulation-sound non-interactive zero-knowledge proof of knowledge, and so on [2,6,11,15,26,28,37,59]. However, given this state of the art, exercising fine-grained control over the signing and decryption rights in a *generic* multi-user confidential and authenticated digital communication or storage system still necessitates implementing both FE and FS for general functionalities sequentially, that entails summing up the cost incurred by both the primitives.

In this work, we put forward a *new* cryptographic paradigm termed as *functional signcryption* (FSC) that unifies the functionalities of both FE and FS. In other words, FSC aims to provide enhanced access control in the context of the traditional digital signcryption. FSC solves the issue of simultaneously managing signing and decryption credentials in a multi-user environment with better efficiency. More precisely, in an FSC scheme, we consider a trusted authority that holds a master secret key and publishes system public parameters. Using its master secret key, the authority can provide a signing key $\textsc{sk}(f)$ for some signing function f to a signcrypter, as well as, a decryption key $\textsc{dk}(g)$ corresponding to some decryption function g to a decrypter after verifying their credentials. Now such a signing key $\textsc{sk}(f)$ enables a signcrypter to signcrypt, i.e., encrypt and authenticate simultaneously only those messages which are in the range of f, while a decryption key $\textsc{dk}(g)$ can be utilized to unsigncrypt a ciphertext, which is the signcryption of some message m to retrieve $g(m)$ only and to verify the authenticity of the ciphertext at the same time.

We define two security notions for FSC, namely, *message confidentiality* and *ciphertext unforgeability*. Roughly speaking, message confidentiality guarantees that arbitrary collusion of decrypters cannot retrieve any additional information about the signcrypted message from a ciphertext beyond the union of what they could obtain individually. On the other hand, ciphertext unforgeability assures that collusion of signcrypters cannot help them to generate a valid signcryption of a message which none of them could have signcrypted on their own.

A motivating practical application of FSC could be the following: Suppose the government of some country is collecting complete photographs of individuals as part of the census and storing the collected data in a large server to allow utilizing it in future by other organizations for various survey purposes. For maintaining the security and improving the quality of the collected photos at the same time, the government is using some photo-processing software that edits the photos and encrypts them before storing them to the server. Now, it is desirable that the software is allowed to perform only some minor touch-ups of the photos such as changing the color scale or removing red eyes, but is not allowed to make more significant changes such as merging two photos or cropping a picture. FSC can naturally address this issue as follows: The government, acting as the trusted authority, would provide the photo-processing software (signcrypter) the signing keys ($\textsc{sk}(f)$) which allows it to signcrypt original photographs with only the allowable modifications (i.e., those in the range of f) and store the signcrypted photos in the server. Later, when some organization (decrypter) wants to access only those informations from stored photos meeting certain criteria (g), e.g., faces of individuals residing in a particular city, the government would give the organization the corresponding functional decryption key ($\textsc{dk}(g)$) after being fully convinced about the credentials of the organization. Now, when the organization would access that data base (i.e., signcryption of m) using the obtained decryption key, it could only obtain the face portion of the photographs of individuals living in that particular city ($g(m)$) and would be convinced that the photos obtained were undergone through only minor photo-editing modifications.

We note that the attempt to introduce fine-grained access control in signcryption setting has already been considered in the literature. In particular, a series of works have investigated a primitive called *attribute-based signcryption* (ABSC) for progressively more expressive access policies [23,24,36,44,49,51–54,58]. ABSC comes in two flavors, namely, *key-policy* and *ciphertext-policy*. In a key-policy ABSC scheme, there is a trusted authority who publishes system public parameters and uses a master secret key to produce signing and decryption keys corresponding to specific signing and decryption policies. Now, the holder of such a signing key can signcrypt messages with respect to any decryption attribute string, and only those signing attribute strings on which the signing policy predicate embedded in the signing key evaluates to 1. The signature and decryption attribute strings are attached in the clear with the ciphertext, so that anyone with a decryption key embedding a decryption policy predicate that outputs 1 on the associated decryption attribute string can verify the authenticity of the ciphertext with respect to the associated signing attribute string, and also retrieve the signcrypted message. In ciphertext-policy ABSC, the roles of policy predicates and attribute strings are reversed.

Thus, it is evident that the objective of ABSC is to provide the functionalities of both ABE and ABS in an unified primitive. But our notion of FSC not merely aims to enforce attribute-based access control in signcryption, but captures much more general