Ouroboros

Sauver la cryptographie avec elle-même

Tandem - Monero

Sécuriser une clé privée d'une façon anonyme

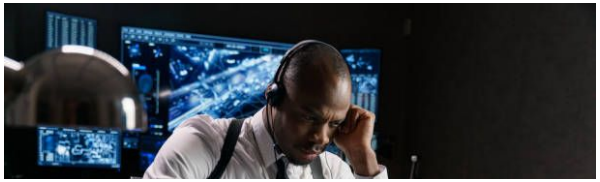
Linus Gasser, C4DT/EPFL

Ce qu'on va voir

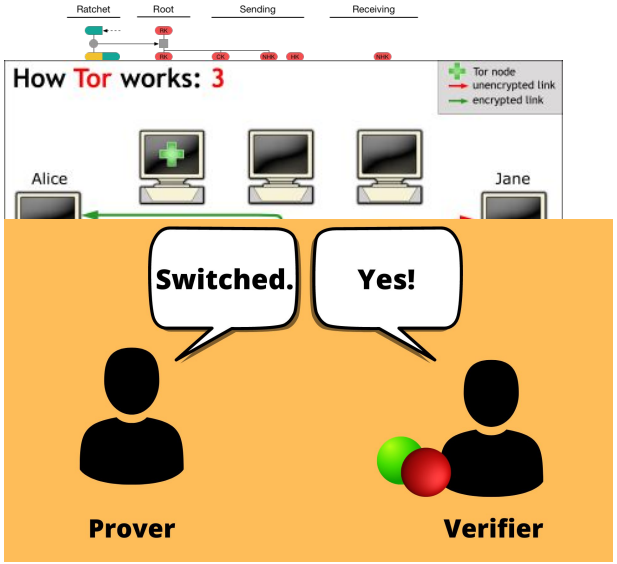
- **Le problème**
  - Pourquoi les clés privées décentralisées sont compliquées?
- **Courbes elliptiques**
  - Plus petit et plus rapide que RSA
- **Monero**
  - Une cryptomonnaie anonyme
- **Tandem**
  - Gestion anonyme de clés privées
- **Conclusion**

Préparez à écrire

La cryptographie pour conserver la sphère privée



Collecte abusive de données



Zero Knowledge Proofs

https://hashdork.com/zero-knowledge-proofs/
https://www.istockphoto.com/de/vektor/konzepte-f%C3%BCr-kreativen-prozess-big-data-filter-daten-tunnel-analyse-gm464806966-58980194
https://outfunnel.com/website-visitor-tracking-tools/

https://www.iusmentis.com/society/privacy/remailers/onionrouting/

Clé privée

+ Remplace le mot de passe
+ Peut directement être utilisée dans des algorithmes cryptographiques
+ Donne plus de possibilités qu'un mot de passe

- Peut être difficile à changer (blockchains)
- Ne peut pas être mémorisée (4433d156e8c53bf5b50af07aa95a29436f29a94e0ccc5d58df8e57bdc8583c32)
- Devient un point très sensible

Attaques aux clés privées

Perte

- Systèmes de révocations manquants (certificats CoViDs en Allemagne)
- Changement quasiment impossible (blockchains)

Donc

- Sauvegarder

Vol / Copie

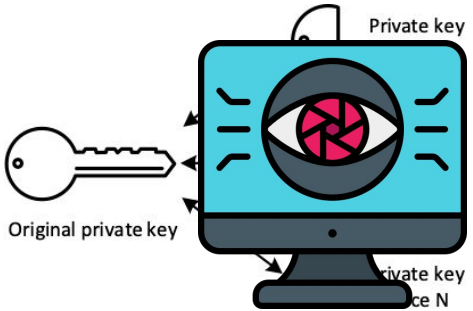
- Des fois même pas détecté
- Donne tous les droits au voleur

Donc

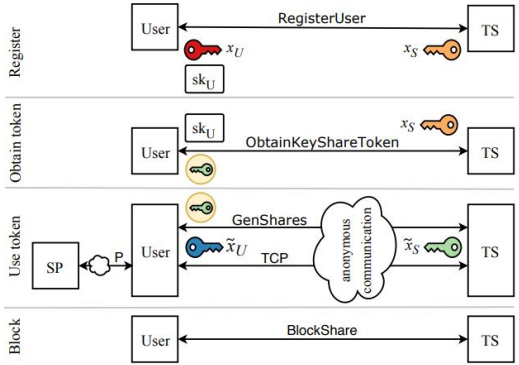
- ???

Quelques solutions

Dépositaire



Fragmentation de clés



Tandem

Ce qu'on va voir

- **Le problème**
  - Pourquoi les clés privées décentralisées sont compliquées?
- **Courbes elliptiques**
  - Plus petit et plus rapide que RSA
- **Monero**
  - Une cryptomonnaie anonyme
- **Tandem**
  - Gestion anonyme de clés privées
- **Conclusion**

Clés dans la cryptographie asymétrique

Clé publique

Facile

Difficile

Clé privée

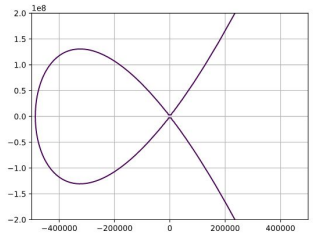
- Peut être partagée
- **Chiffre** les messages
- **Vérifie** les signatures

One-way function
ou
Fonction à sens unique

- Doit rester secret
- **Déchiffre** les messages
- **Signe** les messages

Courbes elliptiques

- 1985 par Neal Koblitz et Victor S. Miller
- Plusieurs paramétrisation pour les courbes elliptiques
- Daniel J. Bernstein en 2008 propose la *curve25519*, utilisée dans le TLS



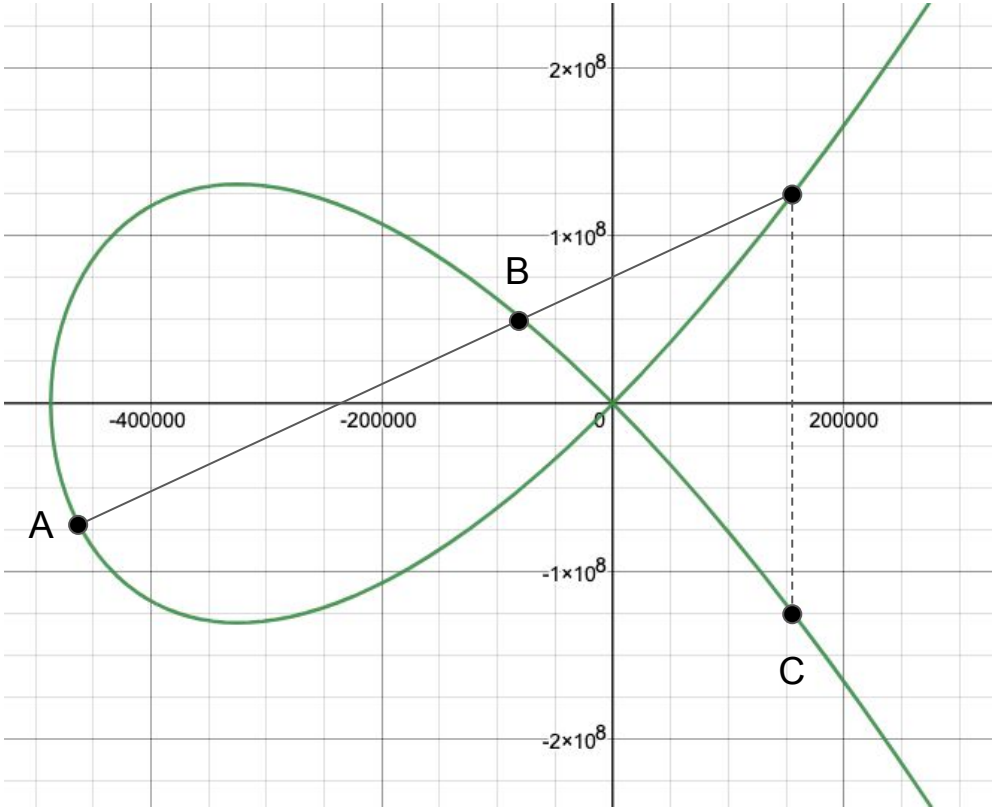
Corps commutatif (Field)

- Permet deux opérations: +, *
- Ainsi que leurs inverses: -, /
- Notation: $\mathbf{F}_n$
- Ici: les nombres naturels de 0..(n-1)
  - n un nombre premier
  - très large: 2^{255}-19

Groupe

- Permet une opération: +
- Ainsi que son inverse: -
- Avec un élément neutre: 0

Courbe Elliptique Ed25519



Montgomery curve

$y^2 = x^3 + 486662x^2 + x$

[Explication courbes elliptiques par Cloudflare]

Notations en courbes elliptiques

Courbes elliptiques - Notations additives:

- ***m***inuscule: scalaire 0..n (nombre entier)
- ***M***ajuscule: point sur la courbe

Opérations:

- scalaire, scalaire: addition, multiplication, et leur inverses
- Point, Point: addition et soustraction
- scalaire, Point: multiplication scalaire
- **commutatif et associatif**

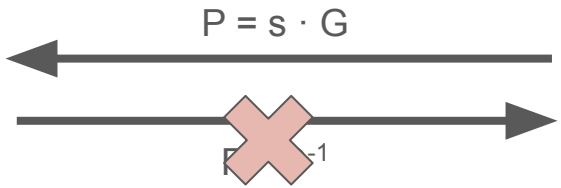
Donc:

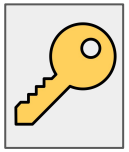
- **s**: clé privée, scalaire
- **G**: générateur, Point - le même pour tout le monde - des fois aussi **B**
- **P**: clé publique, Point
  P = G · s
  Produit de la multiplication scalaire du générateur avec la clé privée

Nous ignorons ici la restriction sur un “Corps fini” (*finite field*) et supposons que toutes les opérations sont suivies d’un modulo.

Clés dans la cryptographie asymétrique

Clé publique





Clé privée

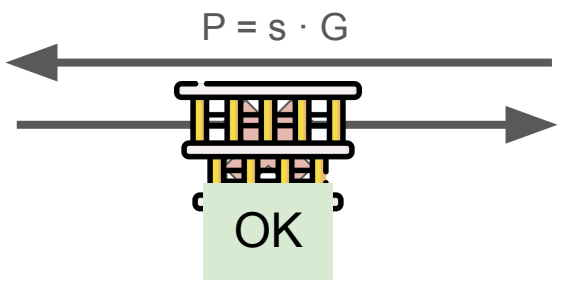
- Peut être partagée
- **Chiffre** les messages
- **Vérifie** les signatures

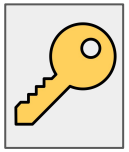
One-way function
ou
Fonction à sens unique

- Doit rester secret
- **Déchiffre** les messages
- **Signe** les messages

Clés dans la cryptographie asymétrique

Clé publique





Clé privée

- Peut être partagée
- **Chiffre** les messages
- **Vérifie** les signatures

[One-way function](#)
ou
[Fonction à sens unique](#)

- Doit rester secret
- **Déchiffre** les messages
- **Signe** les messages

Signature de Schnorr

Étant donné

- clé privée **s** avec clé publique **P = G · s**
- un message **M** et une fonction de hachage cryptographique **H**

Calculer

- un scalaire aléatoire **k**
- **R = G · k**
- **e = H(R || M)**
- **v = k - s · e**

La signature est maintenant **(v, e)**

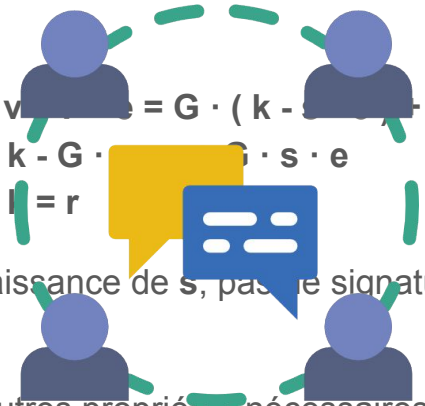
Pour vérifier

- **R = G · v + P · e**
- **e = H(R || M)**

Si **e == e**, la signature est vérifiée.

Preuve

- **R = G · v + P · e = G · (k - s · e) + G · s · e**
  = G · k - G · s · e + G · s · e
  = G · k = r



Sans la connaissance de **s**, pas de signature possible.

Beaucoup d'autres propriétés nécessaires.

Le chiffrement ElGamal avec courbe Elliptique

Alice choisit une clé secrète et aléatoire ***s*** et calcule la clé publique ***P***

- **s**, **P = G · s**

Bob veut envoyer un point secret ***M*** et choisit un nombre aléatoire ***y***, puis il calcule

- $\mathbf{C_1 = G \cdot y}$; $\mathbf{C_2 = M + P \cdot y}$

Et envoie $\mathbf{(C_1, C_2)}$ à Alice.

Un attaquant ne peut pas récupérer **M**, parce que:

- Il ne peut pas inverser **G · y**, donc il ne peut pas trouver **M** à partir de $\mathbf{C_2}$

Devoir: le déchiffrement ElGamal

Alice peut déchiffrer comme suite:

- $M = C_2 - C_1 \cdot s$

Montrez que c'est juste.

Pour la solution, voir ElGamal (Wikipédia)

Ce qu'on va voir

- **Le problème**
  - Pourquoi les clés privées décentralisées sont compliquées?
- **Courbes elliptiques**
  - Plus petit et plus rapide que RSA
- **Monero**
  - Une cryptomonnaie anonyme
- **Tandem**
  - Gestion anonyme de clés privées
- **Conclusion**

Systèmes qui utilisent une clé privée

Sans anonymité

- Chiffrement et signature d'emails
- HTTPS
- DNSSEC
- Bitcoin / Ethereum

Avec anonymat

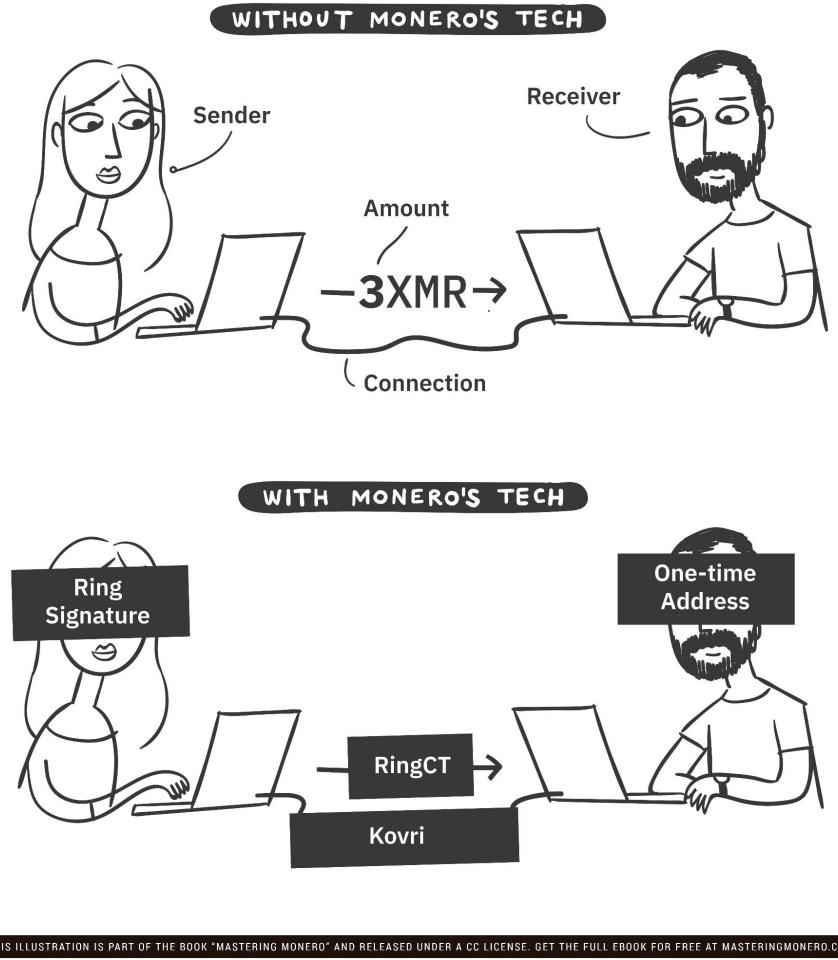
- Gestion d'identités
- Cryptomonnaie Monéro
- Zero-knowledge Proofs blockchains

Monéro

- Anonymise la source, la destination, et le montant d'une transaction
- Utilise des clés privées
- Transactions sont liées à ces clés
- Le/la propriétaires des clés peut créer une transaction

Anonymité dans Monéro

- **Signature de cercle** - ajouter un k-anonymat de l'expéditeur
- **RingCT** - prouver que les moneros entrant égalent aux moneros sortant
- **One-time address** - cacher la destination
- **Kovri** - réseau anonyme (pas fonctionnel pour le moment)



Adresse à usage unique

(s_B^v, s_B^s) and (K_B^v, K_B^s)

Alice

Bob

$K^{out} = H(r \cdot K_B^v) \cdot G + K_B^s$

$(r \cdot G, K^{out})$

$s_B^v \cdot r \cdot G = r \cdot K_B^v$

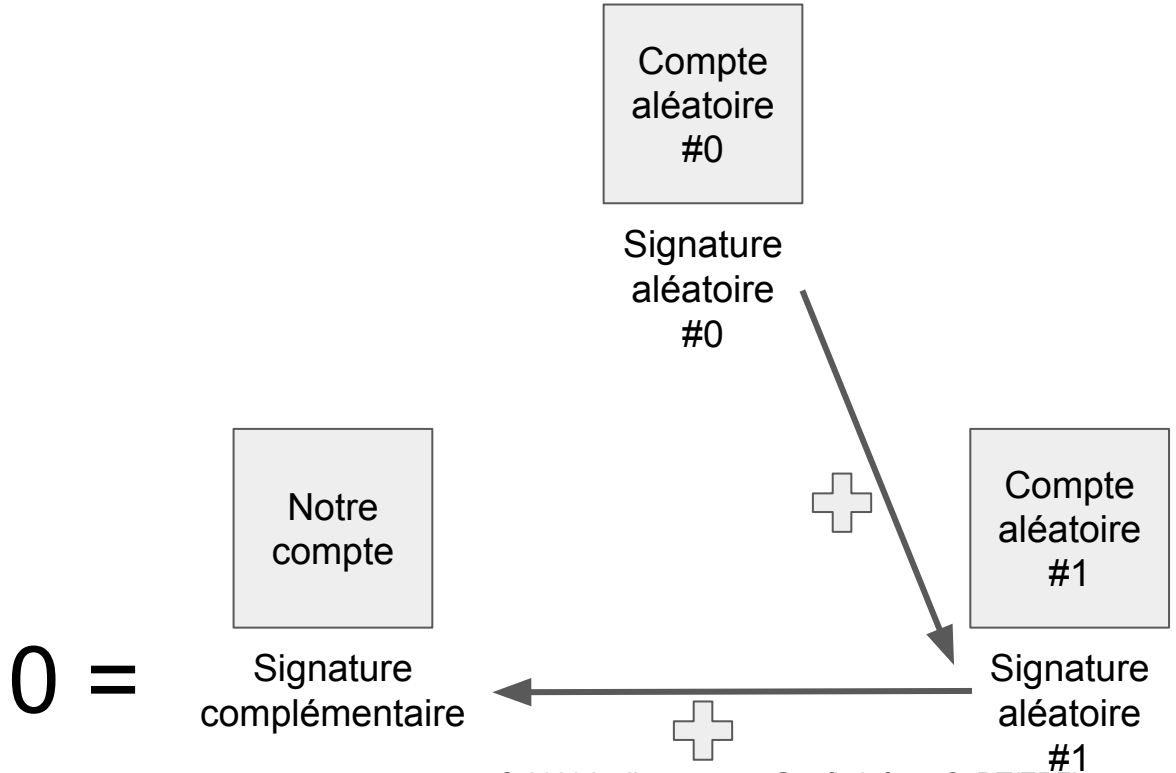
$K^{out} = H(r \cdot K_B^v) \cdot G + K_B^s$

$K^{out} == K^{out}$?

Bob veut savoir si sa clé publique K_B^s a été utilisée

Seul Bob a la clé s_B^v

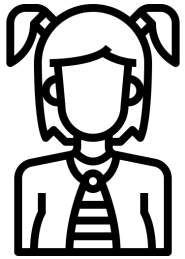
Signature de cercle (Ring Signatures)



Ce qu'on va voir

- **Le problème**
  - Pourquoi les clés privées décentralisées sont compliquées?
- **Courbes elliptiques**
  - Plus petit et plus rapide que RSA
- **Monero**
  - Une cryptomonnaie anonyme
- **Tandem**
  - Gestion anonyme de clés privées
- **Conclusion**

Approche simple

User 

x_U

$x = x_U + x_S$

x_S

 Server

$x \cdot r \cdot G = ?$

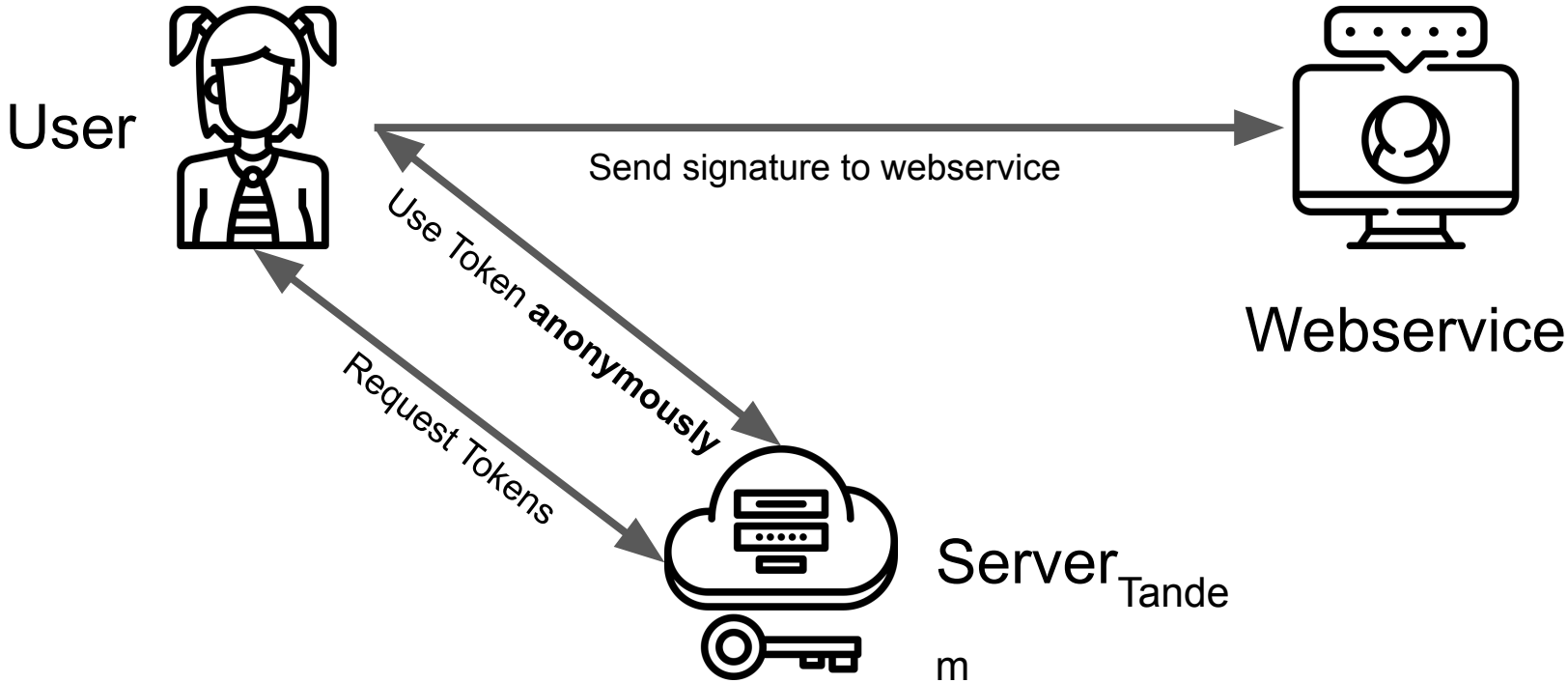
$r \cdot G$

$x_S \cdot r \cdot G$

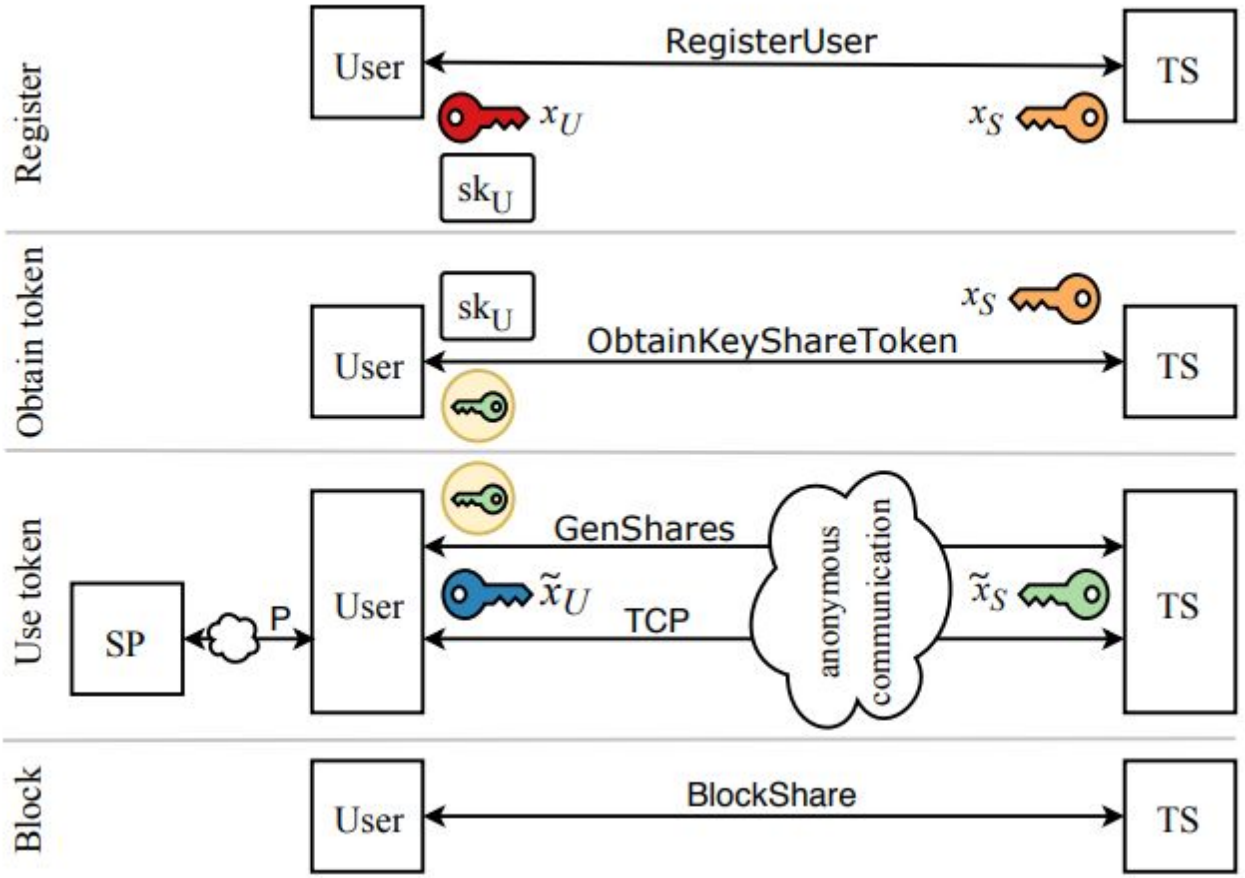
Le serveur sait qui l'utilisateur lui demande de calculer!

$x_S \cdot r \cdot G + x_U \cdot r \cdot G =$
$(x_S + x_U) \cdot r \cdot G =$
$x \cdot r \cdot G$

Tandem



Tandem


Register
User
RegisterUser
TS
x_U
x_S
sk_U
Obtain token
sk_U
x_S
User
ObtainKeyShareToken
TS
Use token
GenShares
SP
P
User
$\tilde{x}_U$
TCP
anonymous communication
$\tilde{x}_S$
TS
Block
User
BlockShare
TS

Linear randomization

Enregistrement:

- $x \rightarrow x_U, x_S$ avec $x_U + x_S = x$: partage de clé entre utilisateur et serveur

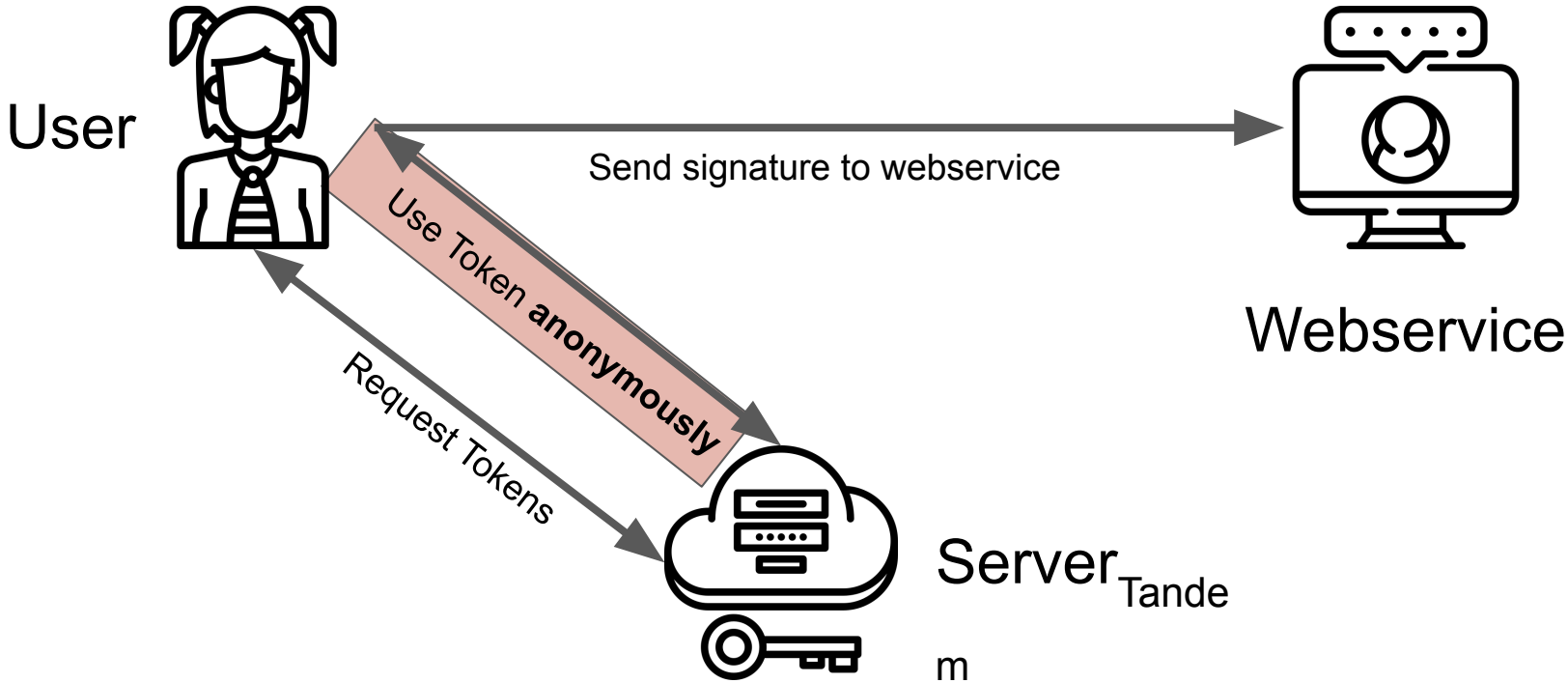
Création jetons:

- $x_U, x_S \rightarrow x_U + \partial, x_S - \partial \rightarrow \underline{x}_U, \underline{x}_S$ avec $\underline{x}_U + \underline{x}_S = x$
  plein de preuves et cachotteries

Utilisation jeton:

- Demander au serveur d'utiliser un $\underline{x}_S$ tout en apprenant ∂

Tandem



Randomizing linearly Monero - 1

(s_B^v, s_B^s) and (K_B^v, K_B^s)

$(s_{B,U}^v, s_{B,U}^s)$ and (K_B^v, K_B^s)

Bob

$(s_{B,S}^v, s_{B,S}^s)$

$(r \cdot G, K^{out})$

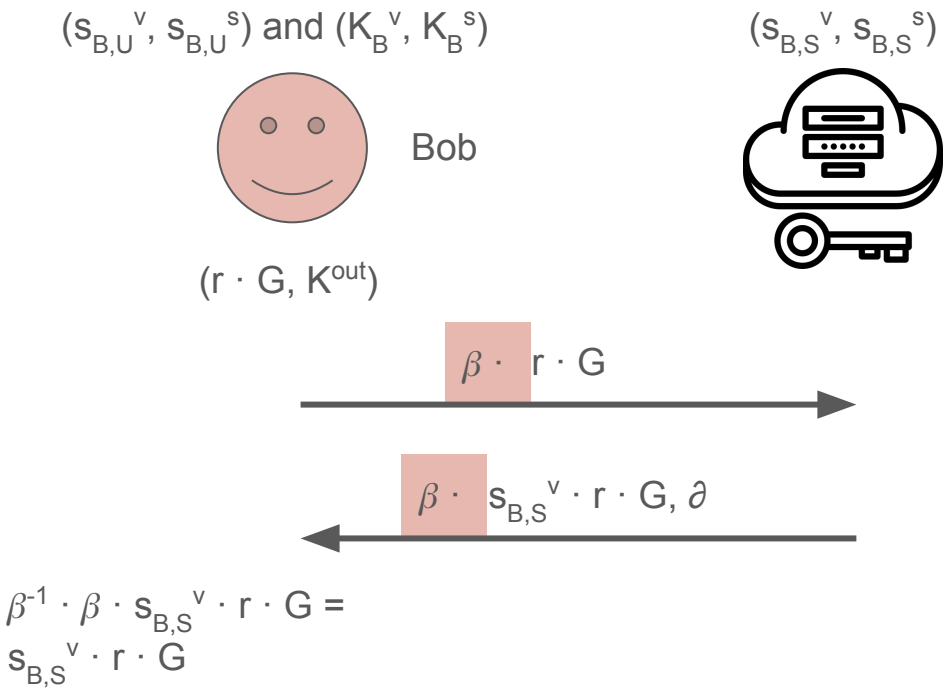
$r \cdot G$

$s_{B,S}^v \cdot r \cdot G, \partial$

$s_{B,U}^v \cdot r \cdot G + s_{B,S}^v \cdot r \cdot G =$
$(s_{B,U}^v + s_{B,S}^v) \cdot r \cdot G =$
$s_B^v \cdot r \cdot G$

Le serveur apprend ce que Bob cherche!

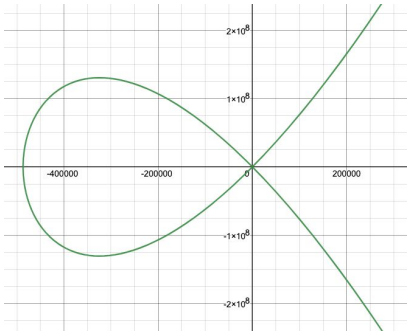
Blindage

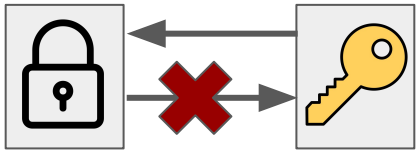


Ce qu'on va voir

- **Le problème**
  - Pourquoi les clés privées décentralisées sont compliquées?
- **Courbes elliptiques**
  - Plus petit et plus rapide que RSA
- **Monero**
  - Une cryptomonnaie anonyme
- **Tandem**
  - Gestion anonyme de clés privées
- **Conclusion**

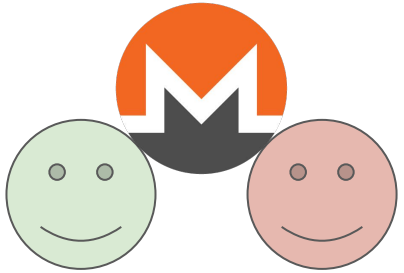
La cryptographie, c'est fun





Clé publique

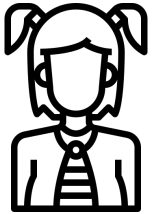
Clé privée



Alice

Bob

User



Webservice

Server

Tandem

$$s_{B,U}{}^{v} \cdot r \cdot G + s_{B,S}{}^{v} \cdot r \cdot G = (s_{B,U}{}^{v} + s_{B,S}{}^{v}) \cdot r \cdot G = s_{B}{}^{v} \cdot r \cdot G$$

$$\beta \cdot r \cdot G \cdot \beta^{-1} = r \cdot G$$

https://go.epfl.ch/ssie-2023

Links

- Cloudflare Elliptic Curves explainer
- Tandem paper by Wouter Lueks
- Using Tandem in Monero by Linus Gasser and Wouter Lueks
- From Zero to Monero for more details on zero-knowledge proofs and ring signatures
- Est-ce que nous vivons dans un univers où la cryptographie est possible?