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 are geometric regions with uniform periphery 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 could have chosen their **private keys** only in order 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 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.

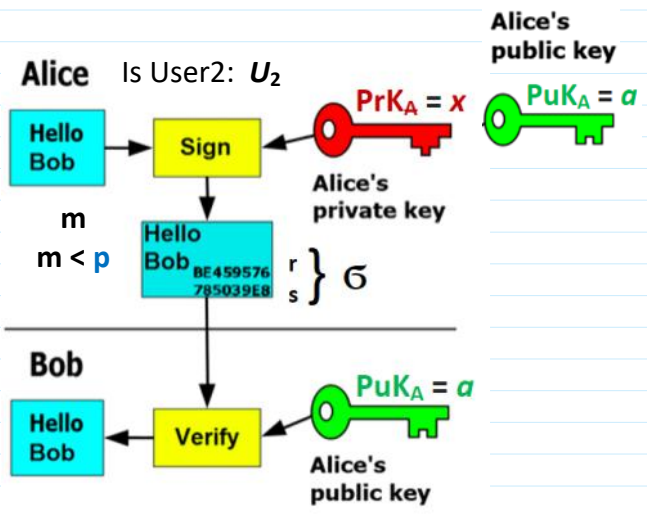
To motivate the title for this paper, suppose that Bob (also known as "Deep Throat - Gili Gerklė") is a member of the cabinet of Lower Kryptonia, and that Bob wishes to leak a juicy fact to a journalist about the escapades-pabėgimai of the Prime Minister, in such a way that Bob remains anonymous, yet 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) = $\sigma$ = (r, s)**

**V=Ver($PuK_A$, h, $\sigma$), V$\in${True, False} $\equiv$ {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 $U_2$:{$U_1$, $U_2$; $U_3$, $U_4$, $U_5$}.

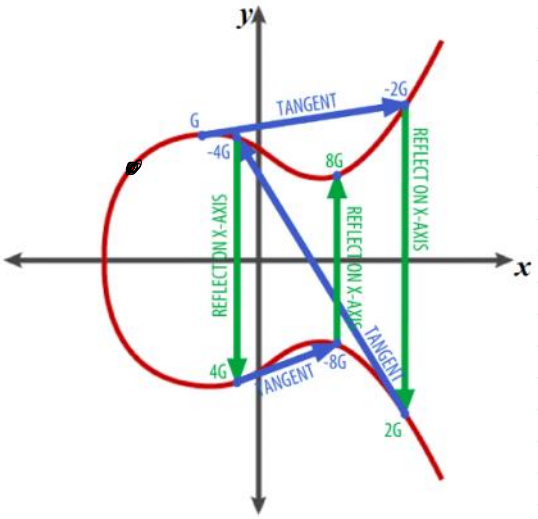
➢ Monero 2018-535.pdf
**Ring signatures** are signatures generated with a single **private key** and a set of unrelated **public keys**.
The whole set of **public keys**, including the one corresponding to the **private key** at hand, is usually called a ring.
Somebody verifying the signature would not 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

Finite field of integers:
$\mathbf{Z_p}$ = **{0, 1, 2, 3, …, p-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 $\mathbf{F_p}$.

| 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). | **PrK$_{ECC}$**=**z**; >> **z**=randi(***p***-1). |
| **PuK**=$a=g^x \bmod p$. | **PuK$_{ECC}$**=$A$=**z**$*G$. |
| **Alice** ***A***: **x**=**1975596**; ***a***=**210649132**; | **Alice** ***A***: **z**=**….**; $A=(x_A, y_A)$; |




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

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

$R = u_1||u_2||u_3||u_4||u_5$

1) $H(u_1||u_2||u_3||u_4||u_5) = h$; $|h| = 256$ bits using sha256( )

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

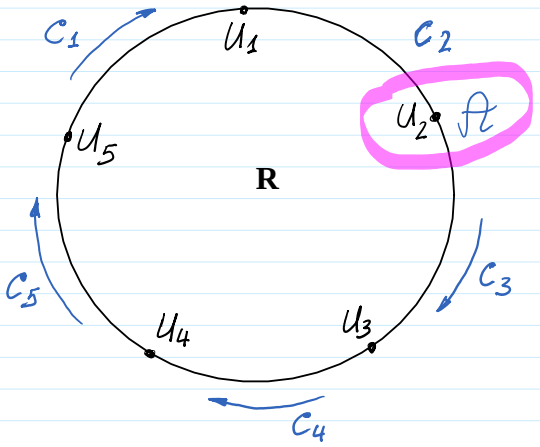
Verification key V

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

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

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

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

$H(M) = m$

$H_{EC}(R) = H_p$

secp256k1: number of points in EC is equal to the prime ***p***.

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

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

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

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

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

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•c)*G ⊞ c*A = t*G ⊞ (- z•c)*G ⊞ c*A = t*G ⊞ c*((-z)*G + A) = t*G ⊞ c*(-A ⊞ A) = t*G mod *p*.**

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

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

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

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

– – – – –

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

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

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

nebistiné

$$q_2'' = r_2 * H_p + c_2 * V$$
$$= (\alpha - z \cdot c_2) * H + c_2 * V$$
$$= \alpha * H_p - (z \cdot c_2) * H_p + c_2 * V$$
$$= \alpha * H_p - c_2 * (z * H_p(R)) + c_2 * V$$
$$= \alpha \cdot H_p - c_2 * V + c_2 * V = \alpha * H_p$$

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

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

$c_2 = c_2'$

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

Till this place