CISPA
HELMHOLTZ CENTER FOR
INFORMATION SECURITY

Non-interactive Blind Signatures for Random Messages

Lucjan Hanzlik

Eurocrypt 2023, Lyon

Two-move Blind Signatures

User/Recipient

Signer

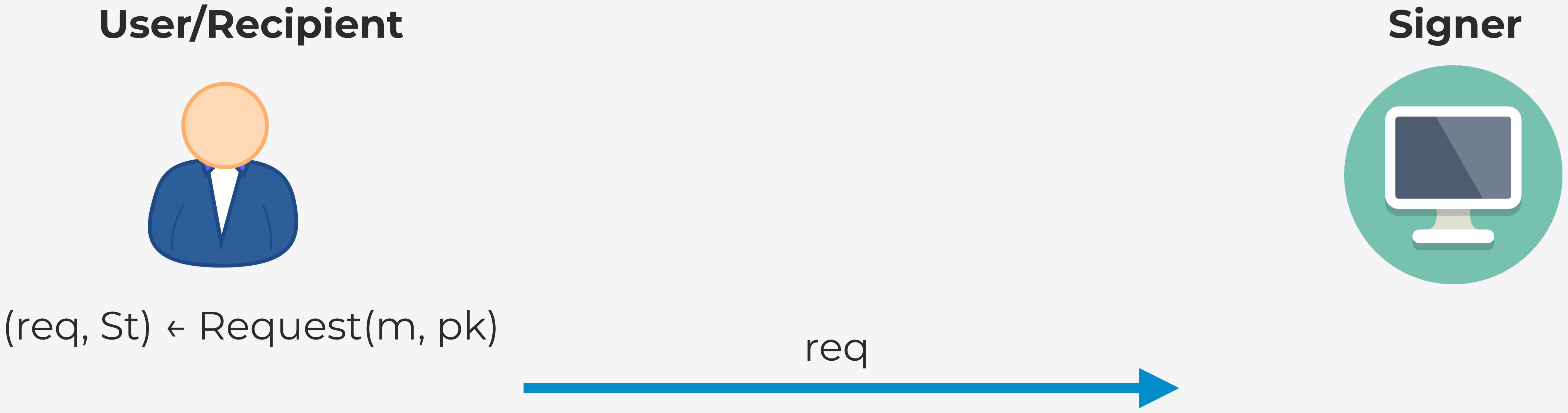
Two-move Blind Signatures

User/Recipient

Signer

$(\mathrm{req}, \mathrm{St}) \leftarrow \mathrm{Request}(m, pk)$

Two-move Blind Signatures



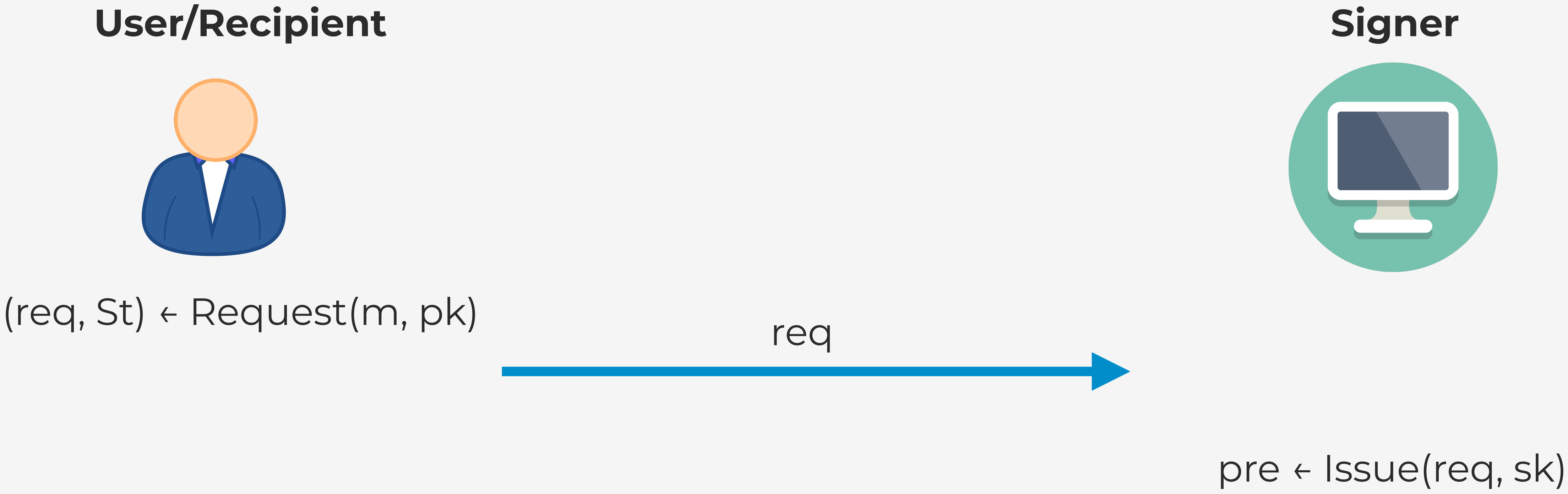
User/Recipient

Signer

(req, St) ← Request(m, pk)

req

Two-move Blind Signatures



User/Recipient

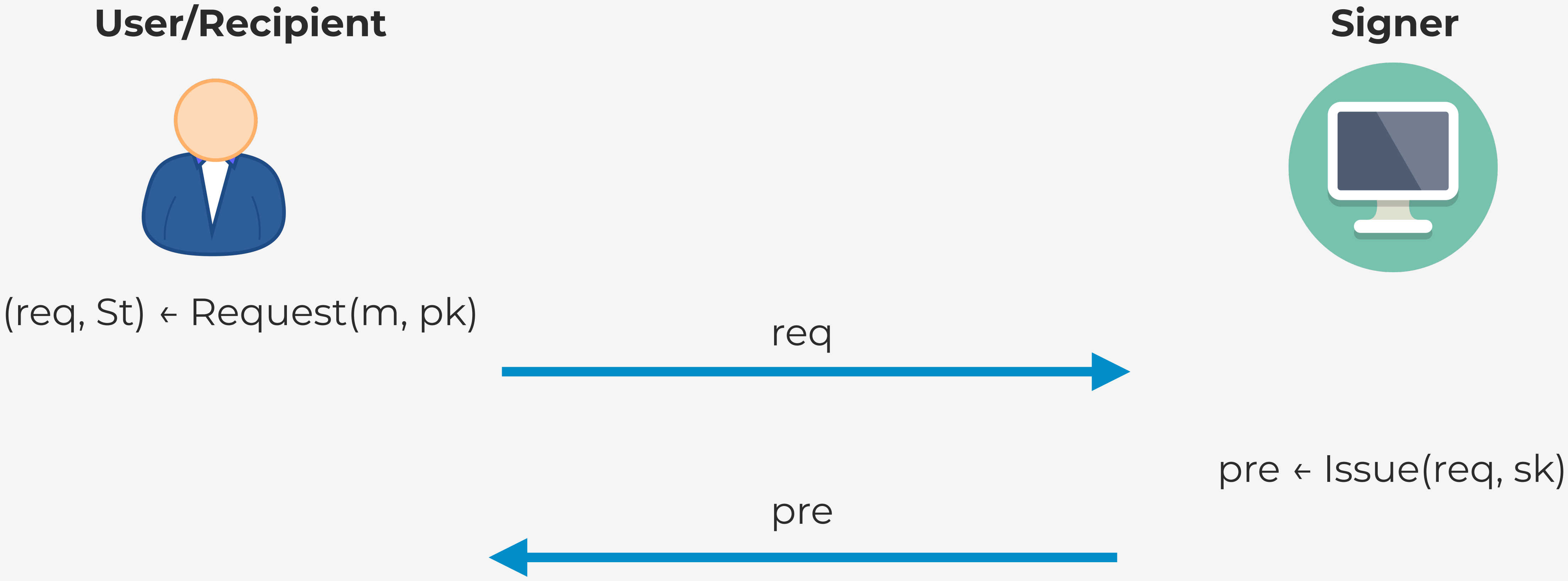
Signer

(req, St) ← Request(m, pk)

req

pre ← Issue(req, sk)

Two-move Blind Signatures



User/Recipient

Signer

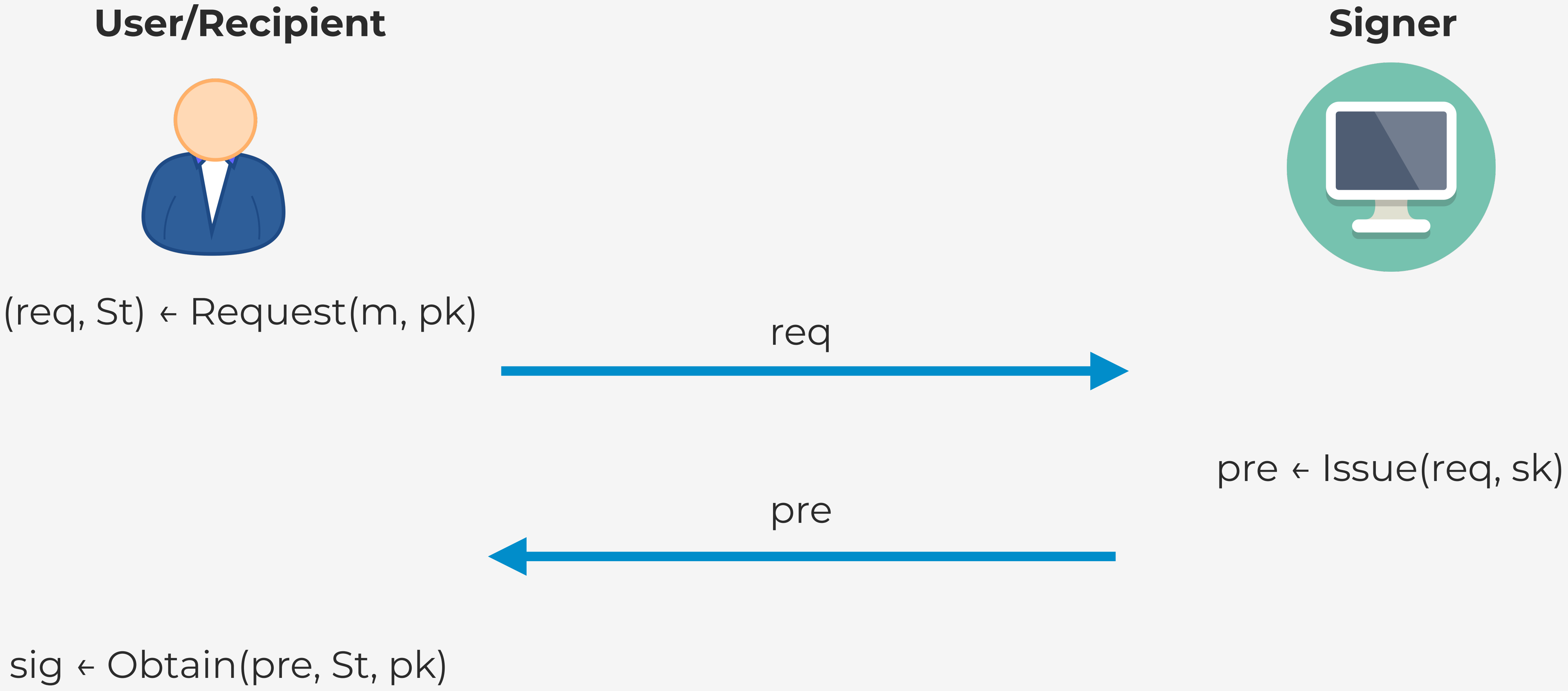
(req, St) ← Request(m, pk)

req

pre ← Issue(req, sk)

pre

Two-move Blind Signatures



User/Recipient

Signer

(req, St) ← Request(m, pk)

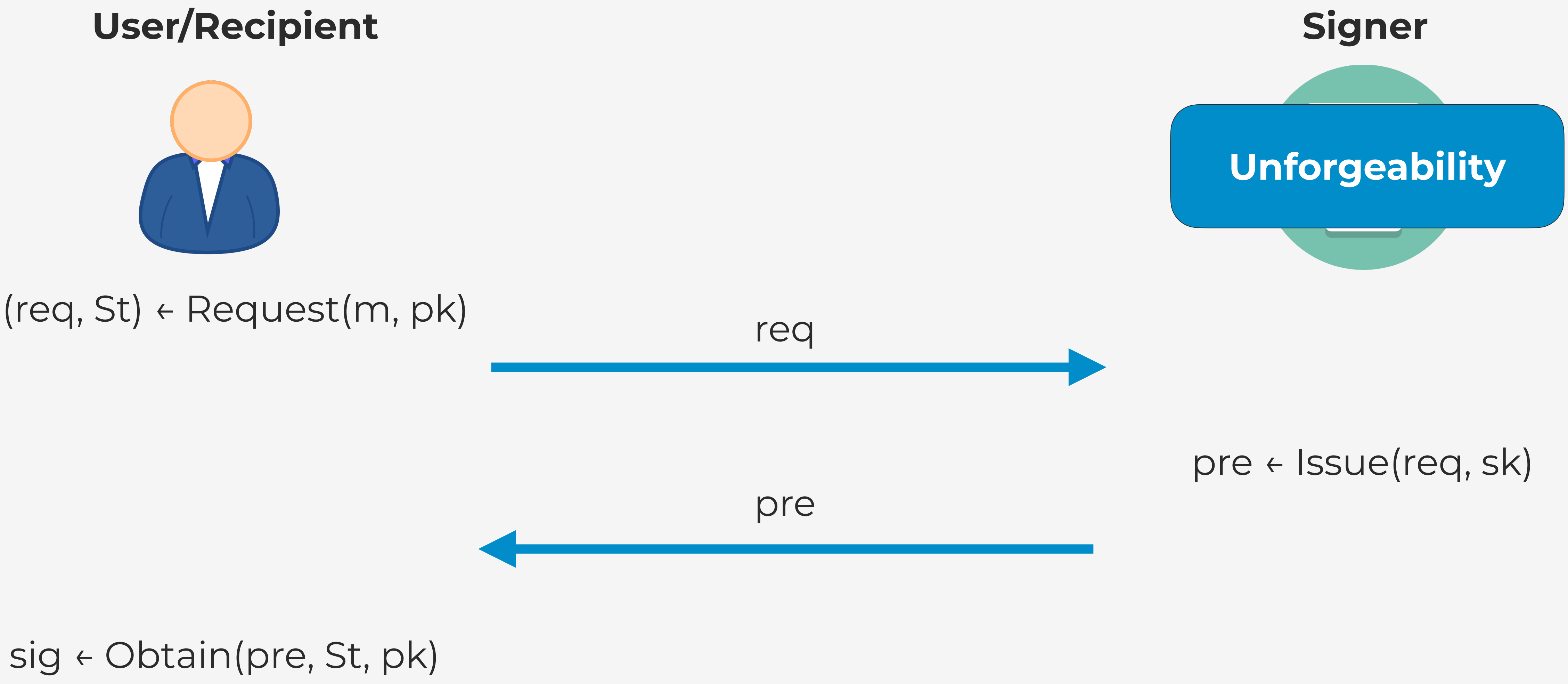
req

pre ← Issue(req, sk)

pre

sig ← Obtain(pre, St, pk)

Two-move Blind Signatures



User/Recipient

Signer

Unforgeability

(req, St) ← Request(m, pk)

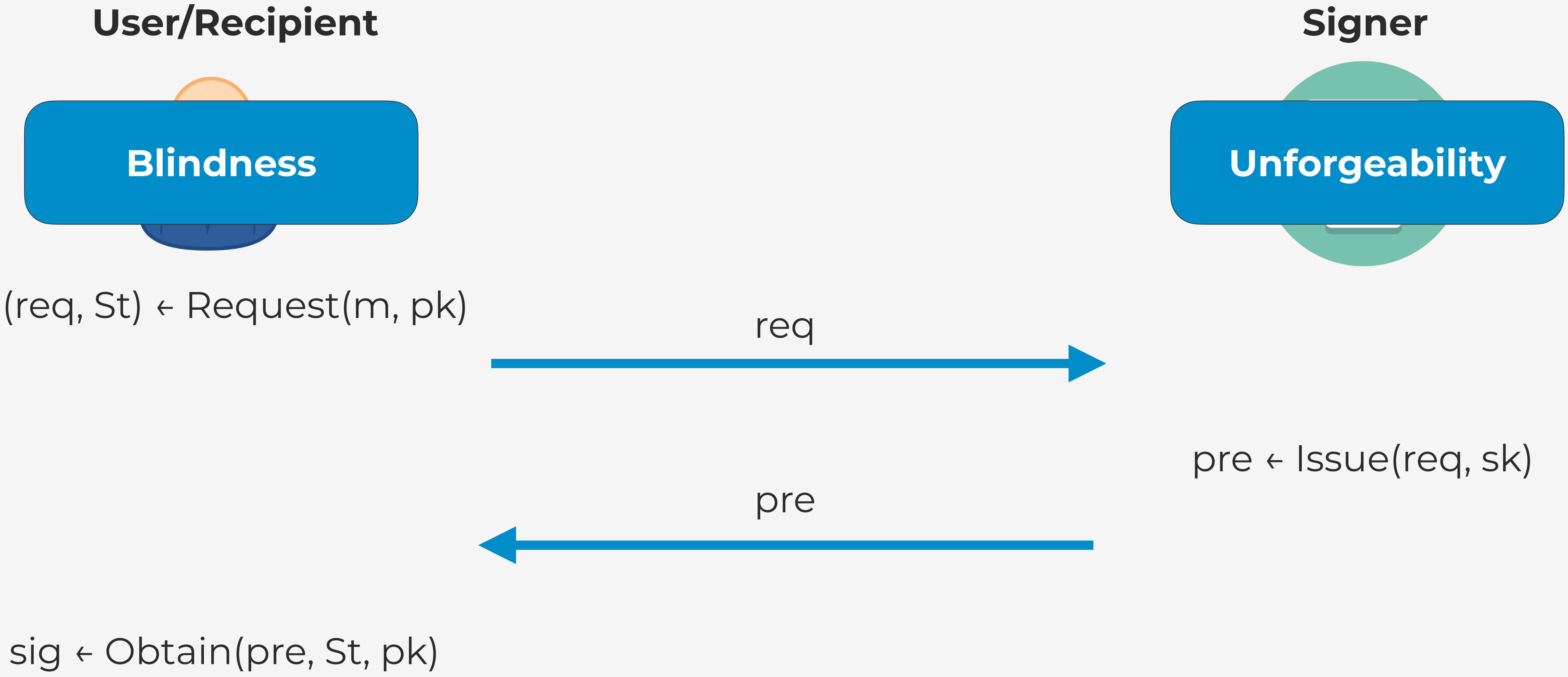
req

pre ← Issue(req, sk)

pre

sig ← Obtain(pre, St, pk)

Two-move Blind Signatures



User/Recipient

Signer

Blindness

Unforgeability

(req, St) ← Request(m, pk)

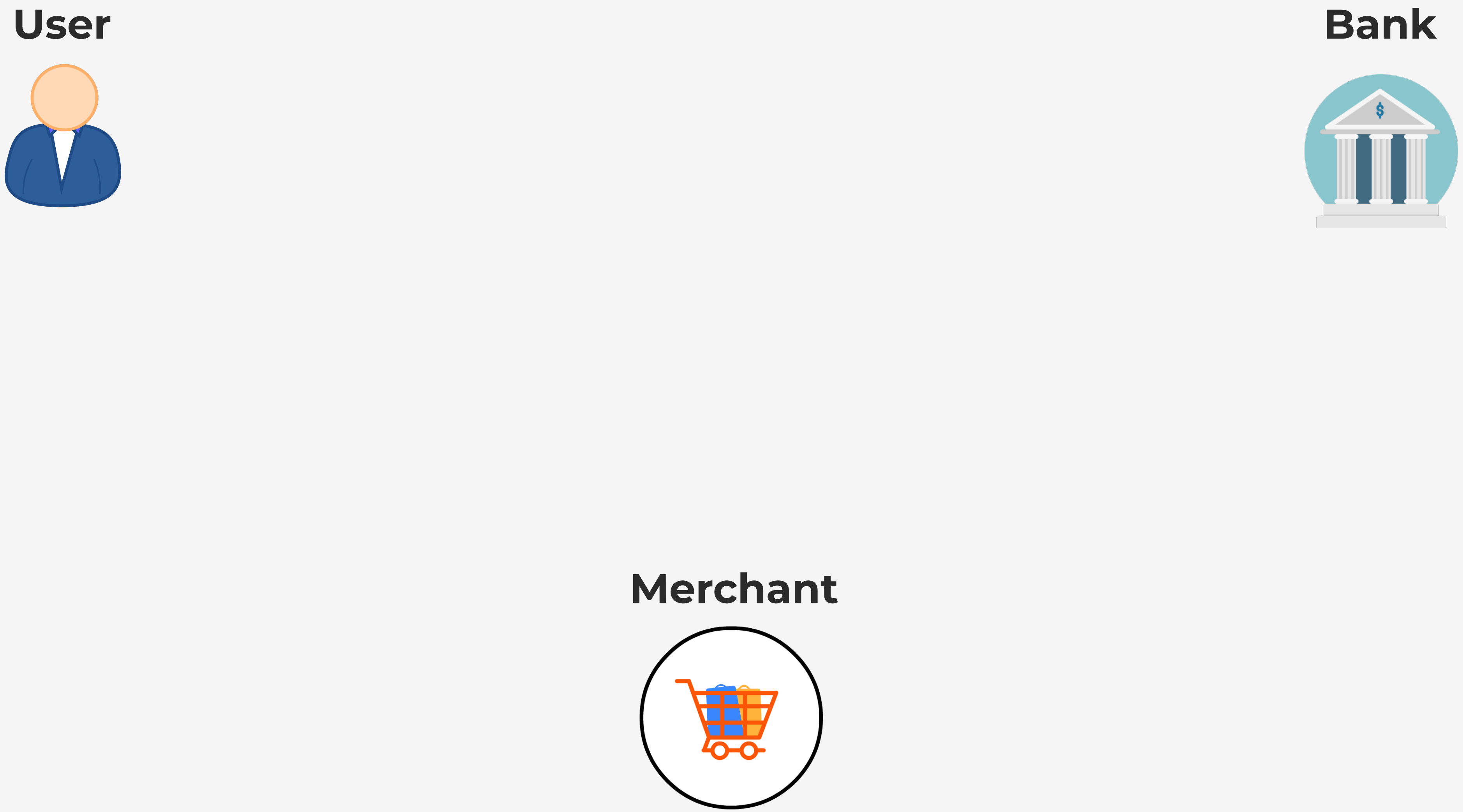
req

pre ← Issue(req, sk)

pre

sig ← Obtain(pre, St, pk)

Chaum’s E-cash

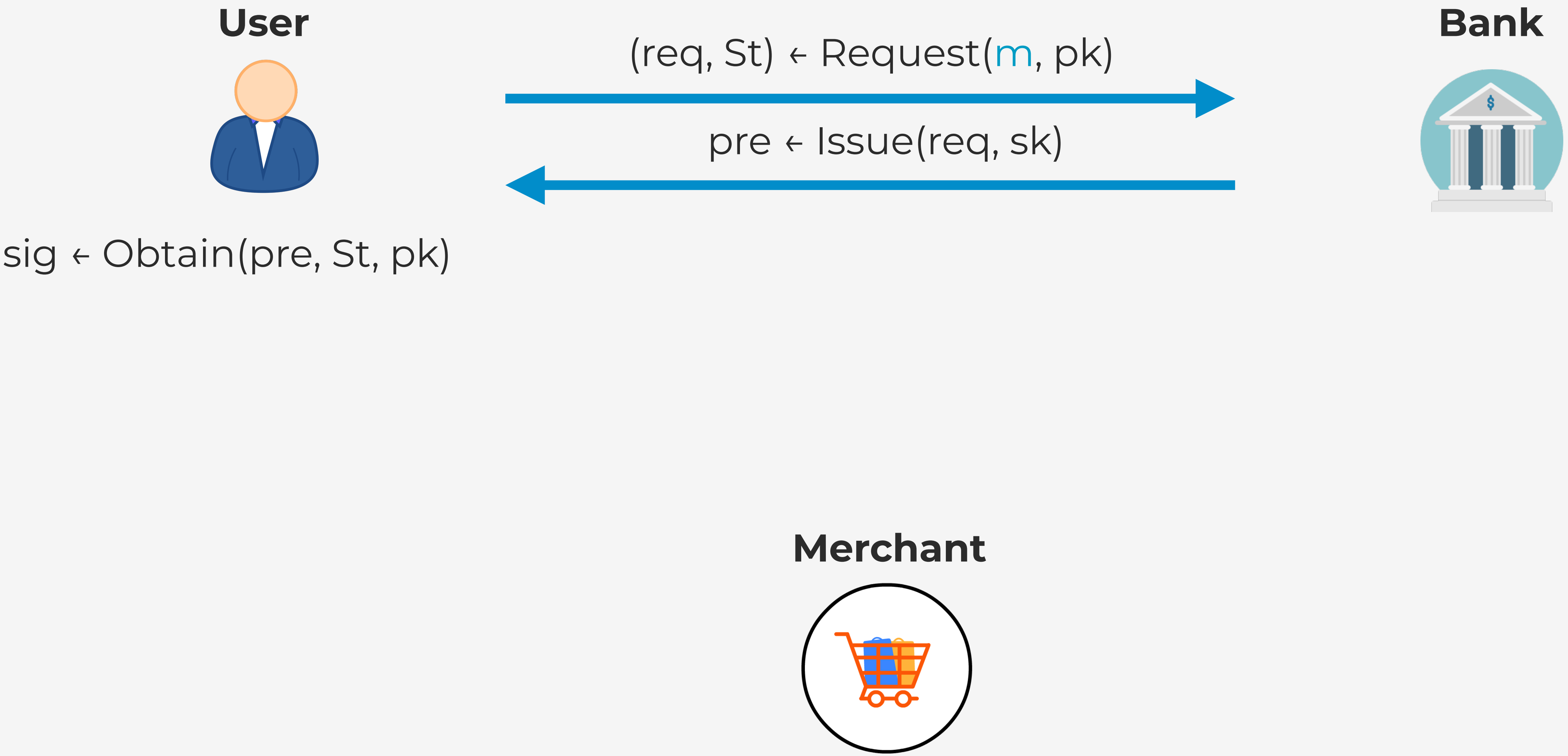


User

Bank

Merchant

Chaum’s E-cash



User

Bank

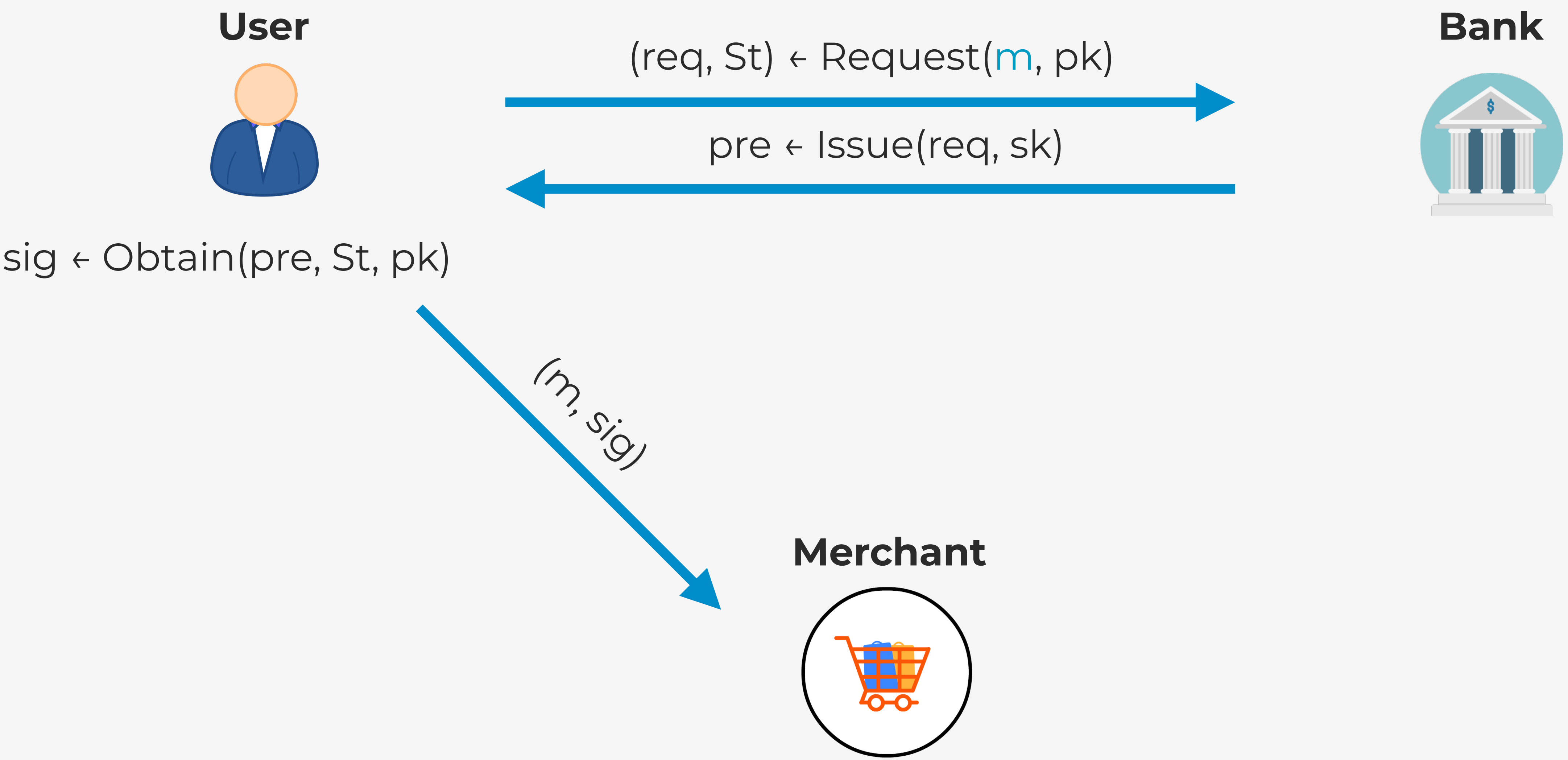
(req, St) ← Request(m, pk)

pre ← Issue(req, sk)

sig ← Obtain(pre, St, pk)

Merchant

Chaum’s E-cash



User

Bank

(req, St) ← Request(m, pk)

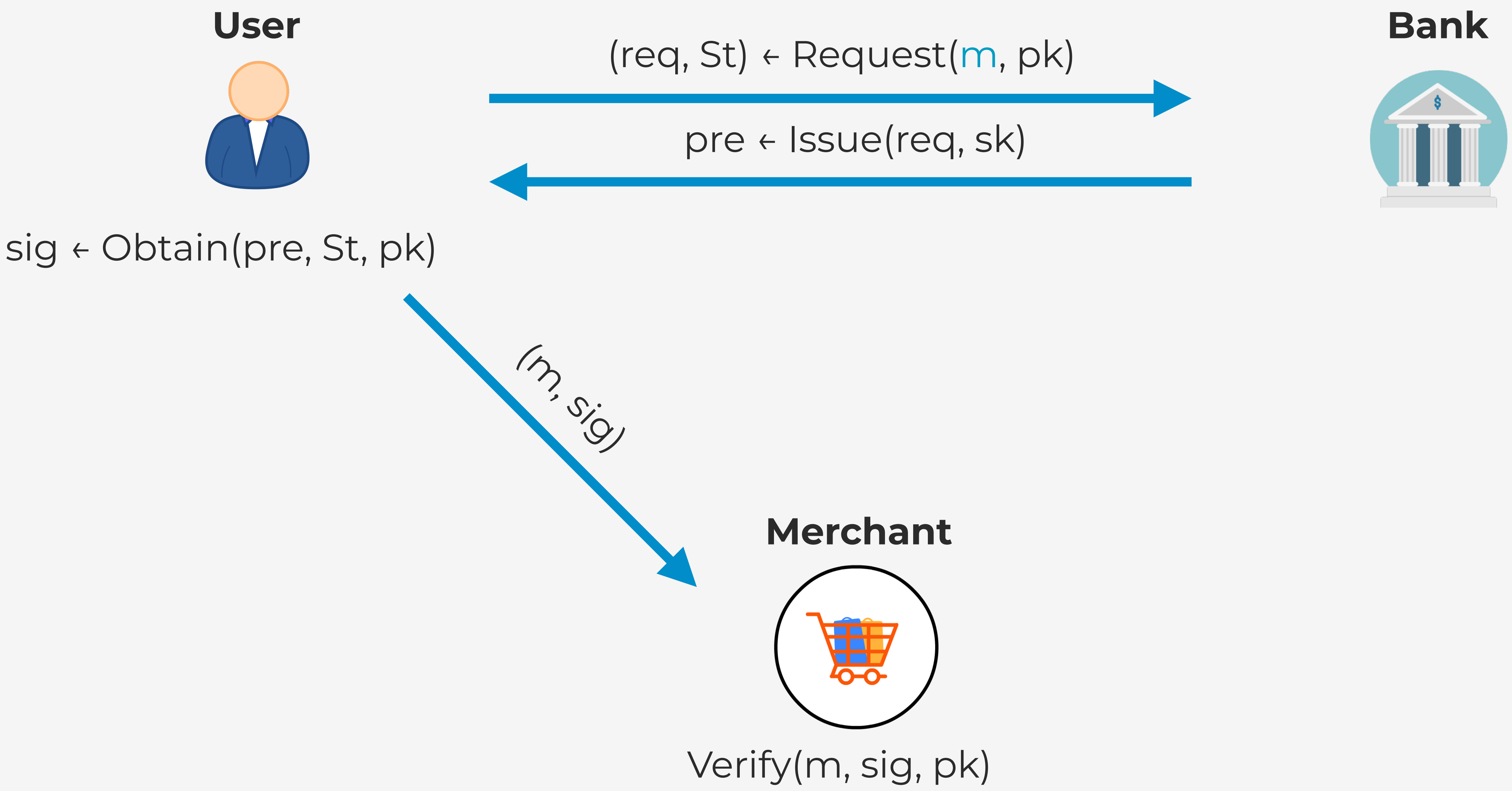
pre ← Issue(req, sk)

sig ← Obtain(pre, St, pk)

(m, sig)

Merchant

Chaum’s E-cash



User

Bank

(req, St) ← Request(m, pk)

pre ← Issue(req, sk)

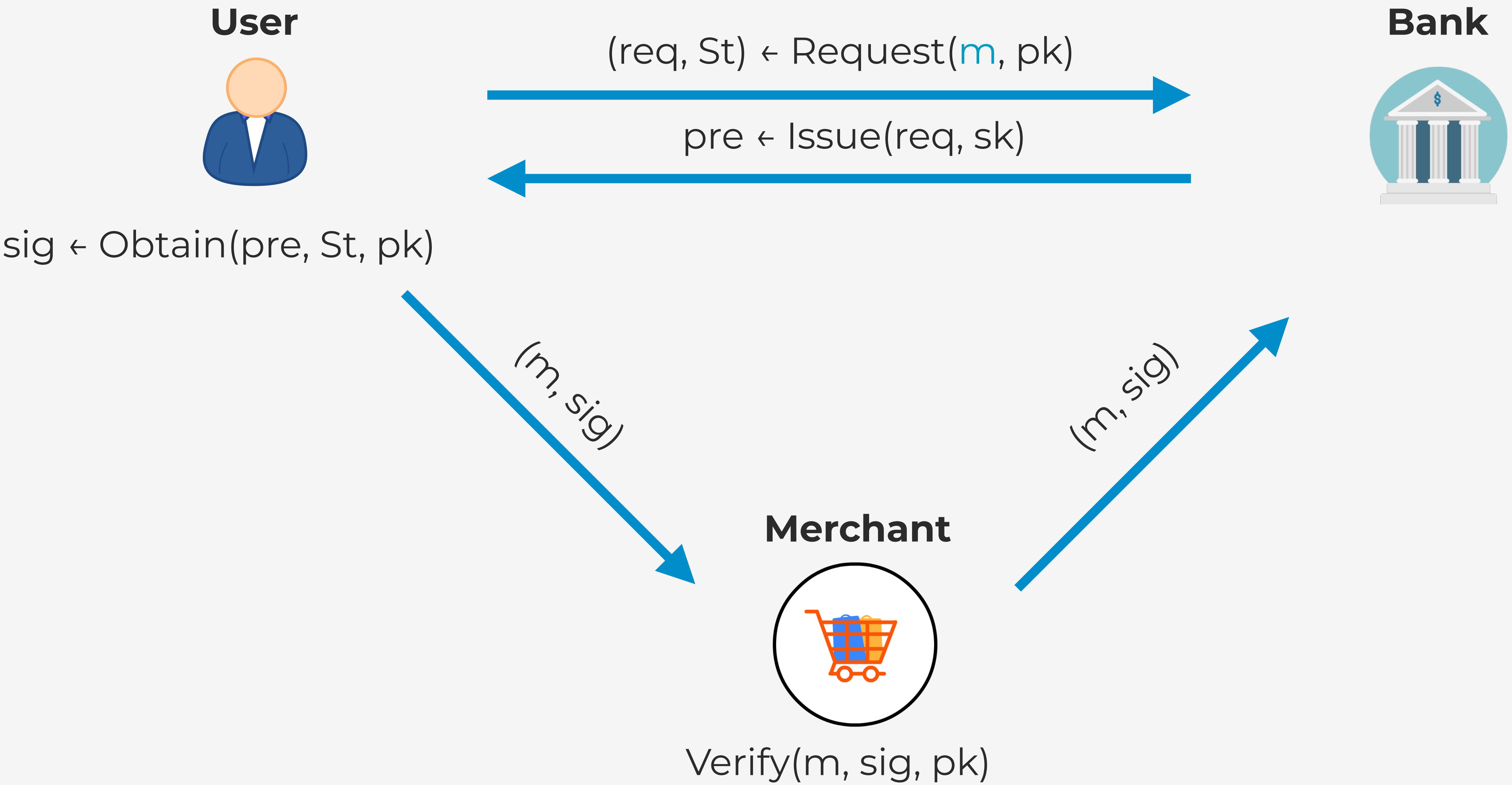
sig ← Obtain(pre, St, pk)

(m, sig)

Merchant

Verify(m, sig, pk)

Chaum’s E-cash



User

Bank

(req, St) ← Request(m, pk)

pre ← Issue(req, sk)

sig ← Obtain(pre, St, pk)

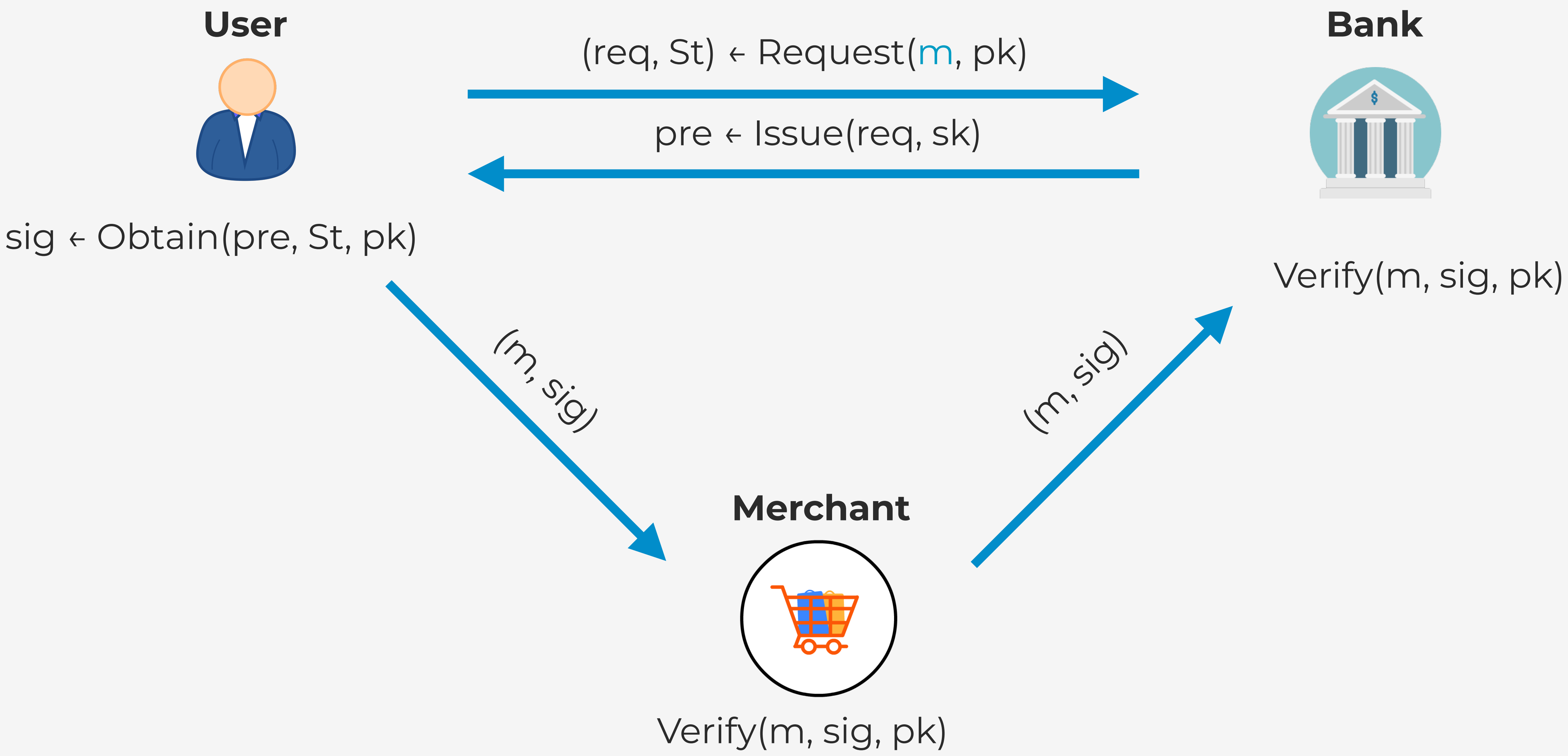
(m, sig)

(m, sig)

Merchant

Verify(m, sig, pk)

Chaum’s E-cash



User

Bank

(req, St) ← Request(m, pk)

pre ← Issue(req, sk)

sig ← Obtain(pre, St, pk)

Verify(m, sig, pk)

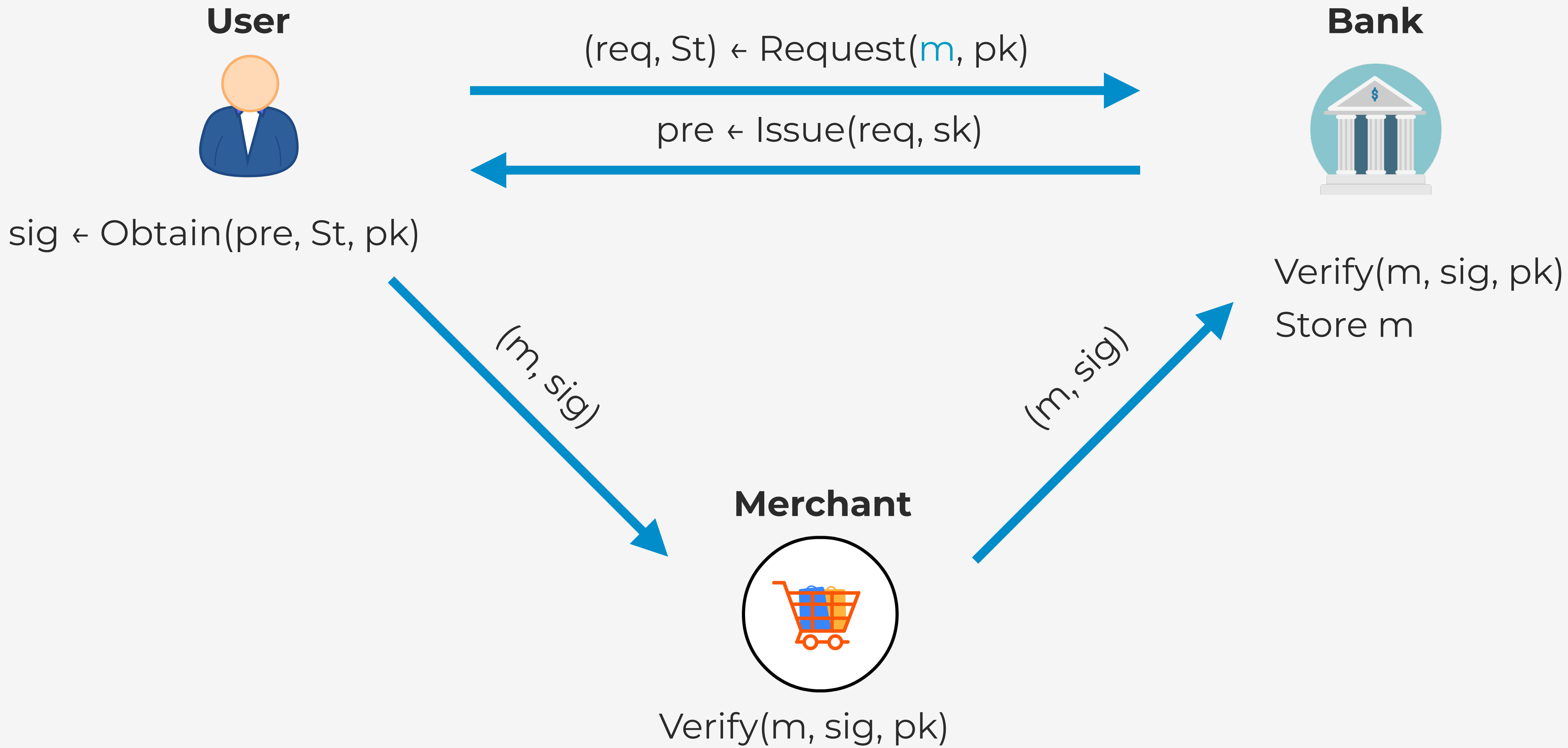
(m, sig)

(m, sig)

Merchant

Verify(m, sig, pk)

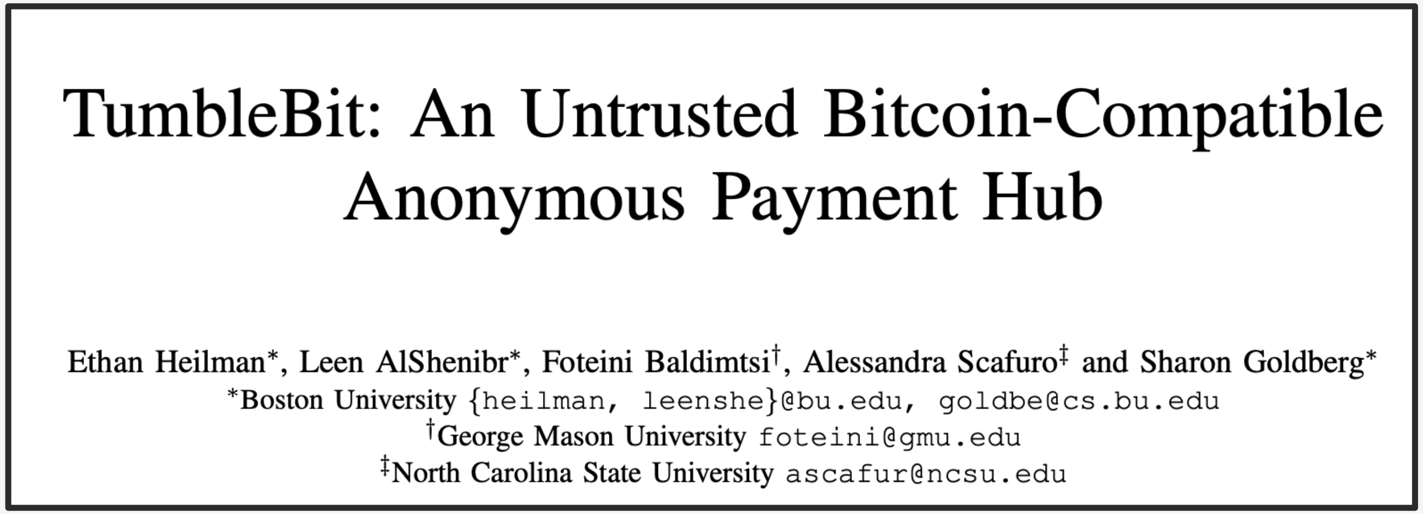
Chaum’s E-cash



Chaum’s E-cash - other scenarios

Chaum’s E-cash - other scenarios

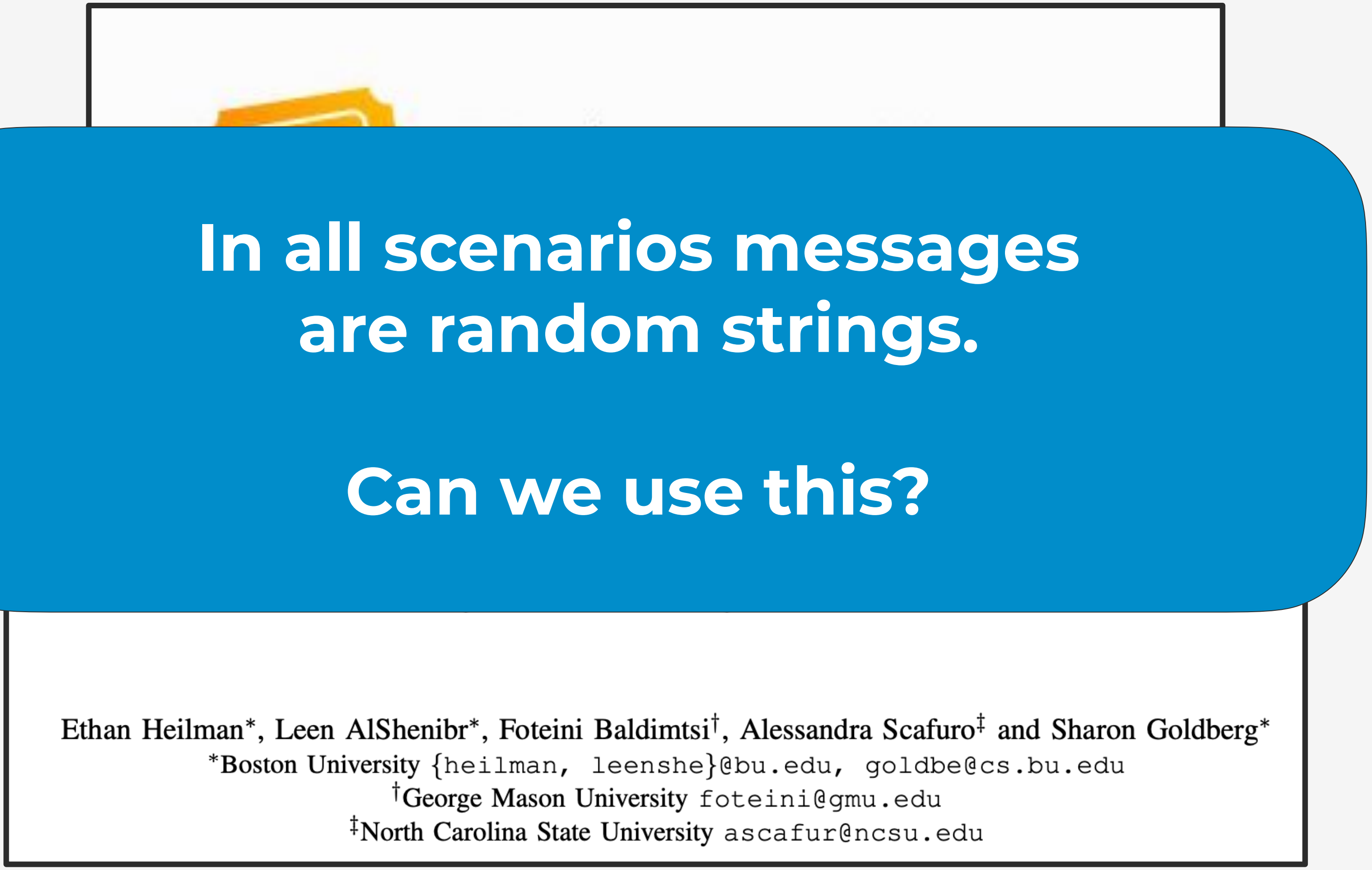
Chaum’s E-cash - other scenarios



TumbleBit: An Untrusted Bitcoin-Compatible Anonymous Payment Hub

Ethan Heilman[*], Leen AlShenibr[*], Foteini Baldimtsi[†], Alessandra Scafuro[‡] and Sharon Goldberg[*]
[*]Boston University {heilman, leenshe}@bu.edu, goldbe@cs.bu.edu
[†]George Mason University foteini@gmu.edu
[‡]North Carolina State University ascafur@ncsu.edu

Chaum’s E-cash - other scenarios

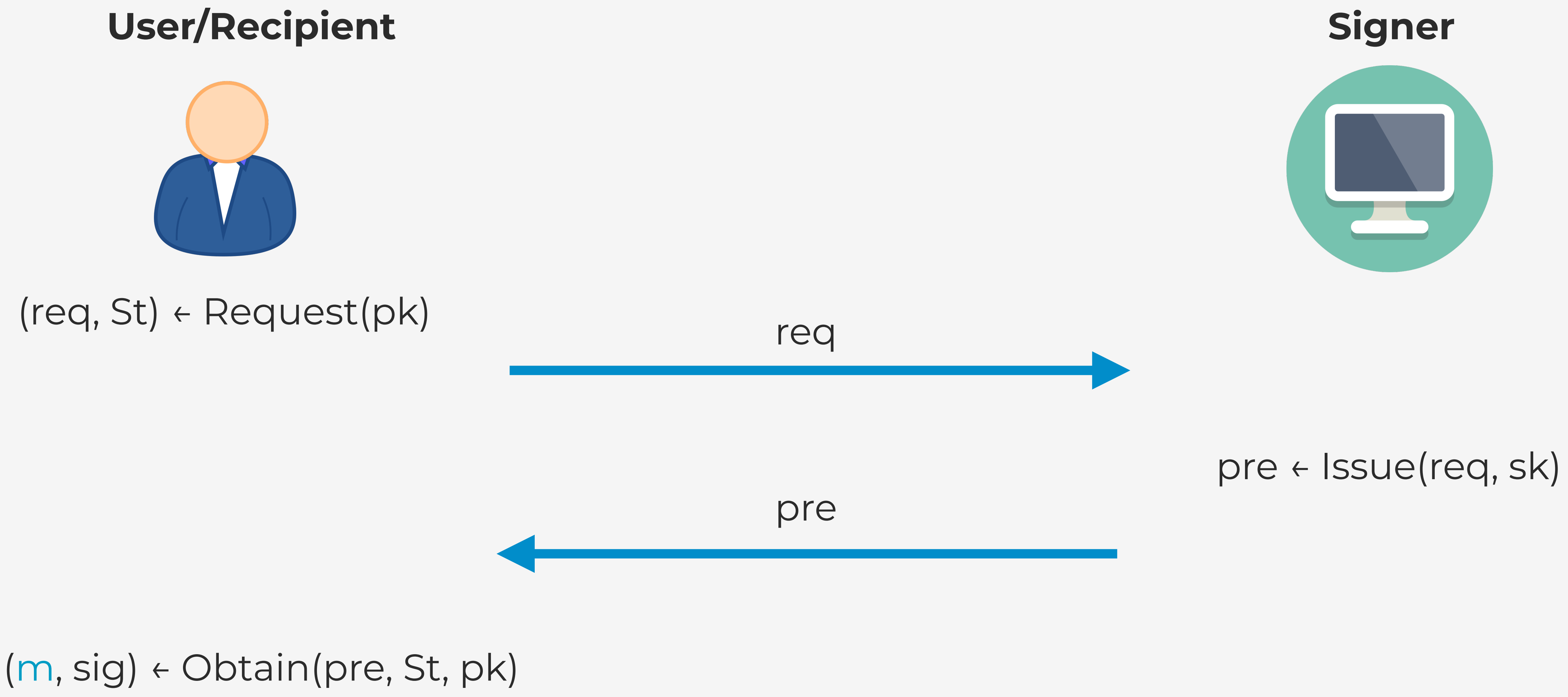


Ethan Heilman*, Leen AlShenibr*, Foteini Baldimtsi†, Alessandra Scafuro‡ and Sharon Goldberg*
*Boston University {heilman, leenshe}@bu.edu, goldbe@cs.bu.edu
†George Mason University foteini@gmu.edu
‡North Carolina State University ascafur@ncsu.edu

In all scenarios messages are random strings.

Can we use this?

Two-move Blind Signatures for Random Messages



User/Recipient

Signer

(req, St) ← Request(pk)

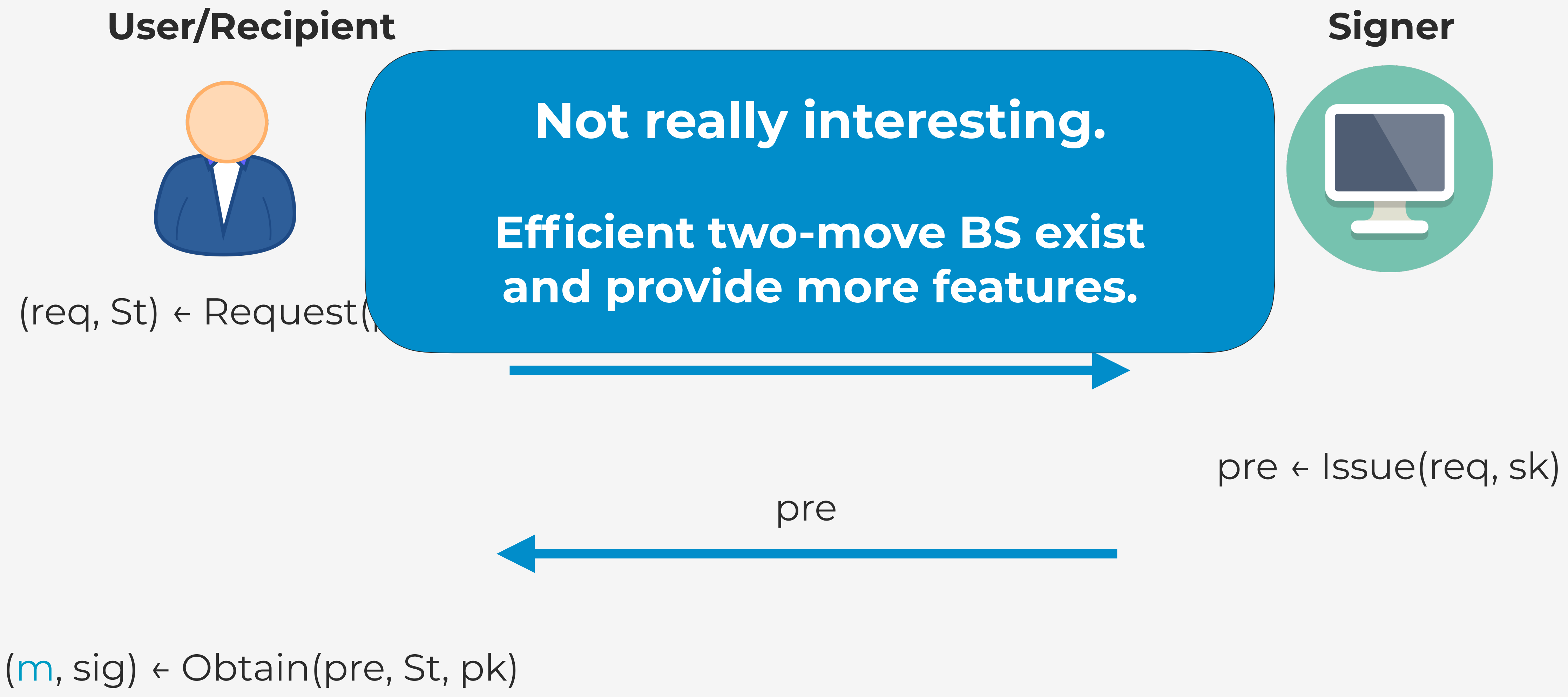
req

pre ← Issue(req, sk)

pre

(m, sig) ← Obtain(pre, St, pk)

Two-move Blind Signatures for Random Messages



User/Recipient

Signer

(req, St) ← Request(

Not really interesting.

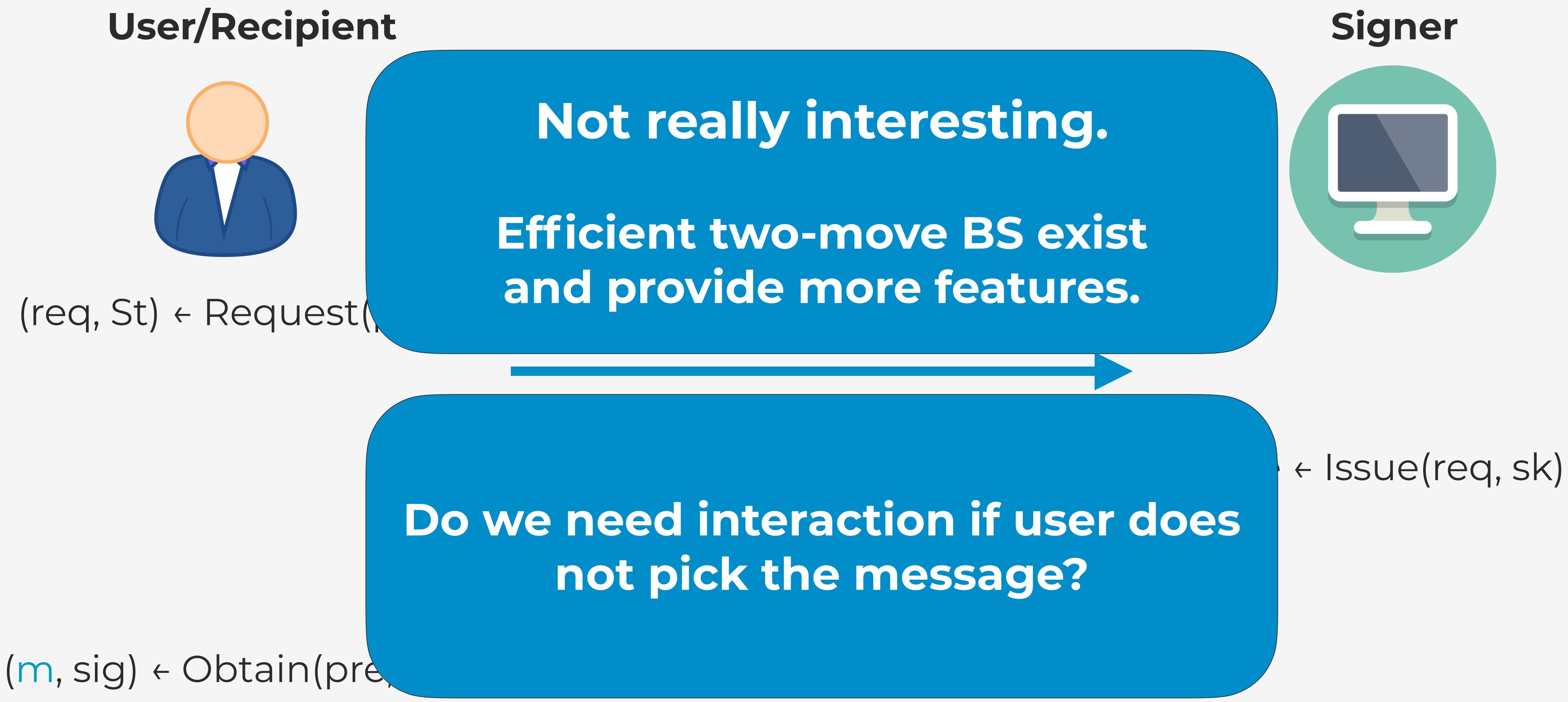
Efficient two-move BS exist and provide more features.

pre ← Issue(req, sk)

pre

(m, sig) ← Obtain(pre, St, pk)

Two-move Blind Signatures for Random Messages



User/Recipient

Signer

(req, St) ← Request(

← Issue(req, sk)

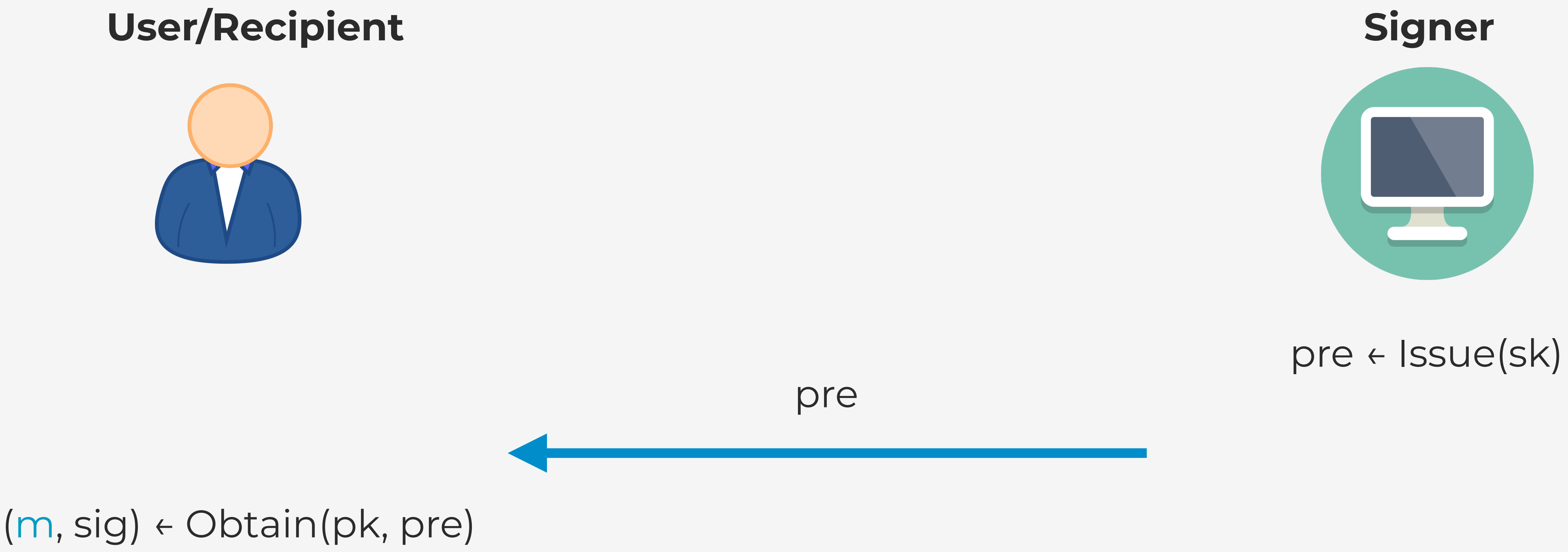
(m, sig) ← Obtain(pre

Not really interesting.

Efficient two-move BS exist and provide more features.

Do we need interaction if user does not pick the message?

Strawman Solution #1



User/Recipient

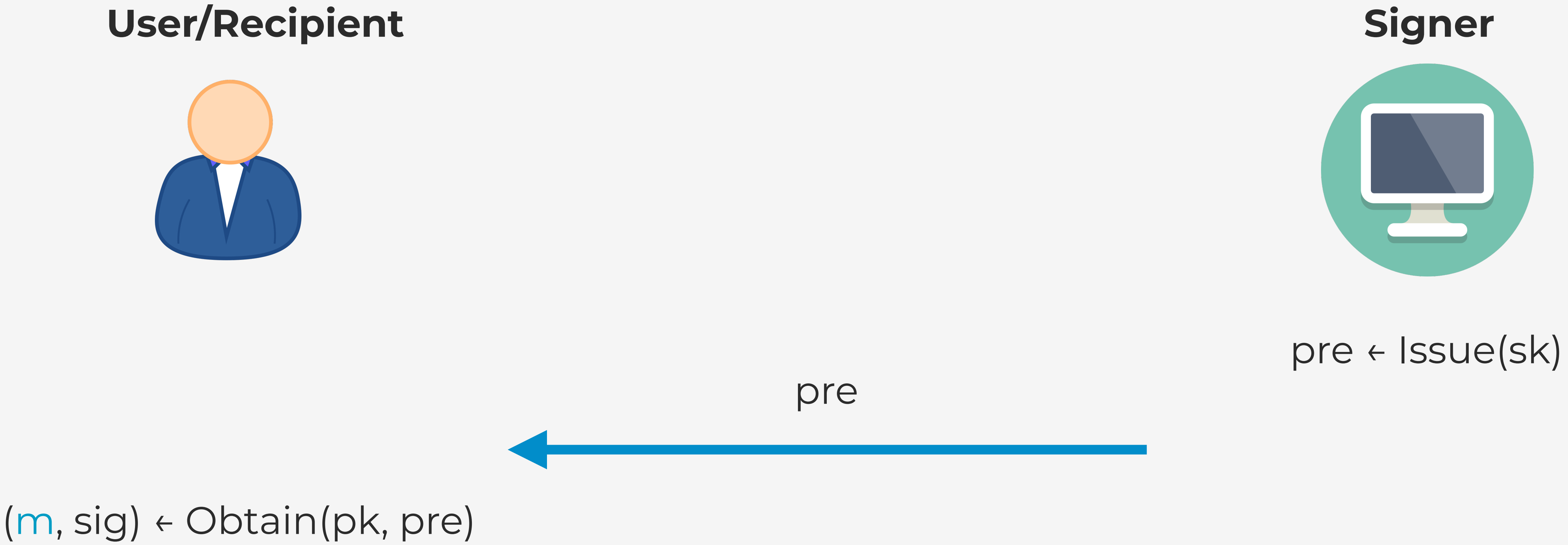
Signer

pre ← Issue(sk)

pre

(m, sig) ← Obtain(pk, pre)

Strawman Solution #1



User/Recipient

Signer

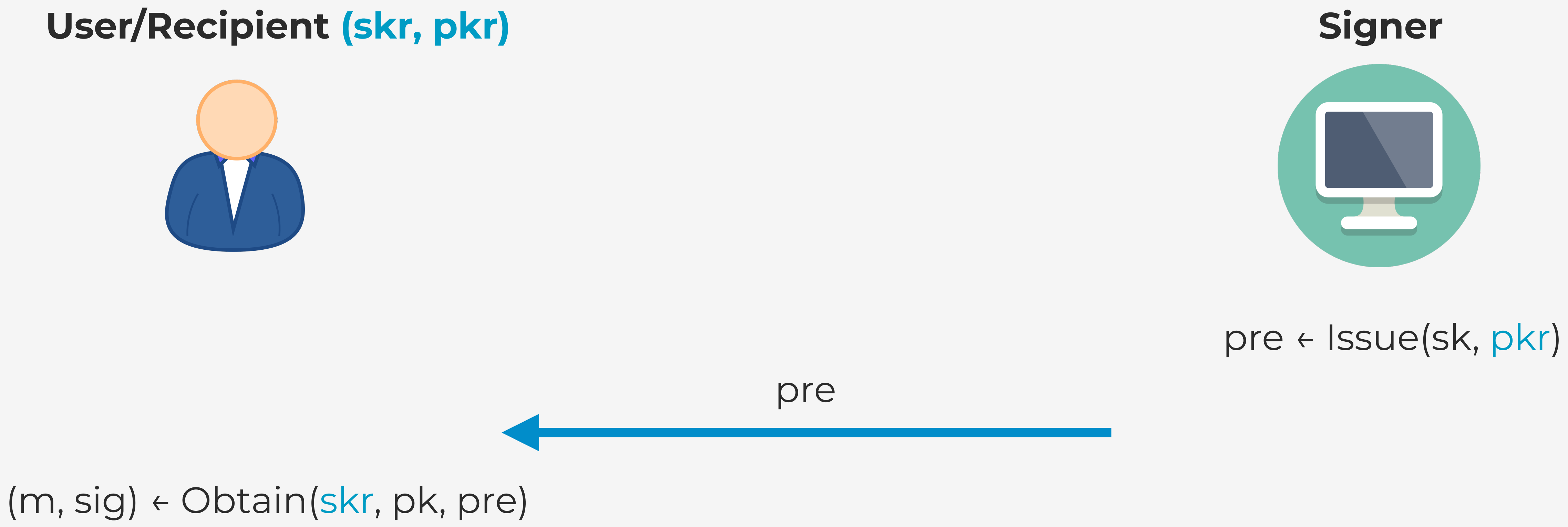
pre ← Issue(sk)

pre

(m, sig) ← Obtain(pk, pre)

User can unblind presignature many times!

Strawman Solution #2



User/Recipient (skr, pkr)

Signer

pre ← Issue(sk, pkr)

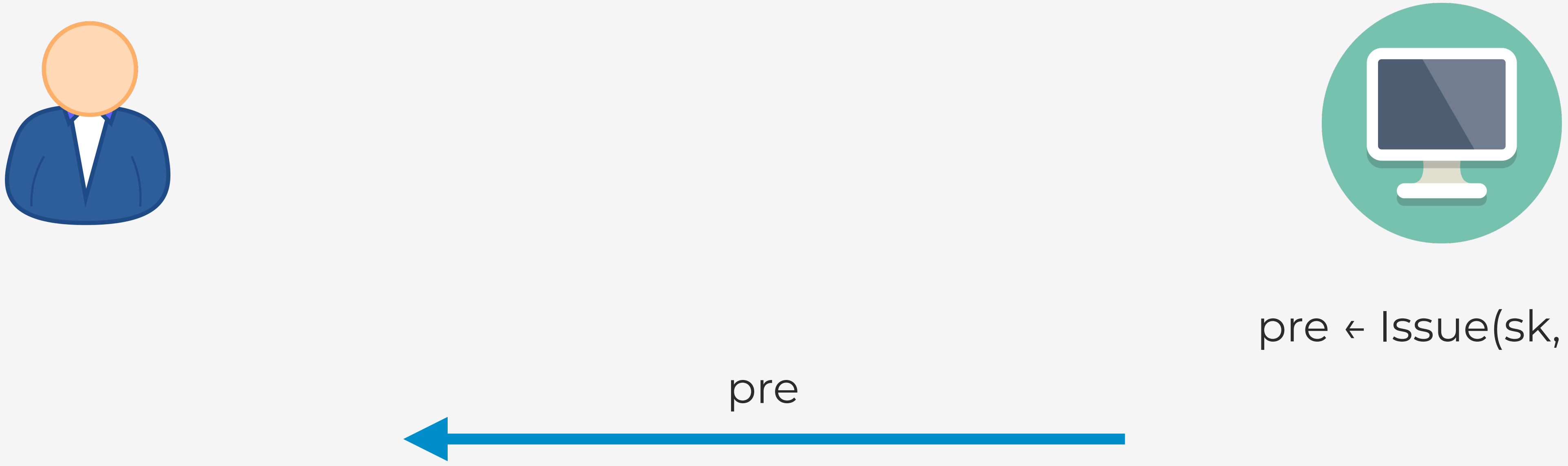
pre

(m, sig) ← Obtain(skr, pk, pre)

Strawman Solution #2

User/Recipient (skr, pkr)

Signer



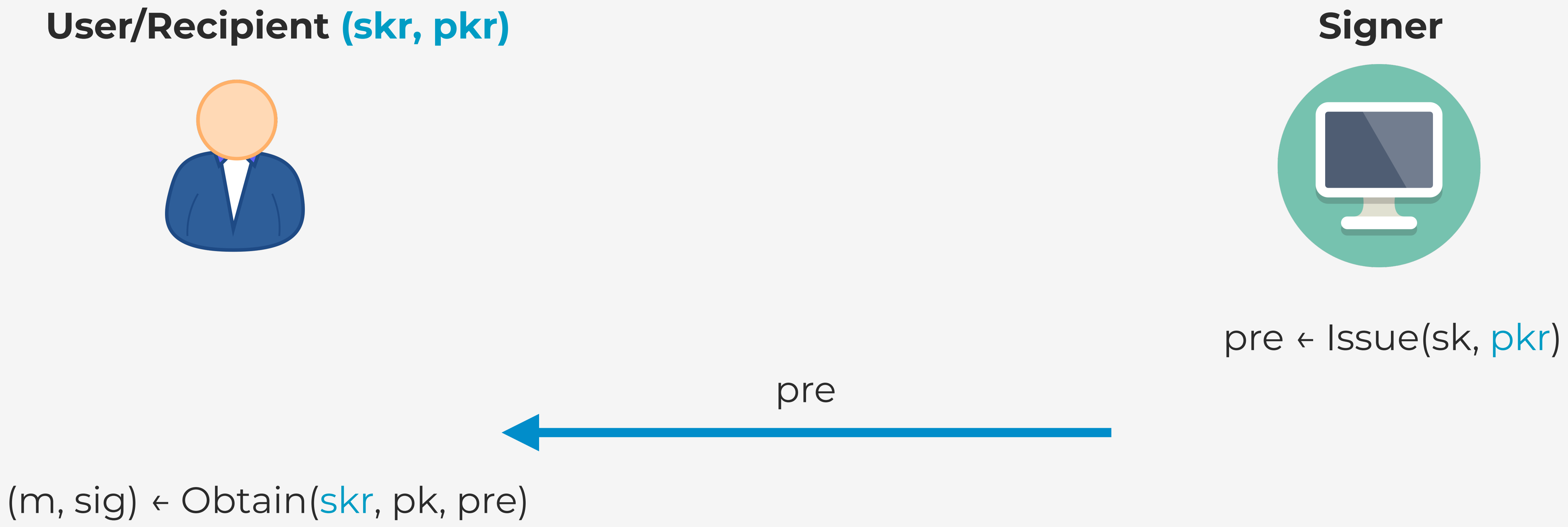
pre ← Issue(sk, pkr)

pre

(m, sig) ← Obtain(skr, pk, pre)

Message m is now a function of skr

Strawman Solution #2



User/Recipient **(skr, pkr)**

Signer

pre ← Issue(sk, pkr)

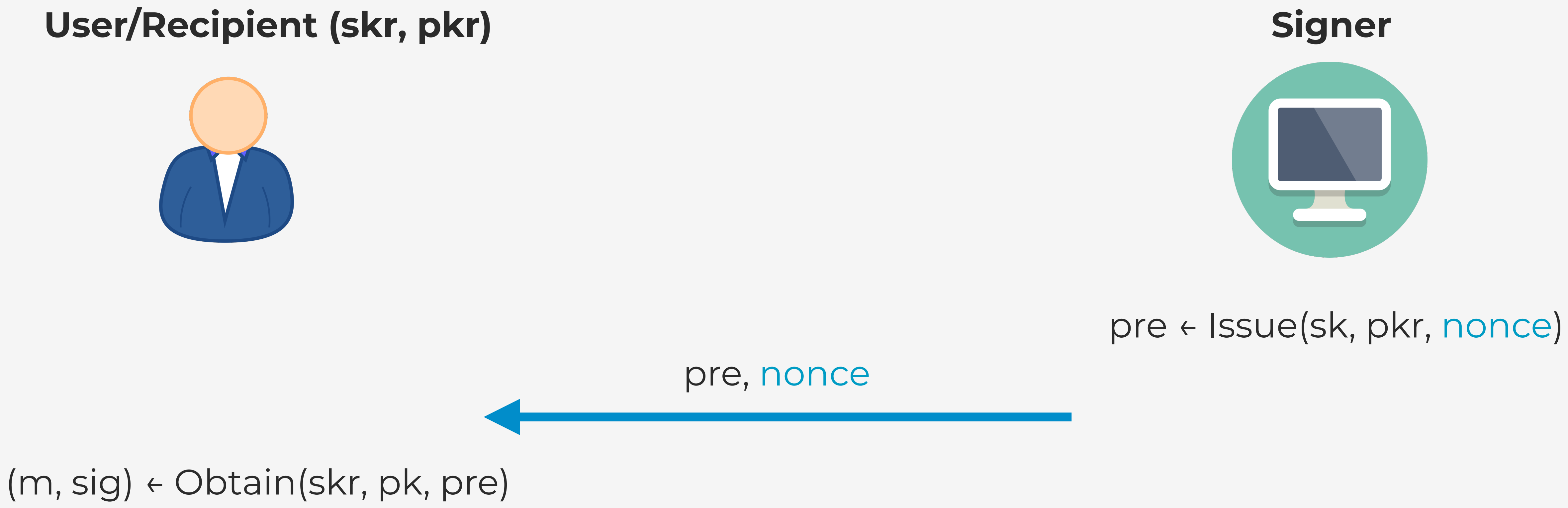
pre

(m, sig) ← Obtain(skr, pk, pre)

Message m is now a function of skr

Only one presignature per pkr

Solution #3



User/Recipient (skr, pkr)

Signer

pre ← Issue(sk, pkr, nonce)

pre, nonce

(m, sig) ← Obtain(skr, pk, pre)

Non-inteactive Blind Signatures (NIBS)

KeyGen(secpar)
outputs signer’s key pair (sk,pk)

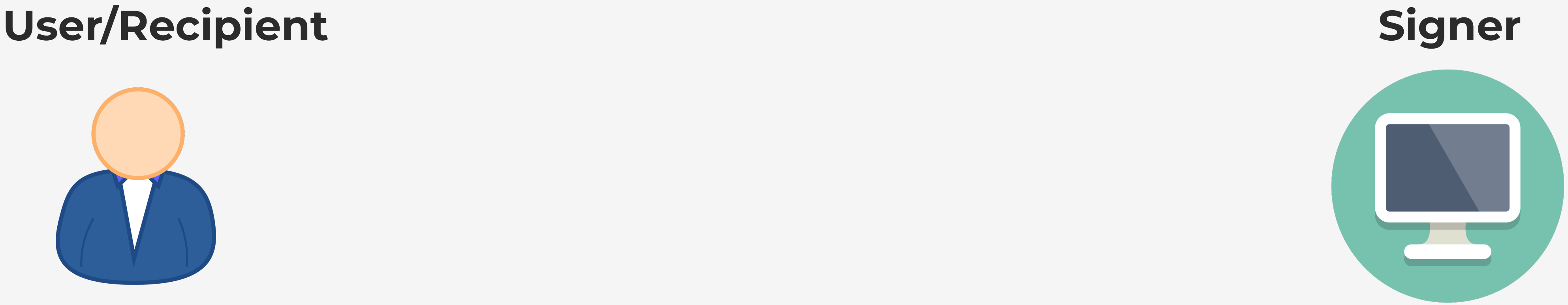
RKeyGen(secpar)
outputs recipient's key pair (skr,pkr)

Issue(sk,pkr,nonce)
outputs presignature (pre)

Obtain(skr,pk,pre,nonce)
outputs message-signature pair (m, sig)

Verify(pk, (m,sig))
outputs validity of message-signature pair

How to use NIBS?



User/Recipient

Signer

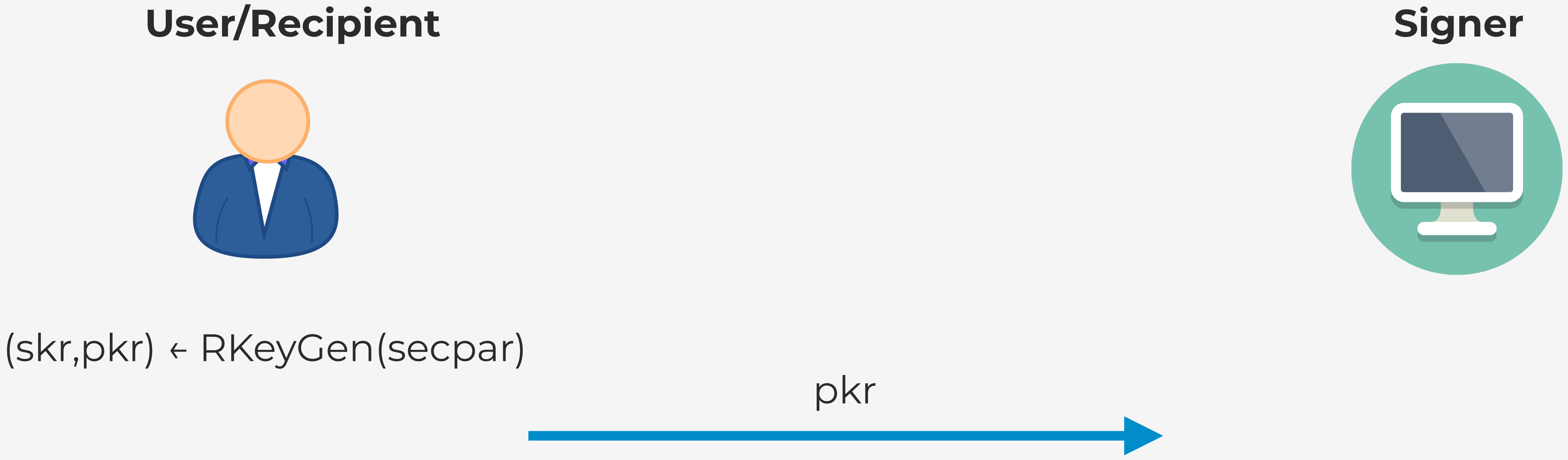
How to use NIBS?

User/Recipient

Signer

(skr,pkr) ← RKeyGen(secpar)

How to use NIBS?



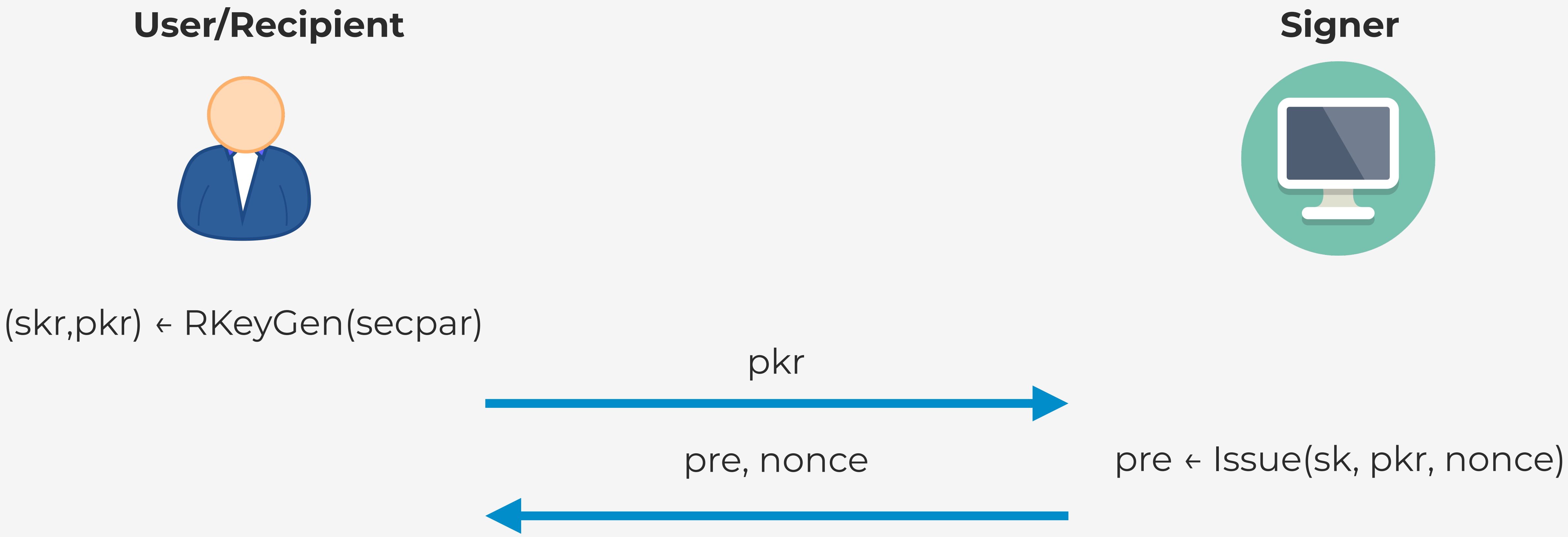
User/Recipient

Signer

(skr,pkr) ← RKeyGen(secpar)

pkr

How to use NIBS?



User/Recipient

Signer

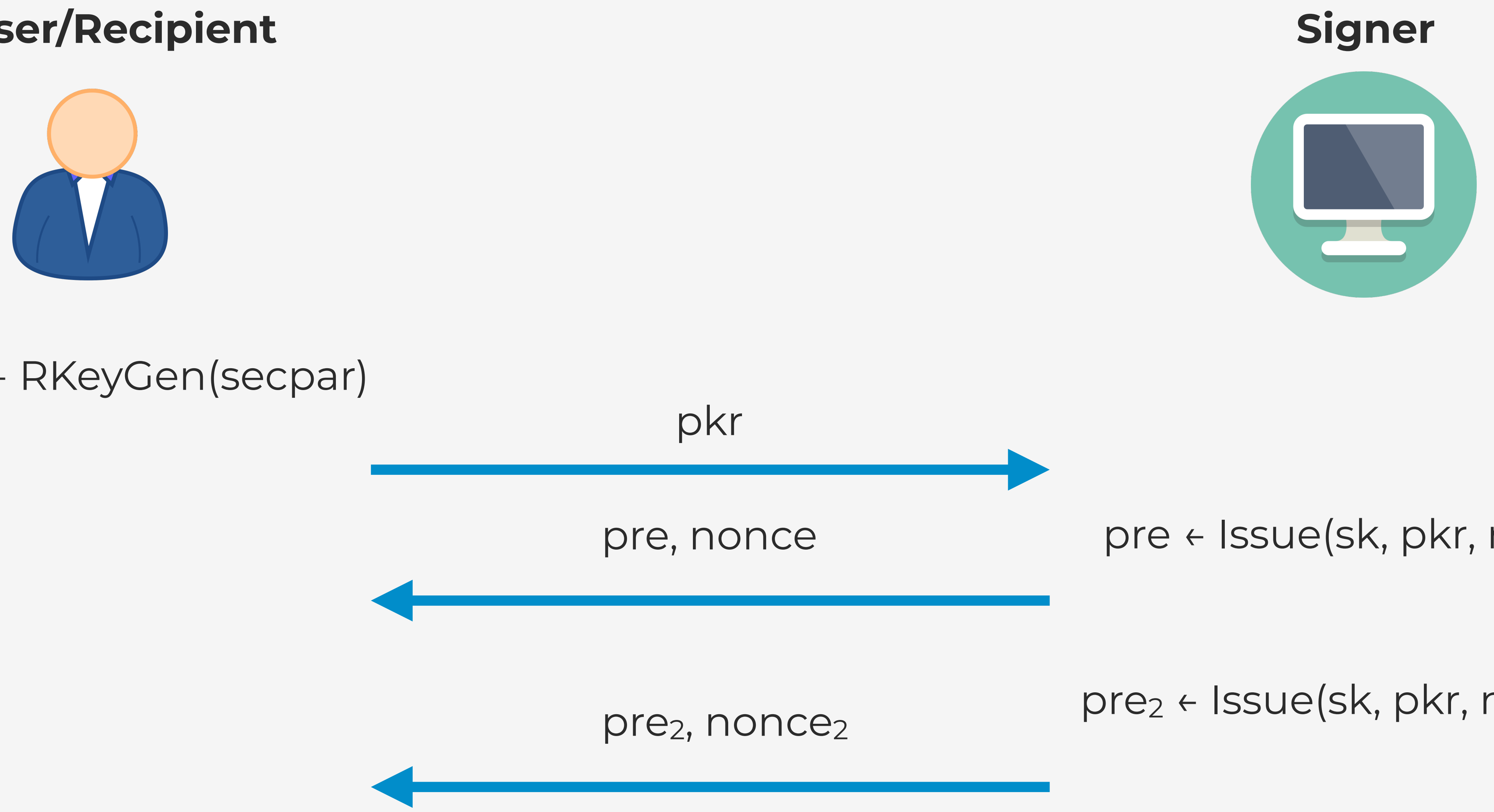
(skr,pkr) ← RKeyGen(secpar)

pkr

pre, nonce

pre ← Issue(sk, pkr, nonce)

How to use NIBS?



User/Recipient

Signer

(skr,pkr) ← RKeyGen(secpar)

pkr

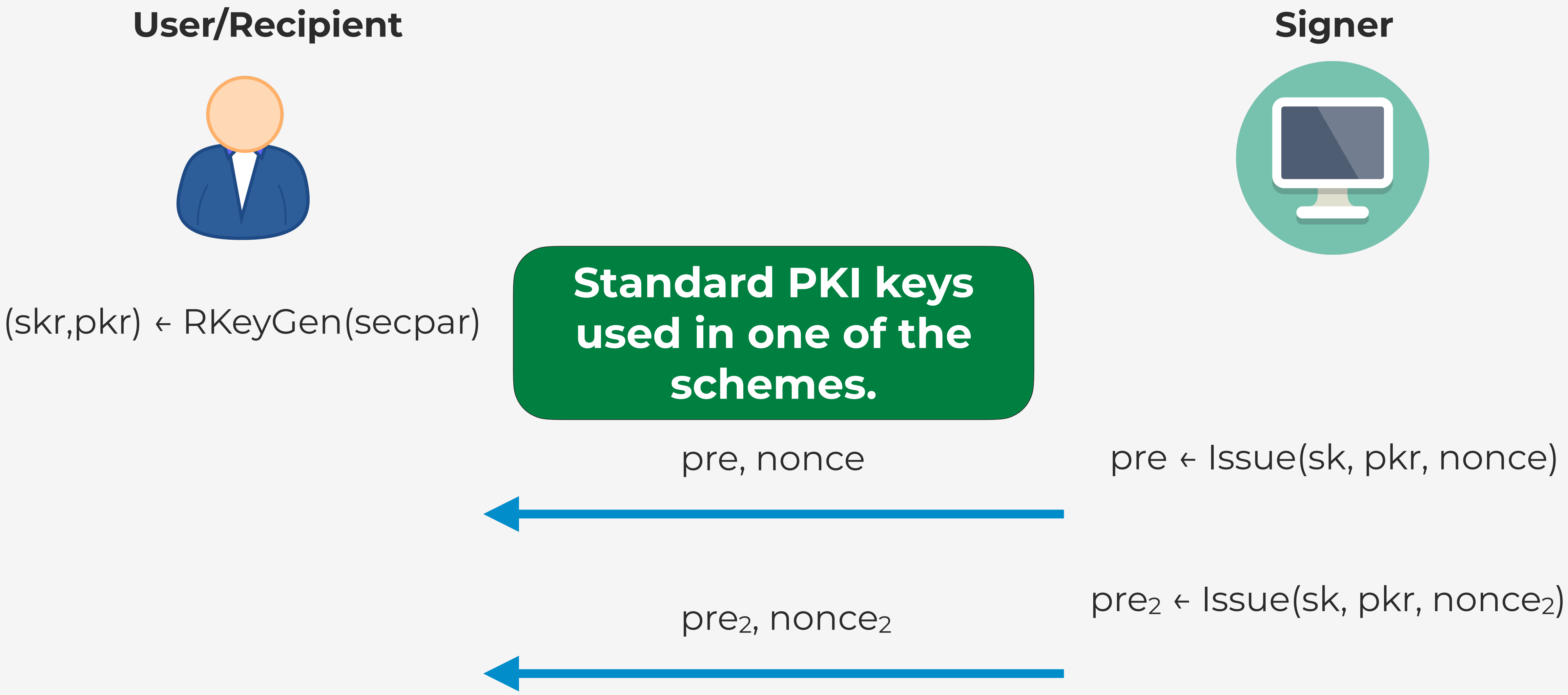
pre, nonce

pre ← Issue(sk, pkr, nonce)

pre$_2$, nonce$_2$

pre$_2$ ← Issue(sk, pkr, nonce$_2$)

How to use NIBS?



User/Recipient

Signer

(skr,pkr) ← RKeyGen(secpar)

Standard PKI keys used in one of the schemes.

pre ← Issue(sk, pkr, nonce)

pre, nonce

pre$_2$ ← Issue(sk, pkr, nonce$_2$)

pre$_2$, nonce$_2$

Applications

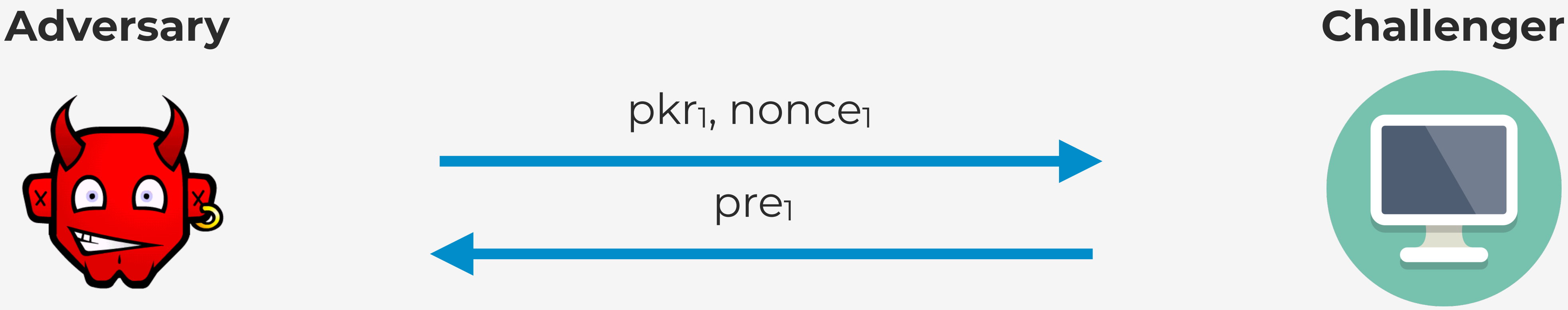
- All e-cash scenarios including Privacy Pass (Batch issuing with a single message)
- Loterry System (Final message unpredictable)
- Whistleblowing System (Using existing PKI to distribute tokens that can be later redeemed)
- Airdropping E-cash (E-cash systems can send free tokens to users)

Unforgeability for NIBS

Adversary

Challenger

Unforgeability for NIBS



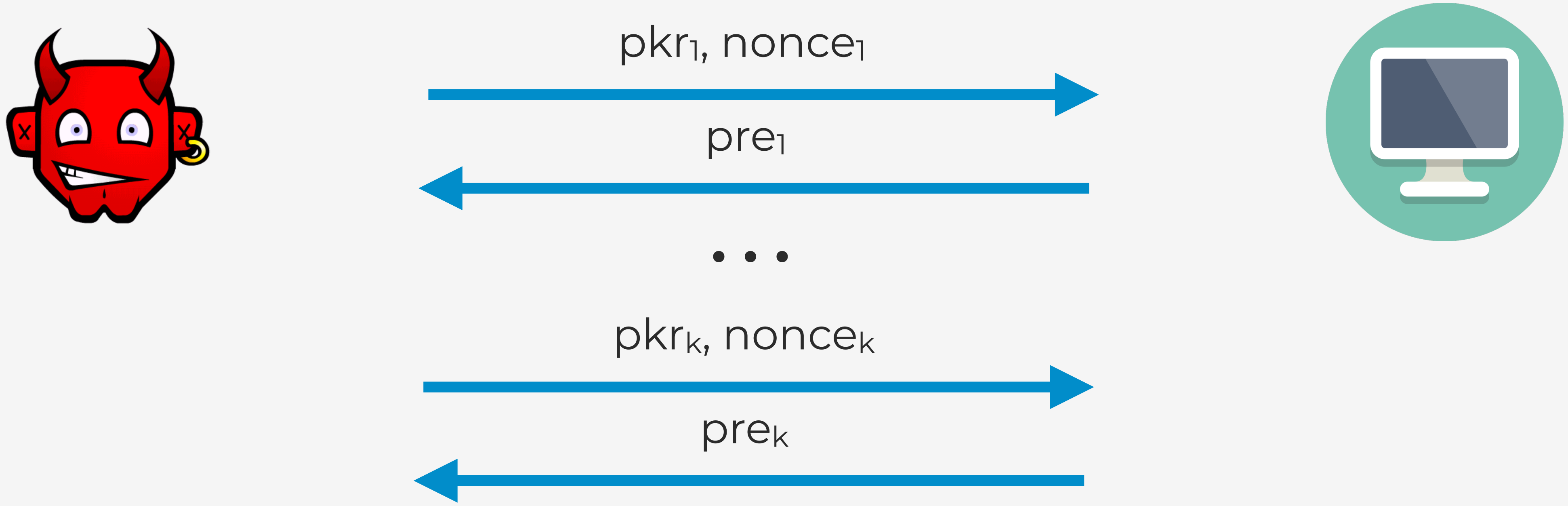
Adversary

Challenger

pkr_1, nonce_1

pre_1

Unforgeability for NIBS



Adversary

Challenger

pkr_1, $nonce_1$

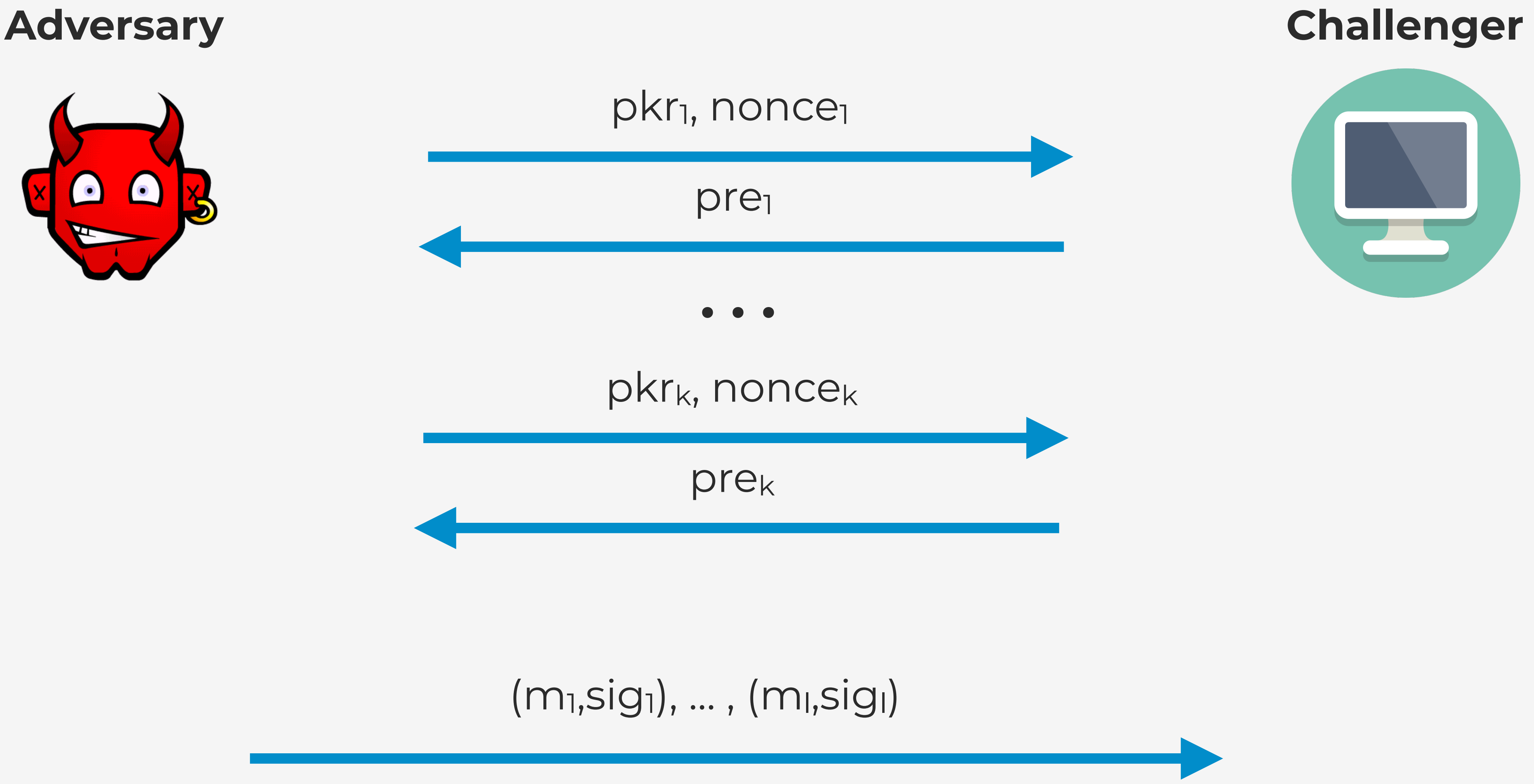
pre_1

. . .

pkr_k, $nonce_k$

pre_k

Unforgeability for NIBS



Adversary

Challenger

pkr_1, $nonce_1$

pre_1

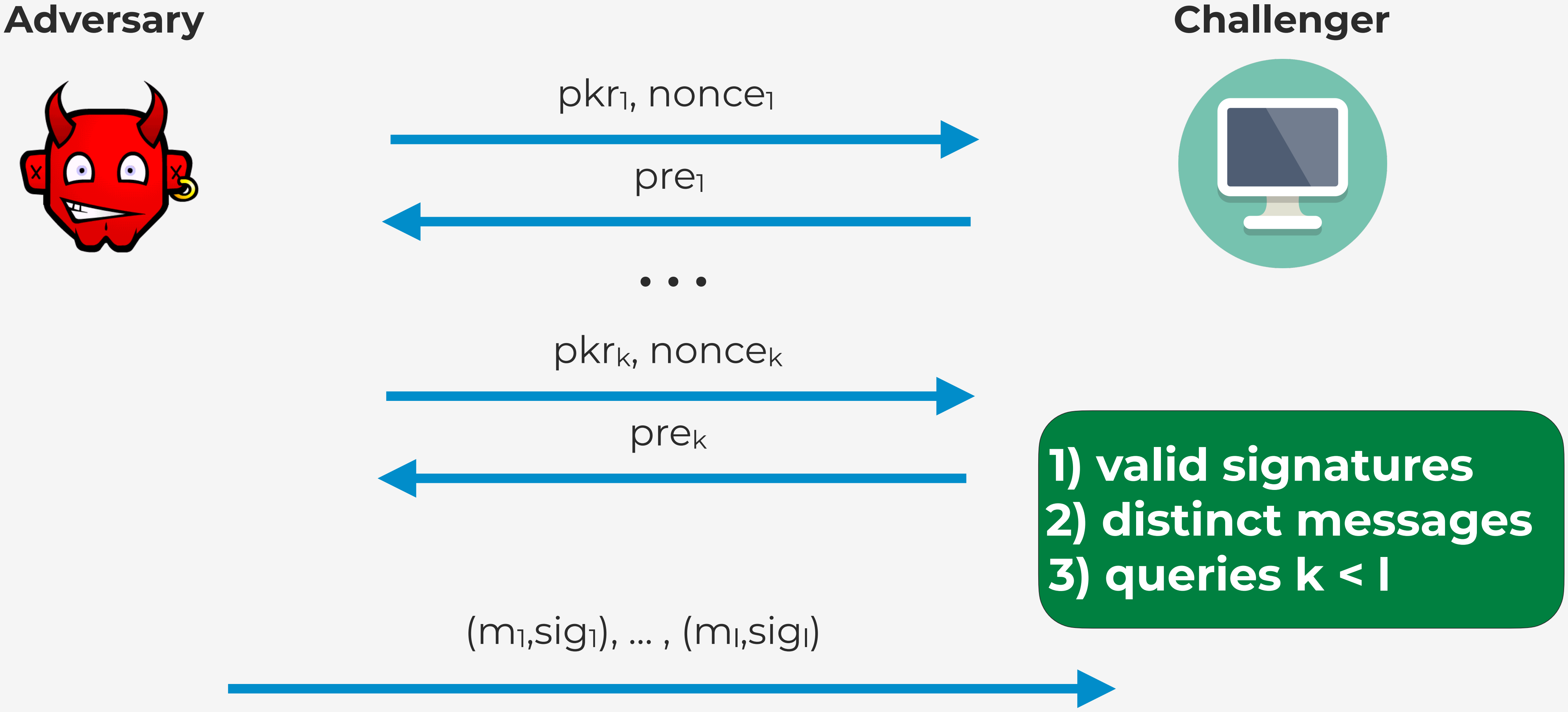
. . .

pkr_k, $nonce_k$

pre_k

(m_1,sig_1), … , (m_l,sig_l)

Unforgeability for NIBS



Adversary

Challenger

pkr_1, $nonce_1$

pre_1

. . .

pkr_k, $nonce_k$

pre_k

(m_1,sig_1), … , (m_l,sig_l)

1) valid signatures
2) distinct messages
3) queries k < l

Blindness for NIBS

Recipient Blindness

Signatures obtained by different recipient are unlinkable.

Preserves the privacy across recipients.

Recipient Blindness

Nonce Blindness

Signatures for the same recipient are unlinkable.

Allows to issue multiple presignatures without breaking blindness.

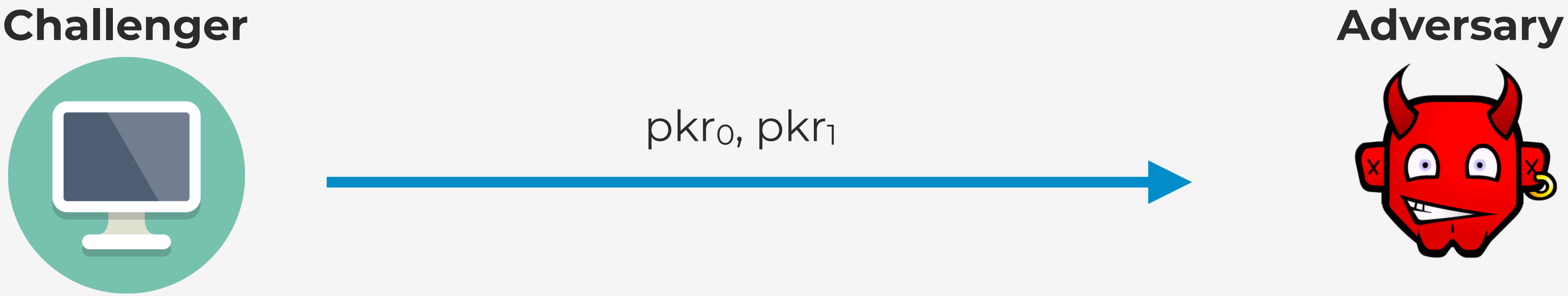
Nonce Blindness

Recipient Blindness

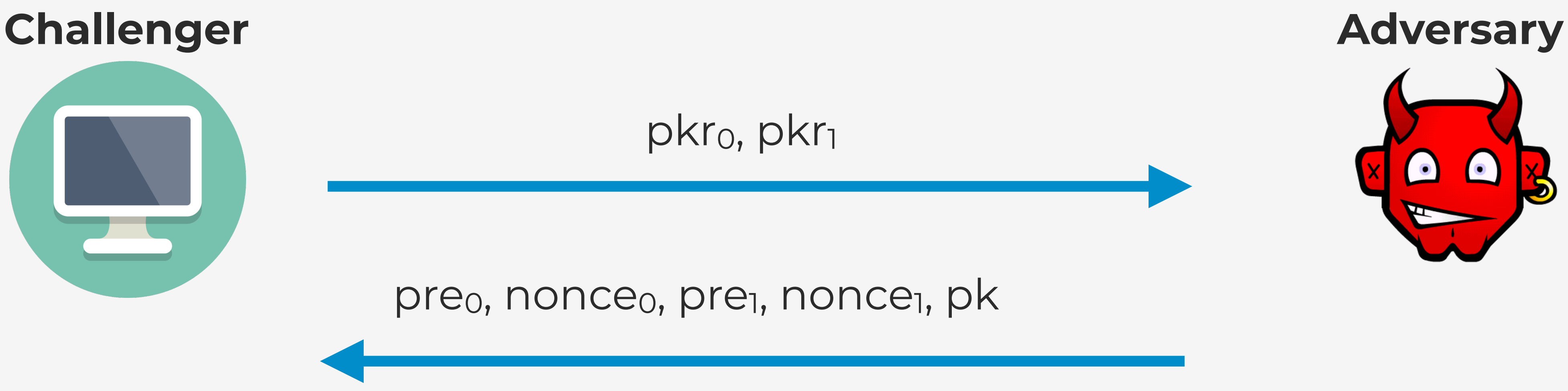
Challenger

Adversary

Recipient Blindness



Recipient Blindness



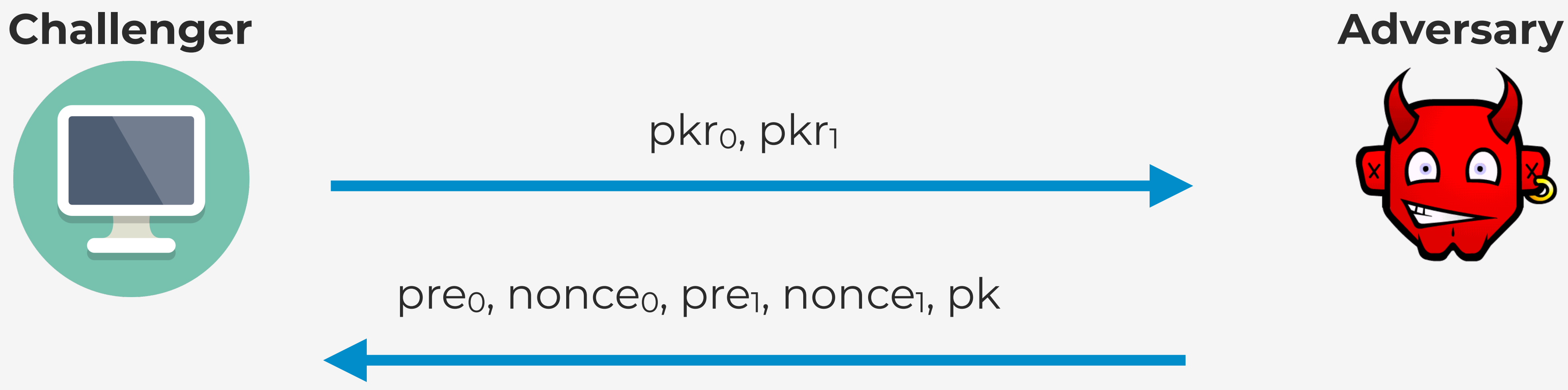
Challenger

Adversary

pkr_0, pkr_1

$pre_0, nonce_0, pre_1, nonce_1, pk$

Recipient Blindness



Challenger

Adversary

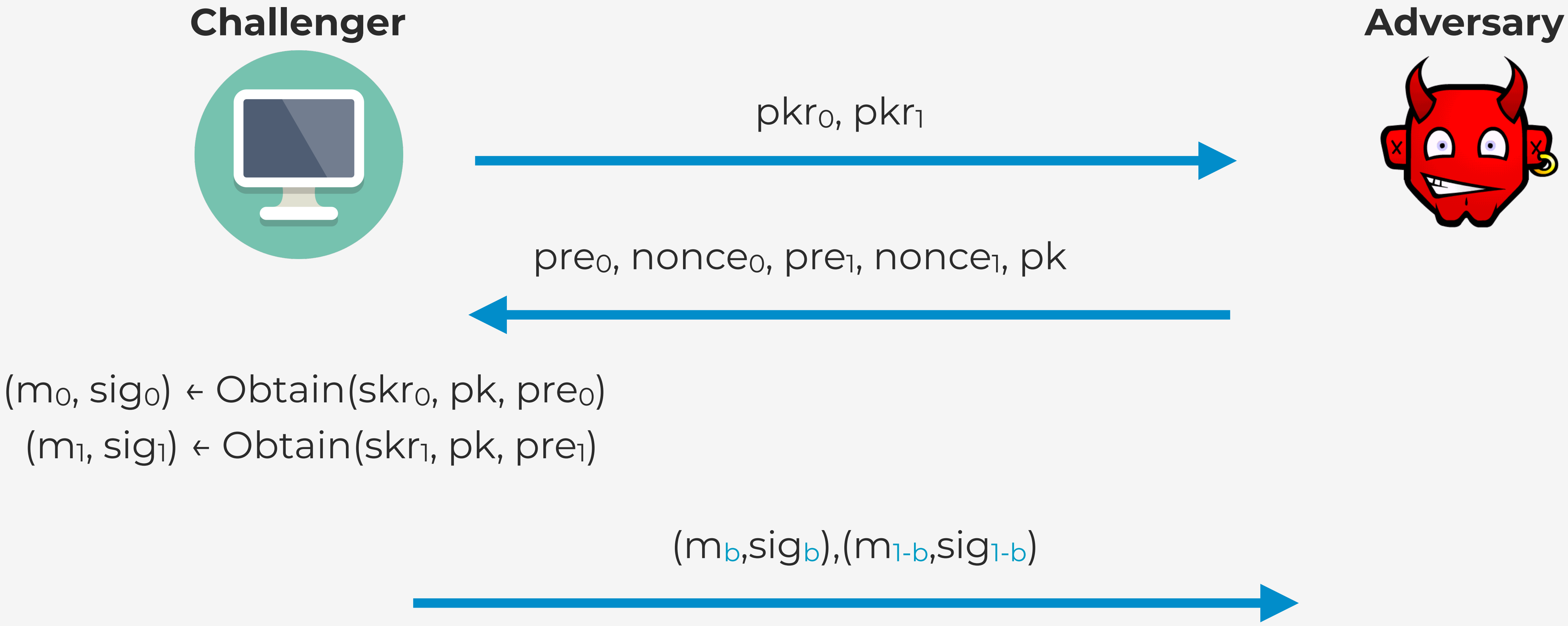
pkr_0, pkr_1

pre_0, $nonce_0$, pre_1, $nonce_1$, pk

$(m_0, sig_0) \leftarrow \text{Obtain}(skr_0, pk, pre_0)$

$(m_1, sig_1) \leftarrow \text{Obtain}(skr_1, pk, pre_1)$

Recipient Blindness



Challenger

Adversary

pkr_0, pkr_1

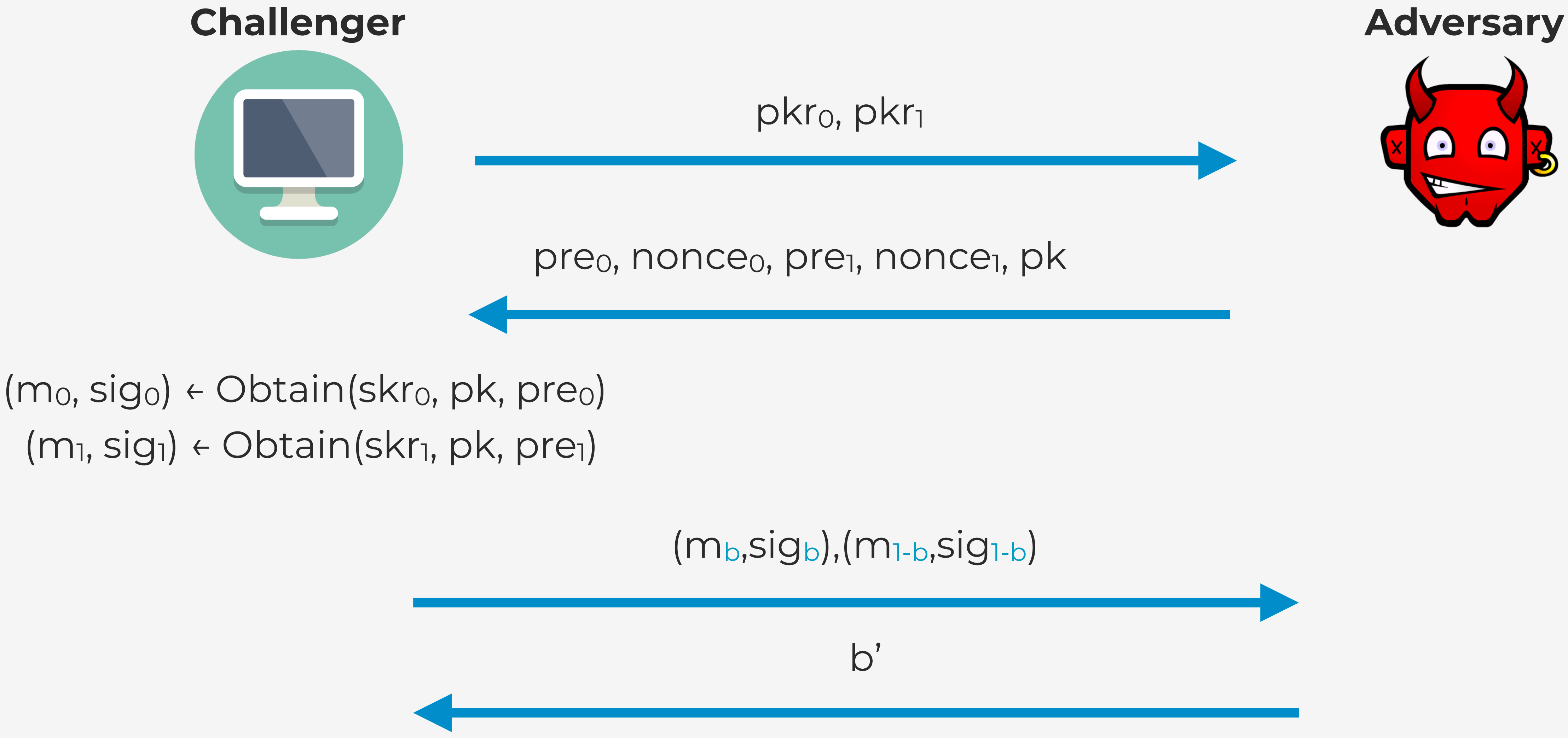
$pre_0, nonce_0, pre_1, nonce_1, pk$

$(m_0, sig_0) \leftarrow \text{Obtain}(skr_0, pk, pre_0)$

$(m_1, sig_1) \leftarrow \text{Obtain}(skr_1, pk, pre_1)$

$(m_b, sig_b), (m_{1-b}, sig_{1-b})$

Recipient Blindness



Challenger

Adversary

pkr_0, pkr_1

pre_0, $nonce_0$, pre_1, $nonce_1$, pk

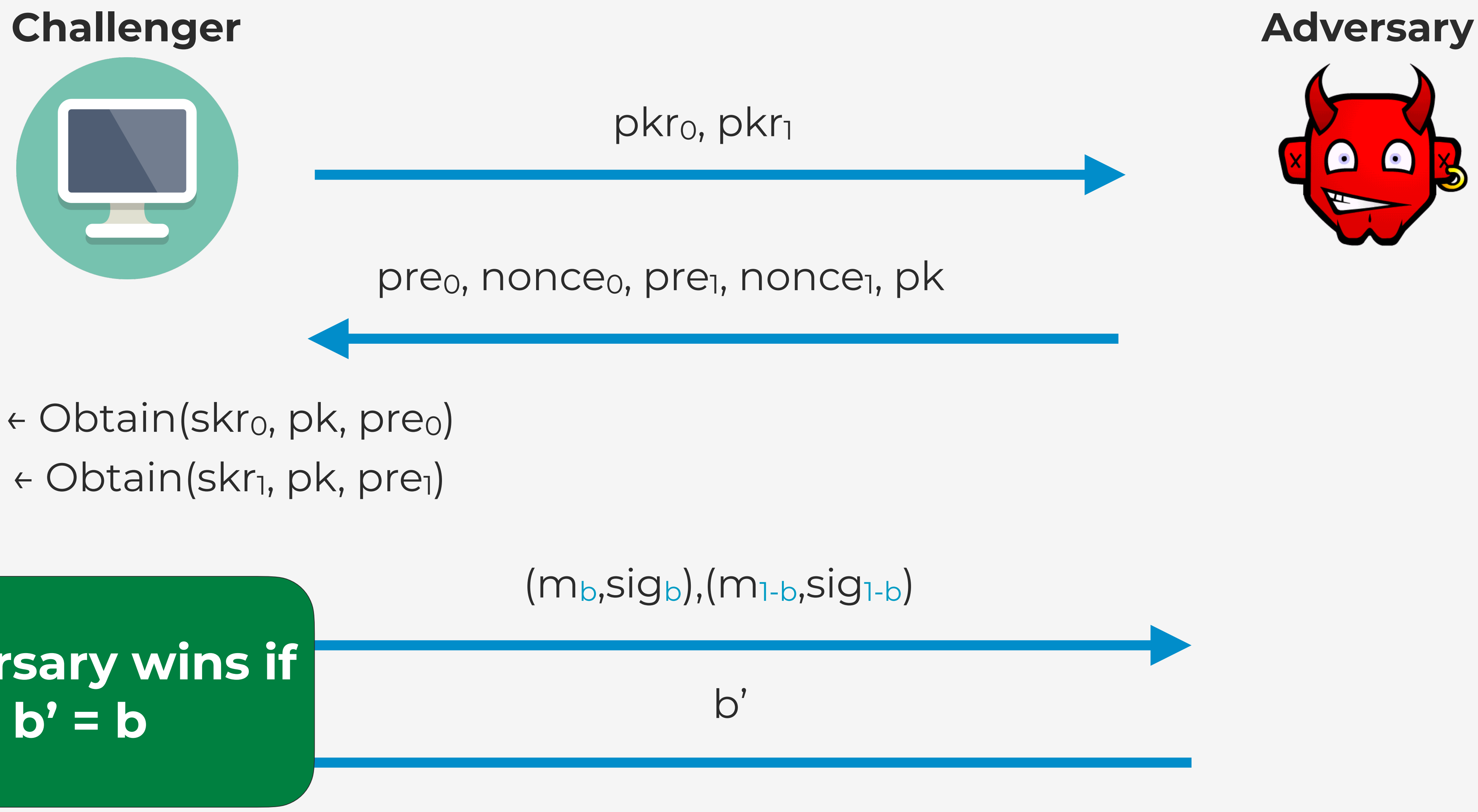
$(m_0, sig_0) \leftarrow \text{Obtain}(skr_0, pk, pre_0)$

$(m_1, sig_1) \leftarrow \text{Obtain}(skr_1, pk, pre_1)$

$(m_b, sig_b), (m_{1-b}, sig_{1-b})$

b’

Recipient Blindness



Challenger

Adversary

pkr_0, pkr_1

$pre_0, nonce_0, pre_1, nonce_1, pk$

$(m_0, sig_0) \leftarrow \text{Obtain}(skr_0, pk, pre_0)$

$(m_1, sig_1) \leftarrow \text{Obtain}(skr_1, pk, pre_1)$

$(m_b, sig_b), (m_{1-b}, sig_{1-b})$

b'

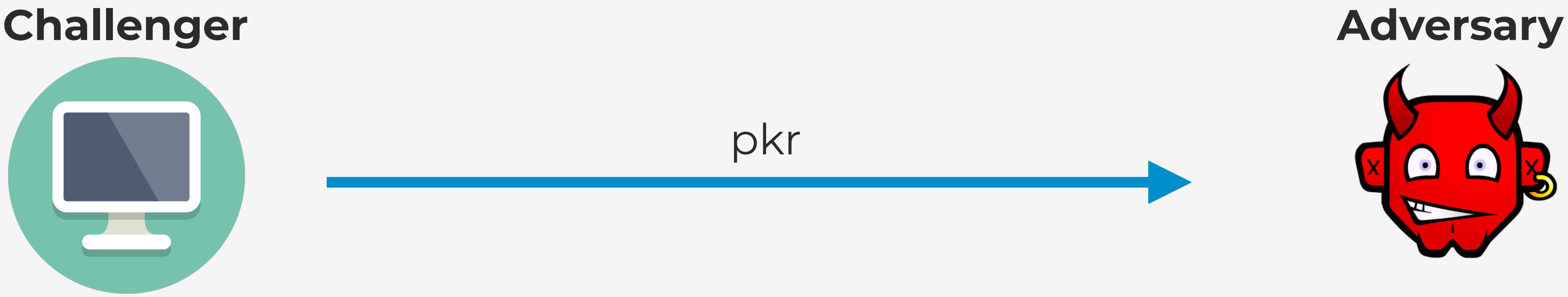
Adversary wins if $b' = b$

Nonce Blindness

Challenger

Adversary

Nonce Blindness

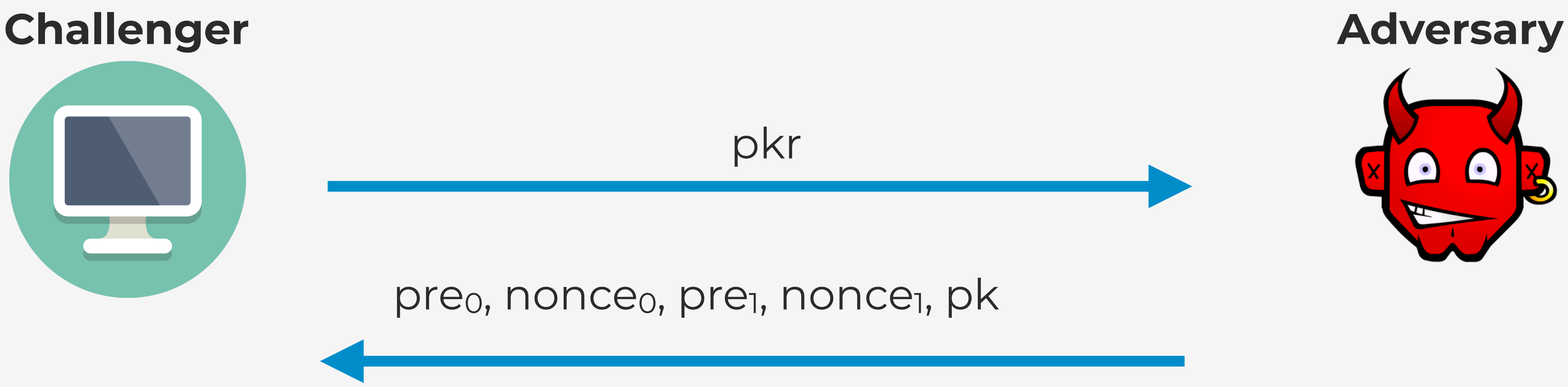


Challenger

pkr

Adversary

Nonce Blindness



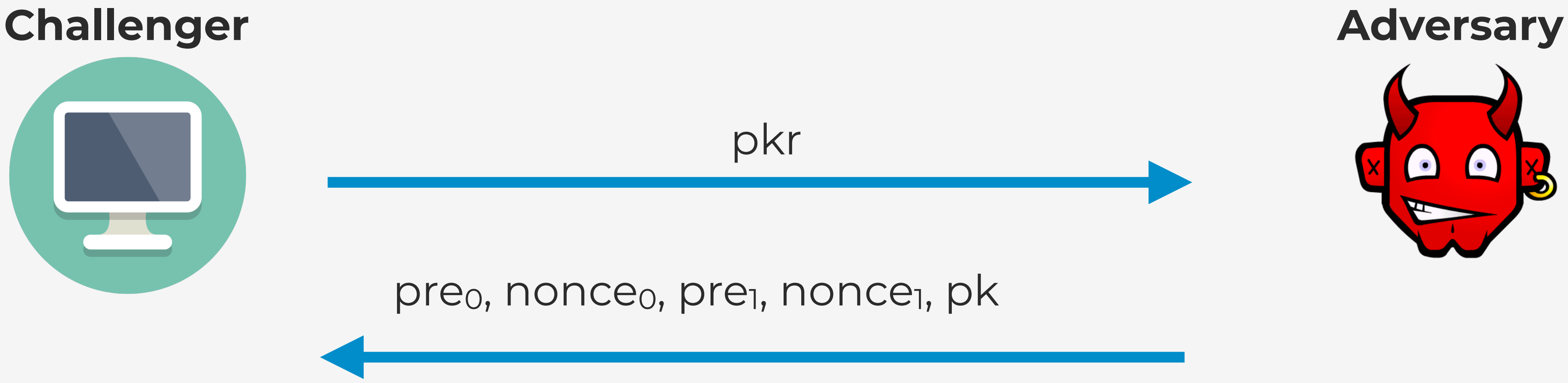
Challenger

Adversary

pkr

pre_0, $nonce_0$, pre_1, $nonce_1$, pk

Nonce Blindness



Challenger

Adversary

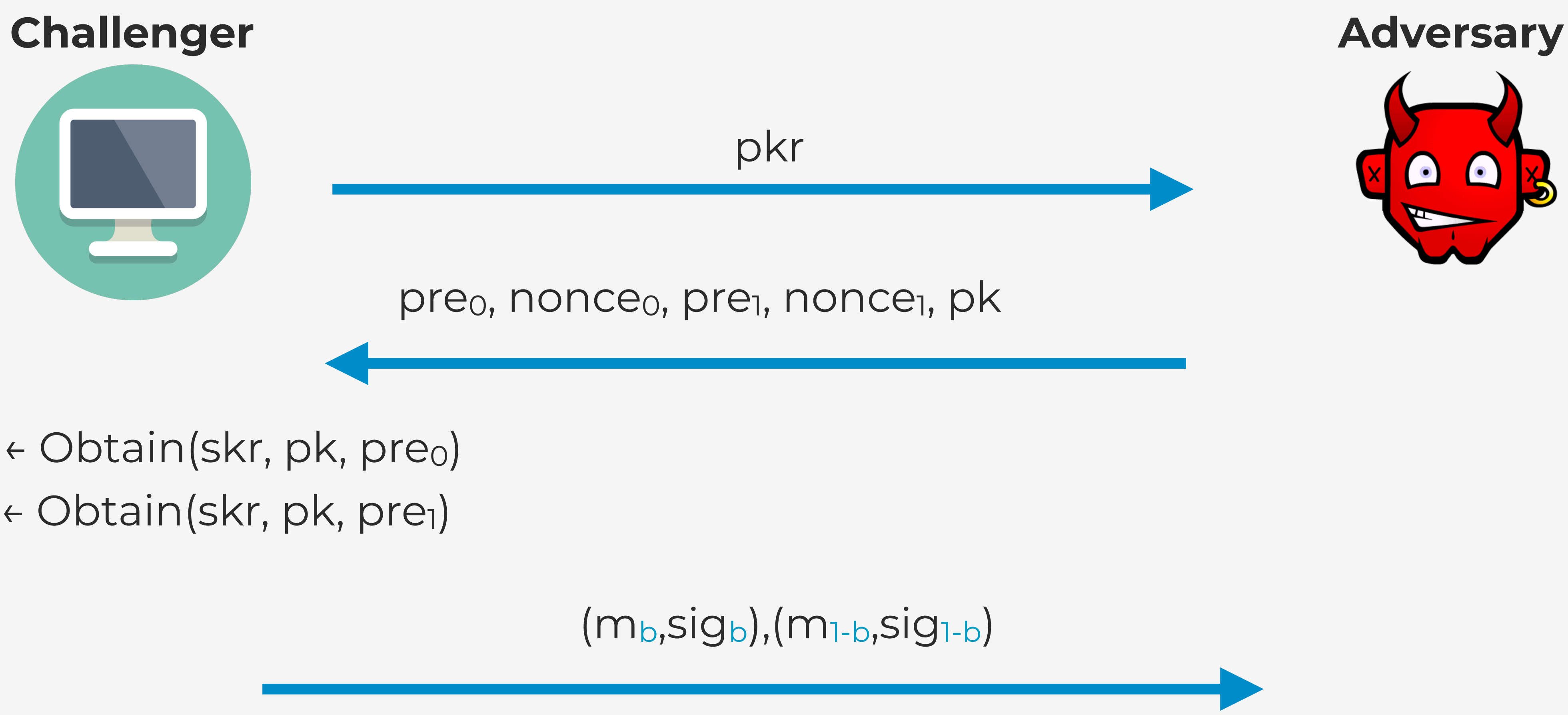
pkr

$pre_0, nonce_0, pre_1, nonce_1, pk$

$(m_0, sig_0) \leftarrow Obtain(skr, pk, pre_0)$

$(m_1, sig_1) \leftarrow Obtain(skr, pk, pre_1)$

Nonce Blindness



Challenger

Adversary

pkr

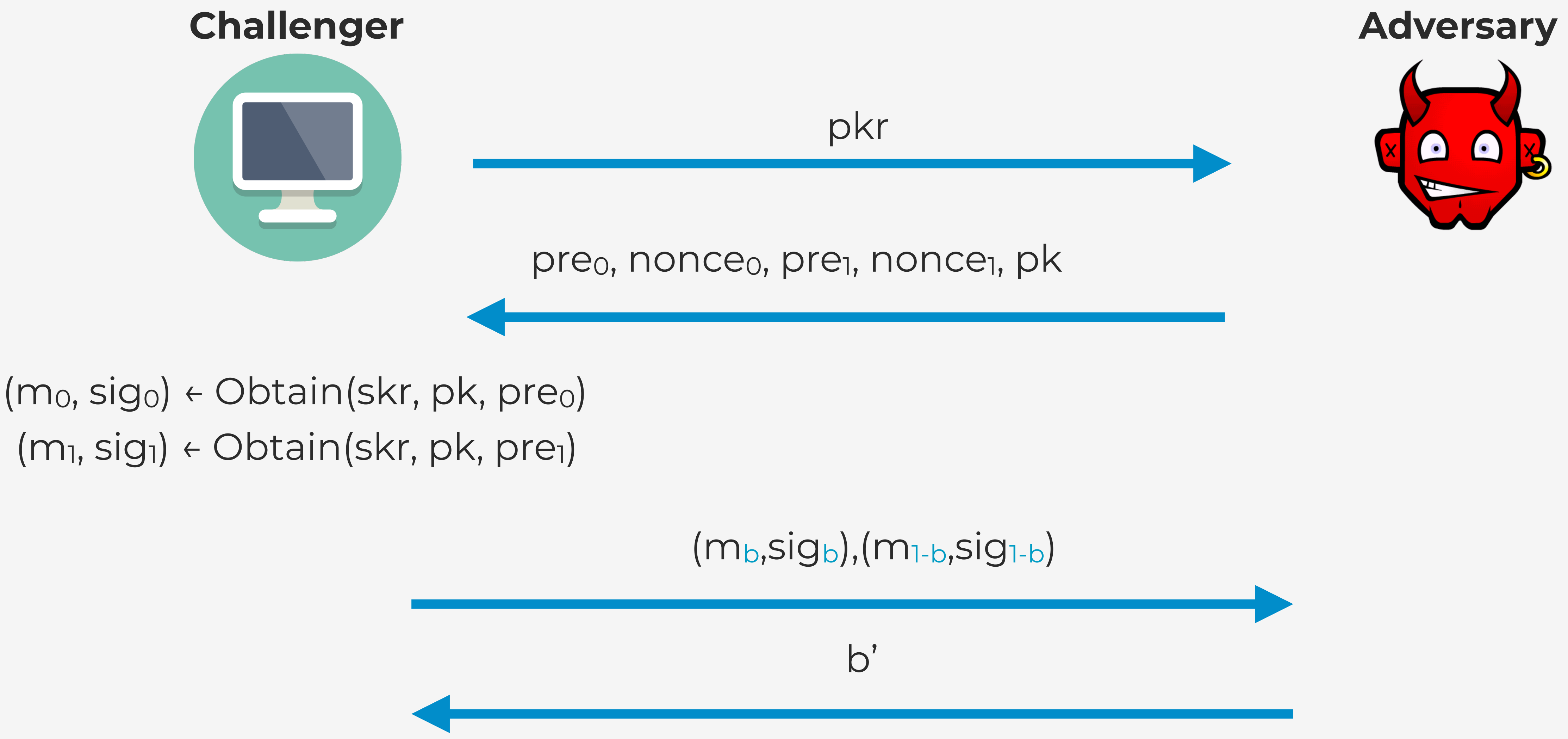
pre_0, $nonce_0$, pre_1, $nonce_1$, pk

$(m_0, sig_0) \leftarrow Obtain(skr, pk, pre_0)$

$(m_1, sig_1) \leftarrow Obtain(skr, pk, pre_1)$

$(m_b, sig_b), (m_{1-b}, sig_{1-b})$

Nonce Blindness



Challenger

Adversary

pkr

pre_0, $nonce_0$, pre_1, $nonce_1$, pk

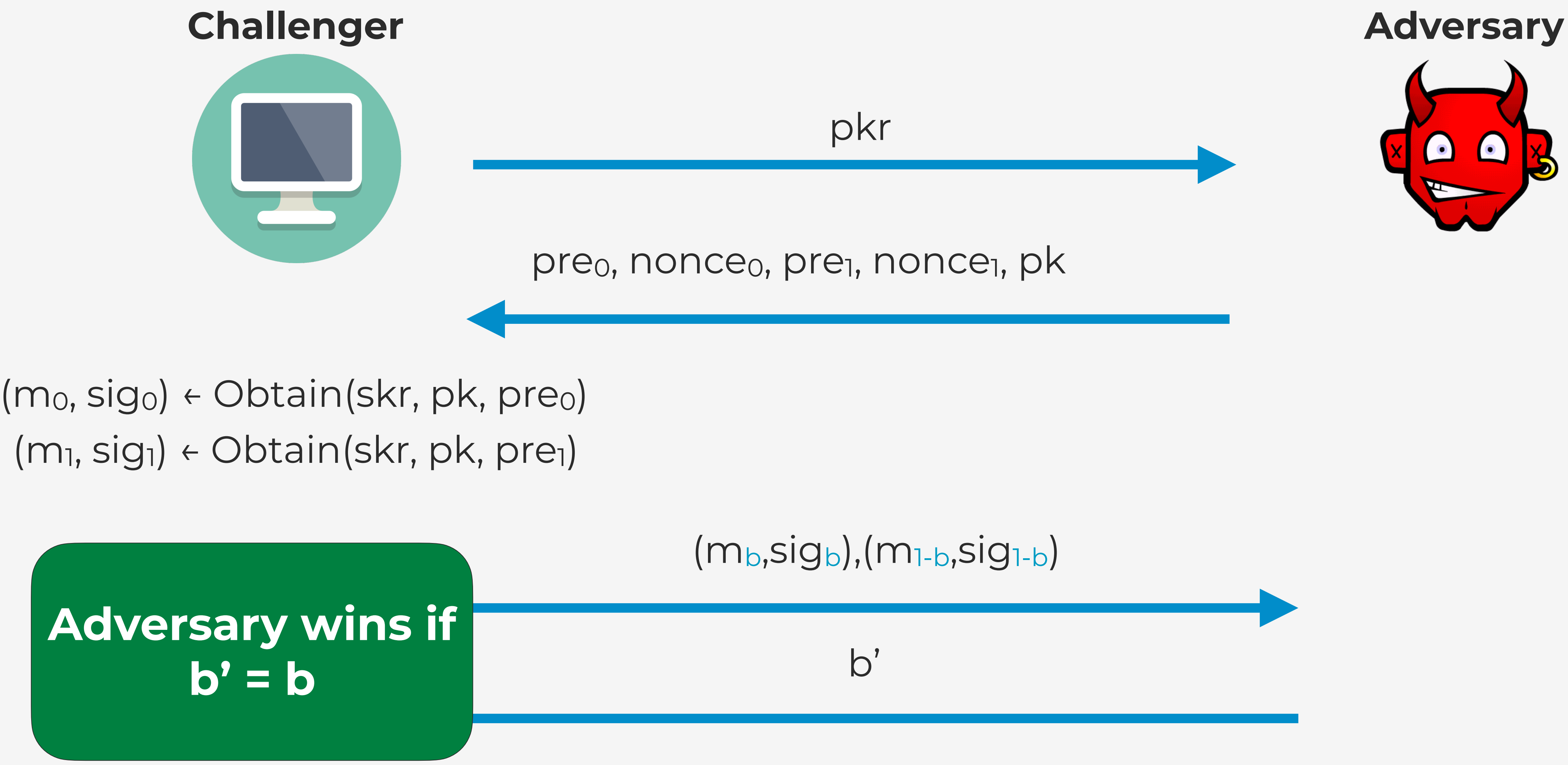
$(m_0, sig_0) \leftarrow \mathrm{Obtain}(skr, pk, pre_0)$

$(m_1, sig_1) \leftarrow \mathrm{Obtain}(skr, pk, pre_1)$

$(m_b,sig_b),(m_{1-b},sig_{1-b})$

b’

Nonce Blindness



Challenger

Adversary

pkr

pre$_0$, nonce$_0$, pre$_1$, nonce$_1$, pk

(m$_0$, sig$_0$) ← Obtain(skr, pk, pre$_0$)

(m$_1$, sig$_1$) ← Obtain(skr, pk, pre$_1$)

(m$_b$,sig$_b$),(m$_{1-b}$,sig$_{1-b}$)

b’

Adversary wins if b’ = b

Preliminaries

Signatures on Equivalence Classes

Preliminaries

Signatures on Equivalence Classes

$$m = (g^{a}, g^{b}) \xleftrightarrow{\text{equivalent messages}} m' = (g^{ra}, g^{rb})$$

Preliminaries

Signatures on Equivalence Classes

$$\begin{tikzcd}
m = (g^{a}, g^{b}) \arrow[rr, leftrightarrow, "\text{equivalent messages}"] & & m' = (g^{ra}, g^{rb}) \\
\mathrm{eqsig}(m) \arrow[rr, "\text{adaptation without}", "\text{signing key}"'] & & \mathrm{eqsig}'(m')
\end{tikzcd}$$

Preliminaries

Signatures on Equivalence Classes

$$\begin{tikzcd}[column sep=huge, row sep=huge]
m = (g^{a}, g^{b}) \arrow[r, leftrightarrow, "\text{equivalent messages}"] & m' = (g^{ra}, g^{rb}) \\
\mathrm{eqsig}(m) \arrow[r, "\text{adaptation without}", "\text{signing key}"'] & \mathrm{eqsig}'(m')
\end{tikzcd}$$

Random signature even if signer malicious

How to efficiently construct NIBS?

skr^{-1}

skr^{-1}

How to efficiently construct NIBS?

Issue(sk,pkr,nonce)

pre := eqsig(pkr, H(nonce))

skr^{-1}

skr^{-1}

How to efficiently construct NIBS?

Issue(sk,pkr,nonce)

pre := eqsig(pkr, H(nonce))

skr^{-1}

Obtain(skr,pk,pre,nonce)

sig := adapt(pre, skr^{-1})

m := $H(nonce)^{skr^{-1}}$

How to efficiently construct NIBS?

Issue(sk,pkr,nonce)

pre := eqsig(pkr, H(nonce))

Obtain(skr,pk,pre,nonce)

sig := adapt(pre, skr^{-1})

m := $H(nonce)^{skr^{-1}}$

sig is actually
eqsig((g, $H(nonce)^{skr^{-1}}$)

How to efficiently construct NIBS?

Issue(sk,pkr,nonce)

pre := eqsig(pkr, H(nonce))

Obtain(skr,pk,pre,nonce)

sig := adapt(pre, skr^{-1})

m := $\text{H(nonce)}^{\text{skr}^{-1}}$

sig is actually
eqsig((g, $\text{H(nonce)}^{\text{skr}^{-1}}$)

pkr is a standard DH key!

Why does it work?

I. Unforgeability from signatures on equivalence signatures

II. $\text{H(nonce)}^{\text{skr}^{-1}}$ is a PRF for the recipient’s key

III. Blindness follows from inverse DDH

Can we Date NIBS? YES!

I. Signer can add a tag to presignatures that will be preserved

II. Security notions can be easily adapted to include the tag

III. Same construction can be used but with tag-based signatures on equivalence classes [HS21]

Summary and Open Problems

- NIBS and TNIBS definitions
- Efficient constructions that work with standard PKI keys
- Generic construction from VRF and NIWI in ROM

- Can we construct PQ NIBS/TNIBS?
- Can we construct NIBS/TNIBS without ROM?

Contact

E-mail

hanzlik@cispa.de

eprint.iacr.org/2023/388

Eprint