On the (in)security of ROS

Fabrice Benhamouda[1], Tancrède Lepoint[2], Michele Orrù[3], and Mariana Raykova[2]

[1] Algorand Foundation, fabrice.benhamouda@gmail.com
[2] Google, {tancrede,marianar}@google.com
[3] École Normale Supérieure, CNRS, PSL University, Paris, France, michele.orru@ens.fr

Abstract. We present an algorithm solving the ROS (Random inhomogeneities in a Overdetermined Solvable system of linear equations) problem in polynomial time for large enough dimensions ℓ. The algorithm implies polynomial-time attacks against blind signatures such as Schnorr and Okamoto–Schnorr blind signatures, threshold signatures such as the one from GJKR (when concurrent executions are allowed), and multisignatures such as CoSI and the two-round version of MuSig.

1 Introduction

The ROS (Random inhomogeneities in a Overdetermined Solvable system of linear equations) problem was introduced to prove the security of blind Schnorr signatures [Sch01, FPS20]. Given a prime number p and given access to a random oracle $\mathrm{H}_{\mathrm{ros}}$ with range in $\mathbb{Z}_p$, the ROS problem (in dimension ℓ) asks to find $(\ell+1)$ affine functions $\boldsymbol{\rho}_i$, $(\ell+1)$ bit strings $\mathsf{aux}_i \in \{0,1\}^*$ (with $i \in [0,\ell]$), and a vector $\mathbf{c} = (c_0, \ldots, c_{\ell-1})$ such that:

$$\mathrm{H}_{\mathrm{ros}}(\boldsymbol{\rho}_i, \mathsf{aux}_i) = \boldsymbol{\rho}_i(\mathbf{c}) \qquad \text{for all } i \in [0,\ell].$$

A formal description is provided in Figure 1. $\mathsf{Pgen}(1^\lambda)$ is a parameter generation algorithm that given as input the security parameter λ in unary outputs a prime p of length λ. We abuse notations and $\boldsymbol{\rho}_i$ denotes both the vector $\boldsymbol{\rho}_i = (\rho_{i,0}, \ldots, \rho_{i,\ell}) \in \mathbb{Z}_p^{\ell+1}$ and the corresponding affine function $\boldsymbol{\rho}_i(\mathbf{x}) = \sum_{j=0}^{\ell-1} \rho_{i,j} \cdot x_j + \rho_{i,\ell}$ (where $\mathbf{x} = (x_0, \ldots, x_\ell)$).

Game $\mathrm{ROS}_{\mathsf{GrGen},\mathsf{A},\ell}(\lambda)$	Oracle $\mathrm{H}_{\mathrm{ros}}(\boldsymbol{\rho}, \mathsf{aux})$
$p \leftarrow \mathsf{Pgen}(1^\lambda)$	**if** $\mathsf{T}_{\mathrm{ros}}[\boldsymbol{\rho}, \mathsf{aux}] = \perp$ **then**
$\mathsf{T}_{\mathrm{ros}} := [\,]$	$\quad \mathsf{T}_{\mathrm{ros}}[\boldsymbol{\rho}, \mathsf{aux}] \leftarrow_{\$} \mathbb{Z}_p$
$\left((\boldsymbol{\rho}_i, \mathsf{aux}_i)_{i\in[0,\ell]}, (c_j)_{j\in[0,\ell-1]}\right) \leftarrow \mathsf{A}^{\mathrm{H}_{\mathrm{ros}}}(p)$	**return** $\mathsf{T}_{\mathrm{ros}}[\boldsymbol{\rho}, \mathsf{aux}]$
return $\big(\forall i \neq j \in [0,\ell],\ \ (\boldsymbol{\rho}_i, \mathsf{aux}_i) \neq (\boldsymbol{\rho}_j, \mathsf{aux}_j)$	
$\qquad \wedge\ \forall i \in [0,\ell],\ \ \sum_{j=0}^{\ell-1} c_j \rho_{i,j} + \rho_{i,\ell} = \mathrm{H}_{\mathrm{ros}}(\boldsymbol{\rho}_i, \mathsf{aux}_i)\big)$	

Fig. 1. The $\mathrm{ROS}_{\mathsf{GrGen},\mathsf{A},\ell}(\lambda)$ game.

Actual security of ROS has been often measured with respect to Wagner's attack [Wag02], which generalizes the birthday bound and provides an algorithm for computing collisions with sub-exponential complexity $O((\ell+1) \cdot 2^{\lambda/(1+\lfloor \lg(\ell+1) \rfloor)})$. In this work, for any $\ell > \log p$, we present an (expected) polynomial-time adversary A that always wins the game $\mathrm{ROS}_{\mathsf{GrGen},\mathsf{A},\ell}(\lambda)$.

Theorem 1. *Let* GrGen *be a group generator algorithm. If* $\ell > \log p$*, there exists a (probabilistic) adversary that runs in expected polynomial time such that* $\mathsf{Adv}^{\mathrm{ros}}_{\mathsf{GrGen},\mathsf{A},\ell}(\lambda) = 1$.

Other formulations. Alternative formulations of ROS have been given in the past. Fuchsbauer et al. [FPS20, Fig. 7] present a variant of ROS the game with linear instead of affine functions $\boldsymbol{\rho}_i$ (i.e., where $\rho_{i,\ell} = 0$). Hauck et al. [HKL19, Fig. 3] allow only for linear functions, and do not allow for auxiliary information aux within $\mathrm{H}_{\mathrm{ros}}$ (i.e., where $\mathsf{aux}_i = \perp$).[4] These formulations are all equivalent.

[4] Our attacks only apply to the case where the scalar set $\mathcal{S}$ is a finite field.

First, any adversary A for an ROS with affine functions as per Figure 1 can be reduced to an adversary B for ROS with linear functions as per [FPS20]: B runs A and for every query of the form $((\rho_{i,0},\ldots,\rho_{i,\ell}),\mathsf{aux}_i)$ to the oracle $\mathrm{H}_{\mathrm{ros}}$ (made by A), it returns $\mathrm{H}_{\mathrm{ros}}((\rho_{i,0},\ldots,\rho_{i,\ell-1}),(\rho_{i,\ell}\|\mathsf{aux}_i))-\rho_{i,\ell}$. Finally, B modifies accordingly the solution output by A by concatenating $\rho_{i,\ell}$ to the corresponding aux_i.
Second, any adversary A for an ROS with linear functions can be reduced to an adversary B for ROS with linear functions and without auxiliary information as per [HKL19]. We assume without loss of generality that A never makes twice the same query. Then B runs A and for every query of the form $((\rho_{i,0},\ldots,\rho_{i,\ell-1},0),\mathsf{aux}_i)$ to the oracle (made by A), it picks a random scalar $r\in\mathbb{Z}_p^*$ and returns $\mathrm{H}_{\mathrm{ros}}((r\cdot\rho_{i,0},\ldots,r\cdot\rho_{i,\ell-1}),\perp)\cdot r^{-1} \bmod p$. When A outputs a solution $(\boldsymbol{\rho}_i,\mathsf{aux}_i)_{i\in[0,\ell]},(c_j)_{j\in[0,\ell-1]}$, B outputs $(r\cdot\boldsymbol{\rho}_i)_{i\in[0,\ell]},(c_j)_{j\in[0,\ell-1]}$. The simulation of the oracle $\mathrm{H}_{\mathrm{ros}}$ is perfect unless there is a collision in the scalar r, which happens with negligible probability in λ.

2 Attack

We construct an adversary for $\mathrm{ROS}_{\mathsf{A},\mathsf{GrGen},\ell}(\lambda)$, where $\ell>\log p$. Recall that to simplify the description of the attack, we use a polynomial formulation of ROS, i.e., we represent vectors $\boldsymbol{\rho}_i=(\rho_{i,0},\ldots,\rho_{i,\ell})$ as linear multivariate polynomials in $\mathbb{Z}_p[x_0,\ldots,x_{\ell-1}]$:

$$\boldsymbol{\rho}_i(x_0,\ldots,x_{\ell-1})=\rho_{i,0}x_0+\cdots+\rho_{i,\ell-1}x_{\ell-1}+\rho_{i,\ell}\ . \tag{1}$$

The goal for the adversary A is to output $(\boldsymbol{\rho}_i,\mathsf{aux}_i)_{i\in[0,\ell]}$ and $\mathbf{c}=(c_0,\ldots,c_{\ell-1})$ such that:

$$\boldsymbol{\rho}_i(\mathbf{c})=\mathrm{H}_{\mathrm{ros}}(\boldsymbol{\rho}_i,\mathsf{aux}_i)\qquad\text{for all } i\in[0,\ell].$$

Define:

$$\boldsymbol{\rho}_i\coloneqq x_i\qquad\text{for } i=0,\ldots,\ell-1,$$

and make two queries with $\mathsf{aux}=0$ and $\mathsf{aux}=1$ to get two hash values: $c_i^{\mathsf{aux}}=\mathrm{H}_{\mathrm{ros}}(\boldsymbol{\rho}_i,\mathsf{aux})$. Then, let:

$$x_i'\coloneqq\frac{x_i-c_i^0}{c_i^1-c_i^0}$$

for all $i=0,\ldots,\ell-1$. If $c_i^0=c_i^1$, redo the above with two different values of aux for this $\boldsymbol{\rho}_i$.[5] We remark that, if $x_i=c_i^b$, then $x_i'=b$ (for $b=0,1$). Define $\boldsymbol{\rho}_\ell\coloneqq\sum_{i=0}^{\ell-1}2^ix_i'$, and query $c_\ell\coloneqq\mathrm{H}_{\mathrm{ros}}(\boldsymbol{\rho}_\ell,\perp)$. Finally, write c_ℓ in binary as:

$$c_\ell=\sum_{i=0}^{\ell-1}2^ib_i\pmod p.$$

(As $2^\ell>p$, it is possible to write $c_{\ell+1}$ this way, and this implicitly defines the b_i's.) Define $b_\ell\coloneqq\perp$. A outputs: $(\boldsymbol{\rho}_0,b_0),\ldots,(\boldsymbol{\rho}_\ell,b_\ell)$ and $\mathbf{c}\coloneqq(c_0^{b_0},\ldots,c_{\ell-1}^{b_{\ell-1}})$. We have indeed that, for $i\in[0,\ell-1]$, $\boldsymbol{\rho}_i(\mathbf{c})=c_i^{b_i}=\mathrm{H}_{\mathrm{ros}}(\boldsymbol{\rho}_i,b_i)$ and therefore:

$$\boldsymbol{\rho}_\ell(\mathbf{c})=\sum_{i=0}^{\ell-1}2^ix_i'(\mathbf{c})=\sum_{i=0}^{\ell-1}2^ib_i=c_\ell.$$

Remark 2. The attack does not apply to modified ROS [FPS20, Sec. 5].

3 Affected blind signatures

3.1 Schnorr blind signatures

Let $\mathbb{G}$ be a cyclic group of prime order p. We use the additive notation for the group law. A Schnorr blind signature [Sch01, FPS20] for a message $m\in\{0,1\}^*$ consists of a pair $(R,s)\in\mathbb{G}\times\mathbb{Z}_p$ such that $sG-cX=R$, where $c\coloneqq\mathsf{H}(R,m)$ and $X\in\mathbb{G}$ is the verification key. A formal description of the protocol can be found in [FPS20, Fig. 6]. We use the notation from [FPS20].

[5] This step is the reason why the algorithm is expected polynomial time instead of polynomial time. Note that since $\mathsf{aux}\in\{0,1\}^*$, there will always be two values $\mathsf{aux}\in\{0,1\}^*$ so that $c_i^0\neq c_i^1$.

Attack. We construct a probabilistic (expected) polynomial-time adversary A that is able to produce $\ell+1$ signatures after opening $\ell \geq \lceil \log p \rceil = \lambda$ parallel sessions. A selects a message $m_\ell \in \{0,1\}^*$ for which a signature will be forged. It opens ℓ parallel sessions, querying $\textsc{Sign}_0()$ and receiving $\mathbf{R} = (R_0, \ldots, R_{\ell-1}) \in \mathbb{G}^\ell$. Let m_i^b be a random message and $c_i^b \coloneqq \mathsf{H}(R_i, m_i^b)$ for $i \in [0, \ell-1]$ and $b \in \{0,1\}$. If $c_i^0 = c_i^1$, two different messages m_i^0 and m_i^1 are chosen until $c_i^0 \neq c_i^1$. Define $\boldsymbol{\rho}_\ell \coloneqq \sum_i 2^i x'_i$ as per Section 2, that is:

$$\boldsymbol{\rho}_\ell(x_0, \ldots, x_{\ell-1}) \coloneqq \sum_{i=0}^{\ell-1} 2^i \cdot \frac{x_i - c_i^0}{c_i^1 - c_i^0} = \sum_{i=0}^{\ell-1} \rho_{\ell,i} x_i + \rho_{\ell,\ell} \ . \tag{2}$$

Let $R_\ell \coloneqq \boldsymbol{\rho}_\ell(\mathbf{R}) - \rho_{\ell,\ell} \cdot X$. Define $c_\ell \coloneqq \mathsf{H}(R_\ell, m_\ell) = \sum_{i=0}^{\ell-1} 2^i b_i$ and let $\mathbf{c} = (c_0^{b_0}, \ldots, c_{\ell-1}^{b_{\ell-1}})$. Complete the ℓ opened sessions querying $\textsc{Sign}_1(i, c_i^{b_i})$, for $i \in [0, \ell-1]$. The adversary thus obtains responses $\mathbf{s} \coloneqq (s_0, \ldots, s_{\ell-1}) \in \mathbb{Z}_p^\ell$ satisfying:

$$s_i G - c_i^{b_i} X = R_i, \qquad \text{for } i \in [0, \ell-1].$$

Let $s_\ell \coloneqq \boldsymbol{\rho}_\ell(\mathbf{s})$. Then $(m_\ell, (R_\ell, s_\ell))$ is a valid forgery. In fact, by perfect correctness of Schnorr blind signatures, we have:

$$\begin{aligned} R_\ell = \boldsymbol{\rho}_\ell(\mathbf{R}) - \rho_{\ell,\ell} X &= \sum_{i=0}^{\ell-1} \rho_{\ell,i} \cdot R_i + \rho_{\ell,\ell} \cdot (G - X) \\ &= \sum_{i=0}^{\ell-1} \rho_{\ell,i} \cdot (s_i G - c_i^{b_i} X) + \rho_{\ell,\ell} \cdot (G - X) \\ &= \boldsymbol{\rho}_\ell(\mathbf{s}) \cdot G - \boldsymbol{\rho}_\ell(\mathbf{c}) \cdot X \\ &= s_\ell G - c_\ell X, \end{aligned}$$

where $c_\ell = \mathsf{H}(R_\ell, m_\ell) = \boldsymbol{\rho}_\ell(\mathbf{c})$ by Equation (2). Let $m_i \coloneqq m_i^{b_i}$ for $i \in [0, \ell-1]$. The adversary outputs $(m_i, (R_i, s_i))$ for $i \in [0, \ell]$.

Remark 3. The attack does not apply to the Clause Blind Schnorr signature scheme [FPS20, Sec. 5], proved secure under the hardness of the modified ROS problem.

3.2 Okamoto–Schnorr blind signatures

An Okamoto–Schnorr blind signature [PS00, Sch01] for a message m consists of a tuple $(R, s, t) \in \mathbb{G} \times \mathbb{Z}_p^2$ such that $sG + tH - cX = R$, where $c \coloneqq \mathsf{H}(R, m)$. The attack of the previous section directly extends to Okamoto–Schnorr signatures: A operates exactly as before until Equation (2). Then, the forgery is constructed as:

$$\big(R_\ell \coloneqq \boldsymbol{\rho}_\ell(\mathbf{R}) + \rho_{\ell,\ell} H - \rho_{\ell,\ell} X, \quad s_\ell \coloneqq \boldsymbol{\rho}_\ell(\mathbf{s}), \quad t_\ell \coloneqq \boldsymbol{\rho}_\ell(\mathbf{t})\big).$$

We note that this does not contradict the analysis of Stern and Pointcheval [PS00], whose security was reduced to $\mathrm{DLOG}_{\mathsf{GrGen},\mathsf{A}}(\lambda)$ for a $\mathsf{polylog}(\lambda)$ number of queries.

4 Affected multisignatures

4.1 CoSi

A multi-signature scheme allows a group of signers $S_1, \ldots, S_n$, each having their own key pair $(\mathsf{pk}_j, \mathsf{sk}_j)$, to collaboratively sign a single message m. CoSi is a multi-signature scheme introduced by Syta et al. [STV+16] with a two-round signing protocol, the signers are organized in a tree and where S_1 is the root of the tree. A signature for a message $m \in \{0,1\}^*$ consists of a pair $(c, s) \in \mathbb{Z}_p^2$ such that $c = \mathsf{H}(sG - c \cdot \mathsf{pk}, m)$ and $\mathsf{pk} = \sum_{j=1}^n \mathsf{pk}_j \in \mathbb{G}$ is the aggregated verification key. A formal description of the protocol can be found in [DEFN18, Sec. 2.5]. We use the same notation as in [DEFN18], except that we use the additive notation xG instead of g^x.

Attack. We present an attack for a two-node tree where the attacker controls the root S_1. The attack can be extended to other settings, as in the attacks described in [DEFN18, Sec. 4.2]. The attack allows the signer S_1 to forge one signature for an arbitrary message $m_\ell \in \{0,1\}^*$ after performing $\ell \geq \lceil \log p \rceil = \lambda$ interactions with the honest signer S_2. Recall that $\mathsf{sk} = \mathsf{sk}_1 + \mathsf{sk}_2$ and $\mathsf{pk} = \mathsf{pk}_1 + \mathsf{pk}_2$.

S_1 opens ℓ parallel sessions with ℓ arbitrary distinct messages $m_0, \ldots, m_{\ell-1} \in \{0,1\}$. For each session, S_1 gets the commitments $t_i = r_i G$ from S_2 at the end of the first round of signing, samples two random values $r_{i,0}, r_{i,1}$ for each $i \in [0, \ell-1]$, defines $\bar{t}_i^0 = r_{i,0}G + t_i$ and $\bar{t}_i^1 = r_{i,1}G + t_i$, and computes $c_i^b = \mathsf{H}(\bar{t}_i^b, m_i)$. As usual, if $c_i^0 = c_i^1$, S_1 sample again $r_{i,0}$ and $r_{i,1}$ until $c_i^0 \neq c_i^1$. S_1 then defines the polynomial $\boldsymbol{\rho} \coloneqq \sum_{i=0}^{\ell-1} 2^i x_i/(c_i^1 - c_i^0)$, computes $t_\ell \coloneqq \boldsymbol{\rho}(t_0, \ldots, t_{\ell-1})$ and $c_\ell \coloneqq \mathsf{H}(t_\ell, m_\ell)$. S_1 computes $d_\ell = c_\ell - \boldsymbol{\rho}(c_0^0, \ldots, c_{\ell-1}^0)$ and writes this value in binary as $d_\ell = \sum_{i=0}^{\ell-1} 2^i b_i$. It then closes the ℓ sessions by using $\bar{t}_i = \bar{t}_i^{b_i}$ and $c_i = c_i^{b_i}$. At the last step of the signing sessions, S_1 obtains values $s_i = r_i + c_i \cdot \mathsf{sk}_2$ from S_2, and closes the sessions honestly using r_{i,b_i}. Next, S_1 concludes its forgery by defining $s_\ell \coloneqq \boldsymbol{\rho}(\mathbf{s}) + c_\ell \cdot \mathsf{sk}_1$: the pair (c_ℓ, s_ℓ) is a valid signature for m_ℓ. In fact:

$$
\begin{aligned}
s_\ell G - c_\ell \cdot \mathsf{pk} = (\boldsymbol{\rho}(\mathbf{s}) + c_\ell \cdot \mathsf{sk}_1)G - c_\ell \cdot \mathsf{pk} &= \sum_{i=0}^{\ell-1} \frac{2^i s_i}{c_i^1 - c_i^0} G - c_\ell \cdot \mathsf{pk}_2 \\
&= \sum_{i=0}^{\ell-1} \frac{2^i (r_i + c_i^{b_i} \cdot \mathsf{sk}_2)}{c_i^1 - c_i^0} G - c_\ell \cdot \mathsf{pk}_2 \\
&= \sum_{i=0}^{\ell-1} \frac{2^i r_i}{c_i^1 - c_i^0} G + \left(\sum_{i=0}^{\ell-1} \frac{2^i c_i^{b_i}}{c_i^1 - c_i^0} - c_\ell \right) \cdot \mathsf{pk}_2 \\
&= \sum_{i=0}^{\ell-1} \frac{2^i t_i}{c_i^1 - c_i^0} + \left(\sum_{i=0}^{\ell-1} 2^i b_i + \sum_{i=0}^{\ell-1} \frac{2^i c_i^0}{c_i^1 - c_i^0} - c_\ell \right) \cdot \mathsf{pk}_2 \\
&= \sum_{i=0}^{\ell-1} \frac{2^i t_i}{c_i^1 - c_i^0} + \underbrace{\left(\sum_{i=0}^{\ell-1} 2^i b_i + \boldsymbol{\rho}(c_0^0, \ldots, c_{\ell-1}^0) - c_\ell \right)}_{= d_\ell - d_\ell = 0} \cdot \mathsf{pk}_2 \\
&= \boldsymbol{\rho}(t_0, \ldots, t_{\ell-1}) = t_\ell \,,
\end{aligned}
$$

and $c_\ell = \mathsf{H}(t_\ell, m_\ell)$ by definition.

4.2 Two-round MuSig

The same technique (with some minor changes) can be applied to the two-round MuSig as initially proposed by Maxwell et al. [MPSW18a], as the main difference between CoSi and two-round MuSig is in how they avoid rogue-key attacks (how the public key is aggregated). Our attack does not apply to the updated MuSig that uses a 3-round signing algorithm [MPSW18b].

5 Affected threshold signatures

5.1 Gennaro et al.'s threshold signature

Gennaro, Jarecki, Krawczyk, Rabin proposed a threshold signature scheme based on Pedersen's distributed key generation (DKG) protocol in [GJKR07, Section 5.2]. At a very high level, Pedersen's DKG protocol allows to generate a random group element $X = \chi G$ so that its discrete logarithm χ is shared both additively and according to Feldman secret sharing [Fel87] scheme, between a set of "qualified" parties. For the attack we present below, all parties $\mathsf{P}_1, \ldots, \mathsf{P}_n$ (included the ones that are controlled by the adversary) will remain qualified.[6] We denote by χ_j the additive share of party P_j. We have $\chi = \sum_{j=1}^{n} \chi_j$. Importantly for the attack, the adversary controlling for example P_1 can see all the group elements $\chi_2 G, \ldots, \chi_n G$ and then can choose its value χ_1. This is due to the way the Feldman secret sharing is performed.

In the threshold signature scheme of Gennaro et al., the parties start the above key generation procedure to produce a verification key $\mathsf{pk} \coloneqq \mathsf{sk} \cdot G \in \mathbb{G}$, where the secret key sk is additively

[6] We do not use the fact that only a threshold $t+1$ of the parties are required to sign in our attack. We assume that all the parties come to sign, to simplify the description of the attack.

shared between the parties: each party P_j has an additive share sk_j, so that $\mathsf{sk} = \sum_{j=1}^{n} \mathsf{sk}_j$. A signature (R, s) for a message $m \in \{0,1\}^*$ is generated as follows. The participants run once again the distributed key generation protocol to produce a commitment $t = rG \in \mathbb{G}$, where r is additively shared between the parties: each party P_j has a share r_j, so that $r = \sum_{j=1}^{n} r_j$. Then, each party computes a share of the response:

$$s_j = r_j + c \cdot \mathsf{sk}_j, \qquad \text{where } c \coloneqq \mathsf{H}(t, m). \tag{3}$$

Let $s \coloneqq \sum_{j=1}^{n} s_j$. Then (c, s) is a valid signature on m. In fact:

$$sG = \sum_{j=1}^{n} r_j G + c \cdot \sum_{j=1}^{n} \mathsf{sk}_j \cdot G = t + c \cdot \mathsf{pk}, \tag{4}$$

where $c = \mathsf{H}(t, m)$.

Attack. Gennaro et al. proved the scheme to be secure in a standalone *non-concurrent* setting, where no two instances of the protocol can be run in parallel.
However, if the adversary is allowed to start $\ell \geq \lceil \log p \rceil$ sessions in parallel, we remark that the attack against CoSi in Section 4.1 can be directly adapted to attack Gennaro et al.'s threshold signature scheme for $n = 2$. Both use the fact that the adversary P_1 (or signer S_1 in CoSi) can see the commitment $t_2 = r_2 G$ of the honest party P_2 (or honest signed S_2) and only then choose r_1 that defines the commitment $t = r_1 G + t_2$. The generalization to any $n \geq 2$ is straightforward.

Scope of the attack. The attack is not an attack against Perdersen's DKG (i.e., JF-DKG from [GJKR07, Fig. 1]), but an attack against the proposed threshold signature scheme when instantiated with Pedersen's DKG. Furthermore, the attack does not work when Perdersen's DKG is replaced by the new DKG protocol from [GJKR07, Fig. 2].

5.2 Original version of FROST

Komlo and Goldberg FROST [KG20] proposed an extension of the above threshold signature scheme that was similarly affected by the above concurrent attack. On 19 July 2020, they updated the signing algorithm in a way that is no more susceptible to the above issue: each party now shares (D_j, E_j) and the commitment is computed as $R = \sum_j D_j + h_j E_j$, where $h_j \coloneqq \mathsf{H}((D_j, E_j, j)_{j \in [t]})$. We direct the reader to [KG20, Fig. 3] for a more detailed illustration of the problem and the fix.

6 Further impact

The following articles present reductions of their schemes to the ROS problem, and may not provide the expected security guarantee: blind anonymous group signatures [CFLW04]; blind identity-based signcryption [YW05]; blind signature schemes from bilinear pairings [CHYC05].

References

CFLW04. Tony K. Chan, Karyin Fung, Joseph K. Liu, and Victor K. Wei. Blind spontaneous anonymous group signatures for ad hoc groups. In *ESAS*, volume 3313 of *Lecture Notes in Computer Science*, pages 82–94. Springer, 2004.

CHYC05. Sherman S. M. Chow, Lucas Chi Kwong Hui, Siu-Ming Yiu, and K. P. Chow. Two improved partially blind signature schemes from bilinear pairings. In Colin Boyd and Juan Manuel González Nieto, editors, *ACISP 05*, volume 3574 of *LNCS*, pages 316–328. Springer, Heidelberg, July 2005.

DEFN18. Manu Drijvers, Kasra Edalatnejad, Bryan Ford, and Gregory Neven. On the provable security of two-round multi-signatures. Cryptology ePrint Archive, Report 2018/417, 2018. https://eprint.iacr.org/2018/417.

Fel87. Paul Feldman. A practical scheme for non-interactive verifiable secret sharing. In *28th FOCS*, pages 427–437. IEEE Computer Society Press, October 1987.

FPS20. Georg Fuchsbauer, Antoine Plouviez, and Yannick Seurin. Blind schnorr signatures and signed ElGamal encryption in the algebraic group model. In Anne Canteaut and Yuval Ishai, editors, *EUROCRYPT 2020, Part II*, volume 12106 of *LNCS*, pages 63–95. Springer, Heidelberg, May 2020.

GJKR07. Rosario Gennaro, Stanislaw Jarecki, Hugo Krawczyk, and Tal Rabin. Secure distributed key generation for discrete-log based cryptosystems. *Journal of Cryptology*, 20(1):51–83, January 2007.

HKL19. Eduard Hauck, Eike Kiltz, and Julian Loss. A modular treatment of blind signatures from identification schemes. In Yuval Ishai and Vincent Rijmen, editors, *EUROCRYPT 2019, Part III*, volume 11478 of *LNCS*, pages 345–375. Springer, Heidelberg, May 2019.

KG20. Chelsea Komlo and Ian Goldberg. FROST: Flexible round-optimized Schnorr threshold signatures. Cryptology ePrint Archive, Report 2020/852, 2020. https://eprint.iacr.org/2020/852.

MPSW18a. Gregory Maxwell, Andrew Poelstra, Yannick Seurin, and Pieter Wuille. Simple Schnorr multi-signature with applications to Bitcoin. Cryptology ePrint Archive, Report 2018/068, Revision 20180118:124757, 2018. https://eprint.iacr.org/2018/068/20180118:124757.

MPSW18b. Gregory Maxwell, Andrew Poelstra, Yannick Seurin, and Pieter Wuille. Simple Schnorr multi-signature with applications to Bitcoin. Cryptology ePrint Archive, Report 2018/068, Revision 20180520:191909, 2018. https://eprint.iacr.org/2018/068/20180520:191909.

PS00. David Pointcheval and Jacques Stern. Security arguments for digital signatures and blind signatures. *Journal of Cryptology*, 13(3):361–396, June 2000.

Sch01. Claus-Peter Schnorr. Security of blind discrete log signatures against interactive attacks. In Sihan Qing, Tatsuaki Okamoto, and Jianying Zhou, editors, *ICICS 01*, volume 2229 of *LNCS*, pages 1–12. Springer, Heidelberg, November 2001.

STV+16. Ewa Syta, Iulia Tamas, Dylan Visher, David Isaac Wolinsky, Philipp Jovanovic, Linus Gasser, Nicolas Gailly, Ismail Khoffi, and Bryan Ford. Keeping authorities "honest or bust" with decentralized witness cosigning. In *2016 IEEE Symposium on Security and Privacy*, pages 526–545. IEEE Computer Society Press, May 2016.

Wag02. David Wagner. A generalized birthday problem. In Moti Yung, editor, *CRYPTO 2002*, volume 2442 of *LNCS*, pages 288–303. Springer, Heidelberg, August 2002.

YW05. Tsz Hon Yuen and Victor K. Wei. Fast and proven secure blind identity-based signcryption from pairings. In Alfred Menezes, editor, *CT-RSA 2005*, volume 3376 of *LNCS*, pages 305–322. Springer, Heidelberg, February 2005.