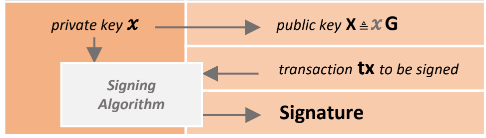
Generic Legacy Signature w/ EC keys



private key x → *public key* $X \triangleq xG$

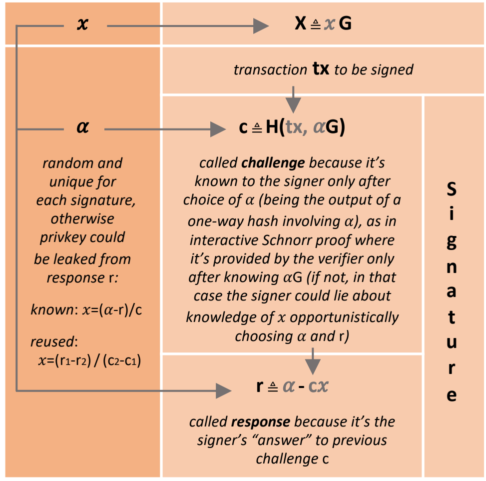
transaction **tx** *to be signed*

Signing Algorithm

Signature

Verifying Algorithm $f(X, tx, \text{Signature})$ → **success** ? **failure**

Non-interactive (Fiat-Shamir) Schnorr



x → $X \triangleq xG$

transaction **tx** *to be signed*

α → $c \triangleq H(tx, \alpha G)$

random and unique for each signature, otherwise privkey could be leaked from response r*:*

known: $x=(\alpha-r)/c$

reused: $x=(r_1-r_2)/(c_2-c_1)$

called ***challenge*** *because it's known to the signer only after choice of α (being the output of a one-way hash involving α), as in interactive Schnorr proof where it's provided by the verifier only after knowing αG (if not, in that case the signer could lie about knowledge of x opportunistically choosing α and r)*

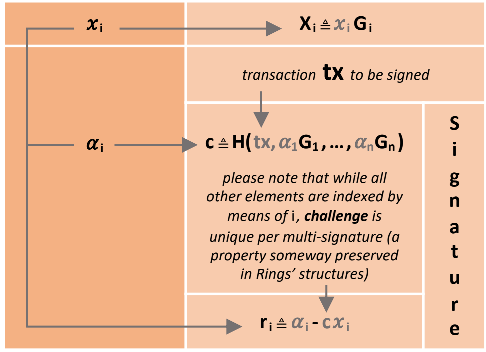
$r \triangleq \alpha - cx$

called ***response*** *because it's the signer's "answer" to previous challenge* c

Signature

$$f(X, tx, c, r)\ \left\{\ H(tx, \overbrace{rG+cX}^{=\,\alpha G \text{ if signature is ok}}) \overset{?}{=} c\right.$$

Multi keys (& bases) n.i. Schnorr (i=1,…,n)



x_i → $X_i \triangleq x_i G_i$

transaction **tx** *to be signed*

α_i → $c \triangleq H(tx, \alpha_1 G_1, \dots, \alpha_n G_n)$

please note that while all other elements are indexed by means of i*,* ***challenge*** *is unique per multi-signature (a property someway preserved in Rings' structures)*

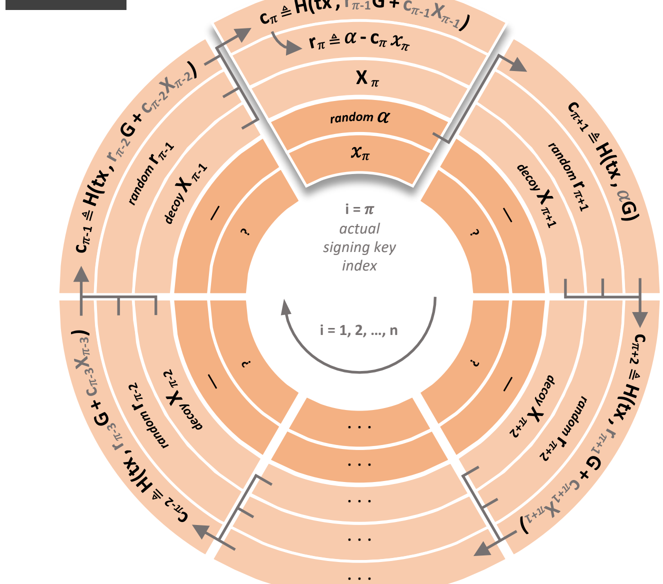
$r_i \triangleq \alpha_i - c x_i$

Signature

$$f(X_i, tx, c, r_i)\ \left\{\ H(tx, \overbrace{r_1G_1+cX_1, \dots, r_nG_n+cX_n}^{\text{it commits to } n \text{ signatures at the same time}}) \overset{?}{=} c\right.$$

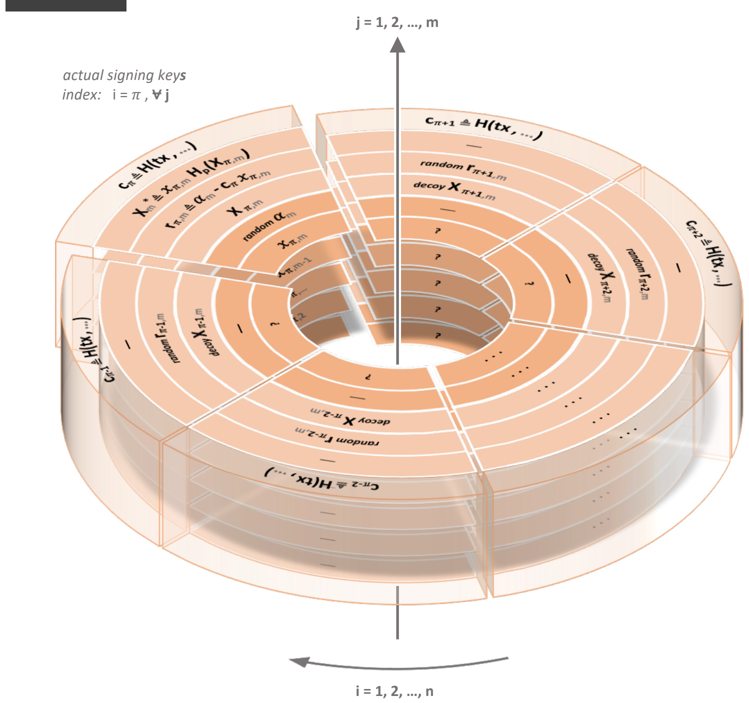
Rings "magic" is about finding flavours of previous schemas with decoys, while still retaining just only one ACTUAL signer (from a technical point of view: needing many X_i in verifying algo but single x in signing algo); and all without coordination between involved keys owners

SAG



$$\underbrace{c_2=H(tx, r_1G+c_1X_1)\quad c_3=H(tx, r_2G+c_2X_2)\quad \dots\quad c_n=H(tx, r_{n-1}G+c_{n-1}X_{n-1})\quad H(tx, r_nG+c_nX_n) \overset{?}{=} c_1}_{f(X_i,\, tx,\, c_1,\, r_i)}$$

MLSAG



$$c_{\pi+1} \triangleq H\left(tx,\ \begin{matrix} \alpha_m G, & \alpha_m H_p(X_{\pi,m}), \\ \dots, & \dots, \\ \alpha_1 G, & \alpha_1 H_p(X_{\pi,1}) \end{matrix}\right)$$

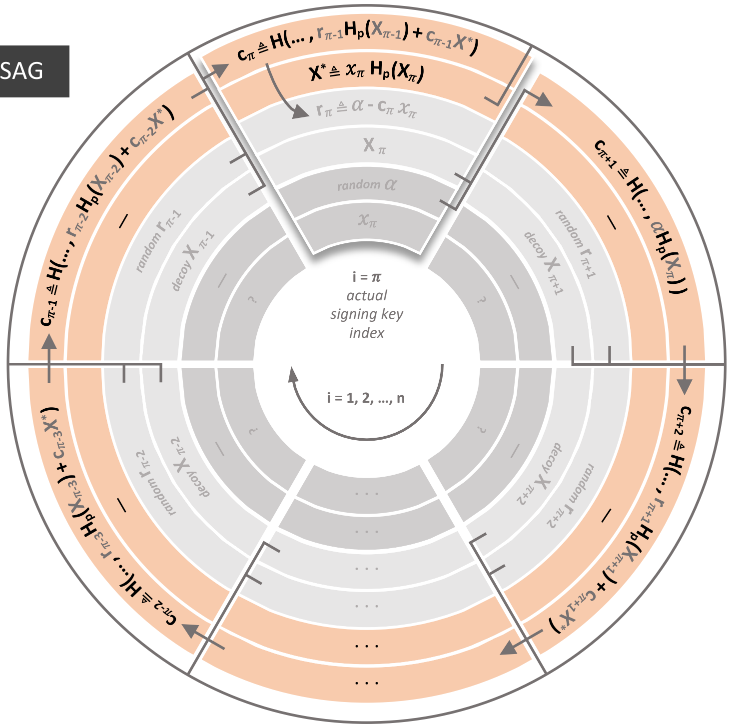
$$c_i \triangleq H\left(tx,\ \begin{matrix} r_{i-1,m}G + c_{i-1}X_{i-1,m}, & r_{i-1,m}H_p(X_{i-1,m}) + c_{i-1}X_m^*, \\ \dots, & \dots, \\ r_{i-1,1}G + c_{i-1}X_{i-1,1}, & r_{i-1,1}H_p(X_{i-1,1}) + c_{i-1}X_1^* \end{matrix}\right)$$

$f(X_{i,j}, tx, c_1, r_{i,j}, X_j^*)$

- X_j^* never seen on-chain before ?
- $l X_j^* \overset{?}{=} 0$
- $H(tx, r_{n,1}G + c_nX_{n,1}, r_{n,1}H_p(X_{n,1}) + c_nX_1^*, \dots, r_{n,m}G + c_nX_{n,m}, r_{n,m}H_p(X_{n,m}) + c_nX_m^*) \overset{?}{=} c_1$

$c_n = c_n(c_{n-1}(\dots(c_2(c_1))))$ *as previously seen in SAG and bLSAG*

bLSAG



$$c_{\pi+1} \triangleq H(tx, \alpha G, \alpha H_p(X_\pi))$$

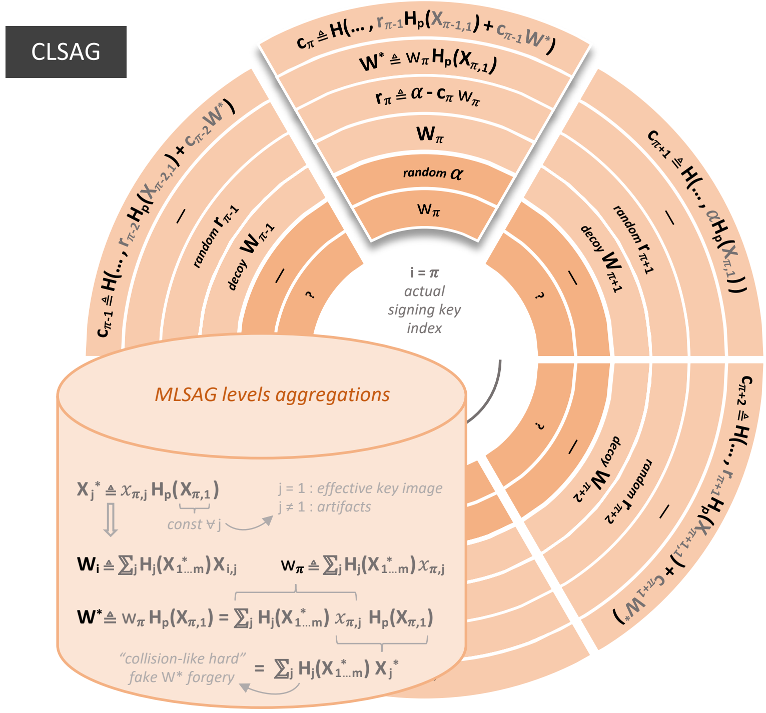
$$c_i \triangleq H(tx, r_{i-1}G + c_{i-1}X_{i-1}, r_{i-1}H_p(X_{i-1}) + c_{i-1}X^*)$$

$f(X_i, tx, c_1, r_i, X^*)$

- X^* never seen on-chain before ?
- $l X^* \overset{?}{=} 0$
- $H(tx, r_nG + \overbrace{c_n(tx, c_1, r_{i\neq n}, X_{i\neq n})}X_n, r_nH_p(X_n) + \overbrace{c_n(tx, c_1, r_{i\neq n}, X_{i\neq n})}X^*) \overset{?}{=} c_1$

$c_n = c_n(c_{n-1}(\dots(c_2(c_1))))$ *as previously seen in SAG*

CLSAG



MLSAG levels aggregations

$X_j^* \triangleq x_{\pi,j} H_p(\underbrace{X_{\pi,1}}_{\text{const } \forall j})$

j = 1 : *effective key image*
j ≠ 1 : *artifacts*

$W_i \triangleq \sum_j H_j(X_{1\dots m}^*) X_{i,j}$

$w_\pi \triangleq \sum_j H_j(X_{1\dots m}^*) x_{\pi,j}$

$W^* \triangleq w_\pi H_p(X_{\pi,1}) = \sum_j H_j(X_{1\dots m}^*)\, \underbrace{x_{\pi,j}\, H_p(X_{\pi,1})}$

$= \sum_j H_j(X_{1\dots m}^*)\, X_j^*$

"collision-like hard" fake W forgery*

$$c_{\pi+1} \triangleq H(tx, \alpha G, \alpha H_p(X_{\pi,1}))$$

$$c_i \triangleq H(tx, r_{i-1}G + c_{i-1}W_{i-1}, r_{i-1}H_p(X_{i-1,1}) + c_{i-1}W^*)$$

$f(X_{i,j}, tx, c_1, r_i, X_j^*)$

- X_1^* never seen on-chain before ?
- $l X_1^* \overset{?}{=} 0$
- $H(tx, r_nG + c_nW_n(X_j^*, X_{n,j}), r_nH_p(X_{n,1}) + c_nW^*(X_j^*)) \overset{?}{=} c_1$

again: $c_n = c_n(c_{n-1}(\dots(c_2(c_1))))$

Rings unleashed notes

SAG (Spontaneous Anonymous Group)

- the index value of actual signer (π) is random, otherwise X_π could be deduced from the order of parameters provided in signature;
- the challenges c_i are built from previous slice elements, with dependencies depicted by the arrows;
- final r_π definition guarantees the dependencies applying to all other c_i still apply to $c_{\pi+1}$ as well (even if originally calculated from α), so challenges form a closed chain, a ring: that's why it's enough to provide c_1 in signature (it's the "someway preserved" single-challenge-per-multi-signature property)

bLSAG (Back's Linkable SAG)

- bLSAG is a SAG extended with a key image X^* (to prevent double spending while still mantaining anonymity, introducing linkability of signatures) and modified challenges c_i to commit to that key image as well;
- $H_p(X_\pi)$ is a carefully chosen function returning a random point in EC basepoint-subgroup of prime-order l, acting as generator point for key image $X^* \triangleq x_\pi H_p(X_\pi)$

<u>some BAD key image generators</u>

$H_p(X_\pi) \triangleq n(X_\pi)G$
$\Rightarrow X^* \triangleq x_\pi n(X_\pi)G = n(X_\pi)x_\pi G = n(X_\pi)X_\pi$
so actual signer could be found by tries

$H_p(X_\pi) \triangleq G_2$
$\Rightarrow X_1^* \triangleq x_{\pi,1}G_2 \qquad X_2^* \triangleq x_{\pi,2}G_2$
$\Rightarrow X_1^* - X_2^* = (x_{\pi,1} - x_{\pi,2})G_2$
but a previous payer to both $X_{\pi,1}$ and $X_{\pi,2}$ can calculate the value between brackets (thanks to Diffie-Hellman-like exchange at the base of Stealth Addresses), so owns heuristics to pair future $X_{\pi,1}$ and $X_{\pi,2}$ usages

$H_p(X_\pi) \triangleq X_\pi \triangleq x_\pi G$
$\Rightarrow X_1^* - X_2^* = (x_{\pi,1}^2 - x_{\pi,2}^2)G$
like in previous case, just a bit more algebra and need to use G to get rid of remaining private spending key in favour of public one

- $l X^* = 0$ check in verifying algorithm is needed to avoid double spending due to key image "malleability". In challenges we have:

  $$c_i = H(\dots c_{i-1}X^*)$$

  however X^* could be substituted by a fake $X^* + P_h$ -where P_h is a point in EC subgroup of order h, the cofactor- if the attacker found (by tries) all c_i multiples of h; in that case:

  $$c_i(X^* + P_h) = c_iX^* + c_iP_h = c_iX^*$$

  because any point multiplied by its subgroup order gives zero. Luckily $l(X^* + P_h) \neq 0$ because, being prime, l cannot be a multiple of h

MLSAG (Multilayer Linkable SAG)

- MLSAG is a stack of many bLSAG, with per-slice challenges c_i (so one single challenge for each 3D slice, commiting to all layers);
- even if it doesn't appear to be a schema requirement, in Monero the index value of actual signer (π) is intended to be random but shared among all layers, offering inter-levels clustering opportunity to an attacker making an educated guess about actual keys: that's why multi-input transactions (where maximum savings could be attained) have preferred to avoid the use of just one single MLSAG

CLSAG (Concise Linkable SAG)

- the schema currently used by Monero, it's a bLSAG for "pseudo keys" w_π and W_i obtained aggregating keys on MLSAG different levels; it provides back-compatible linkability (meaning usual key image generation) only for $X_{\pi,1}$;
- W^* doesn't really prevent double spending by itself but it's built from effective X_1^* and $X_{i\neq1}^*$ artifacts (that's why they are the ones actually used in verifying algorithm)

Credits

This cheatsheet is deeply inspired by <u>Zero to Monero: 2nd Edition</u> (especially chapters 2 and 3 and mentioned sources): the notation is only slightly different and with "minor" omissions to focus on gradual presentation of Rings' core properties (e.g., no key prefixing or domain separation for hashes)