A preliminary version of this paper appears in ACISP 2024. This is the full version.

# Subverting Cryptographic Protocols from A Fine-Grained Perspective - A Case Study on 2-Party ECDSA

Jialiu Cheng [*]
jialiuamy@gmail.com

Yi Wang [†]
wangyi14@nudt.edu.cn

Rongmao Chen [‡]
chromao@nudt.edu.cn

Xinyi Huang [§]
xyhuang81@gmail.com

April 22, 2024

**Abstract**

The revelations of Edward Snowden in 2013 rekindled concerns within the cryptographic community regarding the potential subversion of cryptographic systems. Bellare et al. (CRYPTO'14) introduced the notion of Algorithm Substitution Attacks (ASAs), which aim to covertly leak sensitive information by undermining individual cryptographic primitives. In this work, we delve deeply into the realm of ASAs against protocols built upon cryptographic primitives. In particular, we revisit the existing ASA model proposed by Berndt et al. (AsiaCCS'22), providing a more fine-grained perspective. We introduce a novel ASA model tailored for protocols, capable of capturing a wide spectrum of subversion attacks. Our model features a modular representation of subverted parties within protocols, along with fine-grained definitions of undetectability. To illustrate the practicality of our model, we applied it to Lindell's two-party ECDSA protocol (CRYPTO'17), unveiling a range of ASAs targeting the protocol's parties with the objective of extracting secret key shares. Our work offers a comprehensive ASA model suited to cryptographic protocols, providing a useful framework for understanding ASAs against protocols.

**Keywords.** Algorithm Substitution Attack, Cryptographic Protocol, 2-Party ECDSA.

[*]National University of Defense Technology
[†]National University of Defense Technology
[‡]National University of Defense Technology
[§]Jinan University

# Contents

# 1 Introduction

The Snowden revelations in 2013 prompted the cryptographic community to consider a special class of threats known as subversion attacks. These attacks target the very core of cryptographic systems by tampering with real-world implementations. One prominent form of subversion attack is the Algorithm Substitution Attack (ASA) introduced by Bellare et al. [BPR14] in 2014. ASAs aim to covertly leak sensitive information by undermining cryptographic algorithms. This concept can be seen as a modern-day counterpart to kleptographic attacks [YY97a, YY97b], which were initially explored by Young and Yung in the 1990s and regarded as somewhat far-fetched in the cryptographic community for some time.

Over the past decade, there has been significant progress in understanding the potential hazards of ASAs, both from practical and theoretical perspectives [BPR14, DFP15, BJK15, AMV15, LCWW18, CRT+19, CHY20, AP19a, BL17, RTYZ16, BWP+22, HS21, AP19b, BSKC19, JHZ+23, CEJ23, RTYZ17, RTYZ18, TY17]. Most previous works focused on ASAs against individual cryptographic primitives. For cryptographic protocols built over multiple primitives, subverting a primitive within a protocol trivially leads to an ASA against this protocol. However, the complexity of protocols might permit the existence of other advanced ASAs against the protocols, and the formalization of ASA models for protocols might need careful considerations, especially regarding the notion of undetectability which essentially captures the fact that undetectable attacks are usually more preferred by attackers in the real world.

In most cryptographic protocols, each party's execution within a protocol instance can be divided into multiple stages, each of which could be treated as an algorithm. The attacker may subvert multiple algorithms to leak the secrets of a party through transcripts in a collaborative way. Such ASAs are quite different from conventional subversion on a single algorithm. On the other hand, parties might maintain states that evolve across different stages. These states can be categorized as internal (specific to a single protocol instance) and external (shared across multiple instances). For subverted parties, there might exist additional states (attack states) to facilitate information leakage, and normal states might turn out to be attack states. Thus, the definition of undectectability in ASA models for protocols could be more precise by taking into account detectors with different access rights to the party's states.

Berndt et al. [BWP+22] formally modeled ASAs against protocols based on the work of Russell et al. [RTYZ16]. As their main goal was to demonstrate the threat of ASAs on protocols like TLS, Signal, and WireGuard, their attack model mainly considers the detectors that could obtain the running-time internal states without considering potential attack states. Consequently, subversion attacks that are detectable when the detector has access to all states of the subverted party might be identified as undetectable in their model. Motivated by this observation, our aim is to refine their ASA model for cryptographic protocols to provide a more fine-grained perspective.

## 1.1 Our Contributions

In light of the above considerations, we revisit Berndt et al.'s ASA model and refine it with finer granularity. Our contributions can be summarized as follows.

To enhance the ASA model for protocols, we introduce a modular formalization of a subverted party. This formalization represents a subverted party as a list of honest algorithms with some being substituted by malicious counterparts. These subverted algorithms encompass both normal and attack states, where attack states are further categorized into global and local states. This granular classification allows for precise descriptions of complex subversion attacks.

We provide a fine-grained definition of undetectability with varying strengths, including weak, normal, and strong undetectability. These definitions adjust the type of information that

the detector receives from the oracle. Specifically, for weak undetectability, the oracle provides messages only; for normal undetectability, it includes messages and normal states; and for strong undetectability, it encompasses messages, normal states and attack states.

To highlight the advantages of our refined definitions, we present four concrete ASAs targeting the parties in Lindell's two-party ECDSA protocol [Lin17] with the goal of extracting secret key shares. The undetectability of these attacks varies from weak to strong. Among these, the ASA against the party $P_2$ during the key generation phase, which does not rely on any attack state and leaks the secret key share within a single execution, stands out as the most efficient. Furthermore, the ASA against the party $P_2$ during the signing phase is a collaborative subversion attack, where multiple subverted algorithms sequentially update the global attack state to accelerate the leakage of key shares.

*Remark.* In this work we model substitution attacks on two-party cryptographic protocols. The reason that we choose two-party protocols as an object rather than more general multi-party protocols is to simplify the analysis. However, it is worth noting that the model for substitution attacks against multi-party protocols can be naturally extended from the two-party model.

### 1.2 Comparison with Prior Work

Russell et al. [RTYZ16] formalized three different watchdogs (offline, online, and omniscient) to detect possible subversions. The omniscient watchdog has the strongest capability and can obtain the internal state of the challenger in traditional security games for cryptographic schemes. Their hierarchical models can capture prior ASAs on cryptographic schemes such as [BPR14, BJK15, DFP15, AP19b, AMV15, LCWW18, AP19a, JHZ+23, RTYZ17, RTYZ18]. For ASA against cryptographic protocols with simulation-based security definitions, their model does not explicitly specify the information that the watchdog can access. Based on the work of Russell et al. [RTYZ16], Berndt et al. [BWP+22] employed the omniscient online watchdog to model ASAs against cryptographic protocols. In particular, the watchdog can passively obtain the view of the party (the internal states is included), while cannot access the party's internal states and potential attack states actively during offline testing. This implies that, from a perspective of analyzing attacks, their model may not be able to capture all detection capabilities.

In comparison to their work, we meticulously differentiate between various complex states of the subverted parties. We consider the capabilities of detectors at different levels and provide a granular detection model. Furthermore, we provide concrete attacks to substantiate the validity of our model. These attacks encompass both traditional ASAs against cryptographic schemes adapted to the context of cryptographic protocols and novel ASA techniques of high efficiency.

**Other Related Work.** Bellare et al. [BPR14] formally modeled ASAs against symmetric encryption schemes and proposed several specific practically implementable attacks such as IV-replacement attacks and biased ciphertext attacks. Degabriele, Farshim and Poettering [DFP15] proposed a refined security model that addressed the strong assumption of decryptability of all subverted ciphertexts in previous model. Bellare, Jaeger and Kane [BJK15] further improved upon the results in [BPR14], demonstrating that stateful ASAs can be made stateless with stronger undetectability, and proposed a universal ASA against sufficiently random encryption schemes. While previous attacks [BPR14, BJK15] primarily focused on subverting the encryption algorithm, Armour and Poettering [AP19b] introduced a different approach by subverting the decryption algorithm. Hodges and Stebila [HS21] investigated the possibility of detecting ASAs through state resetting. Berndt and Liśkiewicz [BL17] demonstrated that any randomized algorithm is susceptible to universal and undetectable ASAs. ASAs have also been proposed against digital signatures [AMV15, LCWW18, BSKC19], mes-

sage authentication [AP19a], key encapsulation mechanisms [CHY20], and popular protocols such as TLS, WireGuard, and Signal [BWP+22], Telegram's end-to-end encryption [CEJ23], as well as real-world public-key cryptosystems [JHZ+23]. Russell et al. [RTYZ16] presented hierarchical models to capture various ASAs and the security of defenses against the corresponding attacks. There is also significant research on defenses against subversion attacks, including derandomization [BPR14, AMV15, BH15, BJK15, DFP15], split-program methodology [TY17,RTYZ17,RTYZ18,CRT+19,BCJ21,AFMV19], cryptographic reverse firewall [MS15, DMS16, CDN20, CGPS21, CGS23, CMY+16] and self-guarding mechanism [FM18].

# 2 Preliminaries

**Notation.** Let $x \in \{0,1\}^*$ be a bit string, and the bit-length of $x$ is denoted by $|x|$. We use $[n]$ to represent a set of integers ranging from 1 to $n$, where $n \in \mathbb{N}$. For any index $i$ in the range $[|x|]$, the $i$-th bit of string $x$ is denoted by $x[i]$. The empty string is denoted by $\epsilon$. For two strings $x_0$ and $x_1$, their concatenation is represented as $x = x_0 \| x_1$. If $S$ is a set, by $s \leftarrow\!\$\ S$ we denote picking $s$ at random from $S$. If $A$ is a randomized algorithm, by $y \leftarrow A(x_1, \cdots, x_i; r)$ we denote running $A$ with inputs $x_1, \cdots, x_i$ and coins $r$ to deterministically return the output $y$, and by $y \leftarrow\!\$\ A(x_1, \cdots, x_i)$ we denote picking $r$ at random then computing the output $y$ as $y \leftarrow A(x_1, \cdots, x_i; r)$. Let $\mathsf{negl}(\lambda)$ be a negligible function with $\lambda \in \mathbb{N}$. For any $c \in \mathbb{N}$, there exists $n_0 \in \mathbb{N}$ such that $\mathsf{negl}(n) \leq \frac{1}{n^c}$ for all $n \geq n_0$.

## 2.1 Decisional Diffie-Hellman (DDH) assumption

Let $\mathbb{G}$ be a group of prime order $q$ and $G$ be a random generator of $\mathbb{G}$. Let $(x, y, z) \leftarrow\!\$\ \mathbb{Z}_q^3$. Let $\mathcal{A}$ be a PPT algorithm which takes as input a triple $(X = x \cdot G, Y = y \cdot G, Z)$ where $Z = z \cdot G$ or $Z = x \cdot y \cdot G$, then outputs a bit. The DDH-advantage of $\mathcal{A}$ is defined as

$$Adv^{\mathsf{ddh}}_{\mathcal{A},\mathbb{G}}(\lambda) = \left|\Pr\left[\mathcal{A}(x \cdot G, y \cdot G, z \cdot G) = 1\right] - \Pr\left[\mathcal{A}(x \cdot G, y \cdot G, x \cdot y \cdot G) = 1\right]\right|.$$

And we say that DDH assumption holds in $\mathbb{G}$ if for any PPT algorithm $\mathcal{A}$, $Adv^{\mathsf{ddh}}_{\mathcal{A},\mathbb{G}}(\lambda)$ is negligible in $\lambda$.

## 2.2 Entropy Smoothing Hash Function

Let $\mathcal{H} := \{H_{\hat{k}}\}_{\hat{k} \in \hat{\mathcal{K}}}$ be a family of keyed hash functions where $\mathcal{K}$ is the key space. Each $H_{\hat{k}}$ maps an element of $\mathcal{X}$ to the element of $\mathcal{Y}$. Let $D$ be a PPT algorithm which takes as input a random key $\hat{k}$ from $\mathcal{K}$, an element from $\mathcal{Y}$ which is either a random value of $\mathcal{Y}$ or $H_{\hat{k}}(x)$ where $x$ is a random element of $\mathcal{X}$, then outputs a bit. The ES-advantage of $D$ is defined as

$$\begin{aligned} Adv^{\mathsf{es}}_{D,\mathcal{H}}(\lambda) = \Big|&\Pr\left[D(\hat{k}, H_{\hat{k}}(x)) = 1 | \hat{k} \leftarrow\!\$\ \mathcal{K}, x \leftarrow\!\$\ \mathcal{X}\right] \\ &- \Pr\left[D(\hat{k}, y) = 1 | \hat{k} \leftarrow\!\$\ \mathcal{K}, y \leftarrow\!\$\ \mathcal{Y}\right]\Big|, \end{aligned}$$

and we say that $\mathcal{H}$ is $\mathsf{es}_\epsilon(\lambda)$-entropy smoothing if for any PPT algorithm $D$, $Adv^{\mathsf{es}}_{D,\mathcal{H}}(\lambda)$ is negligible in $\lambda$.

## 2.3 Pseudo Random Function (PRF)

Let $F$ be a function that takes as input a key $K \in \{0,1\}^\lambda$ and an element $x \in \mathcal{X}$, then returns an output $F(K, x) \in \mathcal{Y}$. Let $\mathsf{FUNS}[\mathcal{X}, \mathcal{Y}]$ be the set of all functions mapping from $\mathcal{X}$ to $\mathcal{Y}$.

Consider game $PRF_F^{\mathcal{F}}$ defined in Fig. 1, the PRF-advantage of adversary $\mathcal{F}$ against function $F$ is defined as

$$Adv_{\mathcal{F},F}^{\mathsf{PRF}}(\lambda) = 2\Pr\left[PRF_F^{\mathcal{F}}\right] - 1,$$

and we say $F$ is a PRF if for any PPT adversary $\mathcal{F}$, $Adv_{\mathcal{F},F}^{\mathsf{PRF}}(\lambda)$ is negligible in $\lambda$. If PRF $F$ is also efficiently computable and invertible given $K$, then $F$ is a pseudo random permutation (PRP) [KL07].

| *Game* $PRF_F^{\mathcal{F}}(1^\lambda)$ | $F_N(x)$ |
|---|---|
| $K \leftarrow\$ \{0,1\}^\lambda, f \leftarrow\$ \mathsf{FUNS}[\mathcal{X},\mathcal{Y}]$ | **if** $b_{prf} = 1$ **then** |
| $b_{prf} \leftarrow\$ \{0,1\}$ | $y \leftarrow F(K,x)$ |
| $b' \leftarrow \mathcal{F}^{F_N(\cdot)}$ | **else** $y \leftarrow f(x)$ |
| **return** $(b' = b_{prf})$ | **return** $y$ |

Figure 1: Game $PRF_F^{\mathcal{F}}$ to define PRF-advantage of adversary $\mathcal{F}$ against function $F$

### 2.4 ECDSA

Let $\mathbb{G}$ be an elliptic group of order $q$ generated by a point G. Let $H$ be a hash function which maps $\{0,1\}^*$ to $\mathbb{Z}_q$. Curve coordinates and scalars are represented in $\lambda = log_2(q)$ bits. The ECDSA signature scheme consists of the three algorithms as shown in Fig. 2.

| $\mathsf{Kgen}(1^\lambda)$ | $\mathsf{Sign}((\mathsf{sk}, m \in \{0,1\}^*))$ |
|---|---|
| 1 : $\mathsf{sk} \leftarrow\$ \mathbb{Z}_q$ | 1 : $k \leftarrow\$ \mathbb{Z}_q$ |
| 2 : $\mathsf{pk} = \mathsf{sk} \cdot G$ | 2 : $R = (r_x, r_y) = k \cdot G, r = r_x \bmod q$ |
| 3 : **return** $(\mathsf{sk}, \mathsf{pk})$ | 3 : $s = k^{-1}(H(m) + \mathsf{sk} \cdot r) \bmod q$ |
| | 4 : **return** $(r, s)$ |

| $\mathsf{Vrfy}(\mathsf{pk}, \mathrm{m}, (r,s))$ |
|---|
| 1 : $(r_x, r_y) = R = s^{-1}(H(m) \cdot G + r \cdot \mathsf{pk})$ |
| 2 : **return** 1 iff $r_x = r \bmod q$ |

Figure 2: the ECDSA signature scheme

## 3 ASA Model for 2-Party Protocol

### 3.1 Syntax

To facilitate the formalization of ASA against 2-party protocol, we refine the definition of party in this model as below.

**Definition 3.1** (Party)**.** Let $\Pi = (P_1, P_2)$ be a 2-party cryptographic protocol. For $b \in \{1,2\}$, party $P_b$ is composed of a list of algorithms $({P_b}^{A_1}, \cdots, {P_b}^{A_n})$, and where $n$ is the number of stages of $P_b$.

For $i \in [n]$, ${P_b}^{A_i}$ takes as input $P_b$'s input $x_b$, history $h_{b,i}$ and internal state $st_{b,i}$, outputs message $m_{b,i}^{\Pi}$ and update the internal state to $st_{b,i+1}$. Namely,

$${P_b}^{A_i}(x_b, h_{b,i}, st_{b,i}; r_{b,i}) \to (m_{b,i}, st_{b,i+1}),$$

where $r_{b,i}$ is the randomness of ${P_b}^{A_i}$, message $m_{b,i}$ is sent to $P_{3-b}$, history $h_{b,i}$ includes all the messages sent and received by $P_b$ before running $P_b^{A_i}$.

**Definition 3.2** (Subverted Party)**.** Let $\Pi = (P_1, P_2)$ be a 2-party cryptographic protocol. For $b \in \{1, 2\}$, we say $\widetilde{P_b}$ is a subverted party of $P_b$ with respect to subversion set $S \subseteq [n]$, denoted by $\widetilde{P_b} \approx_S P_b$, if

- For any $i \in [n] \backslash S$, the implementations of ${P_b}^{A_i}$ and ${\widetilde{P_b}}^{A_i}$ are the same, denoted by $P_b^{A_i} = {\widetilde{P_b}}^{A_i}$;
- For any $i \in S$, ${\widetilde{P_b}}^{A_i}$ takes as input embedded key $ek_{b,i}$, $\widetilde{P_b}$'s input $x_b$, history $h_{b,i}$, internal state $st_{b,i}$, global attack state $\sigma_{b,0}$ and local attack state $\sigma_{b,i}$, outputs message $m_{b,i}^{\Pi}$ and updates the attack and internal states as follow.

$${\widetilde{P_b}}^{A_i}(\boxed{ek_{b,i}}, x_b, h_{b,i}, st_{b,i}, \boxed{\sigma_{b,0}, \sigma_{b,i}}; r_{b,i}) \to (m_{b,i}, st_{b,i+1}, \boxed{\sigma_{b,0}, \sigma_{b,i}})$$

We remark that the generation of embedded key $ek_{b,i}$ is specified by ASA against party $P_b$ defined later. Attack state $\sigma_b = \sigma_{b,0} \| \sigma_{b,1} \| \cdots \| \sigma_{b,n}$ is shared among different instances of $\widetilde{P_b}$. Global attack state $\sigma_{b,0}$ is updated by algorithms ${\widetilde{P_b}}^{A_1}, \cdots, {\widetilde{P_b}}^{A_n}$ sequentially, while local attack state $\sigma_{b,i}$ is maintained by ${\widetilde{P_b}}^{A_i}$ only. Intuitively, global attack state $\sigma_{b,0}$ could be used to count the number of leaked bits of secrets through different algorithms, and local attack state $\sigma_{b,i}$ could indicate whether ${\widetilde{P_b}}^{A_i}$ behaves normally or not. Fig. 3 illustrates a subverted party $\widetilde{P_b}$.

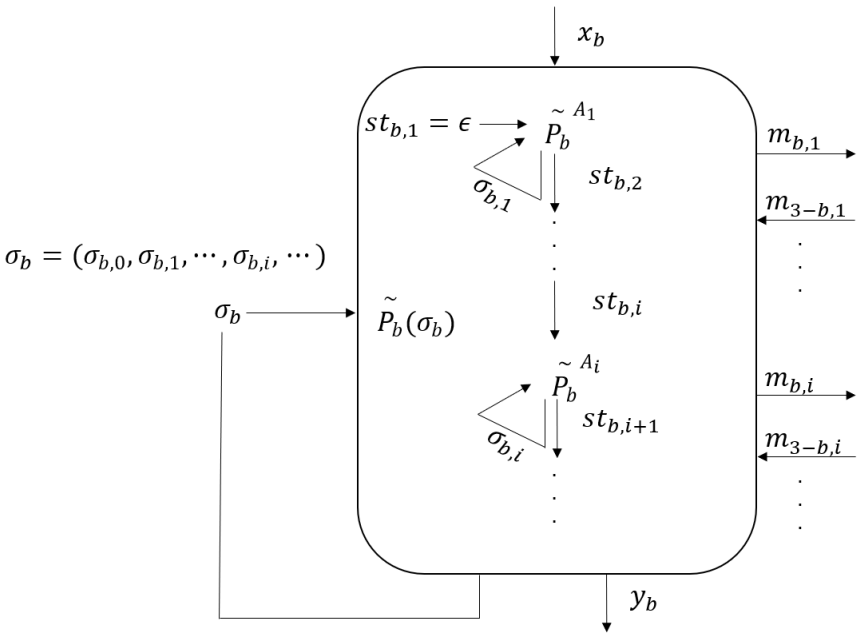


Figure 3: Subverted party $\widetilde{P_b}$

**Definition 3.3** (ASA against Party)**.** Let $\Pi = (P_1, P_2)$ be a 2-party protocol and $\widetilde{P_b}$ be a subverted party of $P_b$ with subversion set $S$ for $b \in \{1, 2\}$, an algorithm substitution attack against party $P_b$, denoted by $A_{P_b} = (\widetilde{P_b}, \mathsf{Gen}, \mathsf{Ext})$, is defined as follows.

- $\mathsf{Gen}$ takes as inputs security parameter $1^\lambda$, outputs embedded key set $EK_b = \{ek_{b,i} | i \in S\}$, extraction key set $XK_b = \{xk_{b,i} | i \in S\}$ and the description of leakage function $L_b$;
- $\mathsf{Ext}$ takes as inputs extraction key set $XK_b$, makes queries to transcript oracle $\mathcal{O}_{\mathsf{trans}}$, and outputs subliminal message $sm_b$.

$A_{P_b}$ is a *symmetric* ASA if $EK_b = XK_b$.

In general, the input of party $P_b$ might include some long-term secrets that are repeatedly used among multiple instances of $P_b$[1]. Besides, during the execution of the protocol, the party might generate some ephemeral secrets which is a part of the party's internal states. So, the goal of ASA against the party is extracting this sensitive information (e.g., secret input, internal states) from the transcripts of the subverted party. The subliminal message $sm_b$ is a leakage function $L_b$ over the inputs $x_b$, internal states $st_b$ and attack state $\sigma_b$, where $st_b = st_{b,1} \| st_{b,2} \| \cdots \| st_{b,n}$ and the attack state $\sigma_b$ might imply the index of leaked information.

In symmetric ASA, the adversary has to keep embedded key set $EK_b$ secret. Otherwise, anyone holding $EK_b$ is able to extract subliminal message $sm_b$.

### 3.2 Fine-Grained Undetectability

Subversion attacks should be undetectable to the detectors. Otherwise, users will refuse to accept the subverted implementation. Previous definitions of undetectability in ASA against primitives can be categorized into three types: offline testing, online testing and the combination of both. Comparing to online testing that can be used to detect subversion attacks such as input-trigger attacks [DFP15] and "time bombs" attacks [DFS16, WS11] by constantly monitoring the execution of the protocol, the offline testing enables the detector to detect other types of subversion attacks more proactively. In this work, we focus on detection model based on offline testing.

Our definition models detectors of different capabilities by adjusting the information returned by the offline testing oracle. Specifically, we categorize the information available to the detector into three types. The first type is limited to only the output message, which is the scenario considered in the detection model of [BWP+22] and [BPR14]. The second type includes access to the internal state in memory, similar to the omniscient watchdog presented in [RTYZ16]. Finally, the third type involves access to the attack state, similar to the detection model formalized in [BJK15]. Specific definitions of detection games are depicted in Fig. 4.

**Definition 3.4** (Undetectability)**.** Let $\Pi = (P_1, P_2)$ be a 2-party protocol and $A_{P_b} = (\widetilde{P_b}, \mathsf{Gen}, \mathsf{Ext})$ be an ASA against $P_b$. For any PPT detector $\mathcal{D}$, its advantage in game $\mathsf{DET}^{\mathcal{D}}_{P_b, A_{P_b}}(\lambda)$, as shown in Fig. 4, is

$$Adv^{\mathsf{DET}}_{\mathcal{D}, P_b, A_{P_b}}(\lambda) = \left| 2\Pr\left[ \mathsf{DET}^{\mathcal{D}}_{P_b, A_{P_b}}(\lambda) = 1 \right] - 1 \right|,$$

where $\mathsf{DET} \in \{\mathsf{wUDET}, \mathsf{UDET}, \mathsf{sUDET}\}$, and function $f$ is defined as follows.

[1] Many cryptographic protocols consist of a setup phase and a main phase. After the setup phase is completed, both parties receive some confidential information that serves as part of their inputs to execute the main phase. In this context, $\Pi$ can be regarded as the main phase of a specific protocol.

$\mathsf{DET}^{\mathcal{D}}_{P_b,A_{P_b}}(\lambda)$

$(EK_b, XK_b, L_b) \leftarrow\!\$\ \mathsf{Gen}(1^\lambda)$
$\sigma_b \leftarrow \epsilon,\ c \leftarrow\!\$\ \{0,1\}$
$c' \leftarrow \mathcal{D}^{\mathcal{O}}(EK_b, L_b)$
**return** $(c' = c)$

$\mathcal{O}(i, x_b, h_{b,i}, st_{b,i})$

**if** $i \notin [n]$ **then return** $\perp$
**elseif** $i \notin S \vee c = 1$ **then**
  $(m_{b,i}, st_{b,i+1}) \leftarrow P_b{}^{A_i}(x_b, h_{b,i}, st_{b,i})$
**else**
  $(m_{b,i}, st_{b,i+1}, \sigma_{b,0}, \sigma_{b,i}) \leftarrow \widetilde{P_b}^{A_i}(ek_{b,i}, x_b, h_{b,i}, st_{b,i}, \sigma_{b,0}, \sigma_{b,i})$
**return** $f(m_{b,i}, st_{b,i+1}, \sigma_{b,0}, \sigma_{b,i})$

Figure 4: The definition of game $\mathsf{DET}^{\mathcal{D}}_{P_b,A_{P_b}}(\lambda)$.

$$f(m_{b,i}, st_{b,i+1}, \sigma_{b,0}, \sigma_{b,i}) = \begin{cases} m_{b,i} & , \mathsf{DET} = \mathsf{wUDET} \\ (m_{b,i}, st_{b,i+1}) & , \mathsf{DET} = \mathsf{UDET} \\ (m_{b,i}, st_{b,i+1}, \sigma_{b,0}, \sigma_{b,i}) & , \mathsf{DET} = \mathsf{sUDET} \end{cases}$$

We say $A_{P_b}$ is *weakly undetectable / undetectable / strongly undetectable if for any PPT detector* $\mathcal{D}$, $Adv^{\mathsf{wUDET}}_{\mathcal{D},P_b,A_{P_b}}(\lambda)$ / $Adv^{\mathsf{UDET}}_{\mathcal{D},P_b,A_{P_b}}(\lambda)$ / $Adv^{\mathsf{sUDET}}_{\mathcal{D},P_b,A_{P_b}}(\lambda)$ *is negligible in* $k$.

*If* $A_{P_b}$ *is a symmetric ASA, detector* $\mathcal{D}$ *in game* $\mathsf{DET}^{\mathcal{D}}_{P_b,A_{P_b}}(\lambda)$ *is not allowed to access* $EK_b$.

### 3.3 Secret Recoverability

A subversion should not only remain undetectable to detectors, but should also show the subversion adversary a dedicated functionality, secret recoverability to break the security of cryptographic protocols. The work of Berndt et al. [BWP+22] considers a passively eavesdropping subversion adversary who has the knowledge of the corresponding extraction key and recovers the secret value by collecting transcripts generated by the subverted party in multiple instances. In fact, most existing ASA works consider mass surveillance adversaries.

**Definition 3.5** (Secret Recoverability)**.** Let $\Pi = (P_1, P_2)$ be a 2-party protocol and $A_{P_b} = (\widetilde{P_b}, \mathsf{Gen}, \mathsf{Ext})$ be an ASA against $P_b$. We say $A_{P_b}$ is secret recoverable if

$$\Pr\left[sm_b \neq L_b(x_b, st_b, \sigma_b) : \begin{array}{l}(EK_b, XK_b, L_b) \leftarrow\!\$\ \mathsf{Gen}(1^\lambda) \\ sm_b \leftarrow \mathsf{Ext}^{\mathcal{O}_{\mathsf{trans}}}(XK_b)\end{array}\right] \leq \mathsf{negl}(\lambda),$$

where oracle $\mathcal{O}_{\mathsf{trans}}$ returns the transcript of subverted protocol $\widetilde{\Pi} = (\widetilde{P_b}, P_{3-b})$.

| NIZKPoK.P$(x, w)$ | NIZKPoK.V$(\pi)$ | COM.G$(m)$ |
|---|---|---|
| $t \leftarrow\$ \mathbb{Z}_q$ | $(x, e, z) \leftarrow \pi$ | $r \leftarrow\$ \{0,1\}^\lambda$ |
| $a \leftarrow t \cdot G$ | $a' \leftarrow z \cdot G - e \cdot x$ | **return** $(H'(m\|\|r), m\|\|r)$ |
| $e \leftarrow H(a\|\|x)$ | $e' \leftarrow H(a'\|\|x)$ | |
| $z \leftarrow t + e \cdot w$ | **if** $e' = e$ **then** | COM.V$(com, dcom)$ |
| $\pi \leftarrow (x, e, z)$ | **return** 1 | $m\|\|r \leftarrow dcom$ |
| **return** $\pi$ | **return** $\perp$ | **if** $com = H'(m\|\|r)$ **then** |
| | | **return** 1 |
| | | **return** $\perp$ |

Figure 5: NIZKPoK and COM for ideal functionalities in Lin-2ECDSA with hash functions $H : \{0,1\}^* \to \mathbb{Z}_q$ and $H' : \{0,1\}^* \to \{0,1\}^\lambda$

# 4 Case Study: Subverting 2ECDSA

In this section, we take 2-party ECDSA (2ECDSA) protocol as an example to study concrete substitution attacks against protocols. Lin-2ECDSA is a well-known 2ECDSA protocol proposed by Lindell [Lin17] at CRYPTO'17. Note that it is presented in $(\mathcal{F}_{zk}^{\mathcal{R}_{DL}}, \mathcal{F}_{com-zk}^{\mathcal{R}_{DL}})$-hybrid model which hides the details of these modules and might hinder the investigation of potential subversion attacks. We instantiate these ideal functionalities with specific schemes[2] in Fig. 5.

We now illustrate how Lin-2ECDSA works. Lin-2ECDSA consists of a distributed key generation sub-protocol $\Pi_{kgen}$ that generates multiplicative shares of the secret key for parties, respectively, and a signing sub-protocol $\Pi_{sign}$ that reaches the same signature of a message for parties without revealing each other's key share. More specifically, in $\Pi_{kgen}$ $P_1$ and $P_2$ randomly select $\mathsf{sk}_1$ and $\mathsf{sk}_2$ as their multiplicative shares of $\mathsf{sk}$ respectively, then run a simulatable Diffie-Hellman key exchange protocol to generate $\mathsf{pk} = \mathsf{sk}_1 \cdot \mathsf{sk}_2 \cdot G$ securely. Additionally, to improve the efficiency of $\Pi_{sign}$, $P_1$ generates a public-private key pair $(\mathsf{pk}_e, \mathsf{sk}_e)$ for the Paillier encryption scheme, then use $\mathsf{pk}_e$ to calculate the ciphertext of $\mathsf{sk}_1$, denoted as $c_{key}$ and send it to $P_2$ together with $\mathsf{pk}_e$. At the end of the execution of $\Pi_{kgen}$, $P_1$ and $P_2$ obtain their respective key shares of $\mathsf{sk}$ and the public key $\mathsf{pk}$, besides $P_2$ gets $c_{key}$. $P_1$ and $P_2$ execute $\Pi_{sign}$ to perform a distributed signing on a message $m$. They first randomly select $k_1$ and $k_2$ respectively, and execute a similar simulatable Diffie-Hellman key exchange protocol to obtain a point $R = k_1 \cdot k_2 \cdot G$, then set $r = r_x \ mod \ q$. $P_2$ uses $\mathsf{pk}_e$ to perform a homomorphic computation on $c_{key}$ to obtain the ciphertext of $s' = k_2^{-1} \cdot H(m) + k_2^{-1} \cdot r \cdot \mathsf{sk}_2 \cdot \mathsf{sk}_1$ and send it to $P_1$, who decrypts it to obtain $s'$ and computes $s = k_1^{-1} \cdot s' \ mod \ q$ then outputs $(r, s)$ or sends it to $P_2$ as the signature on $m$ after verifying its correctness. [3]

The goal of ASA against parties in $\Pi_{kgen}$ and $\Pi_{sign}$ is to retrieve the private key share ($\mathsf{sk}_1$ or $\mathsf{sk}_2$). Consider that $\Pi_{kgen}$ usually executes once and $\Pi_{sign}$ would run multiple times with same key shares as input, different ASAs for these sub-protocols are proposed as below.

## 4.1 Substitution attack against $\Pi_{kgen}$

For clarity, Fig. 6 only presents algorithms $P_1^{A_1}$ and $P_2^{A_1}$ in key generation sub-protocol $\Pi_{kgen}$ that are subverted in following ASAs. See Fig. 14 in Appendix A for a full description of $\Pi_{kgen}$.

[2] The schemes for NIZKPoK and COM we present here are widely used in the implementation of threshold ECDSA protocols (e.g., [Lin17, DKLs18, GG18, XAX+21, CGG+21]).

[3] What we give here is only a rough description. Additional proofs accompanying the messages to be sent are omitted. For more details, refer to [Lin17].

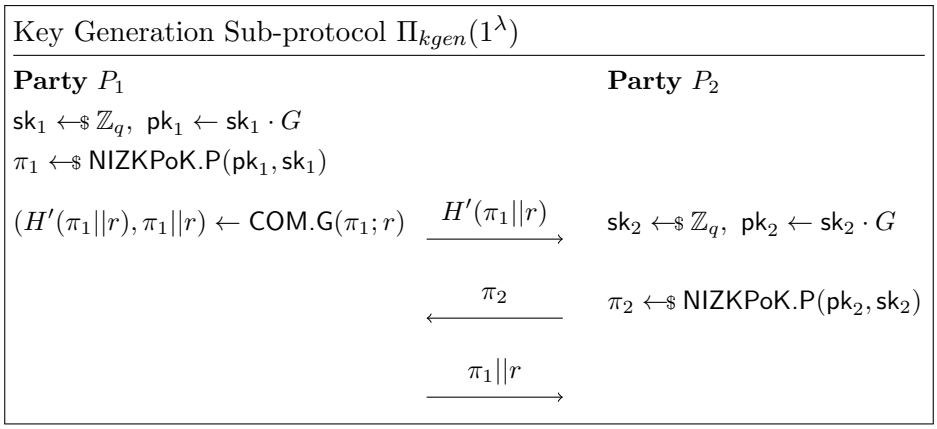


Figure 6: Part of $\Pi_{kgen}$ in Lin-2ECDSA

$\widetilde{P_1}^{A_1}(ek_{1,1}, x_1, \sigma_{1,1})$

$\mathsf{sk}_1 \leftarrow\$ \mathbb{Z}_q, \mathsf{pk}_1 \leftarrow \mathsf{sk}_1 \cdot G$
$\pi_1 \leftarrow\$ \mathsf{NIZKPoK.P}(\mathsf{pk}_1, \mathsf{sk}_1)$
$(H'(\pi_1||r), \pi_1||r, \sigma_{1,1}) \leftarrow \widetilde{\mathsf{COM.G}}(\pi_1, ek_{1,1}, \mathsf{sk}_1, \sigma_{1,1})$
$st_{1,2} \leftarrow \mathsf{sk}_1||\mathsf{pk}_1||\pi_1||r$
**return** $(H'(\pi_1||r), st_{2,1}, \sigma_{1,1})$

$\widetilde{\mathsf{COM.G}}(m, ek, aux, \sigma)$

**if** $\sigma = \epsilon$ **then** $r \leftarrow E(ek, aux)$
**else** $\sigma \leftarrow 0,\ r \leftarrow\$ \{0,1\}^k$
**return** $(H'(m||r), m||r, \sigma)$

$\mathsf{Gen}(1^\lambda)$

$ek_{1,1} \leftarrow\$ \mathcal{K}$
$EK_1 = XK_1 = \{ek_{1,1}\}$
$\mathsf{sk}_1 := L_1(x_1, st_1, \sigma_1)$
**return** $(EK_1, XK_1, L_1)$

$\mathsf{Ext}^{\mathcal{O}_{\mathsf{trans}}}(XK_1)$

$dcom_1 := \pi_1 \| r$
$\mathsf{sk}_1 = E^{-1}(ek_{1,1}, r)$
**return** $\mathsf{sk}_1$

Figure 7: Symmetric ASA against $P_1$ in $\Pi_{kgen}$.

### 4.1.1 ASA against $\Pi_{kgen}.P_1$

One can note that, in Fig. 6, the decommitment $\pi_1||r$ that includes the randomness $r$ in $\mathsf{COM.G}$ is transmitted to $P_2$ in plaintext, and this permits leaking $P_1$'s key share $\mathsf{sk}_1$ in one single execution of $\Pi_{kgen}$ by adopting the IV-replacement attack [BPR14]. More specifically, Fig. 7 depicts a symmetric ASA against $P_1$ in $\Pi_{kgen}$. $\widetilde{P_1}$ is a subverted party of $P_1$ and $\widetilde{P_1}^{A_1}(ek_{1,1}, x_1, \sigma_{1,1})$ is the same as $P_1^{A_1}(x_1)$ except that $(H'(\pi_1||r), \pi_1||r)$ is computed by $\widetilde{\mathsf{COM.G}}(\pi_1, ek_{1,1}, \mathsf{sk}_1, \sigma_{1,1})$ instead of $\mathsf{COM.G}(\pi_1)$.

$\widetilde{\mathsf{COM.G}}$ is the same as $\mathsf{COM.G}$ except that $r$ is computed using $\mathsf{PRP}$ $E : \mathcal{K} \times \{0,1\}^\lambda \to \{0,1\}^\lambda$ when $\sigma = \epsilon$[4]. For the input of $\widetilde{P_1}^{A_1}$, we have $ek_{1,1} \in \mathcal{K}$ and $\sigma_{1,1}$ is $\sigma$ in $\widetilde{\mathsf{COM.G}}$. The leakage function $L_1(x_1, st_1, \sigma_1)$ returns $\mathsf{sk}_1$ in $st_1$.

**Theorem 4.1.** *Assume that $E : \mathcal{K} \times \{0,1\}^\lambda \to \{0,1\}^\lambda$ is a $\mathsf{PRP}$, then ASA $A_{P_1}$ in Fig. 7 is undetectable and secretly recoverable. For any $\mathsf{PPT}$ $\mathcal{D}$ we have*

$$Adv^{\mathsf{UDET}}_{D,P_1,A_{P_1}}(\lambda) \leq 2Adv^{\mathsf{PRF}}_{\mathcal{F}_E,E}(\lambda) \tag{1}$$

[4]In fact $E$ here selects the input message from $\mathbb{Z}_q$, the element in which is represented in $\lambda$ bits.

*where $\mathcal{F}_E$ is a* PRF *adversary attacking $E$.*

*Proof.* Let $G_0$ be the original detection game $\mathsf{UDET}^{\mathcal{D}}_{P_1,A_{P_1}}$, let $W_i$ denote the event that $\mathcal{D}$ returns correct $c'$ in game $G_i$, from the definition we have

$$Adv^{\mathsf{UDET}}_{\mathcal{D},P_1,A_{P_1}}(\lambda) = |2\Pr[W_0] - 1|.$$

Let $G_1$ be the same as $G_0$ except that $r \leftarrow E(ek, aux)$ in $\widetilde{\mathsf{COM.G}}$ of $\widetilde{P_1}^{A_1}$ is replaced by $r \leftarrow\!\!\$\ \{0,1\}^\lambda$. We construct a PRF adversary $\mathcal{F}_E$ attacking $E$ that simulates the game $G_0$ or $G_1$ for the detector $\mathcal{D}$.

Since $A_{P_1}$ is a symmetric ASA, $\mathcal{D}$ does not know $EK_1$. $\mathcal{F}_E$ only need to provide leakage function $L_1$ and simulate the oracle for $\mathcal{D}$ as follow.

Upon query $(1, 1^k)$ when $c = 0$ and $\sigma_{1,1} = \epsilon$, $\mathcal{F}_E$ runs $\widetilde{P_1}^{A_1}$ except that randomness $r$ in $\widetilde{\mathsf{COM.G}}$ is set as the value returned by the challenger of $\mathcal{F}_E$ on query $\mathsf{sk}_1$. For other queries, $\mathcal{F}_E$ simulates the oracle following the description.

If $r = E(ek_{1,1}, \mathsf{sk}_1)$, $\mathcal{F}_E$ simulates $G_0$ for $\mathcal{D}$. Otherwise, the simulation is $G_1$. Hence, we have

$$|\Pr[W_0] - \Pr[W_1]| \leq Adv^{\mathsf{PRF}}_{\mathcal{F}_E,E}(\lambda).$$

Besides, we notice that in $G_1$ the computation of $H'(\pi_1||r)$ is always using the true randomness and thus the view of the detector $\mathcal{D}$ actually does not depend on the chosen bit $c$. We have

$$\Pr[W_1] = 1/2.$$

Putting all above together, Eq. 4.1 holds.

Regarding the secret recoverability of $A_{P_1}$, the subversion adversary can recover the secret key share $\mathsf{sk}_1$ with probability 1 from the transcript (more precisely, the second message sent out by $\widetilde{P_1}$) of the first execution of $\widetilde{P_1}$ returning from $\mathcal{O}_{\mathsf{trans}}$ following the extraction algorithm Ext given in Fig. 7. □

| $\widetilde{P_2}^{A_1}(ek_{2,1}, x_2, h_{2,1})$ | $\mathsf{Gen}(1^\lambda)$ | $\mathsf{Ext}^{\mathcal{O}_{\mathsf{trans}}}(XK_2)$ |
|---|---|---|
| $t \leftarrow\!\!\$\ \mathbb{Z}_q,\ a \leftarrow t \cdot G$ | $xk_{2,1} \leftarrow\!\!\$\ \mathbb{Z}_q$ | $(\mathsf{pk}_2, e, z) \leftarrow \pi_2$ |
| $\mathsf{sk}_2 \leftarrow H_{\hat{k}}(ek_{2,1} \cdot t)$ | $ek_{2,1} \leftarrow xk_{2,1} \cdot G$ | $a \leftarrow z \cdot G - e \cdot \mathsf{pk}_2$ |
| $\mathsf{pk}_2 \leftarrow \mathsf{sk}_2 \cdot G$ | $EK_2 = \{ek_{2,1}\}$ | $\mathsf{sk}_2 \leftarrow H_{\hat{k}}(xk_{2,1} \cdot a)$ |
| $e \leftarrow H(a\|\|\mathsf{pk}_2)$ | $XK_2 = \{xk_{2,1}\}$ | **return** $\mathsf{sk}_2$ |
| $z \leftarrow t + e \cdot \mathsf{sk}_2$ | $\mathsf{sk}_2 := L_2(x_2, st_2, \sigma_2)$ | |
| $\pi_2 \leftarrow (\mathsf{pk}_2, e, z)$ | **return** $(EK_2, XK_2, L_2)$ | |
| **return** $\pi_2$ | | |

Figure 8: ASA against $P_2$ in $\Pi_{kgen}$.

### 4.1.2 ASA against $\Pi_{kgen}.P_2$

Fig. 8 depicts an ASA against $P_2$ in $\Pi_{kgen}$. $\widetilde{P_2}$ is a subverted party of $P_2$ and $\widetilde{P_2}^{A_1}(ek_{2,1}, x_2, h_{2,1})$ is the same as $P_2^{A_1}(x_2, h_{2,1})$ except that $\mathsf{sk}_2$ is not uniformly sampled from $\mathbb{Z}_q$. The leakage function $L_2(x_2, st_2, \sigma_2)$ returns $\mathsf{sk}_2$ in $st_2$.

It is worth noting this ASA against $P_2$ in $\Pi_{kgen}$ does not treat the underlying module (i.e., NIZKPoK.P) as a black box, and is extremely efficient for that the key share $\mathsf{sk}_2$ is leaked in one single execution.

**Theorem 4.2.** *Assume that DDH assumption holds in $\mathbb{G}$ and $\mathcal{H} = \{H_{\hat{k}}\}_{\hat{k}\in\hat{\mathcal{K}}}$ is entropy smoothing, ASA $A_{P_2}$ in Fig. 8 is weakly undetectable and secretly recoverable with high efficiency when $H : \{0,1\}^* \to \mathbb{Z}_q$ is modeled as a random oracle. More specifically, assume that detector $\mathcal{D}$ makes at most $q$ queries on $P_2^{A_1}$ or $\widetilde{P_2}^{A_1}$ and at most $q_H$ queries on $H$, then we have*

$$Adv^{\mathsf{wUDET}}_{\mathcal{D},P_2,A_{P_2}}(\lambda) \leq 2q \cdot (\epsilon_{\mathsf{es}}(\lambda) + Adv^{\mathsf{ddh}}_{\mathcal{A},\mathbb{G}}(\lambda) + \frac{q+q_H}{2^{2\lambda-1}}) \tag{2}$$

*Proof.* We first prove the weak undetectability of $A_{P_2}$ against $P_2$. Let $G_0$ be the same as the original detection game $\mathsf{wUDET}^{\mathcal{D}}_{P_2,A_{P_2}}$ where $H : \{0,1\}^* \to \mathbb{Z}_q$ is modeled as a random oracle. More specifically, $\mathcal{D}$ is given two oracles: one is oracle $\mathcal{O}$ to answer queries with $(i, ek_{b,i}, x_b, st_{b,i}, h_{b,i})$ and one is $\mathcal{O}_H$ which answers hash query. Upon receiving a hash query $h := a||\mathsf{pk}_2$, if $h$ has been queried before, then return the same answer as before. If not, then return a random element $e$ from $\mathbb{Z}_q$. We split the oracle $\mathcal{O}$ into two oracles $\mathcal{O}_0$ and $\mathcal{O}'$, where $\mathcal{O}_0$ only answers the query with $(1, ek_{2,1}, x_2, h_{2,1})$ as shown in Fig. 9, and $\mathcal{O}'$ answers the other queries from the detector $\mathcal{D}$. let $W_i$ denote the event that $\mathcal{D}$ returns correct $c'$ in game $G_i$, from the definition we have

$$Adv^{\mathsf{wUDET}}_{\mathcal{D},P_2,A_{P_2}}(\lambda) = |2\Pr[W_0] - 1|.$$

We define a sequence of games: $\{G_{1,1}, G_{1,2}, \cdots, G_{q,1}, G_{q,2}\}$ in Fig. 9. Specifically, In $G_{i,1}, G_{i,2}$ ($i \in [1,q]$), an internal counter $j$ (initialized to 0) is set for the oracle $\mathcal{O}_i$ and increments upon each query with $(1, ek_{2,1}, x_2, h_{2,1})$ by the detector $\mathcal{D}$. [5] $G_{i,1}$ is the same as $G_{i-1,2}$ ($G_{0,2}$ is the same as $G_0$) except that when $j = i$ then $\mathcal{O}_i$ in $G_{i,1}$ generates $\mathsf{sk}_2$ by $H_{\hat{k}}(A)$ where $A$ is a random element in $\mathbb{G}$. $G_{i,2}$ is the same as $G_{i,1}$ except that when $j = i$ then $\mathcal{O}_i$ in $G_{i,2}$ generates $\mathsf{sk}_2$ by randomly selecting it from $\mathbb{Z}_q$.

We prove that $G_{i-1,2} \approx_c G_{i,1}$ from the view of the detector $D$ by constructing an adversary $\mathcal{A}_i$ attacking DDH assumption relative to the group $(\mathbb{G}, G, q)$. Suppose that $\mathcal{A}_i$ receives $(X, Y, Z)$ from its challenger where $X = x \cdot G, Y = y \cdot G$. Its goal is to tell whether $Z = z \cdot G$ where $z \leftarrow\$ \mathbb{Z}_q$ or $Z = x \cdot y \cdot G$.

$\mathcal{A}_i$ simulates the oracles for the detector $\mathcal{D}$ as depicted in Fig. 10. We denote the game simulated by $\mathcal{A}_i$ when $z$ is a random element in $\mathbb{Z}_q$ as $S_{i,1}$, and the simulated game when $z = x \cdot y$ as $S_{i-1,2}$. Let $\Pr[S_{i,1}]$ and $\Pr[S_{i-1,2}]$ denote the probability that $\mathcal{D}$ returns the correct $c'$ in $S_{i,1}$ and $S_{i-1,2}$ respectively. From the definition we have

$$|\Pr[S_{i,1}] - \Pr[S_{i-1,2}]| \leq Adv^{\mathsf{ddh}}_{\mathcal{A}_i,\mathbb{G}}(\lambda).$$

We notice that $S_{i,1}$ equals to $G_{i,1}$ and $S_{i-1,2}$ equals to $G_{i-1,2}$ except that $h^* := Y||H_{\hat{k}}(Z) \cdot G$ has been queried before. Suppose that $\mathcal{D}$ makes at most $q_H$ queries on $H$, then the probability that $h^*$ has been queried is at most $\frac{q+q_H}{2^{2\lambda}}$. Combined with the difference lemma, we have

$$|\Pr[S_{i,1}] - \Pr[W_{i,1}]| \leq \frac{q+q_H}{2^{2\lambda}} \leq negl(\lambda),$$

and

$$|\Pr[S_{i-1,2}] - \Pr[W_{i-1,2}]| \leq \frac{q+q_H}{2^{2\lambda}} \leq negl(\lambda).$$

[5] To keep consistent, we also include a counter $j$ in the game $G_0$.

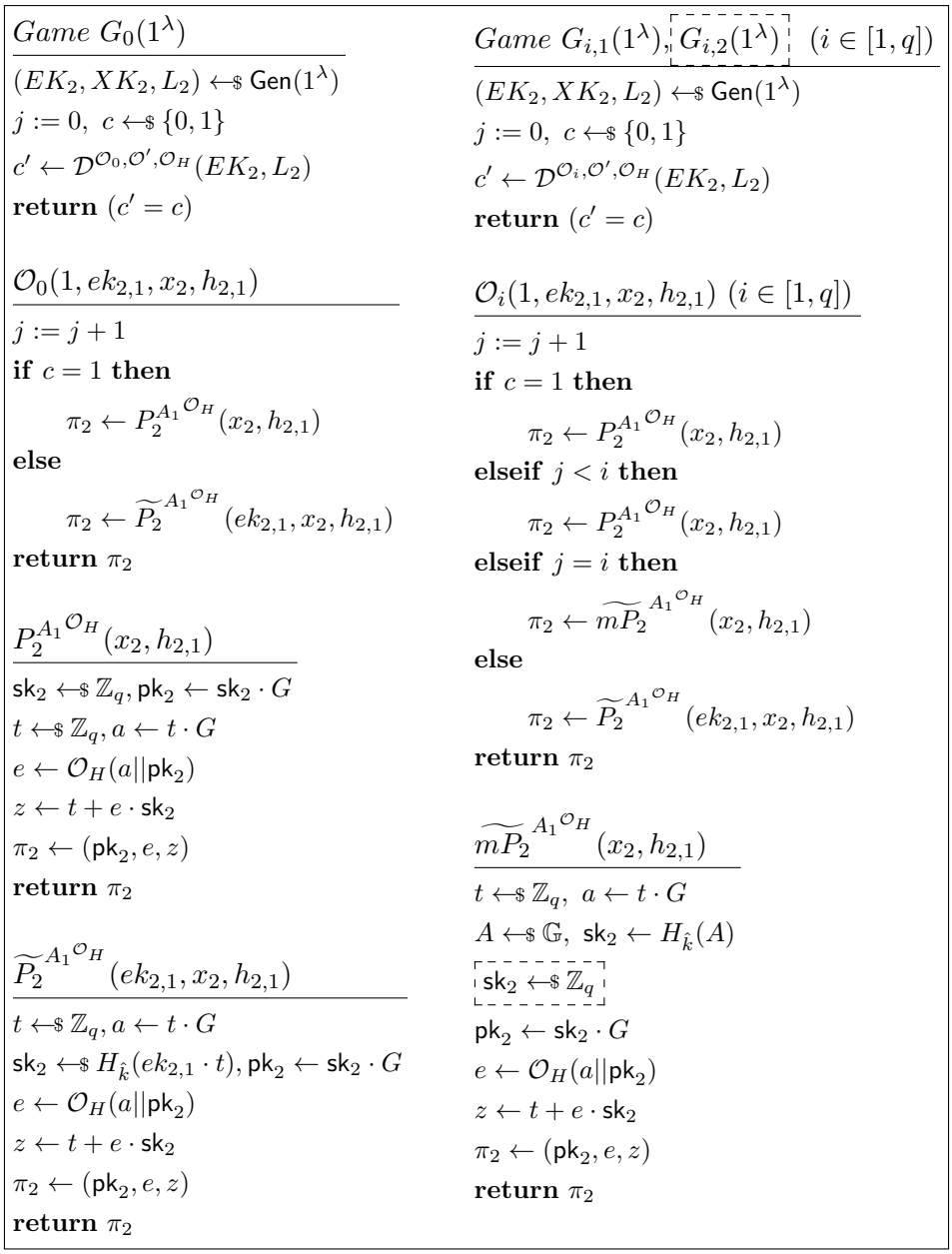


Figure 9: Games $G_0$ and $G_{i,1}, G_{i,2}$ $(i \in [1,q])$ in the proof of Theorem. 4.2

We prove that $G_{i,1} \approx_c G_{i,2}$ from the view of the detector $D$ by constructing an adversary $D_i$ attacking the entropy smoothing hash function $H_{\hat{k}} : \mathbb{G} \to \mathbb{Z}_q$. Suppose that $D_i$ receives $(\hat{k}, y^*)$ from its challenger. Its goal is to tell whether $y^* = H_{\hat{k}}(x)$ where $x \leftarrow\$ \mathbb{G}$ or $y^* \leftarrow\$ \mathbb{Z}_q$.

$D_i$ simulates $G_{i,1}$ and $G_{i,2}$ for the detector $\mathcal{D}$. $D_i$ runs the setup phase then simulates an oracle $\mathcal{O}$ as follows. It keeps an internal counter $j$ which is initialized to 0 and increases with querying with $(1, ek_{2,1}, x_2, h_{2,1})$ from the detector $\mathcal{D}$. For $j < i$, it runs $P_2^{A_1}$ and returns the output, for $j = i$, it runs $\widetilde{P_2}^{A_1}$ except that setting $\mathsf{sk}_2$ as $y^*$ then returns the output. For $j > i$, it runs $\widetilde{P_2}^{A_1}$ and returns the output. To answer $\mathcal{D}$'s queries with $(i, x_2, st_{2,i}, h_{2,i})$ where $i \neq 1$ it follows the description of $\mathcal{O}'$. $D_i$ simulates the oracle $\mathcal{O}_H$ faithfully follow the description. Finally, $D_i$ outputs 1 if the detector $\mathcal{D}$ outputs $(c' = c)$ otherwise outputs 0. From the view of

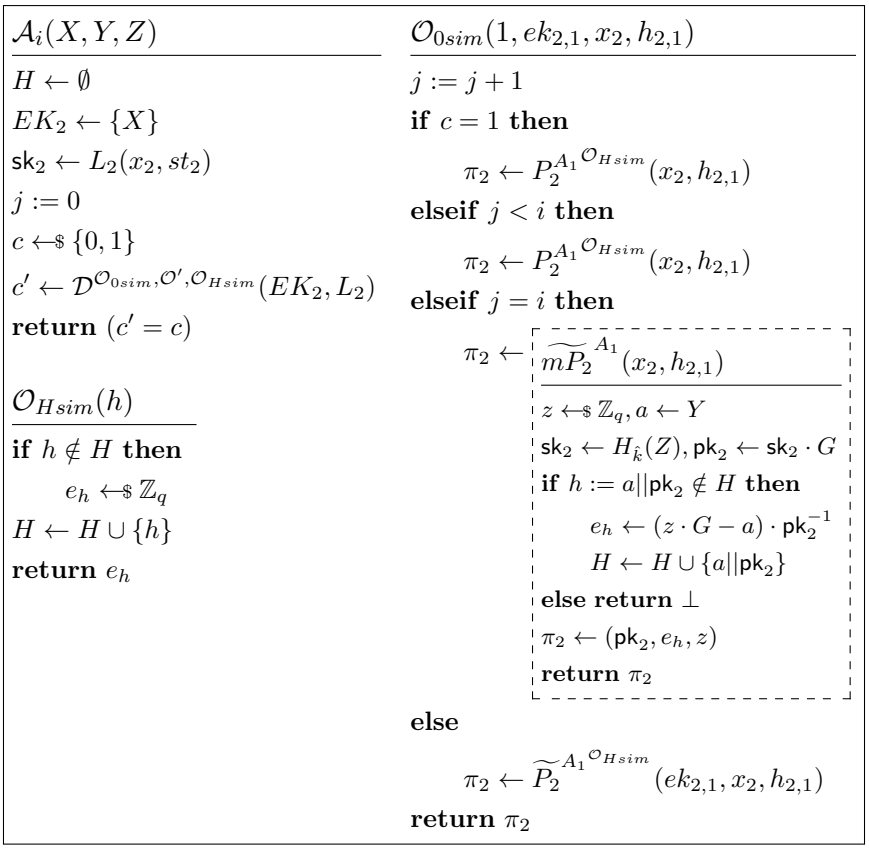


Figure 10: Adversary $\mathcal{A}_i$ attacking DDH assumption relative to the group $(\mathbb{G}, G, q)$ simulates detection game $G_{i-1,2}$ and $G_{i,1}$ for the detector $\mathcal{D}$

the detector $\mathcal{D}$, if $y^*$ is randomly selected from $\mathbb{Z}_q$ then the game simulated by $D_i$ is exactly the game $G_{i,2}$. Otherwise, the simulated game is $G_{i,1}$. Hence, we have

$$|\Pr[W_{i,1}] - \Pr[W_{i,2}]| \leq \epsilon_{\mathsf{es}}(\lambda).$$

Besides, we notice that in game $G_{q,2}$, $\mathcal{O}$ always runs $P_2^{A_1}$ to answer $\mathcal{D}$'s query with $(1, ek_{2,1}, x_2, h_{2,1})$, and thus the view of the detector $\mathcal{D}$ actually does not depend on the chosen bit $c$. Therefore,

$$\Pr[W_{q,2}] = \frac{1}{2}.$$

Putting everything together Eq. 4.2 holds.

Regarding the secret recoverable of $A_{P_2}$, following the extraction algorithm Ext given in Fig. 8 the subversion adversary can recover the secret key share $\mathsf{sk}_2$ with probability 1 from the transcript (returning from $\mathcal{O}_{\mathsf{trans}}$) of *any* execution of $\widetilde{P_2}$ successfully. □

### 4.2 Substitution attack against $\Pi_{sign}$

For clarity, Fig. 11 only presents $P_1^{A_1}$, $P_2^{A_1}$ and $P_2^{A_2}$ in $\Pi_{sign}$ that are subverted in following ASAs. See Fig. 15 in Appendix A for a full description of $\Pi_{sign}$. In $\Pi_{sign}$, $P_1$ and $P_2$ take their long-term secret such as the decryption key ($\mathsf{sk}_e$ of $P_1$) or their private key shares ($\mathsf{sk}_1$ and $\mathsf{sk}_2$) as part of their input in every execution of distributed signing on different messages. Compared with $\Pi_{kgen}$, which only runs once, $\Pi_{sign}$, which can be executed multiple times, provides more possibilities for successful subversion attacks.

Signing Sub-protocol $\Pi_{sign}(sid, m)$

| **Party** $P_1(\mathsf{sk}_1, \mathsf{pk}_e, \mathsf{sk}_e, \mathsf{pk})$ | | **Party** $P_2(\mathsf{sk}_2, \mathsf{pk}_e, c_{key}, \mathsf{pk})$ |
|---|---|---|
| $k_1 \leftarrow\!\!\$\ \mathbb{Z}_q, R_1 \leftarrow k_1 \cdot G$ | | |
| $\pi_1 \leftarrow\!\!\$\ \mathsf{NIZKPoK.P}(R_1, k_1)$ | | $k_2 \leftarrow\!\!\$\ \mathbb{Z}_q, R_2 \leftarrow k_2 \cdot G$ |
| $(H'(\pi_1 \|\| r), \pi_1 \|\| r) \leftarrow \mathsf{COM.G}(\pi_1; r)$ | $\xrightarrow{H'(\pi_1 \|\| r)}$ | $\pi_2 \leftarrow\!\!\$\ \mathsf{NIZKPoK.P}(\mathsf{pk}_2, \mathsf{sk}_2)$ |
| | $\xleftarrow{\pi_2}$ | **if** $\mathsf{COM.V}(H'(\pi_1 \|\| r), \pi_1 \|\| r) = \perp \vee$ |
| | $\xrightarrow{\pi_1 \|\| r}$ | $\mathsf{NIZKPoK.V}(\pi_1) = \perp$ **then** |
| | | **return** $\perp$ |
| | | $R := (r_x, r_y) \leftarrow k_2 \cdot R_1$ |
| | | $r \leftarrow r_x \bmod q,\ \rho \leftarrow \mathbb{Z}_{q^2},\ \tilde{r} \leftarrow \mathbb{Z}_{N^*}$ |
| | | $c_1 \leftarrow \mathsf{Paillier.Enc}(\mathsf{pk}_e, \rho \cdot q +$ |
| | | $[k_2^{-1} \cdot m \bmod q; \tilde{r}])$ |
| | | $v \leftarrow k_2^{-1} \cdot r \cdot \mathsf{sk}_2 \bmod q$ |
| | $\xleftarrow{c_3}$ | $c_2 \leftarrow v \otimes c_{key},\ c_3 \leftarrow c_1 \oplus c_2$ |

Figure 11: Part of $\Pi_{sign}$ in Lin-2ECDSA

$\widetilde{P_1}^{A_1}(ek_{1,1}, x_1, h_{1,1})$

$k_1 \leftarrow\!\!\$\ \mathbb{Z}_q, R_1 \leftarrow k_1 \cdot G$

$\pi_1 \leftarrow\!\!\$\ \widetilde{\mathsf{NIZKPoK}}.\mathsf{P}(R_1, k_1, ek_{1,1}, \mathsf{sk}_1)$

$(H'(\pi_1 \|\| r), \pi_1 \|\| r) \leftarrow \mathsf{COM.G}(\pi_1; r)$

$st_{1,2} \leftarrow k_1 \|\| R_1 \|\| \pi_1 \|\| r$

**return** $(H'(\pi_1 \|\| r), st_{1,2})$

$\widetilde{\mathsf{NIZKPoK}}.\mathsf{P}(x, w, ek, aux)$

**repeat**

$t \leftarrow\!\!\$\ \mathbb{Z}_q, a \leftarrow t \cdot G$

$(b, l) \leftarrow F_1(ek, a)$

**until** $aux[l] = b$

$e \leftarrow H(a \|\| x), z \leftarrow t + e \cdot w$

$\pi \leftarrow (x, e, z)$

**return** $\pi$

$\mathsf{Gen}(1^\lambda)$

$ek_{1,1} \leftarrow\!\!\$\ \mathcal{K}$

$EK_1 = XK_1 = \{ek_{1,1}\}$

$\mathsf{sk}_1 := L_1(x_1, st_1, \sigma_1)$

**return** $(EK_1, XK_1, L_1)$

$\mathsf{Ext}^{\mathcal{O}_{\mathsf{trans}}}(XK_1)$

$\mathsf{T}_{\mathsf{m}_{1,2}} := \{\pi_1^1, \cdots, \pi_1^{\mathsf{i}}, \cdots, \pi_1^{\mathsf{s}}\}$

$i := 1$

**for** $i \in [s]$ **do**

$(x^i, e^i, z^i) \leftarrow \pi_1^i$

$a^i \leftarrow z^i \cdot G - e^i \cdot x^i$

$(b, l) \leftarrow F_1(ek_{1,1}, a^i)$

$\mathsf{sk}_1[l] = b, i \leftarrow i + 1$

**return** $\mathsf{sk}_1$

Figure 12: Symmetric ASA against $P_1$ in $\Pi_{sign}$.

### 4.2.1 ASA against $\Pi_{sign}.P_1$

Inspired by the stateless ASA against symmetric encryption in [BJK15], we provide an ASA against $P_1$ in $\Pi_{sign}$ by subverting the sub-module $\mathsf{NIZKPoK.P}$ of $P_1^{A_1}$.

Fig. 12 depicts a symmetric ASA $A_{P_1}$ against $P_1$ in $\Pi_{sign}$. $A_{P_1} = (\widetilde{P_1}, \mathsf{Gen}, \mathsf{Ext})$ where $\widetilde{P_1}$ is a subverted party of $P_1$ and $\widetilde{P_1}^{A_1}(ek_{1,1}, x_1, h_{1,1})$ is the same as $P_1^{A_1}(x_1, h_{1,1})$ except that $\pi_1$ is computed as $\widetilde{\mathsf{NIZKPoK.P}}(R_1, k_1, ek_{1,1}, \mathsf{sk}_1)$ instead of $\mathsf{NIZKPoK.P}(R_1, k_1)$. $\widetilde{\mathsf{NIZKPoK.P}}$ is the same as $\mathsf{NIZKPoK.P}$ except that it uses a $\mathsf{PRF}$ $F_1 : \mathcal{K} \times \mathbb{G} \to \{0,1\} \times [\lambda]$ to constantly test whether the output $(b, l)$ by applying $F_1$ on $a$ satisfies $aux[l] = b$, until $a$ is generated "properly". For the input of $\widetilde{P_1}^{A_1}$, we have $ek_{1,1} \in \mathcal{K}$. The leakage function $L_1(x_1, st_1, \sigma_1)$ returns $\mathsf{sk}_1$ in $x_1$.

**Theorem 4.3.** *Assume that $F_1 : \mathcal{K} \times \mathbb{G} \to \{0,1\} \times [\lambda]$ is a $\mathsf{PRF}$, then ASA $A_{P_1}$ in Fig. 12 is strongly undetectable and secretly recoverable. For any $\mathsf{PPT}$ $\mathcal{D}$ we have*

$$Adv^{\mathsf{sUDET}}_{D,P_1,A_{P_1}}(\lambda) \leq 2Adv^{\mathsf{PRF}}_{\mathcal{F}_{F_1},F_1}(\lambda) + negl(\lambda) \tag{3}$$

*where $\mathcal{F}_{F_1}$ is a $\mathsf{PRF}$ adversary attacking $F_1$.*

*Proof.* Let $G_0$ be the original detection game $\mathsf{sUDET}^{\mathcal{D}}_{P_1,A_{P_1}}$, let $W_i$ denote the event that $\mathcal{D}$ returns correct $c'$ in game $G_i$, from the definition we have

$$Adv^{\mathsf{sUDET}}_{D,P_1,A_{P_1}}(\lambda) = |2\Pr[W_0] - 1|.$$

Let $G_1$ be the same as $G_0$ except that it implements $F_1$ with a lazily sampled random function. Then we construct a $\mathsf{PRF}$ adversary $\mathcal{F}_{F_1}$ attacking $F_1$ that simulates $G_0$ and $G_1$ for the detector $\mathcal{D}$. $\mathcal{F}_{F_1}$ provides leakage function $L_1$ and simulates the oracle for $\mathcal{D}$ as follow.

Upon query $(1, x_1, h_{1,1})$ when $c = 0$, $\mathcal{F}_{F_1}$ runs $\widetilde{P_1}^{A_1}$ except that receives $(b, l)$ from its challenger. For other queries, $\mathcal{F}_{F_1}$ simulates the oracle following the description. Finally $\mathcal{F}_{F_1}$ outputs 1 when $D$ returns $c' = c$.

When $(b, l)$ returns to $\mathcal{F}_{F_1}$ is the output of $F_1(ek, a)$, the game simulated by $D_F$ is exactly $G_0$, otherwise is $G_1$ from the view of the detector $D$. Therefore, we have

$$|\Pr[W_0] - \Pr[W_1]| \leq Adv^{\mathsf{PRF}}_{\mathcal{F}_{F_1},F_1}(\lambda).$$

Let $G_2$ be the same as $G_1$ except that the lazily sampled random function is replaced by fully random sampling of $(b, l)$. We must bound the probability that a particular value of $a$ will be repeated (call this event *bad*) during the game by substituting the lazily sampled random function with true random sampling. Suppose that $a$ is generated $s$ times in total, the probability of *bad* occurring is therefore bounded by $\binom{s}{2} \cdot \frac{1}{q} \leq \frac{s^2}{2q} \leq \frac{s^2}{2^{k+1}}$ where $q$ is the order of $\mathbb{G}$. Thus we have

$$|\Pr[W_1] - \Pr[W_2]| \leq negl(\lambda).$$

Besides, we notice that in $G_2$ the implementation of $\widetilde{P_1}^{A_1}$ is the same as that of $P_1^{A_1}$ from the view of $\mathcal{D}$. We have

$$\Pr[W_2] = \frac{1}{2}.$$

Putting all above together Eq. 4.3 holds.

As for the secret recoverability of $A_{P_2}$, the subversion adversary receives transcripts from $\mathcal{O}_{\mathsf{trans}}$ of at least $s \geq \lambda \cdot ln(\lambda)$ consecutive runs of $\widetilde{P_1}$, then recovers each bit of $\mathsf{sk}_1$ following the description of $\mathsf{Ext}$. And according to the analysis of coupon collection problem, $s$ transcripts are enough to recover every bit of $\mathsf{sk}_1$ with overwhelming probability [BJK15]. □

### 4.2.2 ASA against $\Pi_{sign}.P_2$

We present an ASA against $P_2$ in $\Pi_{sign}$ that maximizes the utilization of multiple sources of randomness within $P_2$ to improve subversion efficiency. More specifically, we highlight the distinction between ASAs against protocols and traditional ASAs against schemes by employing a global attack state-maintaining substitution attack (*collaborative substitution attack*) against $P_2$. In essence, by maintaining a global attack state, we can fully and efficiently leverage as much randomness as possible within $P_2$'s algorithms.

Fig. 13 depicts a symmetric collaborative ASA $A_{P_2}$ against $P_2$ in $\Pi_{sign}$. $A_{P_2} = (\mathsf{Gen}, \widetilde{\mathsf{P_2}}, \mathsf{Ext})$ where $\widetilde{P_2} = (\widetilde{P_2}^{A_1}, \widetilde{P_2}^{A_2})$ is a subverted party of $P_2$. $\widetilde{P_2}^{A_1}(ek_{2,1}, x_2, h_{2,1}, \sigma_{2,0})$ is the same as $P_2^{A_1}(x_2, h_{2,1})$ except that a global attack state $\sigma_{2,0}$ is maintained, and a PRF $F : \mathcal{K} \times \{0,1\}^* \to \mathbb{Z}_q$ is used in the generation of $k_2$, and the sub-module $\mathsf{NIZKPoK.P}$ is replaced by $\widetilde{\mathsf{stNIZKPoK.P}}$.

$\widetilde{\mathsf{stNIZKPoK.P}}(x, w, ek, aux, \sigma_p)$ is the same as $\mathsf{NIZKPoK.P}(x, w)$ except that it uses a PRF $F : \mathcal{K} \times \{0,1\}^* \to \mathbb{Z}_q$ to generate $t$. For the input of $\widetilde{P_1}^{A_1}$, we have $ek_{2,1} \in \mathcal{K}$. The leakage function $L_2(x_2, st_2, \sigma_2)$ returns $\mathsf{sk}_2$ in $x_2$. For the input of $\widetilde{\mathsf{stNIZKPoK.P}}$, we have $ek_{2,1} \in \mathcal{K}, aux := \mathsf{sk}_2$, and $\sigma_p = \sigma_{2,0}$ $\widetilde{P_2}^{A_2}(ek_{2,2}, x_2, h_{2,2}, st_{2,2}, \sigma_{2,0})$ is the same as $P_2^{A_2}(x_2, h_{2,2}, st_{2,2})$ except that a global attack state $\sigma_{2,0}$ is maintained, and a PRF $F' : \mathcal{K} \times \{0,1\}^* \to \{0,1\}$ is used to test whether the output $w$ by applying $F_1$ on $c_3$ satisfies $\mathsf{sk}_2[i] = b$ where $i$ is a part of $\sigma_{2,0}$.

**Theorem 4.4.** *Assume that $F : \mathcal{K} \times \{0,1\}^* \to \mathbb{Z}_q$ and $F' : \mathcal{K} \times \{0,1\}^* \to \{0,1\}$ are two* PRFs*, then ASA $A_{P_2}$ in Fig. 13 is undetectable and secretly recoverable. Specifically, the detection advantage of any* PPT *detector $\mathcal{D}$ can be bounded as*

$$Adv^{\mathsf{UDET}}_{\mathcal{D},P_2,A_{P_2}}(\lambda) \leq 2Adv^{\mathsf{PRF}}_{\mathcal{F},F}(\lambda) + 2Adv^{\mathsf{PRF}}_{\mathcal{F}',F'}(\lambda) + negl(\lambda) \tag{4}$$

*where $\mathcal{F}$ and $\mathcal{F}'$ are two* PRF *adversary attacking $F$ and $F'$ respectively.*

*Proof.* Let $G_0$ be the original detection game UDET defined in Fig. 4. By the definition, we have $Adv^{\mathcal{D}}_{P_2,A_{P_2}}(\lambda) = |2\Pr[W_0] - 1|$.

Let $G_1$ be the same as $G_0$ except that it implements $F$ in $\widetilde{P_2}^{A_1}$ with a lazily sampled random function. Then we construct a PRF adversary $\mathcal{F}_F$ attacking $F$ that simulates $G_0$ and $G_1$ for the detector $\mathcal{D}$. $\mathcal{F}_F$ provides leakage function $L_2$ and simulates the oracle for $\mathcal{D}$ as follow.

Upon query $(1, x_2, h_{2,1})$ when $c = 0$, $\mathcal{F}_F$ runs $\widetilde{P_2}^{A_1}$ except that receives $k_2$ and $t$ from its challenger. For other queries, $\mathcal{F}_F$ simulates the oracle following the description. Finally $\mathcal{F}_F$ outputs 1 when $D$ returns $c' = c$. Follow the proof of Theorem. 4.3 we have

$$|\Pr[W_0] - \Pr[W_1]| \leq Adv^{\mathsf{PRF}}_{\mathcal{F}_F,F}(\lambda),$$

where $\mathcal{F}_F$ is a PRF adversary attacking function $F$.

Let $G_2$ be the same as $G_1$ except that $F'$ in $\widetilde{P_2}^{A_2}$ is implemented with a lazily sampled random function. Let $G_3$ be the same as $G_2$ except that the lazily sampled random function $F$ is replaced by fully random sampling of $k_2$ and $t$. Let $G_4$ be the same as $G_3$ except that the lazily sampled random function $F'$ is replaced by fully random sampling of $w$. Follow the proof of Theorem. 4.3 we have

$$|\Pr[W_1] - \Pr[W_2]| \leq Adv^{\mathsf{PRF}}_{\mathcal{F}_{F'},F'}(\lambda),\ |\Pr[W_2] - \Pr[W_3]| \leq negl(\lambda),$$
$$|\Pr[W_3] - \Pr[W_4]| \leq negl(\lambda) \text{ and } \Pr[W_4] = 1/2,$$

where $\mathcal{F}_{F'}$ is a PRF adversary attacking $F'$. Putting all above together Eq. 4.4 holds.

$\widetilde{P_2}^{A_1}(ek_{2,1}, x_2, h_{2,1}, \sigma_{2,0})$

$\sigma_{2,0} \leftarrow \epsilon$
**if** $\sigma_{2,0} = \epsilon$ **then**
  $i \leftarrow 0,\ k_2 \leftarrow\$ \mathbb{Z}_q$
**else**
  $(i, \tau) \leftarrow \sigma_{2,0}$
  $k_2 \leftarrow F(ek_{2,1}, \tau, \mathsf{sk}_2[i])$
$R_2 \leftarrow k_2 \cdot G$
$i \leftarrow i + 1 \bmod l_{\mathsf{sk}_2}, \tau \leftarrow R_2$
$\sigma_{2,0} \leftarrow (i, \tau)$
$(\pi_2, \sigma_{2,0}) \leftarrow\$ \widetilde{\mathsf{stNIZKPoK.P}}(R_2, k_2, ek_{2,1}, \mathsf{sk}_2, \sigma_{2,0})$
$st_{2,2} \leftarrow k_2 || R_2$
**return** $(\pi_2, st_{2,2}, \sigma_{2,0})$

$\widetilde{P_2}^{A_2}(ek_{2,2}, x_2, h_{2,2}, st_{2,2}, \sigma_{2,0})$

...
$(i, \tau) \leftarrow \sigma_{2,0}$
**repeat**
$\rho \leftarrow \mathcal{Z}_{q^2}, \tilde{r} \leftarrow \mathbb{Z}_{N^*}$
$c_1 \leftarrow \mathsf{Paillier.Enc}(\mathsf{pk}_e, \rho \cdot q + [k_2^{-1} \cdot m \bmod q; \tilde{r}])$
$v \leftarrow k_2^{-1} \cdot r \cdot \mathsf{sk}_2 \bmod q$
$c_2 \leftarrow v \otimes c_{key},\ c_3 \leftarrow c_1 \oplus c_2$
$w \leftarrow F'(ek_{2,2}, c_3)$
**until** $\mathsf{sk}_2[i] = w$
$i \leftarrow i + 1 \bmod l_{\mathsf{sk}_2},\ \tau \leftarrow c_3$
$\sigma_{2,0} \leftarrow (i, \tau),\ st_{2,3} \leftarrow st_{2,2}$
**return** $(c_3, st_{2,3}, \sigma_{2,0})$

$\widetilde{\mathsf{stNIZKPoK.P}}(x, w, ek, aux, \sigma_p)$

**if** $\sigma_p = \epsilon$ **then** $i \leftarrow 0, t \leftarrow\$ \mathbb{Z}_q$
**else** $(i, \tau) \leftarrow \sigma_p,\ t \leftarrow F(ek, \tau, aux[i])$
$a \leftarrow t \cdot G,\ i \leftarrow i + 1 \bmod l_{aux},\ \tau \leftarrow a$
$\sigma_p \leftarrow (i, \tau), e \leftarrow H(a||x)$
$z \leftarrow t + e \cdot w, \pi \leftarrow (x, e, z)$
**return** $(\pi, \sigma_p)$

$\mathsf{Gen}(1^k)$

$ek_{2,1}, ek_{2,2} \leftarrow\$ \mathcal{K}$
$EK_2 = XK_2 = \{ek_{2,1}, ek_{2,2}\}$
$\mathsf{sk}_2 := L_2(x_2, st_2, \sigma_2)$
**return** $(EK_2, XK_2, L_2)$

$\mathsf{Ext}^{\mathcal{O}_{\mathsf{trans}}}(XK_2)$

$T_{m_{2,1}, m_{2,1}} := \{(R_2^1, a^1, c_3^1), \cdots, (R_2^s, a^s, c_3^s)\}$
$i, j := 1$
**for** $i \in [s]$ **do**
  $\mathsf{sk}_2[l] = 1$
  **if** $F(ek_{2,1}, R_2^i, 0) \cdot G = a^i$ **then**
    $\mathsf{sk}_2[l] = 0$
  $l \leftarrow l + 1 \bmod l_{\mathsf{sk}_2}$
  $\mathsf{sk}_2[l] = F'(ek_{2,2}, c_3^i)$
  $l \leftarrow l + 1 \bmod l_{\mathsf{sk}_2}$
  $(b, l) \leftarrow F_1(ek_{1,1}, a^i), \mathsf{sk}_2[l] = b$
  **for** $i > 1$ **do**
    $\mathsf{sk}_2[l] = 1$
    **if** $F(ek_{2,1}, c_3^{i-1}, 0) \cdot G = R^i$ **then**
      $\mathsf{sk}_2[l] = 0$
  $l \leftarrow l + 1 \bmod l_{\mathsf{sk}_2}, i \leftarrow i + 1$
**return** $\mathsf{sk}_2$

Figure 13: Stateful ASA against $P_2$ in $\Pi_{sign}$.

Regarding the secret recoverability of $A_{P_2}$, the subversion adversary receives transcripts (from $\mathcal{O}_{\mathsf{trans}}$) of the first $s$ consecutive runs of $\widetilde{P_2}$ where $s \geq \lfloor \frac{l_{\mathsf{sk}_2}+1}{3} \rfloor$, then recover each bit of $\mathsf{sk}_2$ following the description of $\mathsf{Ext}$. In one execution of $\widetilde{P_2}$, three bits of $\mathsf{sk}_2$ are leaked within two messages on average. Therefore, $s$ transcripts are enough to recover every bit of $\mathsf{sk}_2$ with high probability. □

# 5 Conclusion

In this work, we present an enhanced model for Algorithm Substitution Attacks (ASAs) tailored to cryptographic protocols. Our model provides a modular formalization of subverted parties, allowing for a detailed categorization of algorithms into normal and attack states, with further subdivisions into global and local states to describe complex subversion attacks accurately. We introduce a fine-grained definition of undetectability with different strengths, enabling the detector to access varying types of information. To illustrate the practicality of our model, we apply it to a two-party ECDSA protocol and demonstrate several ASAs targeting the parties. Moreover, the several concrete attacks we present illustrate the diversity of ASA against cryptographic protocols. The stateless ASA against $P_2$ in $\Pi_{kgen}$ cleverly utilizes multiple random numbers within $P_2^{A_1}$ to achieve a highly efficient attack effect of leaking secret key share in one single execution. Notably, similar efficient stateless attacks are not commonly found in existing works. Therefore, our work aims to improve the understanding of the various possibilities and efficiencies of ASAs in the context of cryptographic protocols, thus advancing defense efforts such as designing subversion-resilliant cryptographic protocols.

**Acknowledgements.** We would like to thank all anonymous reviewers for their valuable comments. This work is supported in part by the National Natural Science Foundation of China (Grant No.62122092, No.62202485, No.62032005).

# References

[AFMV19] G. Ateniese, D. Francati, B. Magri, and D. Venturi. Public immunization against complete subversion without random oracles. In *ACNS 19*, *LNCS* 11464, pages 465–485. Springer, Heidelberg, June 2019.

[AMV15] G. Ateniese, B. Magri, and D. Venturi. Subversion-resilient signature schemes. In *ACM CCS 2015*, pages 364–375. ACM Press, October 2015.

[AP19a] M. Armour and B. Poettering. Substitution attacks against message authentication. *IACR Trans. Symm. Cryptol.*, 2019(3):152–168, 2019.

[AP19b] M. Armour and B. Poettering. Subverting decryption in aead. In *Cryptography and Coding: 17th IMA International Conference, IMACC 2019, Oxford, UK, December 16–18, 2019, Proceedings 17*, pages 22–41. Springer, 2019.

[BCJ21] P. Bemmann, R. Chen, and T. Jager. Subversion-resilient public key encryption with practical watchdogs. LNCS, pages 627–658. Springer, Heidelberg, 2021.

[BH15] M. Bellare and V. T. Hoang. Resisting randomness subversion: Fast deterministic and hedged public-key encryption in the standard model. In *EUROCRYPT 2015, Part II*, *LNCS* 9057, pages 627–656. Springer, Heidelberg, April 2015.

[BJK15] M. Bellare, J. Jaeger, and D. Kane. Mass-surveillance without the state: Strongly undetectable algorithm-substitution attacks. In *ACM CCS 2015*, pages 1431–1440. ACM Press, October 2015.

[BL17] S. Berndt and M. Liskiewicz. Algorithm substitution attacks from a steganographic perspective. In *ACM CCS 2017*, pages 1649–1660. ACM Press, October / November 2017.

[BPR14] M. Bellare, K. G. Paterson, and P. Rogaway. Security of symmetric encryption against mass surveillance. In *CRYPTO 2014, Part I*, *LNCS* 8616, pages 1–19. Springer, Heidelberg, August 2014.

[BSKC19] J. Baek, W. Susilo, J. Kim, and Y.-W. Chow. Subversion in practice: How to efficiently undermine signatures. *IEEE Access*, 7:68799–68811, 2019.

[BWP+22] S. Berndt, J. Wichelmann, C. Pott, T.-H. Traving, and T. Eisenbarth. ASAP: Algorithm substitution attacks on cryptographic protocols. pages 712–726. ACM Press, 2022.

[CDN20] S. Chakraborty, S. Dziembowski, and J. B. Nielsen. Reverse firewalls for actively secure MPCs. In *CRYPTO 2020, Part II*, *LNCS* 12171, pages 732–762. Springer, Heidelberg, August 2020.

[CEJ23] B. Cogliati, J. Ethan, and A. Jha. Subverting telegram's end-to-end encryption. *IACR Transactions on Symmetric Cryptology*, pages 5–40, 2023.

[CGG+21] R. Canetti, R. Gennaro, S. Goldfeder, N. Makriyannis, and U. Peled. Uc non-interactive, proactive, threshold ecdsa with identifiable aborts. Cryptology ePrint Archive, Paper 2021/060, 2021. https://eprint.iacr.org/2021/060.

[CGPS21] S. Chakraborty, C. Ganesh, M. Pancholi, and P. Sarkar. Reverse firewalls for adaptively secure MPC without setup. LNCS, pages 335–364. Springer, Heidelberg, 2021.

[CGS23] S. Chakraborty, C. Ganesh, and P. Sarkar. Reverse firewalls for oblivious transfer extension and applications to zero-knowledge. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 239–270. Springer, 2023.

[CHY20] R. Chen, X. Huang, and M. Yung. Subvert KEM to break DEM: Practical algorithm-substitution attacks on public-key encryption. In *ASIACRYPT 2020, Part II*, *LNCS* 12492, pages 98–128. Springer, Heidelberg, December 2020.

[CMY+16] R. Chen, Y. Mu, G. Yang, W. Susilo, F. Guo, and M. Zhang. Cryptographic reverse firewall via malleable smooth projective hash functions. In *ASIACRYPT 2016, Part I*, *LNCS* 10031, pages 844–876. Springer, Heidelberg, December 2016.

[CRT+19] S. S. M. Chow, A. Russell, Q. Tang, M. Yung, Y. Zhao, and H.-S. Zhou. Let a non-barking watchdog bite: Cliptographic signatures with an offline watchdog. In *PKC 2019, Part I*, *LNCS* 11442, pages 221–251. Springer, Heidelberg, April 2019.

[DFP15] J. P. Degabriele, P. Farshim, and B. Poettering. A more cautious approach to security against mass surveillance. In *FSE 2015*, *LNCS* 9054, pages 579–598. Springer, Heidelberg, March 2015.

[DFS16] S. Dziembowski, S. Faust, and F.-X. Standaert. Private circuits iii: hardware trojan-resilience via testing amplification. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 142–153, 2016.

[DKLs18] J. Doerner, Y. Kondi, E. Lee, and a. shelat. Secure two-party threshold ECDSA from ECDSA assumptions. In *2018 IEEE Symposium on Security and Privacy*, pages 980–997. IEEE Computer Society Press, May 2018.

[DMS16] Y. Dodis, I. Mironov, and N. Stephens-Davidowitz. Message transmission with reverse firewalls—secure communication on corrupted machines. In *CRYPTO 2016, Part I*, *LNCS* 9814, pages 341–372. Springer, Heidelberg, August 2016.

[FM18] M. Fischlin and S. Mazaheri. Self-guarding cryptographic protocols against algorithm substitution attacks. In *CSF 2018Computer Security Foundations Symposium*, pages 76–90. IEEE Computer Society Press, 2018.

[GG18] R. Gennaro and S. Goldfeder. Fast multiparty threshold ECDSA with fast trustless setup. In *ACM CCS 2018*, pages 1179–1194. ACM Press, October 2018.

[HS21] P. Hodges and D. Stebila. Algorithm substitution attacks: state reset detection and asymmetric modifications. *IACR Transactions on Symmetric Cryptology*, pages 389–422, 2021.

[JHZ+23] H. Jiang, J. Han, Z. Zhang, Z. Ma, and H. Wang. Practical algorithm substitution attacks on real-world public-key cryptosystems. *IEEE Transactions on Information Forensics and Security*, 18:5069–5081, 2023.

[KL07] J. Katz and Y. Lindell. *Introduction to modern cryptography: principles and protocols*. Chapman and hall/CRC, 2007.

[LCWW18] C. Liu, R. Chen, Y. Wang, and Y. Wang. Asymmetric subversion attacks on signature schemes. In *ACISP 18*, *LNCS* 10946, pages 376–395. Springer, Heidelberg, July 2018.

[Lin17] Y. Lindell. Fast secure two-party ECDSA signing. In *CRYPTO 2017, Part II*, *LNCS* 10402, pages 613–644. Springer, Heidelberg, August 2017.

[MS15] I. Mironov and N. Stephens-Davidowitz. Cryptographic reverse firewalls. In *EUROCRYPT 2015, Part II*, *LNCS* 9057, pages 657–686. Springer, Heidelberg, April 2015.

[RTYZ16] A. Russell, Q. Tang, M. Yung, and H.-S. Zhou. Cliptography: Clipping the power of kleptographic attacks. In *ASIACRYPT 2016, Part II*, *LNCS* 10032, pages 34–64. Springer, Heidelberg, December 2016.

[RTYZ17] A. Russell, Q. Tang, M. Yung, and H.-S. Zhou. Generic semantic security against a kleptographic adversary. In *ACM CCS 2017*, pages 907–922. ACM Press, October / November 2017.

[RTYZ18] A. Russell, Q. Tang, M. Yung, and H.-S. Zhou. Correcting subverted random oracles. In *CRYPTO 2018, Part II*, *LNCS* 10992, pages 241–271. Springer, Heidelberg, August 2018.

[TY17] Q. Tang and M. Yung. Cliptography: post-snowden cryptography. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 2615–2616, 2017.

[WS11] A. Waksman and S. Sethumadhavan. Silencing hardware backdoors. In *2011 IEEE Symposium on Security and Privacy*, pages 49–63. IEEE Computer Society Press, May 2011.

[XAX+21] H. Xue, M. H. Au, X. Xie, T. H. Yuen, and H. Cui. Efficient online-friendly two-party ECDSA signature. pages 558–573. ACM Press, 2021.

[YY97a] A. Young and M. Yung. Kleptography: Using cryptography against cryptography. In *EUROCRYPT'97*, *LNCS* 1233, pages 62–74. Springer, Heidelberg, May 1997.

[YY97b] A. Young and M. Yung. The prevalence of kleptographic attacks on discrete-log based cryptosystems. In *CRYPTO'97*, *LNCS* 1294, pages 264–276. Springer, Heidelberg, August 1997.

# A Full Description of Lin-2ECDSA

Key Generation Sub-protocol $\Pi_{kgen}(1^\lambda)$

| **Party** $P_1$ | | **Party** $P_2$ |
|---|---|---|
| $\mathsf{sk}_1 \leftarrow\!\!\$\ \mathbb{Z}_q$ | | $\mathsf{sk}_2 \leftarrow\!\!\$\ \mathbb{Z}_q$ |
| $\mathsf{pk}_1 \leftarrow \mathsf{sk}_1 \cdot G$ | | $\mathsf{pk}_2 \leftarrow \mathsf{sk}_2 \cdot G$ |
| $\pi_1 = \mathsf{NIZKPoK.P}(\mathsf{pk}_1, \mathsf{sk}_1)$ | | $\pi_2 = \mathsf{NIZKPoK.P}(\mathsf{pk}_2, \mathsf{sk}_2)$ |
| $(com_1, dcom_1) = \mathsf{COM.G}(\pi_1)$ | | |
| | $com_1 \longrightarrow$ | |
| | $\pi_2 \longleftarrow$ | |
| **if** $\mathsf{NIZKPoK.V}(\pi_2) = \perp$ **then** | | |
| **return** $\perp$ | | |
| $(\mathsf{pk}_e, \mathsf{sk}_e) \leftarrow \mathsf{Paillier.Kgen}(1^\mathsf{k})$ | | |
| $c_{key} \leftarrow \mathsf{Paillier.Enc}(\mathsf{pk}_e, \mathsf{sk}_1)$ $\mathsf{pk}_e, \mathsf{pk}_1; (\mathsf{sk}_1, \mathsf{sk}_e))$ | | |
| | $\mathsf{pk}_e, c_{key}, dcom_1 \longrightarrow$ | |
| | prove in zero-knowledge that $\mathsf{pk}_e \in L_p$ and $(c_{key}, \mathsf{pk}_e, \mathsf{pk}_1) \in L_{PDL}$ | |
| $\mathsf{pk} \leftarrow \mathsf{sk}_1 \cdot \mathsf{pk}_2$ | | $\pi_1 \leftarrow dcom_1$ |
| **return** $(\mathsf{sk}_1, \mathsf{pk}_e, \mathsf{sk}_e, \mathsf{pk})$ | | **if** $\mathsf{NIZKPoK.V}(\pi_1) = \perp \vee$ $\mathsf{COM.V}(com_1,$ $dcom_1) = \perp$ **then** |
| | | **return** $\perp$ |
| | | $\mathsf{pk} \leftarrow \mathsf{sk}_2 \cdot \mathsf{pk}_1$ |
| | | **return** $(\mathsf{sk}_2, \mathsf{pk}_e, c_{key}, \mathsf{pk})$ |

Figure 14: $\Pi_{kgen}$ of Lin-2ECDSA where $L_P = \{(N; \phi(N)) | gcd(N, \phi(N)) = 1\}$ and $L_{PDL} = \{(c_{key}, \mathsf{pk}_e, \mathsf{pk}_1; (\mathsf{sk}_1, \mathsf{sk}_e) | \exists (\mathsf{sk}_1, r) s.t. c_{key} = \mathsf{Paillier.Enc}(\mathsf{pk}_e, \mathsf{sk}_1; r), \mathsf{pk}_1 = \mathsf{sk}_1 \cdot G \text{ and } \mathsf{sk}_1 \in \mathbb{Z}_q\}$

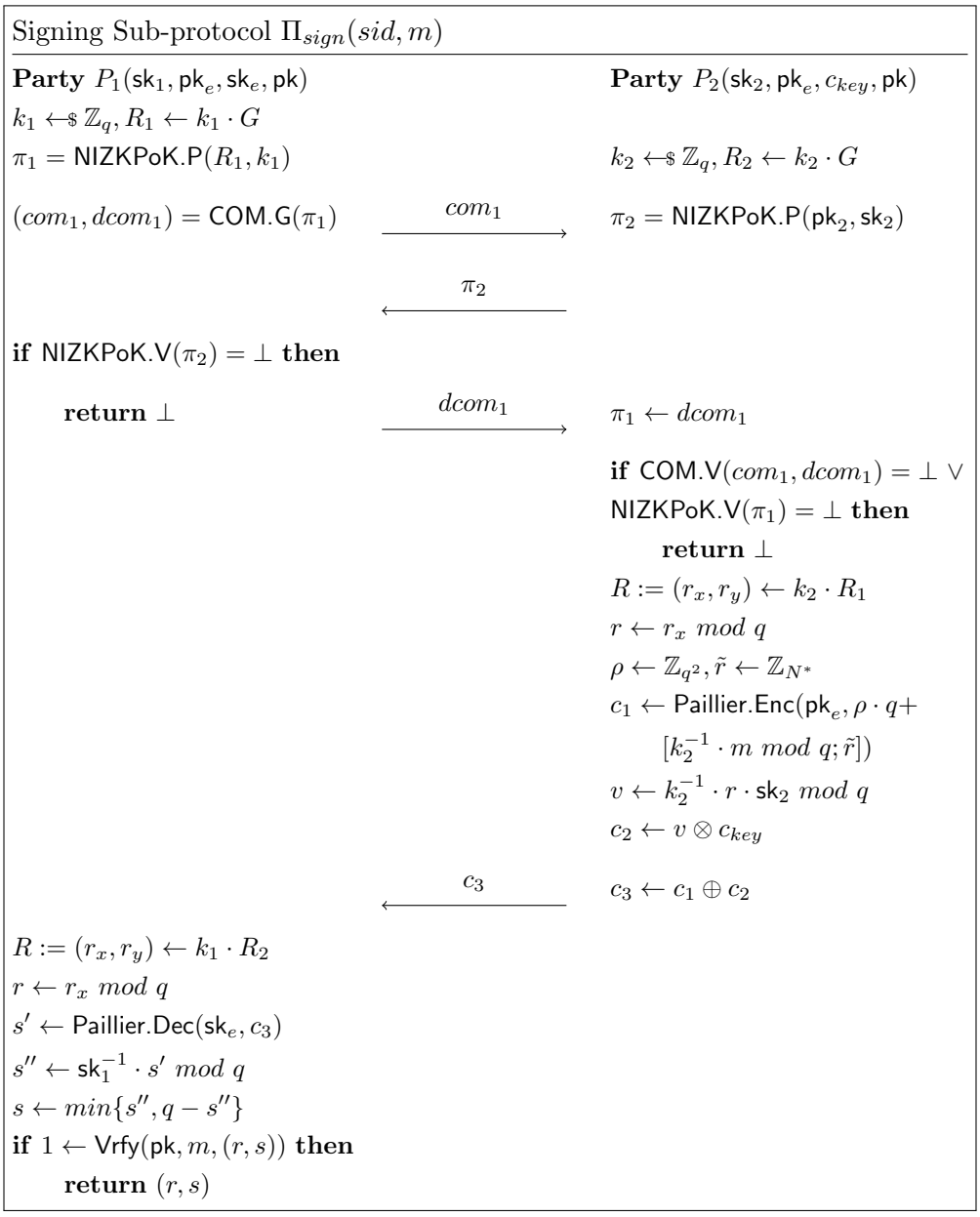


Figure 15: $\Pi_{sign}$ of Lin-2ECDSA