Signature for Objects: Formalizing How to Authenticate Physical Data and More

Ryuya Hayashi[1,2], Taiki Asano[3], Junichiro Hayata[4], Takahiro Matsuda[2], Shota Yamada[2], Shuichi Katsumata[5,2], Yusuke Sakai[2], Tadanori Teruya[2], Jacob C. N. Schuldt[2], Nuttapong Attrapadung[2], Goichiro Hanaoka[2], Kanta Matsuura[1], and Tsutomu Matsumoto[2,6]

[1] The University of Tokyo, Tokyo, Japan
[2] National Institute of Advanced Industrial Science and Technology, Tokyo, Japan
[3] GMO Cybersecurity by Ierae, Inc., Tokyo, Japan
[4] Deloitte Tohmatsu Cyber LLC, Tokyo, Japan
[5] PQShield Ltd, Oxford, UK
[6] Yokohama National University, Kanagawa, Japan
{rhys,kanta}@iis.u-tokyo.ac.jp
asano@gmo-cybersecurity.com
junichiro.hayata@tohmatsu.co.jp
{t-matsuda,yamada-shota,yusuke.sakai,tadanori.teruya,jacob.schuldt,
n.attrapadung,hanaoka-goichiro,matsumoto.tsutomu}@aist.go.jp
shuichi.katsumata@pqshield.com

Abstract. While the integrity of digital data can be ensured via digital signatures, ensuring the integrity of physical data, i.e., objects, is a more challenging task. For example, constructing a digital signature on data extracted from an object does not necessarily guarantee that an adversary has not tampered with the object or replaced this with a cleverly constructed counterfeit. This paper proposes a new concept called *signatures for objects* to guarantee the integrity of objects cryptographically. We first need to consider a mechanism that allows us to mathematically treat objects which exist in the physical world. Thus, we define a model called an *object setting* in which we define physical actions, such as a way to extract data from objects and test whether two objects are identical. Modeling these physical actions via oracle access enables us to naturally enhance probabilistic polynomial-time algorithms to algorithms having access to objects — we denote these *physically enhanced algorithms* (PEAs). Based on the above formalization, we introduce two security definitions for adversaries modeled as PEAs. The first is unforgeability, which is the natural extension of EUF-CMA security, meaning that any adversary cannot forge a signature for objects. The second is confidentiality, which is a privacy notion, meaning that signatures do not leak any information about signed objects. With these definitions in hand, we show two generic constructions: one satisfies unforgeability by signing extracted data from objects; the other satisfies unforgeability and confidentiality by combining a digital signature with obfuscation.

1 Introduction

Cryptography provides a formal ground to authenticate and secure *digital* data. For instance, by using a signature scheme to sign a (digital) message, the verifier can detect fraudulent activities such as message injections and impersonation attacks, while further providing a mechanism to hold the signer accountable for the message.

On the other hand, authenticating *physical* data (or what we simply call *objects*) are much more challenging. Consider a hardware vendor shipping a microchip to a client. While the client may hold a digital receipt of the transaction, this does not prevent an adversary from substituting the product from a counterfeit. Unless the counterfeit is obviously non-functional, it would be difficult for an average consumer to detect the authenticity of the received product. Moreover, even if the product was non-functional, the client would not know if it was an inferior vendor or if a substitution attack occurred since the sender of the product cannot be held accountable by cryptographic means.

The inability to authenticate objects has had a grave economic impact. This is exacerbated by the globalization of supply chains: since each component of a product can be made in different regions and countries, protecting against substitution attacks becomes increasingly difficult. The OECD and the EU's Intellectual Property Office report that 3.3% of global trade, which amounts to 509 Billion USD, is counterfeit or pirated goods [22] out of which the market for counterfeit electronic is 169 Billion USD according to Havocscope [1]. In 2019, the United States Department of Homeland Security reports the estimated manufacture's suggested retail price (MSRP) for seized electronic goods to be 106 Billion USD [28]. The amount of damage could be amplified if we consider indirect consequences of such counterfeit electronics. For example, a tiny microchip was injected in the server's motherboard of Elemental during manufacturing [26]; the server was obliviously used by the Department of Defense data centers, the CIA's drone operations, and the onboard networks of Navy warships.

While various countermeasures are taken by companies and state actors to digitally and physically secure the global supply chains, there is still much to be improved. In the above example, once a counterfeit makes it into the supply chain and received by a consumer, it seems difficult to detect such substitution within the current systems. An ambitious goal would be to cryptographically secure the supply chains by authenticating objects as we do digital data. In other words, can we use cryptography to formally *prove* that it is difficult for adversaries to modify an object without being detected, while further holding the sender of an object accountable? This is the question we tackle in this work.

1.1 Our Contribution

In this work, we lay the foundation of digitally authenticating objects (i.e., physical data) and propose a new concept called *signatures for objects* (SfO). The combination of a SfO and a standard signature for digital data brings us one step closer to the sought-after goal of cryptographically securing the entire global

supply chain: any injection of forged objects and digital data become detectable and the original signer (i.e., sender) of these objects and digital data can be held accountable.

Concretely, we first introduce new tools to formally handle objects in a cryptographically sound manner. To give an idea, an object exists in the physical world while a signature produced by a Turing machine (or a probabilistic polynomial-time (PPT) algorithm) exists in the digital world. Therefore, at the minimum, there needs to be a mechanism to translate objects into digital data which a Turing machine can operate on. While there have been several research aiming to bridge the digital world and the physical world (e.g., [14,23,30]) , we are the first to formalize *physically-enhanced algorithms* (PEA) — a new computational model that enhances standard PPT algorithms by physical properties.

With PEA formalized, we define a signature of objects with an intuitive unforgeability security notion analogous to standard digital signatures. That is, (informally) an adversary should not be able to forge a signature on an object that has not been signed before. We then provide a simple and efficient generic construction based on any standard signature scheme. We further explore a potentially relevant security notion of *confidentiality* — again analogous to those considered for standard signature schemes [7] — and construct an SfO scheme satisfying this security notion based on obfuscation [2,3]. We elaborate on our contributions below.

How to Treat Objects. To treat physical object in a cryptographically sound manner, we need to answer the following fundamental questions: (A) how to capture objects in a well-defined way, (B) how to translate objects into digital data, and (C) how to enhance the definition of PPT algorithms to handle objects. Answering these questions will be the main theoretical contribution of our work.

We first define an *object setting* to answer questions (A) and (B). Informally, an object setting is defined with respect to a *relation function* and a *sensing function*. At a high level, the (possibly non-efficient) relation function decides whether two objects are the same, e.g., two laptops may be considered the same object if they are the same model, or they may be considered different with different MAC addresses for each individual. A sensing function on the other hand takes an object as input and outputs a digital digest of the object. For instance, a sensing function could be a photograph of an object along with its weight, size, and color. The concrete definition of an object setting is necessarily application dependent.

We then define *physically-enhanced algorithms* (PEAs) to answer question (C). A PEA should capture all the intuitive and natural capability of an algorithm having access to an object. Continuing with the above example, given a laptop, we can consider taking some pieces out from the laptop. Formally, we capture these capabilities by enhancing the definition of a PPT algorithm by further giving it oracle access that embeds an object. For instance, a PPT algorithm can query a bit string that represents, say "open a laptop", to the oracle and the oracle will modify the embedded laptop accordingly. Importantly, only giving handles to objects and not the object itself is what allows to naturally enhance

PPT algorithms to PEAs. We also introduce a sub-class of PEAs that we call *sensing algorithms*, whose only physical action is to use the sensing function. More details are provided in Section 3.2. With object settings and PEAs formally defined, we are ready to modify standard cryptographic primitives defined against PPT algorithms to PEAs.

How to Sign Physical Data: Signature for Objects. In this work, we propose a new cryptographic primitive called signature for objects (SfO). At a high level, it is defined analogously to standard digital signatures, where the difference is that signing and verification is done with respect to objects. To formally define such an idea, we rely on an object setting, PEAs, and sensing algorithms as defined above. Specifically, the signing algorithm of a SfO is a sensing algorithm, instead of the usual PPT algorithm, which has oracle access to a sensing function: given an object, the signing algorithm can query the sensing function to obtain a digital digest of the object and finally outputs a *digital* signature.[7] The verification algorithm, which is also a sensing algorithm, is defined similarly. It takes a digital signature along with an object as input and verifies the validity of the signature.

Correctness of SfO is defined using the relation function defined above. If the two objects that the signing and verification algorithms take as input are identical under the relation function, then the signature should verify. In particular, unlike digital data where equivalence is easy to check (i.e., check if the bit strings are identical), we require relation functions to check equivalence of objects.

Security of SfO is captured by *existential unforgeability under chosen-object attacks* (EUF-COA security), which is analogous to existential unforgeability against chosen-message attacks (EUF-CMA) for an ordinary digital signature scheme. The adversary is a PEA and we allow it to query for signatures on different objects. However, the definition requires subtle care since we cannot allow the adversary to query arbitrary objects. For instance, we can consider an adversary that queries a device solving factorization in polynomial time. While such a pathological adversary can break any cryptographic scheme based on the hardness of factorization, it does not appropriately capture a practical adversary. To remove these pathological adversaries, we restrict the adversary so that it can obtain signatures on objects that can be obtained by performing some *action* (e.g., turning in some other direction, heating it up) on the challenge object. Specifically, the adversary can only perform actions on the challenge object by means of oracle calls, where the set of allowed actions will be defined by the object setting. In reality, this reflects the intuition that a counterfeit can be created through modifying the original product.

Finally, once all the definitions of a SfO are formalized, the construction of a SfO is simple and intuitive. We provide an efficient generic construction of an SfO signature scheme that satisfies EUF-COA security, based on any standard digital

[7] Recall that the input and output of a sensing algorithm are the same as PPT. The only difference is that it also has oracle access to sensing functions, which allows to indirectly operate on objects.

signature scheme. See Section 3.2 for the definition of sensing algorithms and PEAs, Section 3.3 for the definitions for SfO, and Section 3.4 for the construction.

Adding Confidentiality. In some applications of SfO, it is possible that the signature on a particular object gets leaked to the public. If the object being signed is sensitive, e.g., an unpublicized hardware, then we would like the signature to leak no information of the object.

To this end, we consider an additional security notion for SfO called *confidentiality under chosen-object attacks* (Conf-COA security). This is a simulation-based security definition, similar to the semantic security of public-key encryption [12] and virtual black-box security of obfuscation [2,3]. Informally, we say that a SfO scheme is Conf-COA secure if for any PEA adversary $\mathcal{A}$ that, given a signature of some object, tries to guess some information about the data of object being signed, there exists a PEA simulator $\mathcal{S}$ that can succeed the guess without seeing the signature. We then show a generic construction and instantiation of an EUF- and Conf-COA secure scheme by combining a standard signature and an obfuscation. See Section 4 for the details.

1.2 Related Works

Relationship to Existing Digital Signature Schemes. Some existing works considered confidentiality for standard signature schemes [7,9]. In our setting, data to be signed is fuzzy (i.e., sensing outputs can be different every time), so there are challenges in applying these technologies to our model. There also exist some works that use fuzzy data for creating signing keys [19,31,33]. However, we sign on fuzzy data rather than generating signing keys from it, so our model is orthogonal to theirs.

Relationship to Existing Physical Cryptographic Protocols. Some works proposed cryptographic protocols that consider physical actions (e.g., position based cryptography [6], physical zero-knowledge [8], card-based cryptography [20]). These works construct protocols using physical information, but they do not formalize physical things in a cryptographically sound manner. Another related work by Ishai et al. [14] studied sensing as a cryptographic function. We again note that we are the first to formalize physical actions cryptographically.

Implementation of Sensing and Identification of Objects. To realize a signature for objects scheme, we need to implement a *sensing* function which allows us to extract data from objects (e.g., photographic images) with which we can identify objects. We can use object detection/recognition tools as sensing, which have been proposed since the recent progress of machine learning (e.g., [13,17,18,24,25]). Another candidate technique to realize sensing and identification of objects is via a physically unclonable function (PUF) [11,23]. (See the paragraph *Examples.* in Section 3.1.)

Supply Chain Security. There has been much interest in supply chain security. Lee et al. [16] showed that a safer supply chain could be achieved at a lower cost by re-designing appropriate management and operational design using information technology. In recent years, research has been conducted using the latest technologies, such as blockchain (e.g., [15,21,27]) and machine learning (e.g., [4,29,32]), to configure more secure supply chains. However, discussions of security in these studies are heuristic and typically there are no formal security models and/or proofs.

2 Preliminaries

In this section, we review basic notation and existing cryptographic notions used in this paper.

Basic Notation. $\mathbb{N}$, $\mathbb{Z}$, and $\mathbb{R}$ denote the sets of all natural numbers, integers, and real numbers, respectively. For $n \in \mathbb{N}$, we define $[n] := \{1, \ldots, n\}$. For a set S, $|S|$ denotes its size. For two strings x and y, $(x \stackrel{?}{=} y)$ is defined to be 1 if $x = y$ and 0 otherwise. For a probabilistic algorithm A, we write $x \leftarrow \mathsf{A}(y)$ to mean that A on input y outputs x, and when we need to make the randomness r used by A explicit, we write $x \leftarrow \mathsf{A}(y; r)$ (in which case the computation of A is deterministic with respect to the inputs y and r). We say that a non-negative function $f : \mathbb{N} \to \mathbb{R}$ is negligible if for all $c \in \mathbb{N}$ there exists $\lambda_0 \in \mathbb{N}$ such that $f(\lambda) \leq \lambda^{-c}$ for all $\lambda \geq \lambda_0$. "negl" denotes an unspecified negligible function, and "poly" denotes an unspecified positive polynomial. PPT stands for *probabilistic polynomial-time*.

2.1 Digital Signature

Here we briefly recall the definition of an ordinary digital signature scheme. A digital signature scheme DS consists of the following three PPT algorithms.

- $\mathsf{DS.KG}(1^\lambda) \to (\mathsf{vk}, \mathsf{sk})$: This is the key generation algorithm, which takes the security parameter 1^λ as input, and outputs a verification/signing key pair $(\mathsf{vk}, \mathsf{sk})$.
- $\mathsf{DS.Sign}(\mathsf{sk}, m) \to \sigma$: This is the algorithm for generating a signature, which takes a signing key sk and a message m as input, and outputs a signature σ.
- $\mathsf{DS.Ver}(\mathsf{vk}, m, \sigma) \to 1/0$: This is the algorithm for verifying a signature, which takes a verification key vk, a message m, and a signature σ as input, and outputs 1 (accept) or 0 (reject).

As the correctness condition, we require that for all $\lambda \in \mathbb{N}$, $(\mathsf{vk}, \mathsf{sk}) \leftarrow \mathsf{DS.KG}(1^\lambda)$, $m \in \{0,1\}^*$, and $\sigma \leftarrow \mathsf{DS.Sign}(\mathsf{sk}, m)$, we have $\mathsf{DS.Ver}(\mathsf{vk}, m, \sigma) = 1$.

We recall the definition of existential unforgeability under chosen-message attacks (EUF-CMA security). For a digital signature scheme $\mathsf{DS} = (\mathsf{DS.KG}, \mathsf{DS.Sign},$

$\mathsf{DS.Ver}$) and a PPT adversary $\mathcal{A}$, we consider the following experiment:

$$\mathsf{Expt}^{\mathsf{EUF\text{-}CMA}}_{\mathsf{DS},\mathcal{A}}(\lambda): \left[\begin{array}{l} \mathsf{Msg} \leftarrow \emptyset; \quad (\mathsf{vk},\mathsf{sk}) \leftarrow \mathsf{DS.KG}(1^\lambda); \\ (m',\sigma') \leftarrow \mathcal{A}^{\mathsf{DS.Sign}(\mathsf{sk},\cdot)}(\mathsf{vk}); \\ \text{If } \mathsf{DS.Ver}(\mathsf{vk},m',\sigma') = 1 \wedge m' \notin \mathsf{Msg} \\ \text{then return 1 else return 0} \end{array}\right]$$

where $\mathsf{DS.Sign}(\mathsf{sk},\cdot)$ is the signing oracle that takes a message m as input and operates as follows: it updates Msg by $\mathsf{Msg} \leftarrow \mathsf{Msg} \cup \{m\}$, computes a signature $\sigma \leftarrow \mathsf{DS.Sign}(\mathsf{sk},m)$, and returns σ to $\mathcal{A}$.

Definition 1 (EUF-CMA). *We say that a digital signature scheme* DS *is EUF-CMA secure if for any PPT adversary* $\mathcal{A}$*, we have* $\mathsf{Adv}^{\mathsf{EUF\text{-}CMA}}_{\mathsf{DS},\mathcal{A}}(\lambda) := \Pr[\mathsf{Expt}^{\mathsf{EUF\text{-}CMA}}_{\mathsf{DS},\mathcal{A}}(\lambda) = 1] = \mathrm{negl}(\lambda)$.

2.2 Obfuscation

Here, we recall the definitions for obfuscation (for circuits) [3] that we use in this paper.

Let $\mathcal{C} = \{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$ be a class of polynomial-size circuits. Let Obf be a PPT algorithm that takes the security parameter 1^λ and a circuit $C \in \mathcal{C}_\lambda$ as input, and outputs some circuit $\widetilde{C}$ (called an "obfuscated circuit"). Obf is said to be an obfuscator for $\mathcal{C}$ if it satisfies the following functional requirements: For all $\lambda \in \mathbb{N}$ and $C \in \mathcal{C}_\lambda$, we have

- *(Correctness (a.k.a. functionality preservation):)* If $\widetilde{C} \leftarrow \mathsf{Obf}(1^\lambda, C)$, we have that $C(x) = \widetilde{C}(x)$ for all inputs x (in the domain of C).
- *(Polynomial slowdown:)* $|\widetilde{C}| = \mathrm{poly}(\lambda, |C|)$.

For a security notion for obfuscation, we will consider *distributional virtual black-box (VBB) security* [2], which is sufficient for our purpose.

Definition 2 (Distributional Virtual Black-Box Security). *Let* Obf *be an obfuscator for a circuit class* $\mathcal{C} = \{\mathcal{C}_\lambda\}_{\lambda\in\mathbb{N}}$*. Let* $\mathcal{D} = \{\mathcal{D}_\lambda\}_{\lambda\in\mathbb{N}}$ *be a class of distributions such that* $\mathcal{D}_\lambda$ *is a distribution over* $\mathcal{C}_\lambda$ *for each* $\lambda \in \mathbb{N}$*. We say that* Obf *is* distributional virtual black-box (VBB) secure *for* $\mathcal{D}$ *if the following holds: For any PPT adversary* $\mathcal{A}$*, there exists a PPT simulator* $\mathcal{S}$ *such that for any PPT predicate* $\mathcal{P}$*, we have* $\mathsf{Adv}^{\mathsf{dVBB}}_{\mathsf{Obf},\mathcal{A},\mathcal{S},\mathcal{P}}(\lambda) := |\Pr[\mathsf{Expt}^{\mathsf{dVBB\text{-}Real}}_{\mathsf{Obf},\mathcal{A},\mathcal{P}}(\lambda) = 1] - \Pr[\mathsf{Expt}^{\mathsf{dVBB\text{-}Sim}}_{\mathsf{Obf},\mathcal{S},\mathcal{P}}(\lambda) = 1]| = \mathrm{negl}(\lambda)$*, where the real experiment* $\mathsf{Expt}^{\mathsf{dVBB\text{-}Real}}_{\mathsf{Obf},\mathcal{A},\mathcal{P}}(\lambda)$ *and the simulator's experiment* $\mathsf{Expt}^{\mathsf{dVBB\text{-}Sim}}_{\mathsf{Obf},\mathcal{S},\mathcal{P}}(\lambda)$ *are defined as in Fig. 1.*

3 Signature for Objects: Definition and Basic Construction

This section provides the definitions for signature for objects (SfO). An SfO scheme enables us to generate a signature on an object and to detect a replacement for a fake one by verifying a signature.

$\mathsf{Expt}^{\mathsf{dVBB\text{-}Real}}_{\mathsf{Obf},\mathcal{A},\mathcal{P}}(\lambda)$:
$C \leftarrow \mathcal{D}_\lambda$
$\widetilde{C} \leftarrow \mathsf{Obf}(1^\lambda, C)$
$g \leftarrow \mathcal{A}(1^\lambda, \widetilde{C})$
Return $(\mathcal{P}(C) \stackrel{?}{=} g)$.

$\mathsf{Expt}^{\mathsf{dVBB\text{-}Sim}}_{\mathsf{Obf},\mathcal{S},\mathcal{P}}(\lambda)$:
$C \leftarrow \mathcal{D}_\lambda$
$g \leftarrow \mathcal{S}^C(1^\lambda, |C|)$
Return $(\mathcal{P}(C) \stackrel{?}{=} g)$.

Fig. 1. The experiments for defining distributional VBB security of an obfuscator Obf. In the simulator's experiment $\mathsf{Expt}^{\mathsf{dVBB\text{-}Sim}}_{\mathsf{Obf},\mathcal{S},\mathcal{P}}(\lambda)$, $\mathcal{S}^C$ means that $\mathcal{S}$ has oracle access to the circuit C.

We note that in order for this type of signature primitive to be possible, there must first exist some mechanism to (1) extract some digital data from an object, and (2) judge whether two objects are the same or not. (For example, imagine a setting of identifying objects based on photographic images of the objects (taken by a camera) by using some technology of image recognition and machine learning.) In this paper, we assume that such a mechanism is given, and define an SfO scheme on top of such a mechanism. Therefore, we first formalize such a mechanism for extracting digital data of objects as well as identifying objects as an *object setting*, and then give a formalization for SfO. For defining the security of SfO, we would like to consider a class of adversaries that can perform some physical actions on objects. To this end, we also introduce the notion of *physically-enhanced algorithms* (PEAs) that captures such a class of adversaries.

The rest of this section is as follows: In Section 3.1, we give a formalization for an object setting. In Section 3.2, we give definitions for a class of algorithms that can handle objects. Then, in Section 3.3, we give definitions for an SfO scheme. Finally, in Section 3.4, we show our basic construction of SfO as well as its proof of security.

3.1 Object Setting

Here, we give the definition for an object setting in which (1) how to extract data from an object, and (2) how to identify two objects, are defined. Looking ahead, an SfO scheme as well as the class of adversaries for it will be based on top of this setting.

Definition 3 (Object Setting). *An object setting $\mathcal{OS}$ consists of* $(\mathbb{X}, \mathbb{V}, \mathcal{X}, \mathbb{D}, \mathsf{Sen}, R_{\mathbb{X}}, R_{\mathbb{D}}, \mathbb{C})$*, each of which is defined as follows:*

- $\mathbb{X}$**:** *This is the set of all objects that can be treated by the sensing function* Sen *explained below.*
- $\mathbb{V}$ $(\subseteq \mathbb{X})$**:** *This is a subset of $\mathbb{X}$. We will refer to the elements in $\mathbb{V}$ as* valid *and will require that for the sensing function* Sen *defined below, it holds that* $\mathsf{Sen}(x) \neq \perp$ *if and only if* $x \in \mathbb{V}$*.*
- $\mathcal{X}$**:** *This is a distribution over $\mathbb{V}$. When a new object is created, it follows this distribution.*

$\mathbb{D}$ ($\subseteq \{0,1\}^*$)**:** *This is the set of all (digital) data that could be generated by the sensing function* Sen *explained below.*

Sen**:** *This is a "sensing" function that takes an object $x \in \mathbb{X}$ as input, and outputs data $D \in \mathbb{D}$ or the special invalid symbol $\perp \notin \mathbb{D}$. This models some device that "extracts" digital information from an object.*
As highlighted above, we require that for all $x \in \mathbb{V}$ we have $\mathsf{Sen}(x) \neq \perp$ whereas for all $x \in \mathbb{X} \setminus \mathbb{V}$, we have $\mathsf{Sen}(x) = \perp$.

$R_{\mathbb{X}} : \mathbb{X} \times \mathbb{X} \to \{0,1\}$**:** *This is a relation between two objects for identifying whether two objects are identical. That is, $x, x' \in \mathbb{X}$ are considered the same objects if and only if $R_{\mathbb{X}}(x, x') = 1$. (We note that such a relation may not be efficiently computable in general, depending on a setting.)*
We require $R_{\mathbb{X}}(x, x) = 1$ for all $x \in \mathbb{X}$.

$R_{\mathbb{D}} : \mathbb{D} \times \mathbb{D} \to \{0,1\}$**:** *This is a relation between data.*
We require that this is computable by a deterministic PT algorithm. We also require that for all $x, x' \in \mathbb{V}$, we have $R_{\mathbb{X}}(x, x') = R_{\mathbb{D}}(\mathsf{Sen}(x), \mathsf{Sen}(x'))$.

$\mathbb{C}$**:** *This is the set of "command" functions, which models physical actions to objects. A command function may be a probabilistic function, and takes an object (or multiple objects[8]) as input, and outputs a new object $x' \in \mathbb{X} \cup \{\perp\}$ and some auxiliary (digial) information $z \in \{0,1\}^*$ about x and x'. Put differently, if $\mathsf{cmd} \in \mathbb{C}$, then $\mathsf{cmd} : \mathbb{X}^* \to (\mathbb{X} \cup \{\perp\}) \times \{0,1\}^*$.*
Note that Sen *can be naturally cast as a command function that takes an object $x \in \mathbb{X}$ as input, and outputs (no object and) data $D \in \mathbb{D}$ as auxiliary information of x, and with this interpretation, we require that $\mathbb{C}$ contain* Sen*. We also require that $\mathbb{C}$ contain the special command* Create *that, when invoked, generates and returns a new object $x \in \mathbb{V}$ by $x \leftarrow \mathcal{X}$ (and no auxiliary information).*

On the Relationship between $R_{\mathbb{X}}$ and $R_{\mathbb{D}}$. Note that we require that for all $x, x' \in \mathbb{V}$, we have $R_{\mathbb{X}}(x, x') = R_{\mathbb{D}}(\mathsf{Sen}(x), \mathsf{Sen}(x'))$. This might seem a somewhat too idealized condition, say if we think of a setting where objects are identified by some image recognition technology based on machine learning, which may not necessarily support error-less identification of two objects by their corresponding data. One possible interpretation of our treatment is that we implicitly assume the identification of objects using solely their corresponding data taken via Sen. For simplicity, we stick to the above treatment, but it will be interesting to investigate whether some relaxation for the relation between $R_{\mathbb{X}}$ and $R_{\mathbb{D}}$ can be introduced.

Examples. Here, we give some examples of an object setting. Consider a setting where electric chips are manufactured in a factory. Suppose the chips can be identified with some photographic images taken by a camera using some technology of image recognition (say, based on machine learning). Then, the set of valid objects $\mathbb{V}$ corresponds to the chips produced in the factory, and Sen corresponds to the camera for taking a photographic image. The set of all data $\mathbb{D}$ is

[8] We assume that the arity of an input is specified for each command function.

then photographic images that can be generated by the camera (i.e., Sen). $R_{\mathbb{X}}$ identifies two chips x and x' iff x and x' are the same chip, and $R_{\mathbb{D}}$ judges if they are the same in the images taken by Sen. $\mathcal{X}$ corresponds to the way a new chip is produced in the factory. $\mathbb{C}$ may contain any action that can be physically performed on a chip, say turning in some other direction, heating it up, and cutting it into two, etc. (Note that $\mathbb{C}$ could contain actions that destroy an object.)

In another example, where an electric chip admits a physically unclonable function (PUF) [11,23] that can be used for identifying two objects, Sen corresponds to obtaining the PUF-value of a given chip, and $R_{\mathbb{D}}$ will identify two objects to be identical if the given PUF-values are "close" (where the features of the PUF will determine the closeness). Other components will remain the same.

3.2 Algorithms that Can Interact with Physical Objects

As a preparation for defining SfO, we introduce two types of algorithms that can treat physical objects, a *sensing algorithm* and a *physically-enhanced algorithm*, which are both associated with an object setting $\mathcal{OS}$. Informally, a sensing algorithm models an algorithm that can extract digital data from a physical object via a sensing function Sen supported in $\mathcal{OS}$. On the other hand, a physically enhanced algorithm (PEA) models an algorithm that can indirectly interact with objects via command functions $\mathsf{cmd} \in \mathbb{C}$ supported in $\mathcal{OS}$.

Our main idea behind the formalization here is that interactions between algorithms and physical objects are done only via the sensing function (in the case of a sensing algorithm) or command functions (in the case of a PEA), and are conducted outside algorithms. Looking ahead, sensing algorithms are a class of algorithms to which the signing and verification algorithms for an SfO scheme belong, while PEAs are a class of algorithms to which an adversary against the security of an SfO scheme belongs.

The formal definitions of these notions are as follows.

Definition 4 (Sensing Algorithm and Physically-Enhanced Algorithm). *Let $\mathcal{OS} = (\mathbb{X}, \mathbb{V}, \mathcal{X}, \mathbb{D}, \mathsf{Sen}, R_{\mathbb{X}}, R_{\mathbb{D}}, \mathbb{C})$ be an object setting.*

- *A* sensing algorithm *with respect to $\mathcal{OS}$ is a PPT algorithm that has access to an object $x \in \mathbb{X}$ given as an input via the sensing function Sen with which data $D \in \mathbb{D}$ from the object x can be extracted.*
  In order to distinguish an object from ordinary (digital) inputs given as an input to a sensing algorithm, we will use the boxed notation. For example, if A is a sensing algorithm and is given an object $x \in \mathbb{X}$ as an input, we write $\mathsf{A}(\boxed{x})$.
- *A* physically-enhanced algorithm *(PEA) with respect to $\mathcal{OS}$ is a PPT algorithm that has access to the* command oracle $\mathcal{O}_{\mathbb{C}}$ *explained next.*
  The command oracle $\mathcal{O}_{\mathbb{C}}$ maintains a counter c (initially 0) and an ordered list $L_{\mathbb{X}}$ of objects (initially empty). $\mathcal{O}_{\mathbb{C}}$ accepts a "command" query consisting of (the name of) a command function $\mathsf{cmd} \in \mathbb{C}$ and optionally a set of indices $(i_1, \ldots, i_n) \in [c]^n$ (where n is the number of inputs that is specified by cmd).

Then, it sets $c \leftarrow c+1$*, computes* $(x_c, z_c) \leftarrow \mathsf{cmd}(x_{i_1}, \ldots, x_{i_n})$*, appends* x_c *to the end of the list* $L_{\mathbb{X}}$*, and returns* z_c *to the caller.*

We will omit "with respect to $\mathcal{OS}$" when it is clear from context.

3.3 Signature for Objects

Let $\mathcal{OS} = (\mathbb{X}, \mathbb{V}, \mathcal{X}, \mathbb{D}, \mathsf{Sen}, R_{\mathbb{X}}, R_{\mathbb{D}}, \mathbb{C})$ be an object setting. A signature for objects (SfO) scheme SfO with respect to $\mathcal{OS}$ consists of the following algorithms.

- $\mathsf{SfO.KG}(1^\lambda) \to (\mathsf{VK}, \mathsf{SK})$: This is a PPT algorithm for key generation. It takes the security parameter 1^λ as input, and outputs a key pair $(\mathsf{VK}, \mathsf{SK})$.
- $\mathsf{SfO.Sign}(\mathsf{SK}, \boxed{x}) \to \sigma$: This is a sensing algorithm for generating a signature. It takes as input a signing key SK and an object $x \in \mathbb{X}$ as input, and outputs a signature σ.
- $\mathsf{SfO.Ver}(\mathsf{VK}, \boxed{x}, \sigma) \to 1/0$: This is a sensing algorithm for verifying a signature. It takes a verification key VK, an object $x \in \mathbb{X}$, and a signature σ as input, and outputs 1 (accept) or 0 (reject).

As the correctness condition, we require that for all $\lambda \in \mathbb{N}$, $x, x' \in \mathbb{V}$ such that $R_{\mathbb{X}}(x, x') = 1$, $(\mathsf{VK}, \mathsf{SK}) \leftarrow \mathsf{SfO.KG}(1^\lambda)$, and $\sigma \leftarrow \mathsf{SfO.Sign}(\mathsf{SK}, \boxed{x})$, we have $\mathsf{SfO.Ver}(\mathsf{VK}, \boxed{x'}, \sigma) = 1$.

Basic Security Definition: Unforgeability. Here, we introduce a natural adoption of EUF-CMA security for ordinary signatures to the setting of SfO schemes, which we call *existential unforgeability under chosen-objects attacks* (EUF-COA security). Similarly to EUF-CMA security for an ordinary signature scheme, EUF-COA security guarantees that it is hard for any PEA adversary to forge a signature on objects for which they have never obtained signatures. Note that a PEA adversary may generate a new object via queries to the command oracle $\mathcal{O}_{\mathbb{C}}$, and it is allowed to obtain signatures for any objects in the object list $L_{\mathbb{X}}$ maintained in $\mathcal{O}_{\mathbb{C}}$.

Formally, the EUF-COA security for an SfO scheme is defined as follows. Let $\mathcal{OS} = (\mathbb{X}, \mathbb{V}, \mathcal{X}, \mathbb{D}, \mathsf{Sen}, R_{\mathbb{X}}, R_{\mathbb{D}}, \mathbb{C})$ be an object setting, and let $\mathsf{SfO} = (\mathsf{SfO.KG}, \mathsf{SfO.Sign}, \mathsf{SfO.Ver})$ be an SfO scheme with respect to $\mathcal{OS}$. Consider the following experiment $\mathsf{Expt}^{\mathsf{EUF\text{-}COA}}_{\mathsf{SfO},\mathcal{A}}(\lambda)$ in which a PEA adversary $\mathcal{A}$ is executed[9]:

$$\mathsf{Expt}^{\mathsf{EUF\text{-}COA}}_{\mathsf{SfO},\mathcal{A}}(\lambda): \left[\begin{array}{l} c \leftarrow 0; \quad L_{\mathbb{X}} \leftarrow \emptyset; \quad \mathsf{Ind} \leftarrow \emptyset; \\ (\mathsf{VK}, \mathsf{SK}) \leftarrow \mathsf{SfO.KG}(1^\lambda); \quad (i', \sigma') \leftarrow \mathcal{A}^{\mathcal{O}_{\mathbb{C}}, \mathcal{O}_{\mathsf{Sign}}}(\mathsf{VK}); \\ \text{If } \mathsf{SfO.Ver}(\mathsf{VK}, \boxed{x_{i'}}, \sigma') = 1 \wedge \forall j \in \mathsf{Ind} : R_{\mathbb{X}}(x_{i'}, x_j) = 0 \\ \text{then return 1 else return 0} \end{array}\right]$$

where $\mathcal{O}_{\mathbb{C}}$ is the command oracle for $\mathcal{A}$ defined in Definition 4 (which updates the counter c and the object list $L_{\mathbb{X}}$ upon a query from $\mathcal{A}$), and $\mathcal{O}_{\mathsf{Sign}}$ is the

[9] We remind the reader that as defined in Definition 4, a PEA has access to the command oracle $\mathcal{O}_{\mathbb{C}}$ that internally maintains the counter c and the object list $L_{\mathbb{X}}$, which we use for defining $\mathcal{A}$'s winning condition here.

$\mathsf{SfO.KG}(1^\lambda)$:	$\mathsf{SfO.Sign}(\mathsf{SK}, \boxed{x})$:	$\mathsf{SfO.Ver}(\mathsf{VK}, \boxed{x'}, \sigma)$:
$(\mathsf{VK}, \mathsf{SK}) \leftarrow \mathsf{DS.KG}(1^\lambda)$	$D \leftarrow \mathsf{Sen}(x)$	If $\sigma = \perp$ then return 0.
Return $(\mathsf{VK}, \mathsf{SK})$.	If $D = \perp$ then return $\sigma := \perp$.	$D' \leftarrow \mathsf{Sen}(x')$
	$\sigma_D \leftarrow \mathsf{DS.Sign}(\mathsf{SK}, D)$	Parse σ as (D, σ_D).
	Return $\sigma := (D, \sigma_D)$.	If $\perp \in \{D', D\}$ then return 0.
		If $\mathsf{DS.Ver}(\mathsf{VK}, D, \sigma_D) = 1$
		and $R_{\mathbb{D}}(D', D) = 1$
		then return 1 else return 0.

Fig. 2. The SfO scheme SfO_1.

signing oracle that takes an index $i \in [c]$ as input, and operates as follows: it updates Ind by $\mathsf{Ind} \leftarrow \mathsf{Ind} \cup \{i\}$, computes a signature $\sigma \leftarrow \mathsf{SfO.Sign}(\mathsf{SK}, \boxed{x_i})$, and returns σ.

Definition 5 (EUF-COA). *We say that an SfO scheme* SfO *with respect to an object setting* $\mathcal{OS}$ *is EUF-COA secure if for any PEA adversary* $\mathcal{A}$*, we have* $\mathsf{Adv}^{\mathsf{EUF\text{-}COA}}_{\mathsf{SfO},\mathcal{A}}(\lambda) := \Pr[\mathsf{Expt}^{\mathsf{EUF\text{-}COA}}_{\mathsf{SfO},\mathcal{A}}(\lambda) = 1] = \mathrm{negl}(\lambda)$.

On Security of Ordinary (Non-physical) Cryptographic Primitives against PEA Adversaries. In this paper, we will construct an SfO scheme using ordinary (non-physical) cryptographic primitives as building blocks, and in the security proof, we would like to reduce the security of the proposed signature for objects schemes to that of the building blocks. However, one can quickly realize that there is a subtle technical problem: Although the security of an SfO scheme is defined with respect to PEA adversaries, that of ordinary cryptographic primitives is defined with respect to standard PPT adversaries that cannot deal with physical objects, and thus there is a mismatch regarding the class of adversaries. To circumvent this subtle problem, we simply assume that the ordinary cryptographic primitives used as building blocks are secure against PEA adversaries. We believe that this is a reasonable assumption, since ordinary cryptographic primitives are defined (and their security are proved) independently of any object setting. (It will be a ground-breaking finding if there is a cryptographic primitive whose security is defined with respect to PPT adversaries can be attacked if an adversary can perform some physical action.)

3.4 Basic Construction

Here we show our first construction SfO_1, which is constructed by simply combining an ordinary digital signature scheme DS and the relation function $R_{\mathbb{D}}$ supported in an underlying object setting $\mathcal{OS}$.

$\mathcal{OS} = (\mathbb{X}, \mathbb{V}, \mathcal{X}, \mathbb{D}, \mathsf{Sen}, R_{\mathbb{X}}, R_{\mathbb{D}}, \mathbb{C})$ be an object setting. Let $\mathsf{DS} = (\mathsf{DS.KG}, \mathsf{DS.Sign}, \mathsf{DS.Ver})$ be an ordinary digital signature scheme. Then, using DS as a building block, we construct an SfO scheme SfO_1 as described in Fig. 2.

It is straightforward to see that SfO_1 satisfies correctness. By the property of $R_{\mathbb{X}}$ and $R_{\mathbb{D}}$, if $x \in \mathbb{V}$ used in signing and $x' \in \mathbb{V}$ used in verification satisfy $R_{\mathbb{X}}(x', x) = 1$, we have $R_{\mathbb{D}}(D', D) = 1$, where $D \leftarrow \mathsf{Sen}(x)$ and $D' \leftarrow \mathsf{Sen}(x')$. As long as the digital signature scheme DS satisfies correctness, we always have $\mathsf{DS.Ver}(\mathsf{VK}, D, \sigma_D) = 1$, where $(\mathsf{VK}, \mathsf{SK}) \leftarrow \mathsf{DS.KG}(1^\lambda)$ and $\sigma_D \leftarrow \mathsf{DS.Sign}(\mathsf{SK}, D)$.

The security of SfO_1 is guaranteed by the following theorem, which is also straightforward to prove.

Theorem 6. *If the digital signature scheme* DS *is EUF-CMA secure against PEA adversaries with respect to* $\mathcal{OS}$*, then* SfO_1 *is EUF-COA secure.*

We defer the formal proof of this theorem to Appendix A. Informally, if a PEA adversary $\mathcal{A}$ attacking the EUF-COA security wins, then it outputs a valid signature for data extracted from unqueried objects. Therefore, we can construct another adversary attacking the EUF-CMA security of DS using $\mathcal{A}$.

4 Conf-COA Security and Construction

In this section, we introduce a security definition for SfO concerning privacy of objects that we call *confidentiality under chosen object attacks* (Conf-COA security). This security notion ensures that given a signature on an object that is generated according to the distribution $\mathcal{X}$ supported in an object setting, it is hard to gain any information on the data corresponding to the object. This security notion is naturally desirable in a supply chain scenario where an object being signed is a product of a company which itself and/or its corresponding data could contain some confidential information, and a signer would like to prevent its information from leaking from a signature to those who are outside the supply chain and need not verify the signature. We note that this security notion is orthogonal to EUF-COA security.

This section provides the security definition of Conf-COA in Section 4.1 and a provably secure construction that satisfies EUF-COA and Conf-COA using an obfuscation in Section 4.2. Moreover, we consider an instantiation of our scheme in Section 4.3.

4.1 Security Definition: Conf-COA

Here, we give a formal definition of Conf-COA security.

Let $\mathcal{OS} = (\mathbb{X}, \mathbb{V}, \mathcal{X}, \mathbb{D}, \mathsf{Sen}, R_{\mathbb{X}}, R_{\mathbb{D}}, \mathbb{C})$ be an object setting, and let $\mathsf{SfO} = (\mathsf{SfO.KG}, \mathsf{SfO.Sign}, \mathsf{SfO.Ver})$ be an SfO scheme with respect to $\mathcal{OS}$. For PEA algorithms $\mathcal{A}$, $\mathcal{S}$, and a predicate $\mathcal{P}$, consider the real experiment $\mathsf{Expt}^{\mathsf{Conf\text{-}COA\text{-}Real}}_{\mathsf{SfO},\mathcal{A},\mathcal{P}}(\lambda)$ and the simulated experiment $\mathsf{Expt}^{\mathsf{Conf\text{-}COA\text{-}Sim}}_{\mathsf{SfO},\mathcal{S},\mathcal{P}}(\lambda)$ as described in Fig. 3. In the experiments, where $\mathcal{O}_{\mathbb{C}}$ and $\mathcal{O}_{\mathsf{Sign}}$ are the command oracle and signing oracle, respectively, that are defined in the same way as in the EUF-CMA experiment. We stress that the "challenge object" x^* is not included in the object list $L_{\mathbb{X}}$ maintained in $\mathcal{O}_{\mathbb{C}}$ in both experiments, and thus $\mathcal{A}$ and $\mathcal{S}$ have no control over it as well as the data D^* extracted from x^*.

$\mathsf{Expt}^{\mathsf{Conf\text{-}COA\text{-}Real}}_{\mathsf{SfO},\mathcal{A},\mathcal{P}}(\lambda)$:
$c \leftarrow 0$; $L_{\mathbb{X}} \leftarrow \emptyset$
$x^* \leftarrow \mathcal{X}$; $D^* \leftarrow \mathsf{Sen}(x^*)$
$(\mathsf{VK},\mathsf{SK}) \leftarrow \mathsf{SfO.KG}(1^\lambda)$
$\sigma^* \leftarrow \mathsf{SfO.Sign}(\mathsf{SK}, \boxed{x^*})$
$g \leftarrow \mathcal{A}^{\mathcal{O}_{\mathbb{C}},\mathcal{O}_{\mathsf{Sign}}}(\mathsf{VK},\sigma^*)$
Return $(\mathcal{P}(D^*) \stackrel{?}{=} g)$.

$\mathsf{Expt}^{\mathsf{Conf\text{-}COA\text{-}Sim}}_{\mathsf{SfO},\mathcal{S},\mathcal{P}}(\lambda)$:
$c \leftarrow 0$; $L_{\mathbb{X}} \leftarrow \emptyset$
$x^* \leftarrow \mathcal{X}$; $D^* \leftarrow \mathsf{Sen}(x^*)$
$g \leftarrow \mathcal{S}^{\mathcal{O}_{\mathbb{C}}}(1^\lambda)$
Return $(\mathcal{P}(D^*) \stackrel{?}{=} g)$.

Fig. 3. The experiments for defining Conf-COA security for an SfO scheme SfO.

$\mathsf{SfO.KG}(1^\lambda)$:	$\mathsf{SfO.Sign}(\mathsf{SK}, \boxed{x})$:	$\mathsf{SfO.Ver}(\mathsf{VK}, \boxed{x'}, \sigma)$:
$(\mathsf{VK},\mathsf{SK}) \leftarrow \mathsf{DS.KG}(1^\lambda)$	$D \leftarrow \mathsf{Sen}(x)$	If $\sigma = \bot$ then return 0.
Return $(\mathsf{VK},\mathsf{SK})$.	If $D = \bot$ then return $\sigma = \bot$.	$D' \leftarrow \mathsf{Sen}(x')$
	Let $R_D(\cdot) := R_{\mathbb{D}}(\cdot, D)$.	Parse σ as $(\widetilde{R}, \sigma_{\widetilde{R}})$.
	$\widetilde{R} \leftarrow \mathsf{Obf}(1^\lambda, R_D)$	If $\bot \in \{D', \widetilde{R}\}$ then return 0.
	$\sigma_{\widetilde{R}} \leftarrow \mathsf{DS.Sign}(\mathsf{SK}, \widetilde{R})$	If $\mathsf{DS.Ver}(\mathsf{VK}, \widetilde{R}, \sigma_{\widetilde{R}}) = 1$
	Return $\sigma := (\widetilde{R}, \sigma_{\widetilde{R}})$.	and $\widetilde{R}(D') = 1$
		then return 1 else return 0.

Fig. 4. The SfO scheme SfO_2.

Definition 7 (Conf-COA). *We say that an SfO scheme* SfO *with respect to an object setting* $\mathcal{OS}$ *is Conf-COA secure if for any PEA adversary* $\mathcal{A}$*, there exists a PEA simulator* $\mathcal{S}$ *such that for any PPT-computable predicate* $\mathcal{P}$*, we have* $\mathsf{Adv}^{\mathsf{Conf\text{-}COA}}_{\mathsf{SfO},\mathcal{A},\mathcal{S},\mathcal{P}}(\lambda) := |\Pr[\mathsf{Expt}^{\mathsf{Conf\text{-}COA\text{-}Real}}_{\mathsf{SfO},\mathcal{A},\mathcal{P}}(\lambda) = 1] - \Pr[\mathsf{Expt}^{\mathsf{Conf\text{-}COA\text{-}Sim}}_{\mathsf{SfO},\mathcal{S},\mathcal{P}}(\lambda) = 1]| = \mathrm{negl}(\lambda)$.

4.2 Our Construction Based on Obfuscation

Here we show our second construction SfO_2. This is a simple variant of our first construction SfO_1, where instead of directly signing the data D in the signing algorithm, we now sign an obfuscated circuit $\widehat{R} \leftarrow \mathsf{Obf}(1^\lambda, R_{\mathbb{D}}(\cdot, D))$, and the relation $R_{\mathbb{D}}$ over the data D contained in a signature and D' computed in the verification is now done using an obfuscated circuit $\widetilde{R}$.

Formally, our construction is as follows. Let $\mathcal{OS} = (\mathbb{X}, \mathbb{V}, \mathcal{X}, \mathbb{D}, \mathsf{Sen}, R_{\mathbb{X}}, R_{\mathbb{D}}, \mathbb{C})$ be an object setting. Let $\mathcal{R}$ be the class of circuits $\{R_D(\cdot) := R_{\mathbb{D}}(\cdot, D)\}_{D \in \mathbb{D}}$, where $R_{\mathbb{D}}(\cdot, D)$ denotes a circuit which has D hardwired, and takes some data D' as input, and returns $R_{\mathbb{D}}(D', D)$. We assume that there is a one-to-one correspondence between a circuit $R_D \in \mathcal{R}$ and $D \in \mathbb{D}$, and D can be extracted in the clear from R_D. Let Obf be an obfuscator for $\mathcal{R}$, and let $\mathsf{DS} = (\mathsf{DS.KG}, \mathsf{DS.Sign}, \mathsf{DS.Ver})$ be an ordinary digital signature scheme. Then, using Obf and DS as building blocks, we construct an SfO scheme SfO_2 as described in Fig. 4.

The correctness of SfO_2 can be seen similarly to SfO_1. As explained earlier, the only essential difference of SfO_2 from SfO_1 is that in the former, the check of $R_{\mathbb{D}}(D', D)$ is done using the obfuscated circuit $\widetilde{R}$ which is computed as $\widetilde{R} \leftarrow$

$\mathsf{Obf}(1^\lambda, R_D)$ and $R_D(\cdot) := R_{\mathbb{D}}(\cdot, D)$. Then the correctness of Obf ensures that if $R_{\mathbb{D}}(D', D) = 1$ then $\widetilde{R}(D') = 1$. The rest is unchanged from SfO_1.

We now show how the EUF-COA and Conf-COA security of SfO_2 can be established.

Theorem 8. *If the digital signature scheme* DS *is EUF-CMA secure against PEA adversaries, then* SfO_2 *is EUF-COA secure.*

We defer the formal proof of this theorem to Appendix B. Informally, if a PEA adversary $\mathcal{A}$ attacking the EUF-COA security wins, then it outputs the index i^* of an object x_{i^*} and a valid signature for an obfuscated circuit $\widetilde{R}$ such that $\widetilde{R}(D^*) = 1$ where $D^* \leftarrow \mathsf{Sen}(x_{i^*})$. From the winning condition for $\mathcal{A}$, we have $\widetilde{R_j}(D^*) = 0$ for all $j \in \mathsf{Ind}$, which means that $\mathcal{A}$ does not know any signature of $\widetilde{R}$ that it obtained via signing queries. Therefore, we can construct another adversary attacking the EUF-CMA security of DS using $\mathcal{A}$.

For the Conf-COA security of SfO_2, we need the property that the distribution of the data taken from a newly generated object via Sen, namely $\{x \leftarrow \mathcal{X}; D \leftarrow \mathsf{Sen}(x) : D\}$, has sufficient amount of entropy, so that for any data $D' \in \mathbb{D}$, the probability that $R_{\mathbb{D}}(D, D') = 1$ occurs is sufficiently small. Following [2,10], we call such a property of a distribution *evasive*, and define it as a property of an object setting.

Definition 9 (Evasiveness). *We say that an object setting* $\mathcal{OS} = (\mathbb{X}, \mathbb{V}, \mathcal{X}, \mathbb{D}, \mathsf{Sen}, R_{\mathbb{X}}, R_{\mathbb{D}}, \mathbb{C})$ *is* ϵ-evasive *if for any* $D' \in \mathbb{D}$, $\Pr[x \leftarrow \mathcal{X}; D \leftarrow \mathsf{Sen}(x) : R_{\mathbb{D}}(D, D') = 1] \leq \epsilon$.

Theorem 10. *If the obfuscator* Obf *satisfies distributional VBB security for the distribution* $\{x \leftarrow \mathcal{X}; D \leftarrow \mathsf{Sen}(x) : D\}$ *against PEA adversaries*[10]*, and the distribution* $\{x \leftarrow \mathcal{X}; D \leftarrow \mathsf{Sen}(x) : D\}$ *is* ϵ*-evasive for some negligible* $\epsilon = \epsilon(\lambda)$*, then* SfO_2 *is Conf-COA secure.*

The formal proof of this theorem is given in Appendix C. We briefly give a proof sketch. From a Conf-COA adversary $\mathcal{A}$, we construct an adversary $\mathcal{B}$ against the distributional VBB security of Obf. $\mathcal{B}$ is initially given as input an obfuscated program $\widetilde{R}^*$ that is generated as $x^* \leftarrow \mathcal{X}$, $D^* \leftarrow \mathsf{Sen}(x^*)$, and $\widetilde{R}^* \leftarrow \mathsf{Obf}(1^\lambda, R_{D^*})$, and perfectly simulates the real Conf-COA experiment for $\mathcal{A}$. Due to the distributional VBB security of Obf, there exists a simulator $\mathcal{S}_{\mathcal{B}}$ such that its output distribution is negligibly close to that of $\mathcal{B}$ (and hence that of $\mathcal{A}$). Then, from $\mathcal{S}_{\mathcal{B}}$, we construct a simulator $\mathcal{S}$ for $\mathcal{A}$. The $\mathcal{S}_{\mathcal{B}}$ is a simulator and can submit a query to the circuit $R_{D^*}(\cdot)$, but the evasiveness of the distribution allows $\mathcal{S}$ to answer it to always 0, which ensures that the distribution of the output of $\mathcal{S}_{\mathcal{B}}$ and that of $\mathcal{S}$ are negligibly close. Then, combining all the arguments, we can conclude that the output distribution of $\mathcal{A}$ is negligibly close to that of $\mathcal{S}$.

[10] When we consider distributional VBB security for an obfuscator against PEA adversaries, we consider not only an adversary but also a simulator to be PEA algorithms.

4.3 Instantiation

Unfortunately, there does not exist an obfuscator that can obfuscate all polynomial-sized circuits with (worst-case) VBB security [3]. However, this impossibility result does not rule out the existence of an obfuscator for a particular class of circuits, in particular the one considered in the previous subsection.[11]

For example, Galbraith and Zobernig [10] showed a distributionally VBB secure obfuscator for "fuzzy Hamming distance" predicates (from the hardness of so-called the decisional distributional modular subset product problem). Here, a fuzzy Hamming distance predicate is a circuit $R_D(\cdot)$ such that $R_D(D') = 1$ iff the Hamming distance of the input D' is close (within some pre-determined distance r) to the value D that is hardcorded in the circuit R_D. By using their obfuscator, our construction SfO_2 yields a Conf-COA secure SfO scheme for an object setting where the relation $R_{\mathbb{D}}$ just tests the closeness of the inputs by the Hamming distance.

We do not claim this scheme can be used in a realistic scenario (such as a supply-chain scenario), but hopefully techniques of obfuscation will be developed and more complicated classes of circuits can be obfuscated with distributionally VBB security in the future, so that our Conf-COA secure construction can be instantiated for a wider class of object settings.

Acknowledgement

The authors would like to thank the anonymous referees for their valuable comments and helpful suggestions. This work was partially supported by JSPS KAKENHI Grant Number JP19H01109, JST AIP Acceleration Research JPMJCR22U5, and JST CREST Grant Number JPMJCR22M1, Japan.

References

1. Havocscope. `https://havocscope.com/`. Accessed: 2022-09-22.
2. Boaz Barak, Nir Bitansky, Ran Canetti, Yael Tauman Kalai, Omer Paneth, and Amit Sahai. Obfuscation for evasive functions. In Yehuda Lindell, editor, *TCC 2014*, volume 8349 of *LNCS*, pages 26–51. Springer, Heidelberg, February 2014.
3. Boaz Barak, Oded Goldreich, Russell Impagliazzo, Steven Rudich, Amit Sahai, Salil P. Vadhan, and Ke Yang. On the (im)possibility of obfuscating programs. In Joe Kilian, editor, *CRYPTO 2001*, volume 2139 of *LNCS*, pages 1–18. Springer, Heidelberg, August 2001.

[11] Another possibility to bypass the impossibility result of [3] is to resort to an idealized model: Brakerski and Rothblum [5] showed that we can construct a (worst-case, and hence distributionally) VBB secure obfuscator for circuits in the generic graded encoding model. Unfortunately, as far as we know, all candidates of graded encodings are broken and hence cannot be used to securely instantiate the obfuscator of [5].

4. George Baryannis, Samir Dani, and Grigoris Antoniou. Predicting supply chain risks using machine learning: The trade-off between performance and interpretability. *Future Generation Computer Systems*, 101:993–1004, 2019.
5. Zvika Brakerski and Guy N. Rothblum. Virtual black-box obfuscation for all circuits via generic graded encoding. In Yehuda Lindell, editor, *TCC 2014*, volume 8349 of *LNCS*, pages 1–25. Springer, Heidelberg, February 2014.
6. Nishanth Chandran, Vipul Goyal, Ryan Moriarty, and Rafail Ostrovsky. Position based cryptography. In Shai Halevi, editor, *CRYPTO 2009*, volume 5677 of *LNCS*, pages 391–407. Springer, Heidelberg, August 2009.
7. Alexander W. Dent, Marc Fischlin, Mark Manulis, Martijn Stam, and Dominique Schröder. Confidential signatures and deterministic signcryption. In Phong Q. Nguyen and David Pointcheval, editors, *PKC 2010*, volume 6056 of *LNCS*, pages 462–479. Springer, Heidelberg, May 2010.
8. Ben Fisch, Daniel Freund, and Moni Naor. Physical zero-knowledge proofs of physical properties. In Juan A. Garay and Rosario Gennaro, editors, *CRYPTO 2014, Part II*, volume 8617 of *LNCS*, pages 313–336. Springer, Heidelberg, August 2014.
9. Nils Fleischhacker, Felix Günther, Franziskus Kiefer, Mark Manulis, and Bertram Poettering. Pseudorandom signatures. In Kefei Chen, Qi Xie, Weidong Qiu, Ninghui Li, and Wen-Guey Tzeng, editors, *ASIACCS 13*, pages 107–118. ACM Press, May 2013.
10. Steven D. Galbraith and Lukas Zobernig. Obfuscated fuzzy hamming distance and conjunctions from subset product problems. In Dennis Hofheinz and Alon Rosen, editors, *TCC 2019, Part I*, volume 11891 of *LNCS*, pages 81–110. Springer, Heidelberg, December 2019.
11. Blaise Gassend, Dwaine E. Clarke, Marten van Dijk, and Srinivas Devadas. Silicon physical random functions. In Vijayalakshmi Atluri, editor, *ACM CCS 2002*, pages 148–160. ACM Press, November 2002.
12. Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *Journal of Computer and System Sciences*, 28(2):270–299, 1984.
13. Kaiming He, Georgia Gkioxari, Piotr Dollar, and Ross Girshick. Mask R-CNN. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Oct 2017.
14. Yuval Ishai, Eyal Kushilevitz, Rafail Ostrovsky, and Amit Sahai. Cryptographic sensing. In Alexandra Boldyreva and Daniele Micciancio, editors, *CRYPTO 2019, Part III*, volume 11694 of *LNCS*, pages 583–604. Springer, Heidelberg, August 2019.
15. Nir Kshetri and Elena Loukoianova. Blockchain adoption in supply chain networks in asia. *IT Professional*, 21(1):11–15, 2019.
16. Hau L. Lee and Seungjin Whang. Higher supply chain security with lower cost: Lessons from total quality management. *International Journal of Production Economics*, 96(3):289–300, 2005. Quality in Supply Chain Management and Logistics.
17. Tsung-Yi Lin, Piotr Dollar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, July 2017.
18. Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. SSD: Single shot multibox detector. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Computer Vision – ECCV 2016*, pages 21–37, Cham, 2016. Springer International Publishing.

19. Hemanta K. Maji, Manoj Prabhakaran, and Mike Rosulek. Attribute-based signatures. In Aggelos Kiayias, editor, *CT-RSA 2011*, volume 6558 of *LNCS*, pages 376–392. Springer, Heidelberg, February 2011.
20. Takaaki Mizuki and Hiroki Shizuya. A formalization of card-based cryptographic protocols via abstract machine. *International Journal of Information Security*, 13(1):15–23, Feb 2014.
21. Javid Moosavi, Leila M. Naeni, Amir M. Fathollahi-Fard, and Ugo Fiore. Blockchain in supply chain management: a review, bibliometric, and network analysis. *Environmental Science and Pollution Research*, Feb 2021.
22. OECD/EUIPO. *Global Trade in Fakes: A Worrying Threat*. Illicit Trade. OECD Publishing, Paris, Jun. 2021.
23. Ravikanth Pappu, Ben Recht, Jason Taylor, and Neil Gershenfeld. Physical one-way functions. *Science*, 297(5589):2026–2030, 2002.
24. Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 779–788, 2016.
25. Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 28. Curran Associates, Inc., 2015.
26. Jordan Robertson and Michael Riley. The Big Hack: How China Used a Tiny Chip to Infiltrate U.S. Companies. `https://www.bloomberg.com/news/features/2018-10-04/the-big-hack-how-china-used-a-tiny-chip-to-infiltrate-america-s-top-companies`, Bloomberg, Oct. 2018. Accessed: 2022-09-22.
27. Sara Saberi, Mahtab Kouhizadeh, Joseph Sarkis, and Lejia Shen. Blockchain technology and its relationships to sustainable supply chain management. *International Journal of Production Research*, 57(7):2117–2135, 2019.
28. Homeland Security. Intellectual Property Rights Seizure Statistics — Fiscal Year 2019. `https://www.iprcenter.gov/file-repository/trade-fy2017-ipr-seizures.pdf/view`, 2019. Accessed: 2022-09-22.
29. Rohit Sharma, Sachin S. Kamble, Angappa Gunasekaran, Vikas Kumar, and Anil Kumar. A systematic literature review on machine learning applications for sustainable agriculture supply chain performance. *Computers & Operations Research*, 119:104926, 2020.
30. He Sun, Kun Sun, Yuewu Wang, and Jiwu Jing. TrustOTP: Transforming smartphones into secure one-time password tokens. In Indrajit Ray, Ninghui Li, and Christopher Kruegel, editors, *ACM CCS 2015*, pages 976–988. ACM Press, October 2015.
31. Kenta Takahashi, Takahiro Matsuda, Takao Murakami, Goichiro Hanaoka, and Masakatsu Nishigaki. A signature scheme with a fuzzy private key. In Tal Malkin, Vladimir Kolesnikov, Allison Bishop Lewko, and Michalis Polychronakis, editors, *ACNS 15*, volume 9092 of *LNCS*, pages 105–126. Springer, Heidelberg, June 2015.
32. Erfan Babaee Tirkolaee, Saeid Sadeghi, Farzaneh Mansoori Mooseloo, Hadi Rezaei Vandchali, and Samira Aeini. Application of machine learning in supply chain management: A comprehensive overview of the main areas. *Mathematical Problems in Engineering*, 2021:1476043, Jun 2021.
33. Piyi Yang, Zhenfu Cao, and Xiaolei Dong. Fuzzy identity based signature. Cryptology ePrint Archive, Paper 2008/002, 2008. `https://eprint.iacr.org/2008/002`.

Appendix

A Proof of Theorem 1

Proof. Let $\mathcal{A}$ be a PEA adversary attacking the EUF-COA security of SfO_1. We construct another PEA adversary $\mathcal{B}$ against the EUF-CMA security of the underlying digital signature scheme DS with the same success probability as $\mathcal{A}$, which implies the theorem. The description of $\mathcal{B}$ is as follows.

$\mathcal{B}$ initially receives a verification key VK in its EUF-CMA experiment regarding DS. Then, $\mathcal{B}$ runs $\mathcal{A}(\mathsf{VK})$, and responds to each of $\mathcal{A}$'s queries as follows:

- For each command query from $\mathcal{A}$, $\mathcal{B}$ forwards the query to $\mathcal{B}$'s own command oracle $\mathcal{O}_{\mathbb{C}}$, and returns the result from $\mathcal{O}_{\mathbb{C}}$ back to $\mathcal{A}$. (Note that this causes an update of the counter c and the list $L_{\mathbb{X}}$ maintained in $\mathcal{O}_{\mathbb{C}}$.)
- For each signing query $i \in [c]$ from $\mathcal{A}$ (where c denotes the current counter maintained by $\mathcal{O}_{\mathbb{C}}$), $\mathcal{B}$ makes a command query asking $\mathcal{B}$'s oracle $\mathcal{O}_{\mathbb{C}}$ to use Sen on x_i (which must have been generated as a result of $\mathcal{A}$'s command queries made so far and contained in $L_{\mathbb{X}}$ in $\mathcal{O}_{\mathbb{C}}$), and receives $D_i \leftarrow \mathsf{Sen}(x_i)$ from $\mathcal{O}_{\mathbb{C}}$. $\mathcal{B}$ now proceeds as follows:
  - If $D_i = \bot$, then $\mathcal{B}$ returns $\sigma := \bot$ to $\mathcal{A}$.
  - Otherwise, $\mathcal{B}$ sends D_i to its signing oracle to obtain a signature σ_{D_i}. Then, $\mathcal{B}$ returns $\sigma := (D_i, \sigma_{D_i})$ to $\mathcal{A}$.

  When $\mathcal{A}$ outputs a pair $(i^*, \sigma^* = (D^*, \sigma^*_{D^*}))$ and terminates, $\mathcal{B}$ outputs $(D^*, \sigma^*_{D^*})$ as its own message/signature pair for DS and terminates.

The above completes the description of $\mathcal{B}$. It is straightforward to see that $\mathcal{B}$ simulates the EUF-COA experiment perfectly for $\mathcal{A}$. If $\mathcal{A}$'s output satisfies the winning condition of the EUF-COA experiment, we have $\mathsf{DS.Ver}(\mathsf{VK}, D^*, \sigma^*_{D^*}) = 1$ and $R_{\mathbb{D}}(D'^*, D^*) = 1$ where $D'^* \leftarrow \mathsf{Sen}(x_{i^*})$, and $R_{\mathbb{X}}(x_{i^*}, x_j) = 0$ for all $j \in \mathsf{Ind}$, where Ind denotes the set of signing queries submitted by $\mathcal{A}$. Then, the properties of $R_{\mathbb{X}}$ and $R_{\mathbb{D}}$ imply that either $D_j = \bot$ or $R_{\mathbb{D}}(D'^*, D_j) = 0$ holds for all $j \in \mathsf{Ind}$ as well, and together with the condition $R_{\mathbb{D}}(D'^*, D^*) = 1$, we must have $D^* \notin \{D_j\}_{j \in \mathsf{Ind}} \setminus \{\bot\}$. Note that the set of signing queries Msg made by $\mathcal{B}$ is exactly $\{D_j\}_{j \in \mathsf{Ind}} \setminus \{\bot\}$. These imply that $\mathcal{B}$ has not submitted D^* as a signing query. Hence, whenever $\mathcal{A}$'s output satisfies the winning condition of the EUF-COA experiment, $\mathcal{B}$'s output also satisfies the winning condition of the EUF-CMA experiment. Putting everything together, we can conclude that $\mathsf{Adv}^{\mathsf{EUF\text{-}CMA}}_{\mathsf{DS},\mathcal{B}}(\lambda) = \mathsf{Adv}^{\mathsf{EUF\text{-}COA}}_{\mathsf{SfO}_1,\mathcal{A}}(\lambda)$, as required. ■

B Proof of Theorem 2

Proof. Let $\mathcal{A}$ be a PEA adversary attacking the EUF-COA security of SfO_2. We construct another PEA adversary $\mathcal{B}$ against the EUF-CMA security of the underlying digital signature scheme DS whose success probability is the same as that of $\mathcal{A}$, which implies the theorem. The description of $\mathcal{B}$ is as follows.

$\mathcal{B}$ initially receives a verification key VK in its EUF-CMA experiment regarding DS. Then, $\mathcal{B}$ runs $\mathcal{A}(\mathsf{VK})$, and responds to each of $\mathcal{A}$'s queries as follows:

- For each command query from $\mathcal{A}$, $\mathcal{B}$ forwards the query to $\mathcal{B}$'s own command oracle $\mathcal{O}_{\mathbb{C}}$, and returns the result from $\mathcal{O}_{\mathbb{C}}$ back to $\mathcal{A}$. (Note that this causes an update of the counter c and the list $L_{\mathbb{X}}$ maintained in $\mathcal{O}_{\mathbb{C}}$.)
- For each signing query $i \in [c]$ from $\mathcal{A}$ (where c denotes the current counter maintained by $\mathcal{O}_{\mathbb{C}}$), $\mathcal{B}$ makes a command query asking $\mathcal{B}$'s oracle $\mathcal{O}_{\mathbb{C}}$ to use Sen on x_i (which must have been generated as a result of $\mathcal{A}$'s command queries made so far and contained in $L_{\mathbb{X}}$ in $\mathcal{O}_{\mathbb{C}}$), and receives $D_i \leftarrow \mathsf{Sen}(x_i)$ from $\mathcal{O}_{\mathbb{C}}$. $\mathcal{B}$ now proceeds as follows:
  - If $D_i = \perp$ then $\mathcal{B}$ returns $\sigma := \perp$ to $\mathcal{A}$.
  - Otherwise, $\mathcal{B}$ computes $\widetilde{R}_i \leftarrow \mathsf{Obf}(1^\lambda, R_{D_i})$, sends $\widetilde{R}_i$ to its signing oracle to obtain a signature $\sigma_{\widetilde{R}_i}$. Finally, $\mathcal{B}$ returns $\sigma := (\widetilde{R}_i, \sigma_{\widetilde{R}_i})$ to $\mathcal{A}$.

  When $\mathcal{A}$ outputs a pair $(i^*, \sigma^* = (\widetilde{R}^*, \sigma^*_{\widetilde{R}^*}))$ and terminates, $\mathcal{B}$ outputs $(\widetilde{R}^*, \sigma^*_{\widetilde{R}^*})$ as its own message/signature pair for DS and terminates.

The above completes the description of $\mathcal{B}$. It is straightforward to see that $\mathcal{B}$ simulates the EUF-COA experiment perfectly for $\mathcal{A}$. If $\mathcal{A}$'s output satisfies the winning condition of the EUF-COA experiment, we have $\mathsf{DS.Ver}(\mathsf{VK}, \widetilde{R}^*, \sigma^*_{\widetilde{R}^*}) = 1$ and $\widetilde{R}^*(D^*) = 1$ where $D^* \leftarrow \mathsf{Sen}(x_{i^*})$, and $R_{\mathbb{X}}(x_{i^*}, x_j) = 0$ for all $j \in \mathsf{Ind}$, where Ind denotes the set of signing queries submitted by $\mathcal{A}$. The properties of $R_{\mathbb{X}}$ and $R_{\mathbb{D}}$ imply that either $D_j = \perp$ or $R_{\mathbb{D}}(D^*, D_j) = 0$ holds for all $j \in \mathsf{Ind}$ as well. Note that the set of signing queries Msg made by $\mathcal{B}$ is $\{\widetilde{R}_j\}_{j \in \mathsf{Ind}} \setminus \{\perp\}$, where for convenience we interpret $\widetilde{R}_j = \perp$ when $D_j = \perp$. We now show that there does not exist $j' \in \mathsf{Ind}$ at which $\widetilde{R}^* = \widetilde{R}_{j'}$ holds. Let us assume that such j' exists for the sake of contradiction. Then, for this j', we have $\widetilde{R}_{j'}(D^*) = 1$, which in turn implies $R_{\mathbb{D}}(D^*, D_{j'}) = 1$. The properties of $R_{\mathbb{X}}$ and $R_{\mathbb{D}}$ imply that $R_{\mathbb{X}}(x_{i^*}, x_{j'}) = 1$, but this contradicts the winning condition of $\mathcal{A}$. Hence, there does not exist $j' \in \mathsf{Ind}$ such that $\widetilde{R}^* = \widetilde{R}_{j'}$, and thus $\mathcal{B}$ has not submitted $\widetilde{R}^*$ as a signing query. Therefore, whenever $\mathcal{A}$'s output satisfies the winning condition of the EUF-COA experiment for SfO_2, $\mathcal{B}$'s output satisfies the winning condition of the EUF-CMA experiment for DS. In conclusion, we have $\mathsf{Adv}^{\mathsf{EUF\text{-}CMA}}_{\mathsf{DS},\mathcal{B}}(\lambda) = \mathsf{Adv}^{\mathsf{EUF\text{-}COA}}_{\mathsf{SfO}_2,\mathcal{A}}(\lambda)$, as required. ■

C Proof of Theorem 3

Proof. Let $\mathcal{A}$ be an arbitrary PEA adversary attacking the Conf-COA security of SfO_2. We will construct a PEA simulator $\mathcal{S}$, and then show that for any PPT predicate $\mathcal{P}$, we have

$$|\Pr[\mathsf{Expt}^{\mathsf{Conf\text{-}COA\text{-}Real}}_{\mathsf{SfO}_2,\mathcal{A},\mathcal{P}}(\lambda) = 1] - \Pr[\mathsf{Expt}^{\mathsf{Conf\text{-}Conf\text{-}Sim}}_{\mathsf{SfO}_2,\mathcal{S},\mathcal{P}}(\lambda) = 1]| = \mathrm{negl}(\lambda). \quad (1)$$

For showing such a simulator $\mathcal{S}$ for $\mathcal{A}$, we first consider the following PEA adversary $\mathcal{B}$ that attacks the distributional VBB security of Obf:

$\mathcal{B}$ is initially given as input an obfuscated program $\widetilde{R}^*$ that is generated as $x^* \leftarrow \mathcal{X}$, $D^* \leftarrow \mathsf{Sen}(x^*)$, and $\widetilde{R}^* \leftarrow \mathsf{Obf}(1^\lambda, R_{D^*})$. Then, $\mathcal{B}$ runs $(\mathsf{VK}, \mathsf{SK}) \leftarrow$

$\mathsf{DS.KG}(1^\lambda)$ and $\sigma^*_{\widetilde{R}^*} \leftarrow \mathsf{DS.Sign}(\mathsf{SK}, \widetilde{R}^*)$, and sets $\sigma^* \leftarrow (\widetilde{R}^*, \sigma^*_{\widetilde{R}^*})$. Then, $\mathcal{B}$ runs $\mathcal{A}(\mathsf{VK}, \sigma^*)$, and responds to each of $\mathcal{A}$'s queries as follows:

- For each command query from $\mathcal{A}$, $\mathcal{B}$ forwards the query to $\mathcal{B}$'s own command oracle $\mathcal{O}_{\mathbb{C}}$, and returns the result from $\mathcal{O}_{\mathbb{C}}$ back to $\mathcal{A}$. (Note that this causes an update of the counter c and the list $L_{\mathbb{X}}$ maintained in $\mathcal{O}_{\mathbb{C}}$.)
- For each signing query $i \in [c]$ from $\mathcal{A}$ (where c denotes the current counter maintained by $\mathcal{O}_{\mathbb{C}}$), $\mathcal{B}$ makes a command query asking $\mathcal{B}$'s oracle $\mathcal{O}_{\mathbb{C}}$ to use Sen on x_i (which must have been generated as a result of $\mathcal{A}$'s command queries made so far and contained in $L_{\mathbb{X}}$ in $\mathcal{O}_{\mathbb{C}}$), and receives $D_i \leftarrow \mathsf{Sen}(x_i)$ from $\mathcal{O}_{\mathbb{C}}$. $\mathcal{B}$ now proceeds as follows:
  - If $D_i = \bot$ then $\mathcal{B}$ returns $\sigma := \bot$ to $\mathcal{A}$.
  - Otherwise, $\mathcal{B}$ computes $\widetilde{R}_i \leftarrow \mathsf{Obf}(1^\lambda, R_{D_i})$ and $\sigma_{\widetilde{R}_i} \leftarrow \mathsf{DS.Sign}(\mathsf{SK}, \widetilde{R}_i)$, and then returns $\sigma := (\widetilde{R}_i, \sigma_{\widetilde{R}_i})$ to $\mathcal{A}$.

  When $\mathcal{A}$ terminates with output some value g as a guess for $\mathcal{P}(D^*)$, $\mathcal{B}$ outputs this g as a guess for the predicate $\mathcal{P}'(R_{D^*}) := \mathcal{P}(D^*)$, and terminates.

The above completes the description of $\mathcal{B}$. Note that $\mathcal{B}$ perfectly simulates the real Conf-COA experiment for $\mathcal{A}$. Therefore, we have

$$\Pr[\mathsf{Expt}^{\mathsf{Conf\text{-}COA\text{-}Real}}_{\mathsf{SfO}_2,\mathcal{A},\mathcal{P}}(\lambda) = 1] = \Pr[\mathsf{Expt}^{\mathsf{dVBB\text{-}Real}}_{\mathsf{Obf},\mathcal{B},\mathcal{P}'}(\lambda) = 1]. \tag{2}$$

Furthermore, by the distributional VBB security of Obf, there exists a PEA simulator $\mathcal{S}_{\mathcal{B}}$ for $\mathcal{B}$ such that we have

$$|\Pr[\mathsf{Expt}^{\mathsf{dVBB\text{-}Real}}_{\mathsf{Obf},\mathcal{B},\mathcal{P}'}(\lambda) = 1] - \Pr[\mathsf{Expt}^{\mathsf{dVBB\text{-}Sim}}_{\mathsf{Obf},\mathcal{S}_{\mathcal{B}},\mathcal{P}'}(\lambda) = 1]| = \mathrm{negl}(\lambda). \tag{3}$$

Now, using $\mathcal{S}_{\mathcal{B}}$, we define the simulator $\mathcal{S}$ for $\mathcal{A}$ that runs in $\mathsf{Expt}^{\mathsf{Conf\text{-}COA\text{-}Sim}}_{\mathsf{SfO}_2,\mathcal{S},\mathcal{P}}(\lambda)$ as follows.[12]

$\mathcal{S}$ simply runs $\mathcal{S}_{\mathcal{B}}(1^\lambda, |R_{D^*}|)$, and responds to each of $\mathcal{S}_{\mathcal{B}}$'s queries as follows:

- For each command query from $\mathcal{S}_{\mathcal{B}}$, $\mathcal{S}$ forwards the query to $\mathcal{S}$'s own command oracle $\mathcal{O}_{\mathbb{C}}$, and returns the result from $\mathcal{O}_{\mathbb{C}}$ back to $\mathcal{S}_{\mathcal{B}}$. (Note that this causes an update of the counter c and the list $L_{\mathbb{X}}$ maintained in $\mathcal{O}_{\mathbb{C}}$.)
- For each query $D' \in \mathbb{D}$ from $\mathcal{S}_{\mathcal{B}}$ to a circuit that $\mathcal{S}_{\mathcal{B}}$ expects to have access to, $\mathcal{S}$ always returns 0 to $\mathcal{S}_{\mathcal{B}}$.

When $\mathcal{S}_{\mathcal{B}}$ terminates with output some value g as a guess for $\mathcal{P}'(R_{D^*}) := \mathcal{P}(D^*)$, $\mathcal{S}$ simply outputs this g (as a guess for $\mathcal{P}(D^*)$) and terminates.

The above completes the description of $\mathcal{S}$. Note that $\mathcal{S}$ perfectly simulates the command oracle for $\mathcal{S}_{\mathcal{B}}$ due to the use of its command oracle. Therefore, unless $\mathcal{S}_{\mathcal{B}}$ submits data D' such that $R_{\mathbb{D}}(D^*, D') = 1$ as a circuit query, $\mathcal{S}$ perfectly simulates $\mathsf{Expt}^{\mathsf{dVBB\text{-}Sim}}_{\mathsf{Obf},\mathcal{S}_{\mathcal{B}},\mathcal{P}'}(\lambda)$ for $\mathcal{S}_{\mathcal{B}}$. By the ϵ-evasiveness property of $\mathcal{OS}$ and ϵ

[12] Note that in the simulated experiment $\mathsf{Expt}^{\mathsf{Conf\text{-}COA\text{-}Sim}}_{\mathsf{SfO}_2,\mathcal{S},\mathcal{P}}(\lambda)$, the data D^* with which $\mathcal{P}$ is considered is generated so that $x^* \leftarrow \mathcal{X}$, $D^* \leftarrow \mathsf{Sen}(x^*)$, outside $\mathcal{S}$. Note also that $\mathcal{S}_{\mathcal{B}}$ expects to have oracle access to the circuit $R_{D^*}(\cdot)$.

being negligible, the probability that $\mathcal{S}_{\mathcal{B}}$ submits such a query is bounded by $\mathrm{negl}(\lambda)$. Hence, we have

$$|\Pr[\mathsf{Expt}^{\mathsf{Conf\text{-}COA\text{-}Sim}}_{\mathsf{SfO}_2,\mathcal{S},\mathcal{P}}(\lambda)=1]-\Pr[\mathsf{Expt}^{\mathsf{dVBB\text{-}Sim}}_{\mathsf{Obf},\mathcal{S}_{\mathcal{B}},\mathcal{P}'}(\lambda)=1]|=\mathrm{negl}(\lambda). \quad (4)$$

Combining Eqns. (2) to (4), we have Eqn. (1), as required. Therefore SfO_2 satisfies Conf-COA security. ■