Pateikta

Rivest, Ronald L., Adi Shamir, and Yael Tauman. "How to leak a secret." *International Conference on the Theory and Application of Cryptology and Information Security*. Springer, Berlin, Heidelberg, 2001.

The general notion of a **group signature** scheme was introduced in 1991 by Chaum and van Heyst [2].
In such a scheme, a trusted group ***manager*** predefines certain groups of users and distributes specially designed keys to their members.
Individual members can then use these keys to anonymously sign messages on behalf of their group.
The signatures produced by different group members look indistinguishable to their verifiers, but not to the group ***manager*** who can revoke the anonymity of misbehaving signers.
In this paper we formalize the related notion of **ring signature** schemes.
These are *simplified group signature* schemes which have only users and **no managers**
(we call such signatures "***ring signatures***" instead of "***group signatures***" since
rings is some uniform community and no center).
**Group signatures** are useful when the members want to cooperate, while **ring signatures** are useful when the members do not want to cooperate.
Both **group signatures** and **ring signatures** are signer-ambiguous, but in a **ring signature** scheme there are no **prearranged groups of users**, there are **no procedures for setting**, changing, or deleting groups, there is **no way to distribute specialized keys**, and there is **no way to revoke the anonymity of the actual signer** (unless he decides to expose himself).
Our only assumption is that each member is already associated with the **public key** of some standard signature scheme such as RSA or ECDSA.
To produce a **ring signature**, the actual signer declares an arbitrary set of possible signers that includes himself, and computes the signature entirely by himself using only his **private key** and the others' **public keys**.
In particular, the other possible signers have their **private keys** and can to conduct e-commerce over the internet, and may be completely unaware that their **public keys** are used by a stranger to produce such a **ring signature** on a message they have never seen and would not wish to sign.

**Terminology**: We call a set of possible signers a **ring**. We call the ring member who produces the actual signature the ***signer*** and each of the other ring members a ***non-signer***.

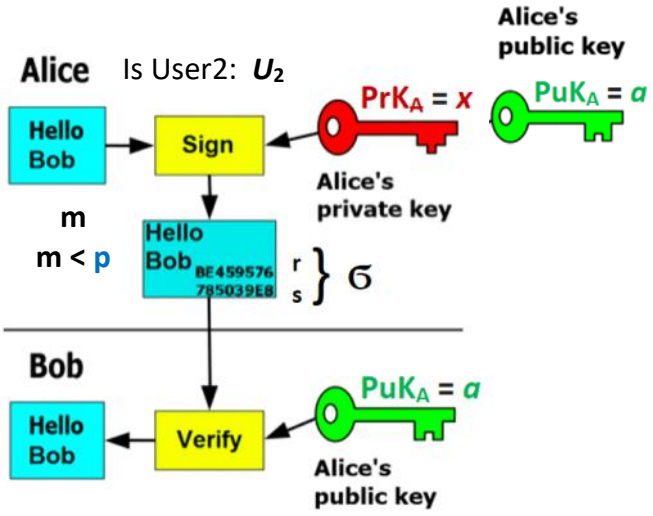
A **ring signature** scheme is set-up free: The ***signer*** does not need the knowledge, consent, or assistance of the other ring members to put them in the ring - all he needs is knowledge of their regular **public keys**.
Different members can use different independent **public key** signature schemes, with different key and signature sizes. *Size of **ring** signature depends of the number of **ring members***.
Verification must satisfy the usual soundness and completeness conditions, but in addition we want the signatures to be signer-ambiguous in the sense that the verifier should be ***unable*** to determine the ***identity*** of the actual ***signer*** in a ring of size ***r*** with probability greater than 1/***r***.
This limited anonymity can be either computational or unconditional.
Our main construction provides unconditional anonymity in the sense that even an infinitely powerful adversary with access to an unbounded number of chosen-message signatures produced by the same ring member cannot guess his identity with any advantage, and cannot link additional signatures to the same signer.

To motivate the title for this paper, suppose that Bob (also known as "Deep Throat - Gili Gerklė") is a member of the cabinet of Lower Kryptonia, and that Bob wishes to leak a juicy fact to a journalist about the escapades-pabėgimai of the Prime Minister, in such a way that Bob remains anonymous, yet such that the journalist is convinced that the leak was indeed from a cabinet member.
Bob cannot send to the journalist a standard digitally signed message, since such a message, although it convinces the journalist that it came from a cabinet member, does so by directly revealing Bob's identity.
It also doesn't work for Bob to send the journalist a message through a standard anonymizer, since the *anonymizer* strips off all source identification and authentication: the journalist would have no reason to believe that the message really came from a cabinet member at all.
A standard **group signature** scheme does not solve the problem, since it requires the prior cooperation of the other group members to set up group by manger, and leaves Bob vulnerable to later identification by the group manager, who may be controlled by the Prime Minister.
The correct approach is for Bob to send the story to the journalist through an *anonymizer*, signed with a ring signature scheme that names each cabinet member (including himself) as a ring member.
The journalist can verify the **ring signature** on the message, and learn that it definitely came from a cabinet member.
He can even post the **ring signature** in his paper or web page, to prove to his readers that the juicy story came from a reputable source.
However, neither he nor his readers can determine the actual source of the leak, and thus the whistleblower-informatorius has perfect protection even if the journalist is later forced by a judge to reveal his "source" (the signed document).

## Asymmetric Signing - Verification

**Sign($PrK_A$, h) = σ = (r, s)**

**V=Ver($PuK_A$, h, σ), V∈{True, False} ≡ {1, 0}**




Public keys of other ring members {$U_1$; $U_3$, $U_4$, $U_5$}
User1: $U_1$; User3: $U_3$; User4: $U_4$; User5 : $U_5$.

$PuK_1=A_1$; $PuK_3=A_3$; $PuK_4=A_4$; $PuK_5=A_5$;

Alice is User2: $U_2$;

$PuK_A=a$-->$PuK_2=A_2$; $PrK_2=z$.

All ring members including Alice $U_2$:{$U_1$, $U_2$; $U_3$, $U_4$, $U_5$}.

➢ Monero 2018-535.pdf

**Ring signatures** are signatures generated with a single **private key** and a set of unrelated **public keys**. The whole set of **public keys**, including the one corresponding to the **private key** at hand, is usually called a **ring**.

Somebody verifying the signature would not be able to tell which **private key** from the ring was used to produce the signature.

**Ring signatures** were originally called **group signatures** in that they were thought of as a way of proving that a signer belongs to a group, *without necessarily identifying the individual at hand.*

In the context of Monero transactions, they will help making currency flows untraceable.

**Ring signature** schemes can display a number of properties that will be useful for producing ***confidential transactions***:

**Anonymity**. An observer should not be able to determine the identity of the true ***signer*** of the message. Only that the **private key** used corresponds to one of the **public keys** in the ring.

**Linkability**. If a **private key** is used to sign two different messages, then the messages will become linked and the *duplicity will be uncovered*. In the case of Monero, this property will help preventing *double-spending attacks*.

**Exculpability** - **pateisinamumas**. A ring member whose **public key** has been used twice in two ring signatures, but is not the true signer for both, will not be linked.

Originally, **group signature** schemes required trusted group members, ***manager***, to manage the collective signatures, who had the theoretical possibility of disclosing the original ***signer***.

Relying on a single signature ***manager*** is not at all desirable, since it causes a dependency on a single group member, something that translates into a disclosure risk.

A more interesting scheme was presented by Liu et al.

The authors detailed an algorithm to cater for Linkable and Spontaneous group signatures, not requiring the collaboration of any possible co-signers.

In other words, the ***signer*** could select any set of involuntary co-signers to anonymize his own signature.

## Ring signature based on Elliptic Curves - EC

**Ethereum**: standard H-function is **keccak256**.

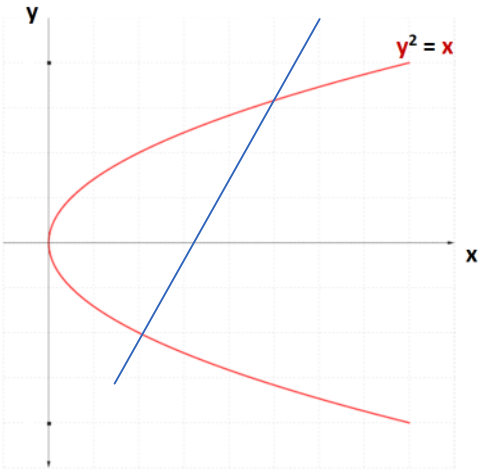
Finite field of integers:
$\mathbf{Z_p} = \{0, 1, 2, 3, \ldots, p\text{-}1\}$; $p$ is prime, $p = 2^{255}-19$; $+_{\mathbf{mod}\ p}$, $-_{\mathbf{mod}\ p}$, $\bullet_{\mathbf{mod}\ p}$, $:_{\mathbf{mod}\ p}$.
It is a finite field named also as Galois filed and alternatively denoted by $\mathbf{F_p}$.

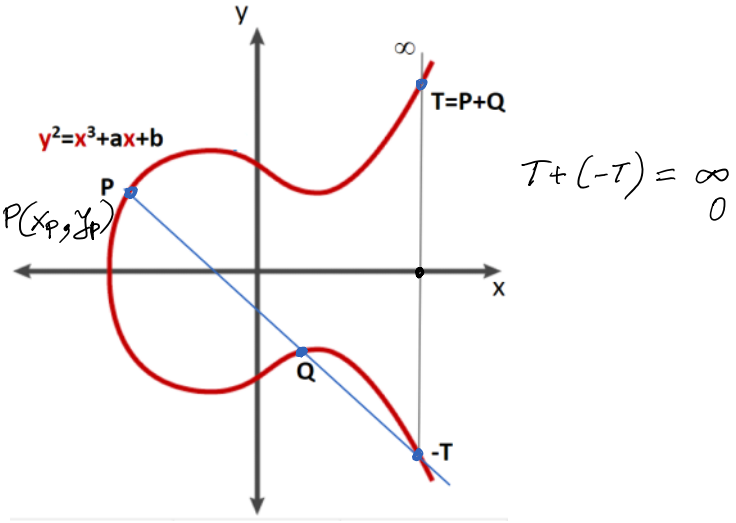
## Elliptic Curve Cryptosystem - ECC

In Figures below parabola and elliptic curve (EC) are presented in the plane XOY of real numbers and are expressed by the equations:

| $y^2 = x$ | $y^2 = x^3 + ax + b$ |
|---|---|

In EC the point addition operation is defined using two facts:
1. The line crossing any two points in EC intersects with the third point in the curve.
2. The curve is symmetric with respect to axis $x$ since there is $y^2$ in the left side of EC equation.




The points in EC forms an algebraic additive group with a very special addition operation between points illustrated in EC figure.
Then according to the algebraic group definition the addition of any two points must yield the third point in elliptic curve as a line crossing these two points intersection with the EC.
**Question**: where line crossing $-T$ and $T$ intersects the third point in EC?
**Answer**: at the infinity.
**Paradox**: this infinity is named as a zero of EC group since any additive group must have a neutral element called zero: $T + (-T) = 0$, and $T + 0 = T$.

Finite field is denoted by $F_p$ (or rarely $Z_p$), when $p$ is prime.
$F_p = \{0, 1, 2, 3, \ldots, p\text{-}1\}$; where addition, multiplication, substraction and division operations are performed **mod** $p$: $+_{\mathbf{mod}\ p}$, $-_{\mathbf{mod}\ p}$, $\bullet_{\mathbf{mod}\ p}$, $:_{\mathbf{mod}\ p}$.
Cyclic Group: $Z_p^* = \{1, 2, 3, \ldots, p\text{-}1\}$; $\bullet_{\mathbf{mod}\ p}$, $:_{\mathbf{mod}\ p}$.

Let us consider abstract EC defined in the plane XOY with coordinates in finite field and $F_p = \{0, 1, 2, \ldots, p\text{-}1\}$ and expressed by the equation:

$$y^2 = x^3 + ax + b \ \mathbf{mod}\ p.$$

EC points are computed by choosing coordinate $x$ and computing coordinate $y^2$.

To compute coordinate $y$ it is needed to extract root square of $y^2$.

$$y = \pm\sqrt{y^2} \bmod p.$$

Notice that from $y^2$ we obtain 2 points in EC, namely $+y$ and $-y$ no matter computations are performed with integers **mod** $p$ or with real numbers.

Notice also that since EC is symmetric with respect to $x$-axis, the points $+y$ and $-y$ are symmetric in EC.

Since all arithmetic operations are computed **mod** $p$ then according to the definition of negative points in $F_p$ points $y$ and $-y$ must satisfy the condition

$$y + (-y) = 0 \bmod p.$$

Then evidently

$$y^2 = (-y)^2 \bmod p.$$

For example: $p = 11$

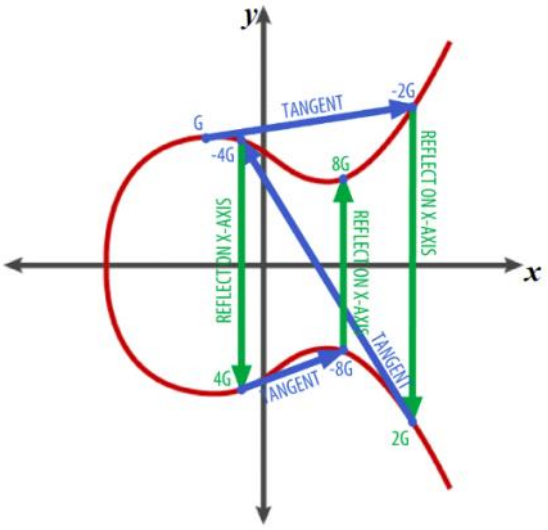
-2 mod 11 = 9

$2^2$ mod 11 = 4 & $9^2$ mod 11 = 4

>> mod(9^2,11)

ans = 4

| ElGamal Cryptosystem (CS) | Elliptic Curve Cryptosystem (CS) |
|---|---|
| **PP**=(strongprime $p$, generator $g$); $p$=255996887; $g$=22; | **PP**=(EC secp256k1; BasePoint-Generator $G$; prime $p$; param. $a$, $b$); Parameters $a$, $b$ defines EC equation $y^2=x^3+ax+b \bmod p$ over $F_p$. |
| **PrK**=$x$; >> x=randi($p$-1). | $\mathbf{PrK_{ECC}}$=z; >> z=randi($p$-1). |
| **PuK**=$a=g^x \bmod p$. | $\mathbf{PuK_{ECC}}=A=z*G$. |
| **Alice** $A$: x=1975596; $a$=210649132; | **Alice** $A$: z=…..; $A=(x_A, y_A)$; |




EC: $y^2=x^3+ax+b \bmod p$

Let we computed $\mathbf{PuK_{ECC}}=A=(x_A, y_A)=z*G$.

Then $(y_A)^2 = (x_A)^3+a(x_A)+b \bmod p$ is computed.

By extracting square root from $(y_A)^2$ we obtain 2 points: $z*G$ and $-z*G$ with coordinates $(x_A, y_A)$ and( $x_A$, $-y_A$).

According to the property of arithmetics of integers **mod** $p$ either $y_A$ is **even** and $-y_A$ is **odd** or $y_A$ is **odd** and $-y_A$ is **even**.

The reason is that $y_A+(-y_A)=0 \bmod p$ as in the example above when $p$=11.

Then we can compress $\mathbf{PuK_{ECC}}$ representation with 2 coordinates $(x_A, y_A)$ by representing it with 1 coordinate $x_A$ and adding prefix either 02 if $y_A$ is even or 03 if $y_A$ is odd.

$2^3 = 8$ sums

**Signature creation** for message $M$

Signature is formed on the h-value $h$ of Hash function of $M$.

Recommended to use SHA256 algorithm

1. $h$ = H($M$)=SHA256($M$);
2. $i$ <-- randi; $|i| \leq$ 256 bits; gcd(i, p)=1 --> E! $i^{-1}$ mod p.

3. $\boldsymbol{R} = \boldsymbol{i}*\boldsymbol{G} = (x_R, y_R)$;
4. $\boldsymbol{r} = x_R \bmod p$;
5. $\boldsymbol{s} = (\boldsymbol{h} + \boldsymbol{z} \bullet \boldsymbol{r}) \bullet \boldsymbol{i}^{-1} \bmod p$; $|\boldsymbol{s}| \leq 256$ bits; // Since $p$ is prime, then exists $\boldsymbol{i}^{-1} \bmod p$.
// >> i_m1=mulinv(i,p) % in Octave
6. **Sign**($\mathbf{PrK_{ECC}}$=**z**, $\boldsymbol{h}$) = $\boldsymbol{\sigma} = (\boldsymbol{r}, \boldsymbol{s})$

**Signature vrification: Ver(PuK, σ, *h*)**
1. Calculate $\boldsymbol{u}_1 = \boldsymbol{h} \bullet \boldsymbol{s}^{-1} \bmod p$ and $\boldsymbol{u}_2 = \boldsymbol{r} \bullet \boldsymbol{s}^{-1} \bmod p$
2. Calculate the curve point $\boldsymbol{V} = \boldsymbol{u}_1*\boldsymbol{G} \boxplus \boldsymbol{u}_2*\boldsymbol{A} = \boldsymbol{V}(x_V, y_V)$
3. The signature is valid if $\boldsymbol{R}=\boldsymbol{V}$; $\boldsymbol{r}=x_V=x_R \bmod p$.

Let $\boldsymbol{u}$, $\boldsymbol{v}$ are integers $< p$. and $\boldsymbol{P}$, $\boldsymbol{Q}$ are the points in EC.
Property 1: $(\boldsymbol{u} + \boldsymbol{v})*\boldsymbol{P} = \boldsymbol{u}*\boldsymbol{P} \boxplus \boldsymbol{v}*\boldsymbol{P}$ replacement to --> $\underline{(\boldsymbol{u} + \boldsymbol{v})\boldsymbol{P} = \boldsymbol{u}\boldsymbol{P} + \boldsymbol{v}\boldsymbol{P}}$
Property 2: $(\boldsymbol{u})*(\boldsymbol{P} \boxplus \boldsymbol{Q}) = \boldsymbol{u}*\boldsymbol{P} \boxplus \boldsymbol{u}*\boldsymbol{Q}$ replacement to --> $\boldsymbol{u}(\boldsymbol{P} + \boldsymbol{Q}) = \boldsymbol{u}\boldsymbol{P} + \boldsymbol{u}\boldsymbol{Q}$

Let $\boldsymbol{t}$, $\boldsymbol{z}$, $\boldsymbol{c}$ are integers.
Important identity used e.g. in Ring Signature:
$\mathbf{(t-z \bullet c)*G \boxplus c*A = t*G \boxplus (- z \bullet c)*G \boxplus c*A = t*G \boxplus (c*(-z)*G + A)) = t*G \boxplus c*(-A \boxplus A) = t*G \bmod p}$.

In **Monero** Elliptic Curve - EC is denoted by **Ed25519** satisfying equation is used

$$-x^2 + y^2 = 1 - \frac{121665}{121666}x^2y^2 \quad \bmod p$$

Public Parameters: **PP** = ( **p**, **G**, **h**(.), $\mathbf{H_{EC}}$(.) ), where **h**(.) is ordinary h-function, e.g. SHA256.
Let **M** - message to be signed.
Then Ring signature is placed on the h-value
**h(M)=m**, |**m**|=256 bits.
$\mathbf{H_{EC}(D) = H} \in$ **Ed25519**; **H** is a point of EC, i.e. $\mathbf{H_{EC}}$(.) maps data **D** to some point **H** of **EC**.
The realization of $\mathbf{H_{EC}}$(.) is simple, for example let Ring data is hashed by **h(R)=d**:

$\mathbf{H_{EC}(D) = h(D)*G = d*G = H} \in$ **Ed25519**.

Public keys of ring members $\{\boldsymbol{U}_1; \boldsymbol{U}_3, \boldsymbol{U}_4, \boldsymbol{U}_5\}$
User1: $\boldsymbol{U}_1$; User3: $\boldsymbol{U}_3$; User4: $\boldsymbol{U}_4$; User5 : $\boldsymbol{U}_5$.
$\mathbf{PuK_1}=\mathbf{A_1}$; $\mathbf{PuK_3}=\mathbf{A_3}$; $\mathbf{PuK_4}=\mathbf{A_4}$; $\mathbf{PuK_5}=\mathbf{A_5}$;

Alice is User2: $\boldsymbol{U}_2$;
$\mathbf{PuK_A}=\boldsymbol{a}$-->$\mathbf{PuK_2}=\mathbf{A_2}$; $\mathbf{PrK_2}=\mathbf{z}$.

All ring members including Alice $\mathbf{U_2}$ are the following: $\{\boldsymbol{U}_1, \boldsymbol{U}_2; \boldsymbol{U}_3, \boldsymbol{U}_4, \boldsymbol{U}_5\} = \boldsymbol{R}$
In Monero standard H-function is SHA256().

In **Monero** Elliptic Curve - EC is denoted by **Ed25519** satisfying equation is used

$$-x^2 + y^2 = 1 - \frac{121665}{121666}x^2y^2 \quad \bmod p$$

SHA256()

There are used 2 H-functions : $H(\ )$ ; $H_{EC}(\ )$ → an EC point $H_p$.

$H_{EC}(U_1||U_2||U_3||U_4||U_5) = H_{EC}(R) = H_p$

1) $H(U_1 || U_2 || U_3 || U_4 || U_5) = h$

2) $H_{EC}(R) = h * G = H_p$

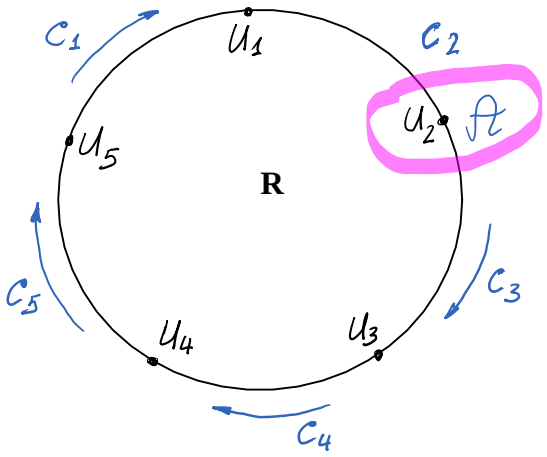
Verification key **V**

$U_2$ : $V = z * H_{EC}(U_1||U_2||U_3||U_4||U_5) = z * H_{EC}(R) = z * H_p$

Ring of Users:

**R = { $U_1$, $U_2$, $U_3$, $U_4$, $U_5$ }**

**$PuK_1=A_1$** **($PrK_2=z$, $PuK_2=A_2=z*G$)** **$PuK_3=A_3$** **$PuK_4=A_4$** **$PuK_5=A_5$**




$U_2$; computations

1) $\alpha \leftarrow randi(\mathcal{I}_p)$ ; $\mathcal{I}_p = \{0, 1, 2, \ldots, p-1\}$

$r_1, r_2, r_3, r_4, r_5 \leftarrow randi(\mathcal{I}_p)$

2) M - message to be signed, then M is hashed by functions $H(\ )$ ; $H_{EC}(\ )$:

$H(M) = m$ SHA256()

$H_{EC}(R) = H_p$

3) $c_3 = H(R, V, m, \alpha * G, \alpha * H_{EC})$

4) $c_4 = H(R, V, m, r_3 * G + c_3 * A_3, r_3 * H_p + c_3 * V)$

5) $c_5 = H(R, V, m, r_4 * G + c_4 * A_4, r_4 * H_p + c_4 * V)$

6) $c_1 = H(R, V, m, r_5 * G + c_5 * A_5, r_5 * H_p + c_5 * V)$

7) $c_2 = H(R, V, m, r_1 * G + c_1 \cdot A_1, r_1 * H_p + c_1 * V)$

7) $c_2 = H(R, V, m, r_1 * G + c_1 \cdot A_1, \ldots)$

A computes: $r_2 = \alpha - z \cdot c_2 \mod p$

$Sign(M) = (c_1, r_1, r_2, r_3, r_4, r_5, V) = \sigma$

Let ***u***, ***v*** are integers < ***p*** and ***P, Q*** are the points in EC.
Property 1: $(u + v)*P = u*P \boxplus v*P$ replacement to --> $\underline{(u + v)P = uP + vP}$
Property 2: $(u)*(P \boxplus Q) = u*P \boxplus u*Q$ replacement to --> $u(P + Q) = uP + uQ$

Let ***t***, ***z***, ***c*** are integers.
Important identity used, e.g. in Ring Signature:
$(t - z \bullet c)*G \boxplus c*A = t*G \boxplus (-z \bullet c)*G \boxplus c*A = t*G \boxplus (c*(-z)*G + A)) = t*G \boxplus c*(-A \boxplus A) = t*G \bmod p$.

Verification: $Ver(V, \sigma, m) \in \{T, F\}$

for $i = 1,2,3,4,5$ compute, replacing $5+1 \to 1$: $H_{EC}(R) = H$.

$q_1' = r_1 * G + c_1 * A_1$; $q_1'' = r_1 * H_p + c_1 * V$;
$c_2' = H(R, V, m, q_1', q_1'')$.

$q_2' = r_2 * G + c_2 * A_2$; $q_2'' = r_1 * H_p + c_2 * V$;
$c_3' = H(R, V, m, q_2', q_2'')$

– – – – –

$q_5' = r_5 * G + c_5 * A_5$; $q_5'' = r_5 * H_p + c_5 * V$
$c_1' = H(R, V, m, q_5', q_5'')$

Signature is valid if: $c_1' = c_1$

Correctness: if $i \neq 2$, then $c_{i+1}$ is defined as in signature algorithm.
if $i = 2$, then

$$q_2' = r_2 * G + c_2 * A_2 = (\alpha - z \cdot c_2) * G + c_2 * A_2$$
$$= \alpha * G - (c_2 \cdot z) * G + c_2 * A_2$$
$$= \alpha * G - c_2 * (z * G) + c_2 * A_2$$
$$= \alpha * G - c_2 * A_2 + c_2 * A_2 = \alpha * G$$

$q_2'' = r_2 * H_p + c_2 * V$

$$
\begin{aligned}
q_2'' &= r_2 * H_p + c_2 * V \\
&= (\alpha - z \cdot c_2) * H_p + c_2 * V \\
&= \alpha * H_{EC} - (z \cdot c_2) * H_p + c_2 * V \\
&= \alpha * H_{EC} - c_2 * (z * H_p\ (R)) + c_2 * V \\
&= \alpha \cdot H - c_2 * V + c_2 * V = \alpha * H_p
\end{aligned}
$$

$$
c_2' = H(R, V, m, q_2', q_2'') = H(R, V, m, \alpha * G, \alpha * H_p)
$$

$$
c_2 = H(R, V, m, r_1 * G + c_1 \cdot A_1, r_1 * V + c_1 * V')
$$

$$
\Big\} \; c_2 = c_2'
$$