

Threshold Signatures and Accountability

Andrew Poelstra

Director of Research, Blockstream

4 February 2019

Schnorr Signatures

$$P = xG$$

$$R = kG$$

$$e = H(P, R, m)$$

$$s = k + ex$$

Schnorr Signatures

$$P = xG$$

$$R = kG$$

$$e = H(P, R, m)$$

$$sG = kG + exG$$

$$P = xG$$

$$R^0 = kG$$

$$R = R^0 + H(R^0 \| c)G$$

$$e = H(P, R, m)$$

$$s = (k + H(R^0 \| c))G + ex$$

$$P = xG$$

$$R^0 = kG$$

$$R = R^0 + H(R^0 \| c)G$$

$$e = H(P, R, m)$$

$$sG = (k + H(R^0 \| c))G + exG$$

Sign-to-Contract Replay Attack

Suppose $k = H(x||m)$.

$$s = (k + H(R^0||c)) + ex$$

$$- s = (k + H(R^0||c')) + e'x$$

$$0 = H(R^0||c) - H(R^0||c') + (e - e')x$$

So we'd better have $k = H(x||m||c)$!

Sign-to-Contract as an Anti-Nonce-Sidechannel Measure

- If the hardware device knows c before producing R^0 it can grind k so that $(k + H(R^0 \| c))$ has detectable bias.
- If it doesn't know c how can it prevent replay attacks?
- Send hardware device $H(c)$ and receive R^0 before giving it c .
- Then $k = H(x \| m \| H(c))$.

Schnorr Multisignatures

$$P_i = x_i G$$

$$P = \sum P_i$$

$$R_i = k_i G$$

$$R = \sum R_i$$

$$e = H(P, R, m)$$

$$s_i = k_i + e x_i$$

$$s = \sum k_i + \sum e x_i$$

Schnorr Multisignatures

$$P_i = x_i G$$

$$P = \sum P_i$$

$$R_i = k_i G$$

$$R = \sum R_i$$

$$e = H(P, R, m)$$

$$s_i G = k_i G + e x_i G$$

$$s G = \sum k_i G + \sum e x_i G$$

Schnorr Multisignatures

$$\mu_i = H[H(P_1 \| P_2 \| \dots \| P_n) \| i]$$

$$P_i = \mu_i x_i G$$

$$P = \sum P_i$$

$$R_i = k_i G$$

$$R = \sum R_i$$

$$e = H(P, R, m)$$

$$s_i G = k_i G + e x_i G$$

$$sG = \sum k_i G + \sum \mu_i e x_i G$$

Verifiable Secret Sharing

Suppose a party with secret x_i wants to split her secret such that k parties may produce a signature with it.

$$p_i(X) = x_i + \gamma_{i,1}X + \gamma_{i,2}X^2 + \cdots + \gamma_{i,k}X^k$$

$$\begin{aligned}\zeta_{i,j} &= p_i(j) \\ &= x_i + j\gamma_{i,1} + j^2\gamma_{i,2} + \cdots + j^k\gamma_{i,k}\end{aligned}$$

$$\begin{aligned}p_i(0) &= x_i \\ &= \sum_{j \in \text{signers}} \lambda_{i,j} \zeta_{i,j}\end{aligned}$$

Verifiable Secret Sharing

Suppose a party with secret x_i wants to split her secret such that k parties may produce a signature with it.

$$p_i(X) = x_i + \gamma_{i,1}X + \gamma_{i,2}X^2 + \cdots + \gamma_{i,k}X^k$$

$$\begin{aligned}\zeta_{i,j}G &= p_i(j)G \\ &= x_iG + j\gamma_{i,1}G + j^2\gamma_{i,2}G + \cdots + j^k\gamma_{i,k}G\end{aligned}$$

$$\begin{aligned}p_i(0) &= x_i \\ &= \sum_{j \in \text{signers}} \lambda_{i,j} \zeta_{i,j}\end{aligned}$$

Verifiable Secret Sharing

$$\begin{aligned}xG &= \sum_{i \in \text{everyone}} \mu_i x_i G \\ &= \sum_{i \in \text{everyone}} \mu_i p_i(0) G \\ &= \sum_{i \in \text{everyone}} \mu_i \sum_{j \in \text{signers}} \lambda_{i,j} \zeta_{i,j} G \\ &= \sum_{j \in \text{signers}} \left[\sum_{i \in \text{everyone}} \lambda_{i,j} \mu_i \zeta_{i,j} G \right] \\ &= \sum_{j \in \text{signers}} \begin{bmatrix} \vdots \\ \vdots \end{bmatrix}_j\end{aligned}$$

$$P = \sum_j \begin{bmatrix} \vdots & \vdots \end{bmatrix}_j G$$

$$R_j = k_j G$$

$$R = \sum R_j$$

$$e = H(P, R, m)$$

$$s_j = k_j + e \begin{bmatrix} \vdots & \vdots \end{bmatrix}_j$$

$$s = \sum k_j + \sum e \begin{bmatrix} \vdots & \vdots \end{bmatrix}_j$$

$$P = \sum_j \begin{bmatrix} \vdots & \vdots \end{bmatrix}_j G$$

$$R_j = k_j G$$

$$R = \sum R_j$$

$$e = H(P, R, m)$$

$$s_j G = k_j G + e \begin{bmatrix} \vdots & \vdots \end{bmatrix}_j G$$

$$sG = \sum k_j G + \sum e \begin{bmatrix} \vdots & \vdots \end{bmatrix}_j G$$

- Recall the equation $P = \sum_{j \in \text{signers}} \begin{bmatrix} \vdots & \vdots \\ & j \end{bmatrix}$.
- What is this set “signers”?
- In fact any set will do; $\lambda_{i,j}$ depends on the particular set but nothing else does.
- Importantly **the signature does not depend on this set.** Such signatures are *unaccountable*.

- What does an accountable signature look like?
- Bitcoin-style “concatenate individual signatures” threshold signatures, for one.
- Can we get a constant-size accountable signature? I doubt it.

$$P = \sum_j \begin{bmatrix} \vdots & \vdots \end{bmatrix}_j G$$

$$R_j = k_j G$$

$$R^0 = \sum R_j$$

$$R = R^0 + H(R^0 \| c) G$$

$$e = H(P, R, m)$$

$$s_j = k_j + e \begin{bmatrix} \vdots & \vdots \end{bmatrix}_j$$

$$s = \sum k_j + \sum e \begin{bmatrix} \vdots & \vdots \end{bmatrix}_j$$

$$P = \sum_j \begin{bmatrix} \vdots & \vdots \end{bmatrix}_j G$$

$$R_j = k_j G$$

$$R^0 = \sum R_j$$

$$R = R^0 + H(R^0 \| c) G$$

$$e = H(P, R, m)$$

$$s_j G = k_j G + e \begin{bmatrix} \vdots & \vdots \end{bmatrix}_j G$$

$$sG = \sum k_j G + \sum e \begin{bmatrix} \vdots & \vdots \end{bmatrix}_j G$$

Semi-Accountability

- Suppose that c commits to an accountable threshold signature.
- Then we have an unaccountable signature that *commits to an accountable signature*.
- Signers can refuse to participate if this commitment is missing or invalid; hardware enforced.

- Then assuming at least one party in the signature is honest and will publish the committed accountable signature, the result is “accountable”.
- (Of course, this doesn't help if nobody is honest, which is often what you need accountability for. . .)

Open Questions

- Can we construct a commitment that can be reconstructed or brute-forced by third parties?
- Can we get deniability, *i.e.* can a non-participant prove non-participation without help?
- Extension to BLS which has no space for committing data?

Thank you.

Andrew Poelstra
clauspschnorr@wpsoftware.net