Finanziato dall'Unione europea NextGenerationEU

Ministero dell'Università e della Ricerca

Italiadomani PIANO NAZIONALE DI RIPRESA E RESILIENZA

SERICS SECURITY AND RIGHTS IN THE CYBERSPACE

CrypTO CONFERENCE

Politecnico di Torino 1859

Telsy A TIM ENTERPRISE BRAND

ISMA

CISPA
HELMHOLTZ CENTER FOR
INFORMATION SECURITY

On the Security of Schnorr-type Blind Signatures with Repetitions

Eugenio Paracucchi

CrypTO | 23 Maggio 2025 | Torino

IND-ice

I. Introduzione alle blind signatures
II. One-more unforgeability
III. Schnorr-type blind signatures
IV. Attacchi
V. Conclusioni

Blind Signatures

Introdotte da David Chaum negli anni ’80. Utilizzate per rilasciare **one-time anonymous tokens**

Blind Signatures

Introdotte da David Chaum negli anni ’80. Utilizzate per rilasciare **one-time anonymous tokens**

💰 **e-Cash**

Blind Signatures

Introdotte da David Chaum negli anni ’80. Utilizzate per rilasciare **one-time anonymous tokens**

- 💰 **e-Cash**
- 🪪 **Anonymous Credentials**

Blind Signatures

Introdotte da David Chaum negli anni ’80. Utilizzate per rilasciare **one-time anonymous tokens**

- 💰 **e-Cash**
- 🪪 **Anonymous Credentials**
- 🎟️ **Privacy Pass**

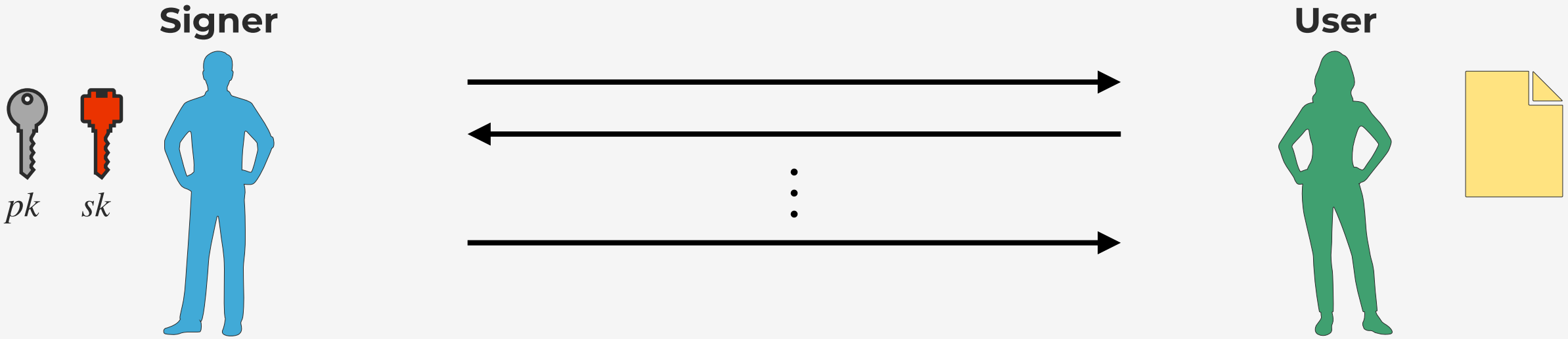
Blind Signatures

Introdotte da David Chaum negli anni ’80. Utilizzate per rilasciare **one-time anonymous tokens**

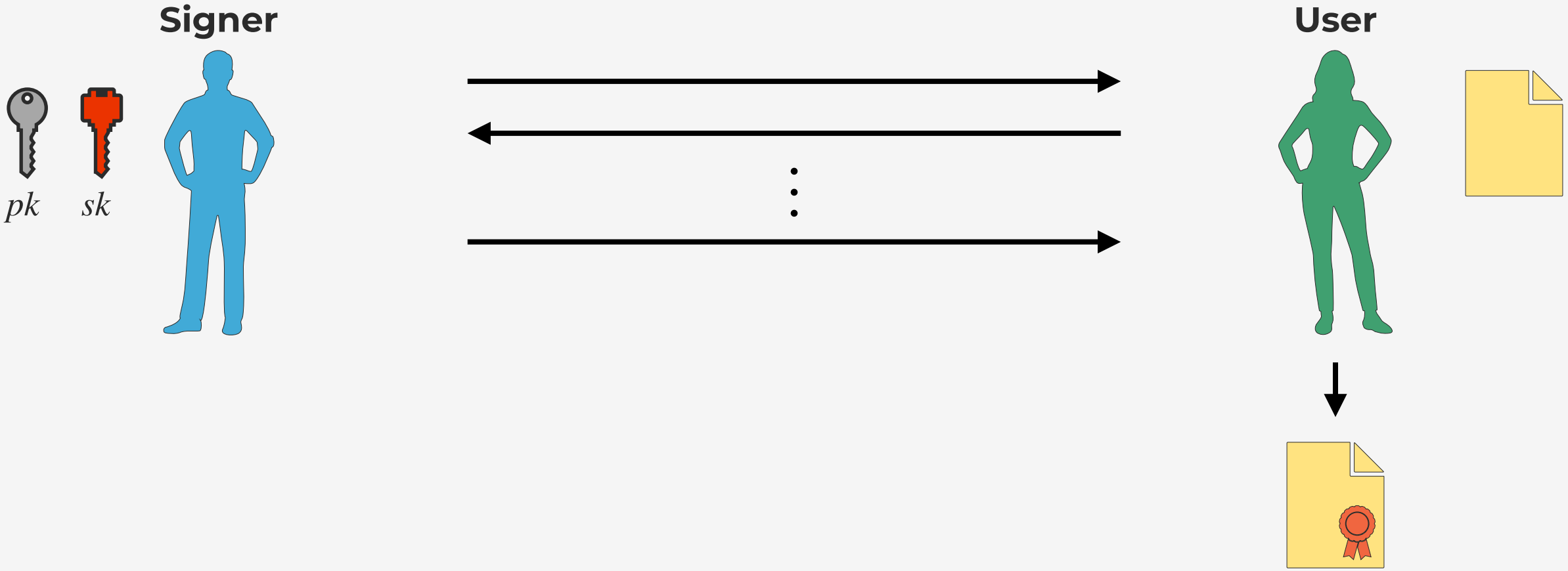
- 💰 **e-Cash**
- 🪪 **Anonymous Credentials**
- 🎟️ **Privacy Pass**

Dunque è importante studiarne la **sicurezza**

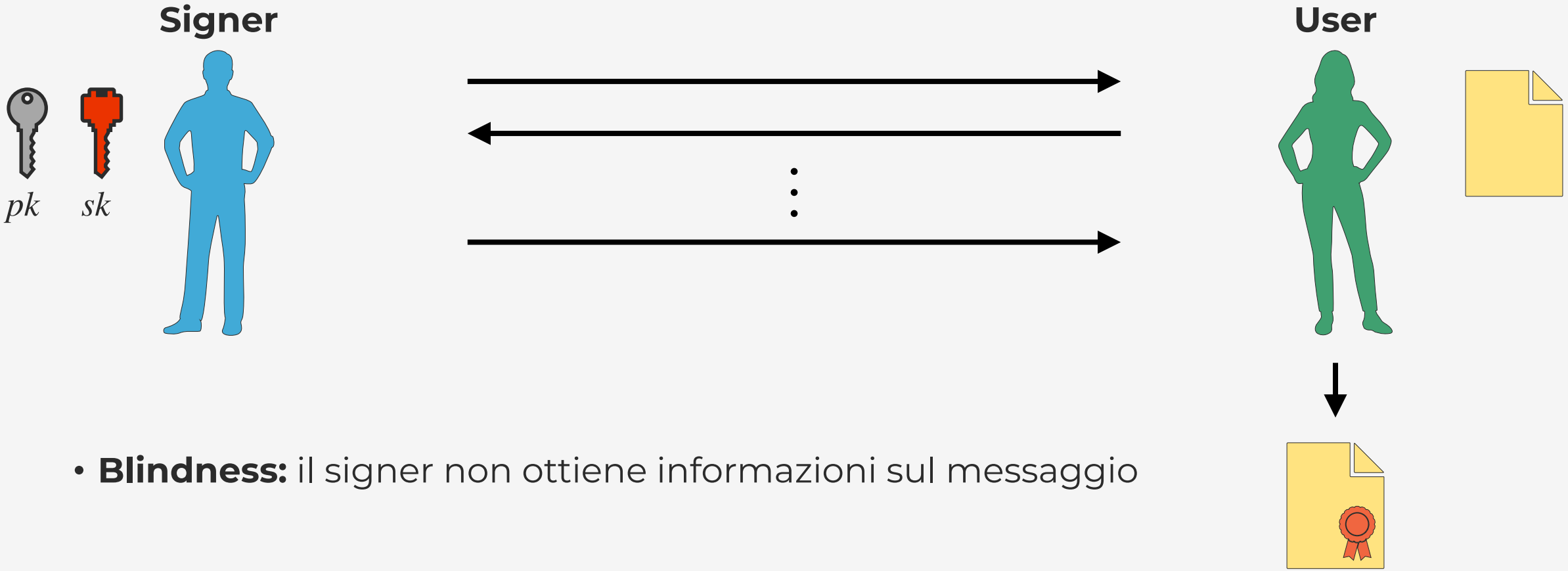
Blind Signatures



Blind Signatures

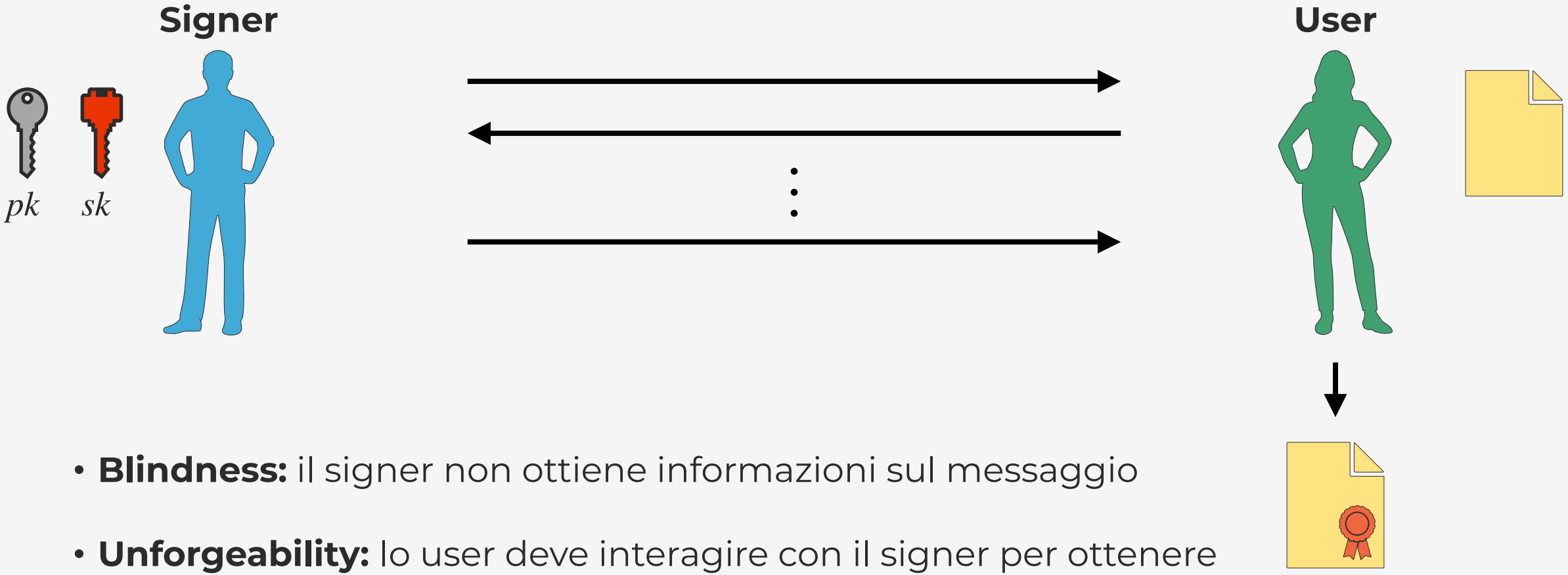


Blind Signatures



- **Blindness:** il signer non ottiene informazioni sul messaggio

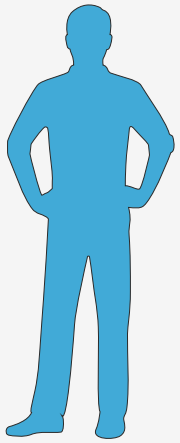
Blind Signatures



- **Blindness:** il signer non ottiene informazioni sul messaggio
- **Unforgeability:** lo user deve interagire con il signer per ottenere una firma valida

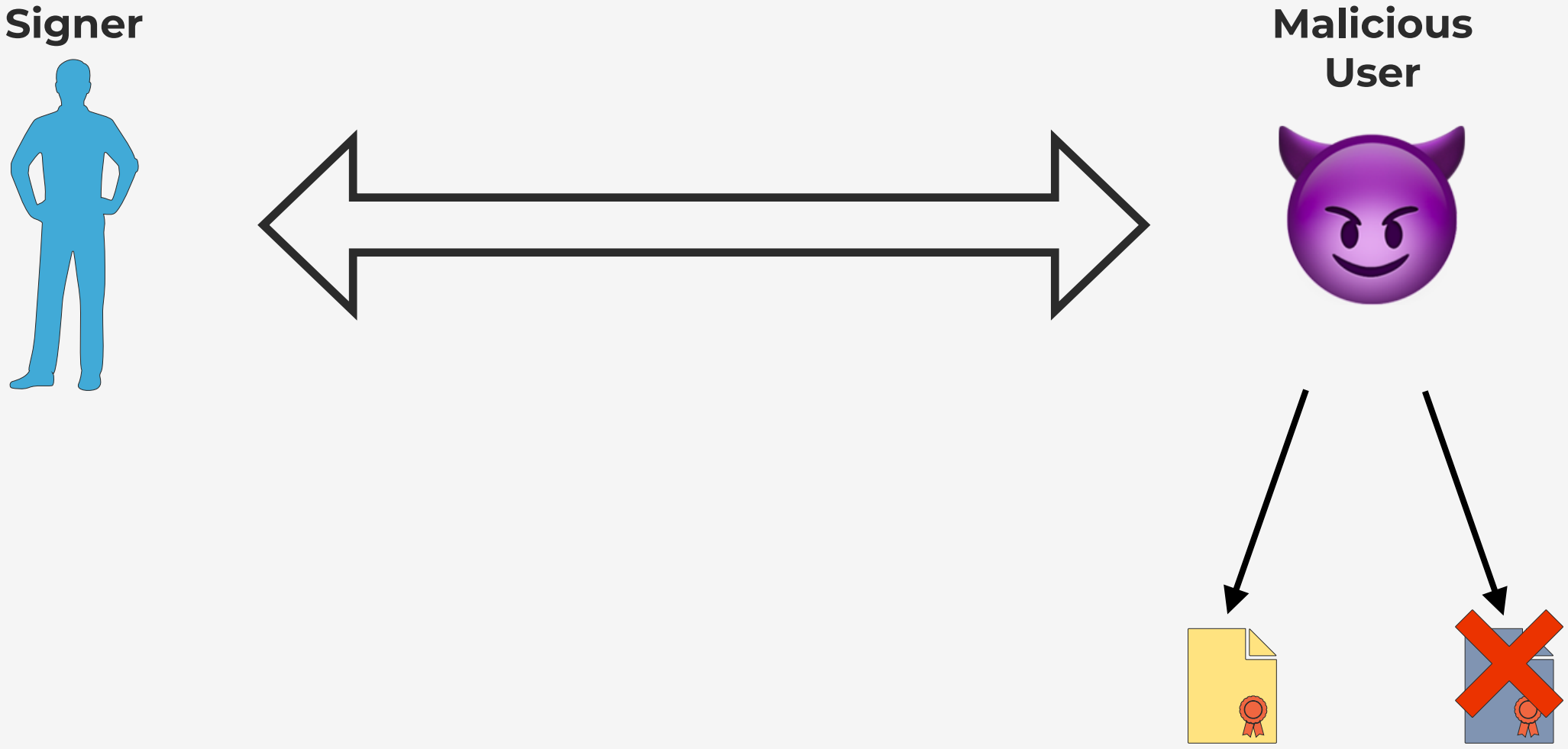
One More Unforgeability

Signer

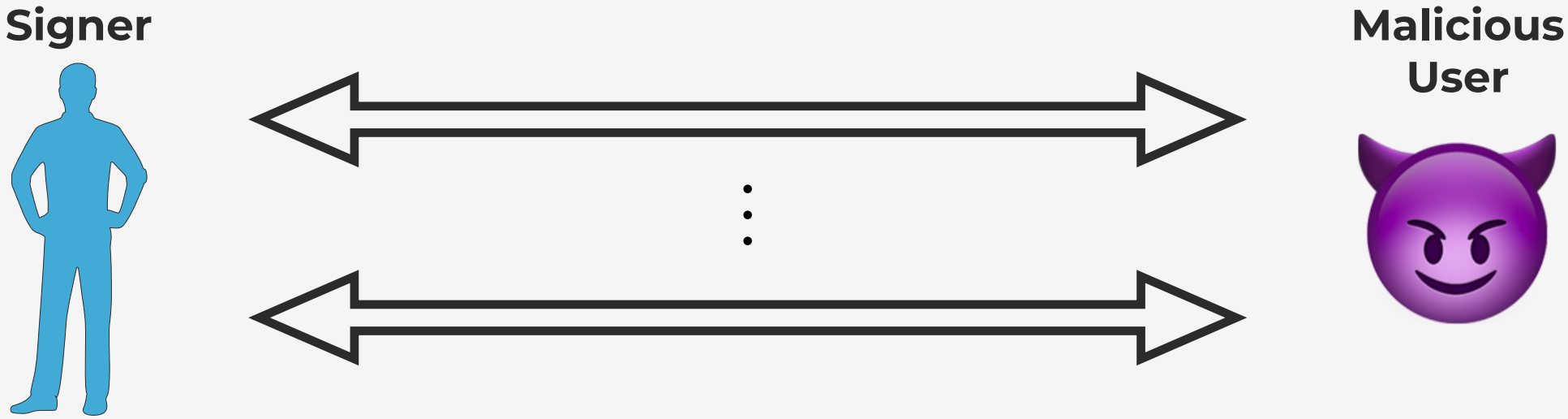


Malicious User

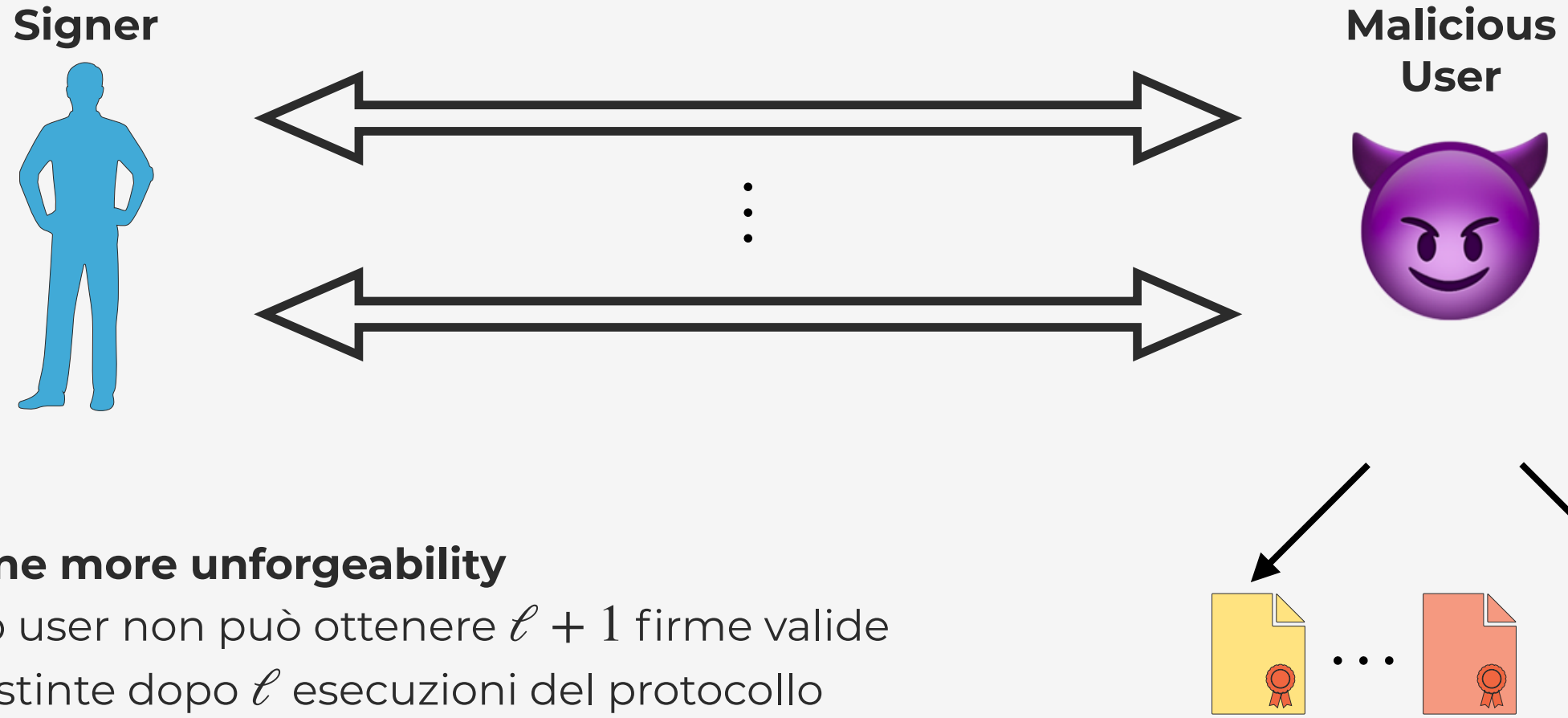
One More Unforgeability



One More Unforgeability



One More Unforgeability



One more unforgeability

Lo user non può ottenere $\ell + 1$ firme valide distinte dopo ℓ esecuzioni del protocollo con il signer

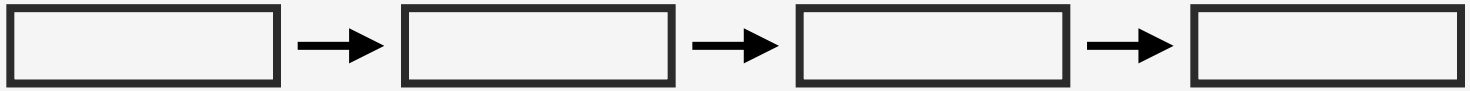
Sequential vs Concurrent

A secoda di come vengono aperte le sessioni abbiamo due setting

Sequential vs Concurrent

A secoda di come vengono aperte le sessioni abbiamo due setting

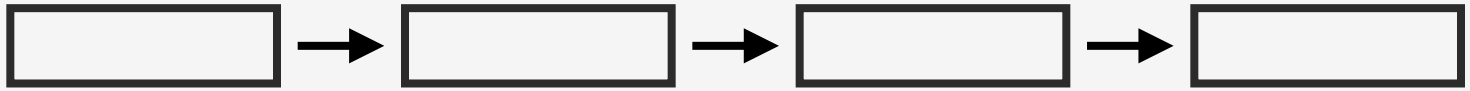
- **Sequential**: una nuova sessione viene aperta dopo aver chiuso quella precedente



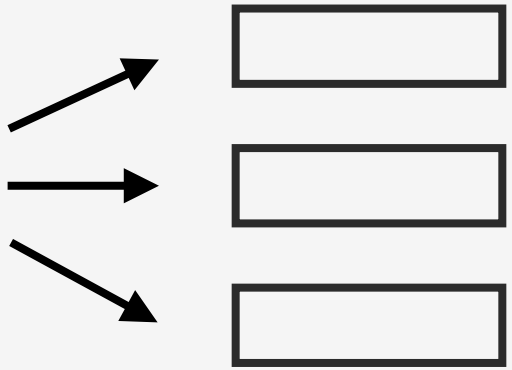
Sequential vs Concurrent

A secoda di come vengono aperte le sessioni abbiamo due setting

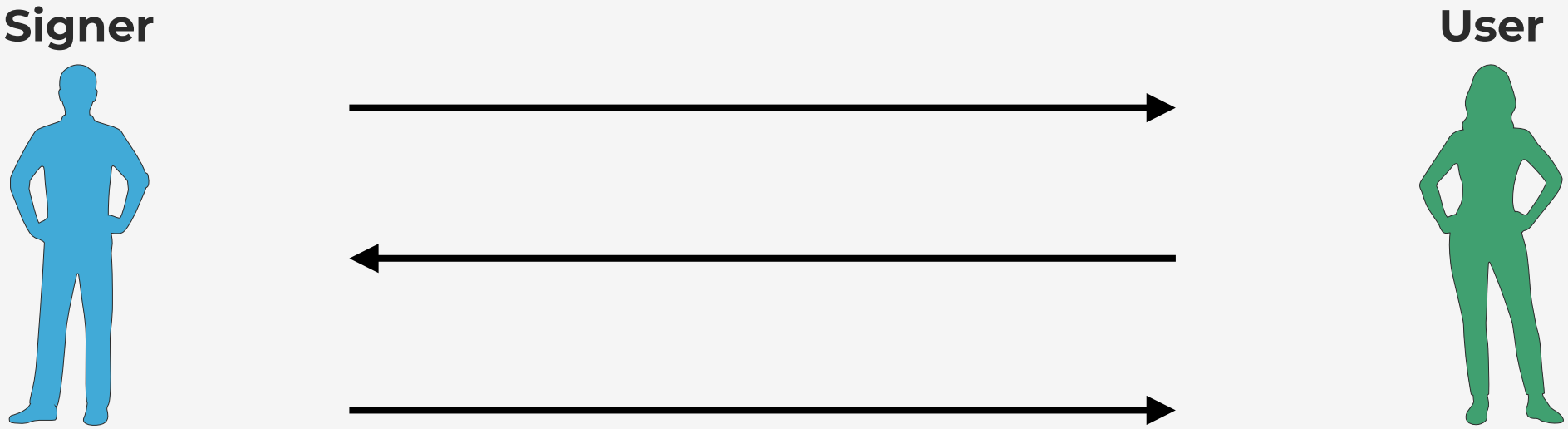
- **Sequential**: una nuova sessione viene aperta dopo aver chiuso quella precedente



- **Concurrent**: gli utenti possono aprire le sessioni in parallelo



Schnorr-type Blind Signatures



Schnorr-type Blind Signatures

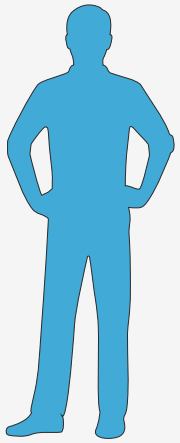
Blind signatures costruite sul paradigma **identification schemes** (sigma protocols) più **Fiat-Shamir transform**

Schnorr-type Blind Signatures

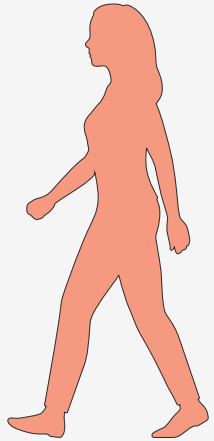
Blind signatures costruite sul paradigma **identification schemes** (sigma protocols) più **Fiat-Shamir transform**

Identification Schemes

Prover

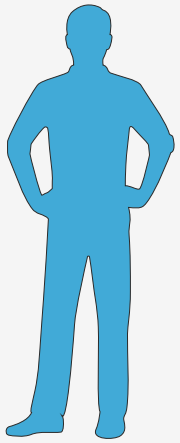


Verifier



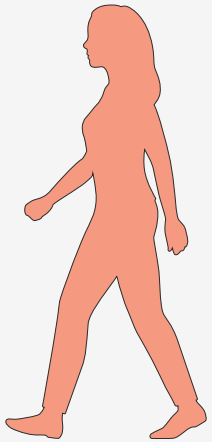
Identification Schemes

Prover



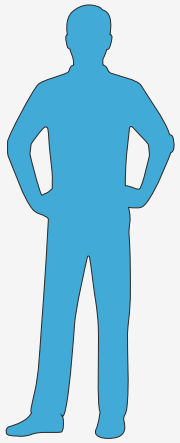
commitment: R

Verifier



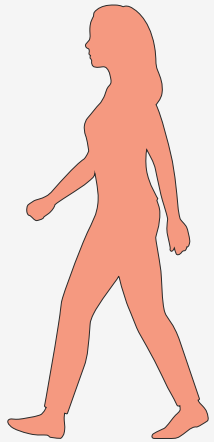
Identification Schemes

Prover **Verifier**



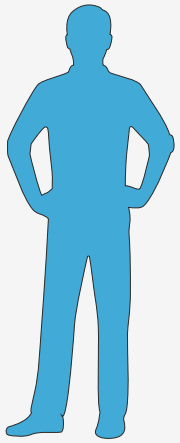
commitment: R

challenge: $c \in \mathscr{C}$



Identification Schemes

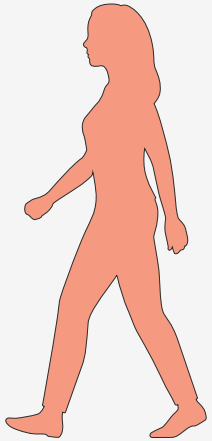
Prover **Verifier**



commitment: R →

← challenge: $c \in \mathscr{C}$

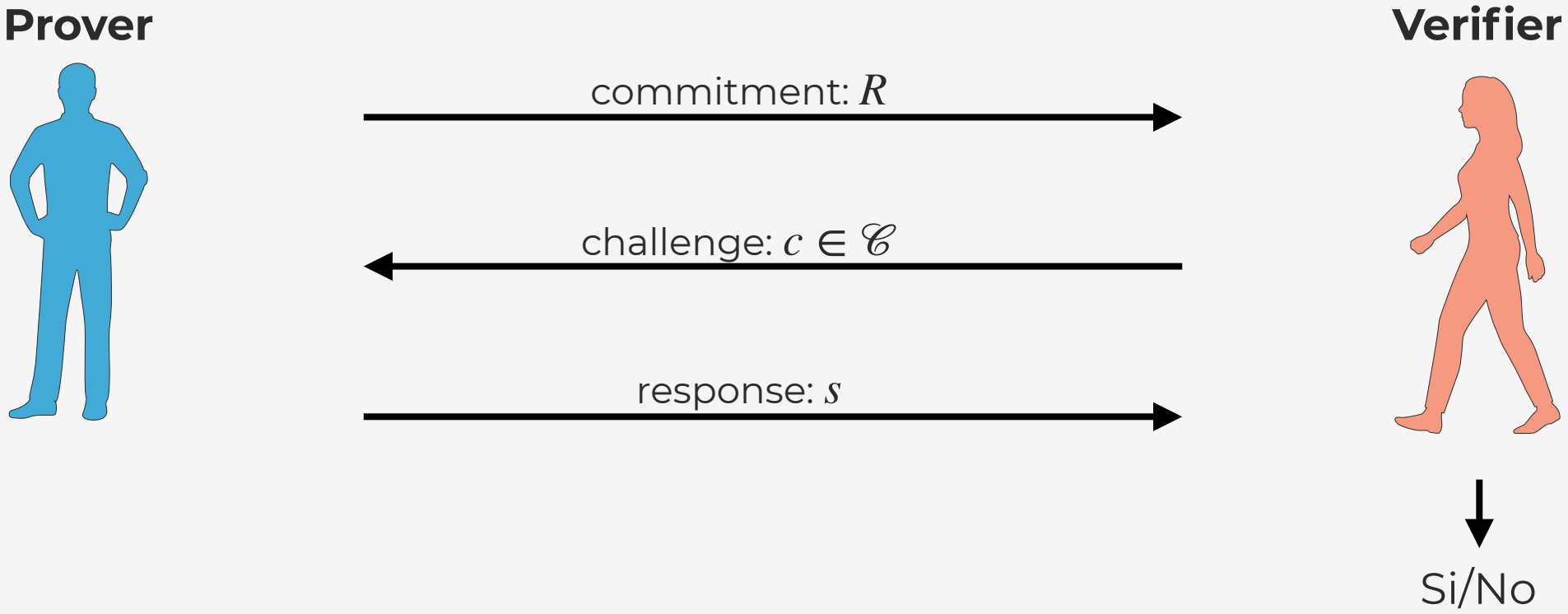
response: s →



↓

Si/No

Identification Schemes

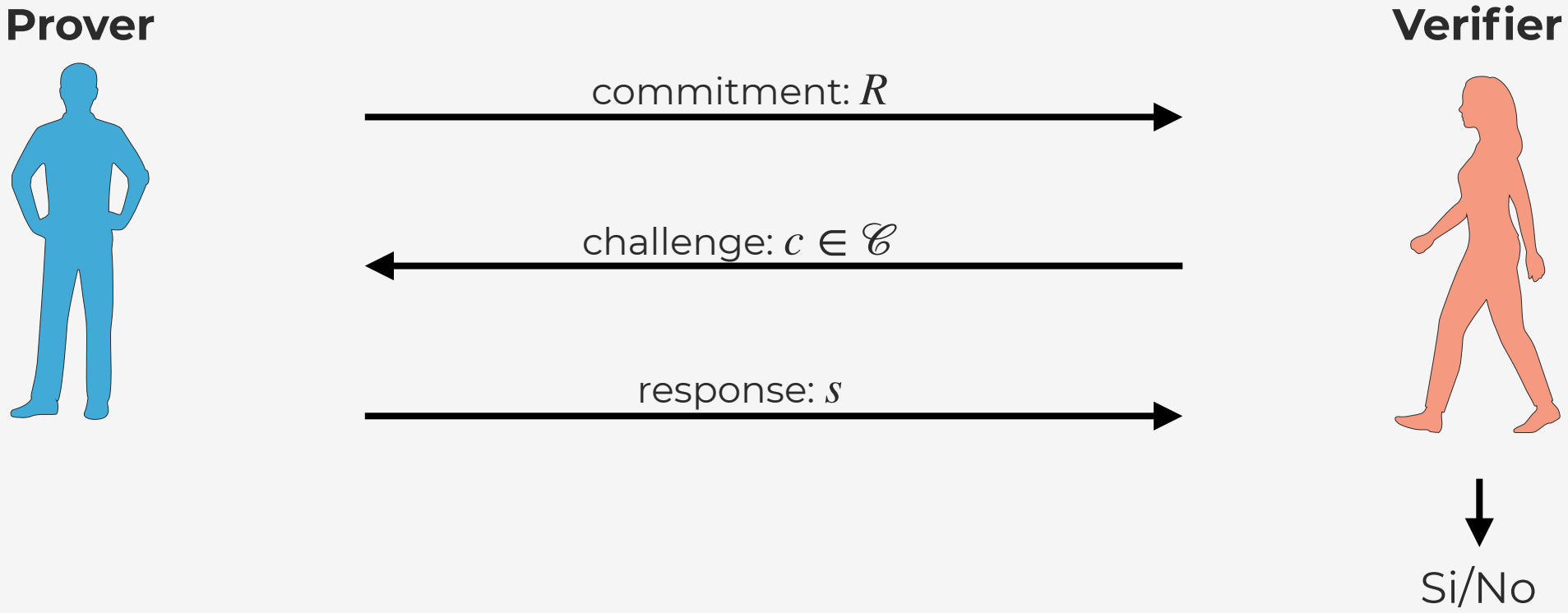


- **Correctness:** un prover onesto riesce sempre a convincere il verifier

Identification Schemes

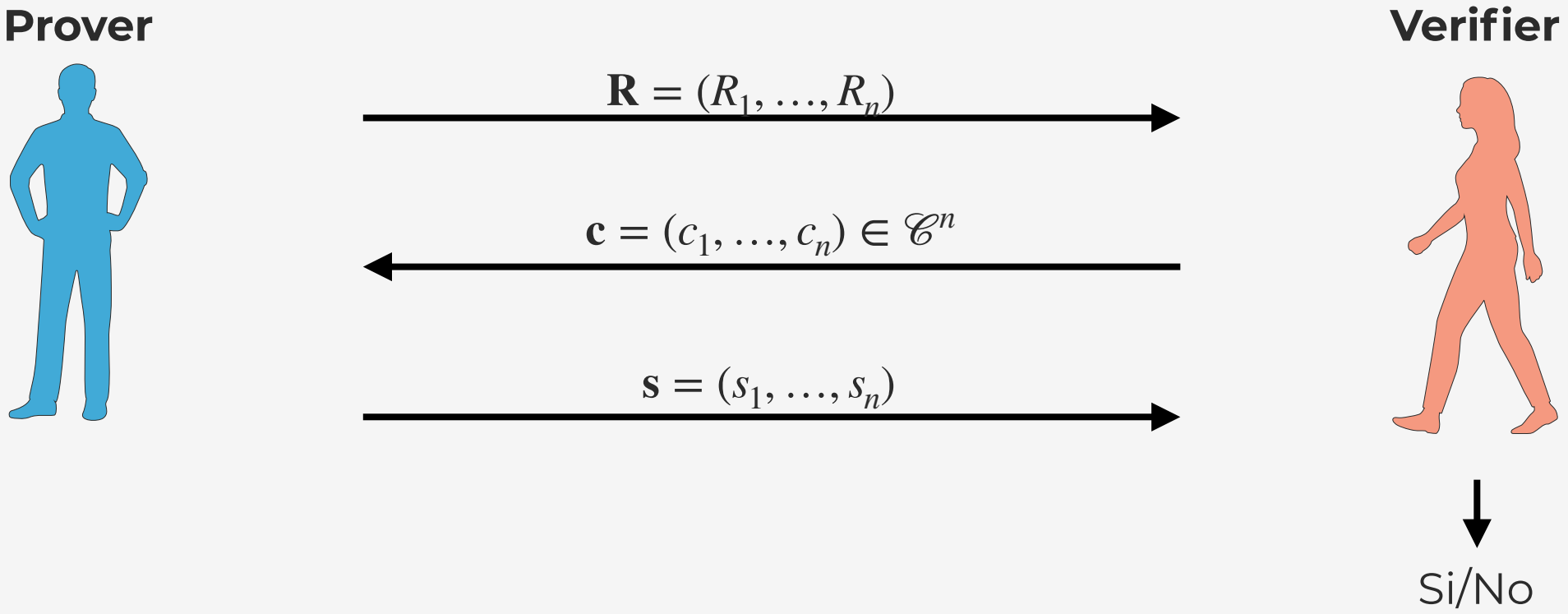
- **Correctness:** un prover onesto riesce sempre a convincere il verifier
- **Soundness:** un prover disonesto riesce con probabilità $1/|\mathscr{C}|$

Identification Schemes



- **Correctness:** un prover onesto riesce sempre a convincere il verifier
- **Soundness:** un prover disonesto riesce con probabilità $1/|\mathscr{C}|$
- **HVZK:** esiste un simulatore che, data una challenge $c \in \mathscr{C}$, costruisce un transcript valido del protocollo

Ripetizioni in Parallelo



Se $\mathscr{C}$ è piccolo allora ripetiamo il protocollo n volte in parallelo per aumentare la sicurezza: la probabilità di imbrogliare per un prover disonesto è $1/|\mathscr{C}|^n$

Fiat-Shamir Transform

Rimpiazziamo l'interazione con il verifier con una chiamata a un random oracle $\mathscr{H} : \{0,1\}^* \to \mathscr{C}$

Sign

$m \longrightarrow$

$sk \longrightarrow$

$$\boxed{\begin{array}{l} 1.\ R \leftarrow \mathsf{commit}(sk) \\ 2.\ c \leftarrow \mathscr{H}(R, m) \\ 3.\ s \leftarrow \mathsf{resp}(R, c, sk) \end{array}} \longrightarrow (R, s)$$

Ripetizioni in parallelo

Se $|\mathscr{C}|$ è piccolo, ripetiamo n volte in parallelo per aumentare la sicurezza

Sign

$m \longrightarrow$

$sk \longrightarrow$

$$\boxed{\begin{array}{l} 1.\ \mathbf{R} \leftarrow \mathsf{commit}(sk) \\ 2.\ \mathbf{c} \leftarrow \mathscr{H}(\mathbf{R}, m) \\ 3.\ \mathbf{s} \leftarrow \mathsf{resp}(\mathbf{R}, \mathbf{c}, sk) \end{array}} \longrightarrow (\mathbf{R}, \mathbf{s})$$

Schnorr-type Blind Signatures

Signer(pk **,** sk **)**

User(pk **,** m **)**

$R \leftarrow \mathsf{commit}(sk)$

$\xrightarrow{\quad R \quad}$

$c \leftarrow \mathscr{H}(R, m)$

$\xleftarrow{\quad c \quad}$

$s \leftarrow \mathsf{resp}(R, c, sk)$

$\xrightarrow{\quad s \quad}$

$\downarrow$

(R, s)

Schnorr-type Blind Signatures

Signer(pk **,** sk **)**

User(pk **,** m **)**

$R \leftarrow \text{commit}(sk)$

$\xrightarrow{\quad R \quad}$

$R' \leftarrow \text{blind}(R)$

$c' \leftarrow \mathscr{H}(R', m)$

$c \leftarrow \text{blind}(c')$

$\xleftarrow{\quad c \quad}$

$s \leftarrow \text{resp}(R, c, sk)$

$\xrightarrow{\quad s \quad}$

$s' \leftarrow \text{unblind}(s)$

$\downarrow$

(R', s')

Schnorr-type Blind Signatures con Ripetizioni

Signer(pk **,** sk **)**

User(pk **,** m **)**

$\mathbf{R} \leftarrow \text{commit}(sk)$

$\xrightarrow{\mathbf{R}}$

$\mathbf{R}' \leftarrow \text{blind}(\mathbf{R})$

$\mathbf{c}' \leftarrow \mathscr{H}(\mathbf{R}', m)$

$\mathbf{c} \leftarrow \text{blind}(\mathbf{c}')$

$\xleftarrow{\mathbf{c}}$

$\mathbf{s} \leftarrow \text{resp}(\mathbf{R}, \mathbf{c}, sk)$

$\xrightarrow{\mathbf{s}}$

$\mathbf{s}' \leftarrow \text{unblind}(\mathbf{s})$

$\downarrow$

$(\mathbf{R}', \mathbf{s}')$

L’Attacco [DHP24]

Costruiamo un avversario $\mathscr{A}$ contro la **one more unforgeability** di una Schnorr-type blind signature con:

- challenge space $\mathscr{C}$ piccolo (polinomiale nel parametro di sicurezza n)
- n ripetizioni parallele

L'Attacco [DHP24]

Costruiamo un avversario $\mathscr{A}$ contro la **one more unforgeability** di una Schnorr-type blind signature con:

- challenge space $\mathscr{C}$ piccolo (polinomiale nel parametro di sicurezza n)
- n ripetizioni parallele

I. **n-out-of-n**: $n + 1$ firme dopo n sessioni concurrent

L'Attacco [DHP24]

Costruiamo un avversario $\mathscr{A}$ contro la **one more unforgeability** di una Schnorr-type blind signature con:

- challenge space $\mathscr{C}$ piccolo (polinomiale nel parametro di sicurezza n)
- n ripetizioni parallele

I. **n-out-of-n**: $n + 1$ firme dopo n sessioni concurrent

II. **2-out-of-n**: $n + 1$ firme dopo n sessioni per uno schema che supporta al più due sessioni concurrent

L'Attacco [DHP24]

Costruiamo un avversario $\mathscr{A}$ contro la **one more unforgeability** di una Schnorr-type blind signature con:

- challenge space $\mathscr{C}$ piccolo (polinomiale nel parametro di sicurezza n)
- n ripetizioni parallele

I. **n-out-of-n**: $n + 1$ firme dopo n sessioni concurrent

II. **2-out-of-n**: $n + 1$ firme dopo n sessioni per uno schema che supporta al più due sessioni concurrent

Runtime: $\mathcal{O}(n \cdot |\mathscr{C}|)$

L'Attacco [DHP24]

Costruiamo un avversario $\mathscr{A}$ contro la **one more unforgeability** di una Schnorr-type blind signature con:

- challenge space $\mathscr{C}$ piccolo (polinomiale nel parametro di sicurezza n)
- n ripetizioni parallele

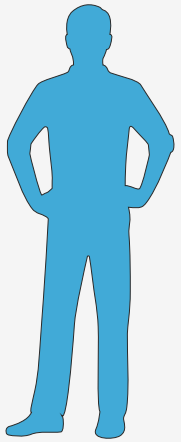
Resto del talk

I. **n-out-of-n**: $n + 1$ firme dopo n sessioni concurrent

II. **2-out-of-n**: $n + 1$ firme dopo n sessioni per uno schema che supporta al più due sessioni concurrent

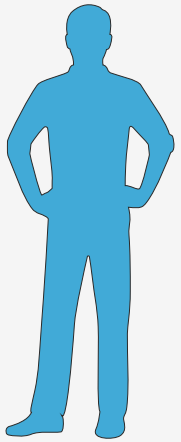
Runtime: $\mathcal{O}(n \cdot |\mathscr{C}|)$

Idea Principale (Unblind, n=3)



$$\mathbf{R} = (R_1, R_2, R_3) \longrightarrow$$

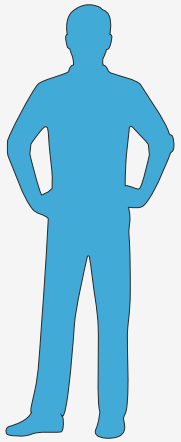
Idea Principale (Unblind, n=3)



$$\xrightarrow{\mathbf{R} = (R_1, R_2, R_3)}$$

- Simulare un transcript valido (e, d, f) e sostituire $\mathbf{R}$ con (e, R_2, R_3)
- Trovare m tale che $\mathbf{c} = \mathscr{H}(m, (e, R_2, R_3)) = (d, c_2, c_3)$
- Richiede $\mathcal{O}(|\mathscr{C}|)$ queries ad $\mathscr{H}$

Idea Principale (Unblind, n=3)



$$\mathbf{R} = (R_1, R_2, R_3) \longrightarrow$$

- Simulare un transcript valido (e, d, f) e sostituire $\mathbf{R}$ con (e, R_2, R_3)
- Trovare m tale che $\mathbf{c} = \mathscr{H}(m, (e, R_2, R_3)) = (d, c_2, c_3)$
- Richiede $\mathcal{O}(|\mathscr{C}|)$ queries ad $\mathscr{H}$

$$\longleftarrow (*, c_2, c_3)$$

$$\mathbf{s} \longrightarrow$$

Vantaggio: l'attaccante può utilizzare il response per R_1 in un'altra firma

N-out-of-n (Unblind, n=3)

Sessione 1	$R_{1,1}$	$R_{1,2}$	$R_{1,3}$
Sessione 2	$R_{2,1}$	$R_{2,2}$	$R_{2,3}$
Sessione 3	$R_{3,1}$	$R_{3,2}$	$R_{3,3}$

N-out-of-n (Unblind, n=3)

Sessione 1	$R_{1,1}$	$R_{1,2}$	$R_{1,3}$
Sessione 2	$R_{2,1}$	$R_{2,2}$	$R_{2,3}$
Sessione 3	$R_{3,1}$	$R_{3,2}$	$R_{3,3}$

N-out-of-n (Unblind, n=3)

Sessione 1		$R_{1,2}$	$R_{1,3}$
Sessione 2	$R_{2,1}$		$R_{2,3}$
Sessione 3	$R_{3,1}$	$R_{3,2}$	

Forgery	$R_{1,1}$	$R_{2,2}$	$R_{3,3}$

N-out-of-n (Unblind, n=3)

Sessione 1		$R_{1,2}$	$R_{1,3}$
Sessione 2	$R_{2,1}$		$R_{2,3}$
Sessione 3	$R_{3,1}$	$R_{3,2}$	

Forgery	$R_{1,1}$	$R_{2,2}$	$R_{3,3}$

$$\mathscr{H}(m^*, (R_{1,1}, R_{2,2}, R_{3,3})) = (c_{4,1}, c_{4,2}, c_{4,3})$$

N-out-of-n (Unblind, n=3)

Sessione 1	e_1	$R_{1,2}$	$R_{1,3}$
Sessione 2	$R_{2,1}$	e_2	$R_{2,3}$
Sessione 3	$R_{3,1}$	$R_{3,2}$	e_3

Forgery	$R_{1,1}$	$R_{2,2}$	$R_{3,3}$

$$\mathscr{H}(m^*, (R_{1,1}, R_{2,2}, R_{3,3})) = (c_{4,1}, c_{4,2}, c_{4,3})$$

- Genera dei transcript (e_i, d_i, f_i) per $i = 1,2,3$

N-out-of-n (Unblind, n=3)

Sessione 1	e_1	$R_{1,2}$	$R_{1,3}$
Sessione 2	$R_{2,1}$	e_2	$R_{2,3}$
Sessione 3	$R_{3,1}$	$R_{3,2}$	e_3

$$\mathbf{c}_1 = \mathscr{H}(m_1, (e_1, R_{1,2}, R_{1,3})) = (d_1, *, *)$$

$$\mathbf{c}_2 = \mathscr{H}(m_2, (R_{2,1}, e_2, R_{2,3})) = (*, d_2, *)$$

$$\mathbf{c}_3 = \mathscr{H}(m_3, (R_{3,1}, R_{3,2}, e_3)) = (*, *, d_3)$$

Forgery	$R_{1,1}$	$R_{2,2}$	$R_{3,3}$

$$\mathbf{c}_4 = \mathscr{H}(m^*, (R_{1,1}, R_{2,2}, R_{3,3})) = (c_{4,1}, c_{4,2}, c_{4,3})$$

- Genera dei transcript (e_i, d_i, f_i) per $i = 1,2,3$
- Trova dei messaggi m_i per $i = 1,2,3$

N-out-of-n (Unblind, n=3)

Sessione 1	e_1	$R_{1,2}$	$R_{1,3}$
Sessione 2	$R_{2,1}$	e_2	$R_{2,3}$
Sessione 3	$R_{3,1}$	$R_{3,2}$	e_3
Forgery	$R_{1,1}$	$R_{2,2}$	$R_{3,3}$

$$\mathbf{c}_1 = \mathscr{H}(m_1, (e_1, R_{1,2}, R_{1,3})) = (d_1, *, *)$$

$$\mathbf{c}_2 = \mathscr{H}(m_2, (R_{2,1}, e_2, R_{2,3})) = (*, d_2, *)$$

$$\mathbf{c}_3 = \mathscr{H}(m_3, (R_{3,1}, R_{3,2}, e_3)) = (*, *, d_3)$$

$$\mathbf{c}_4 = \mathscr{H}(m^*, (R_{1,1}, R_{2,2}, R_{3,3})) = (c_{4,1}, c_{4,2}, c_{4,3})$$

- Genera dei transcript (e_i, d_i, f_i) per $i = 1,2,3$
- Trova dei messaggi m_i per $i = 1,2,3$
- Manda al signer: $(c_{4,1}, c_{1,2}, c_{1,3})$, $(c_{2,1}, c_{4,2}, c_{2,3})$, $(c_{3,1}, c_{3,2}, c_{4,3})$ e riceve i responses

N-out-of-n: Generalizzazione

Trovare un messaggio m tale che $\mathscr{H}(m, \mathbf{R}) = (d, c_2, \ldots, c_n)$ richiede $\mathcal{O}(|\mathscr{C}|)$ queries e n sessioni

N-out-of-n: Generalizzazione

Trovare un messaggio m tale che $\mathscr{H}(m, \mathbf{R}) = (d, c_2, \ldots, c_n)$ richiede $\mathcal{O}(|\mathscr{C}|)$ queries e n sessioni

Trovare un messaggio m tale che $\mathscr{H}(m, \mathbf{R}) = (d_1, d_2, \ldots, d_s, c_{s+1}, \ldots, c_n)$, richiede $\mathcal{O}(|\mathscr{C}|^s)$ queries e $\lceil n/s \rceil$ sessioni

Runtime: $\mathcal{O}(\lceil n/s \rceil \cdot |\mathscr{C}|^s)$

N-out-of-n: Generalizzazione

Trovare un messaggio m tale che $\mathscr{H}(m, \mathbf{R}) = (d, c_2, \ldots, c_n)$ richiede $\mathcal{O}(|\mathscr{C}|)$ queries e n sessioni

Trovare un messaggio m tale che $\mathscr{H}(m, \mathbf{R}) = (d_1, d_2, \ldots, d_s, c_{s+1}, \ldots, c_n)$, richiede $\mathcal{O}(|\mathscr{C}|^s)$ queries e $\lceil n/s \rceil$ sessioni

Runtime: $\mathcal{O}(\lceil n/s \rceil \cdot |\mathscr{C}|^s)$

$\Longrightarrow$ trade-off tra numero di queries ad $\mathscr{H}$ e numero di sessioni

Implicazioni

⚠️ **Risultato di impossibilità**: le Schnorr-type blind signatures con ripetizioni di un challenge space piccolo **non sono sicure concurrent**

Implicazioni

⚠️ **Risultato di impossibilità**: le Schnorr-type blind signatures con ripetizioni di un challenge space piccolo **non sono sicure concurrent**

👉 Per costruire delle blind signatures potenzialmente sicure con questo pardigma abbiamo bisogno di un identification scheme base con **challenge space grande** (esponenziale)

Implicazioni

⚠️ **Risultato di impossibilità**: le Schnorr-type blind signatures con ripetizioni di un challenge space piccolo **non sono sicure concurrent**

👉 Per costruire delle blind signatures potenzialmente sicure con questo pardigma abbiamo bisogno di un identification scheme base con **challenge space grande** (esponenziale)

🦦 **Schema colpito**: CSI-Otter [KLLQ23], la prima blind signature isogeny-based. Questo attacco nella pratica forgia **129 firme valide** dopo **128 sessioni concurrent**

Conclusione

🙂 CSI-Otter è **sequential sicuro** (nell'algebraic group action model). Risultato indipendente da [KLLQ23]

Conclusione

🙂 CSI-Otter è **sequential sicuro** (nell'algebraic group action model). Risultato indipendente da [KLLQ23]

🛠️ **Work in progress**: contromisure a questo attacco e firme sicure concurrent

Conclusione

🙂 CSI-Otter è **sequential sicuro** (nell'algebraic group action model). Risultato indipendente da [KLLQ23]

🛠️ **Work in progress**: contromisure a questo attacco e firme sicure concurrent

🔮 **Prospettive future**

- Generalizzazioni di questo attacco a challenge space esponenziale
- Sicurezza sequential (risultati generali)