Proactive Threshold Wallets With Offline Devices

Yashvanth Kondi, Bernardo Magri, Claudio Orlandi, Omer Shlomovits

Northeastern University

AARHUS UNIVERSITY

KZen

This Work

This Work

- We study proactive security where **dishonest majority speaks** during refresh, rest stay offline and “catch up” later

This Work

- We study proactive security where **dishonest majority speaks** during refresh, rest stay offline and “catch up” later

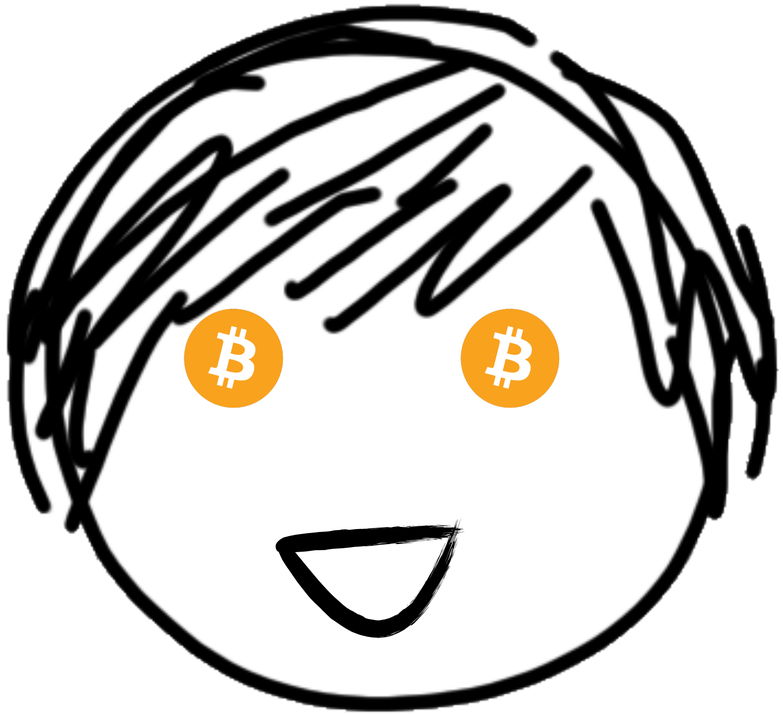
- Formalize notion of **unanimous erasure**

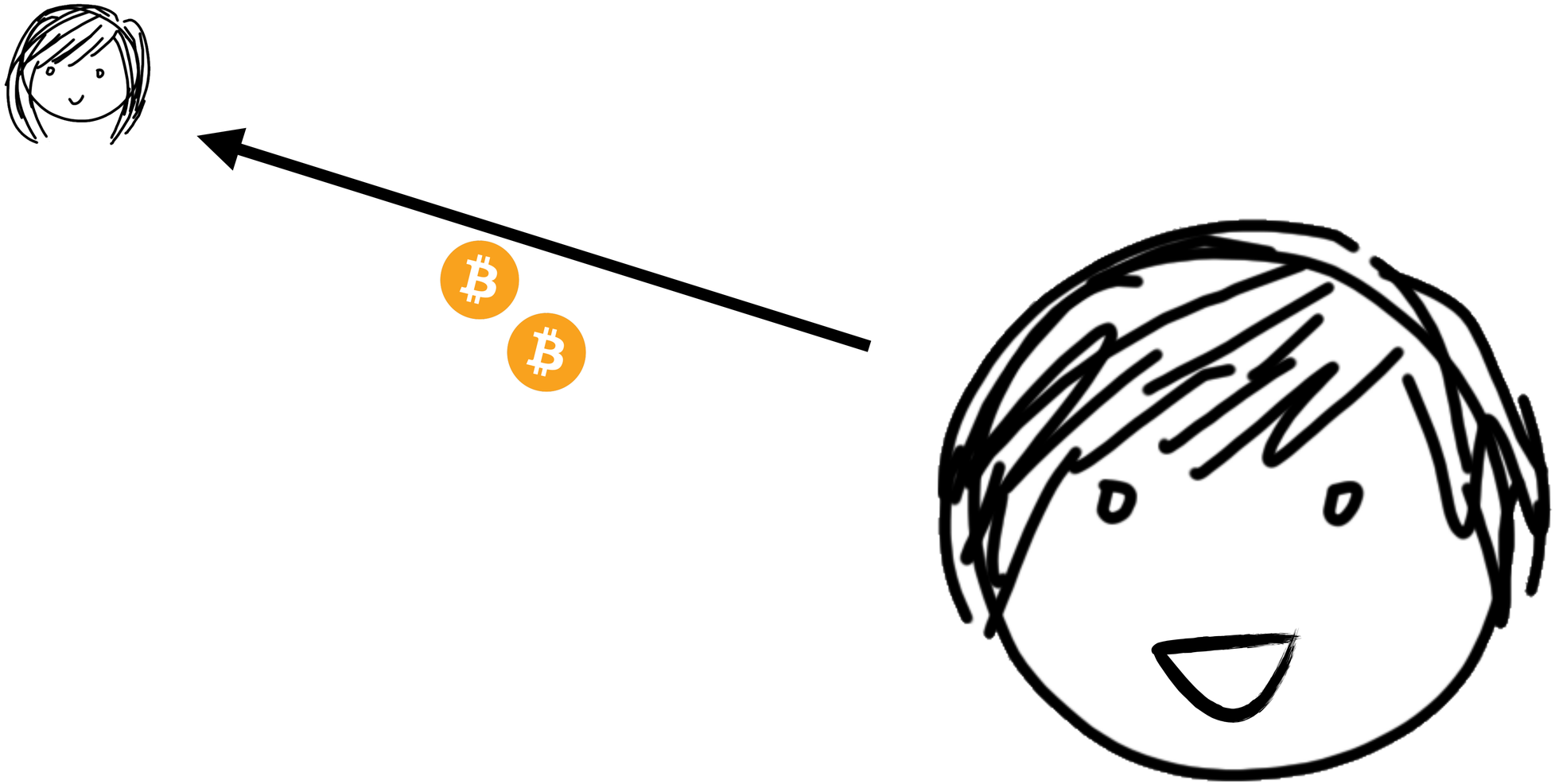
This Work

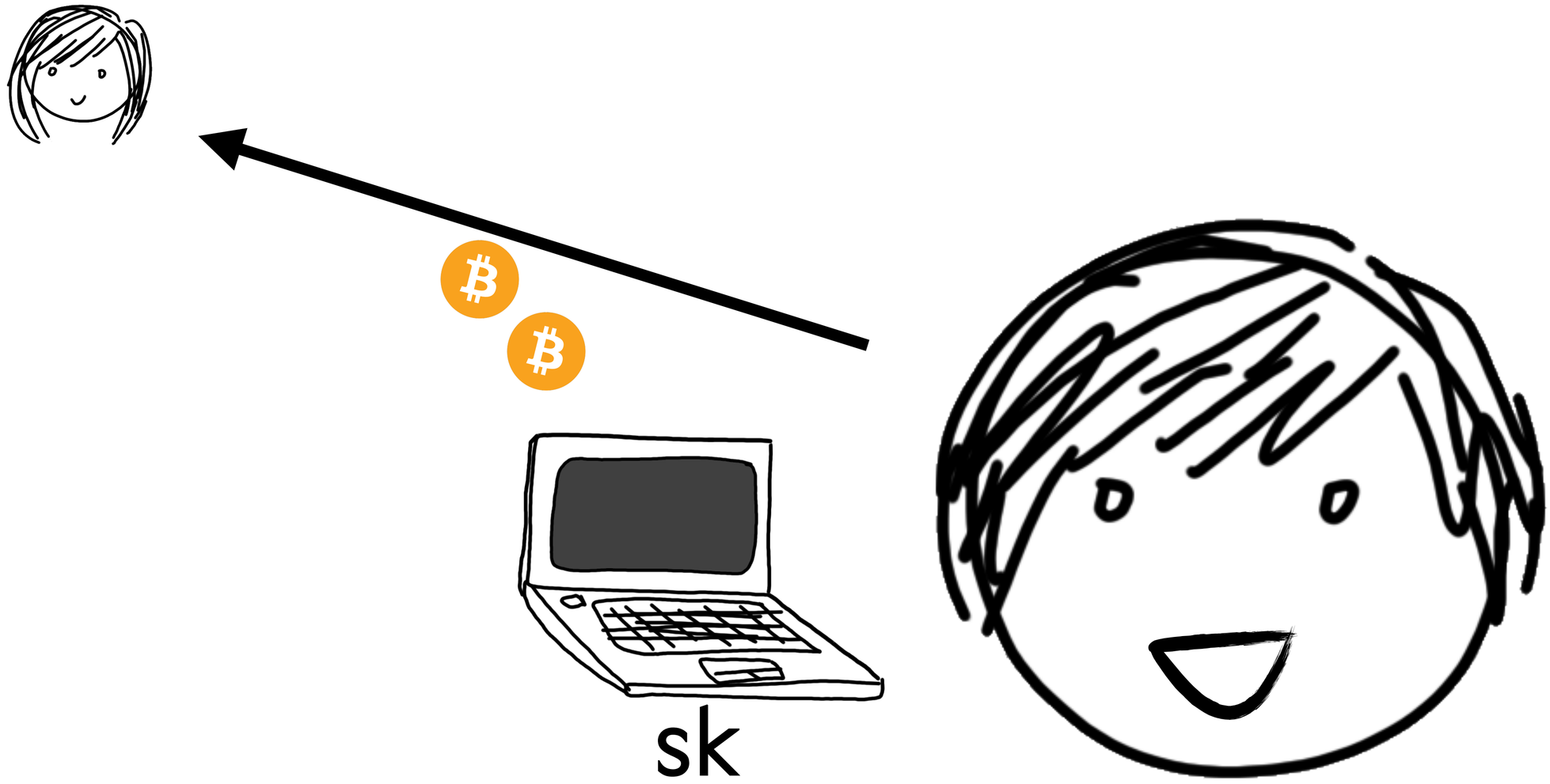
- We study proactive security where **dishonest majority speaks** during refresh, rest stay offline and “catch up” later
- Formalize notion of **unanimous erasure**
- **(2,*n*) setting**: **novel protocol** native to mode of operation for wallets, shown practical via implementation

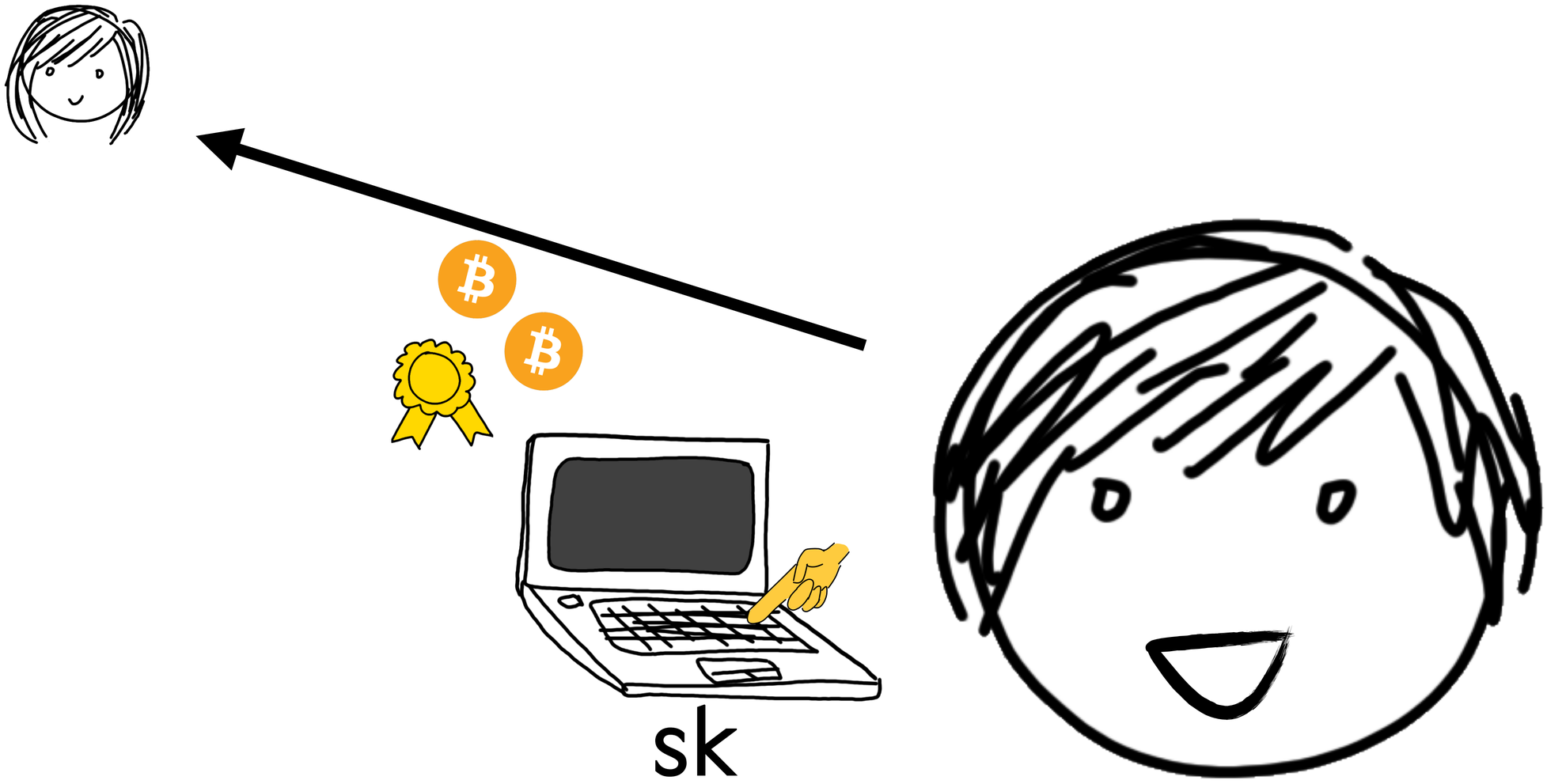
This Work

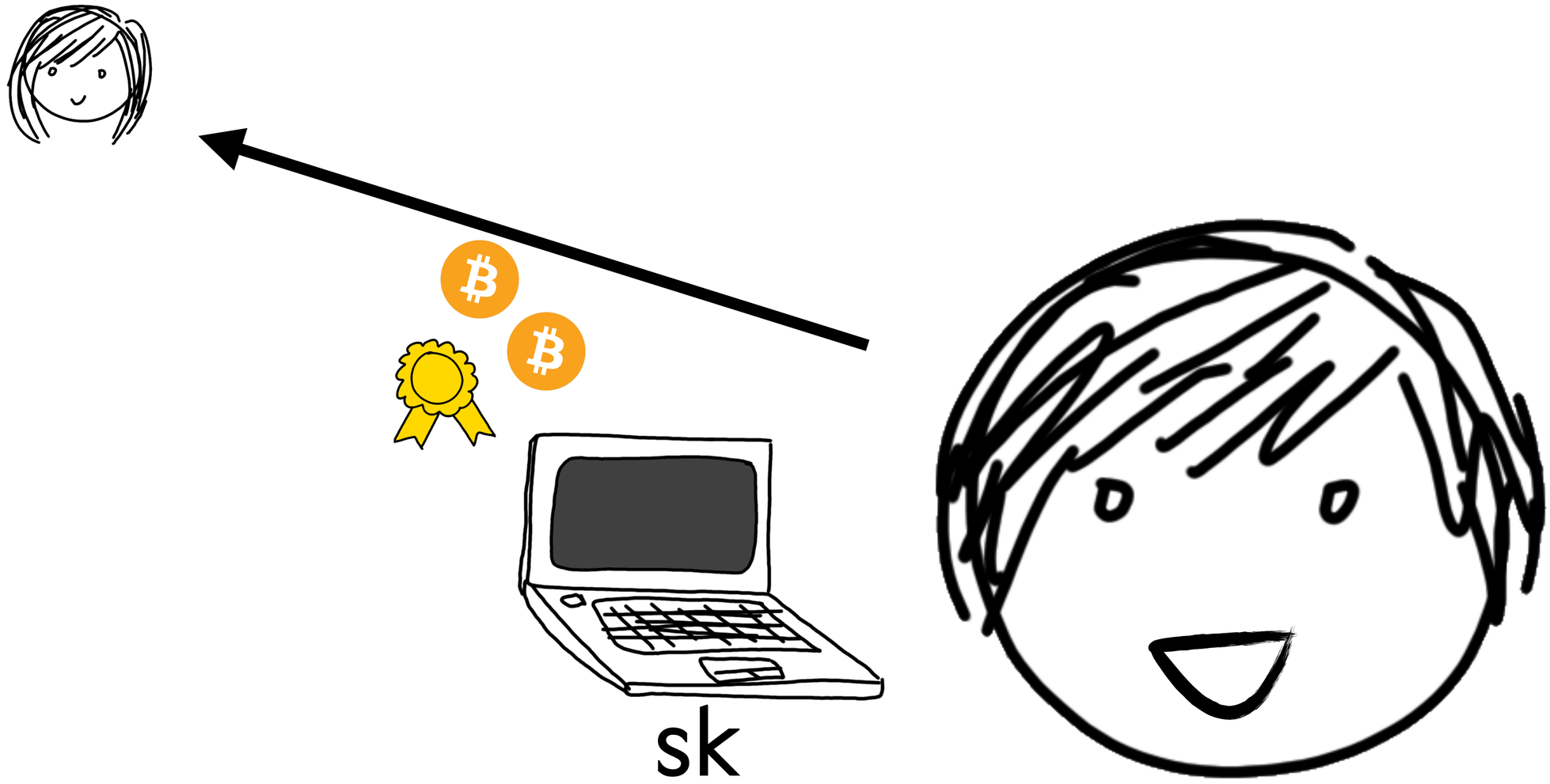
- We study proactive security where **dishonest majority speaks** during refresh, rest stay offline and “catch up” later
- Formalize notion of **unanimous erasure**
- **(2,*n*) setting**: **novel protocol** native to mode of operation for wallets, shown practical via implementation
- **(*t*,*n*) setting**: prove it’s **impossible** to achieve unanimous erasure in standard model (even given trusted setup, ledger)

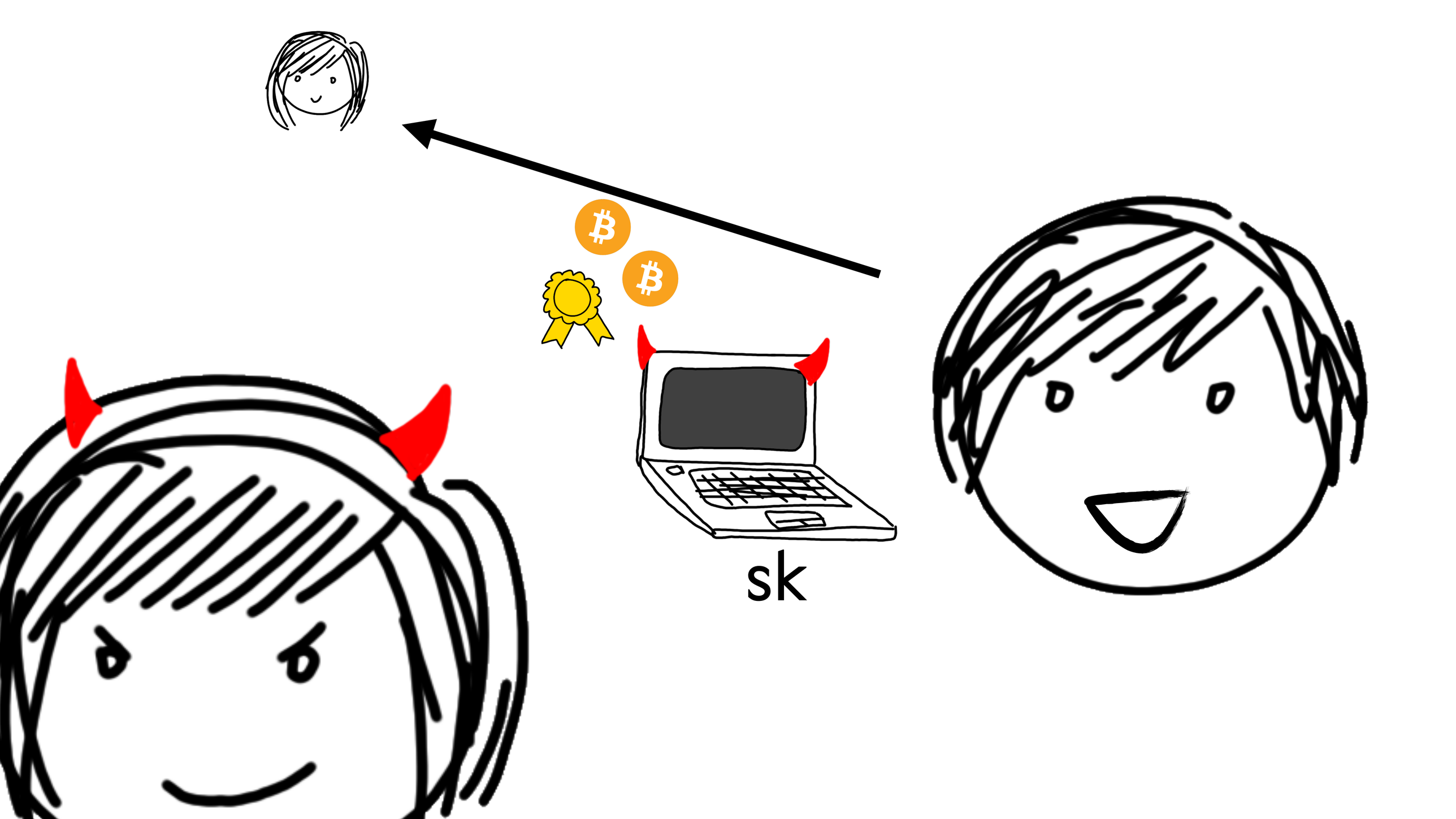


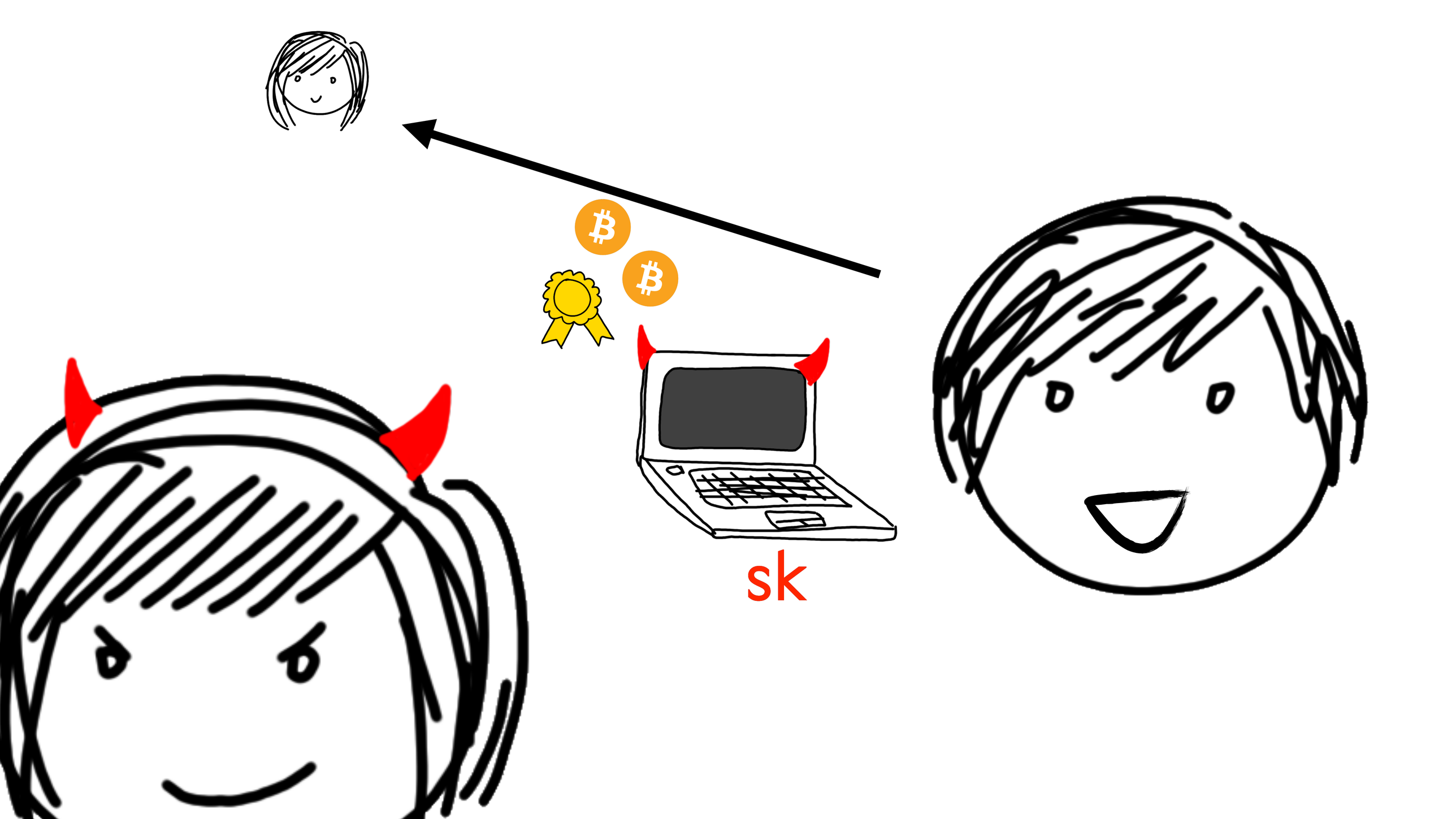


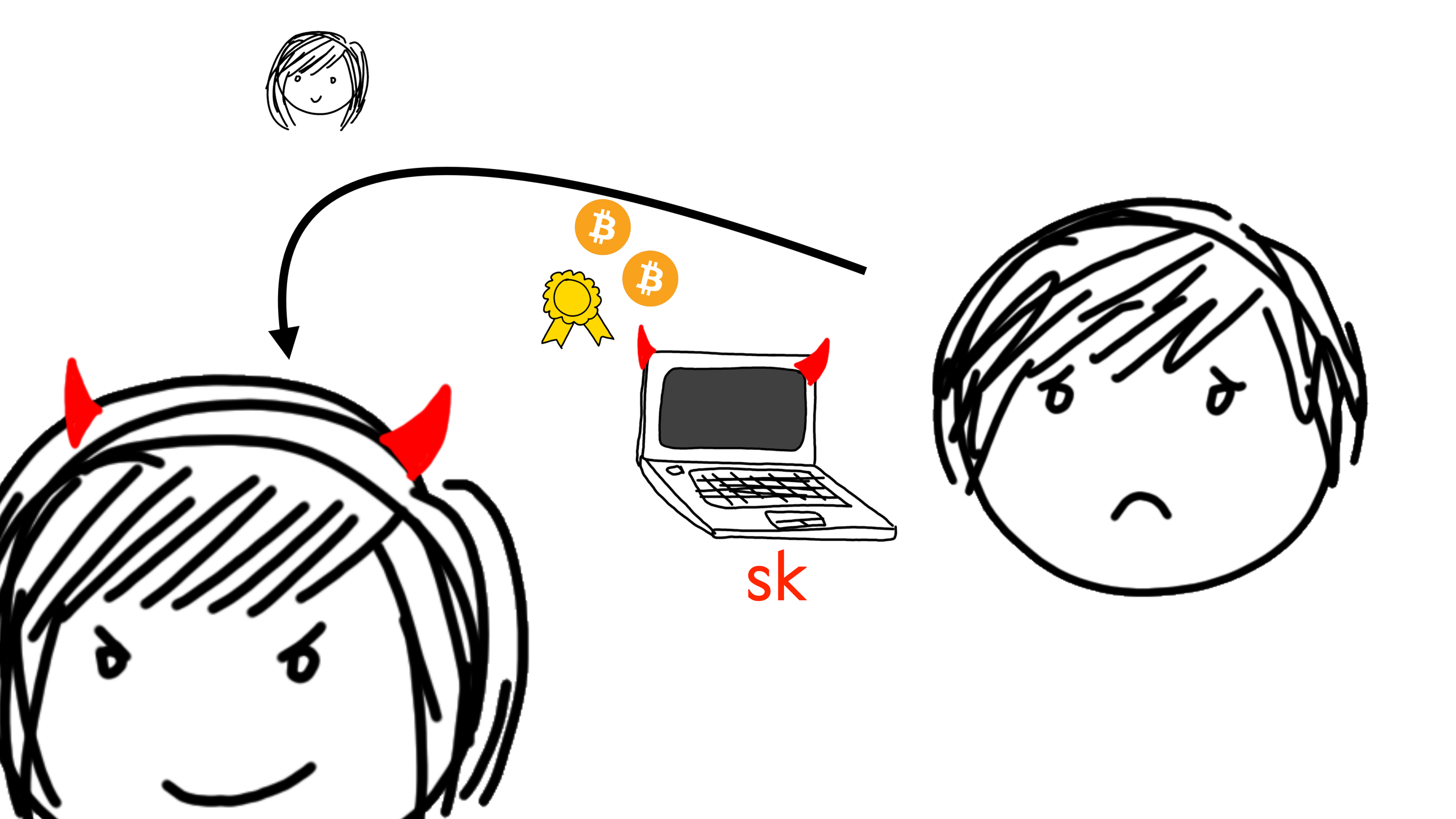

sk

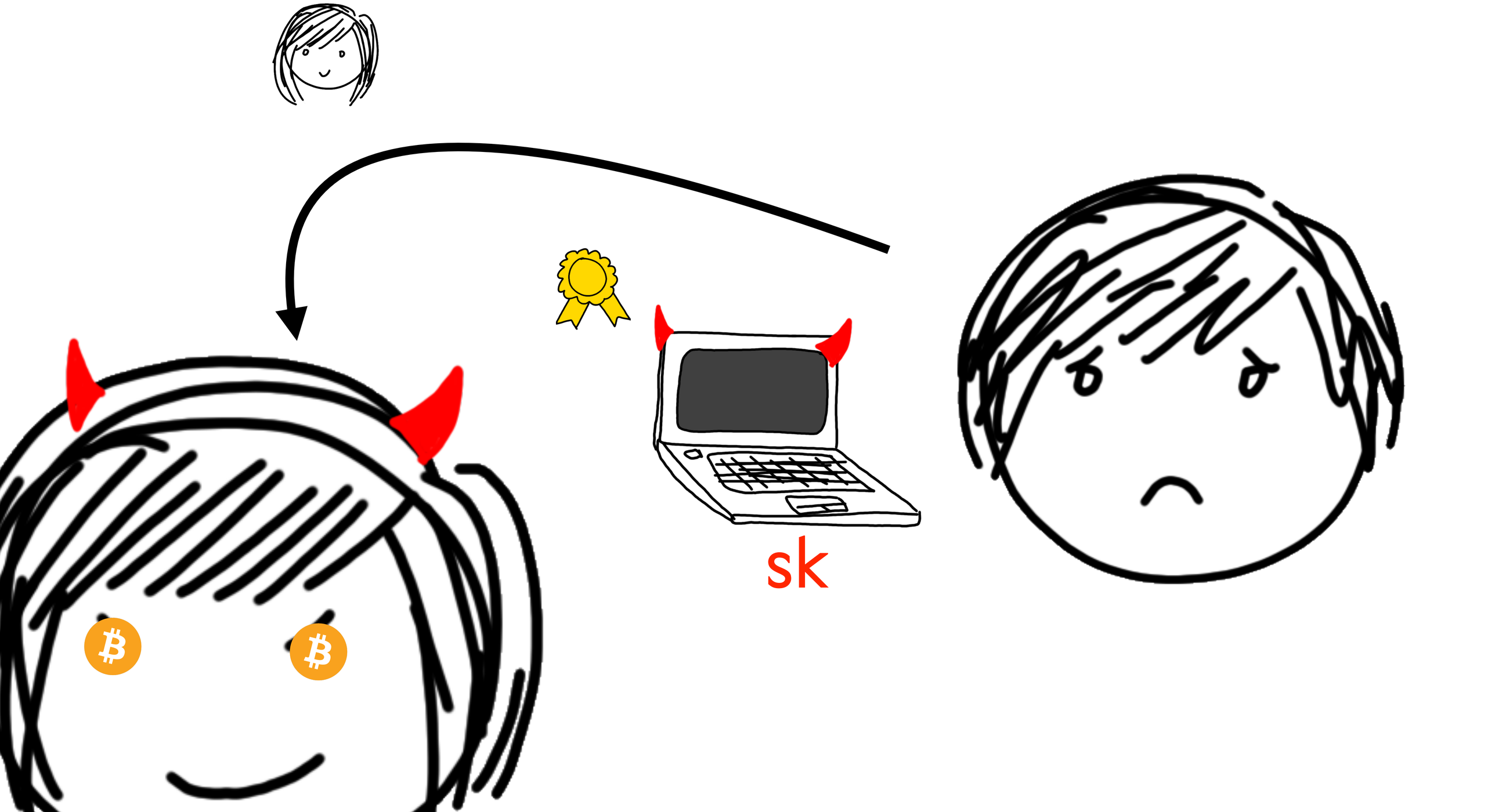

sk

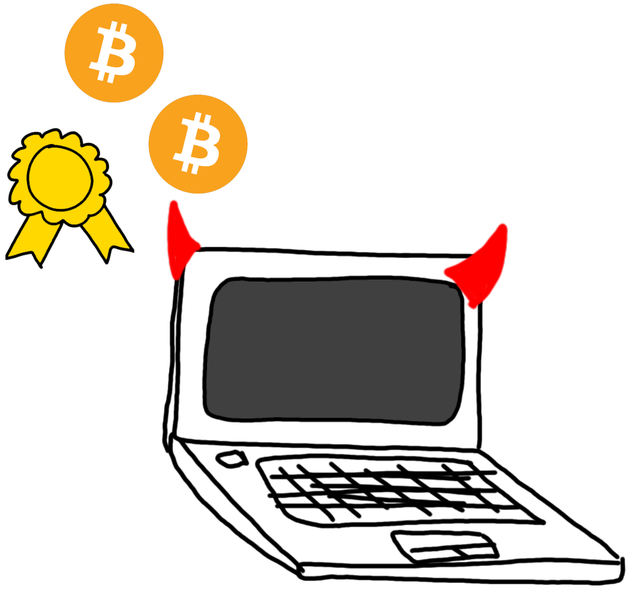

sk

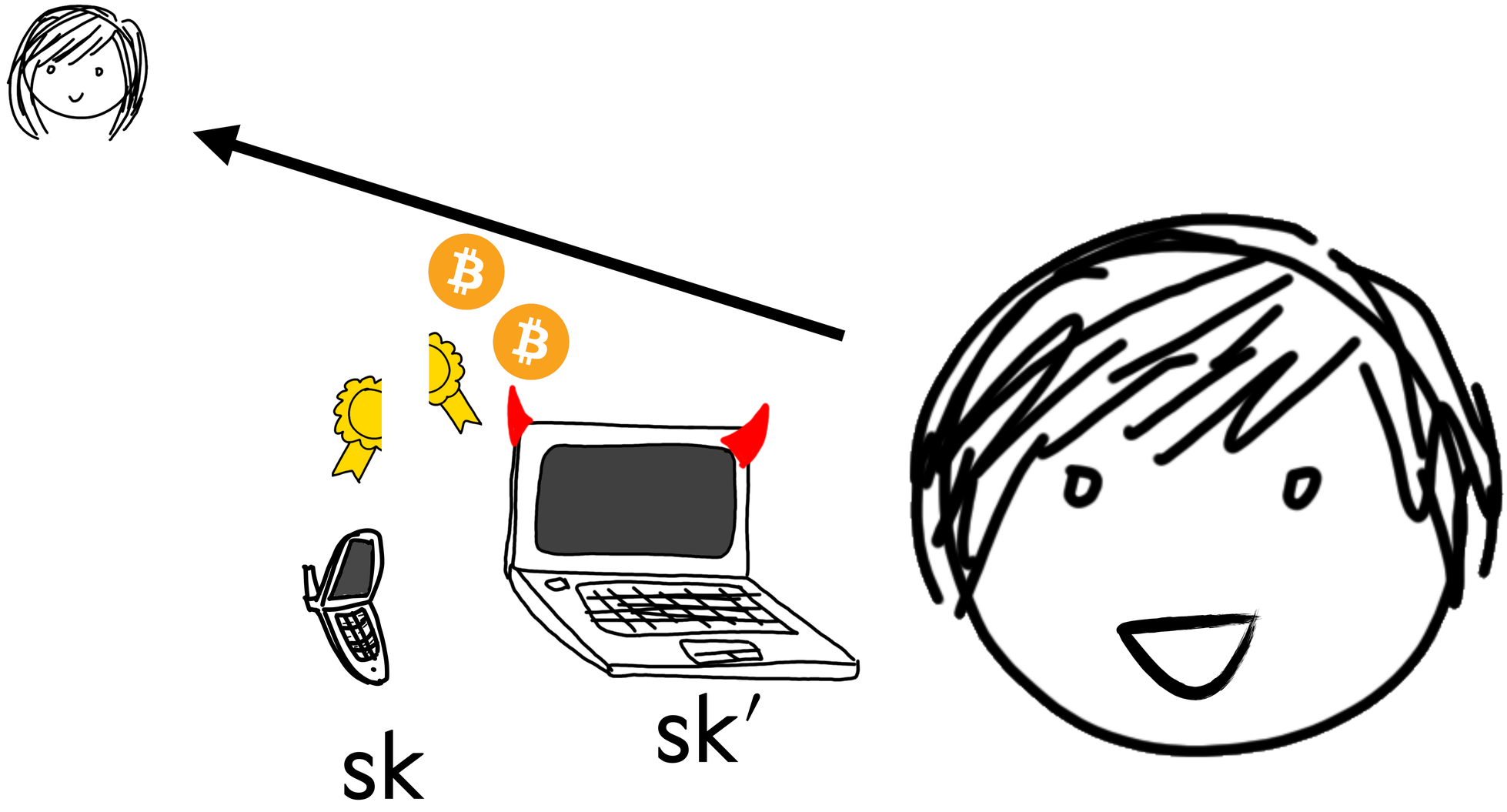

sk


sk

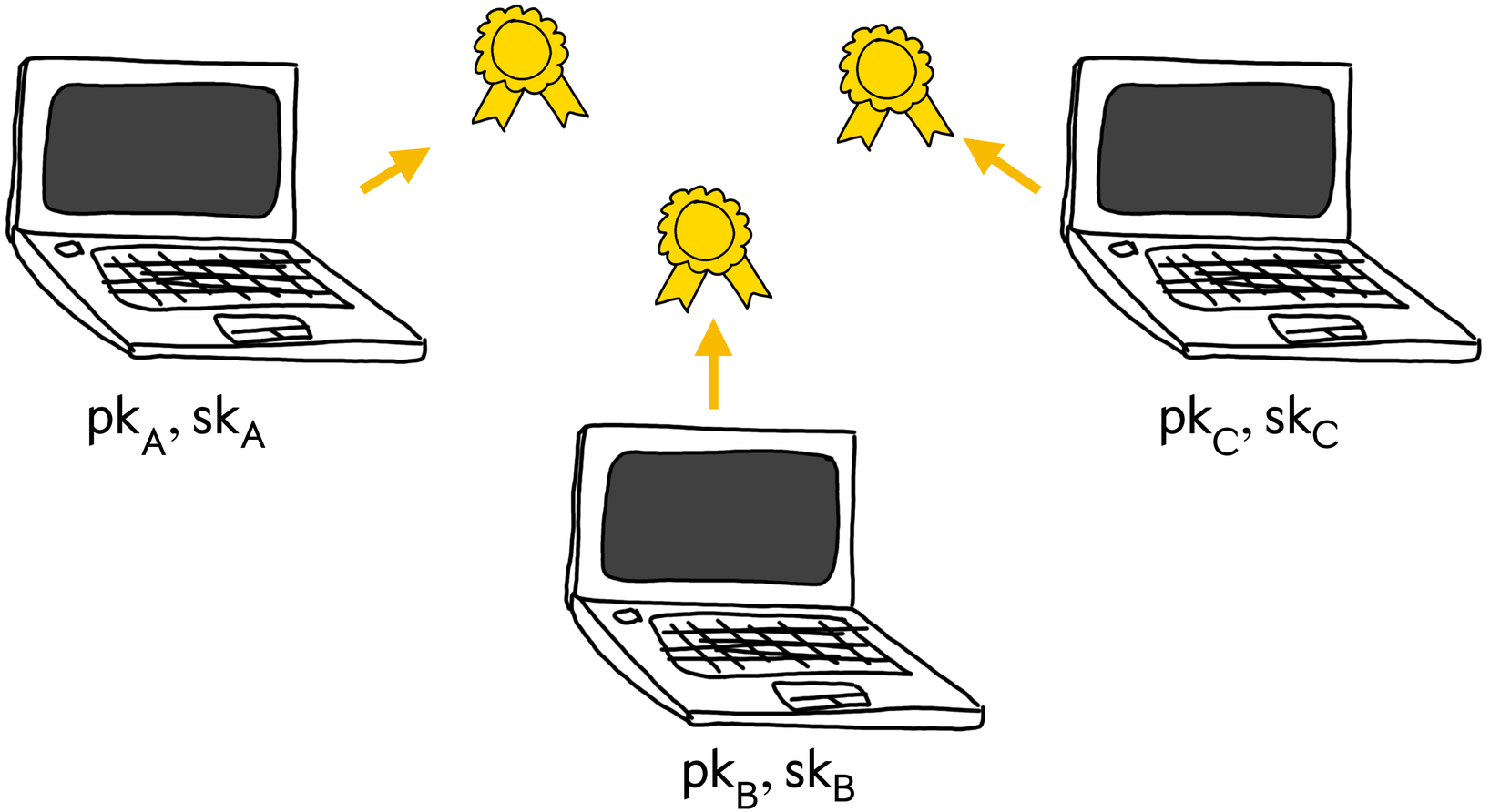

sk


sk




sk
sk′

Multi-Sig



Disadvantages

Disadvantages

- No Anonymity (org structure revealed)
- Size is linear in party count
- Not drop-in replacement

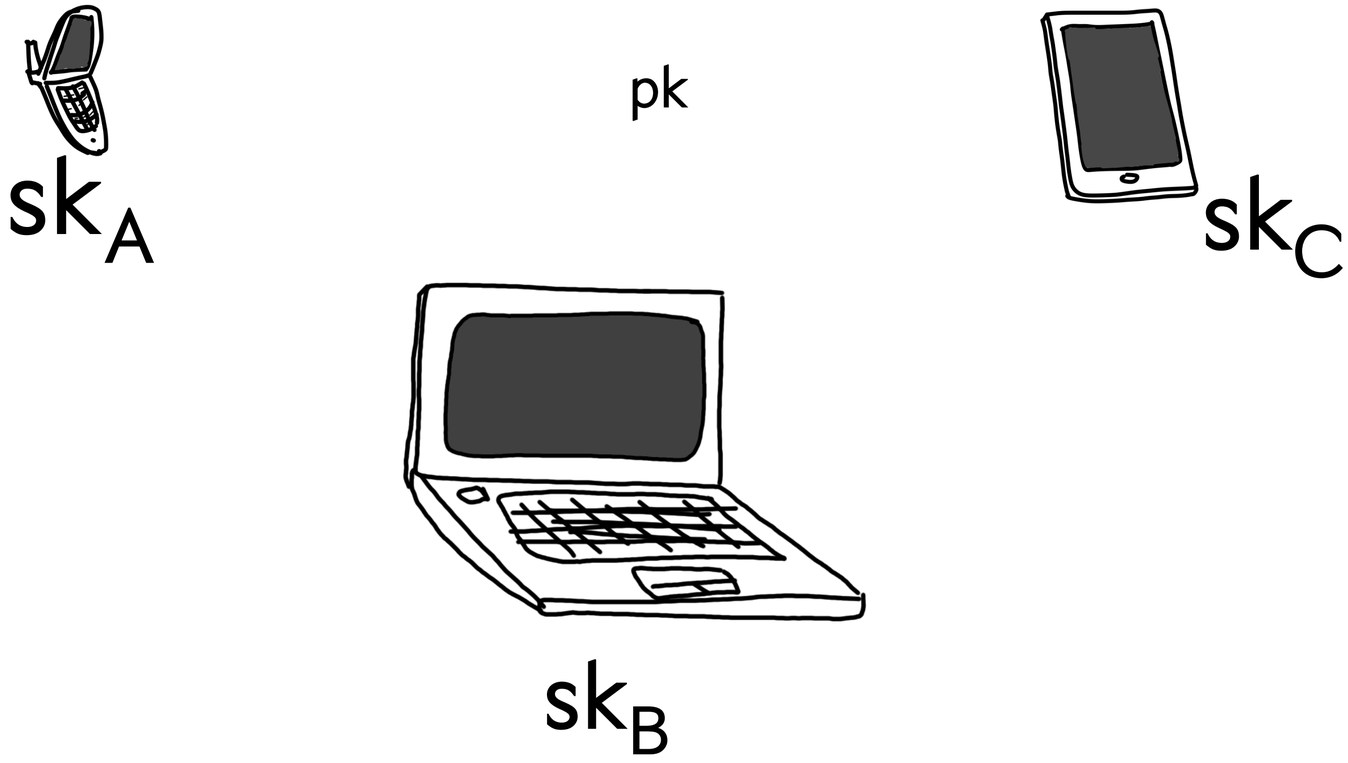
Threshold Signature

$$\{\mathsf{sk}_A, \mathsf{sk}_B, \mathsf{sk}_C\} \leftarrow \mathrm{Share}(\mathsf{sk})$$

pk

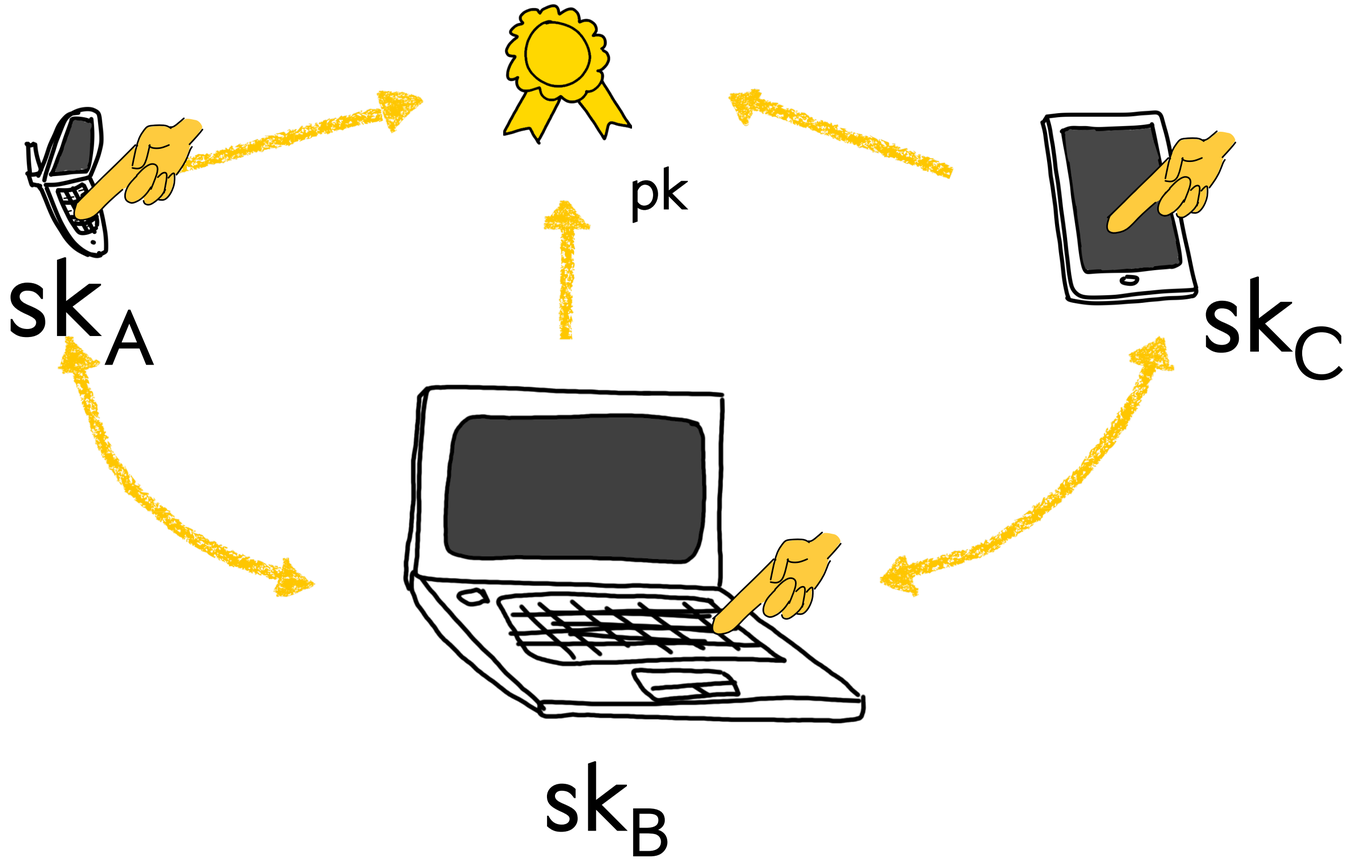
Threshold Signature

$$\{\mathsf{sk}_A, \mathsf{sk}_B, \mathsf{sk}_C\} \leftarrow \mathrm{Share}(\mathsf{sk})$$



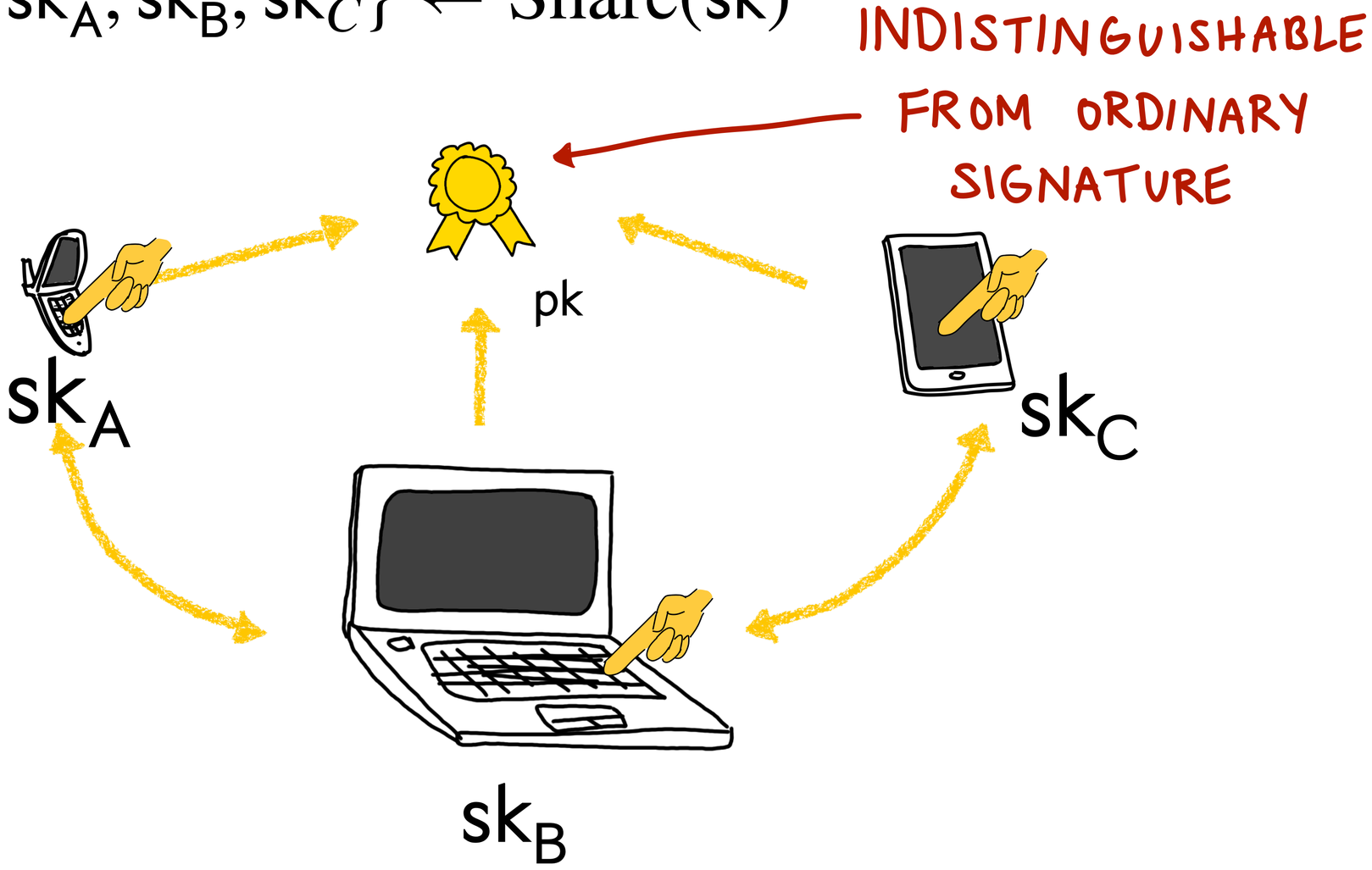
Threshold Signature

$$\{\mathsf{sk}_A, \mathsf{sk}_B, \mathsf{sk}_C\} \leftarrow \mathrm{Share}(\mathsf{sk})$$

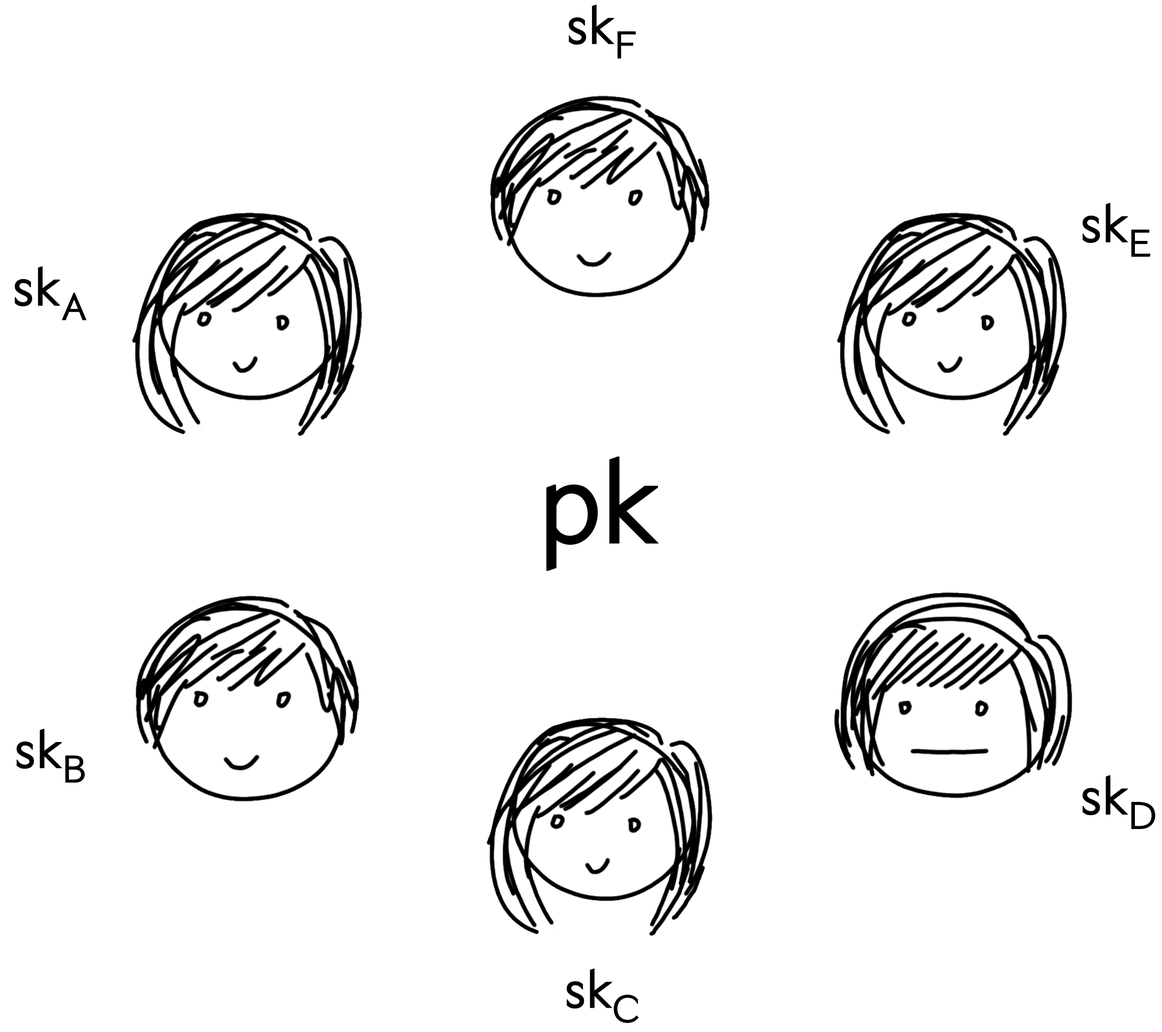


Threshold Signature

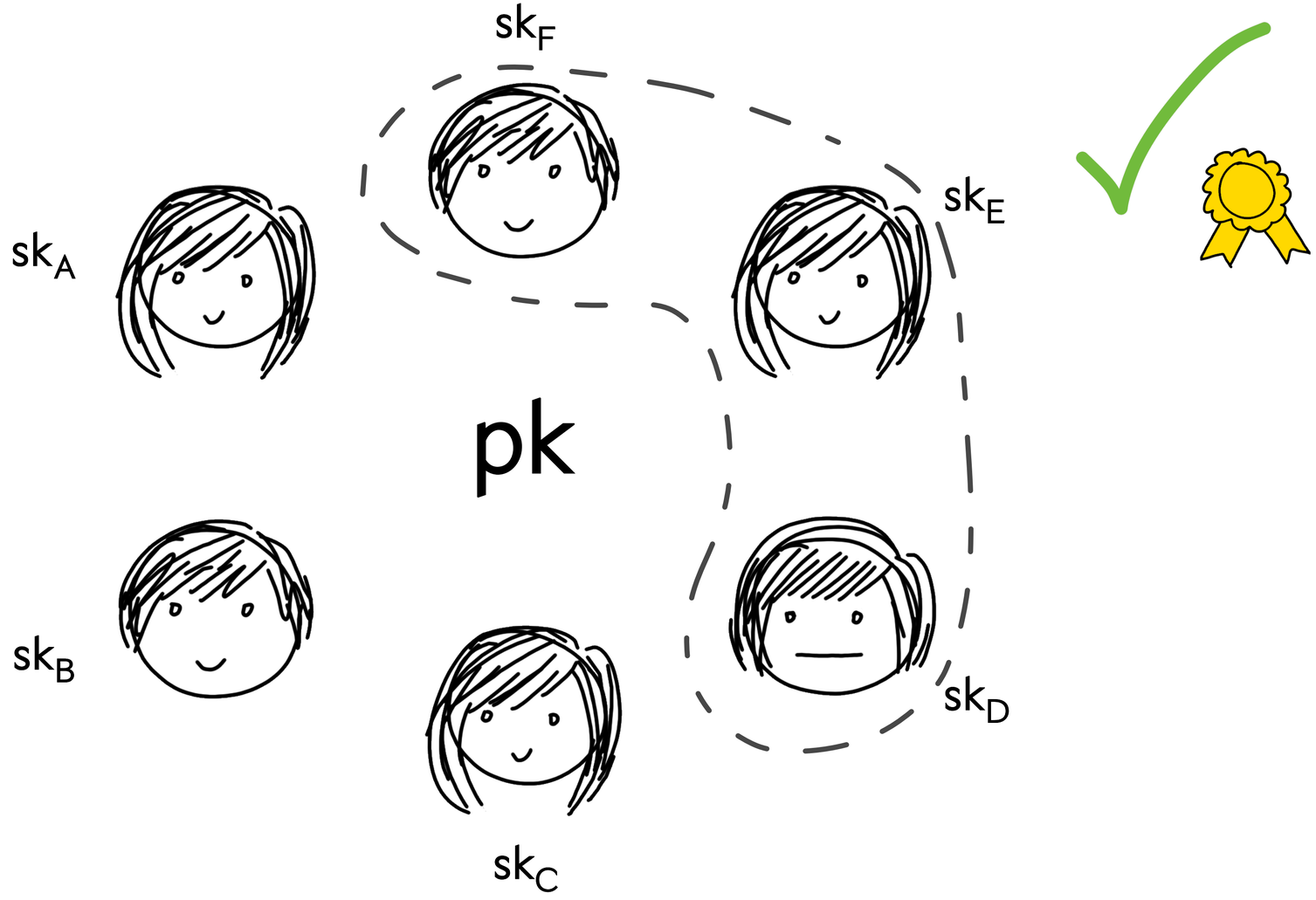
$$\{sk_A, sk_B, sk_C\} \leftarrow \text{Share}(sk)$$



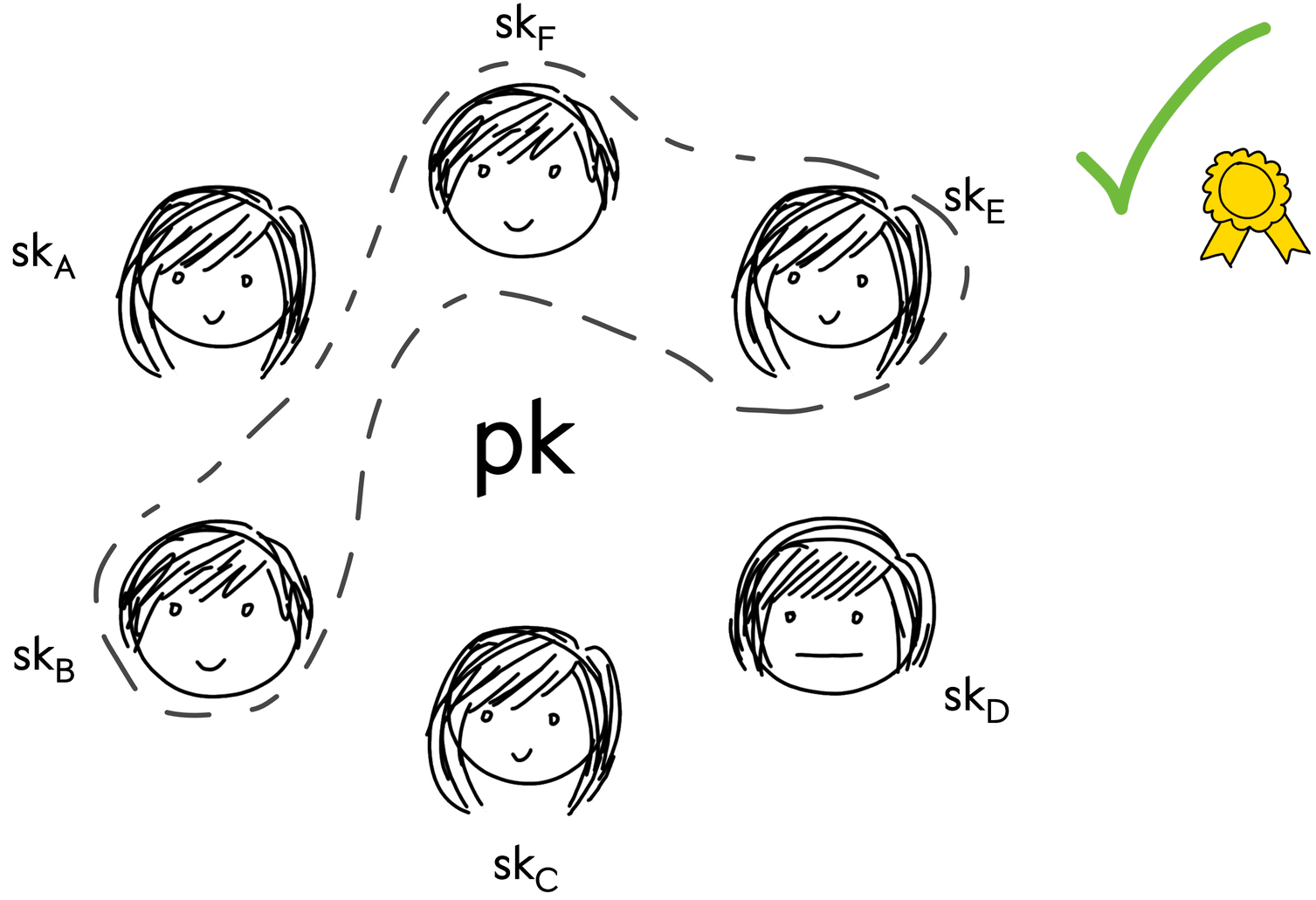
3-of-n Signature Scheme



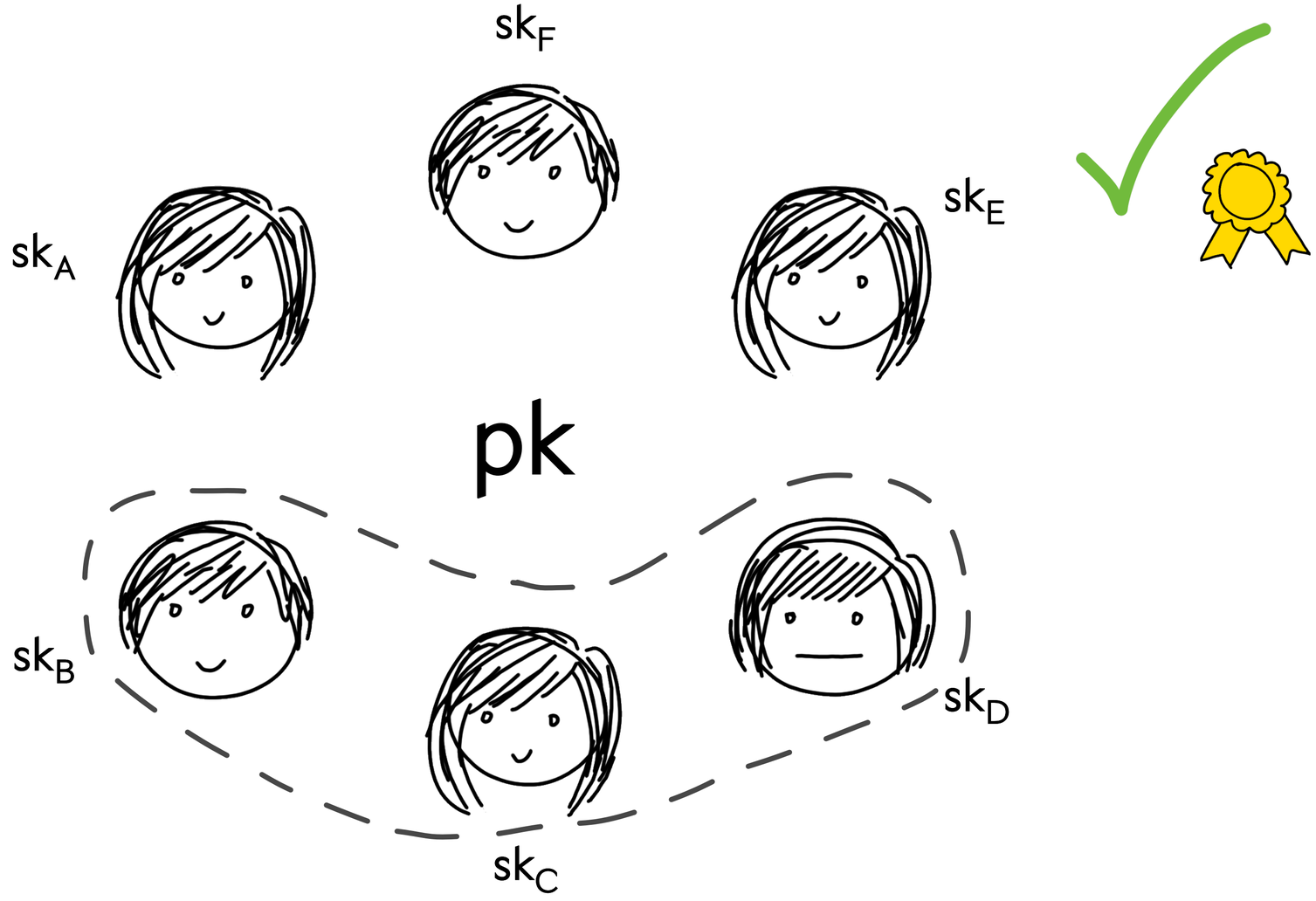
3-of-n Signature Scheme



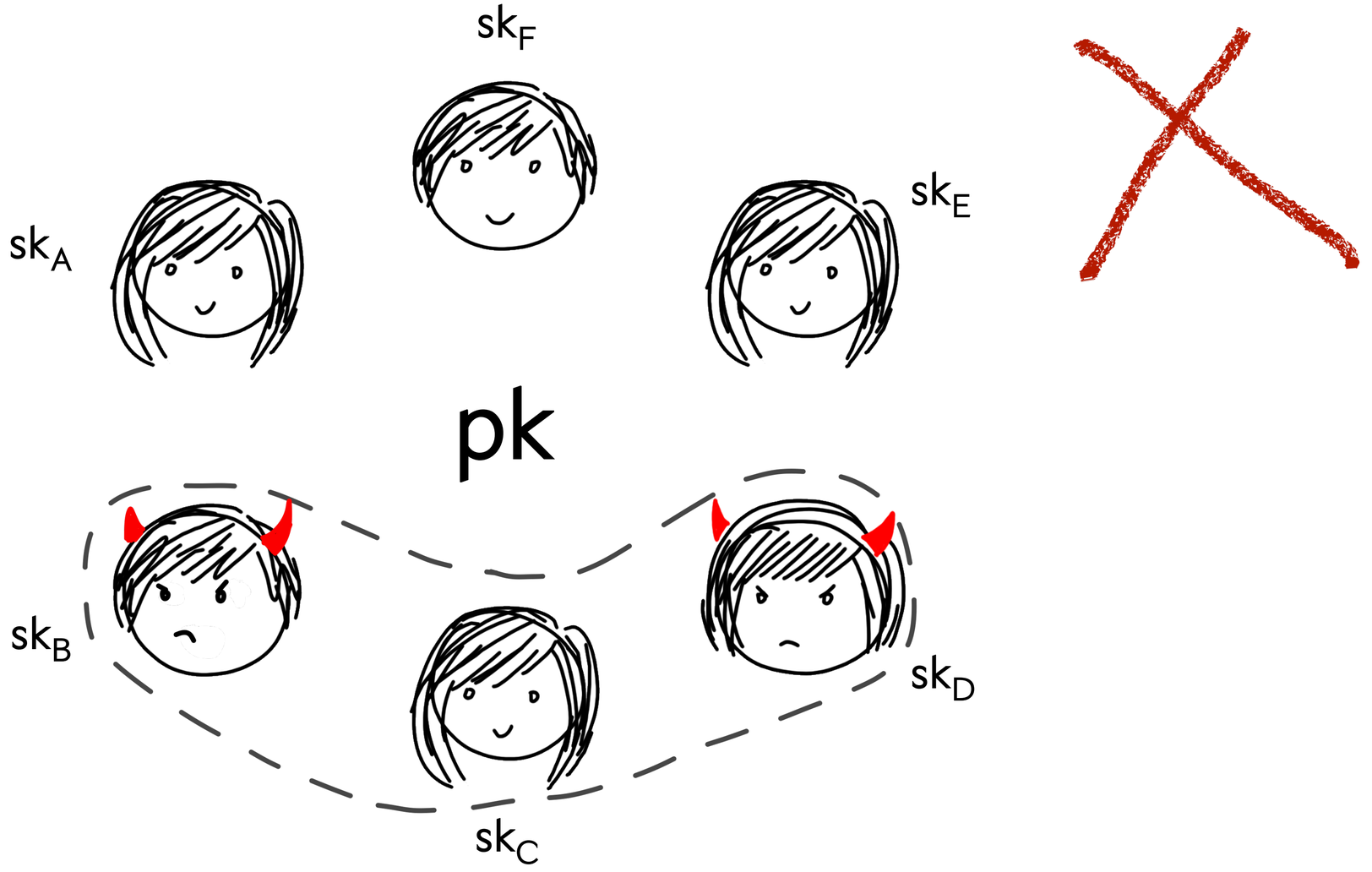
3-of-n Signature Scheme



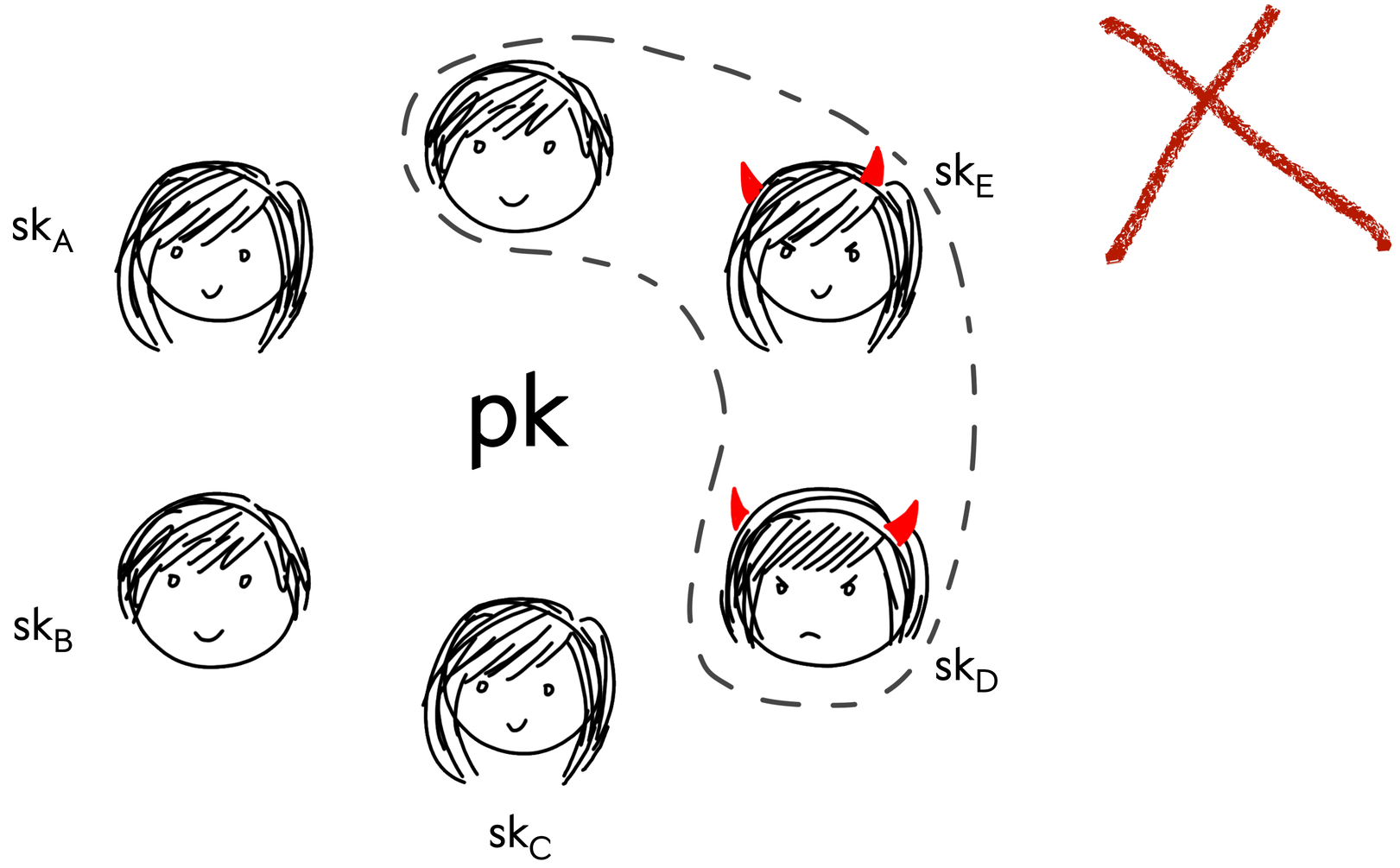
3-of-n Signature Scheme

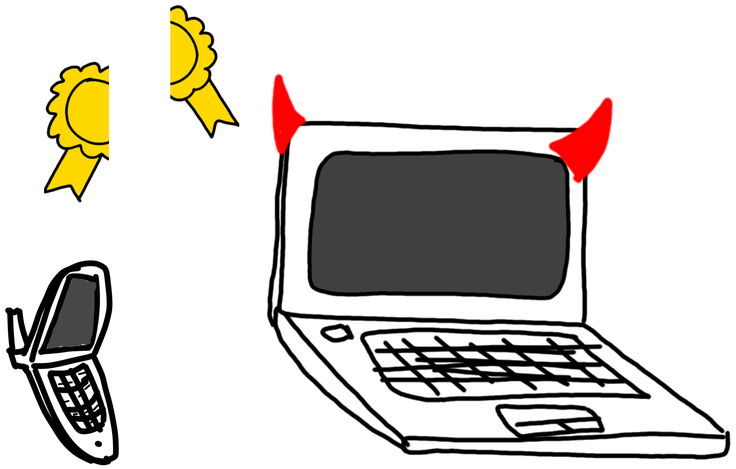


3-of-n Signature Scheme



3-of-n Signature Scheme





sk

sk′

sk′

sk′

sk

sk′ sk

sk′

sk′

sk

sk′ sk

sk′

sk′

sk′

sk′ sk

sk

sk′

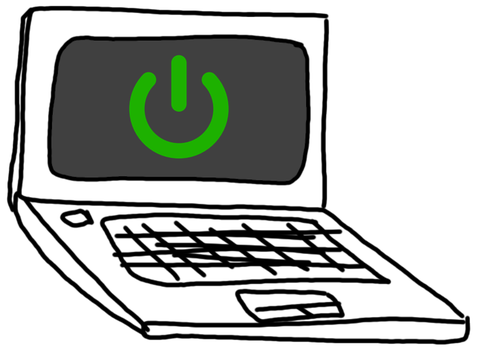
sk′ sk

sk′

sk

MON

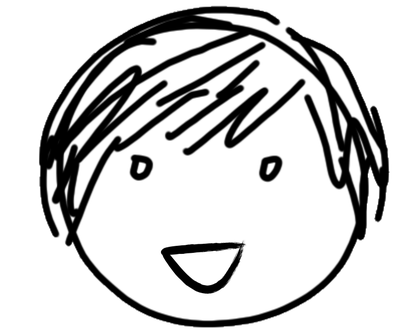
sk′



sk

TUE

sk′



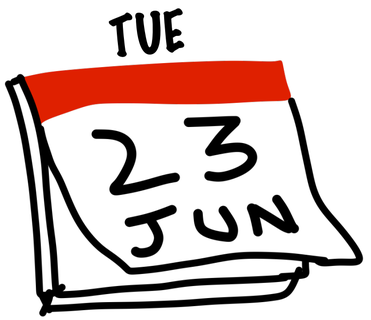
sk′ sk

sk′

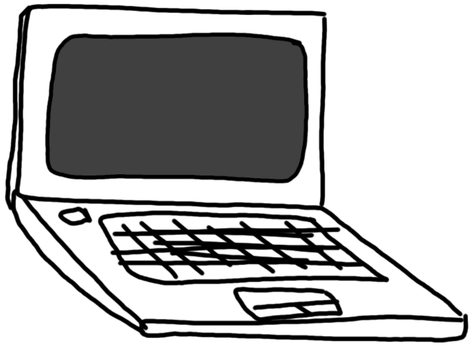
sk

sk′

sk



sk′



sk

sk′ sk

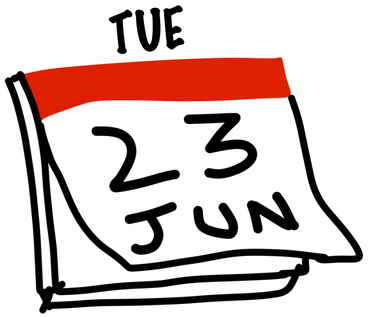
sk′ sk

sk′

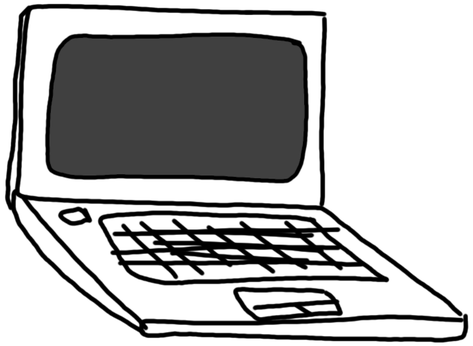
sk

sk′

sk



sk′



sk

sk′ sk

sk′ sk

sk′

sk

MON

sk′

sk

TUE

sk′

sk

WED

sk′ sk

sk′ sk

sk′

sk

MON

sk′

sk

TUE

sk′

sk

WED

sk′ sk

sk′ sk

sk′

sk

MON

sk′

sk

TUE

sk′

sk

WED

sk′

sk

sk'

sk'

sk

sk

sk'

sk

sk'

$sk + sk' = x$

sk

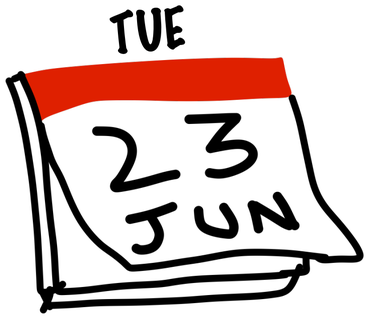
sk'

sk

sk'

$sk + sk' = x$

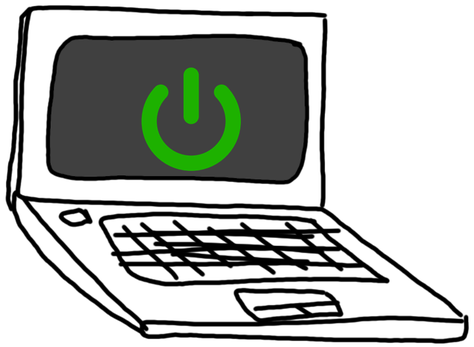
sk



sk'

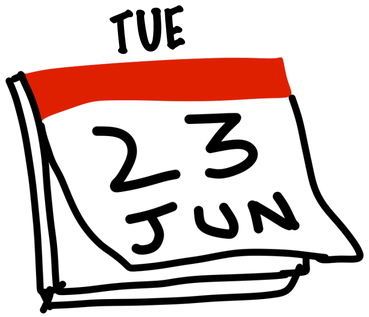
sk'

$sk + sk' = x$



sk

sk



sk'

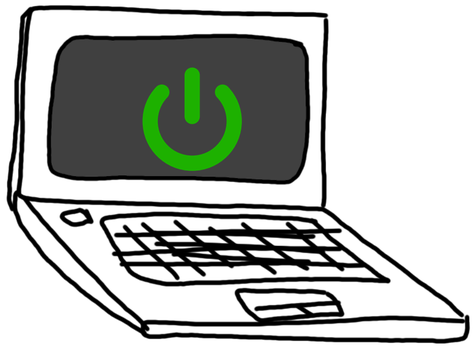
sk

sk'

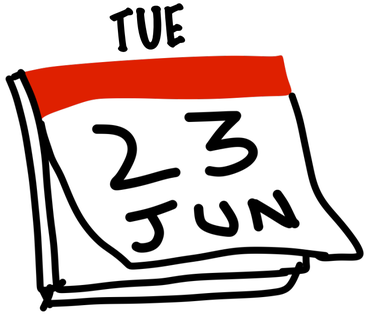
$sk + sk' = x$

sk

sk'_T



sk_T



sk'

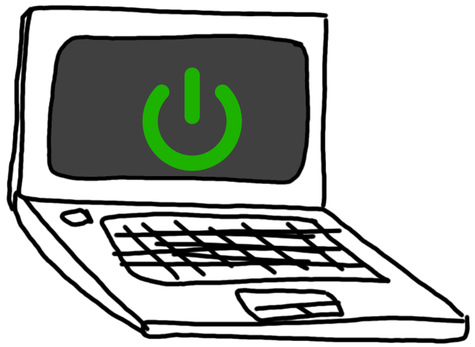
sk

sk'

$sk + sk' = x$

sk

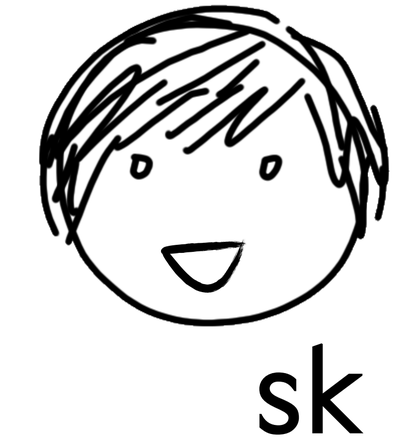
sk'_T



$sk_T + sk'_T = x$

sk_T

sk'



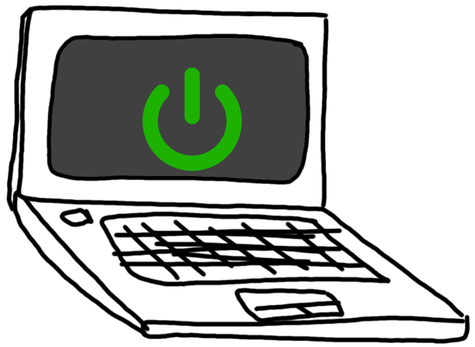
sk

sk'

$sk + sk' = x$

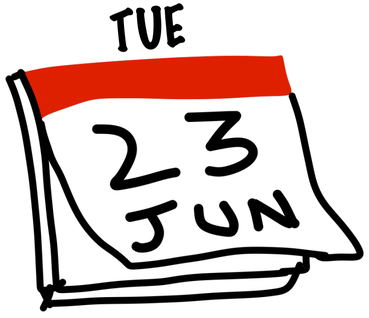
sk

sk'_T

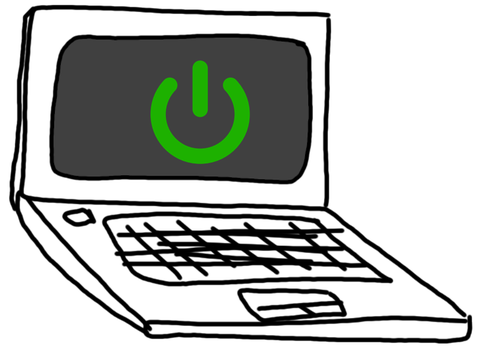


$sk_T + sk'_T = x$

sk_T



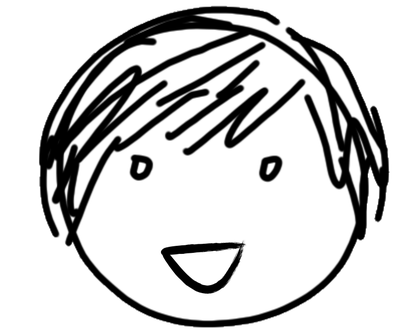
sk'_W



$sk'_W + sk_W = x$

sk_W

sk'



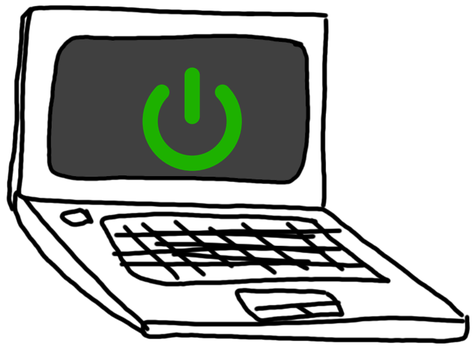
sk

sk'

$sk + sk' = x$

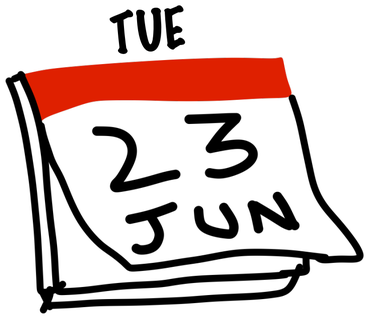
sk

sk'_T

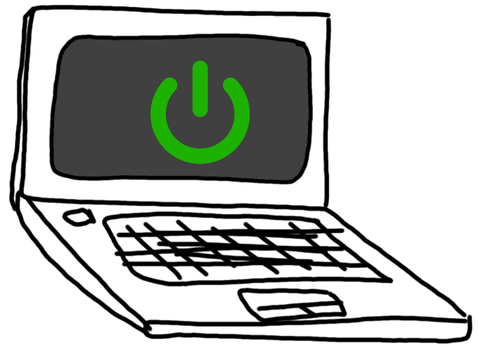


$sk_T + sk'_T = x$

sk_T



sk'_W



$sk'_W + sk_W = x$

sk_W

sk' sk_w

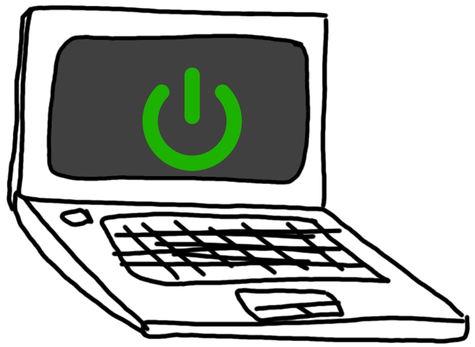
sk

sk'

$sk + sk' = x$

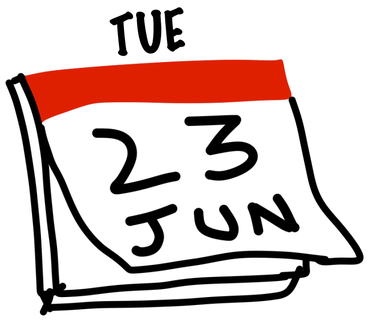
sk

sk'_T

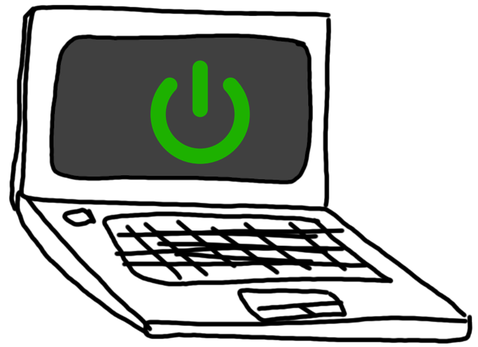


$sk_T + sk'_T = x$

sk_T



sk'_W



$sk'_W + sk_w = x$

sk_w

sk' sk_w

sk

$sk + sk' = x$

$sk_T + sk'_T = x$

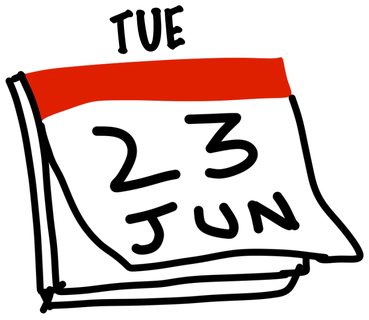
$sk'_W + sk_W = x$

sk' sk_w

sk

? + sk' = ?

? + sk_w = ?



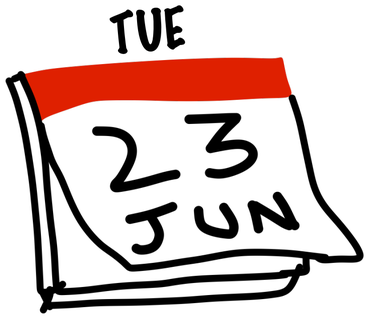
2 Equations in 3 variables

sk' sk_w

sk

? ? ?

? + sk' = ? ? + sk_w = ?



Proactive Security

Proactive Security

- Conceived by Ostrovsky & Yung (PODC ’91)

Proactive Security

- Conceived by Ostrovsky & Yung (PODC ’91)
- Many follow-ups for a variety of scenarios

Proactive Security

- Conceived by Ostrovsky & Yung (PODC ’91)
- Many follow-ups for a variety of scenarios
  - Asynchronous networks [CKLS02]

Proactive Security

- Conceived by Ostrovsky & Yung (PODC ’91)
- Many follow-ups for a variety of scenarios
  - Asynchronous networks [CKLS02]
  - Threshold signatures [HJJKY97, ADN06]

Proactive Security

- Conceived by Ostrovsky & Yung (PODC ’91)
- Many follow-ups for a variety of scenarios
  - Asynchronous networks [CKLS02]
  - Threshold signatures [HJJKY97, ADN06]
  - Dishonest majority [EOPY18, CMP20]

Proactive Security

- Conceived by Ostrovsky & Yung (PODC ’91)
- Many follow-ups for a variety of scenarios
  - Asynchronous networks [CKLS02]
  - Threshold signatures [HJJKY97, ADN06]
  - Dishonest majority [EOPY18, CMP20]
  - Dynamic committees [MZWLZJS19]

What Gap Do We Address?

What Gap Do We Address?

- In order to progress, in all prior works:

What Gap Do We Address?

- In order to progress, in all prior works:
  - Either **honest majority** must speak

What Gap Do We Address?

- In order to progress, in all prior works:
  - Either **honest majority** must speak
  - Or **everyone** **comes online**

What Gap Do We Address?

- In order to progress, in all prior works:
  - Either **honest majority** must speak
  - Or **everyone comes online**

} Not ideal for (*t*,*n*) wallets
➡ *t* to sign but 2*t* to refresh
➡ Inconvenient
➡ More correlated risk

What Gap Do We Address?

- In order to progress, in all prior works:
  - Either **honest majority** must speak
  - Or **everyone comes online**

} Not ideal for (*t*,*n*) wallets
➡ *t* to sign but 2*t* to refresh
➡ Inconvenient
➡ More correlated risk

- We study: refresh where **dishonest majority speaks**

What Gap Do We Address?

- In order to progress, in all prior works:
  - Either **honest majority** must speak
  - Or **everyone comes online**

} Not ideal for (*t*,*n*) wallets
➡ *t* to sign but 2*t* to refresh
➡ Inconvenient
➡ More correlated risk

- We study: refresh where **dishonest majority speaks**
  - Correct definition is subtle

What Gap Do We Address?

- In order to progress, in all prior works:
  - Either **honest majority** must speak
  - Or **everyone comes online**

} Not ideal for (*t*,*n*) wallets
➡ *t* to sign but 2*t* to refresh
➡ Inconvenient
➡ More correlated risk

- We study: refresh where **dishonest majority speaks**
  - Correct definition is subtle
  - (*2*,*n*) setting: Efficient new protocol **native** to wallets

What Gap Do We Address?

- In order to progress, in all prior works:
  - Either **honest majority** must speak
  - Or **everyone** **comes online**

} Not ideal for (*t*,*n*) wallets
➡ *t* to sign but 2*t* to refresh
➡ Inconvenient
➡ More correlated risk

- We study: refresh where **dishonest majority** **speaks**
  - Correct definition is subtle
  - (*2*,*n*) setting: Efficient new protocol **native** to wallets
  - (*t*,*n*) setting: **Impossible**!

This Work

- Correct definition is subtle

- (*2*,*n*) setting: Efficient new protocol **native** to wallets

- (*t*,*n*) setting: **Impossible**!

This Work

- Correct definition is subtle

- (2,n) setting: Efficient new protocol **native** to wallets

- (t,n) setting: **Impossible**!

This Work

- Correct definition is subtle
  - Guaranteed progress is impossible

- (2,n) setting: Efficient new protocol **native** to wallets

- (t,n) setting: **Impossible**!

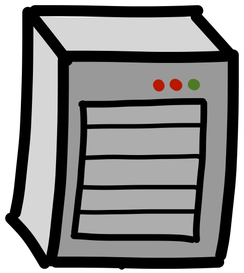
This Work

- Correct definition is subtle
  - Guaranteed progress is impossible

- (2,n) setting: Efficient new protocol **native** to wallets

- (t,n) setting: **Impossible**!

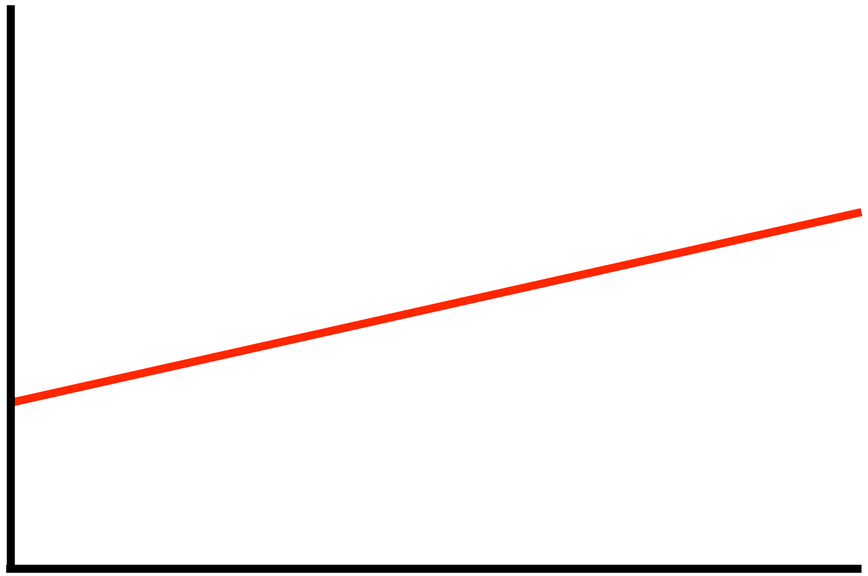
Defining Offline Refresh

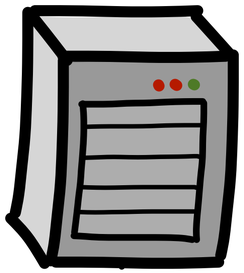


Defining Offline Refresh

$$f \leftarrow \mathbb{Z}_q[x]$$

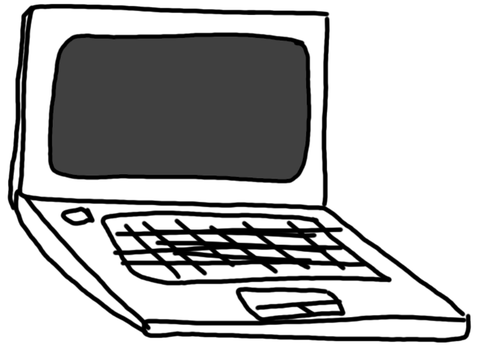
Degree 1



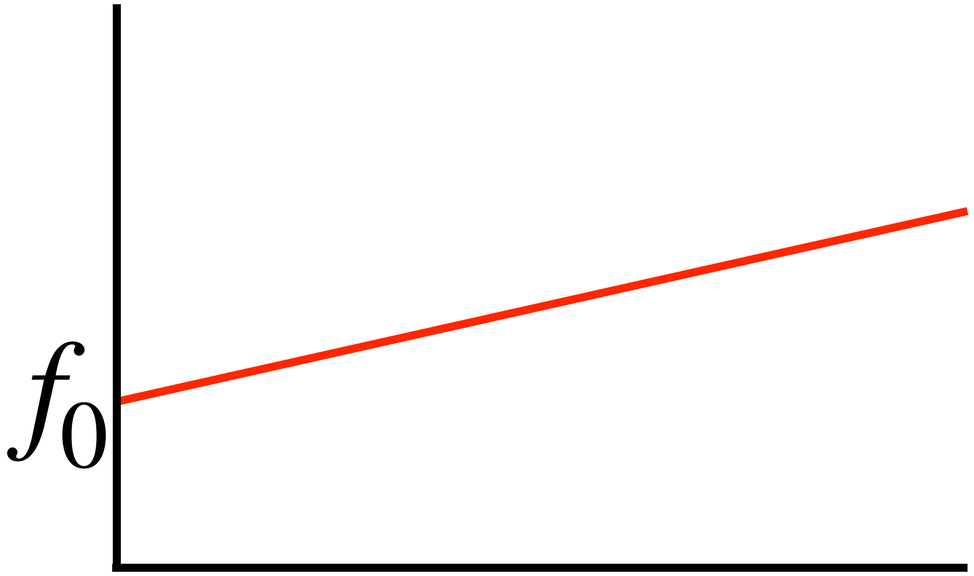


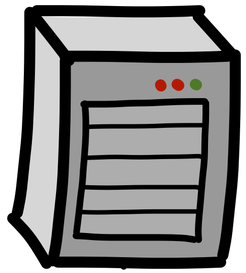
Defining Offline Refresh

$f \leftarrow \mathbb{Z}_q[x]$



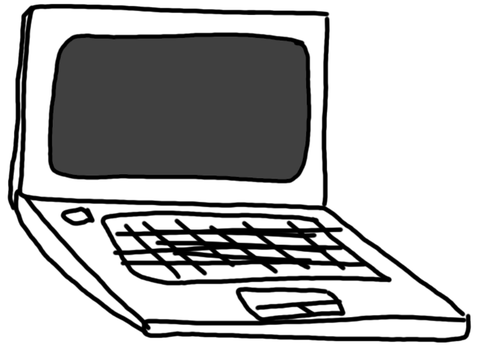
Degree 1



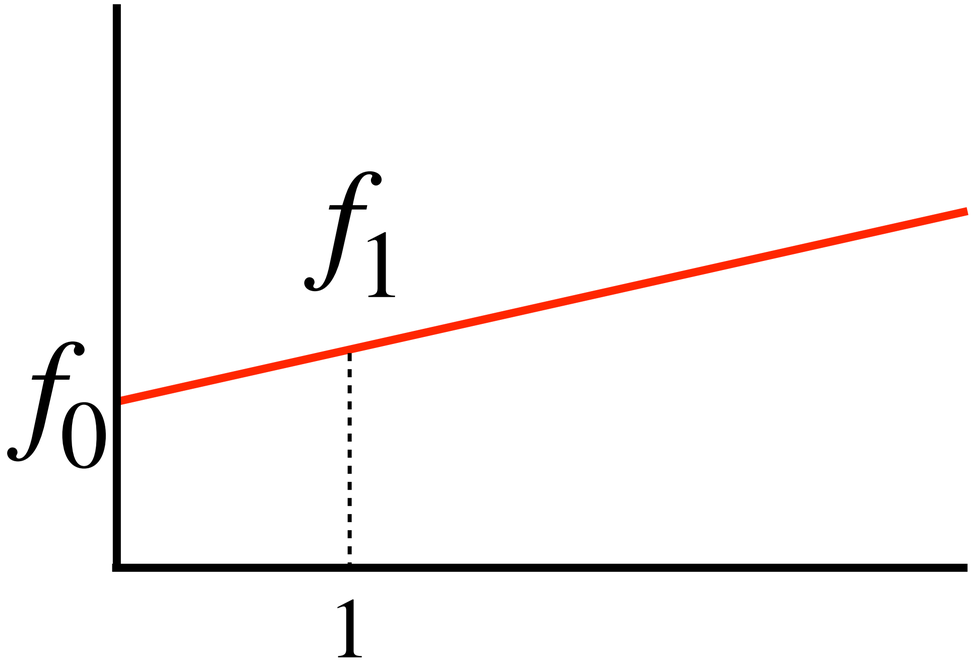


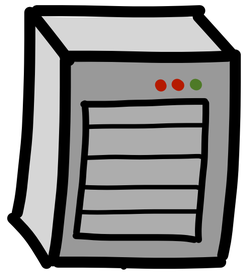
Defining Offline Refresh

$$f \leftarrow \mathbb{Z}_q[x]$$



Degree 1

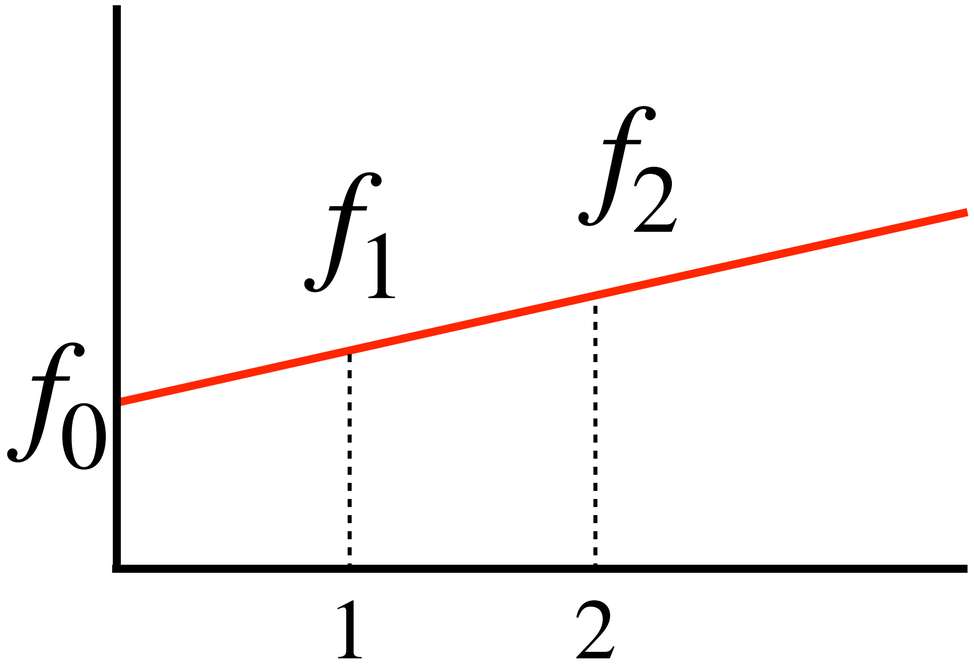


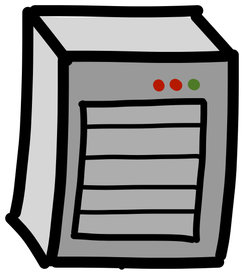


Defining Offline Refresh

$$f \leftarrow \mathbb{Z}_q[x]$$

Degree 1

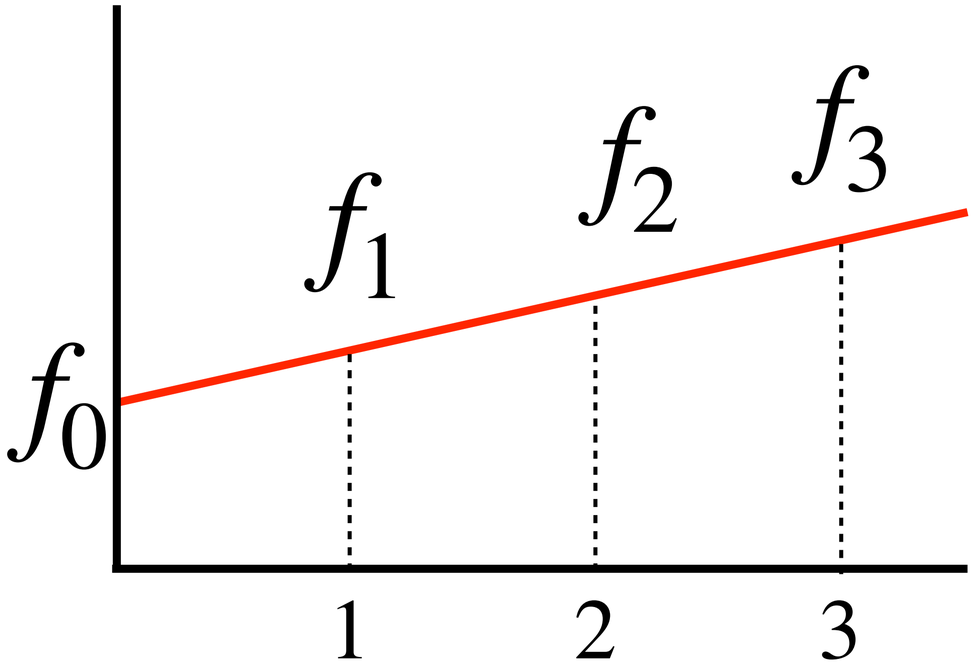


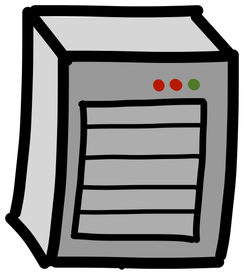


Defining Offline Refresh

$$f \leftarrow \mathbb{Z}_q[x]$$

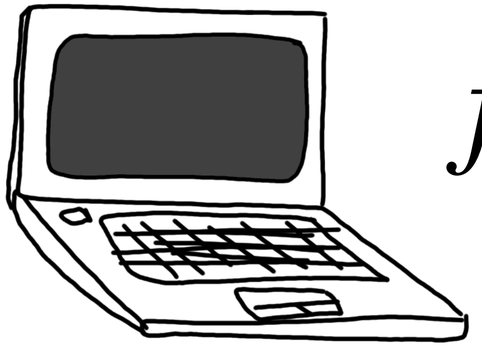
Degree 1



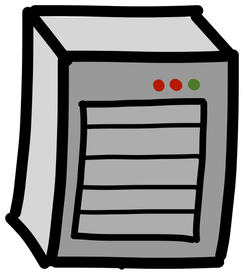


Defining Offline Refresh

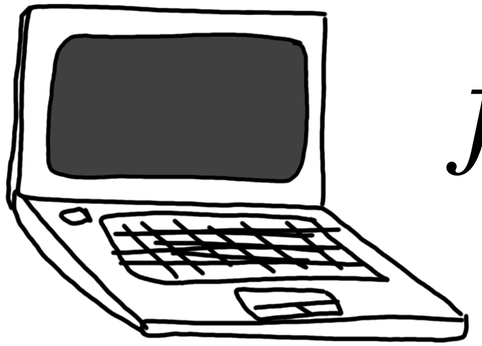
f_1 

 f_2

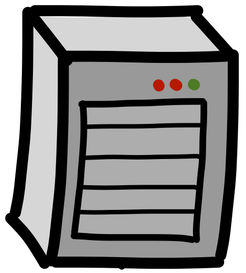
f_0

 f_3

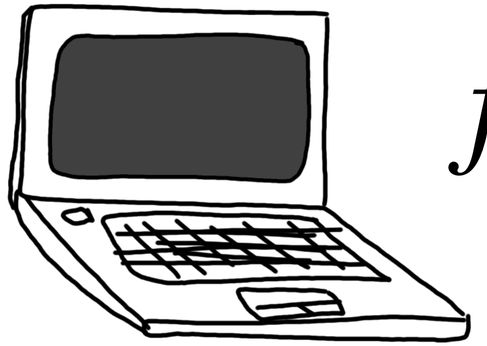
Defining Offline Refresh

f_1   f_2

$\mathsf{sk} = f_0$

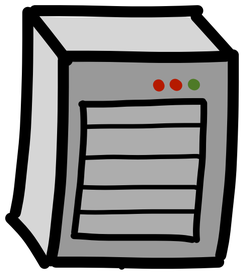
 f_3

Defining Offline Refresh

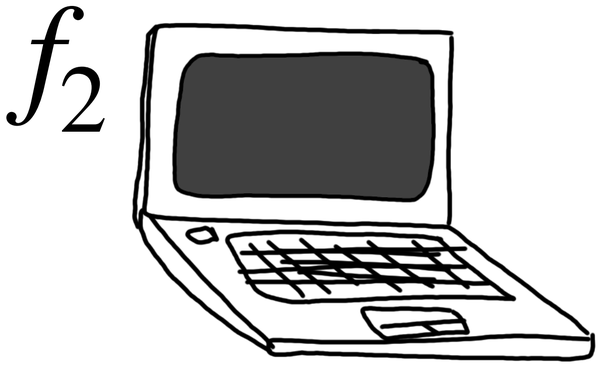
f_1   f_2

$$\mathsf{sk} = f_0$$

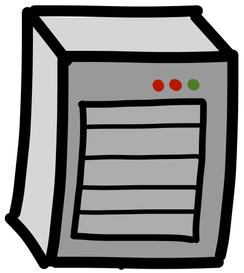
Enough information to:
- Sign with two online
- Recover from a crash

 f_3

Defining Offline Refresh

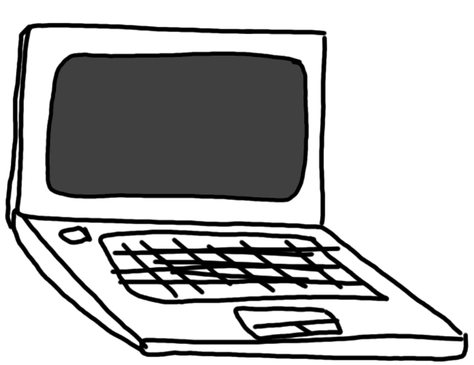


$\mathsf{sk} = f_0$



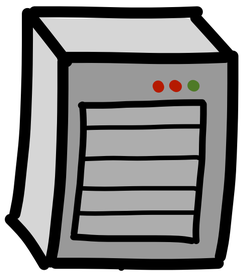
Defining Offline Refresh

f_1



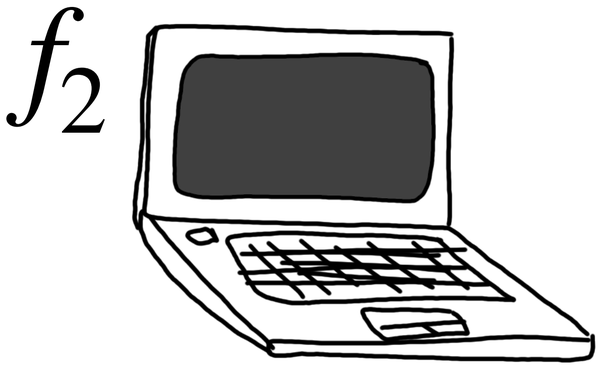
f_2

$\mathsf{sk} = f_0$

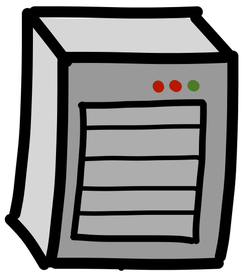


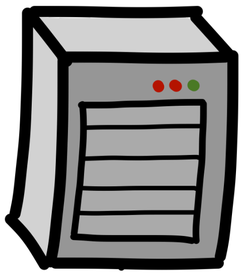
f_3

Defining Offline Refresh

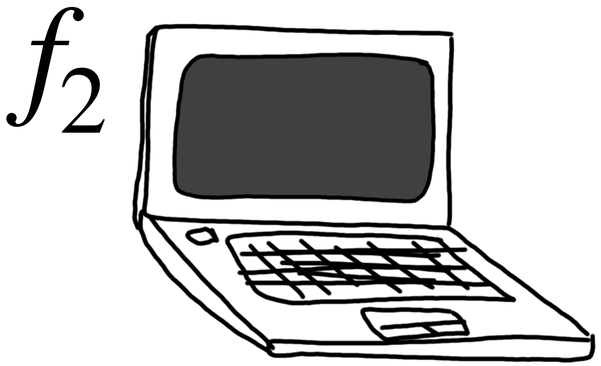


$\mathsf{sk} = f_0$

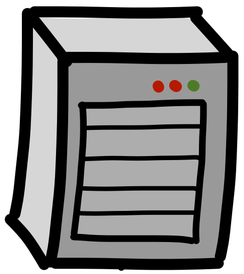


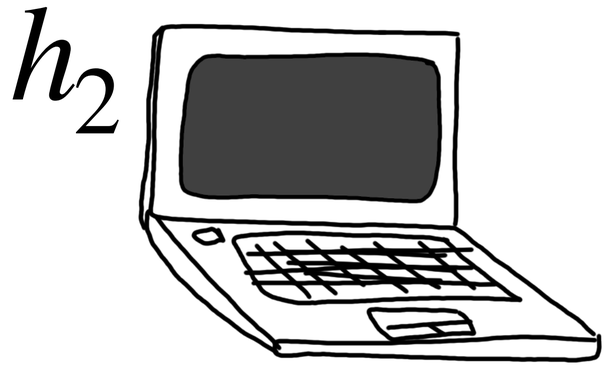


Defining Offline Refresh

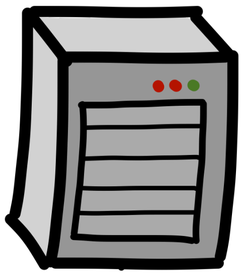


$$\mathsf{sk} = f_0$$





$$\mathsf{sk} = h_0 = f_0$$

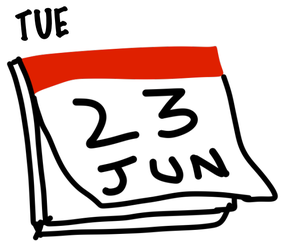


Defining Offline Refresh

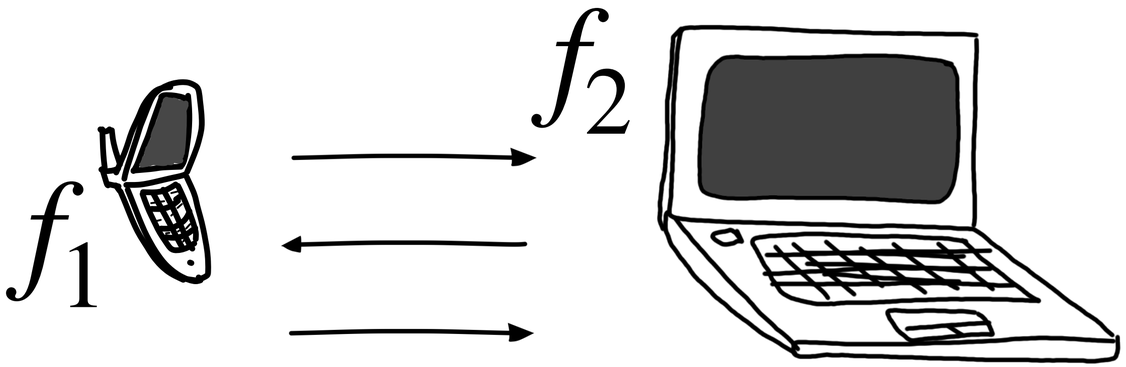
f_1

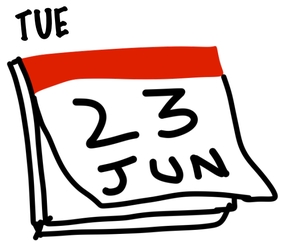
f_2

f_3

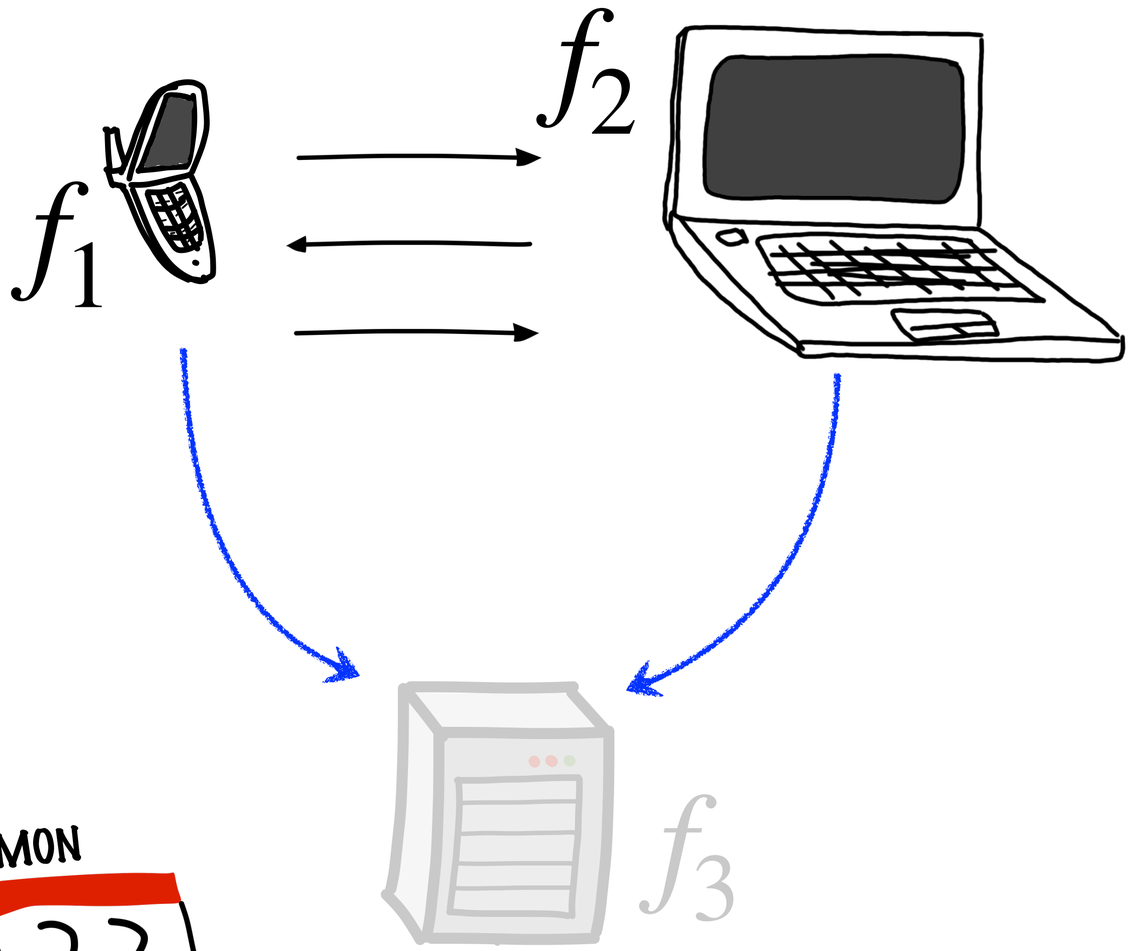


Defining Offline Refresh

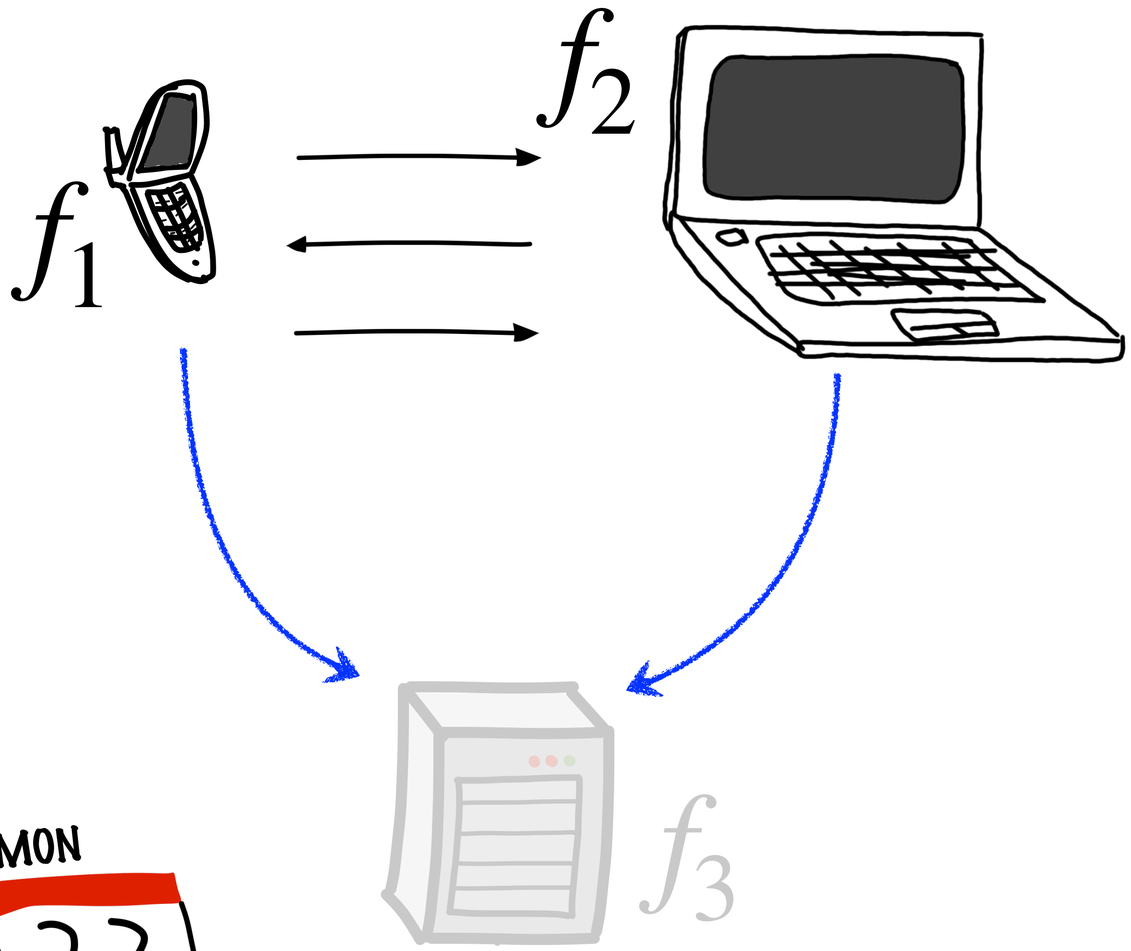


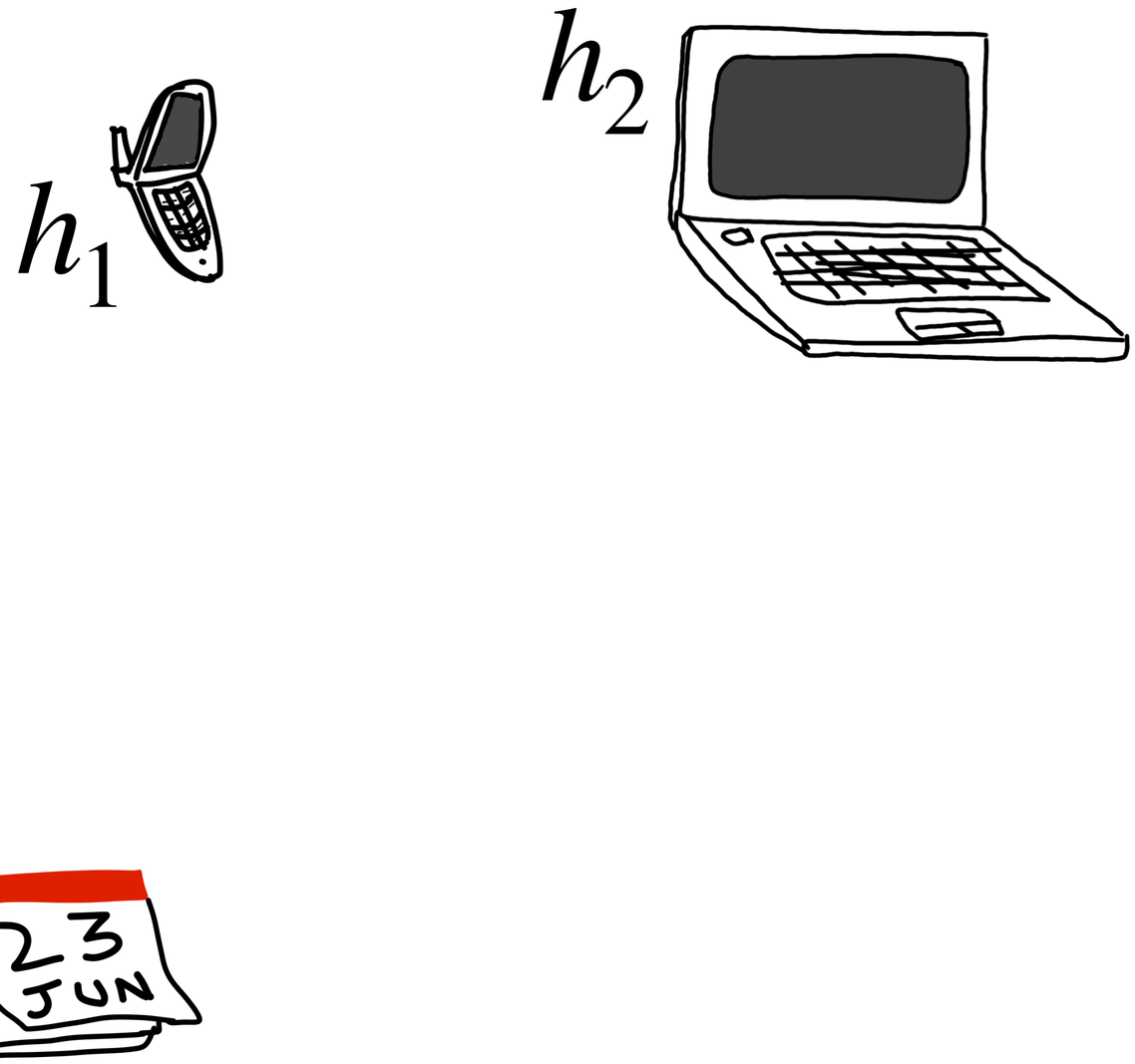


Defining Offline Refresh

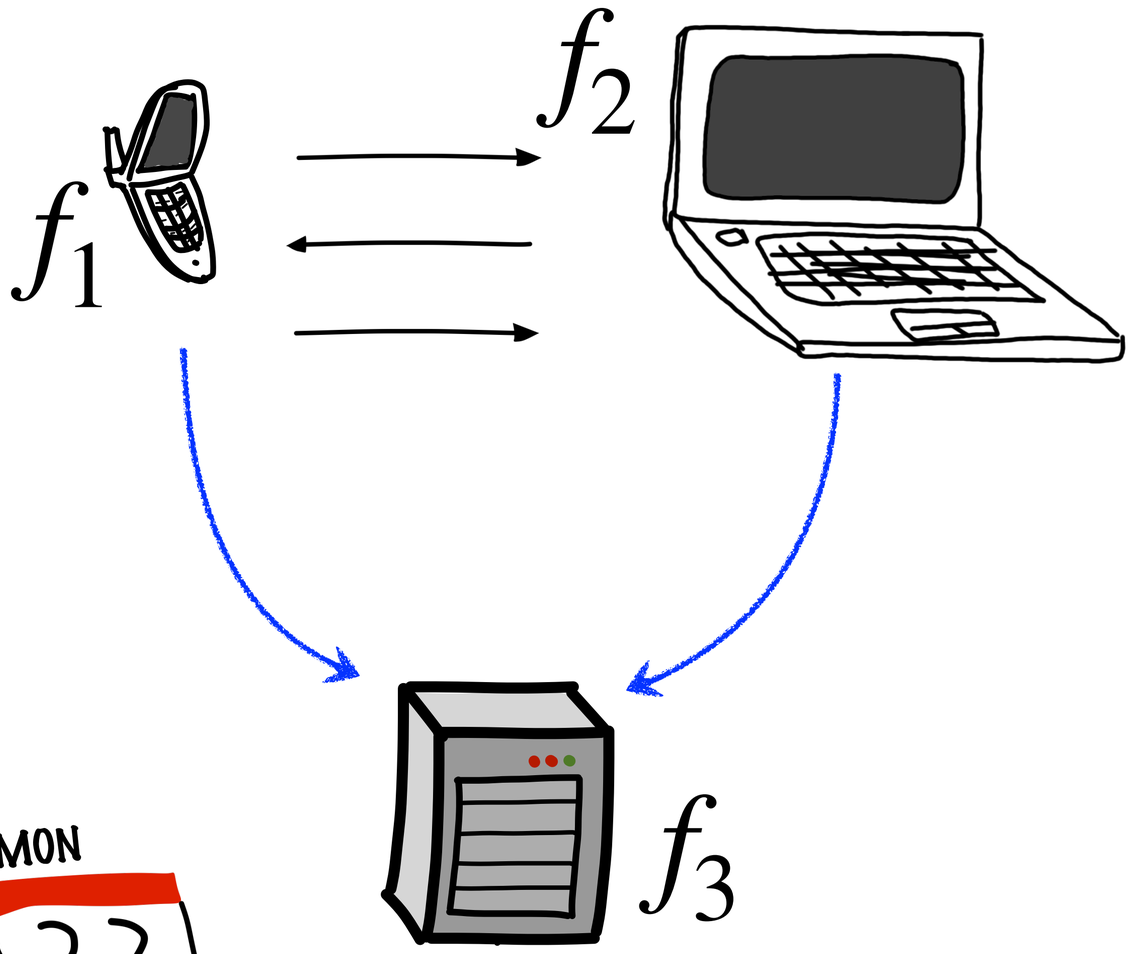


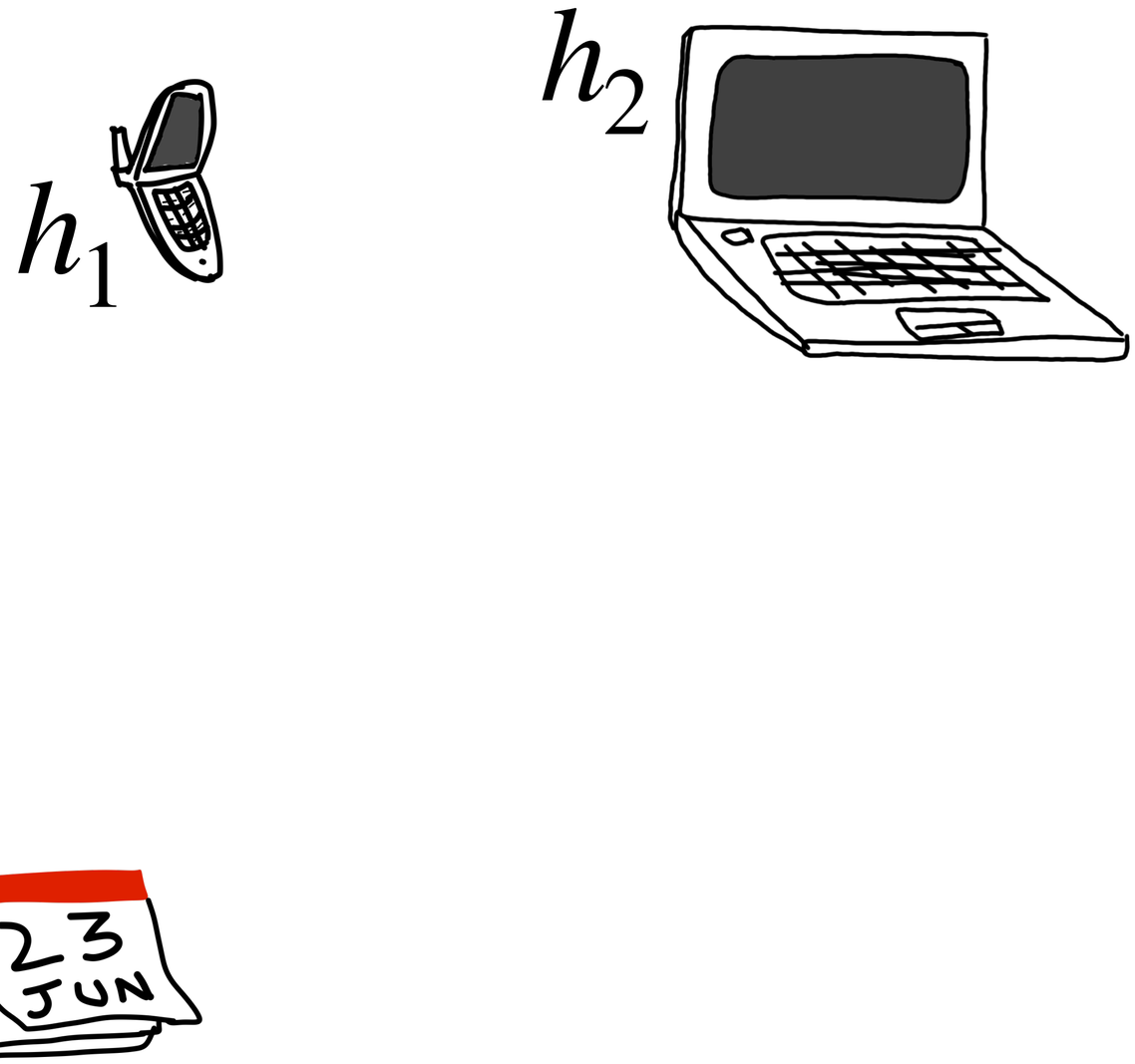
Defining Offline Refresh



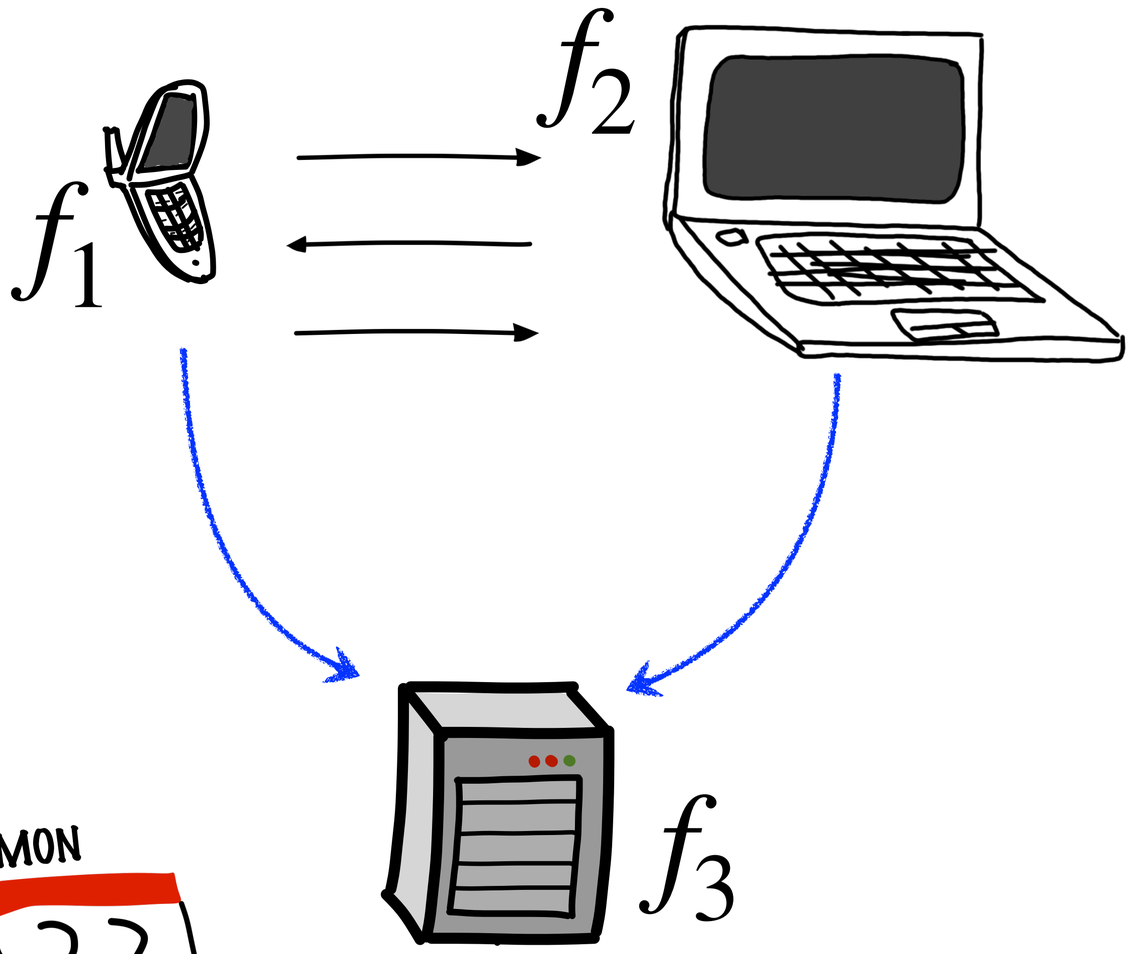


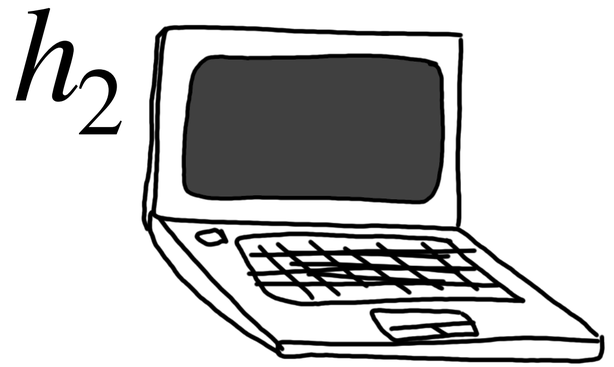
Defining Offline Refresh



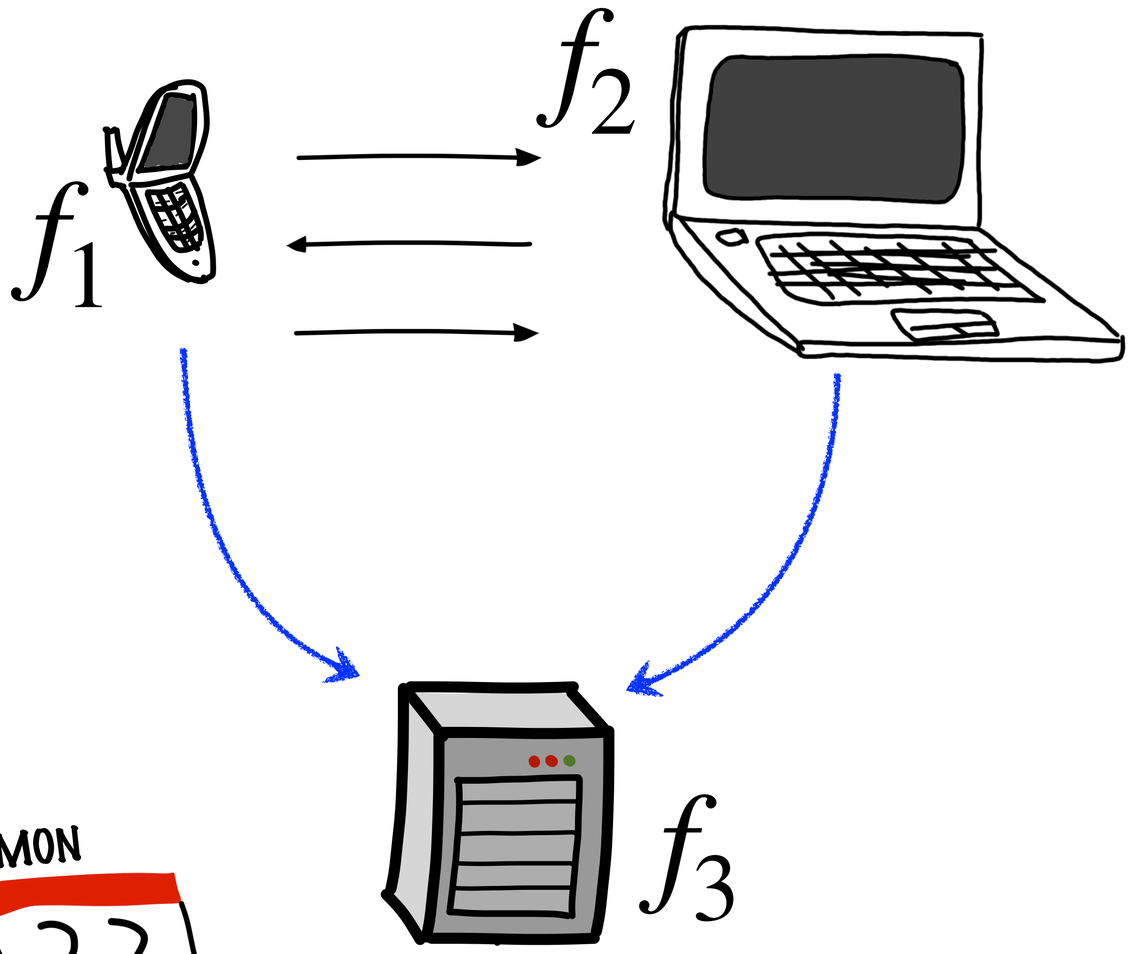


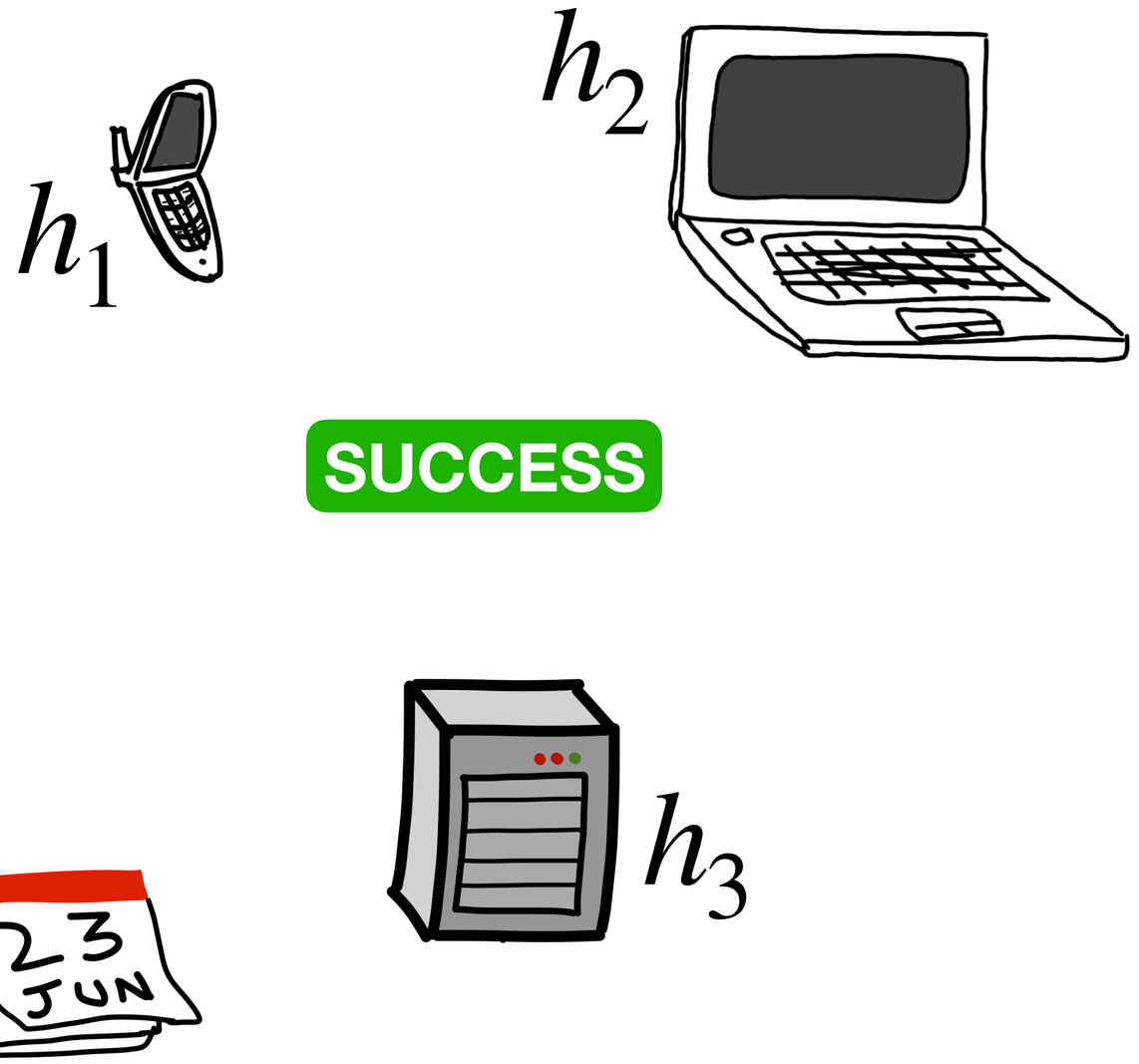
Defining Offline Refresh





Defining Offline Refresh



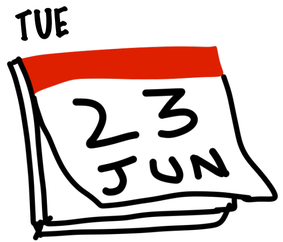


Defining Offline Refresh

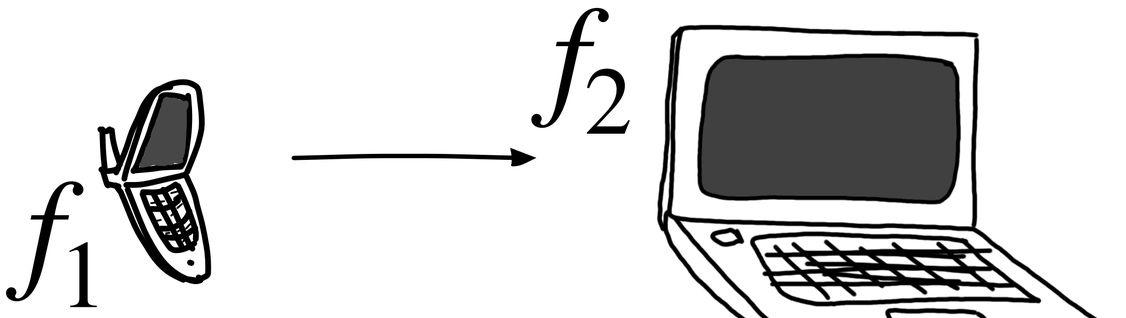
f_1

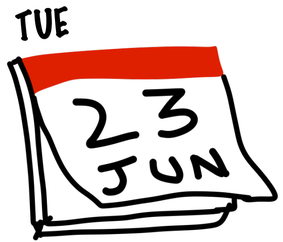
f_2

f_3

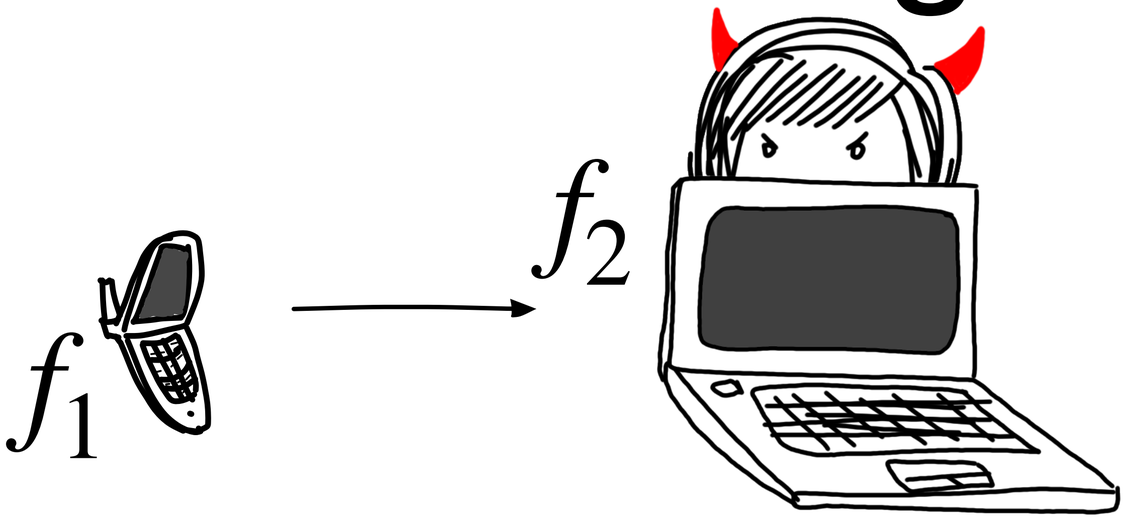


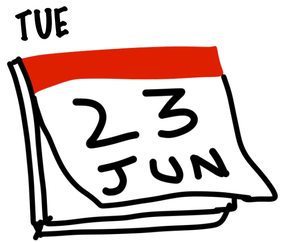
Defining Offline Refresh



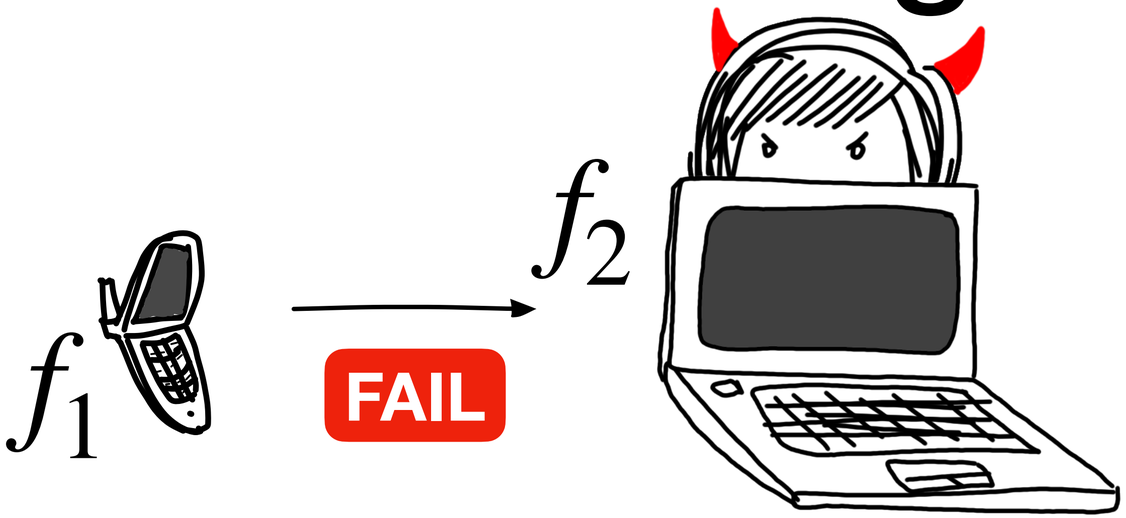


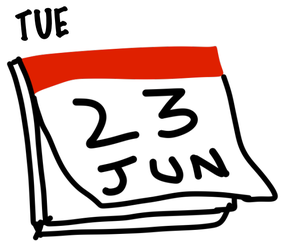
Defining Offline Refresh



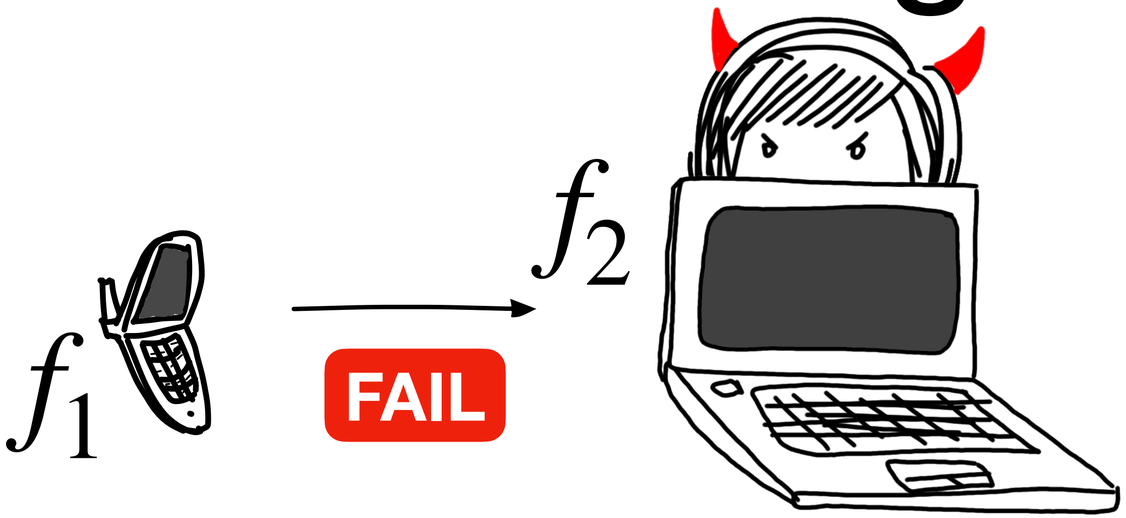


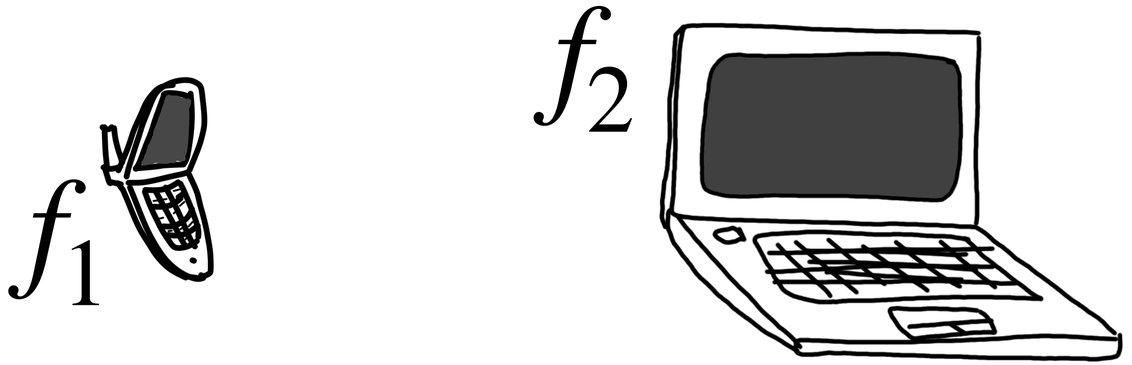
Defining Offline Refresh

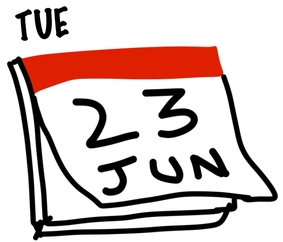




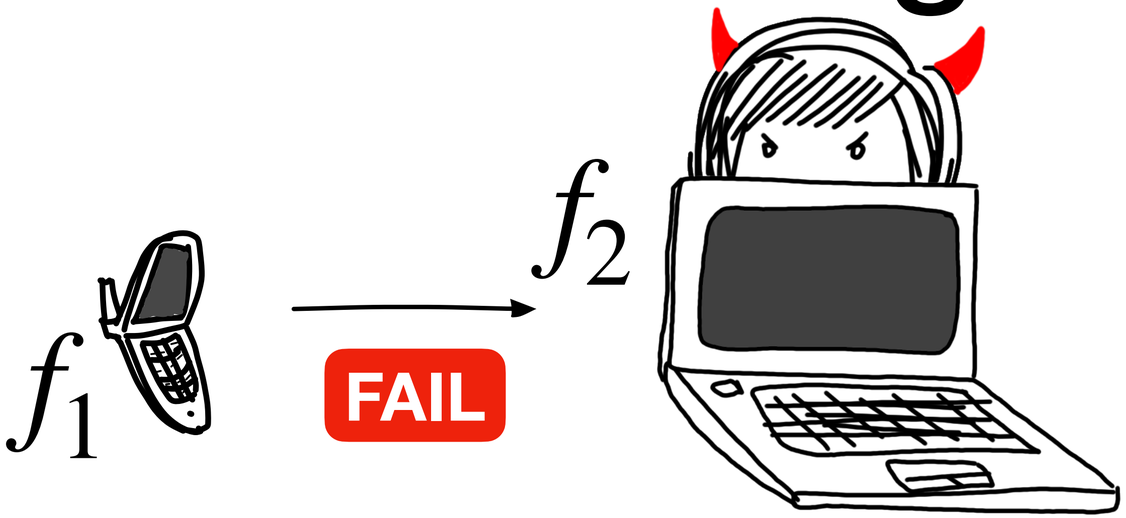
Defining Offline Refresh

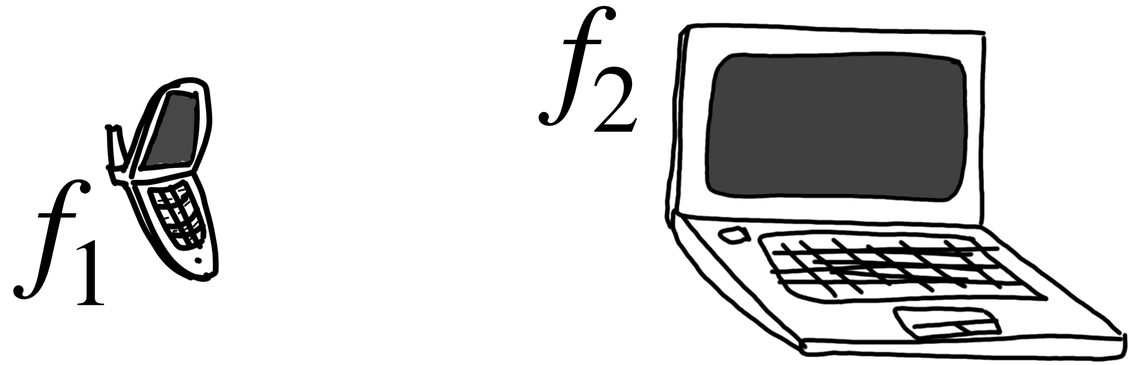


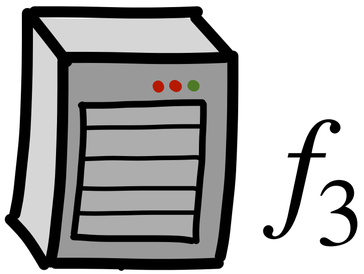


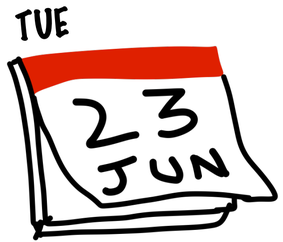


Defining Offline Refresh

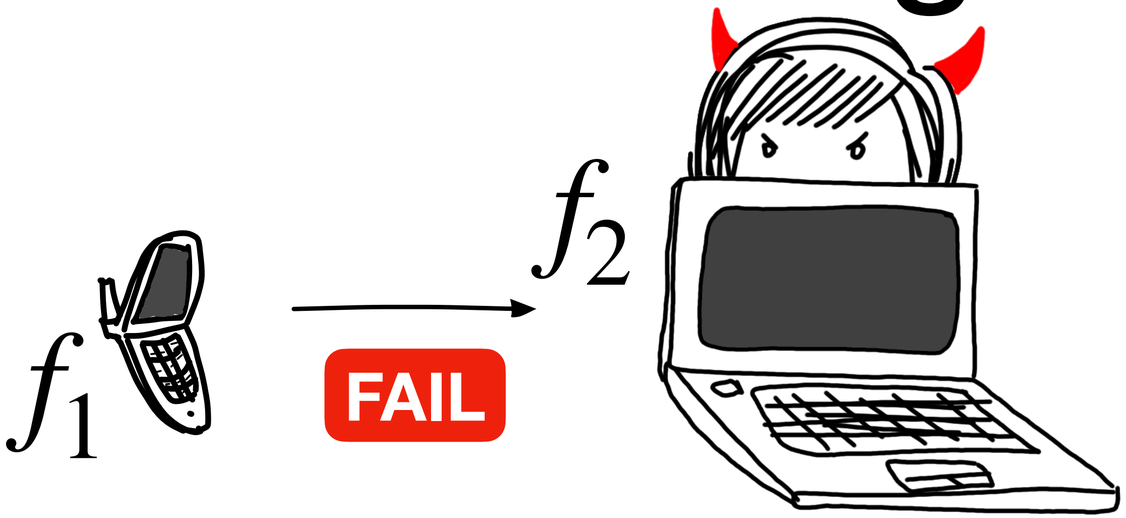


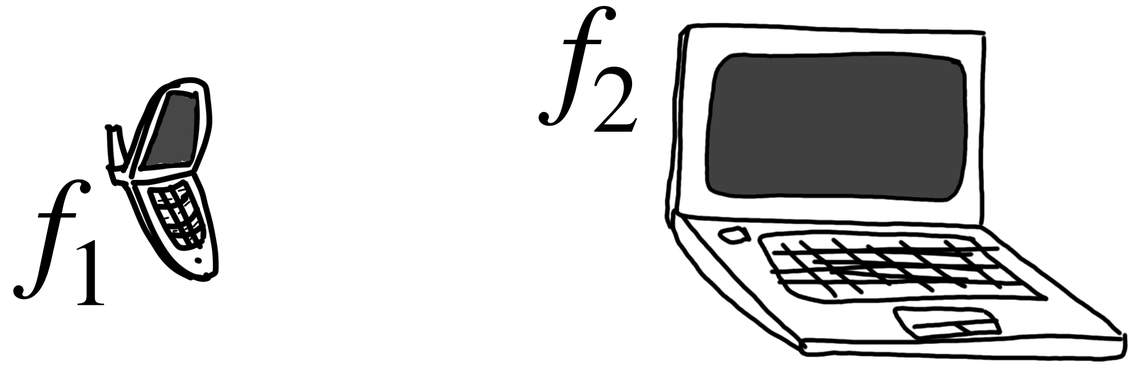


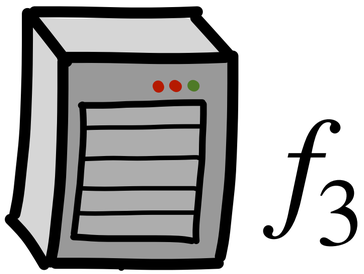


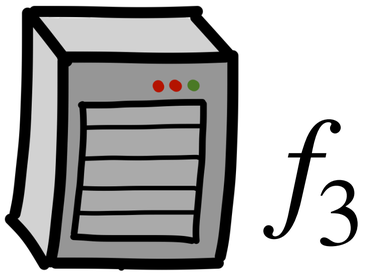


Defining Offline Refresh







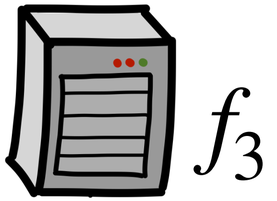


Defining Offline Refresh

Defining Offline Refresh

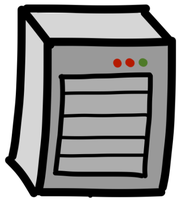
SUCCESS

FAIL



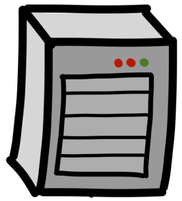
Defining Offline Refresh

SUCCESS



HMMMMMM

FAIL

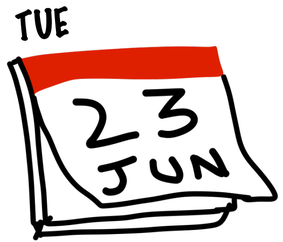


Defining Offline Refresh

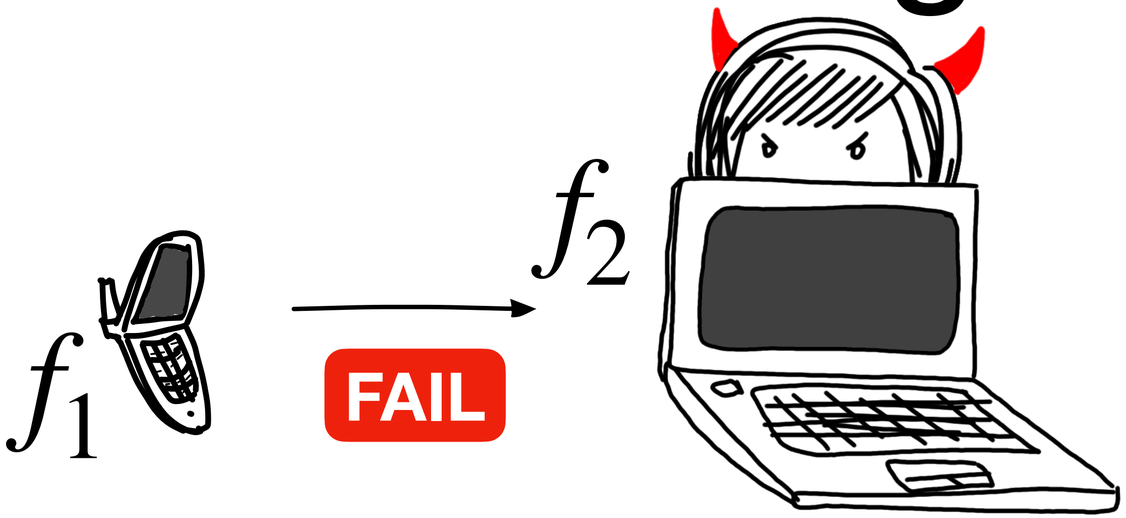
f_1

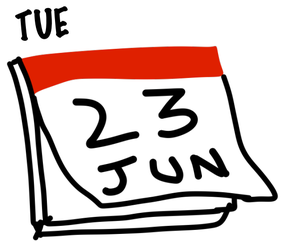
f_2

f_3

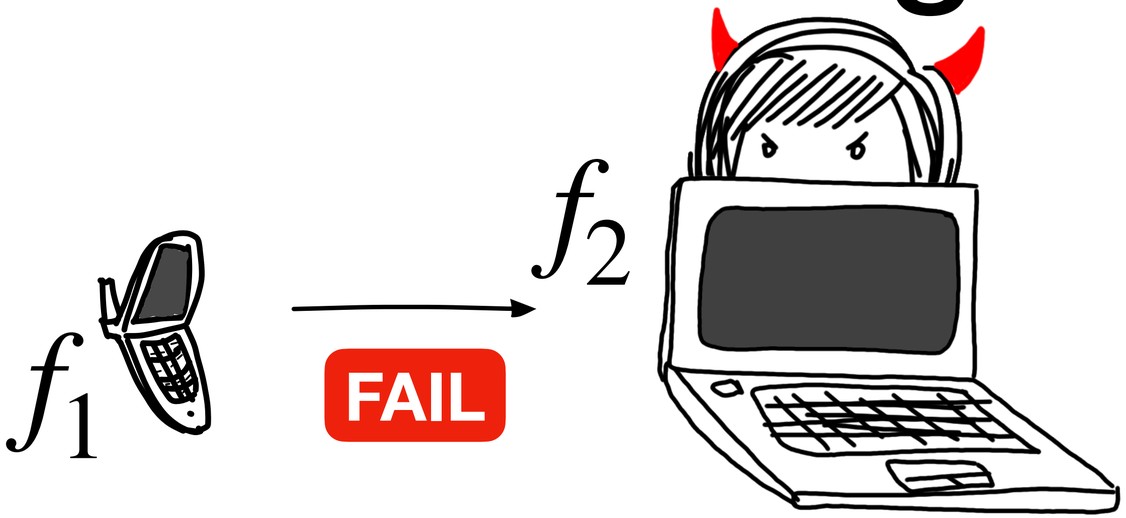


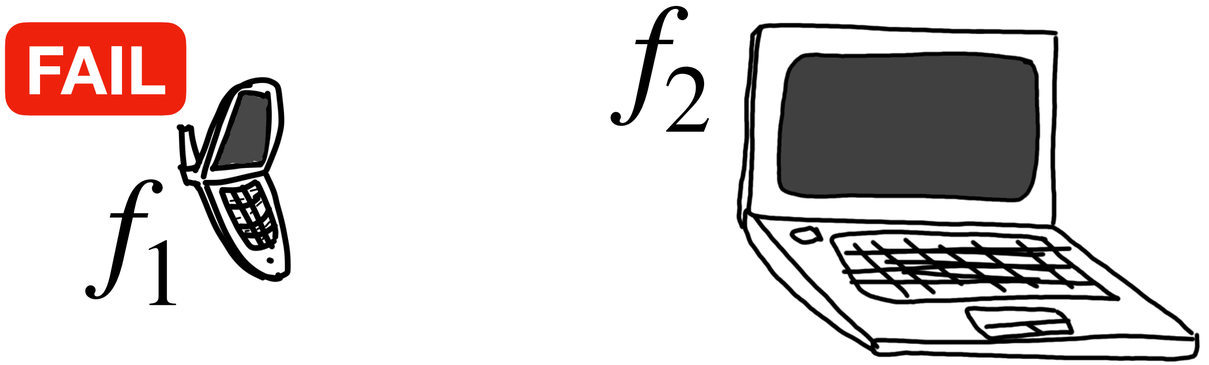
Defining Offline Refresh

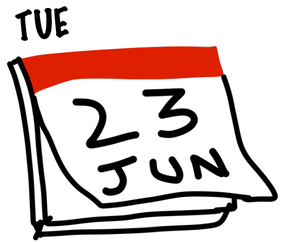




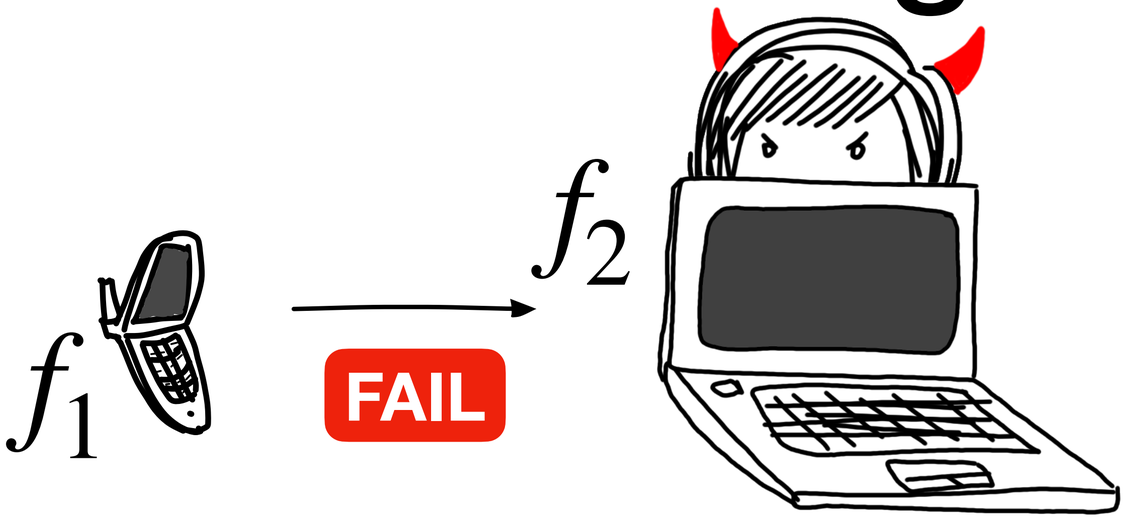
Defining Offline Refresh

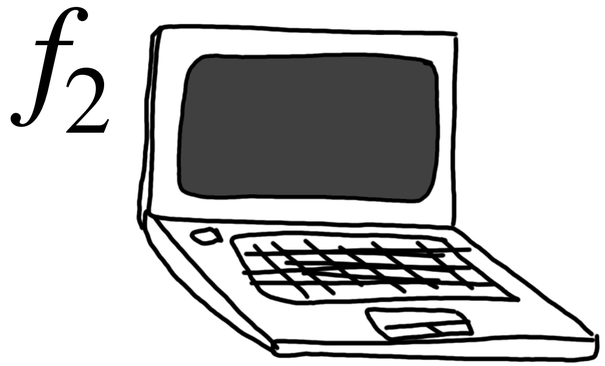


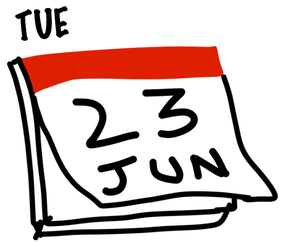




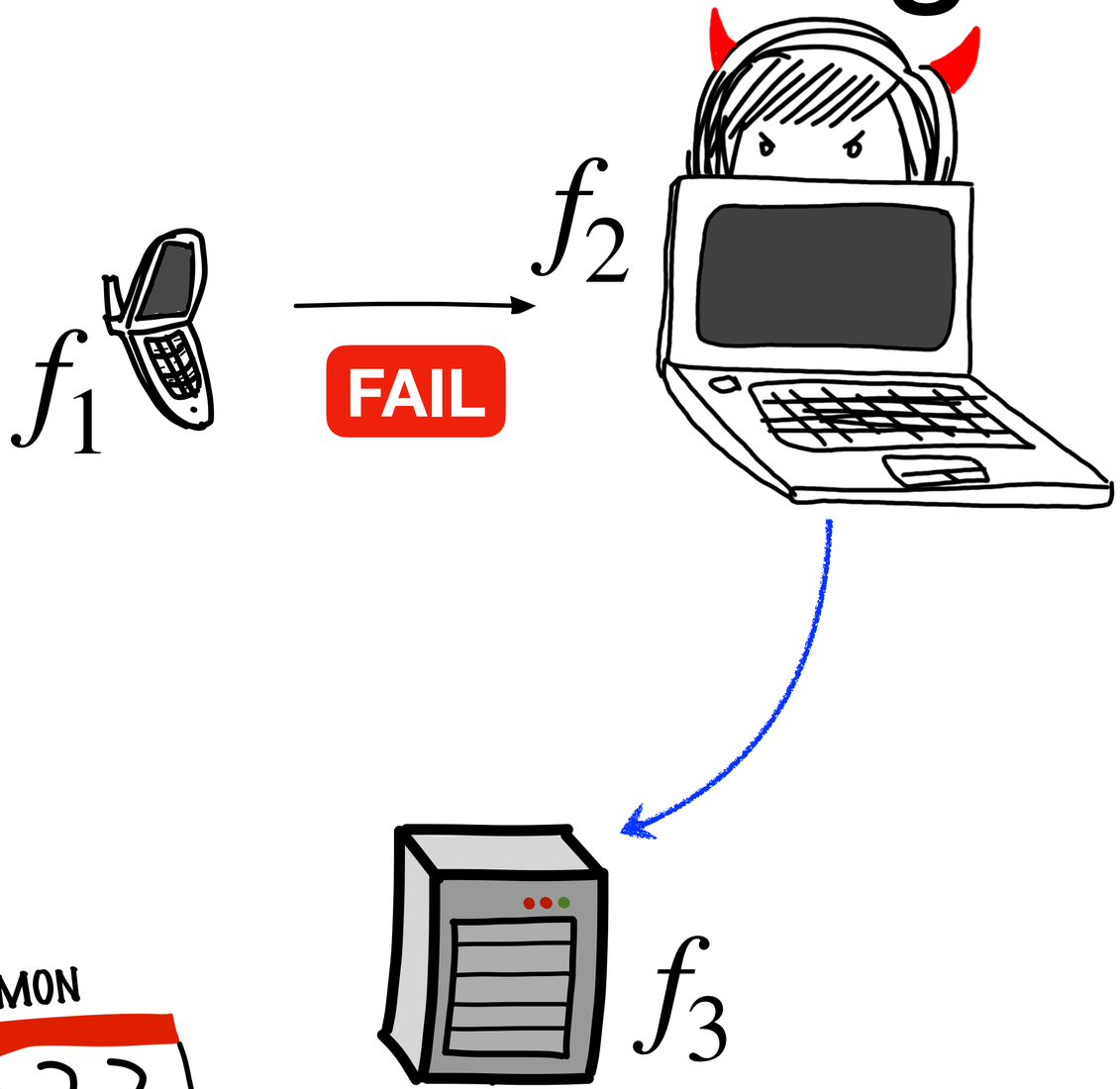
Defining Offline Refresh



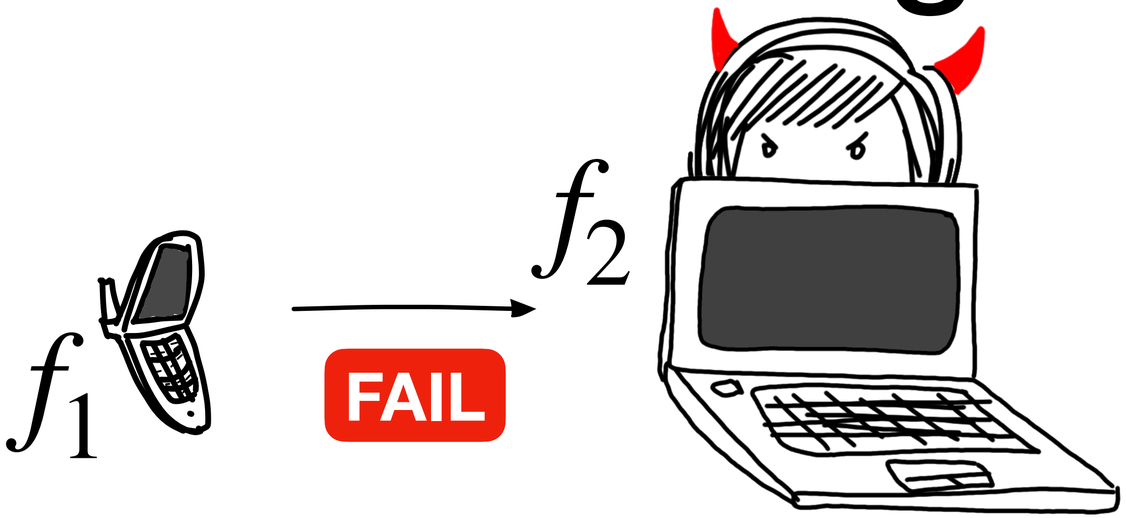


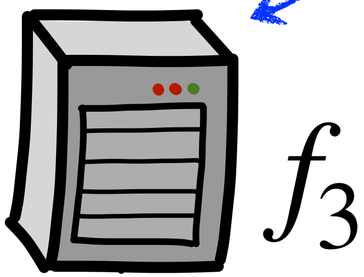


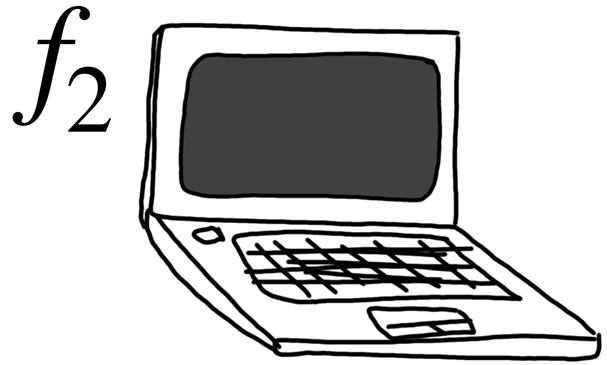
Defining Offline Refresh

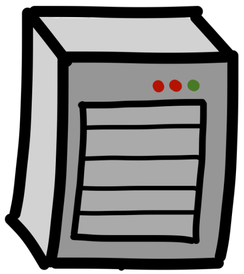


Defining Offline Refresh



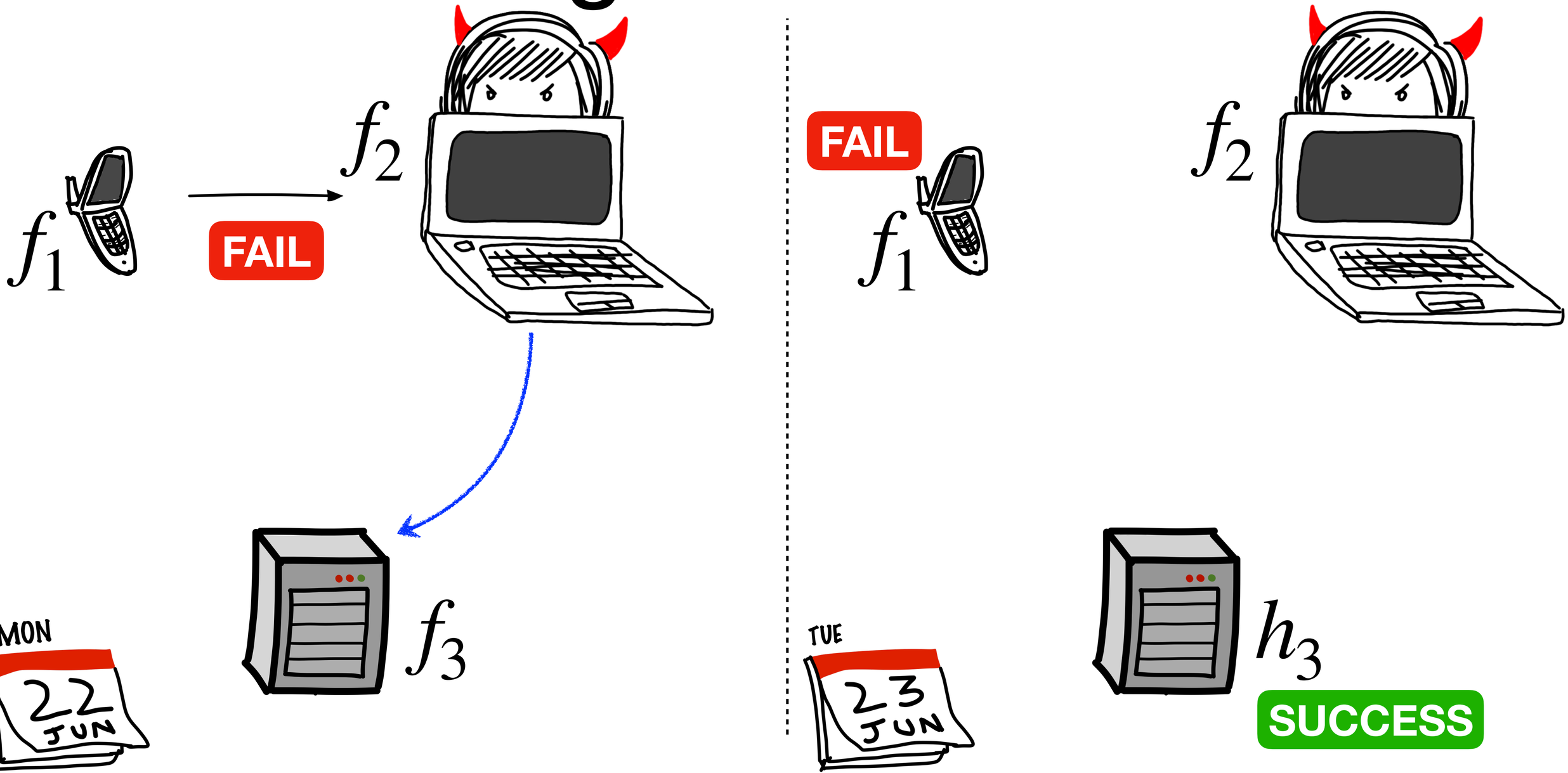




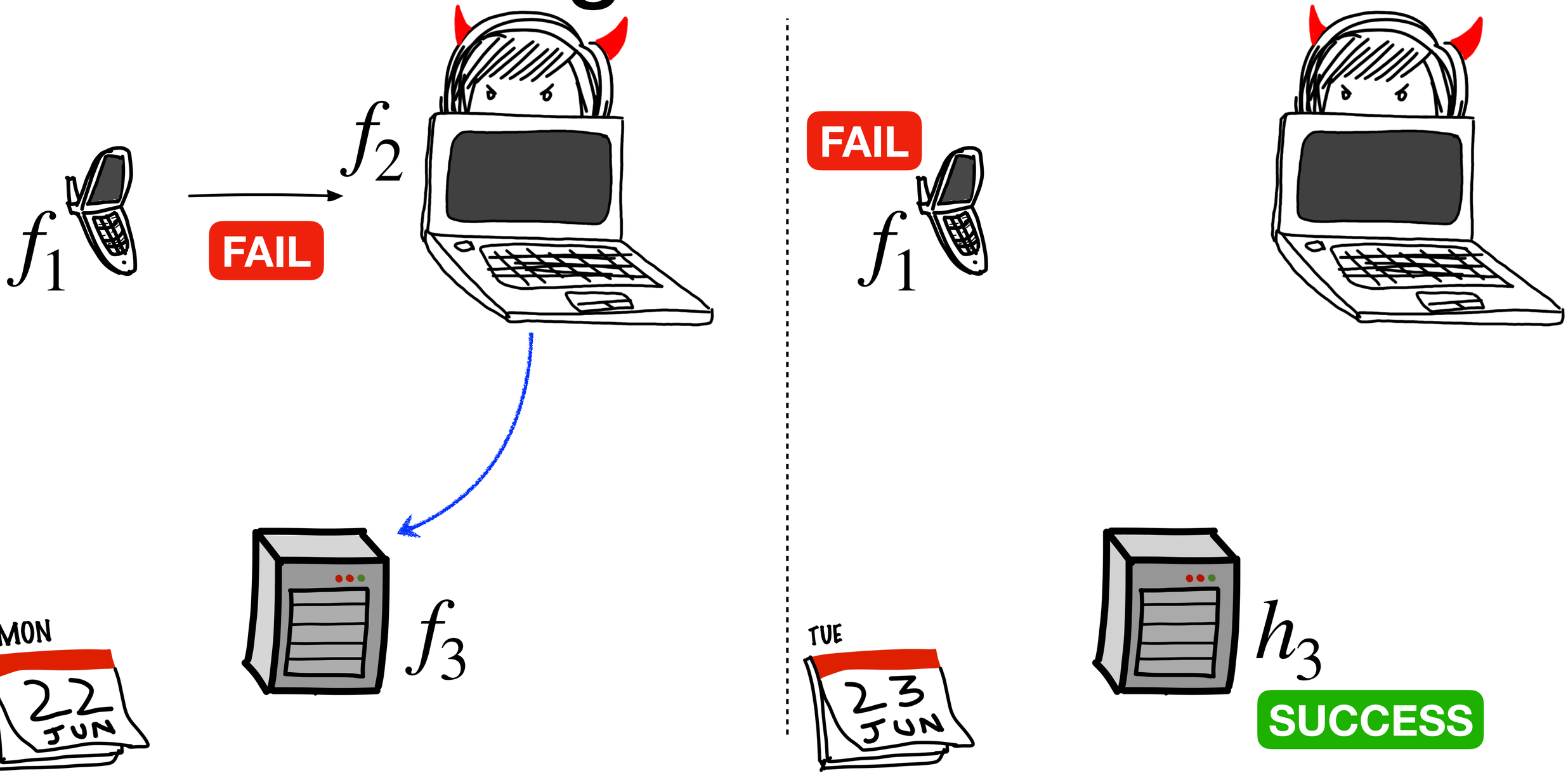


SUCCESS

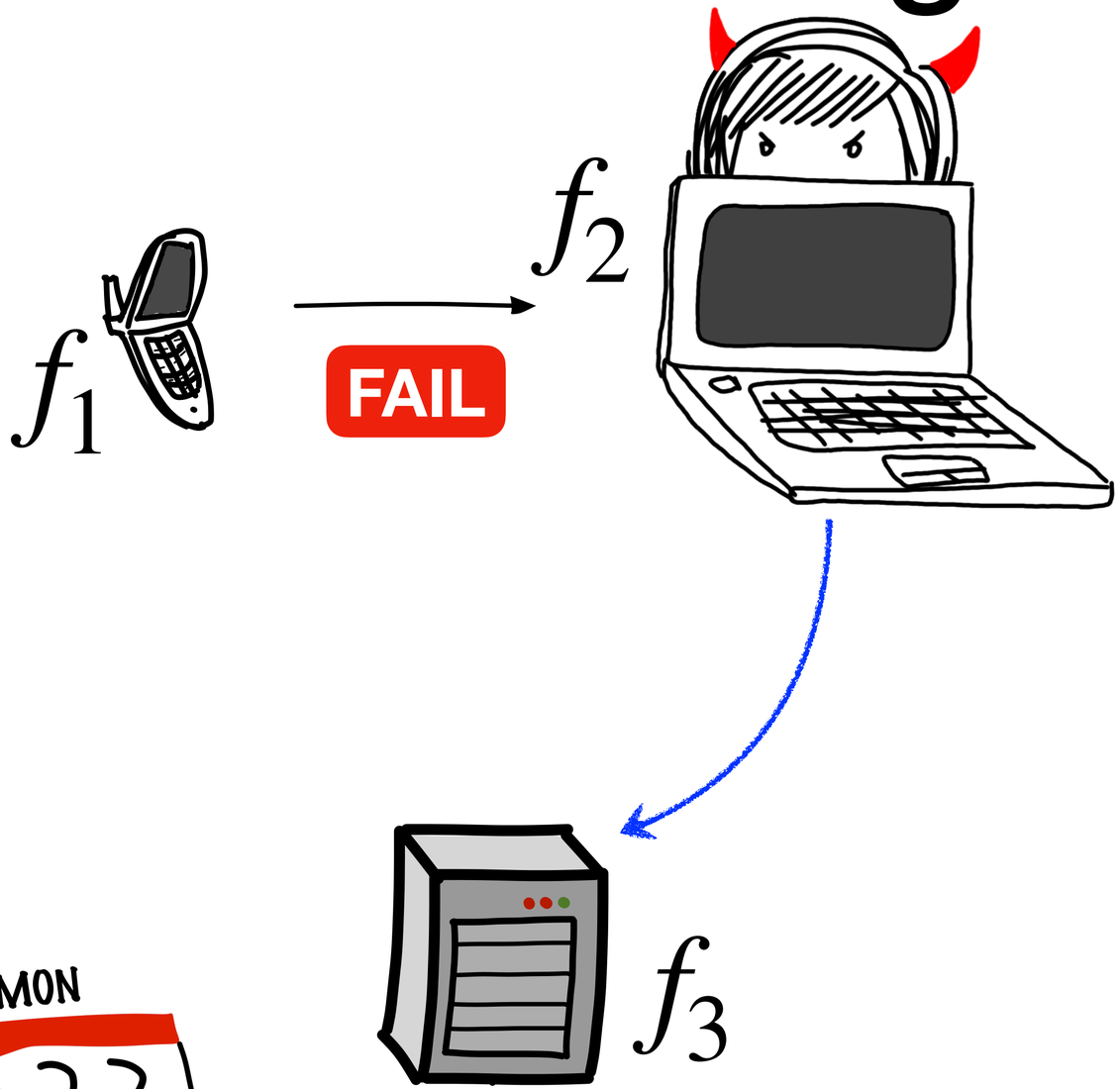
Defining Offline Refresh



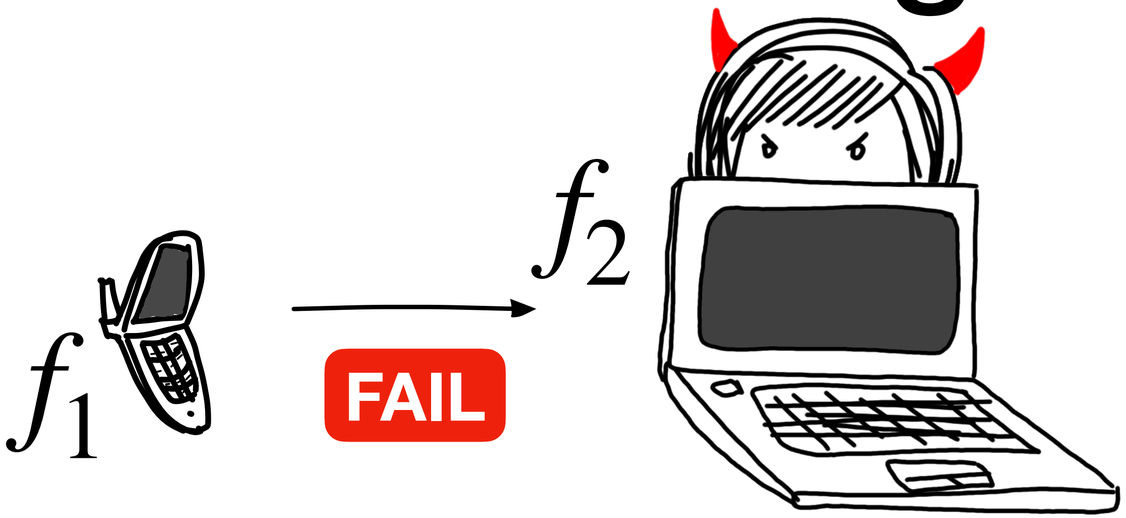
Defining Offline Refresh

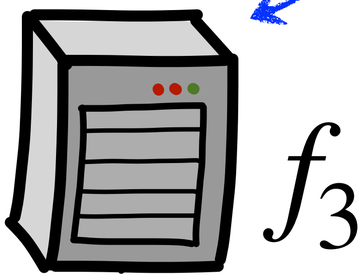


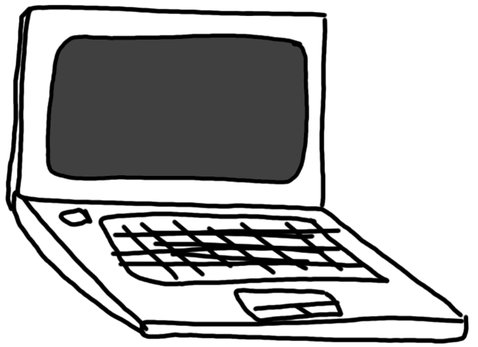
Defining Offline Refresh



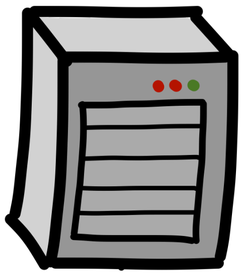
Defining Offline Refresh



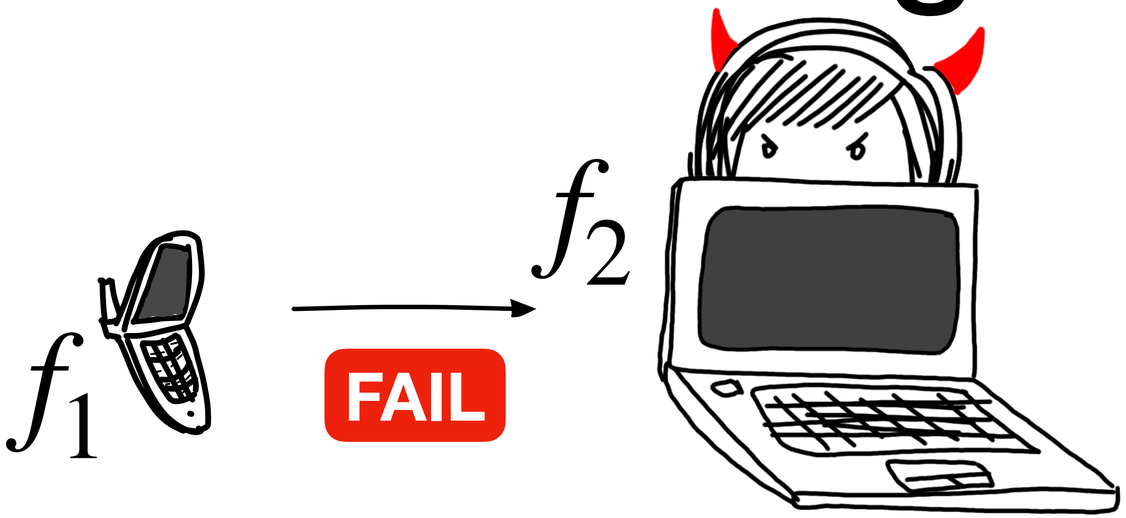


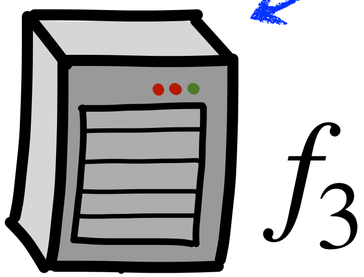


Agree to FAIL

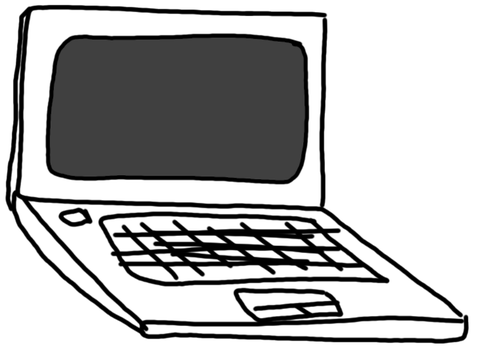


Defining Offline Refresh



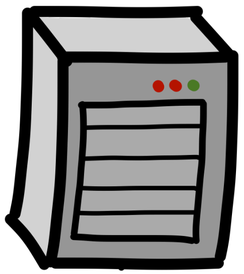


FAIL



Agree to FAIL

i.e. Unanimous Erasure



FAIL

This Work

- Correct definition is subtle
  - Guaranteed progress is impossible
  - We formulate **unanimous erasure**

- (*2*,*n*) setting: Efficient new protocol **native** to wallets

- (*t*,*n*) setting: **Impossible**!

This Work

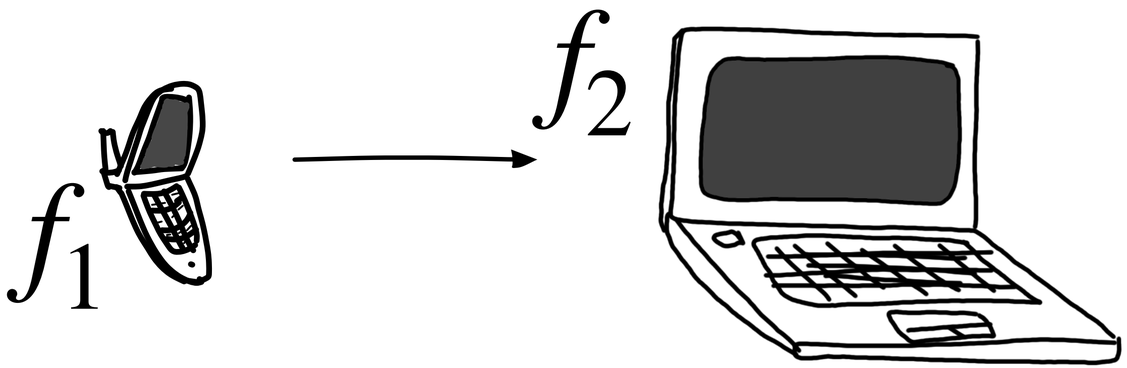
- Correct definition is subtle
  - Guaranteed progress is impossible
  - We formulate **unanimous erasure**

- (*2*,*n*) setting: Efficient new protocol **native** to wallets

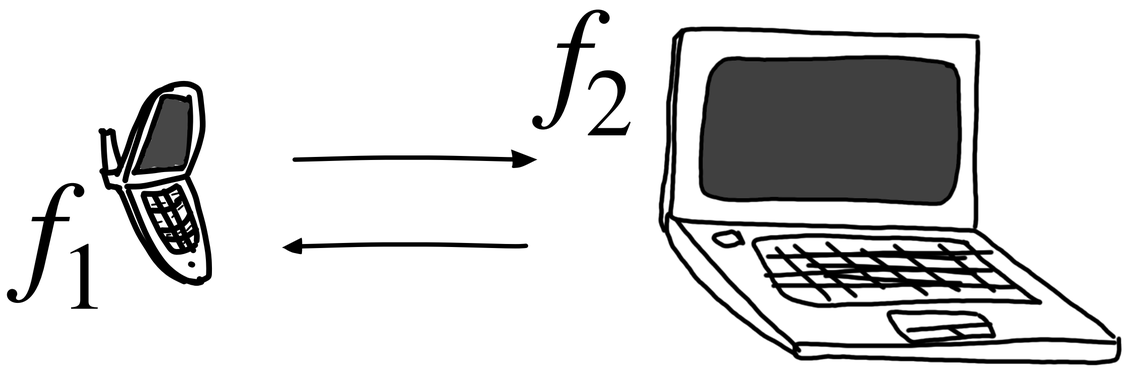
- (*t*,*n*) setting: **Impossible**!

Challenge: 2PC is Unfair

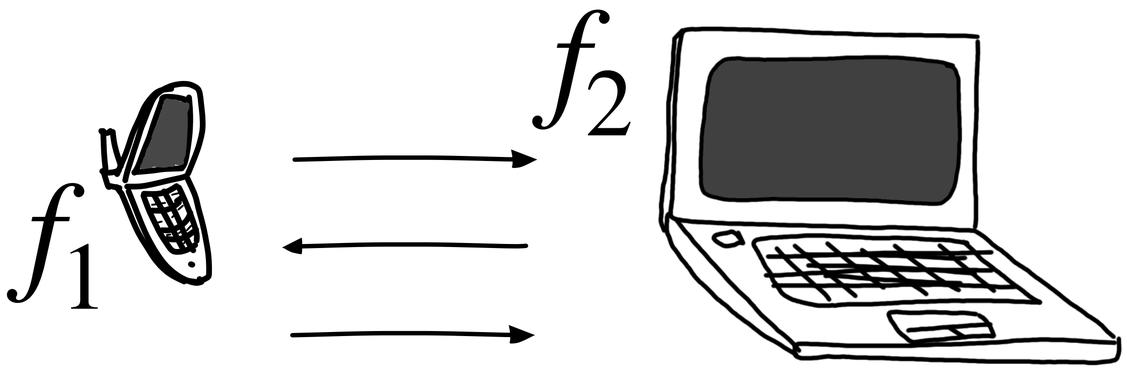
Challenge: 2PC is Unfair



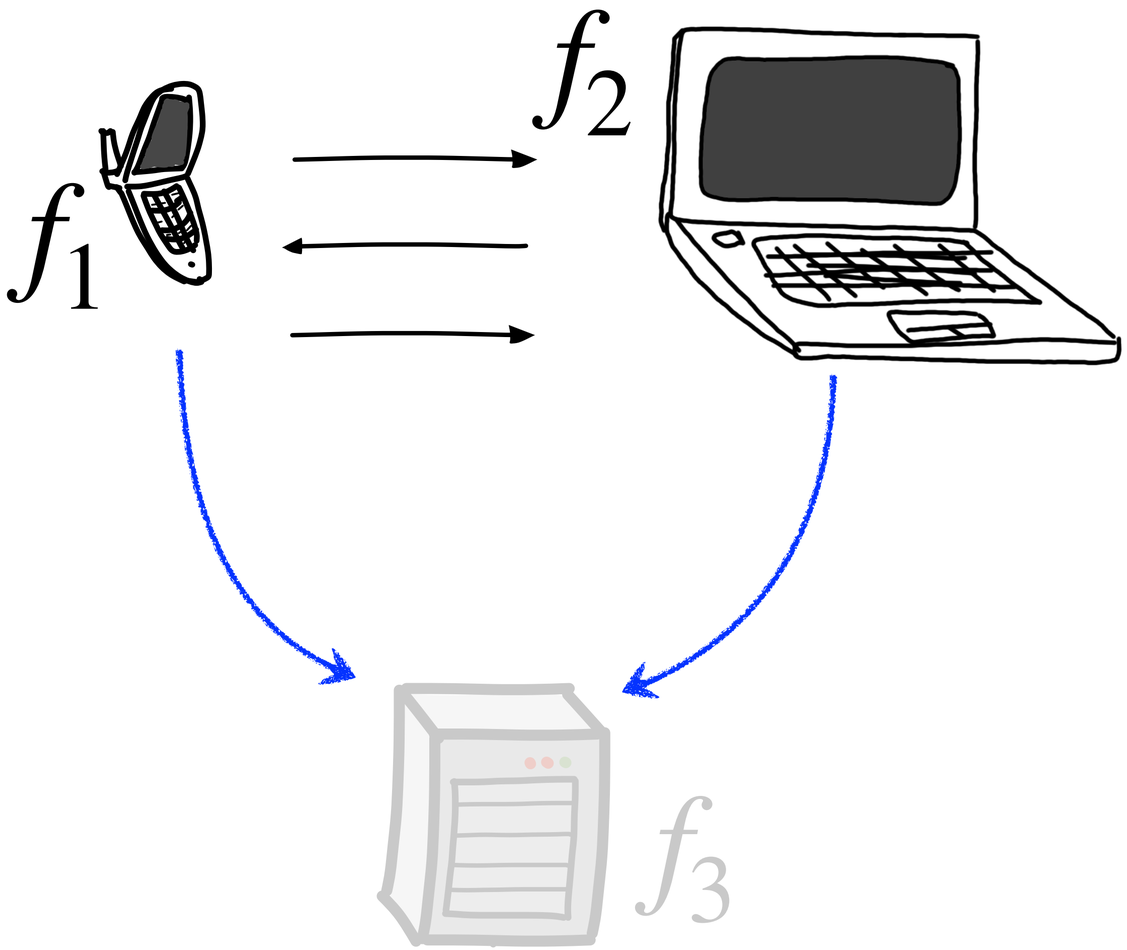
Challenge: 2PC is Unfair



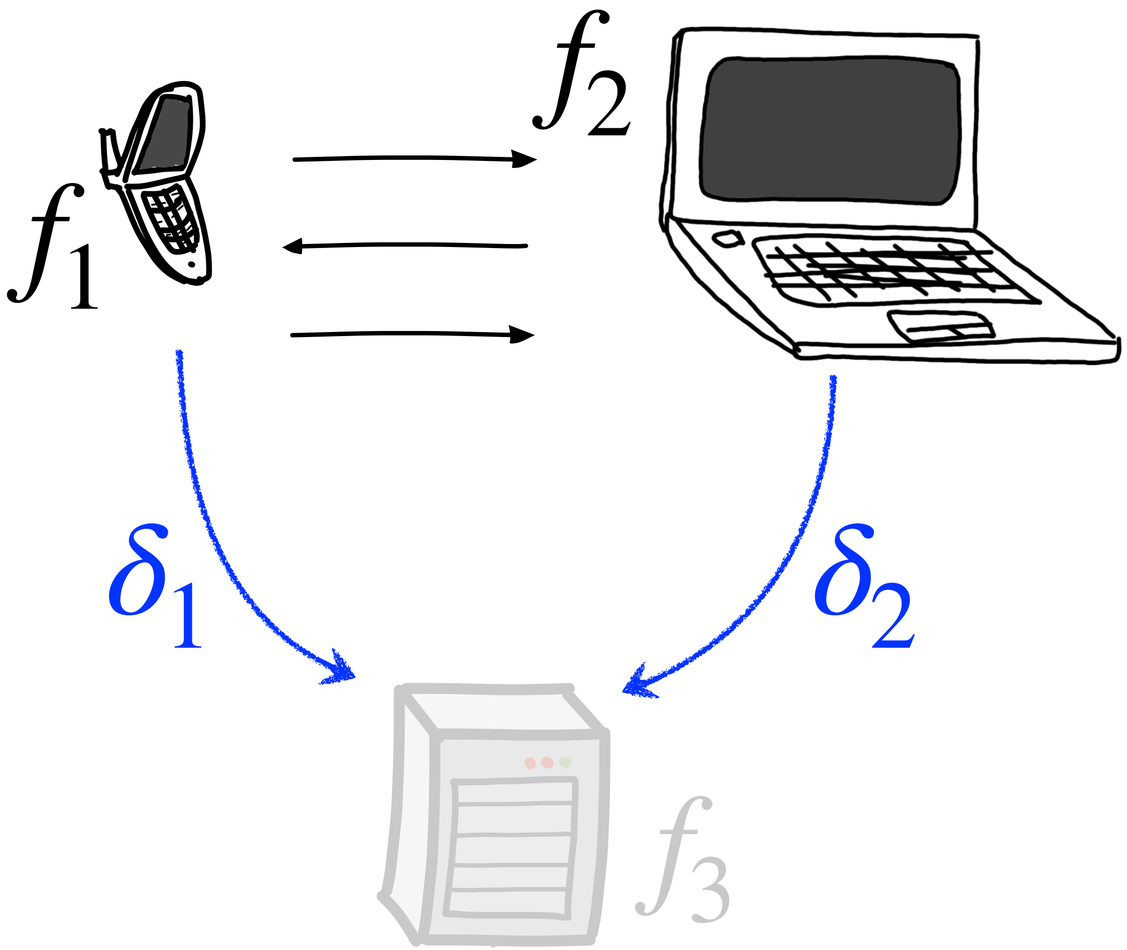
Challenge: 2PC is Unfair



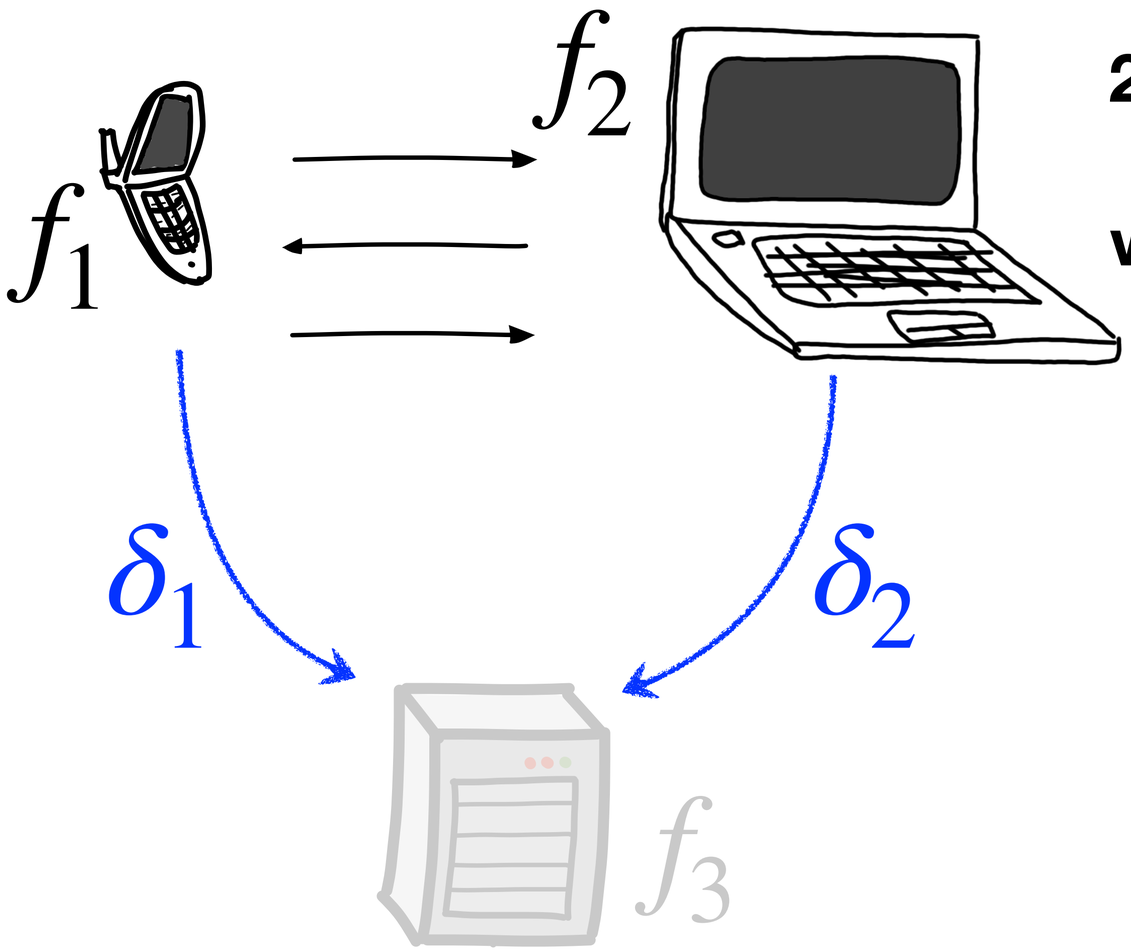
Challenge: 2PC is Unfair



Challenge: 2PC is Unfair

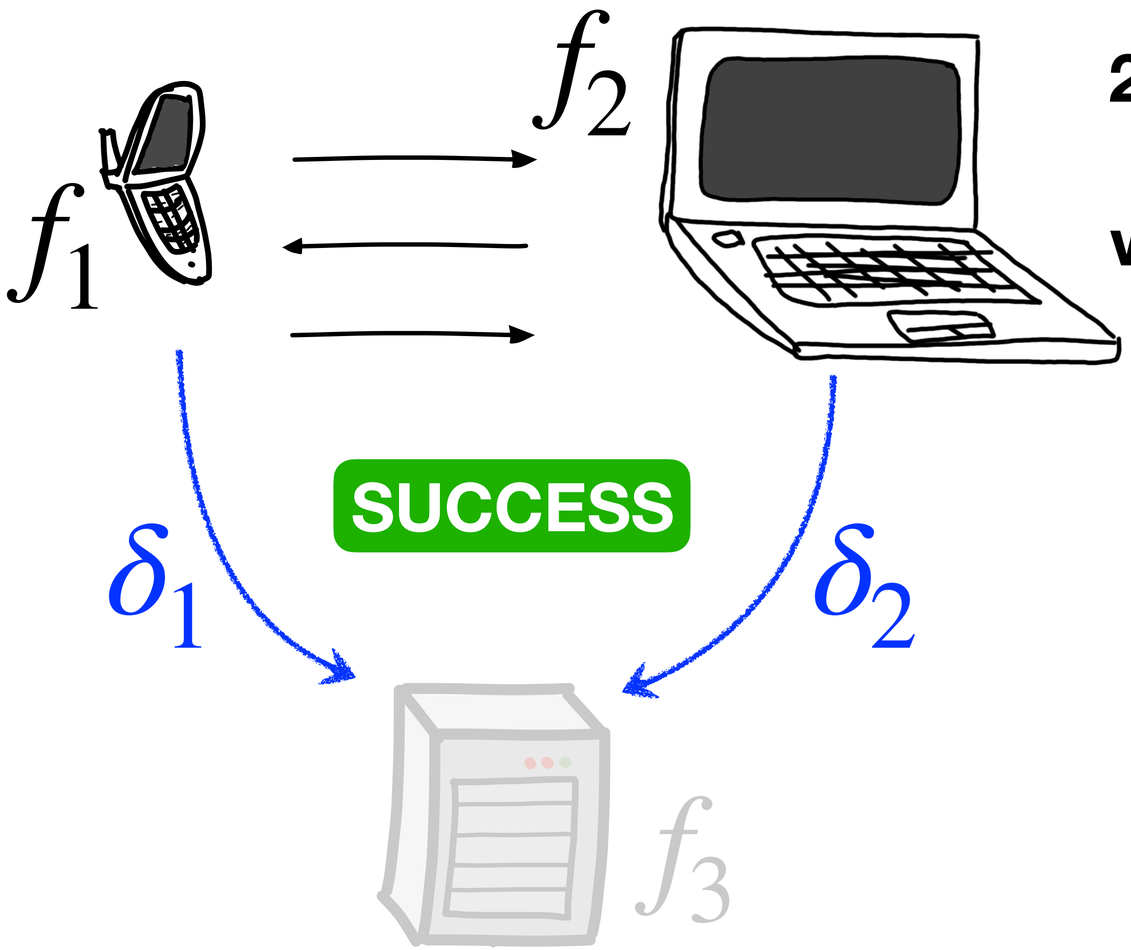


Challenge: 2PC is Unfair



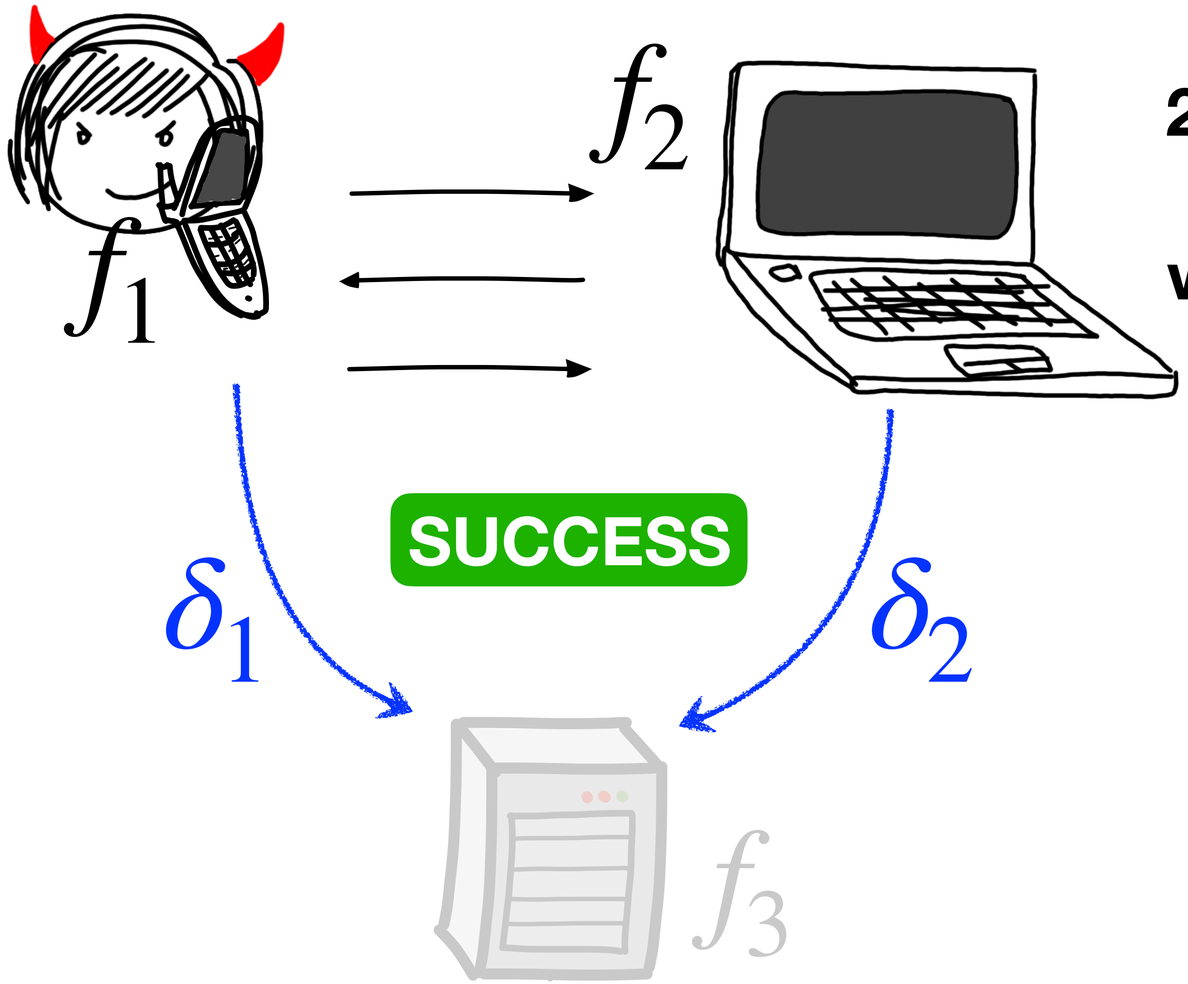
2 party computation (2PC) protocol with outputs δ_1, δ_2.

Challenge: 2PC is Unfair



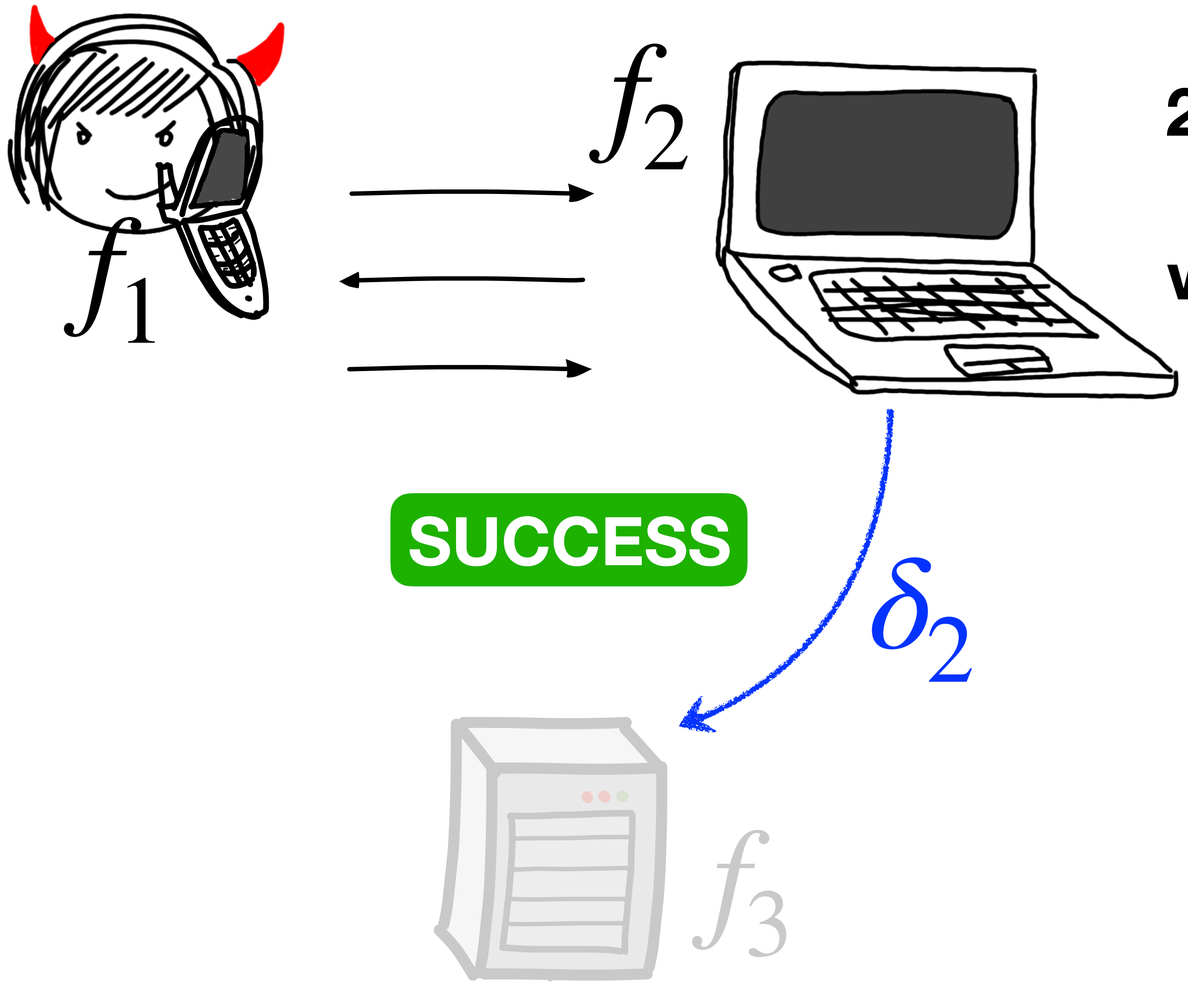
2 party computation (2PC) protocol with outputs δ_1, δ_2.

Challenge: 2PC is Unfair



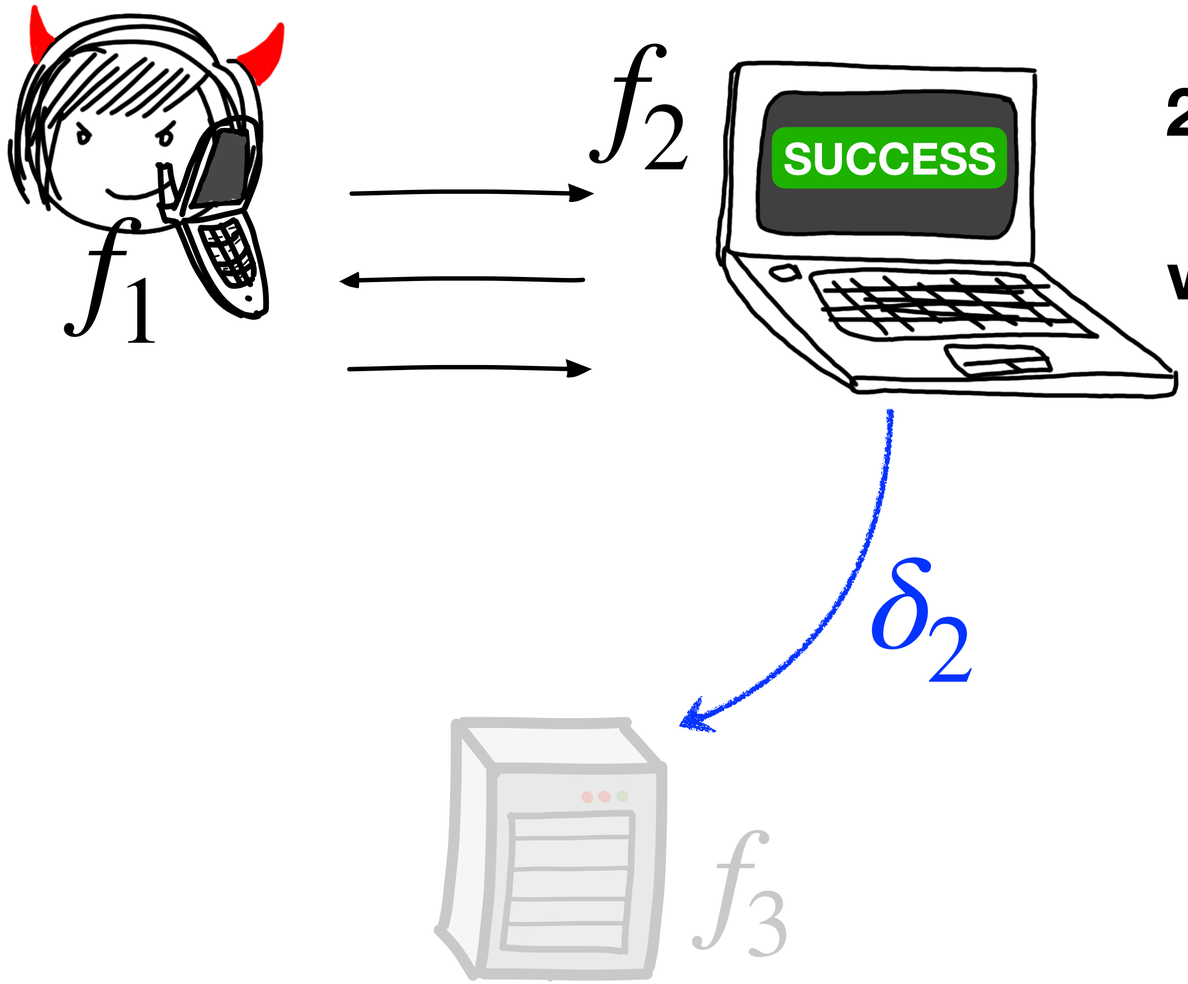
2 party computation (2PC) protocol with outputs δ_1, δ_2.

Challenge: 2PC is Unfair



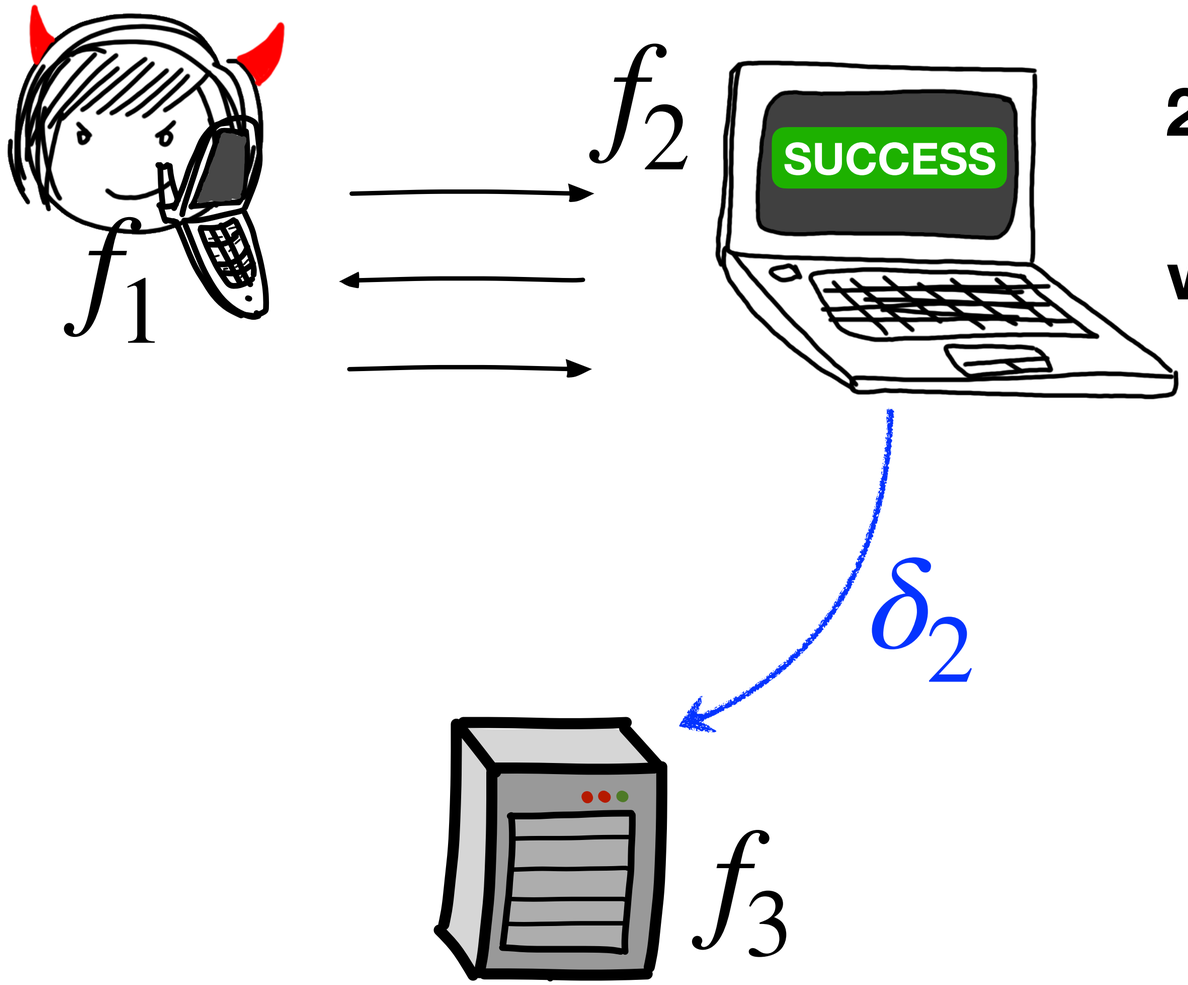
2 party computation (2PC) protocol with outputs δ_1, δ_2.

Challenge: 2PC is Unfair



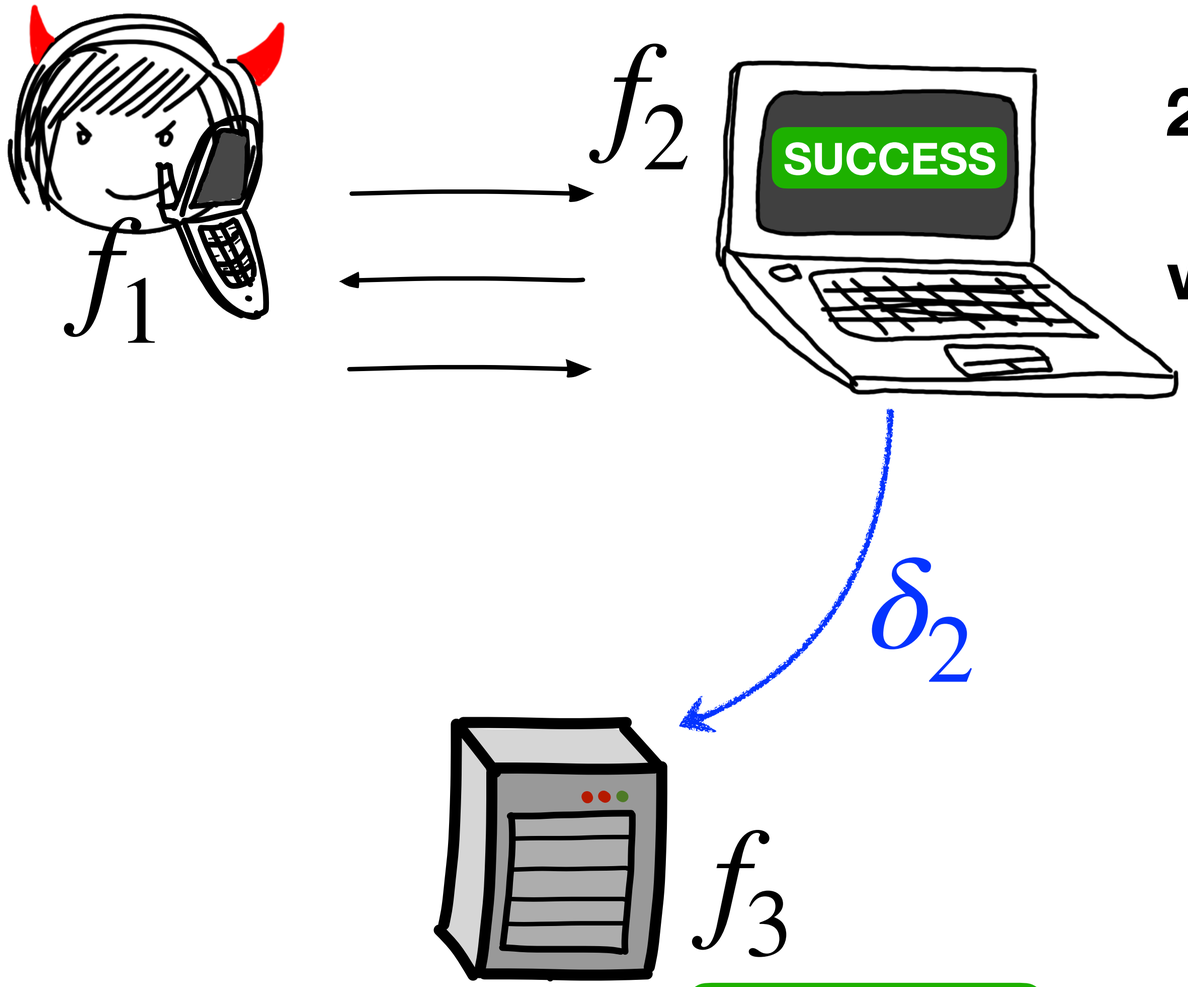
2 party computation (2PC) protocol with outputs δ_1, δ_2.

Challenge: 2PC is Unfair



2 party computation (2PC) protocol with outputs δ_1, δ_2.

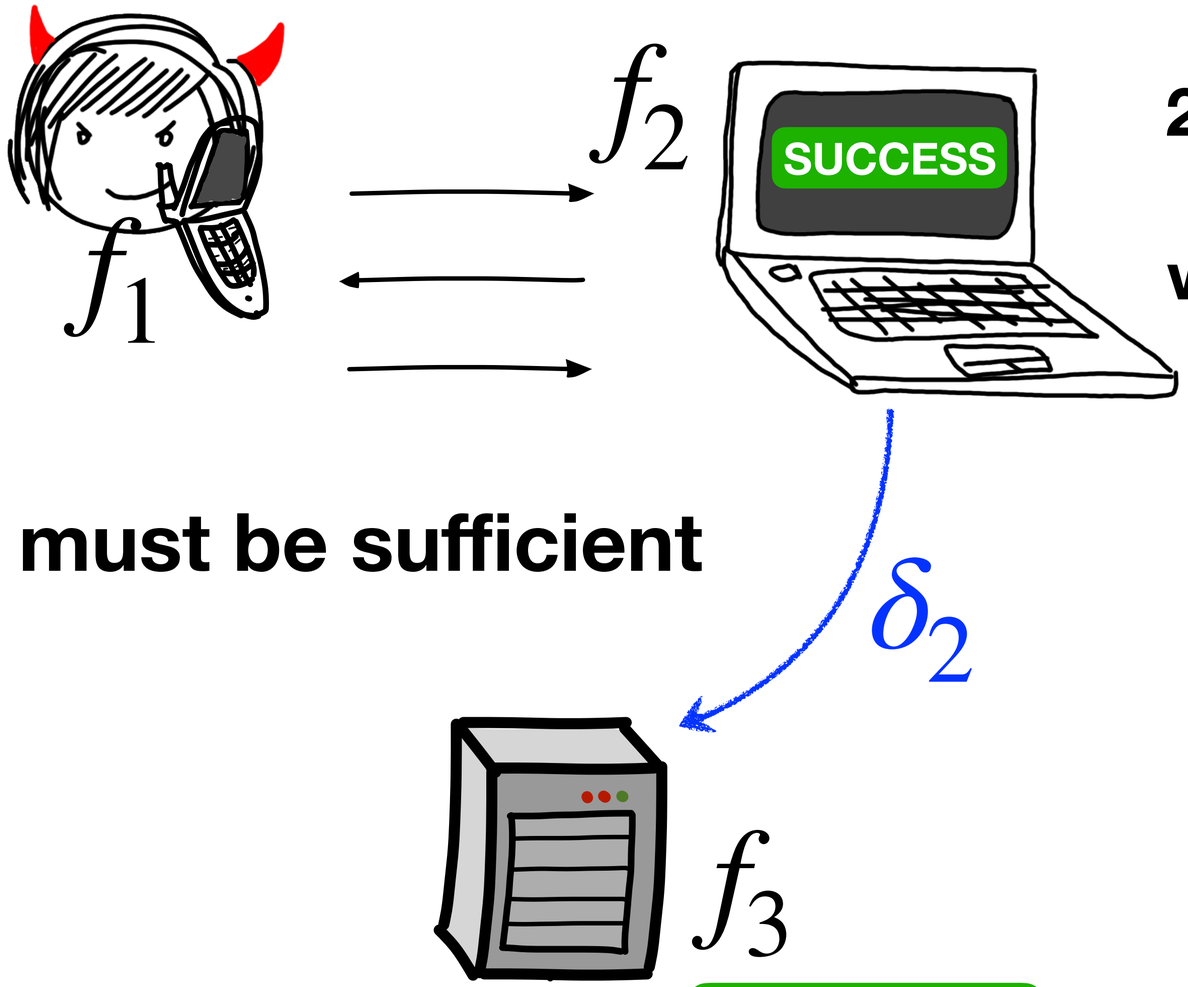
Challenge: 2PC is Unfair



2 party computation (2PC) protocol with outputs δ_1, δ_2.

SUCCESS (unanimous erasure)

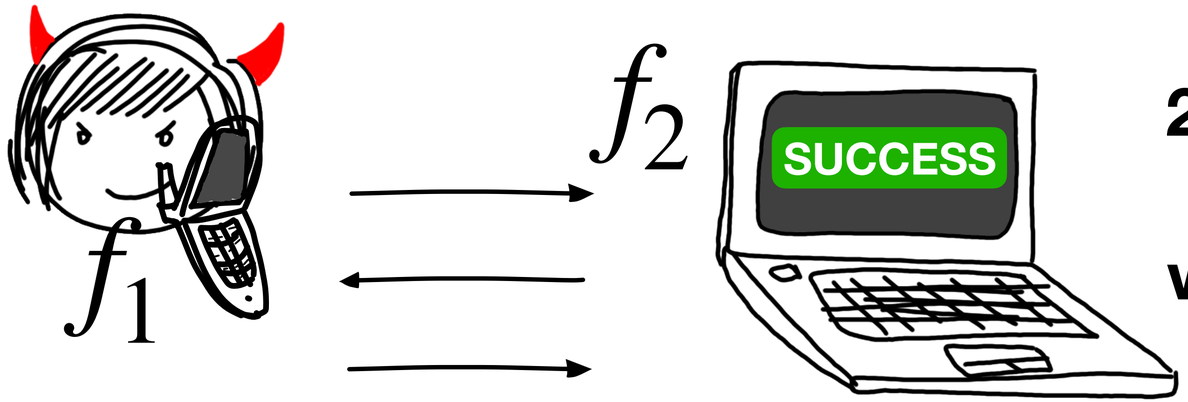
Challenge: 2PC is Unfair



2 party computation (2PC) protocol with outputs δ_1, δ_2.

Lesson: δ_2 must be sufficient

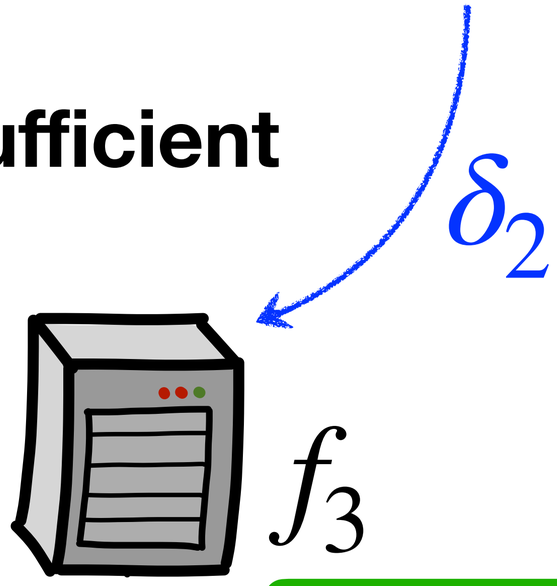
Challenge: 2PC is Unfair



2 party computation (2PC) protocol with outputs δ_1, δ_2.

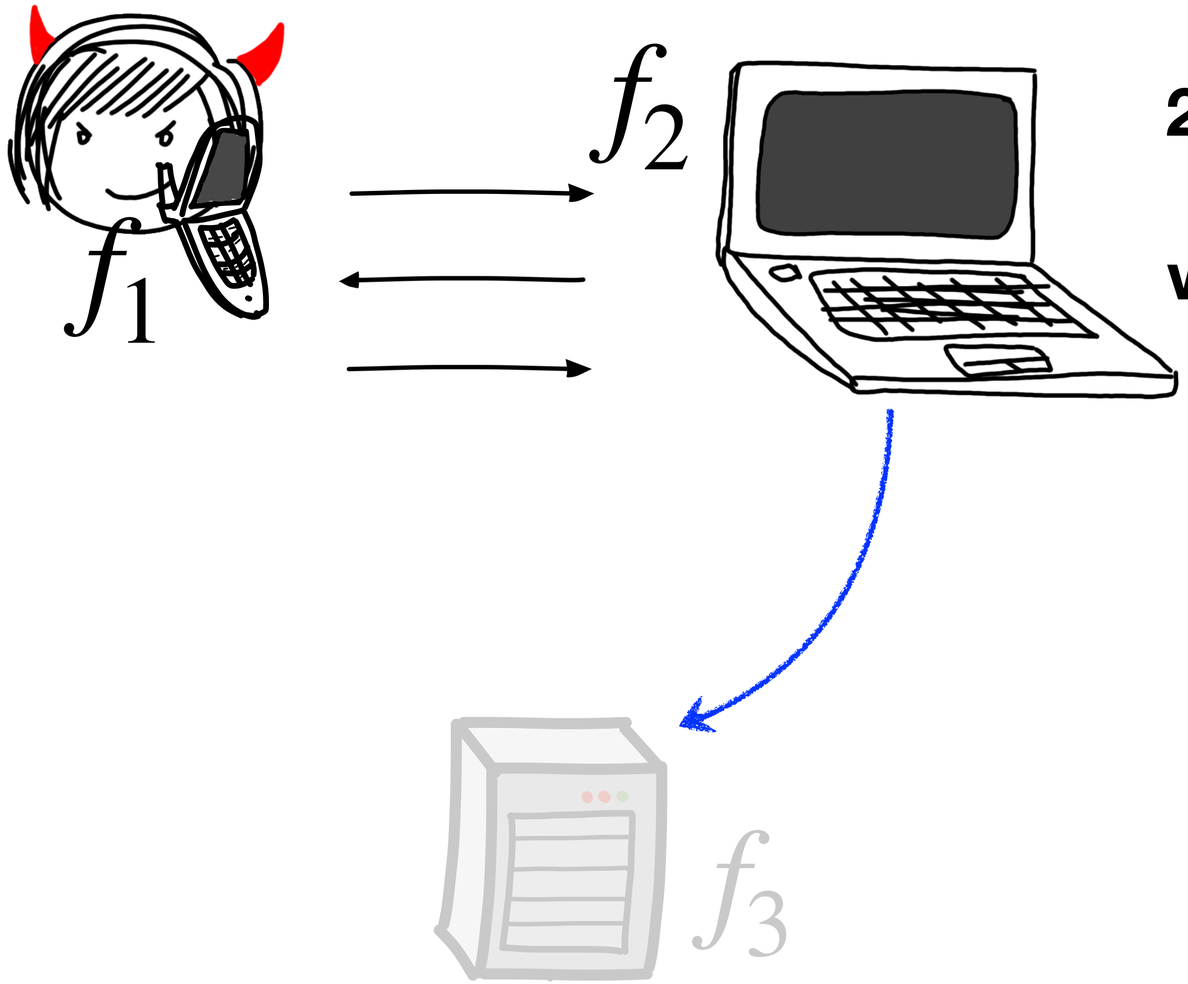
Lesson: δ_2 must be sufficient

(Equivalently δ_1)



SUCCESS (unanimous erasure)

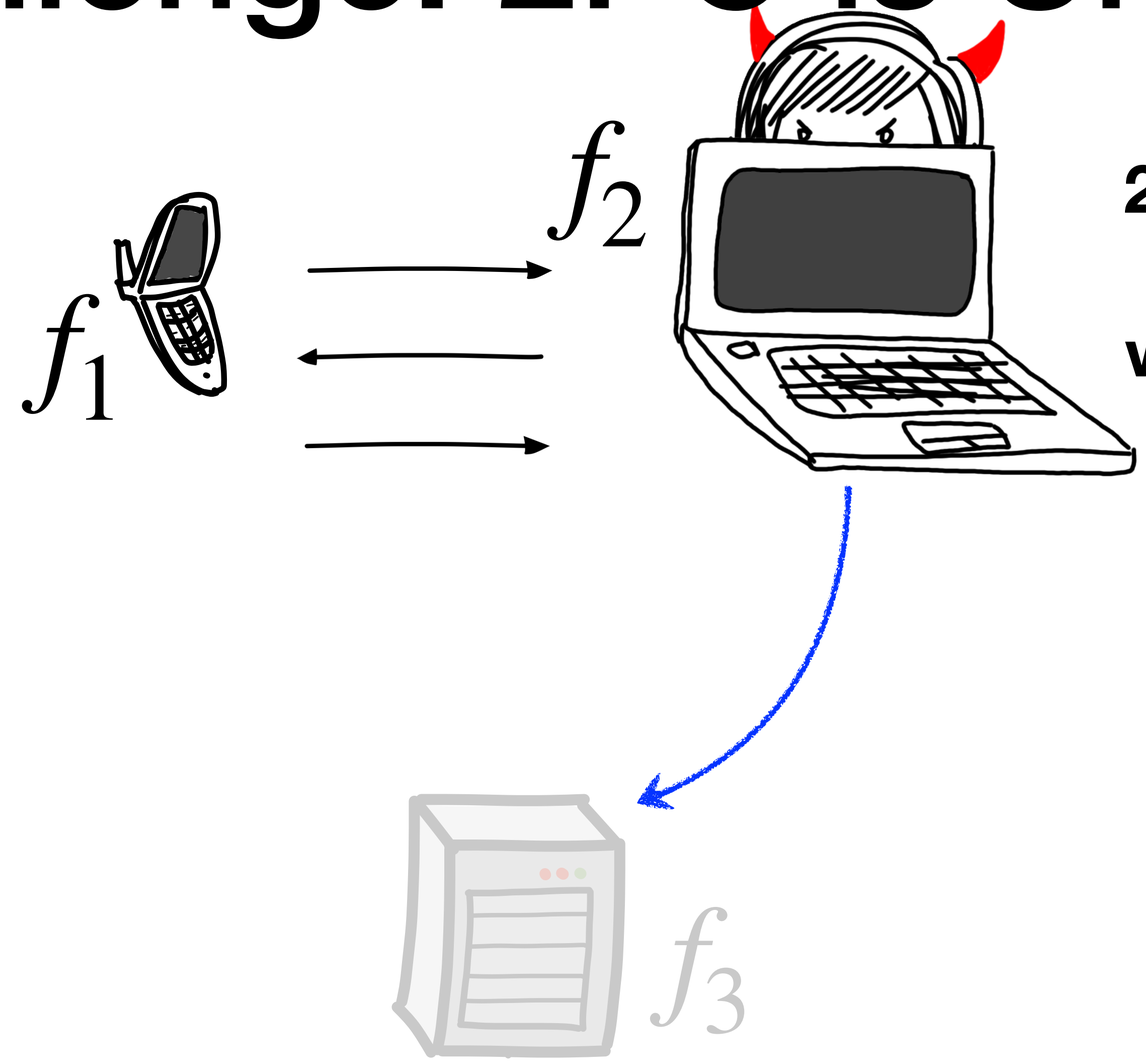
Challenge: 2PC is Unfair



2 party computation (2PC) protocol with outputs δ_1, δ_2.

Lesson: δ_2 must be sufficient

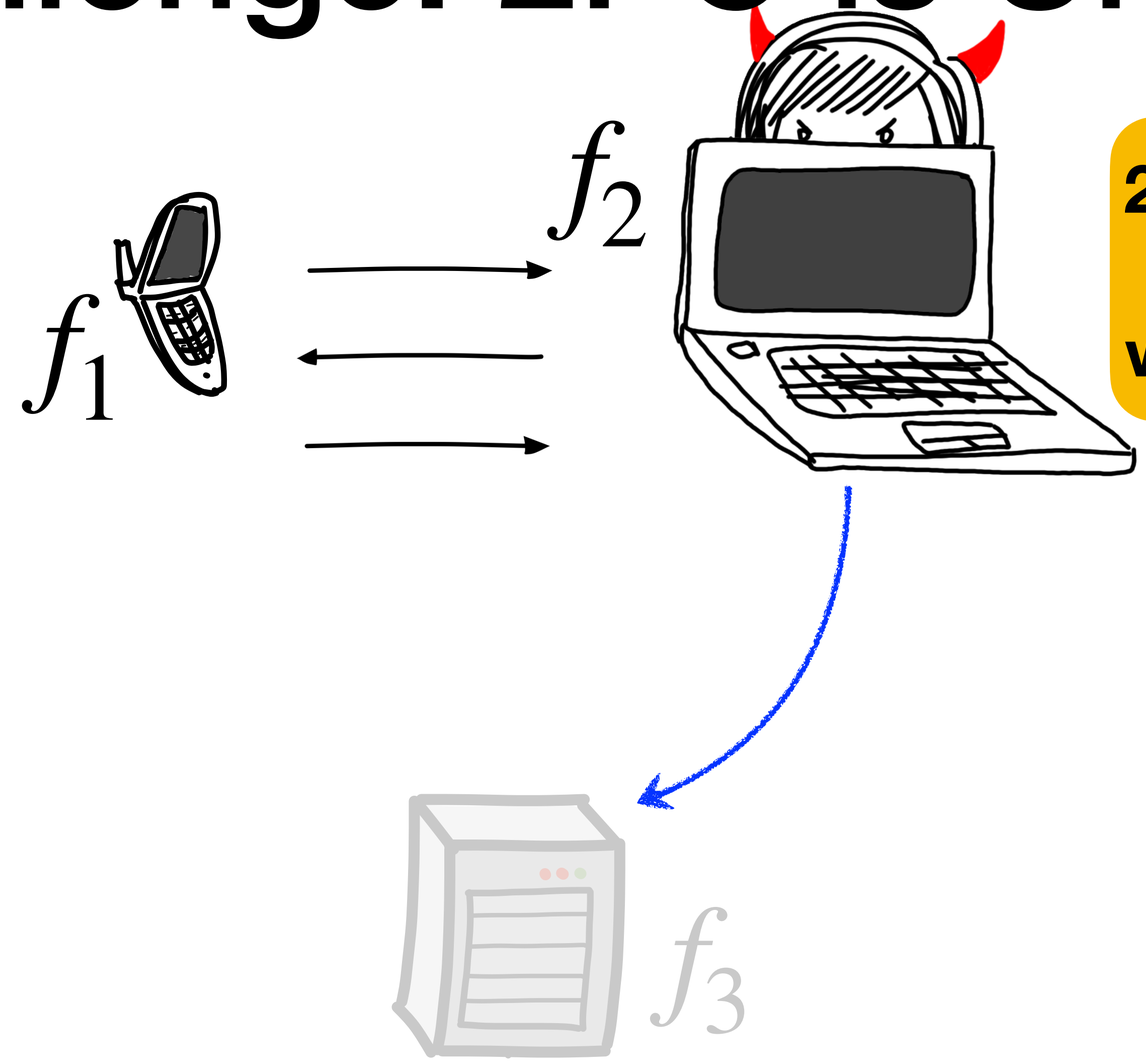
Challenge: 2PC is Unfair



2 party computation (2PC) protocol with outputs δ_1, δ_2.

Lesson: δ_2 must be sufficient

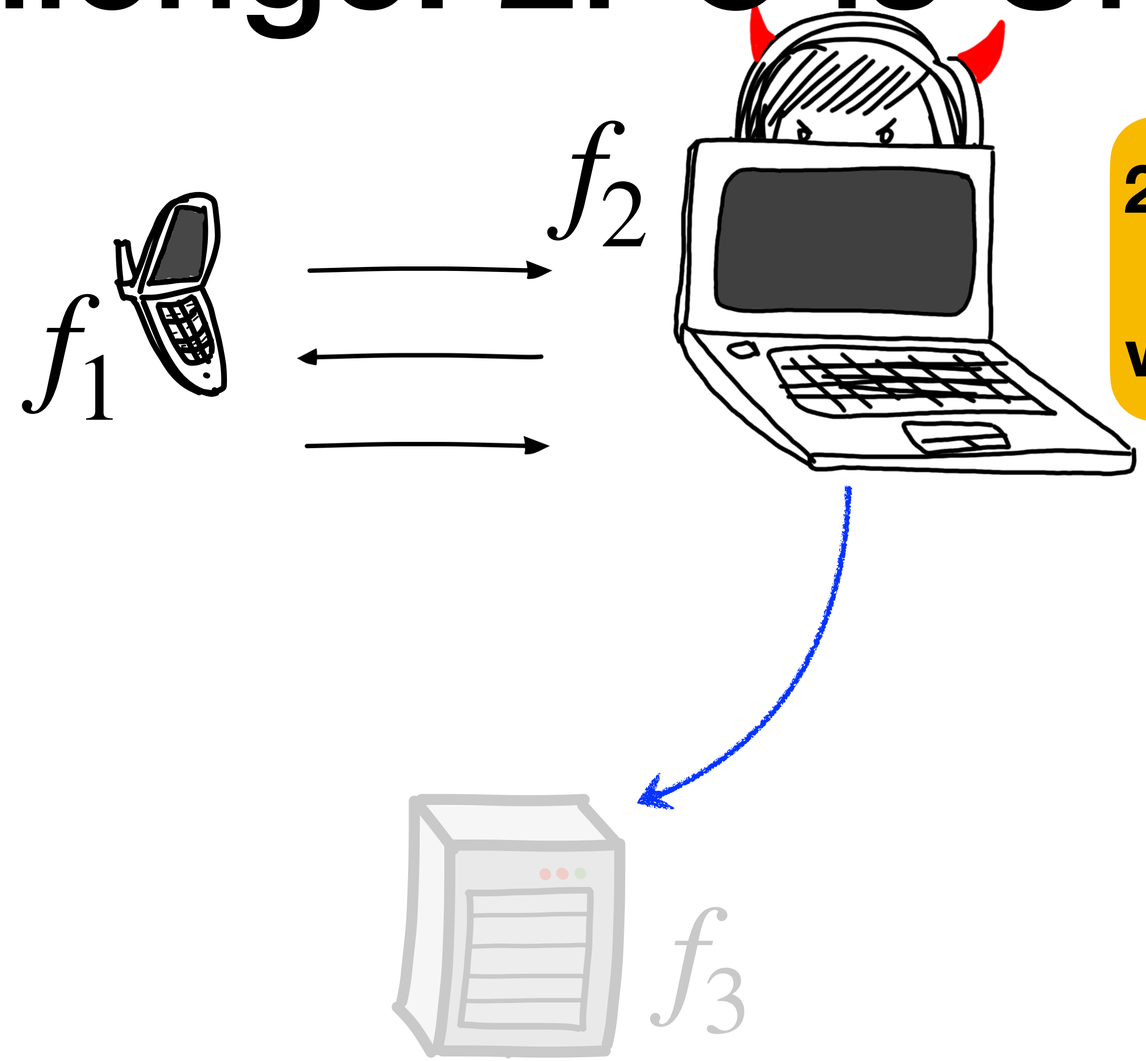
Challenge: 2PC is Unfair



2 party computation (2PC) protocol with outputs δ_1, δ_2.

Lesson: δ_2 must be sufficient

Challenge: 2PC is Unfair

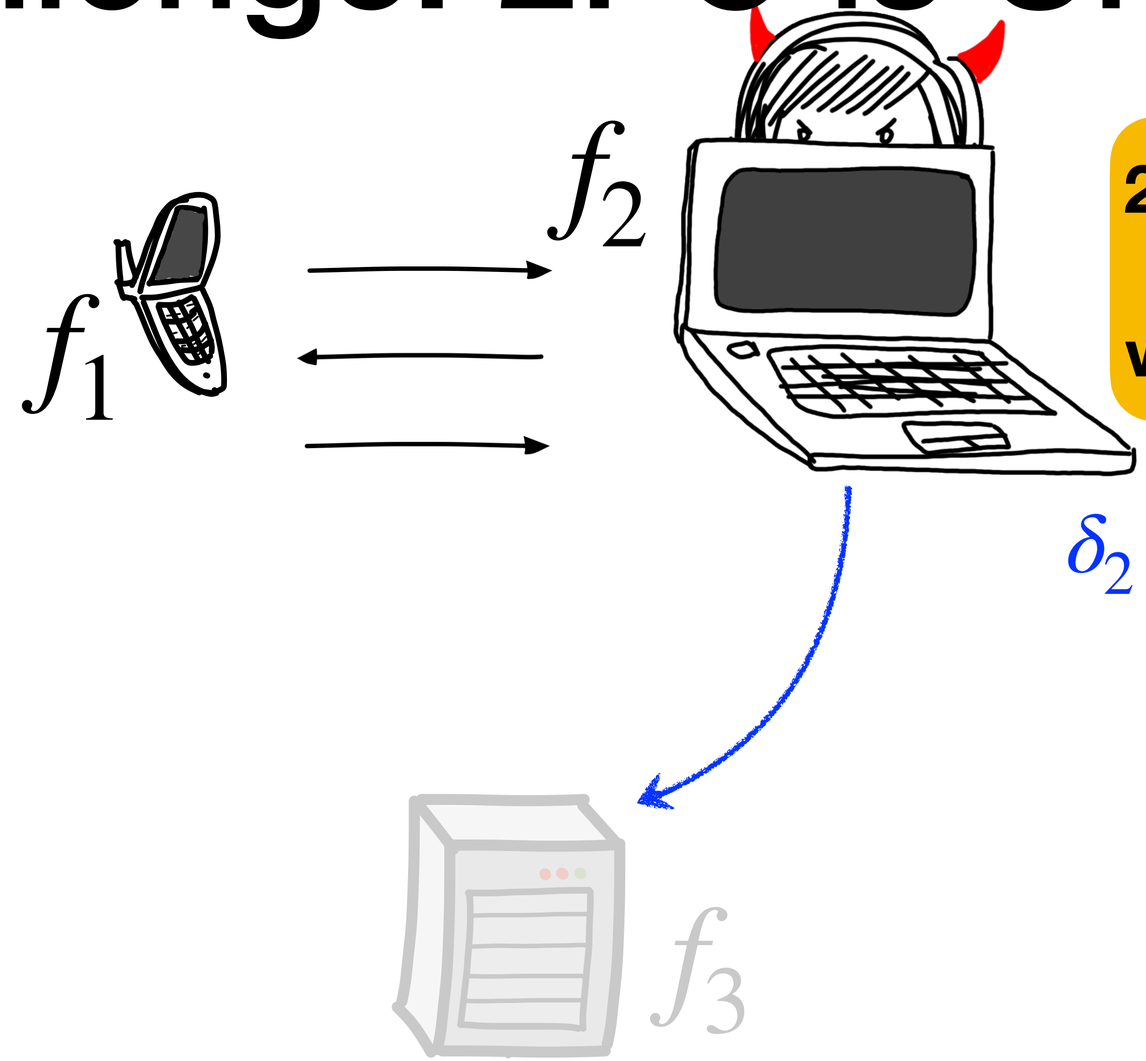


2 party computation (2PC) protocol with outputs δ_1, δ_2.

2PC is UNFAIR

Lesson: δ_2 must be sufficient

Challenge: 2PC is Unfair



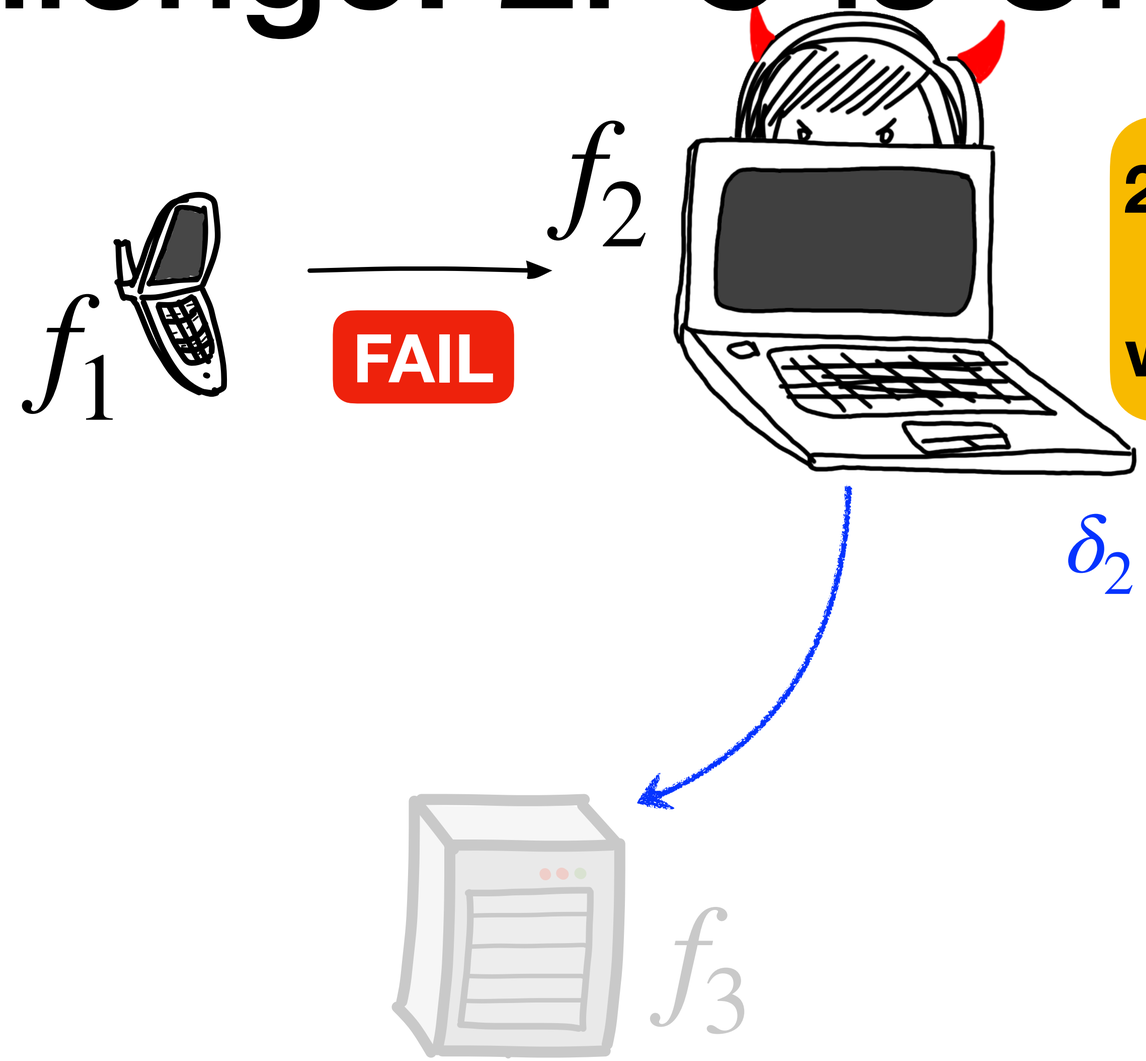
2 party computation (2PC) protocol with outputs δ_1, δ_2.

2PC is UNFAIR

δ_2 is delivered before δ_1

Lesson: δ_2 must be sufficient

Challenge: 2PC is Unfair



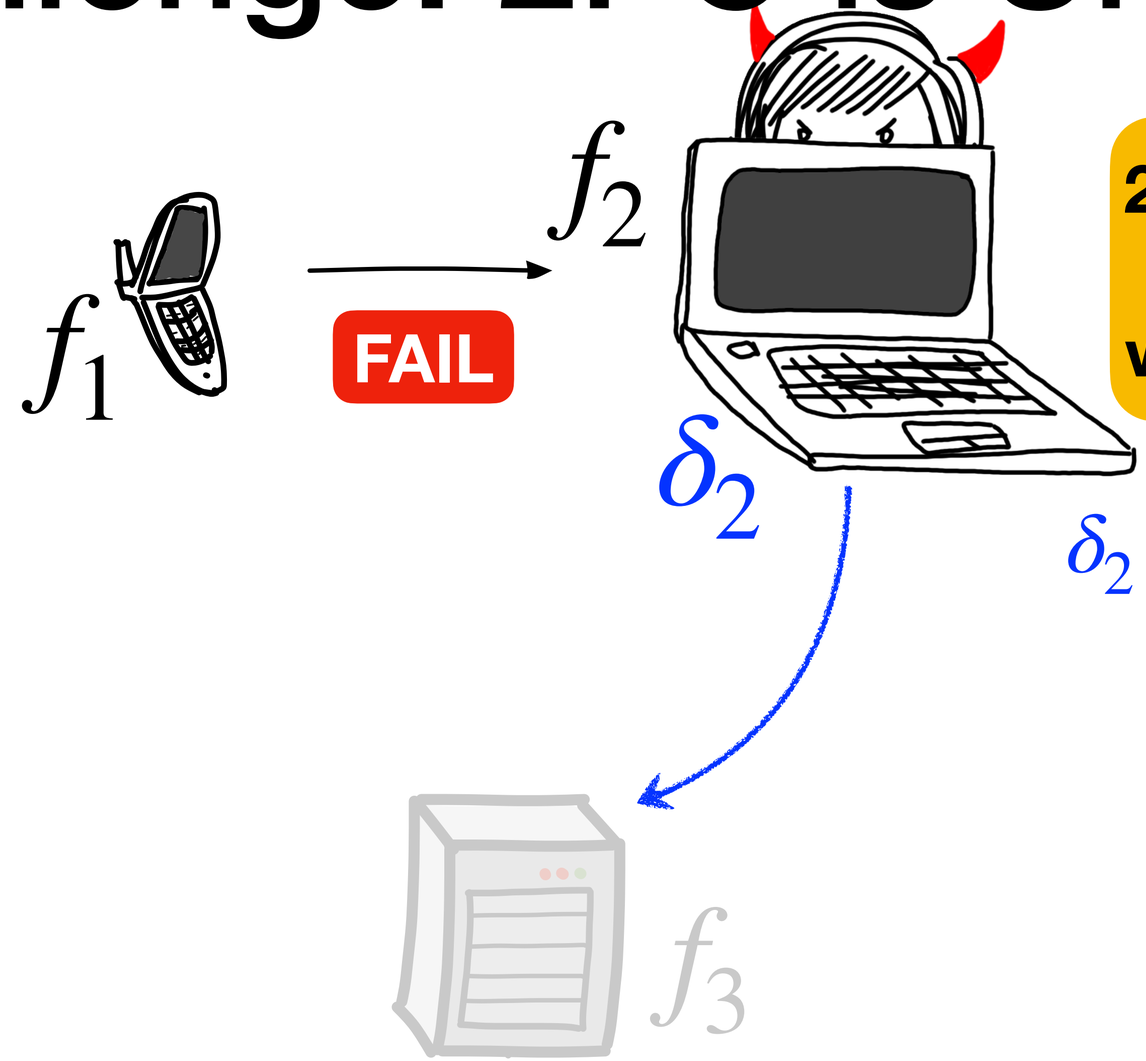
2 party computation (2PC) protocol with outputs δ_1, δ_2.

2PC is UNFAIR

δ_2 is delivered before δ_1

Lesson: δ_2 must be sufficient

Challenge: 2PC is Unfair



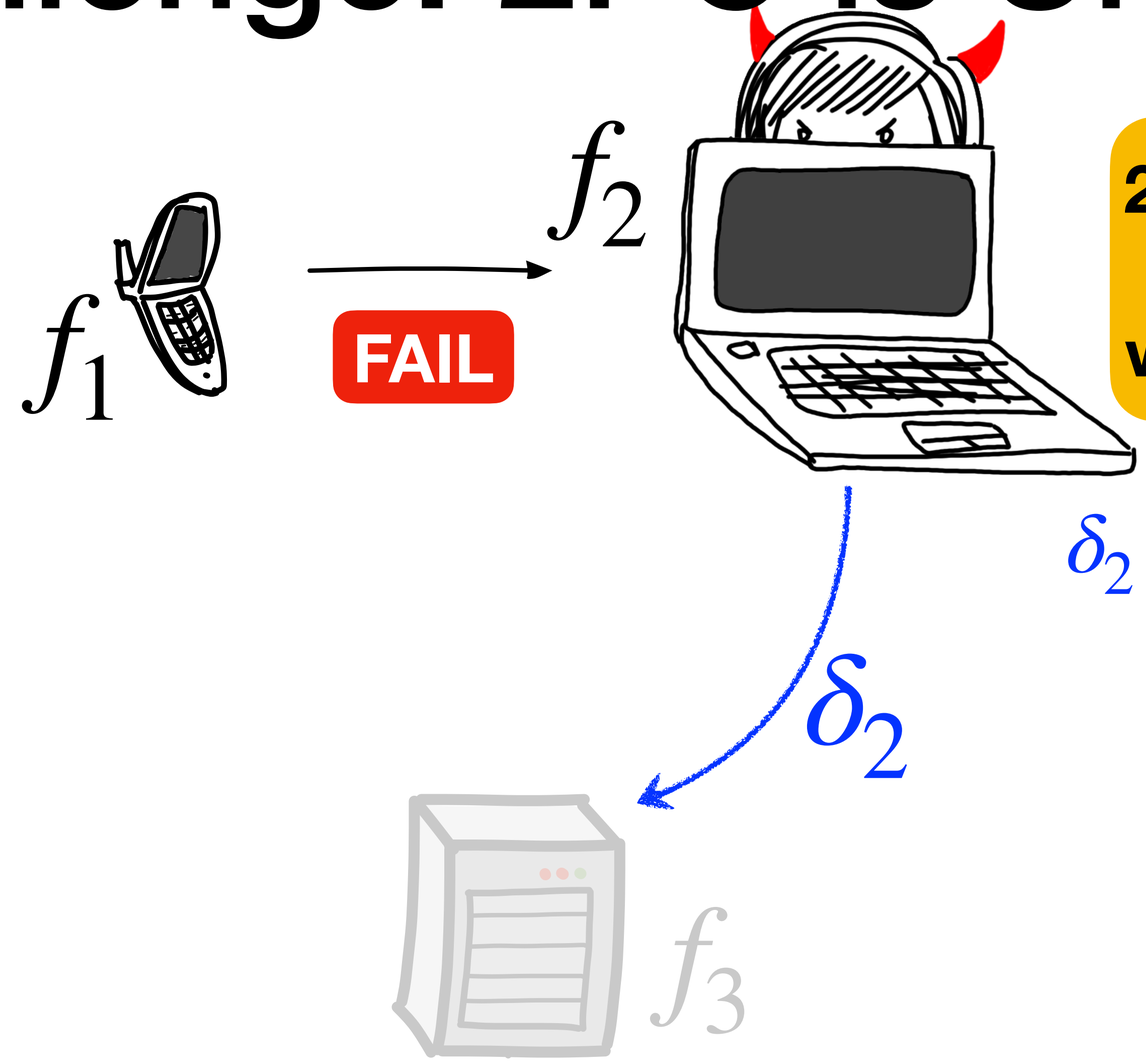
2 party computation (2PC) protocol with outputs δ_1, δ_2.

2PC is UNFAIR

δ_2 is delivered before δ_1

Lesson: δ_2 must be sufficient

Challenge: 2PC is Unfair



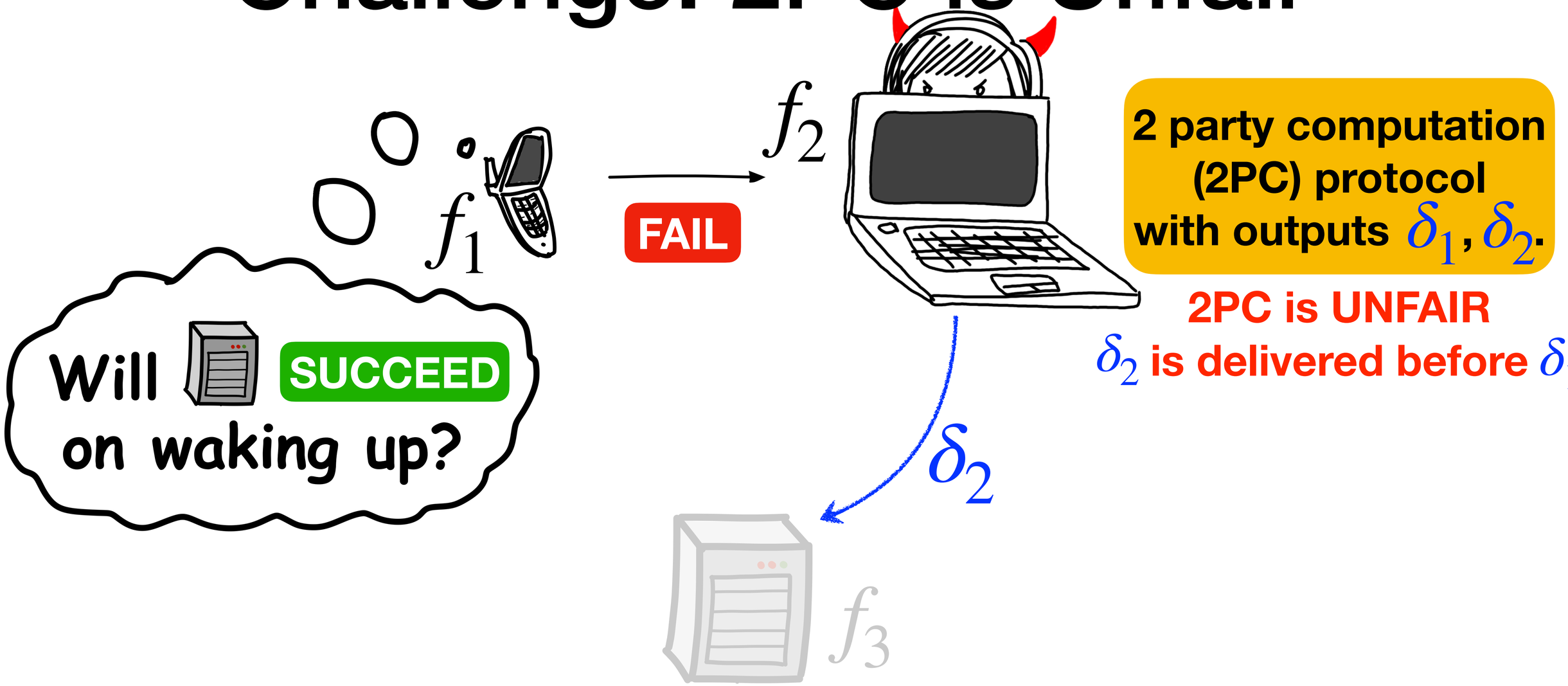
2 party computation (2PC) protocol with outputs δ_1, δ_2.

2PC is UNFAIR

δ_2 is delivered before δ_1

Lesson: δ_2 must be sufficient

Challenge: 2PC is Unfair



2 party computation (2PC) protocol with outputs δ_1, δ_2.

2PC is UNFAIR

δ_2 is delivered before δ_1

Lesson: δ_2 must be sufficient

Challenge: 2PC is Unfair

2 party computation (2PC) protocol with outputs δ_1, δ_2.

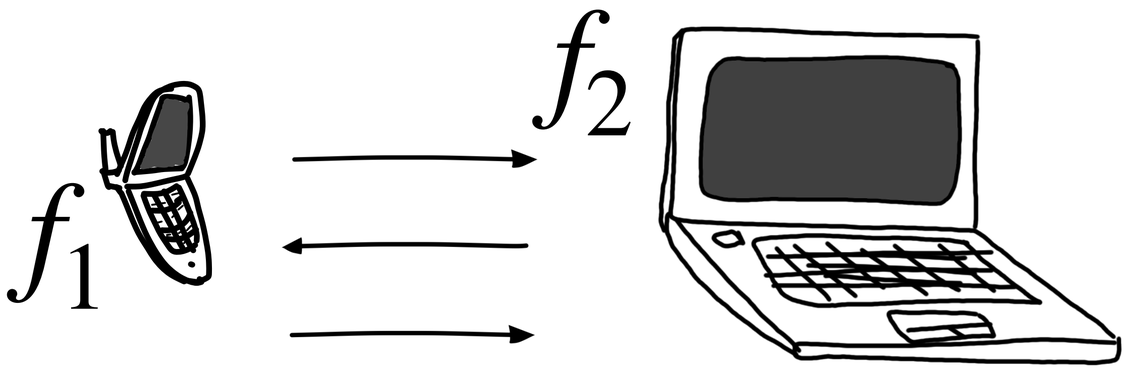
2PC is UNFAIR

δ_2 is delivered before δ_1

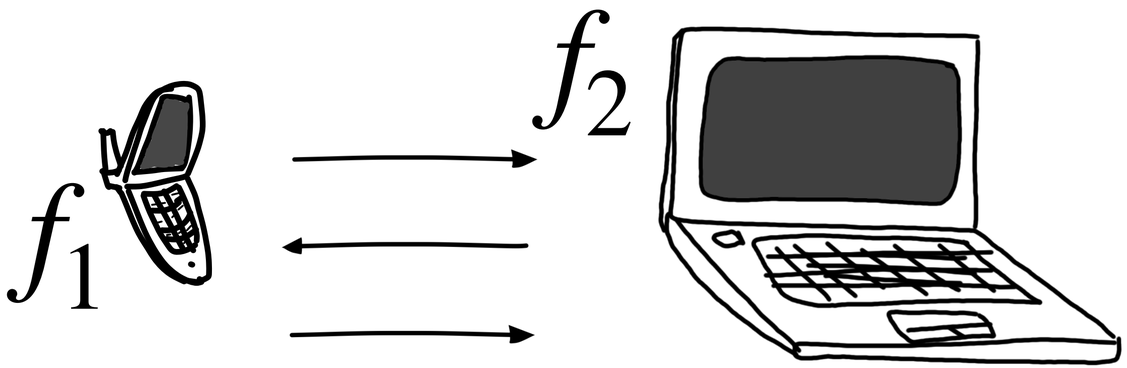
f_3

Lesson: δ_2 must be sufficient

Towards a Solution

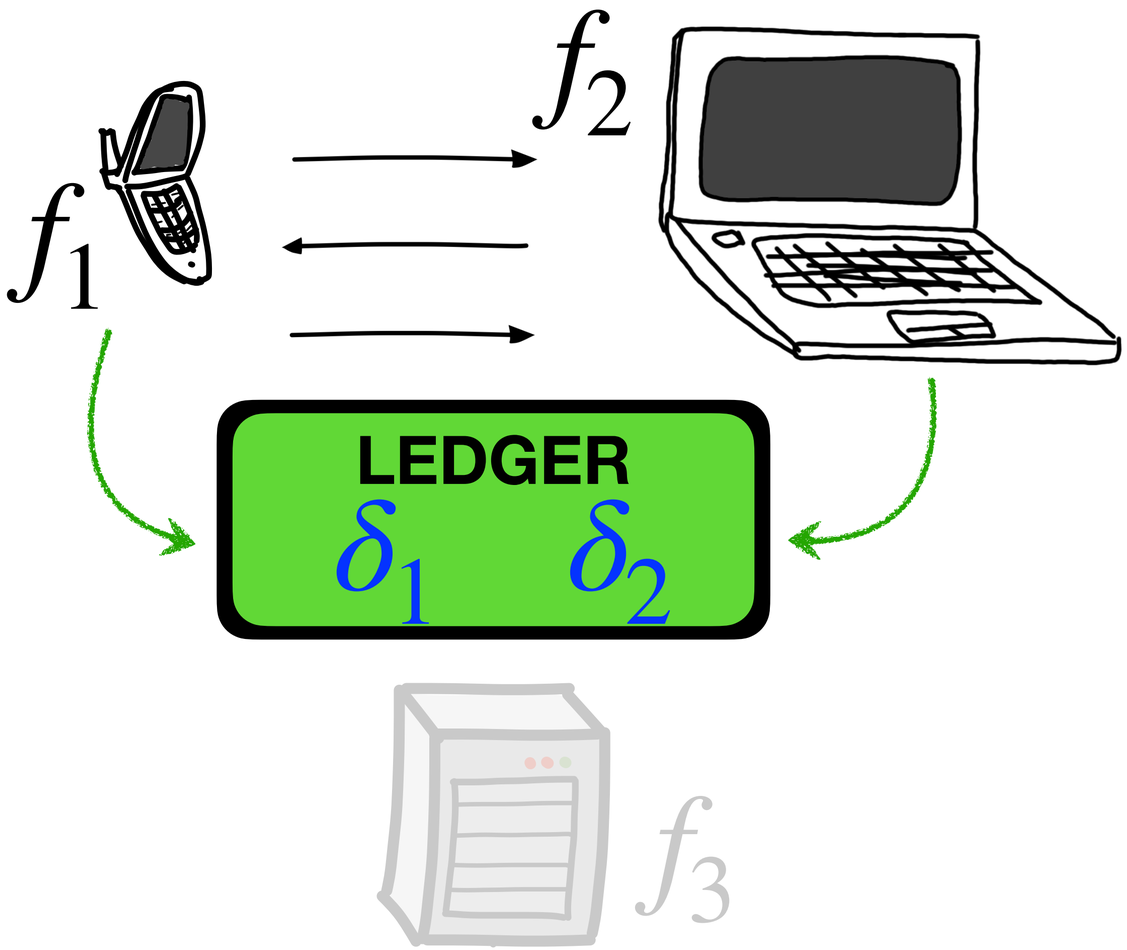


Towards a Solution

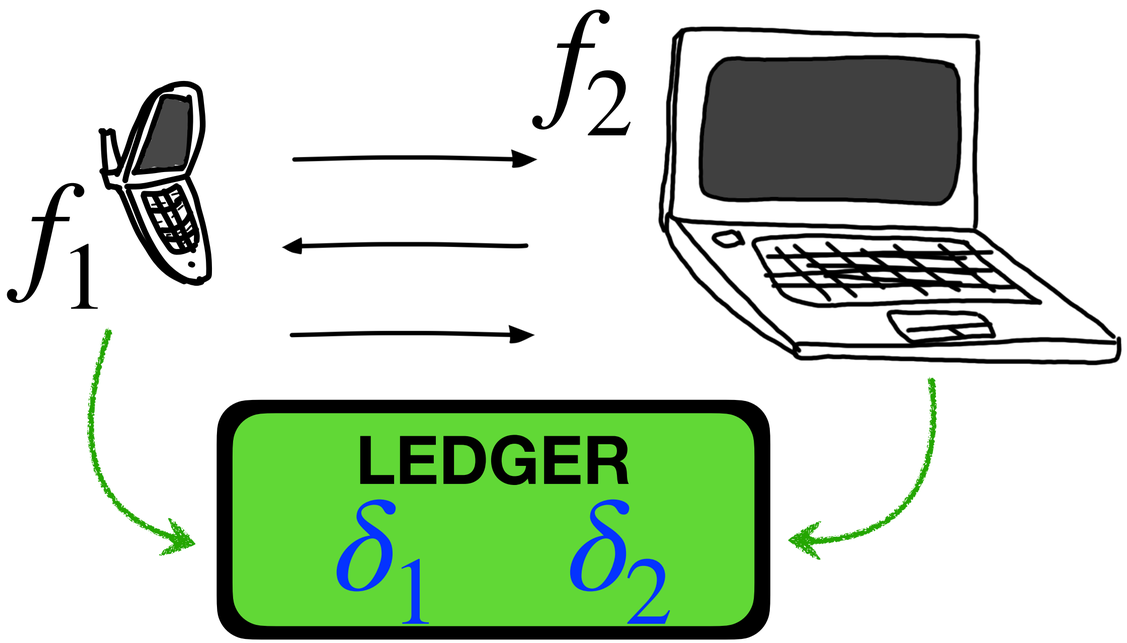


LEDGER

Towards a Solution

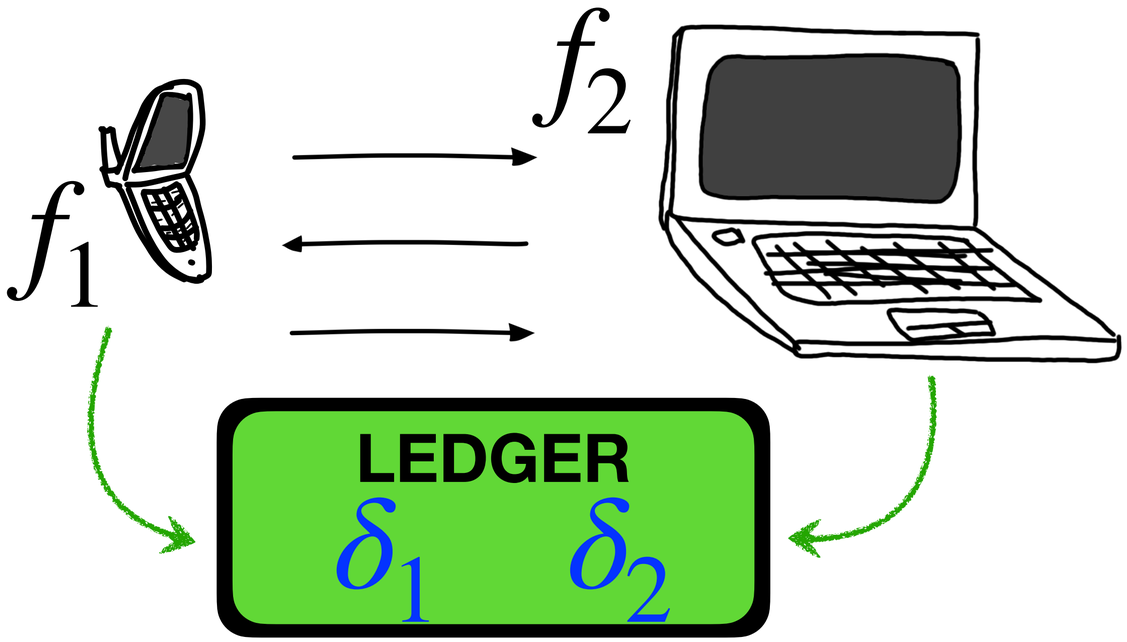


Towards a Solution

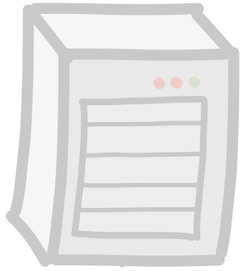


No ambiguity about what will see on waking up

Towards a Solution



No ambiguity about what  will see on waking up



Solves unanimous erasure, but kills privacy

General Problem Flavour

General Problem Flavour

- P2P channels convey information privately, but can’t be verified

General Problem Flavour

- P2P channels convey information privately, but can’t be verified
- Public channels can be verified but can’t convey private information

General Problem Flavour

- P2P channels convey information privately, but can't be verified
- Public channels can be verified but can't convey private information
- **Our approach**: use P2P channels to convey δ_1, δ_2 and ledger to achieve consensus on whether or not to use them.

General Problem Flavour

- P2P channels convey information privately, but can't be verified
- Public channels can be verified but can't convey private information
- **Our approach**: use P2P channels to convey δ_1, δ_2 and ledger to achieve consensus on whether or not to use them.
- Public and private values are linked via nonces of sigs created by *interleaved threshold signing*

ECDSA / Schnorr

ECDSA / Schnorr

Discrete logarithm based signatures

ECDSA / Schnorr

Discrete logarithm based signatures

Public key: $X = x \cdot G$

ECDSA / Schnorr

Discrete logarithm based signatures

Public key: $X = x \cdot G$

ECDSA / Schnorr

Discrete logarithm based signatures

Public key: $X = x \cdot G$

$X \in \mathbb{G}$ $x \in \mathbb{Z}_q$

ECDSA / Schnorr

Discrete logarithm based signatures

Public key: $X = x \cdot G$

$X \in \mathbb{G}$, $x \in \mathbb{Z}_q$

Signatures: (R, σ)

ECDSA / Schnorr

Discrete logarithm based signatures

Public key: $X = x \cdot G$

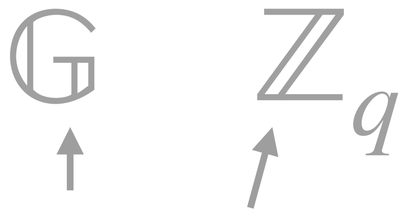
$X \in \mathbb{G}$, $x \in \mathbb{Z}_q$

Signatures: (R, σ)

ECDSA / Schnorr

Discrete logarithm based signatures

Public key: $X = x \cdot G$

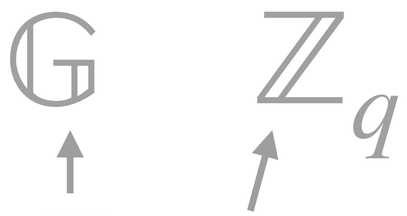


Signatures: (R, σ)

ECDSA / Schnorr

Discrete logarithm based signatures

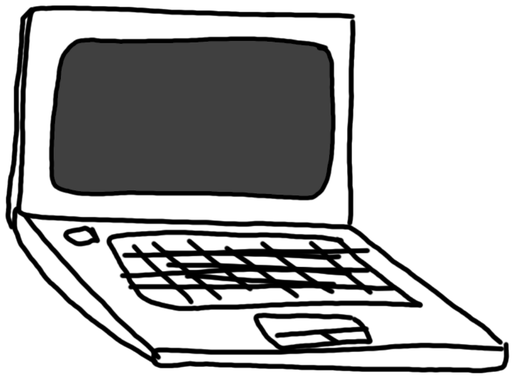
Public key: $X = x \cdot G$



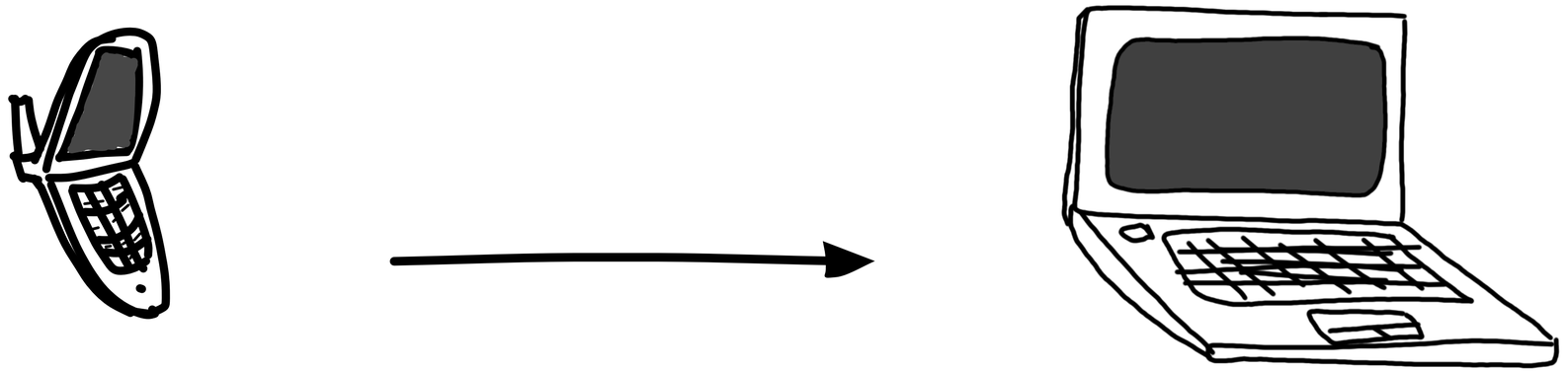
Signatures: (R, σ)

Random

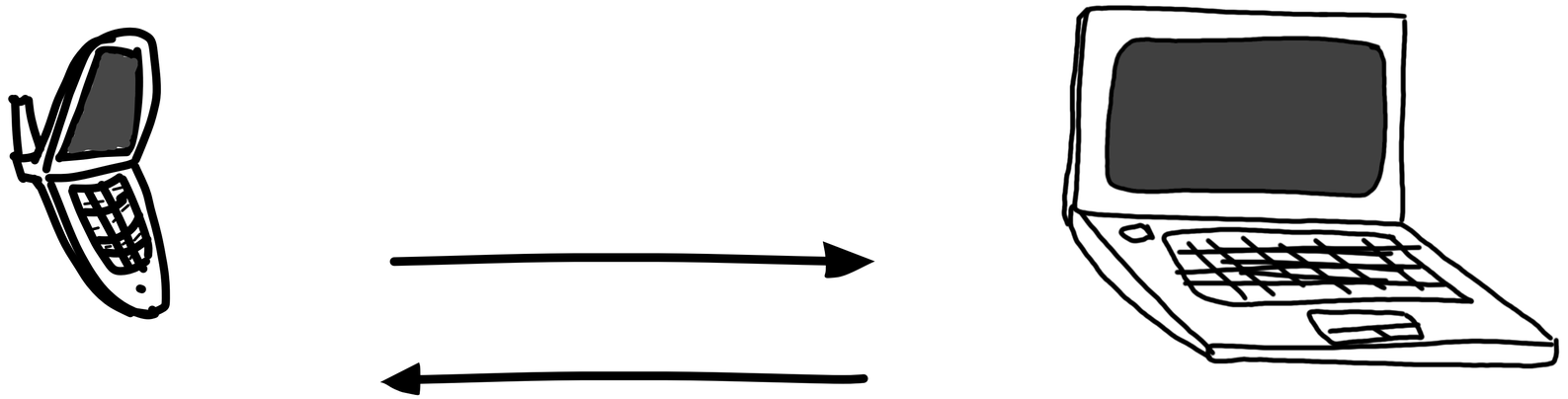
Threshold ECDSA/Schnorr



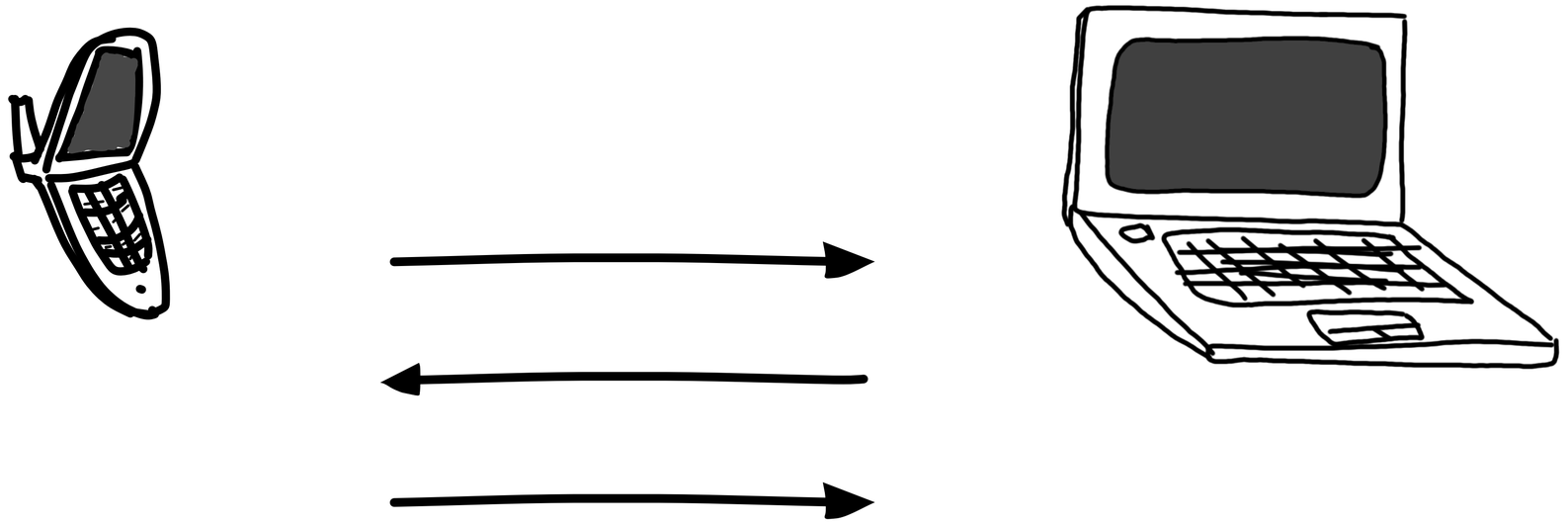
Threshold ECDSA/Schnorr



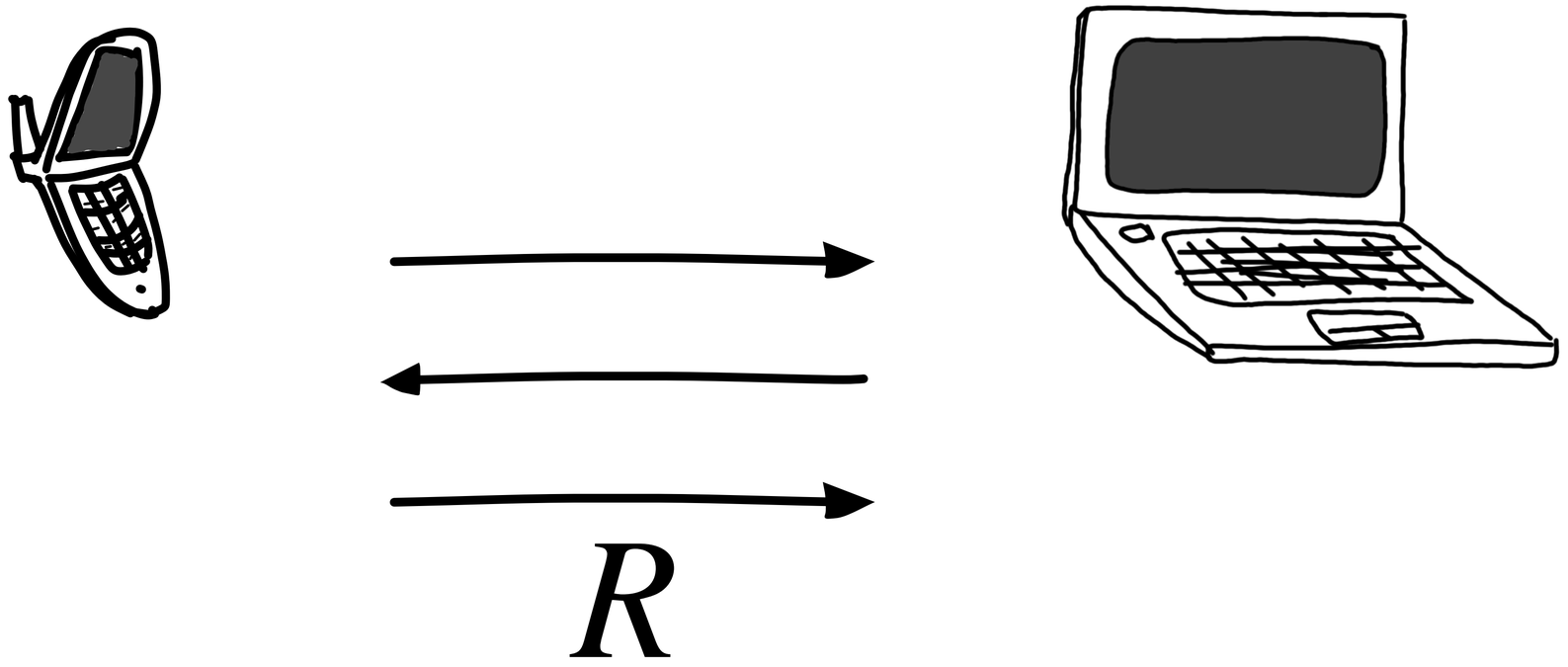
Threshold ECDSA/Schnorr



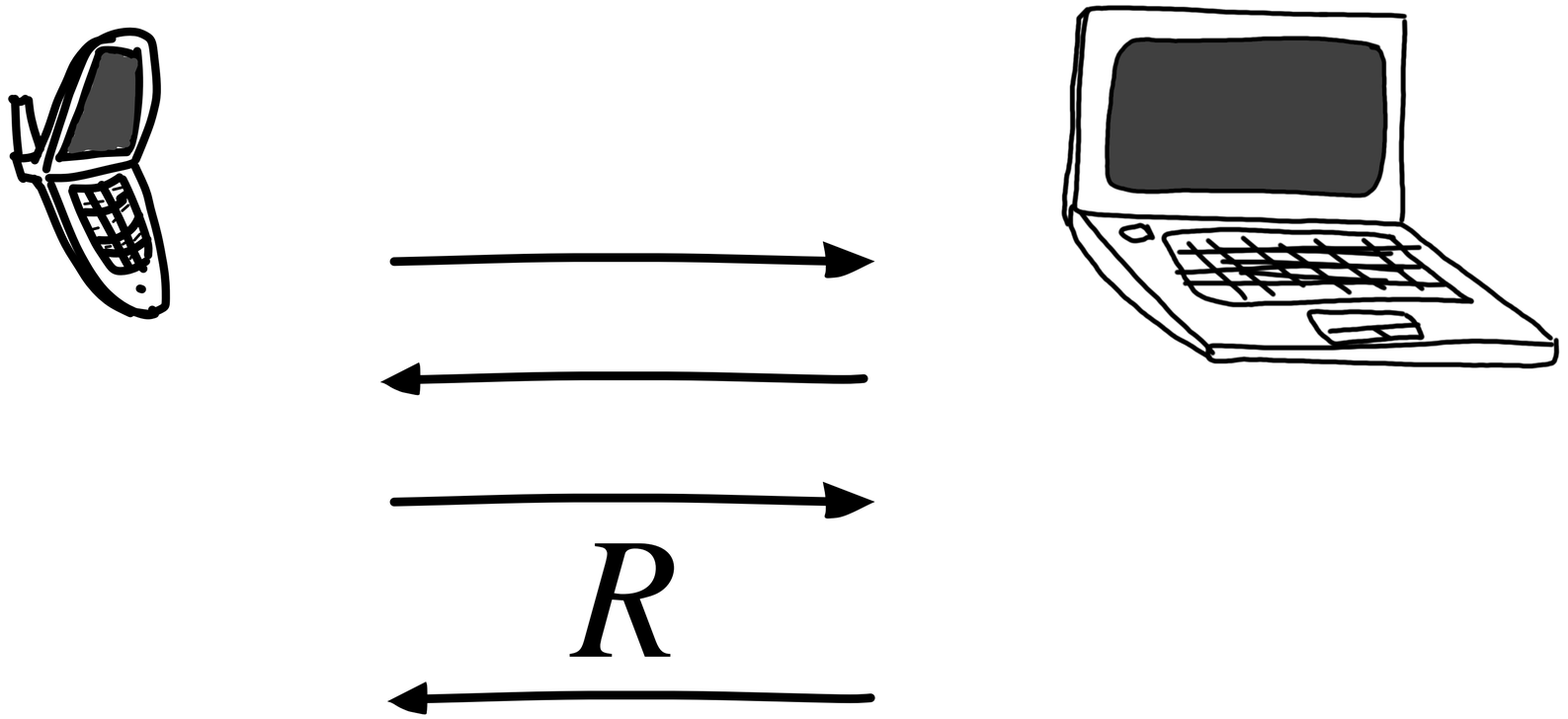
Threshold ECDSA/Schnorr



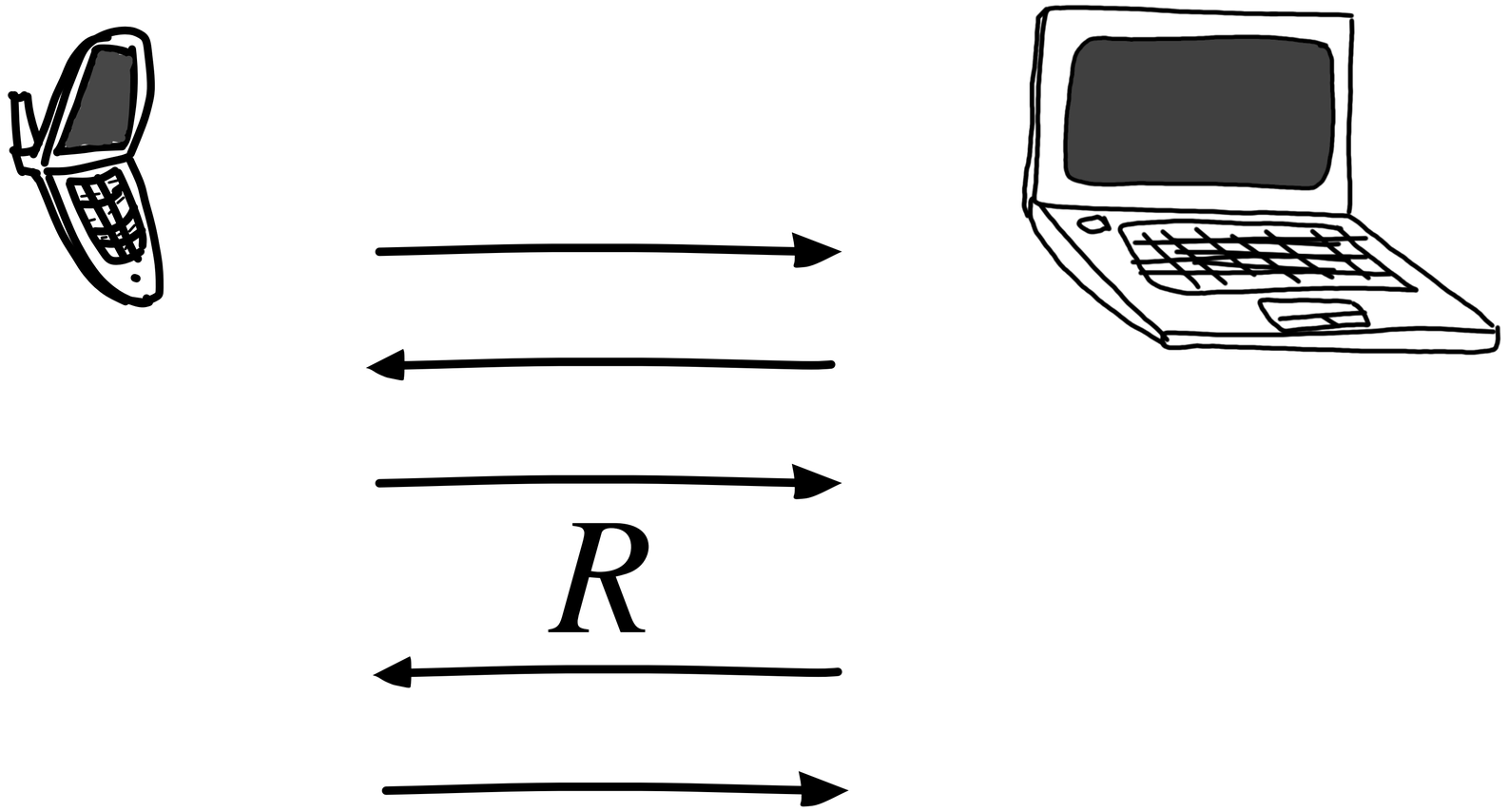
Threshold ECDSA/Schnorr



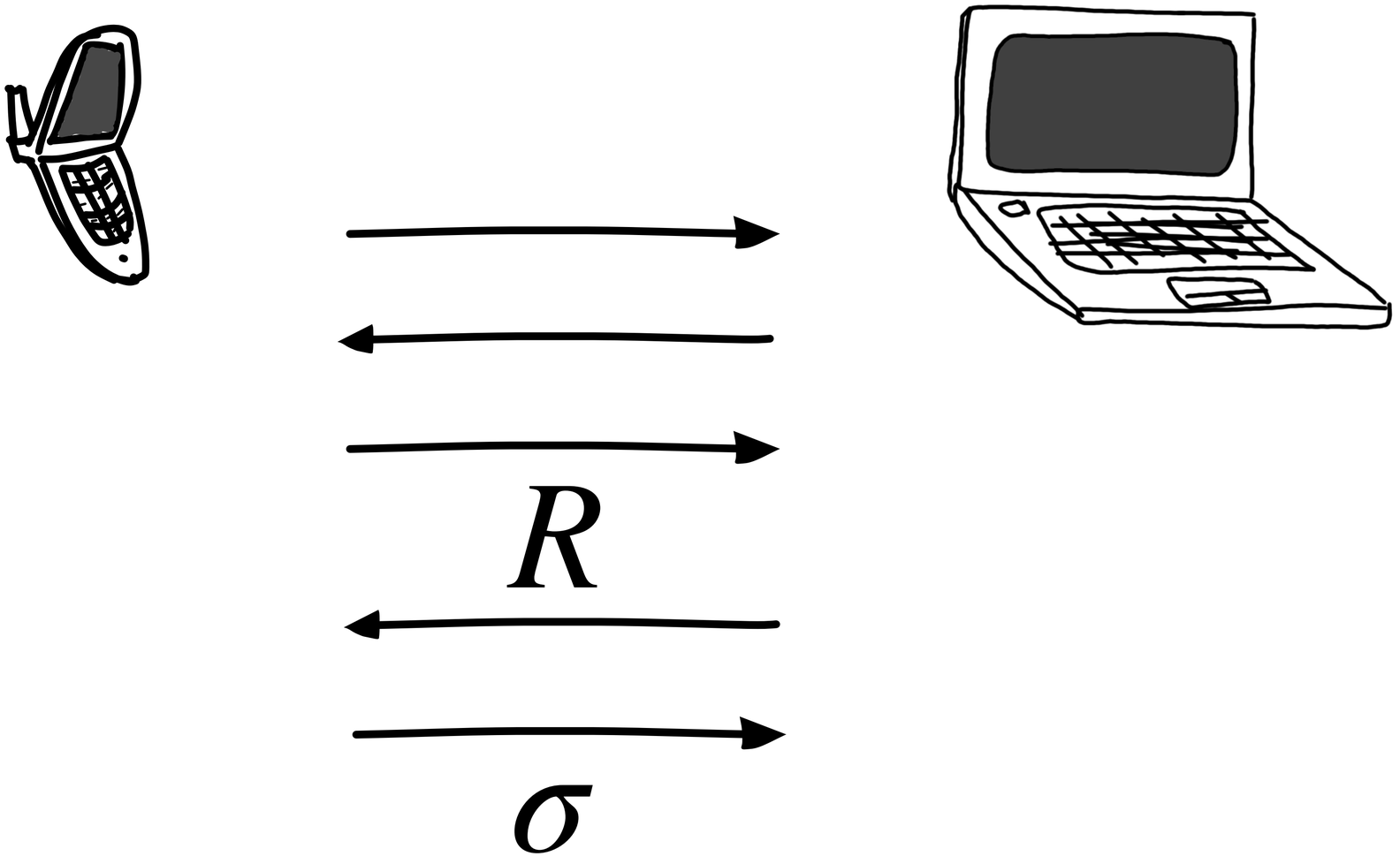
Threshold ECDSA/Schnorr



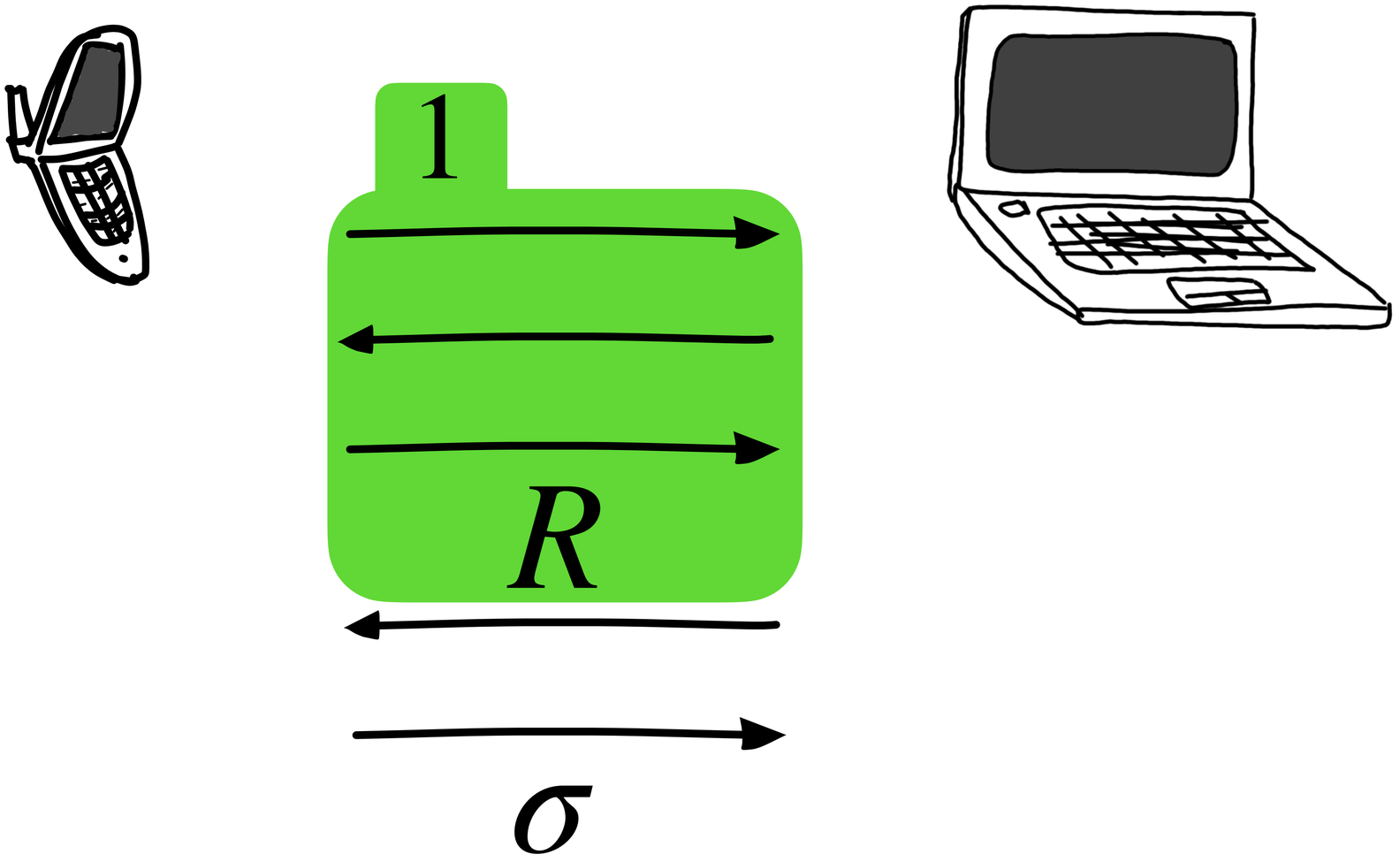
Threshold ECDSA/Schnorr



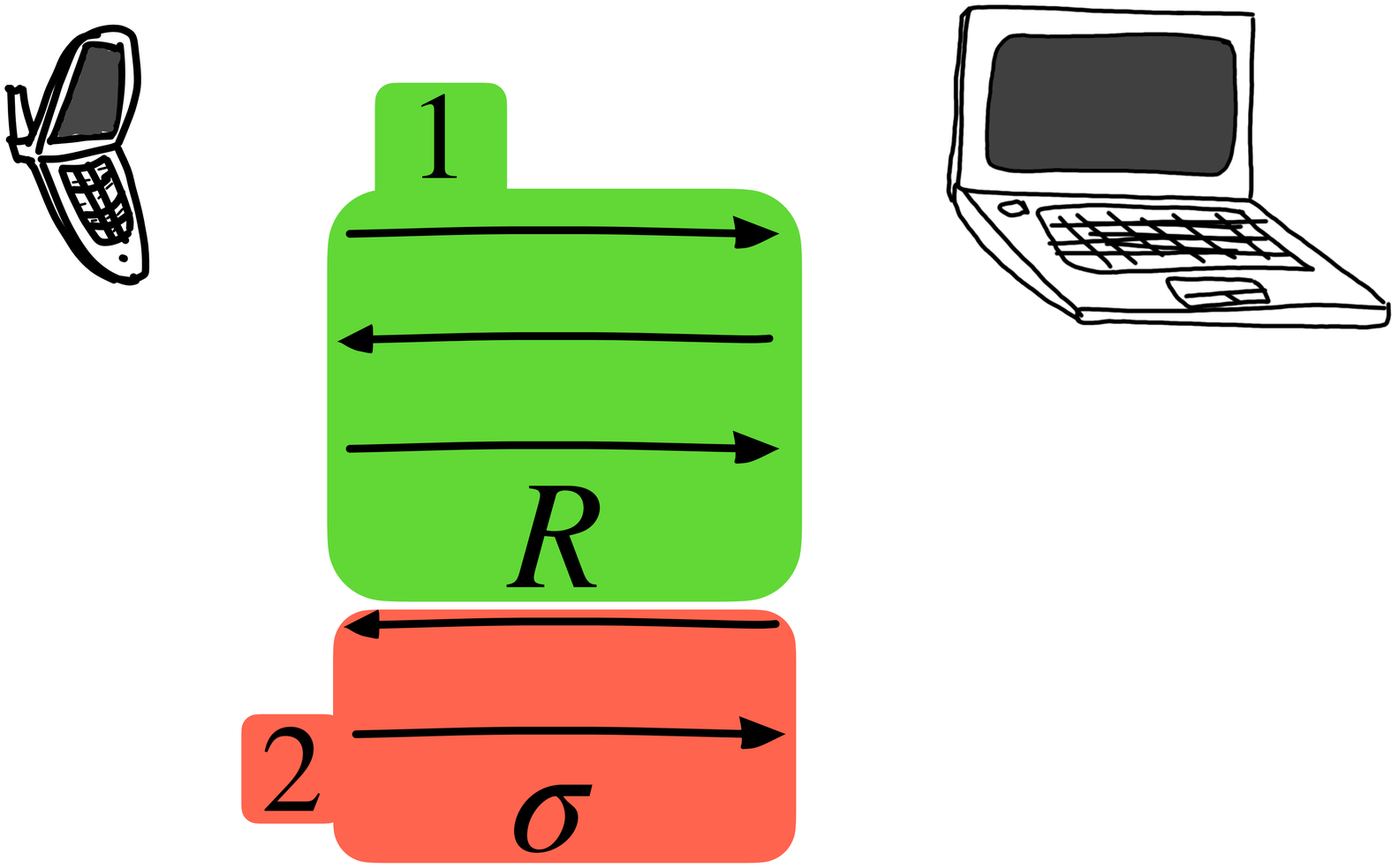
Threshold ECDSA/Schnorr



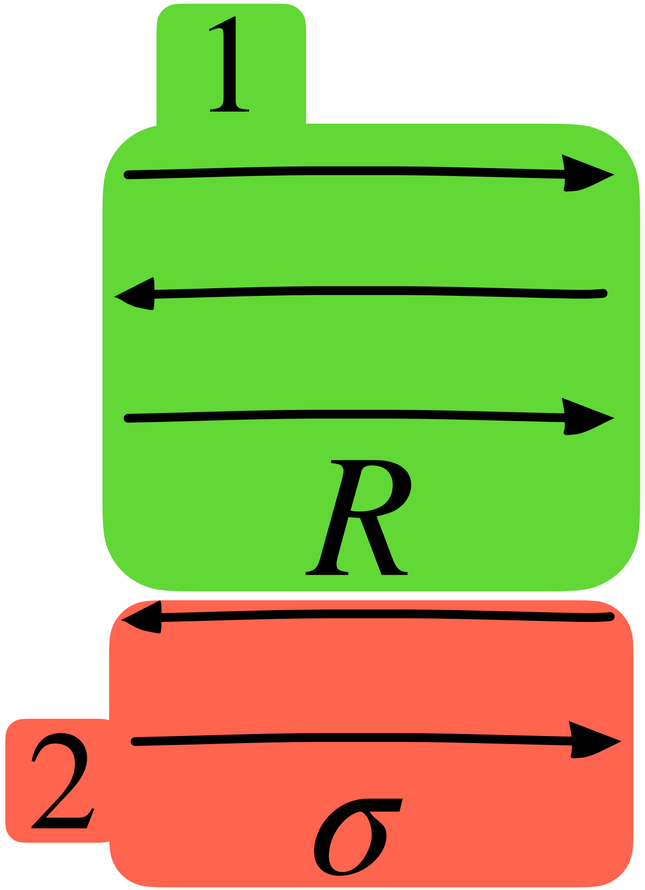
Threshold ECDSA/Schnorr



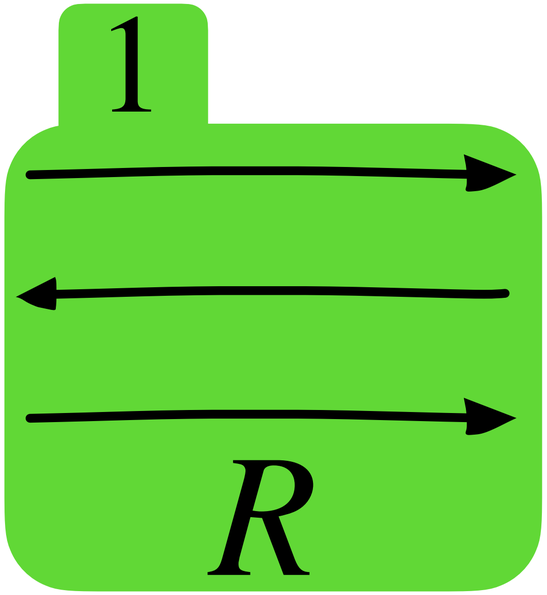
Threshold ECDSA/Schnorr



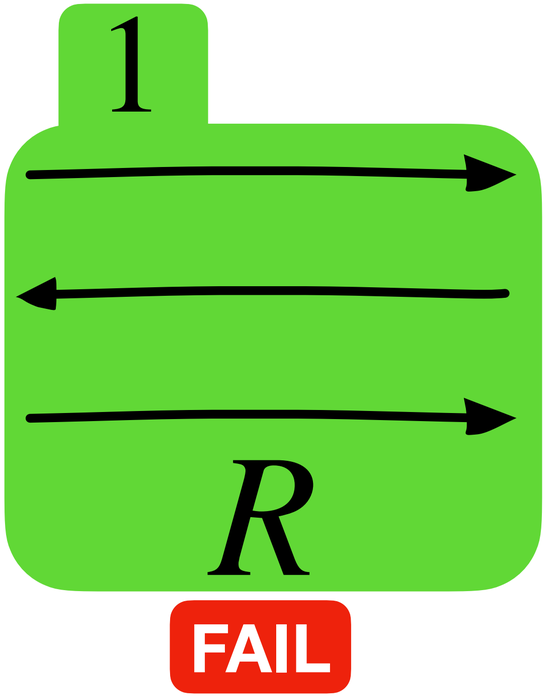
Threshold ECDSA/Schnorr



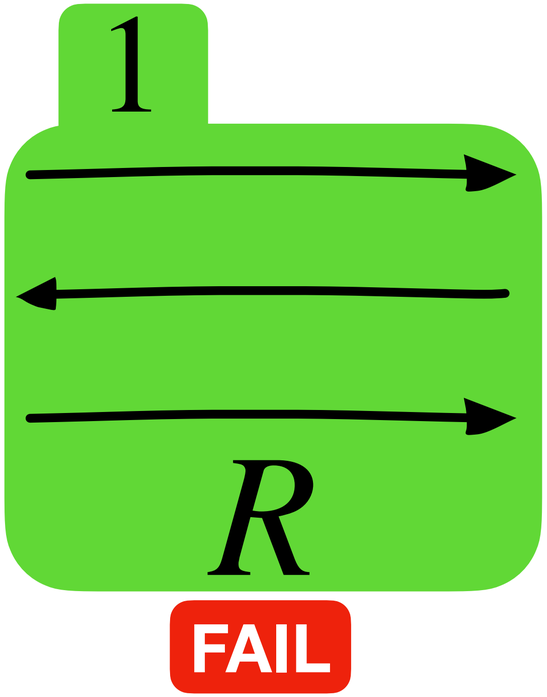
Threshold ECDSA/Schnorr



Threshold ECDSA/Schnorr

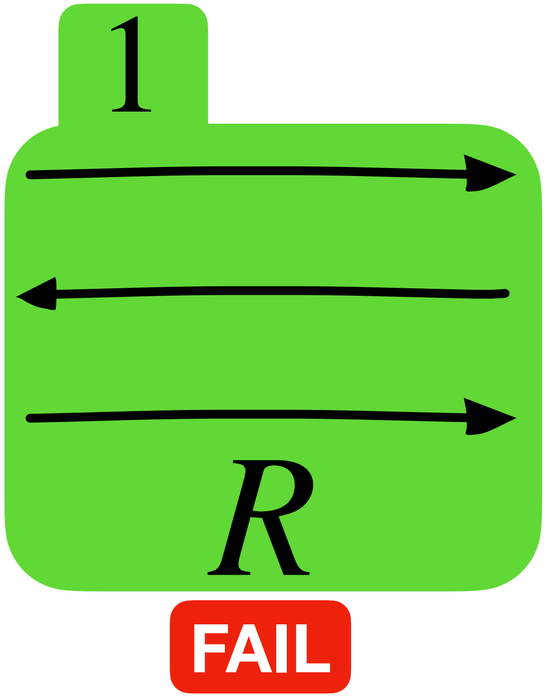


Threshold ECDSA/Schnorr



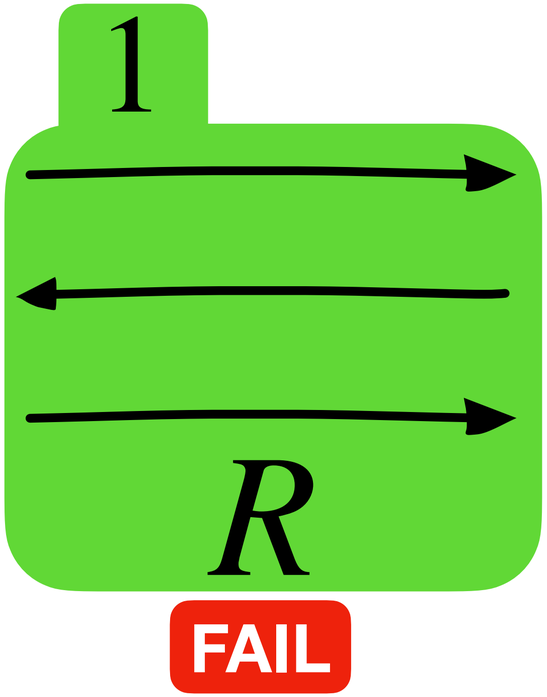
Signatures: (R, σ)

Threshold ECDSA/Schnorr



Signatures: (R, σ)

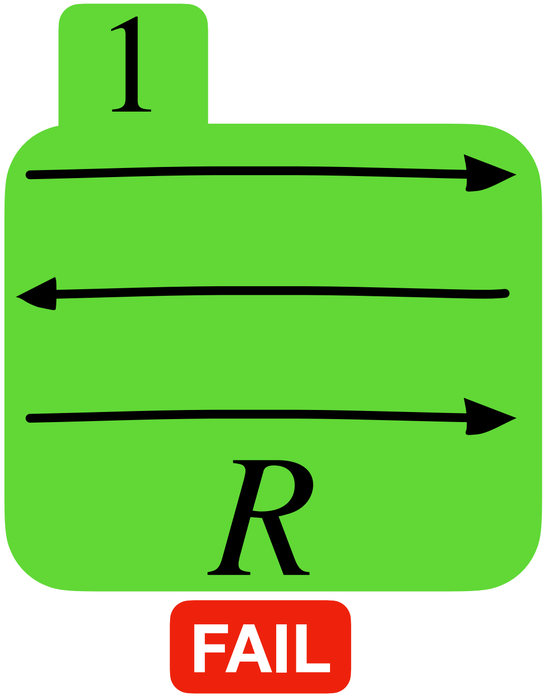
Threshold ECDSA/Schnorr



No advantage in computing this

Signatures: (R, σ)

Threshold ECDSA/Schnorr



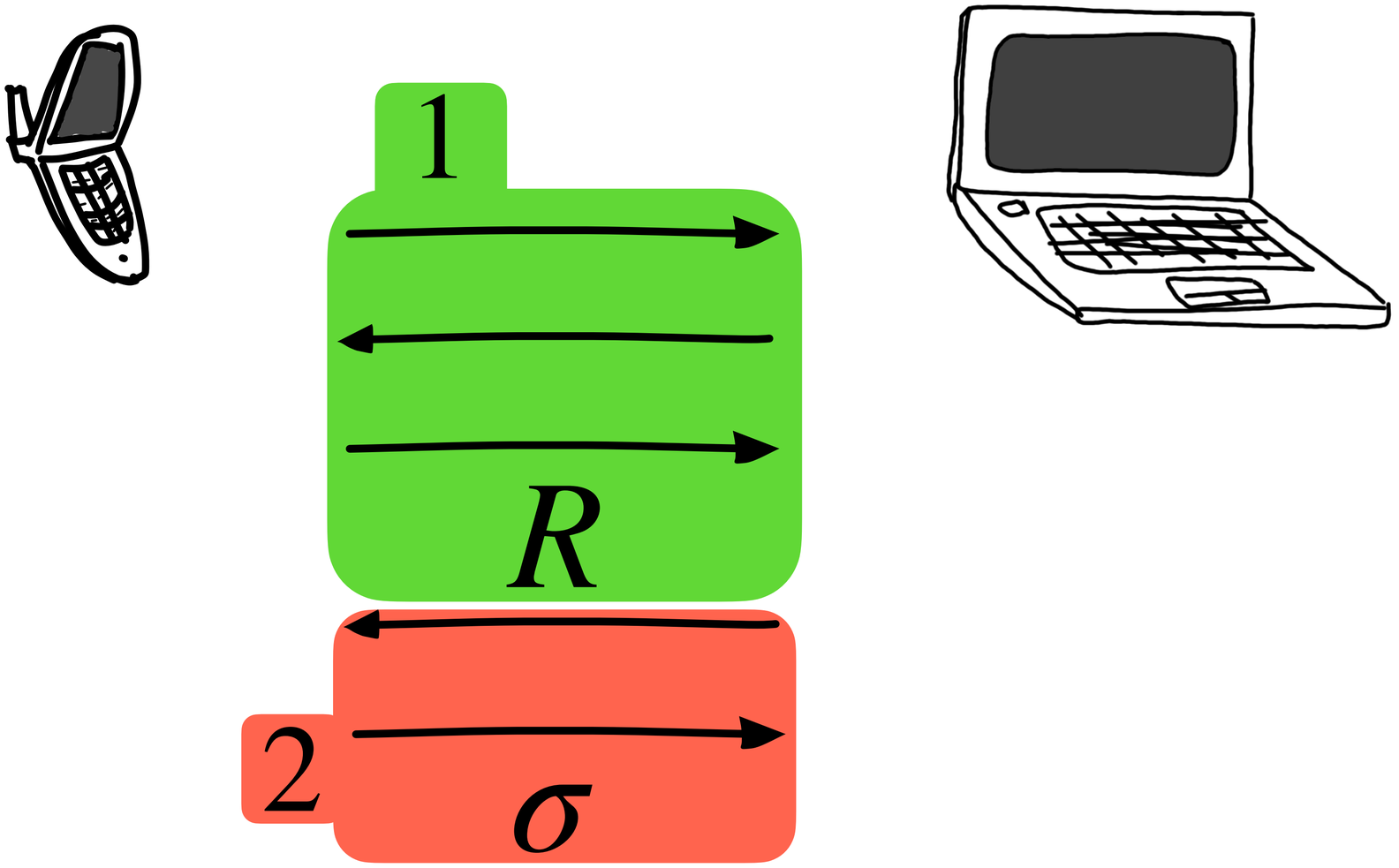
Achieved by most natural thresh Schnorr and ECDSA schemes

[DKLs19, GG18, LNR18, GJKR07]

No advantage in computing this

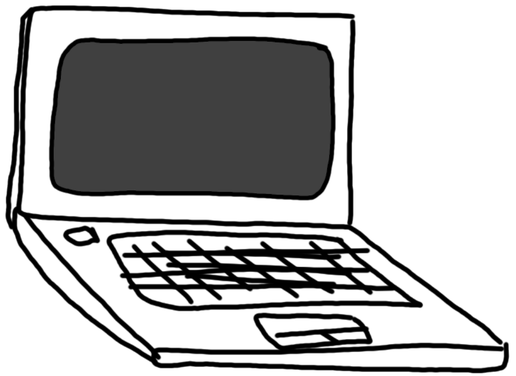
Signatures: (R, σ)

Threshold ECDSA/Schnorr



Interleaved Threshold Signing

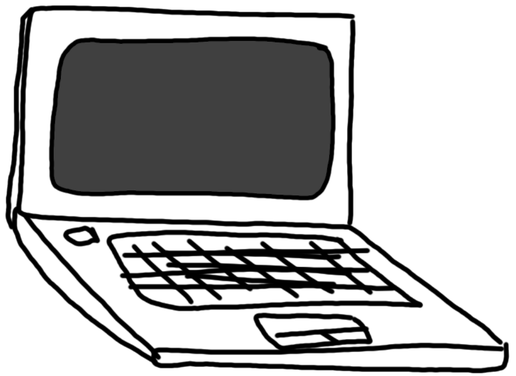
1 R



2 σ

Interleaved Threshold Signing

$1 \quad R$



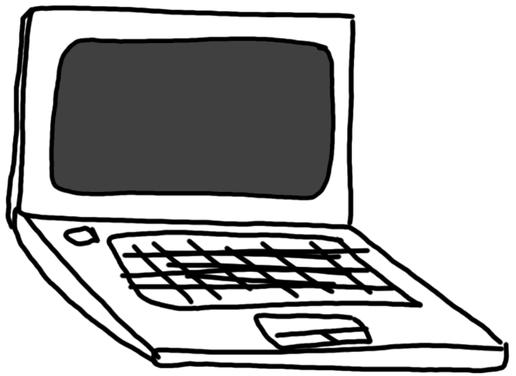
$\delta \longleftrightarrow \delta$

$2 \quad \sigma$

Interleaved Threshold Signing

Interleaved Threshold Signing

$1 \quad R$



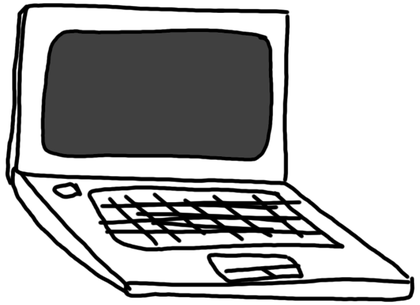
$\delta \quad \mathrm{sig}_{\delta,R}$

$2 \quad \sigma$

Interleaved Threshold Signing

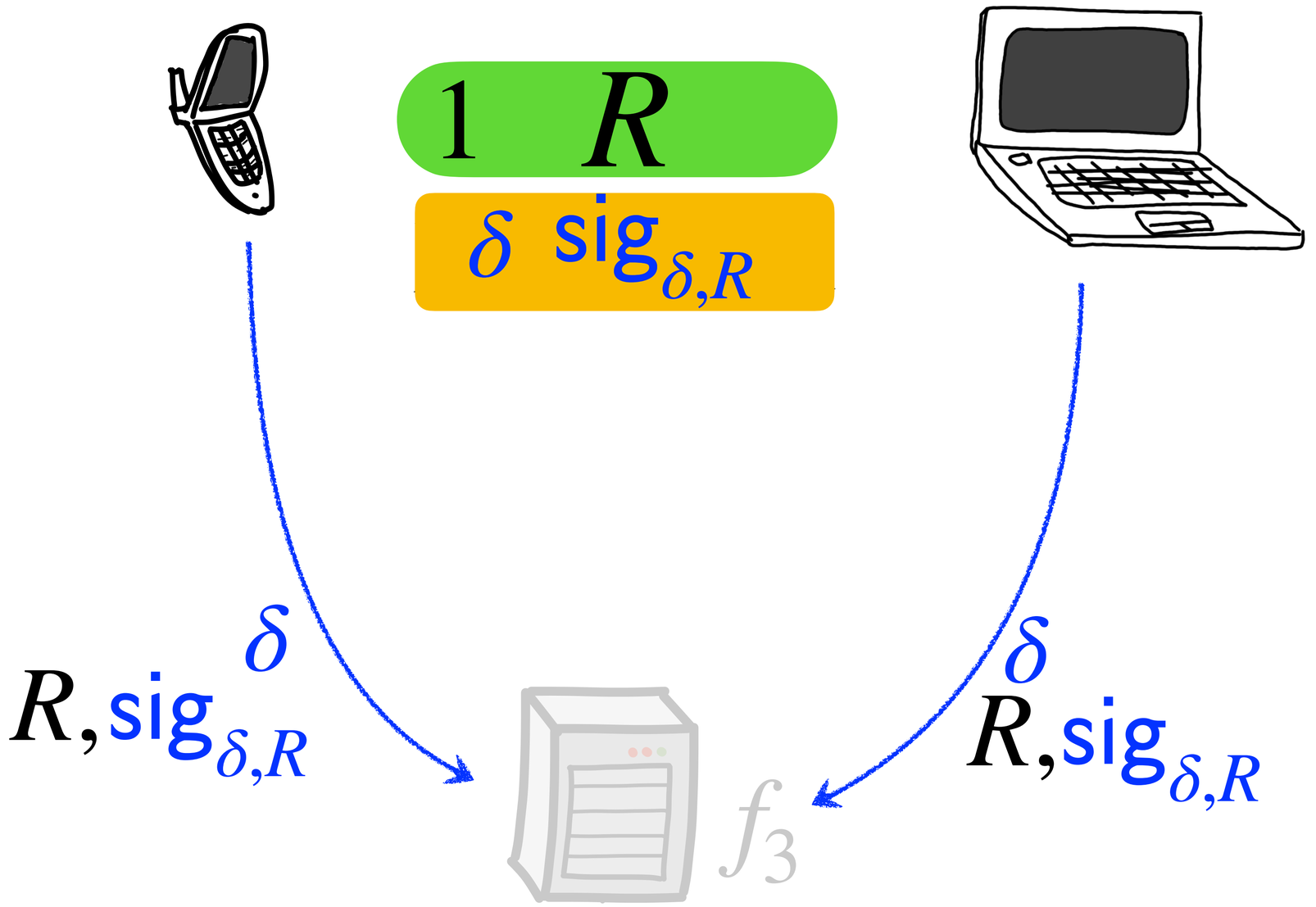
$1 \quad R$

$\delta \quad \mathrm{sig}_{\delta,R}$

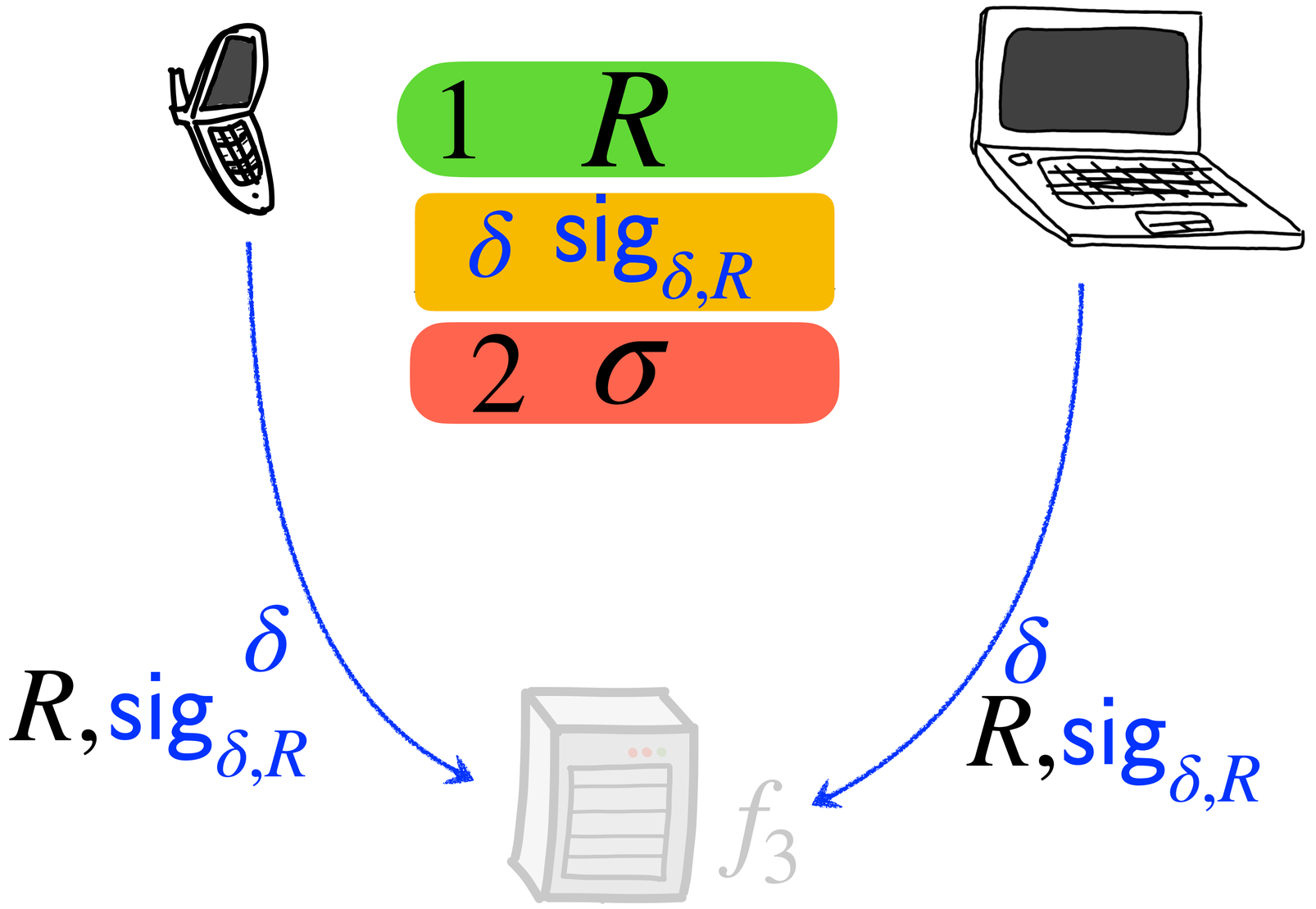


f_3

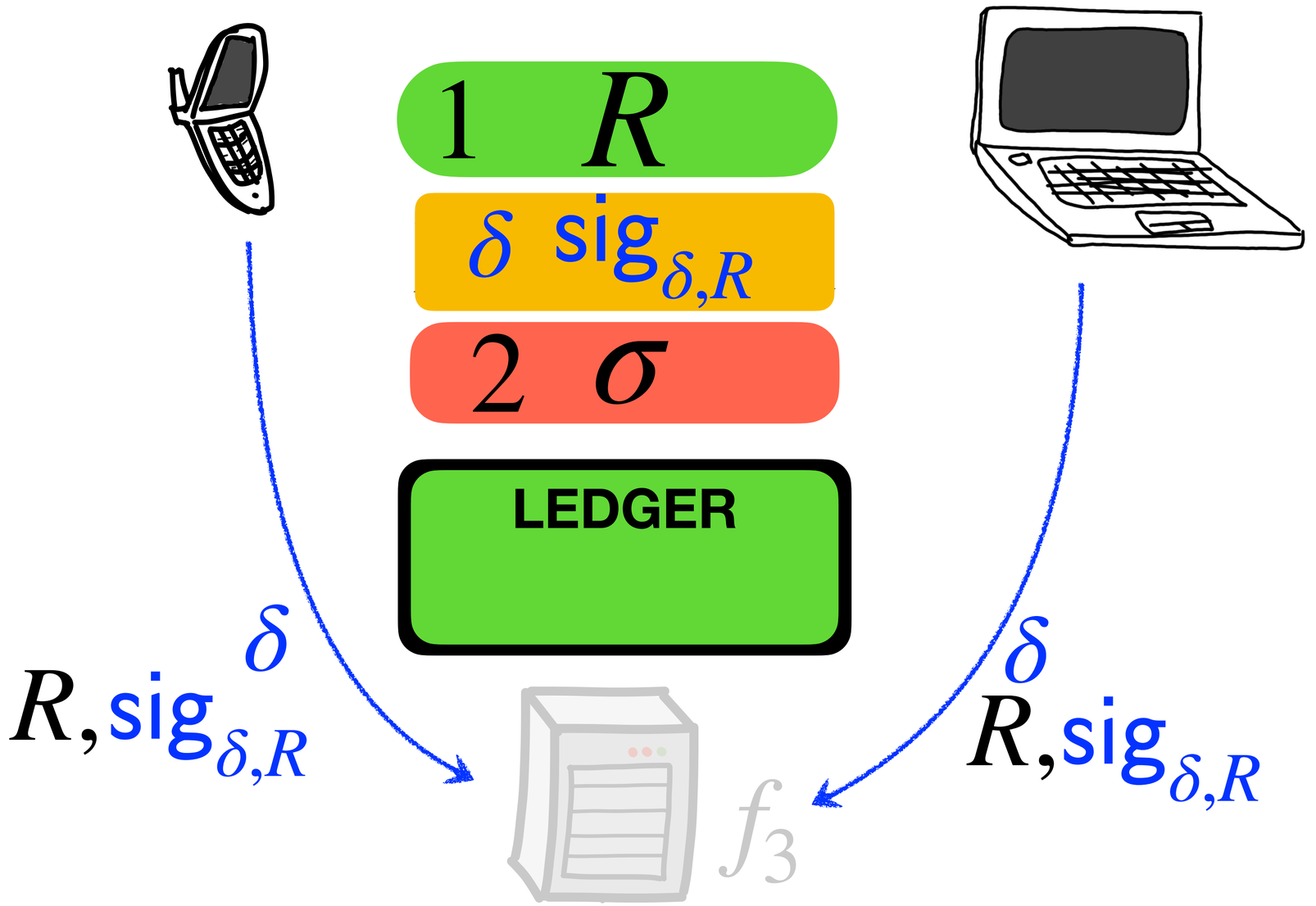
Interleaved Threshold Signing



Interleaved Threshold Signing

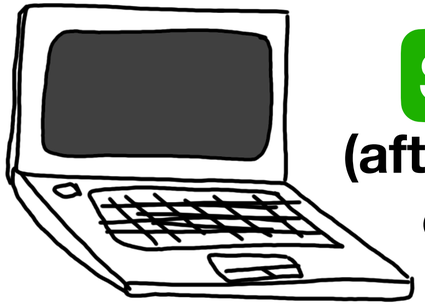


Interleaved Threshold Signing



Interleaved Threshold Signing

SUCCESS
(after tx,R appears on LEDGER)



SUCCESS
(after tx,R appears on LEDGER)

$1 \quad R$

$\delta \quad \text{sig}_{\delta,R}$

$2 \quad \sigma$

LEDGER
tx, R, σ

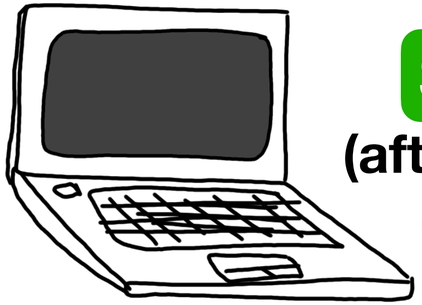
δ
$R, \text{sig}_{\delta,R}$

δ
$R, \text{sig}_{\delta,R}$

f_3

Interleaved Threshold Signing

SUCCESS
(after tx,R appears on LEDGER)



SUCCESS
(after tx,R appears on LEDGER)

$1 \quad R$

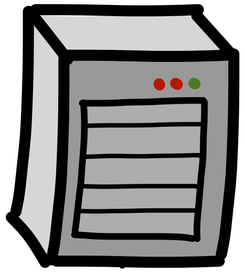
$\delta \quad \mathrm{sig}_{\delta,R}$

$2 \quad \sigma$

LEDGER
tx, R, σ

δ
$R, \mathrm{sig}_{\delta,R}$

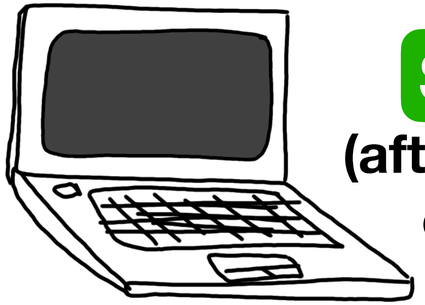
δ
$R, \mathrm{sig}_{\delta,R}$



f_3

Interleaved Threshold Signing

SUCCESS
(after tx,R appears on LEDGER)



SUCCESS
(after tx,R appears on LEDGER)

$1 \quad R$

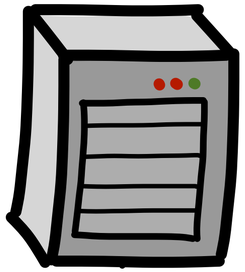
$\delta \quad \mathrm{sig}_{\delta,R}$

$2 \quad \sigma$

LEDGER
tx, R, σ

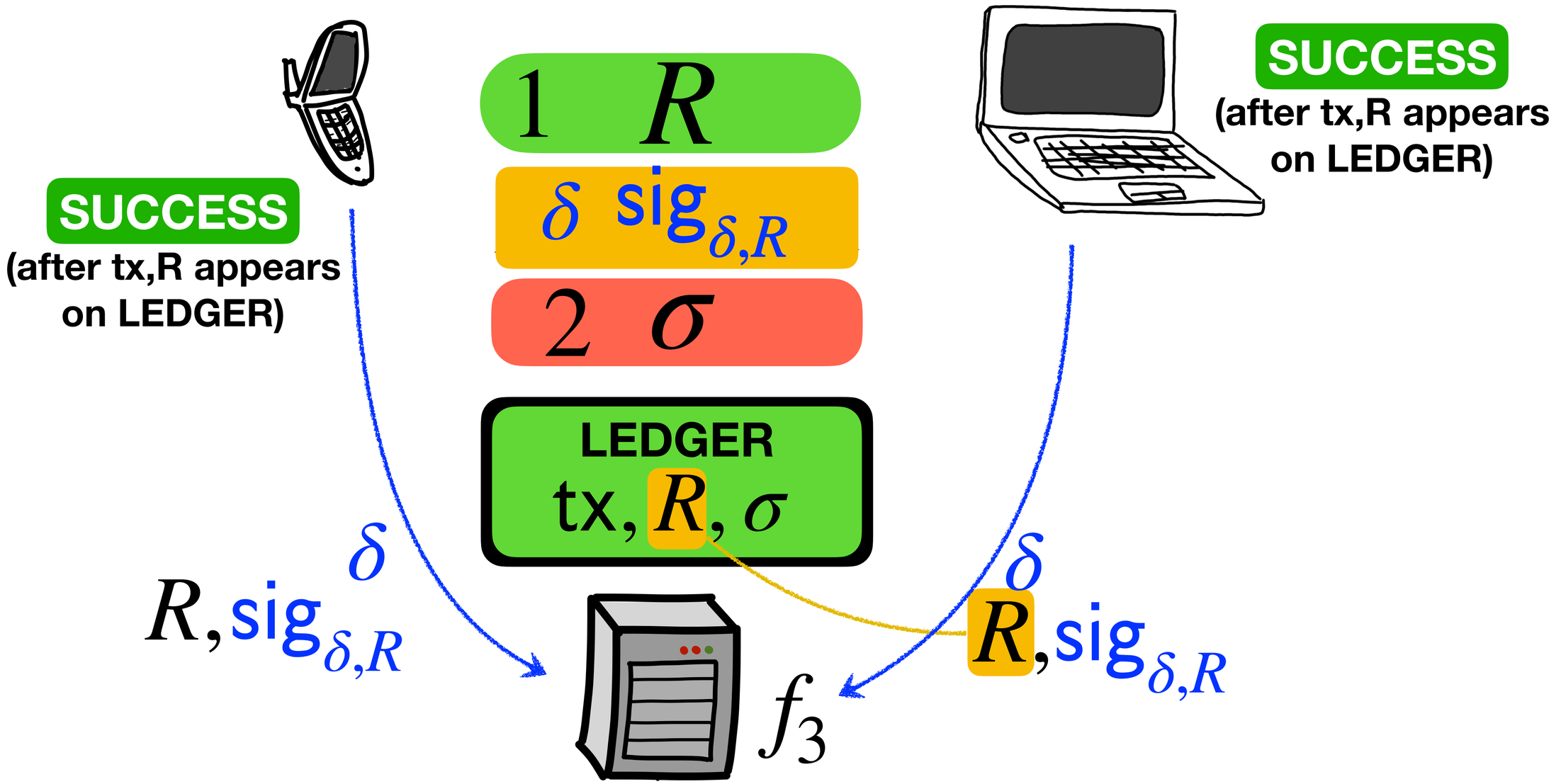
δ
$R, \mathrm{sig}_{\delta,R}$

δ
$R, \mathrm{sig}_{\delta,R}$

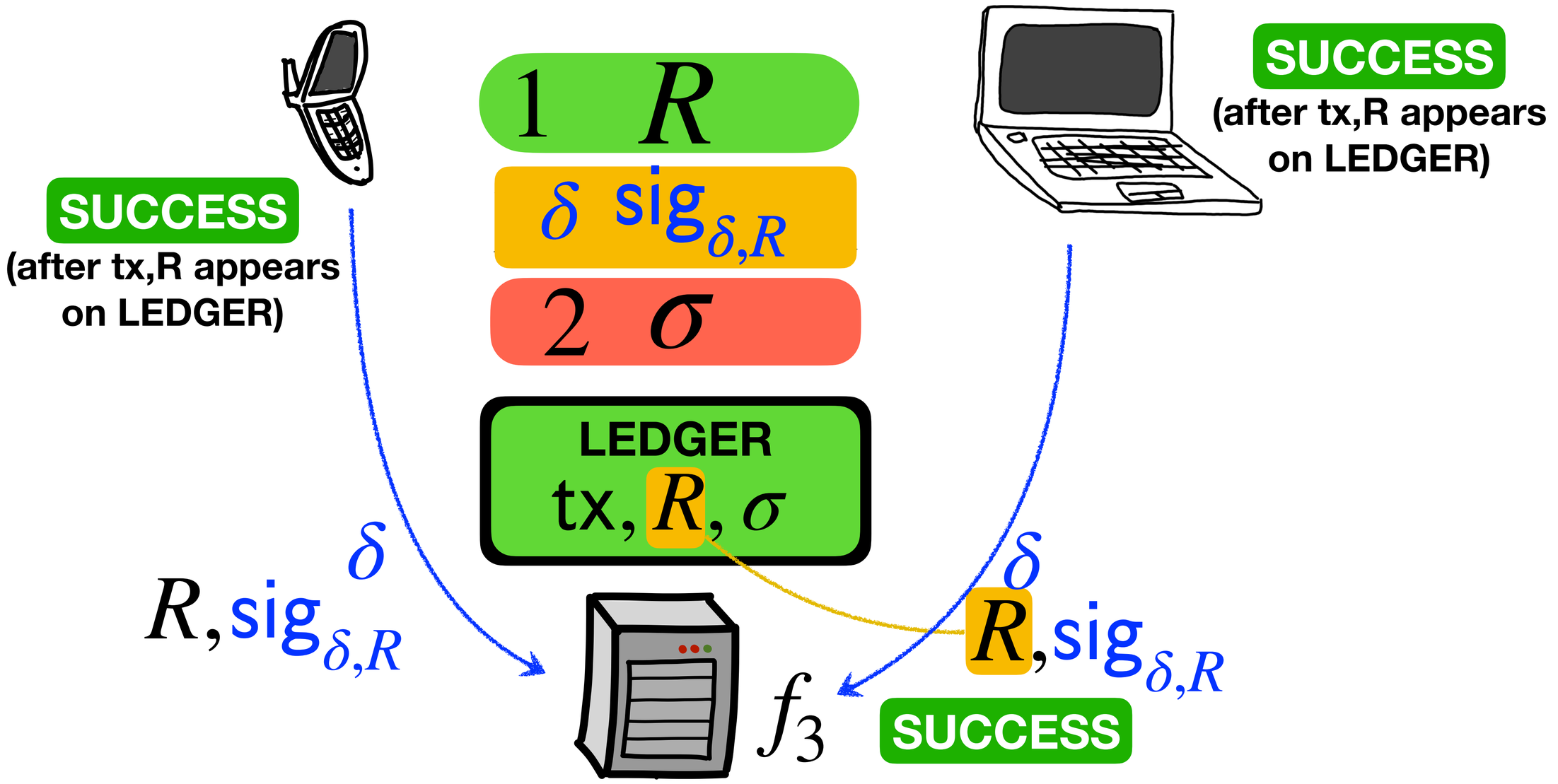


f_3

Interleaved Threshold Signing

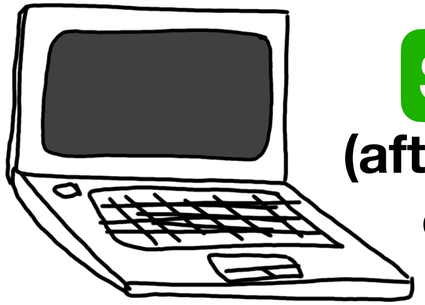


Interleaved Threshold Signing



Interleaved Threshold Signing

SUCCESS
(after tx,R appears on LEDGER)



SUCCESS
(after tx,R appears on LEDGER)

1 R

δ $\text{sig}_{\delta,R}$

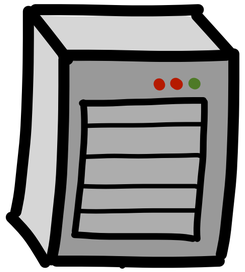
2 σ

LEDGER
tx, R, σ

Native: this was going on the ledger anyway

δ
$R, \text{sig}_{\delta,R}$

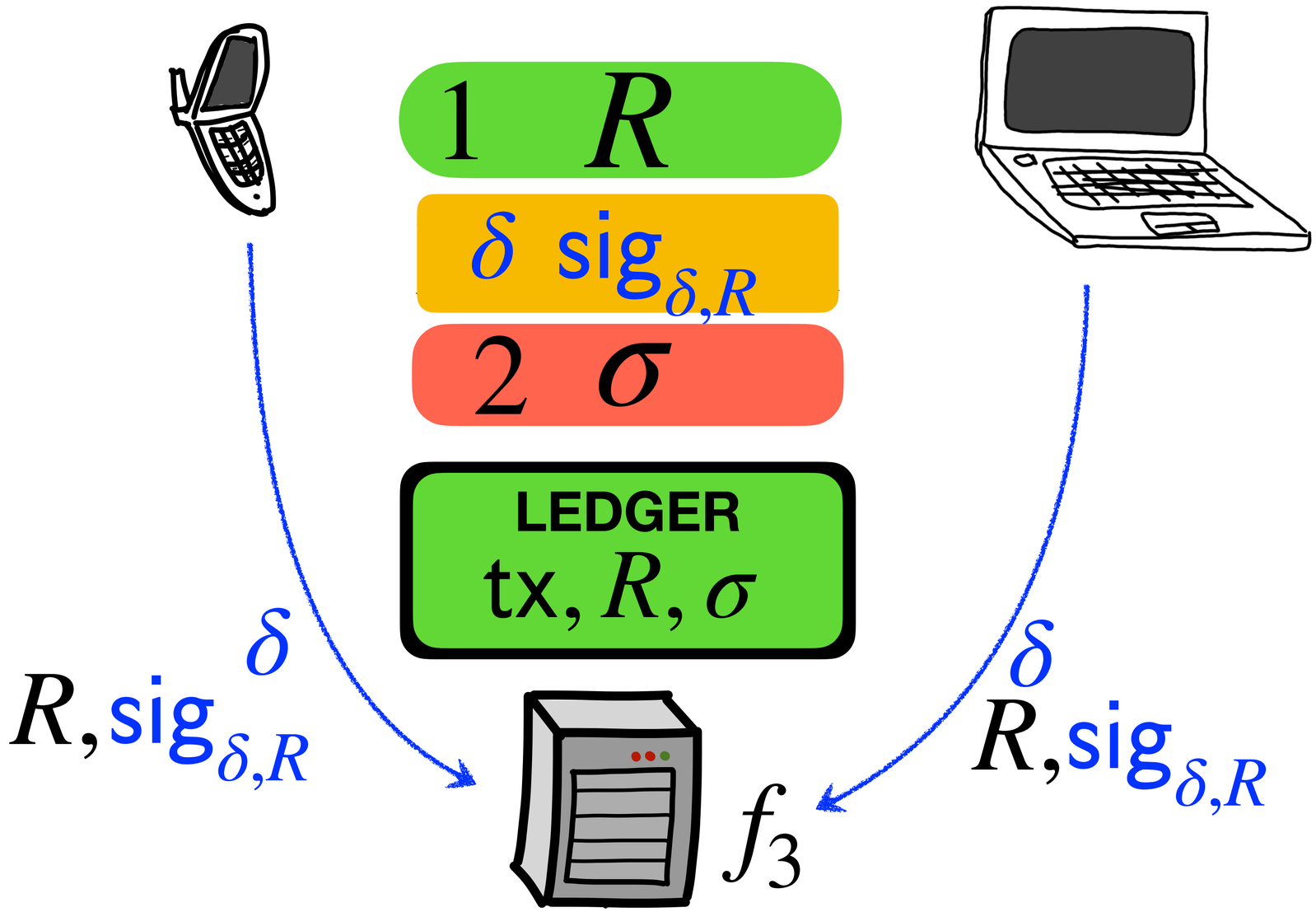
δ
$R, \text{sig}_{\delta,R}$



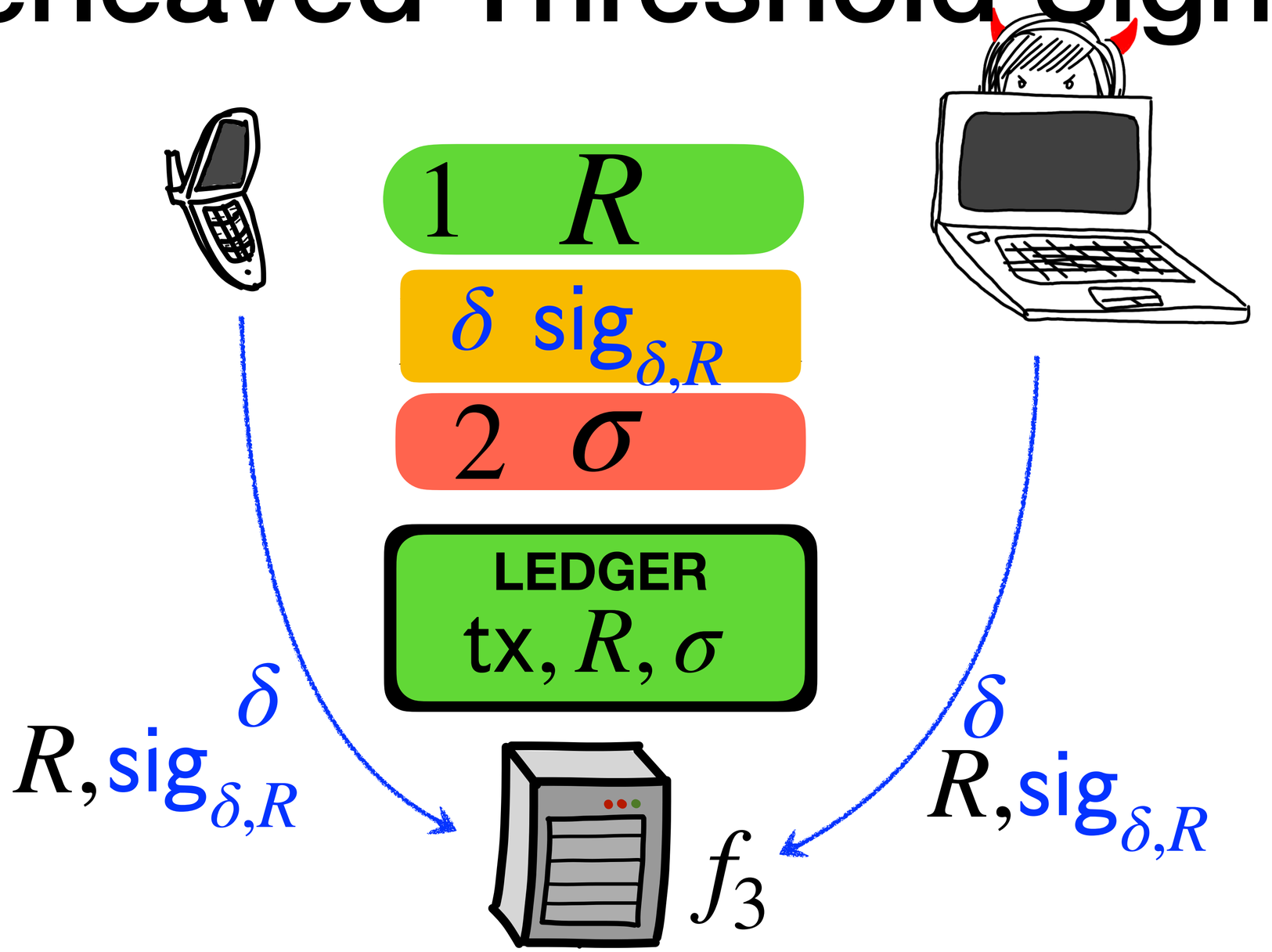
f_3

SUCCESS

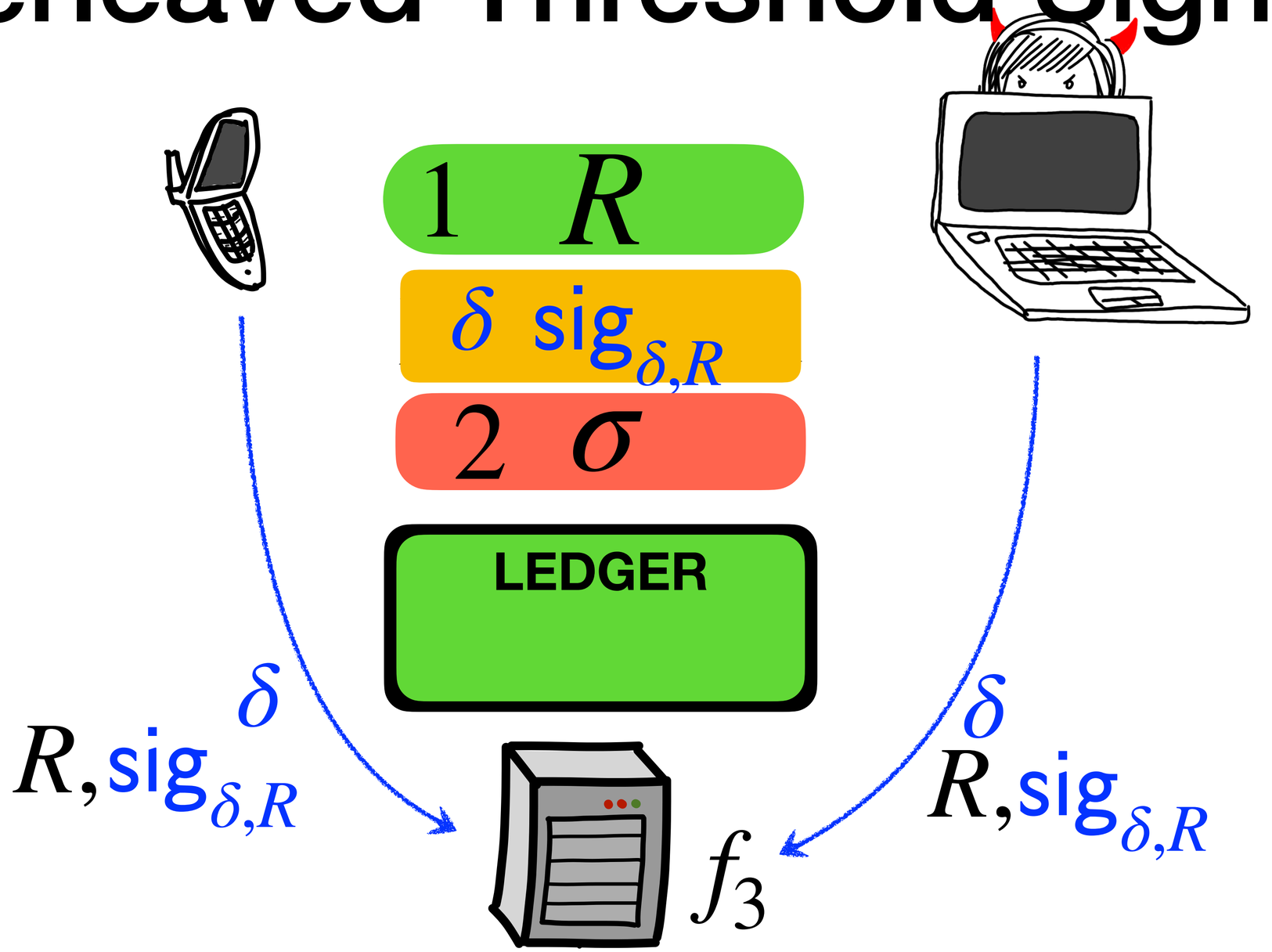
Interleaved Threshold Signing



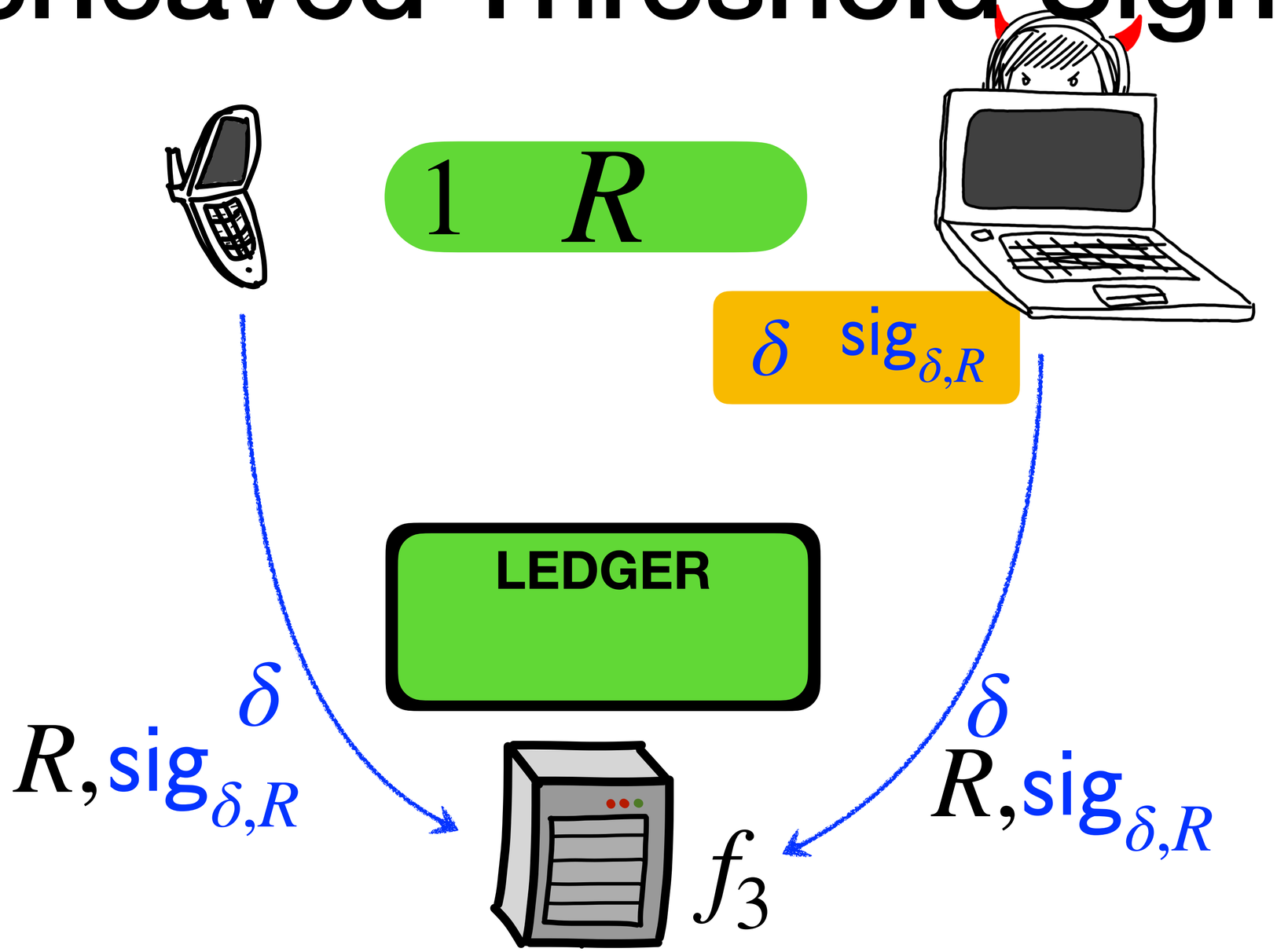
Interleaved Threshold Signing



Interleaved Threshold Signing



Interleaved Threshold Signing

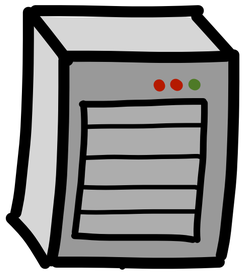


Interleaved Threshold Signing

$1 \quad R$

$\delta \quad \mathrm{sig}_{\delta,R}$

LEDGER



f_3

δ

$R, \mathrm{sig}_{\delta,R}$

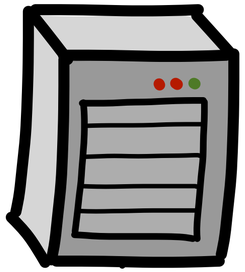
Interleaved Threshold Signing

$1 \quad R$

$\delta \quad \text{sig}_{\delta,R}$

LEDGER will never receive valid signature under *R* (Phase 2 never run)

LEDGER



f_3

δ

$R, \text{sig}_{\delta,R}$

Interleaved Threshold Signing

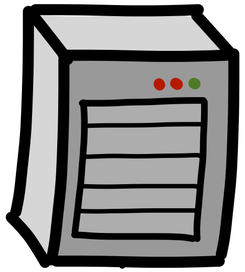
$1 \quad R$

$\delta \quad \text{sig}_{\delta,R}$

LEDGER will never receive valid signature under R (Phase 2 never run)

Reuse negligibly likely

LEDGER



f_3

δ

$R, \text{sig}_{\delta,R}$

Interleaved Threshold Signing

$1 \quad R$

$\delta \quad \text{sig}_{\delta,R}$

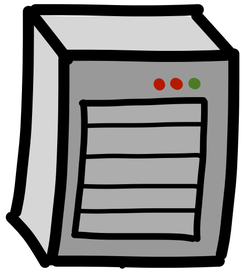
LEDGER will never receive valid signature under R (Phase 2 never run)

Reuse negligibly likely

LEDGER

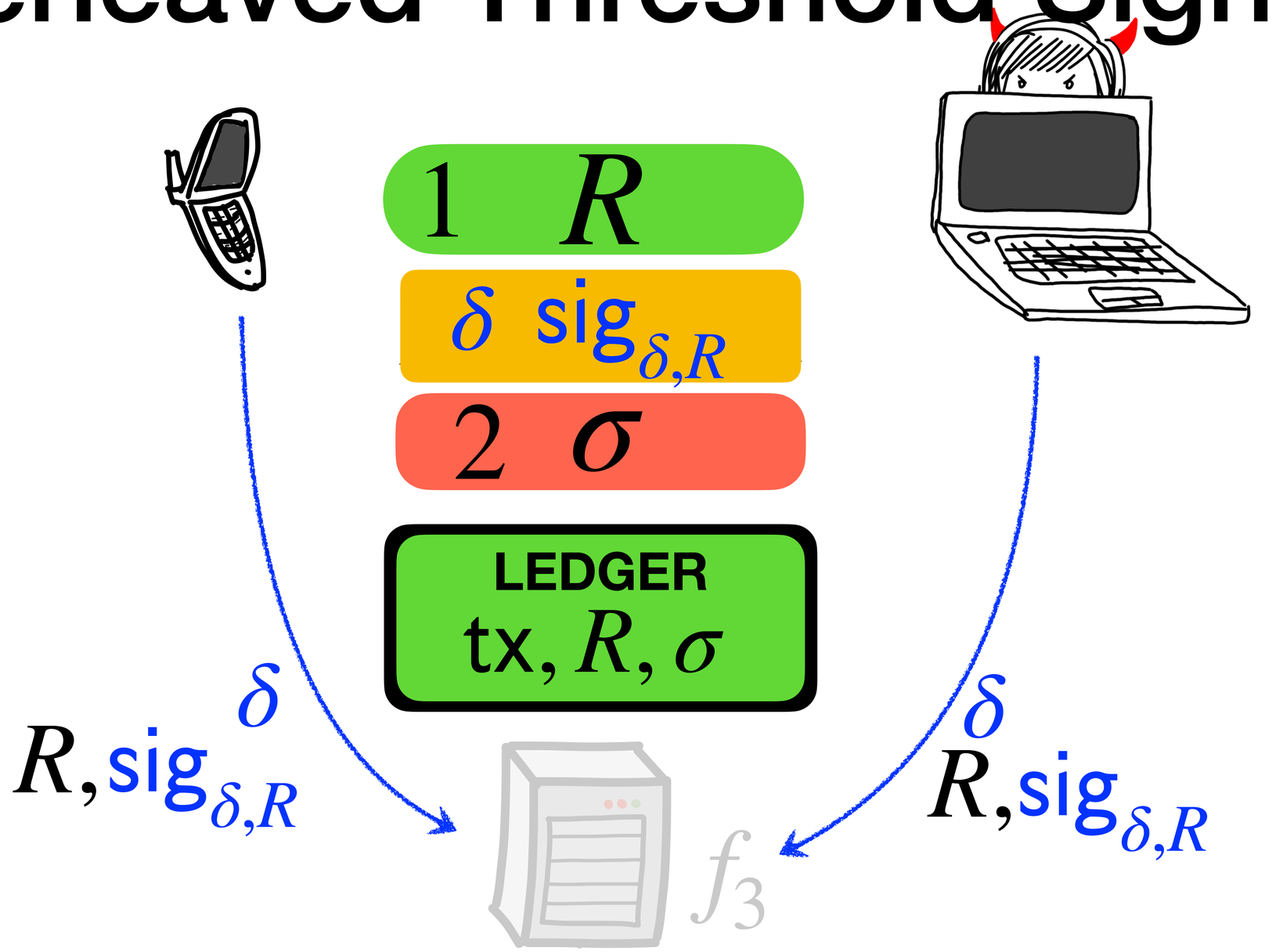
Useless

δ
$R, \text{sig}_{\delta,R}$

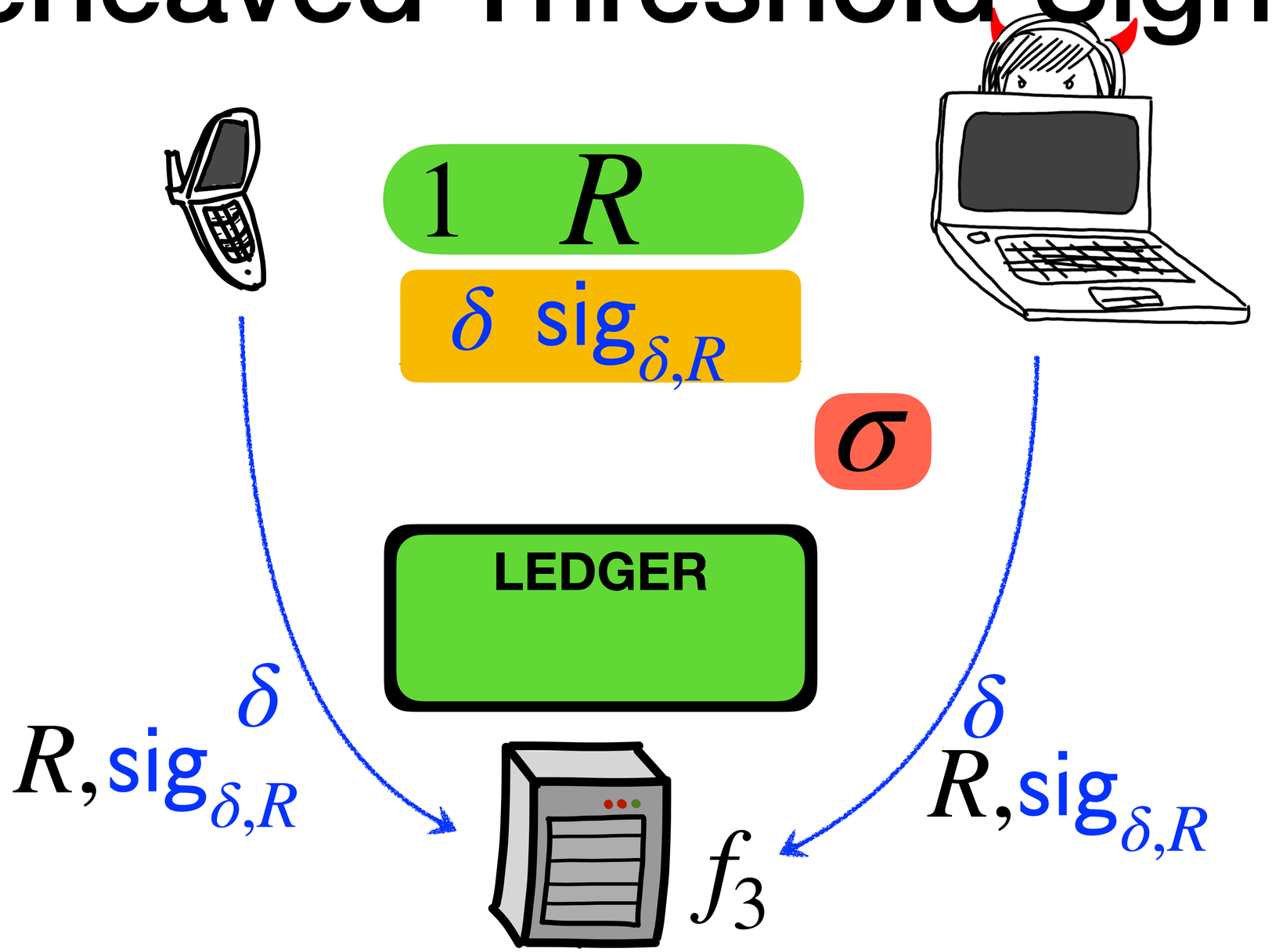


f_3

Interleaved Threshold Signing



Interleaved Threshold Signing



Interleaved Threshold Signing

$1 \quad R$

$\delta \;\; \text{sig}_{\delta,R}$

σ

Case 1:

never posts

tx, R, σ to ledger

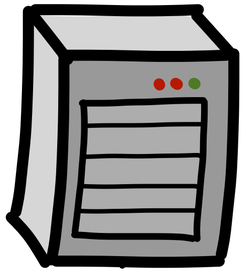
LEDGER

δ

$R, \text{sig}_{\delta,R}$

δ

$R, \text{sig}_{\delta,R}$



f_3

Interleaved Threshold Signing

$1 \quad R$

$\delta \;\; \text{sig}_{\delta,R}$

σ

Case 1:

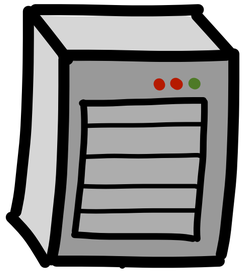
never posts

tx, R, σ to ledger

LEDGER

δ
$R, \text{sig}_{\delta,R}$

Never used



f_3

δ
$R, \text{sig}_{\delta,R}$

Interleaved Threshold Signing

$1 \quad R$

$\delta \;\; \text{sig}_{\delta,R}$

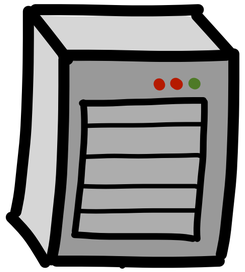
σ

Case 2:

does post tx, R, σ to ledger

LEDGER
tx, R, σ

δ
$R, \text{sig}_{\delta,R}$



f_3

δ
$R, \text{sig}_{\delta,R}$

Interleaved Threshold Signing

SUCCESS

$1 \quad R$

$\delta \;\; \text{sig}_{\delta,R}$

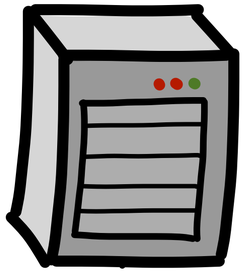
σ

Case 2:

does post tx, R, σ to ledger

LEDGER
tx, R, σ

δ
$R, \text{sig}_{\delta,R}$



f_3

δ
$R, \text{sig}_{\delta,R}$

SUCCESS

Interleaved Threshold Signing

SUCCESS

$1 \quad R$

$\delta \quad \text{sig}_{\delta,R}$

σ

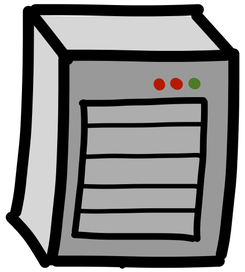
Case 2:

does post tx, R, σ to ledger

Unanimous erasure

LEDGER
tx, R, σ

δ
$R, \text{sig}_{\delta,R}$



f_3

δ
$R, \text{sig}_{\delta,R}$

SUCCESS

Implementation

Implementation

- Augmented existing implementations of (2,n) **ECDSA** [DKLs19, GG18]

Implementation

- Augmented existing implementations of (2,n) **ECDSA** [DKLs19, GG18]

Thanks Jack Doerner!

Implementation

- Augmented existing implementations of (2,n) **ECDSA** [DKLs19, GG18]

Thanks Jack Doerner!

- ECDSA on secp256k1 (Bitcoin's curve) including **novel OT Multiplier state refresh**

Implementation

- Augmented existing implementations of (2,n) **ECDSA** [DKLs19, GG18]

Thanks Jack Doerner!

- ECDSA on secp256k1 (Bitcoin's curve) including **novel OT Multiplier state refresh**

- Experiments on Amazon's AWS EC2 using t3.small

Implementation

- Augmented existing implementations of (2,n) **ECDSA** [DKLs19, GG18]

Thanks Jack Doerner!

- ECDSA on secp256k1 (Bitcoin's curve) including **novel OT Multiplier state refresh**
- Experiments on Amazon's AWS EC2 using t3.small
- **Computation overhead: <25%**

Implementation

- Augmented existing implementations of (2,n) **ECDSA** [DKLs19, GG18]

Thanks Jack Doerner!

- ECDSA on secp256k1 (Bitcoin's curve) including **novel OT Multiplier state refresh**
- Experiments on Amazon's AWS EC2 using t3.small
- **Computation overhead: <25%**
- **Communication: 200 bytes, no extra rounds**

This Work

- Correct definition is subtle
  - Guaranteed progress is impossible
  - We formulate **unanimous erasure**

- (*2*,*n*) setting: Efficient new protocol **native** to wallets
  - New interleaved threshold sig technique
  - Offline parties can miss arbitrary number of epochs
  - Implementation shows practicality

- (*t*,*n*) setting: **Impossible**!

This Work

- Correct definition is subtle
  - Guaranteed progress is impossible
  - We formulate **unanimous erasure**

- (*2*,*n*) setting: Efficient new protocol **native** to wallets
  - New interleaved threshold sig technique
  - Offline parties can miss arbitrary number of epochs
  - Implementation shows practicality

- (*t*,*n*) setting: **Impossible**!

Background

Background

- Usually VSS/DKG in a given setting translates to equivalent proactive secret sharing

Background

- Usually VSS/DKG in a given setting translates to equivalent proactive secret sharing
- VSS/DKG where only t parties speak (with t-1 corrupt) is known to be feasible [GMW91]

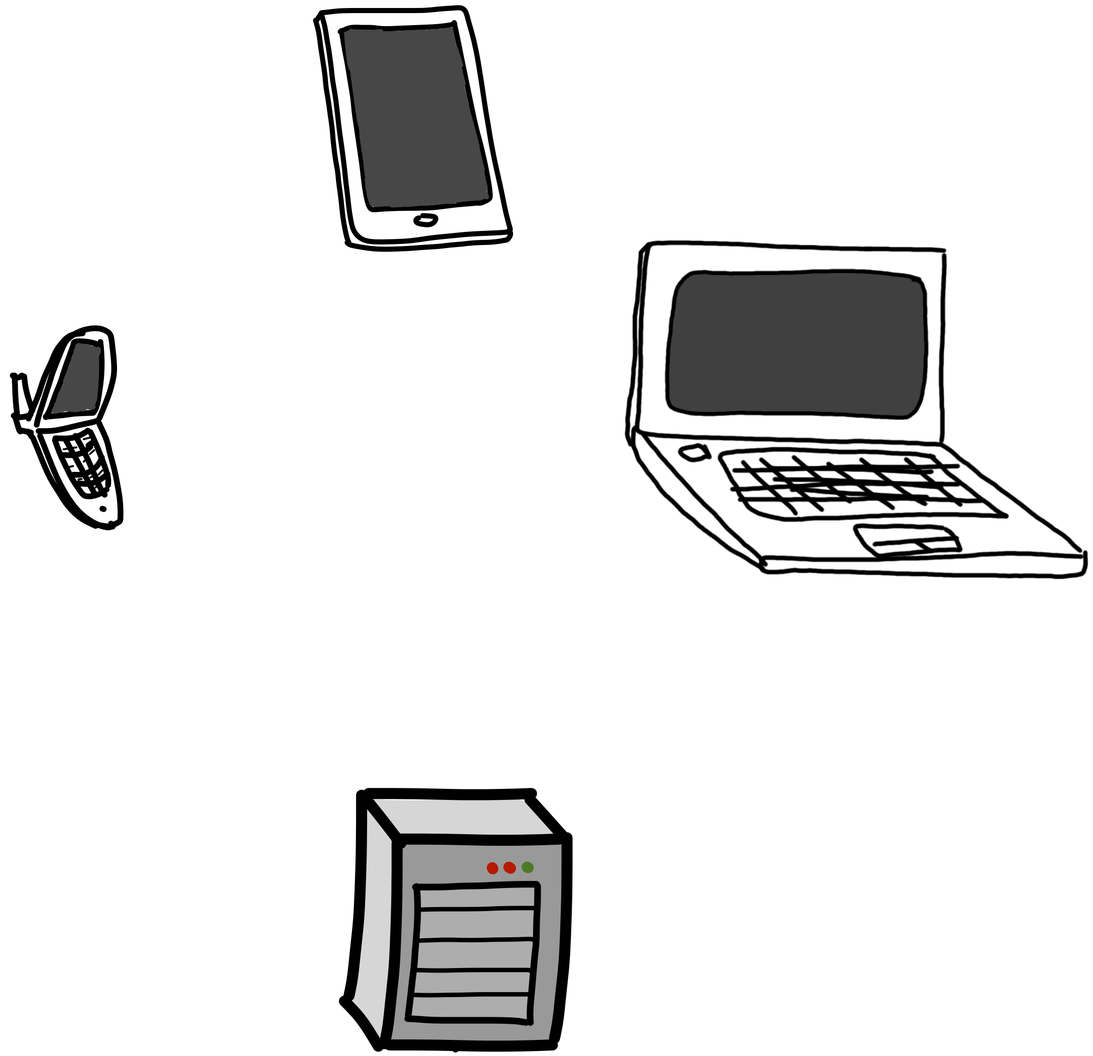
Background

- Usually VSS/DKG in a given setting translates to equivalent proactive secret sharing
- VSS/DKG where only t parties speak (with t-1 corrupt) is known to be feasible [GMW91]
- Indicates intuition that (t,n) proactivization with offline refresh should be solvable with heavy tools

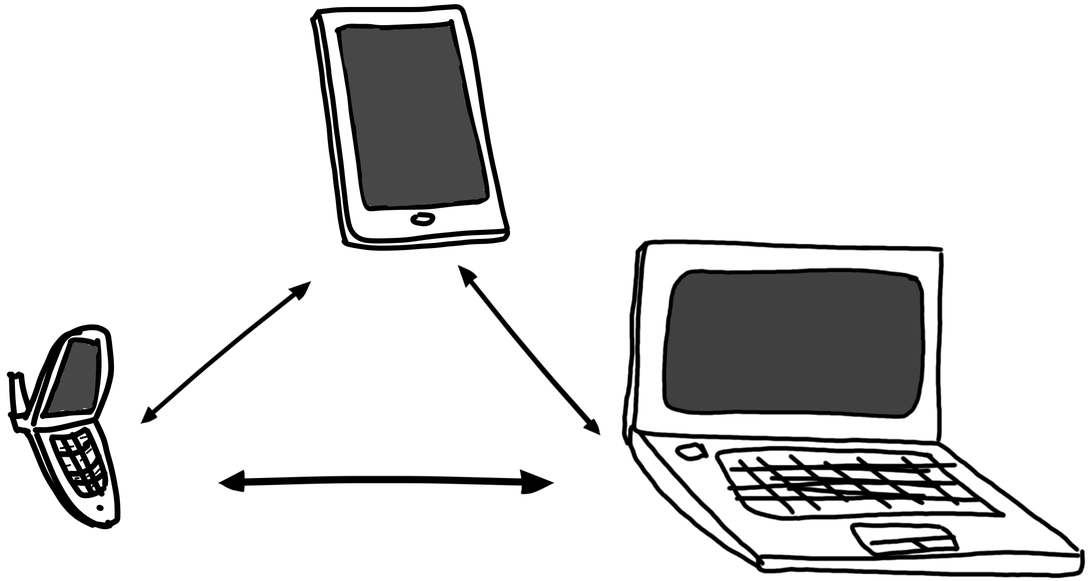
Background

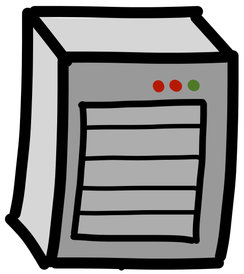
- Usually VSS/DKG in a given setting translates to equivalent proactive secret sharing
- VSS/DKG where only t parties speak (with t-1 corrupt) is known to be feasible [GMW91]
- Indicates intuition that (t,n) proactivization with offline refresh should be solvable with heavy tools
- Intuition turns out to be wrong!

Intuition: (3,4) case

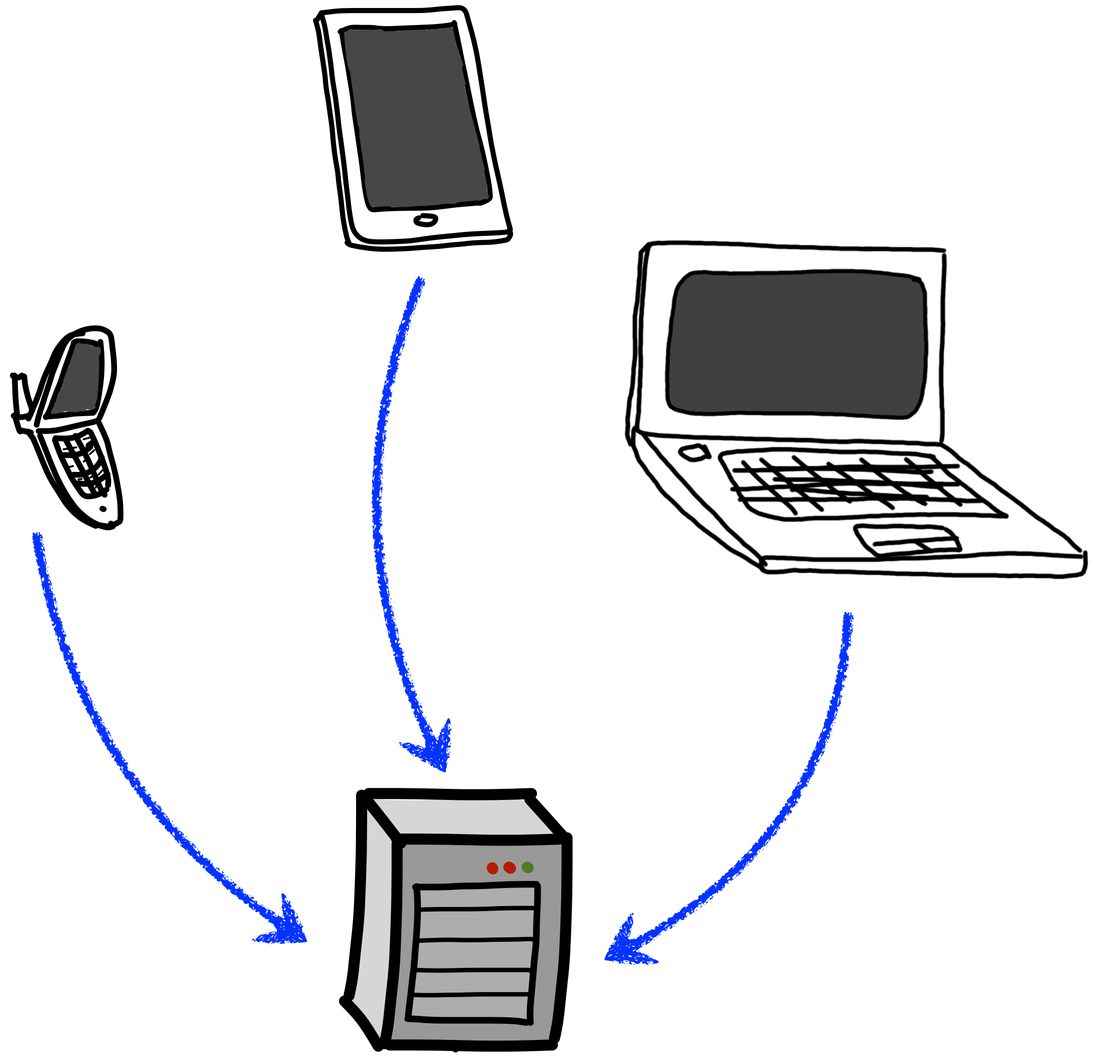


Intuition: (3,4) case

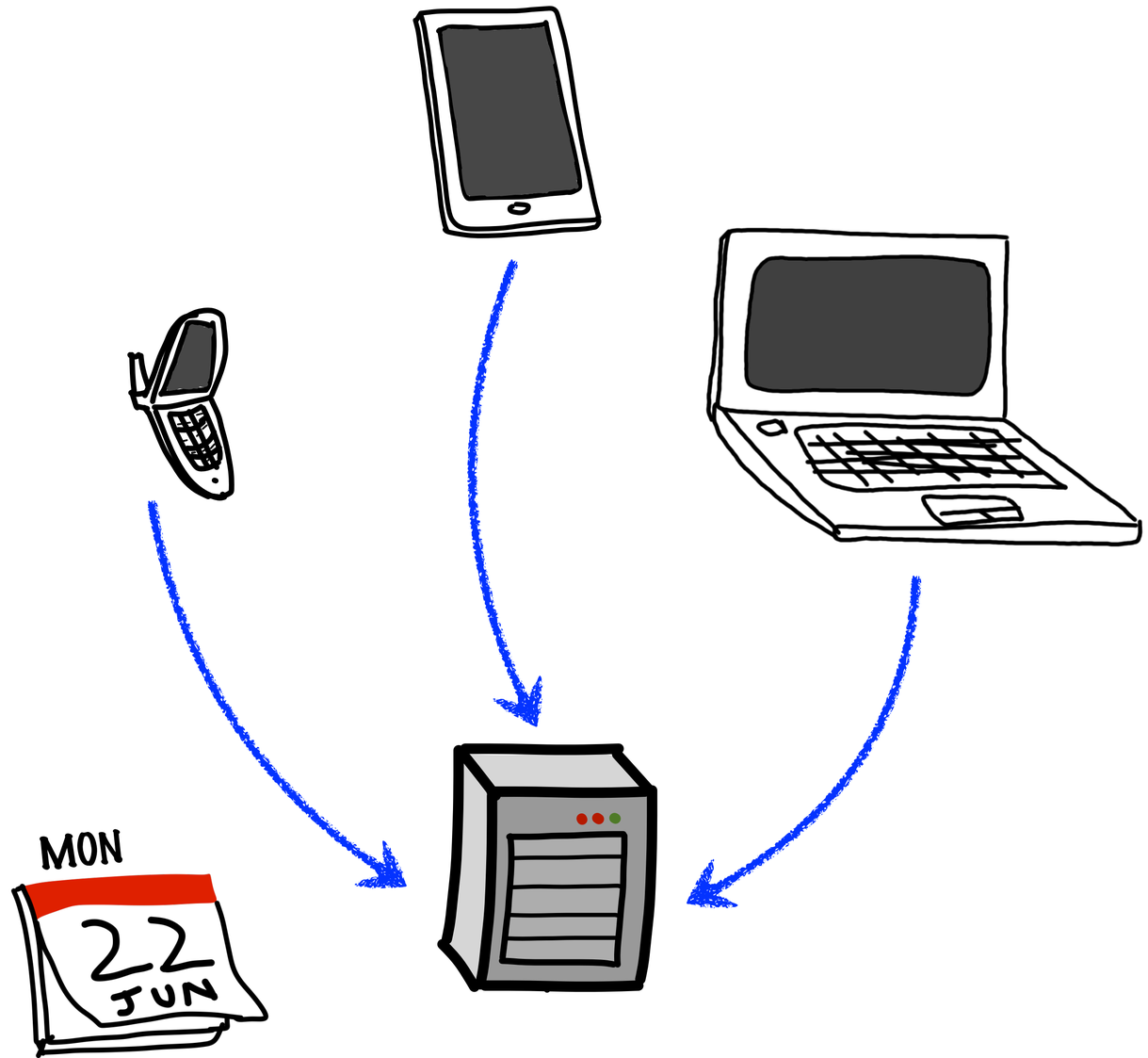


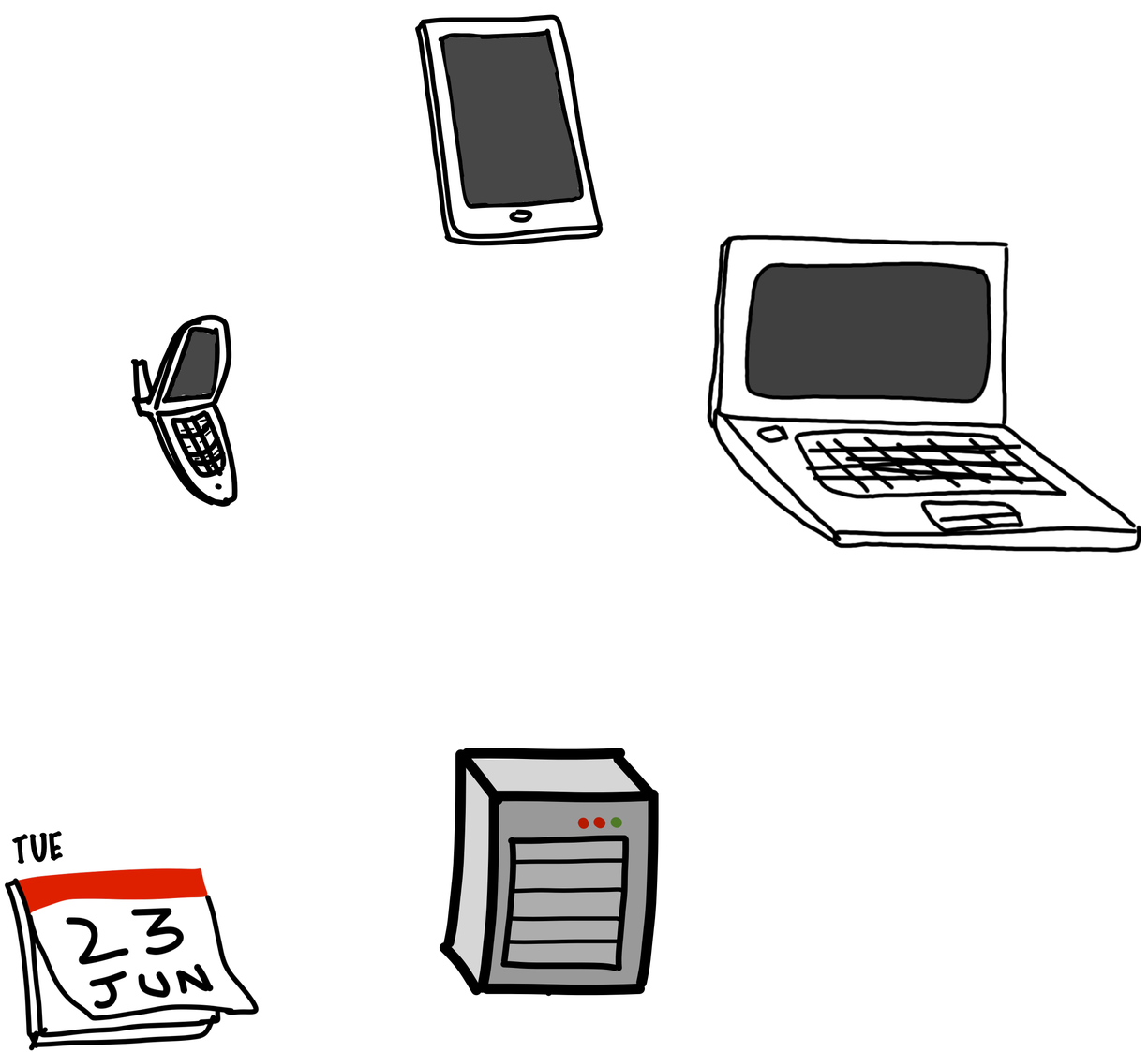


Intuition: (3,4) case

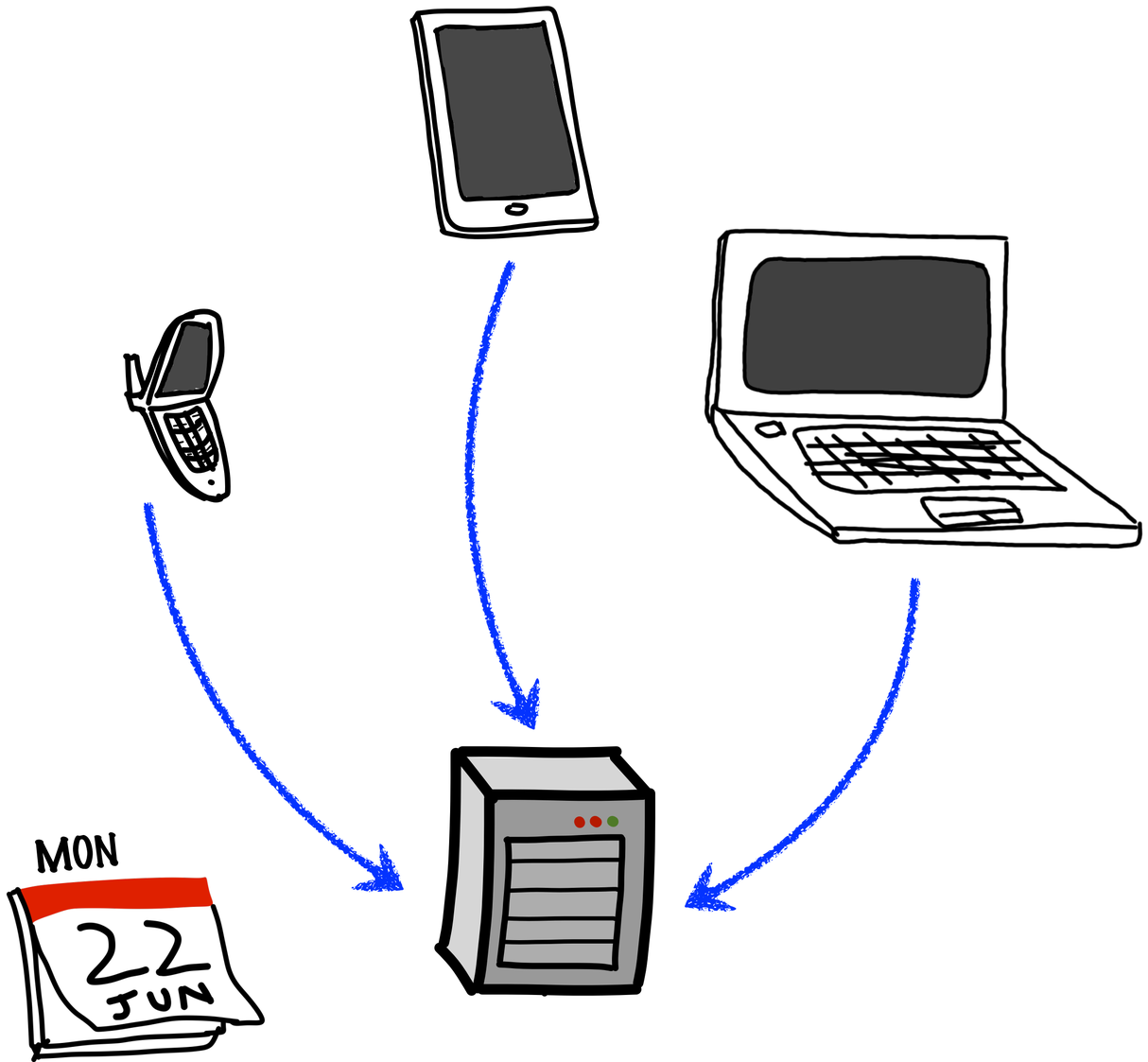


Intuition: (3,4) case

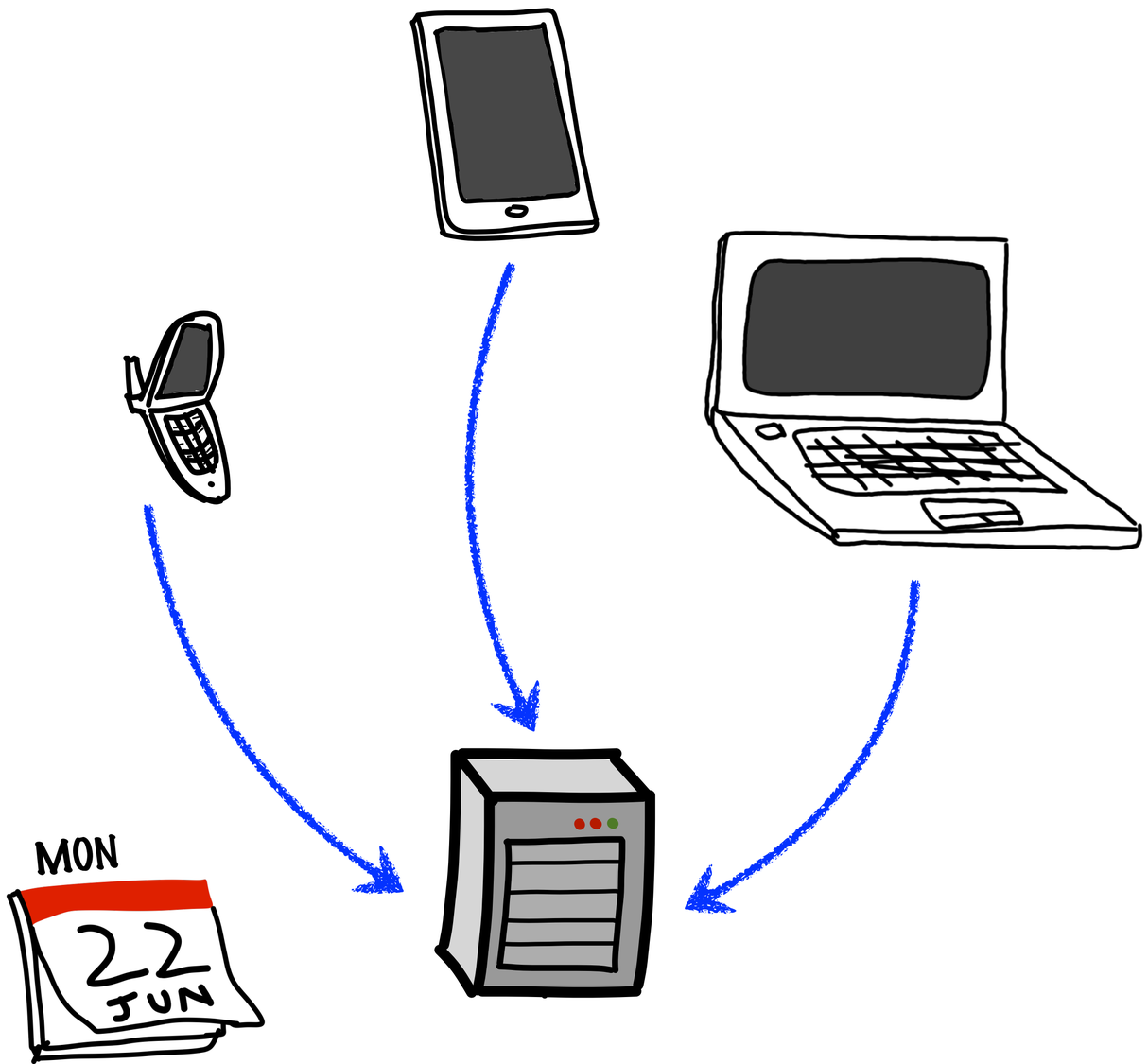




Intuition: (3,4) case

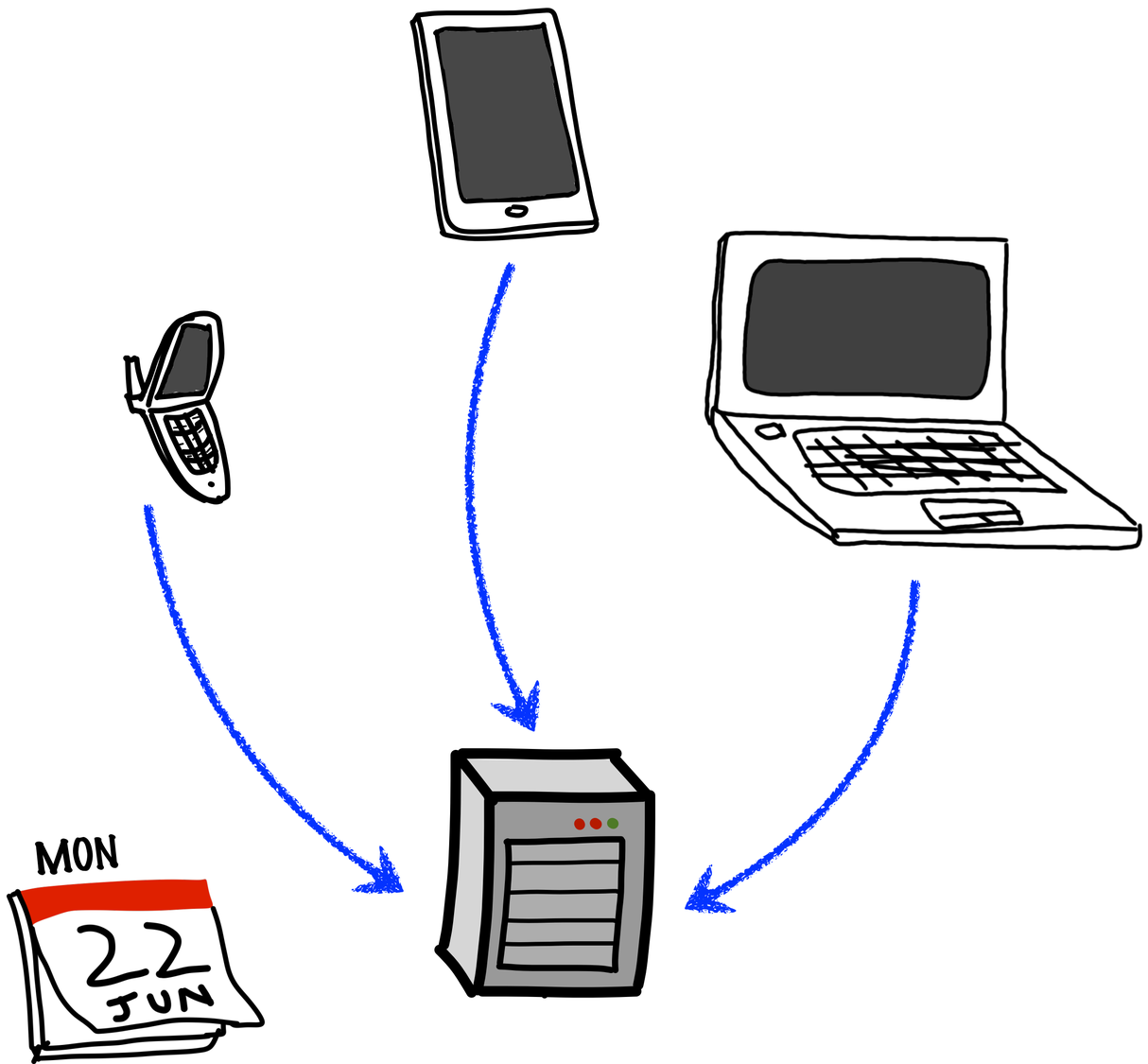


Intuition: (3,4) case



Assumption:
secure against corruption
of two parties

Intuition: (3,4) case

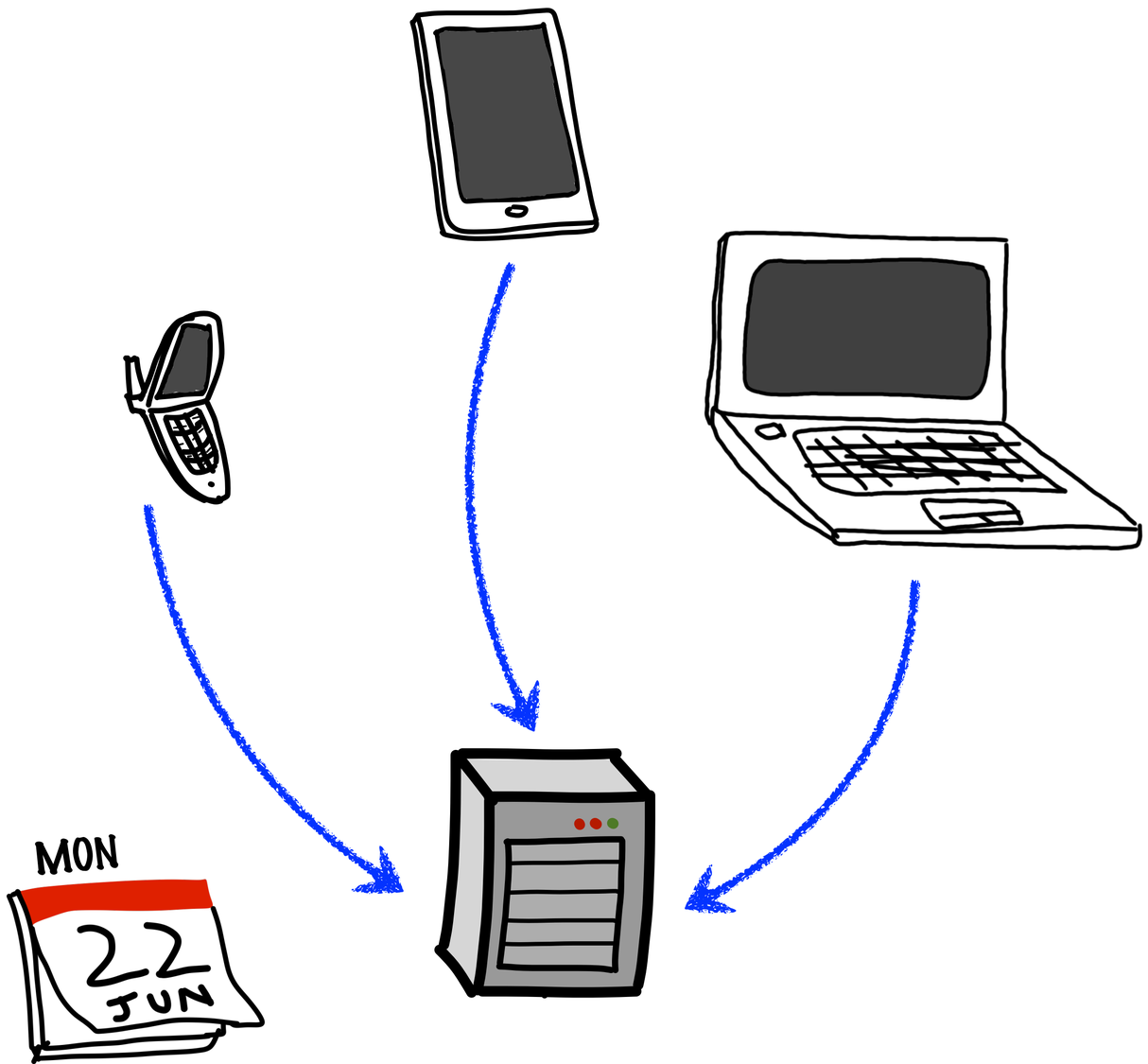


Assumption:
secure against corruption of two parties

Claim:

View of  has enough info for  to refresh

Intuition: (3,4) case

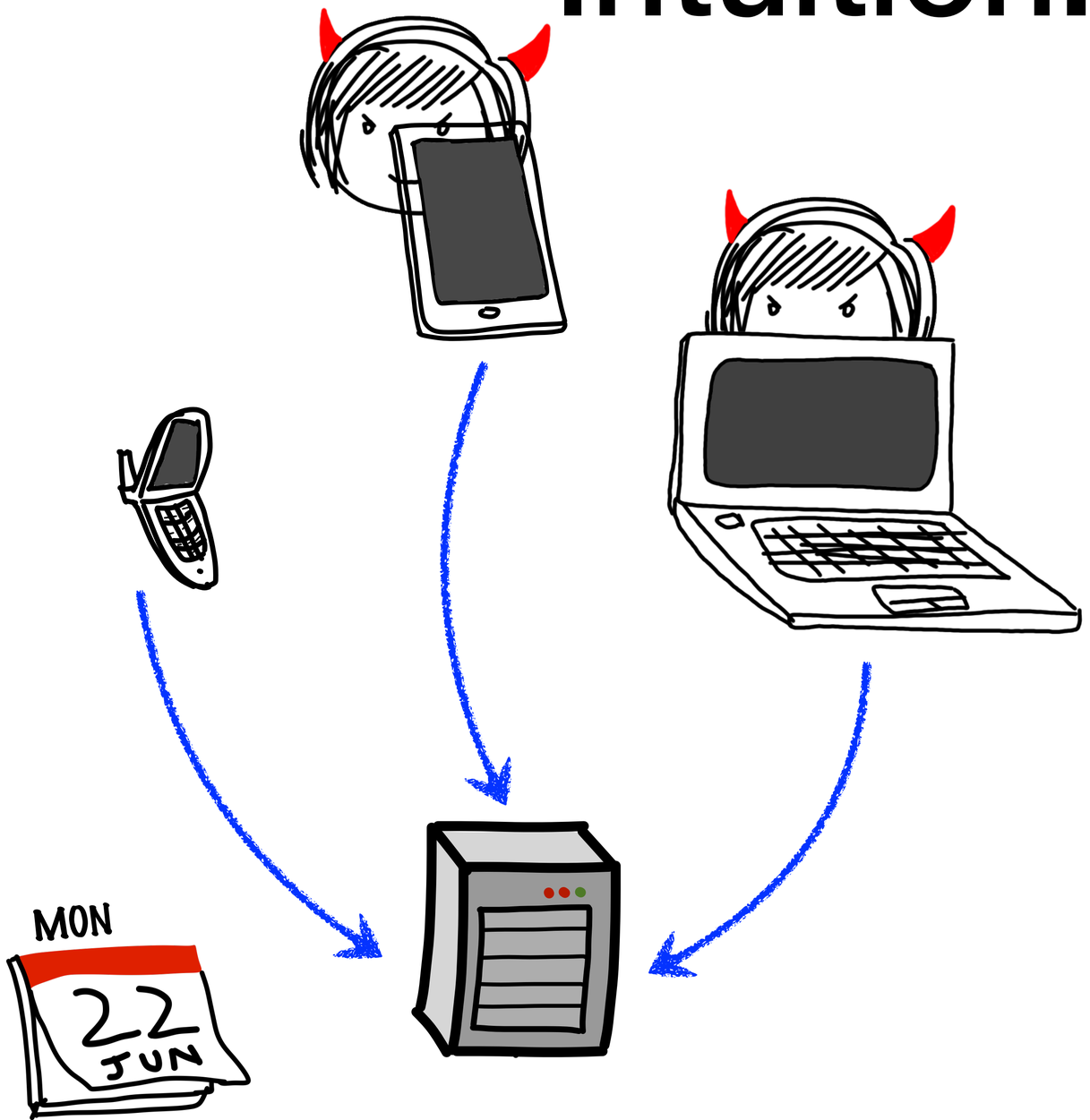


Assumption:
secure against corruption of two parties

Claim:

View of  has enough info for  to refresh

Intuition: (3,4) case

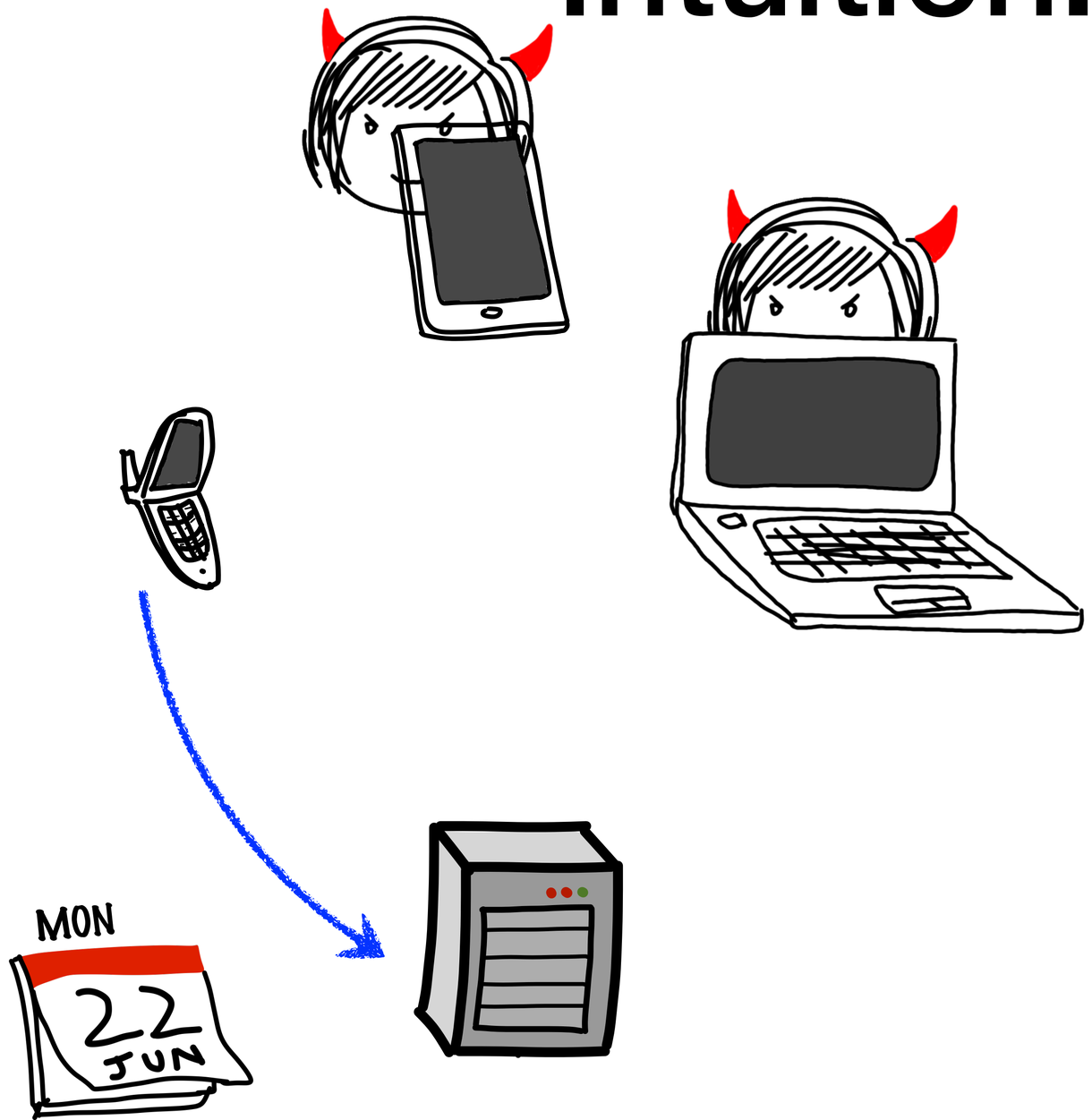


Assumption:
secure against corruption of two parties

Claim:

View of  has enough info for  to refresh

Intuition: (3,4) case

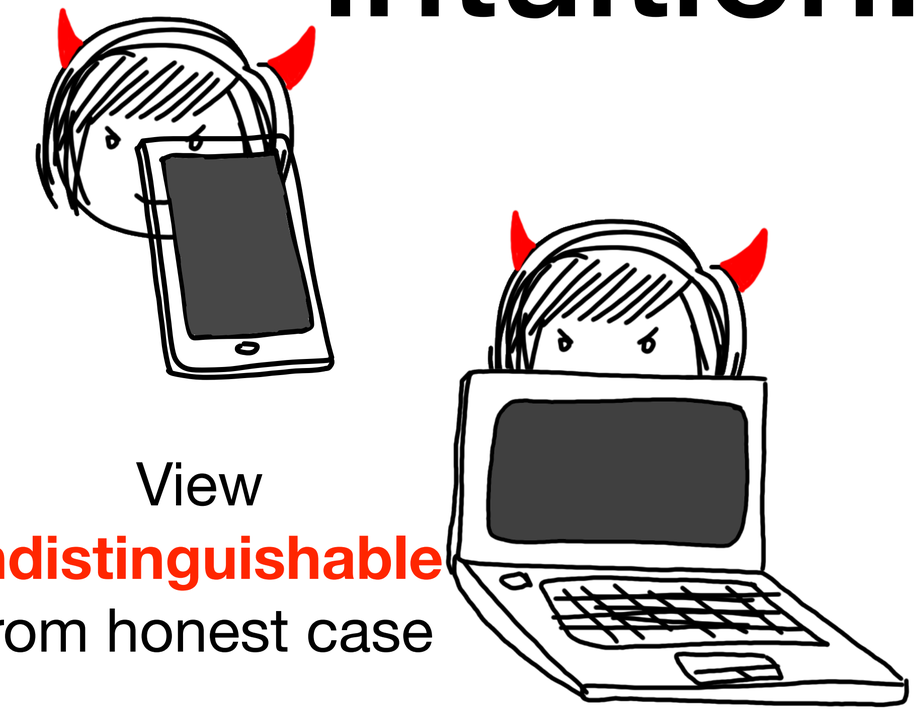


Assumption:
secure against corruption of two parties

Claim:

View of  has enough info for  to refresh

Intuition: (3,4) case



Assumption:
secure against corruption of two parties

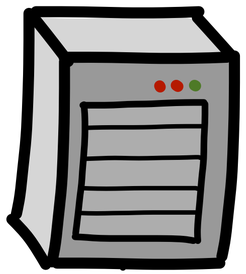
Claim:

View of  has enough info for  to refresh

Intuition: (3,4) case

View
indistinguishable
from honest case

SUCCESS



Assumption:
secure against corruption of two parties

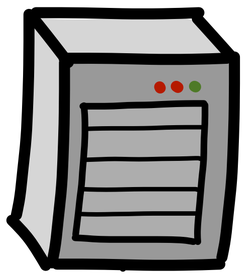
Claim:

View of  has enough info for  to refresh

Intuition: (3,4) case

View **indistinguishable** from honest case

SUCCESS



SUCCESS

By unanimous erasure

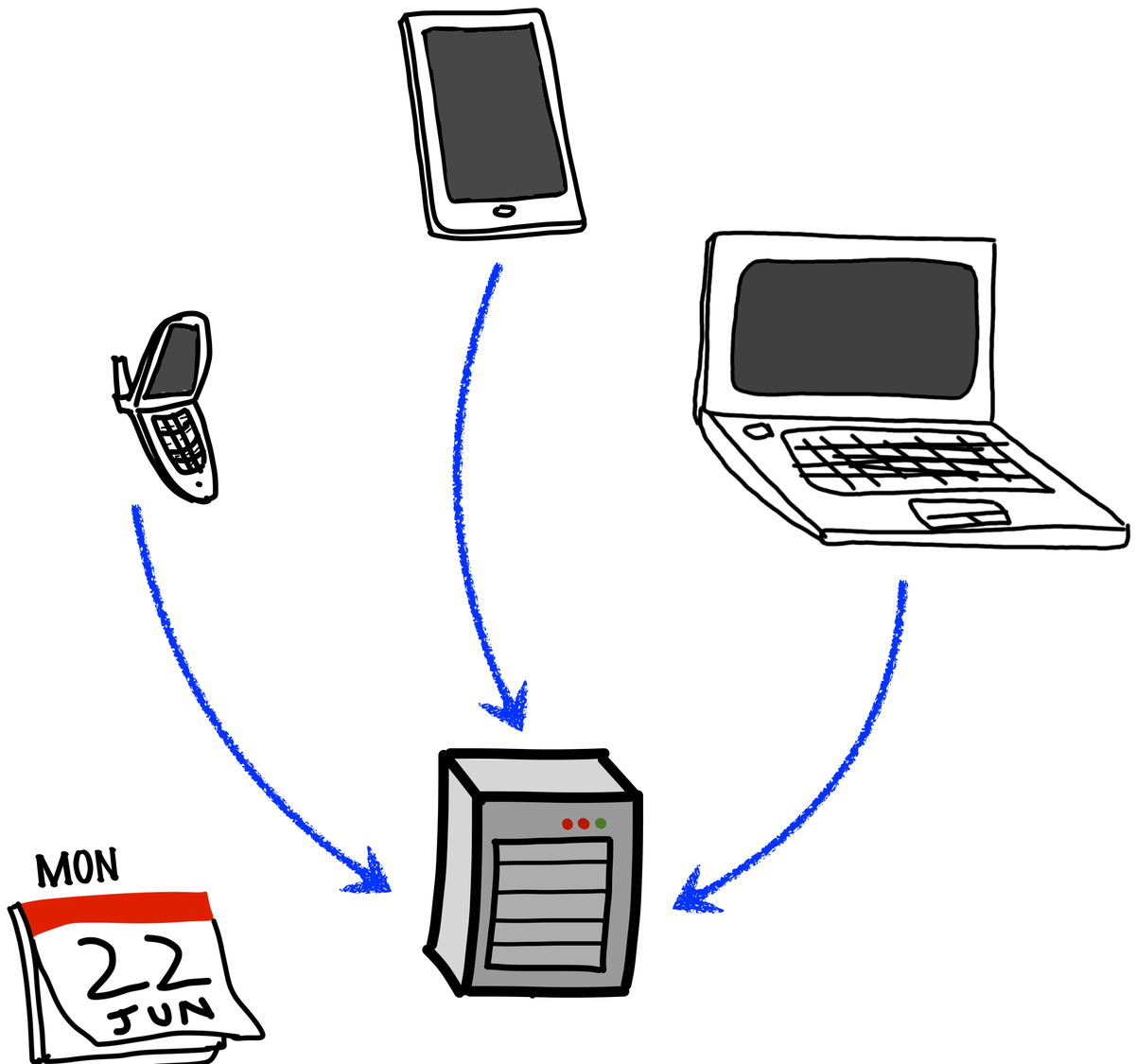
Assumption:

secure against corruption of two parties

Claim:

View of  has enough info for  to refresh

Intuition: (3,4) case

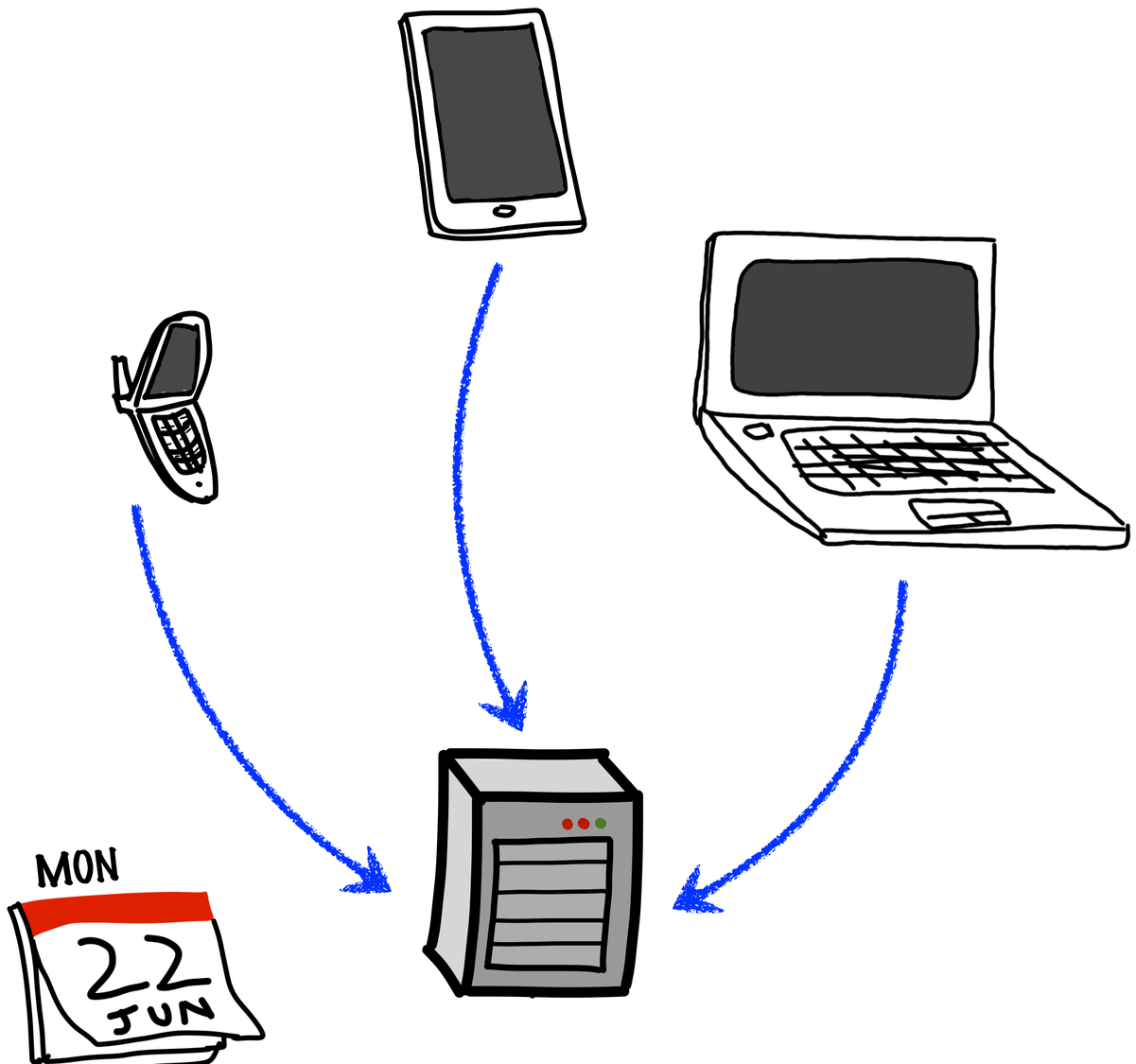


Assumption:
secure against corruption of two parties

Claim:

View of  has enough info for  to refresh

Intuition: (3,4) case

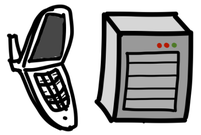


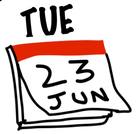
Assumption:
secure against corruption of two parties

Claim:

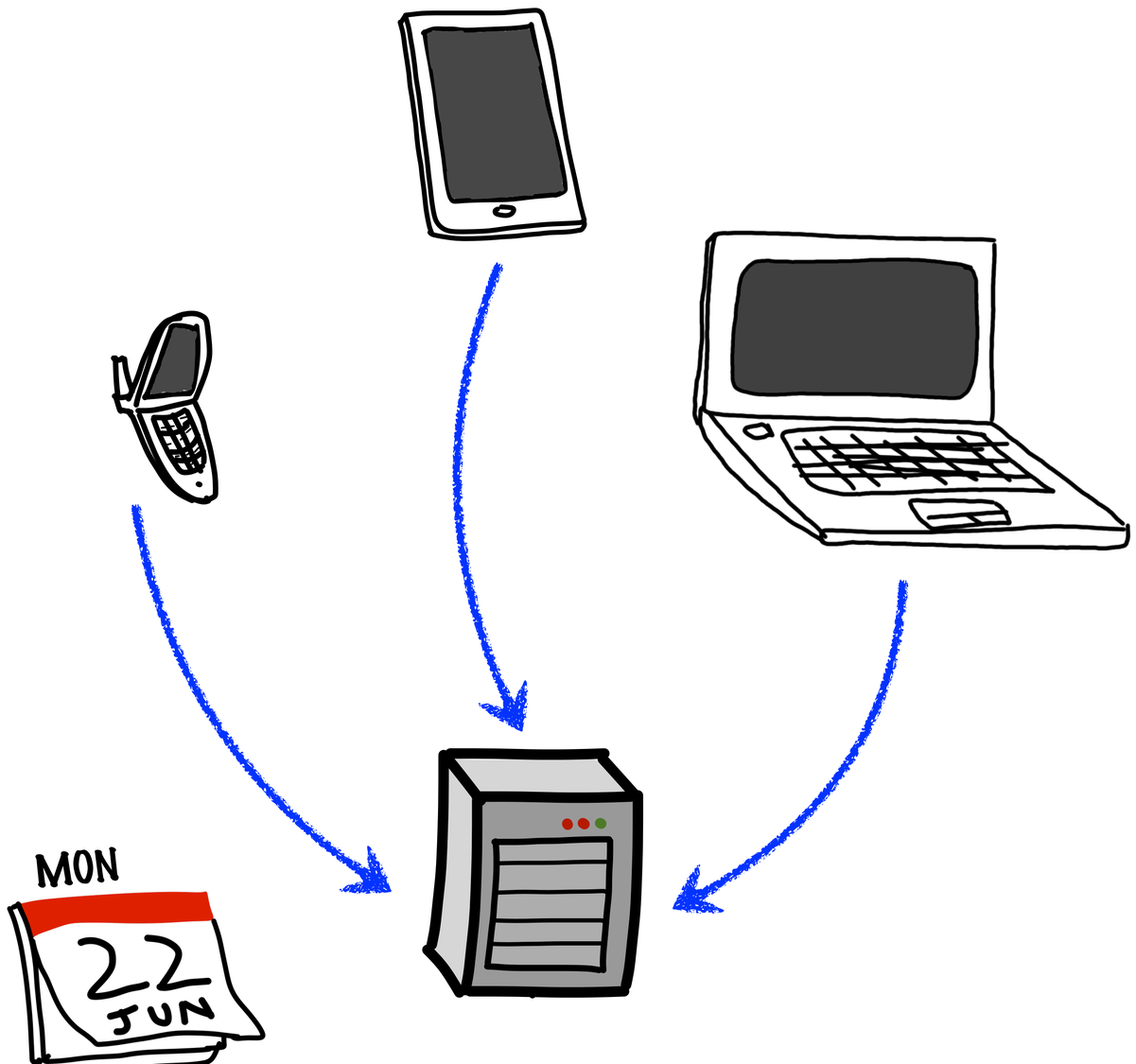
View of  has enough info for  to refresh

Corollary:

Corrupting  on 

Gives secrets of  on 

Intuition: (3,4) case



Assumption:
secure against corruption
of two parties

Claim:

View of  has enough info
for  to refresh

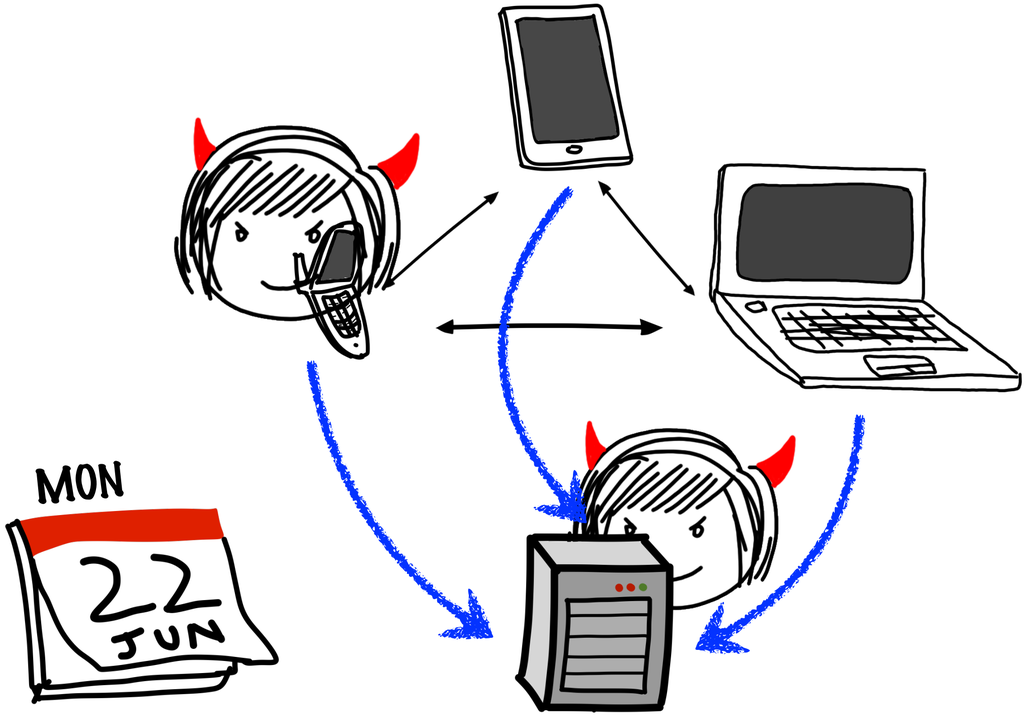
Corollary:

Even after uncorruption

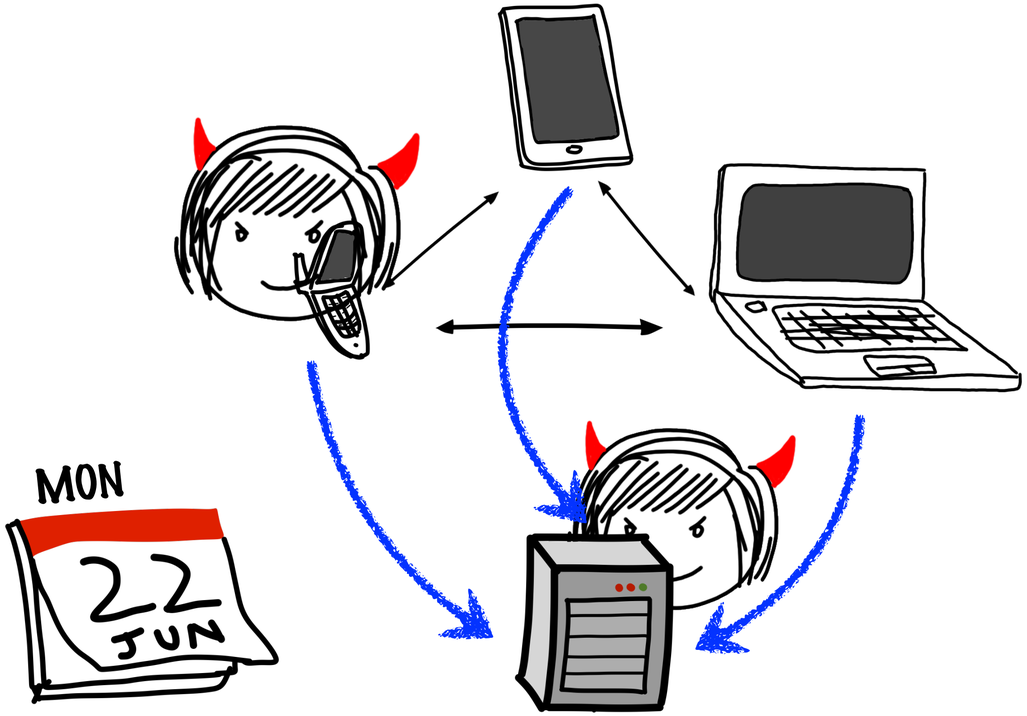
Corrupting  on 

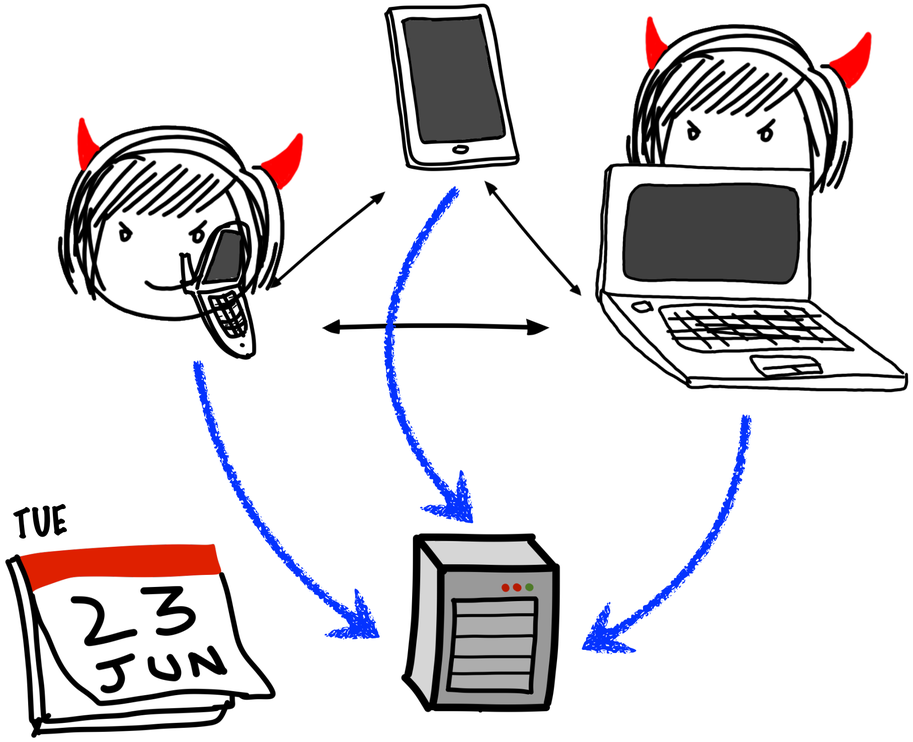
Gives secrets of  on 

General Attack Strategy

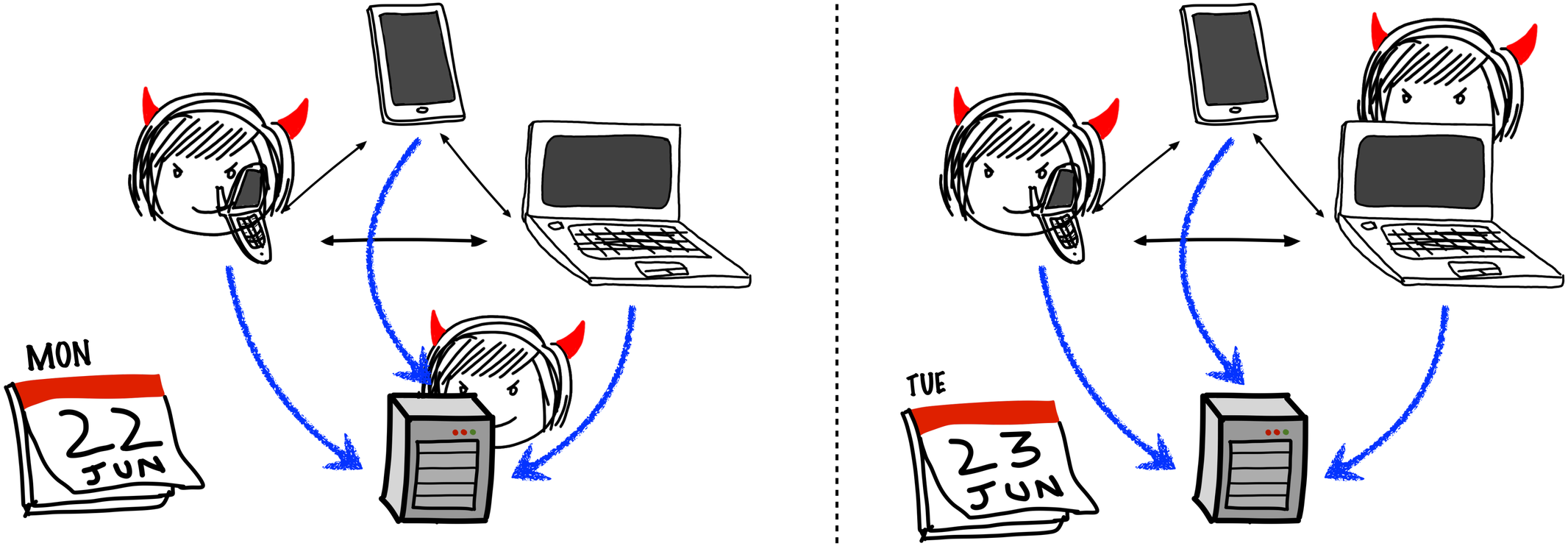


General Attack Strategy



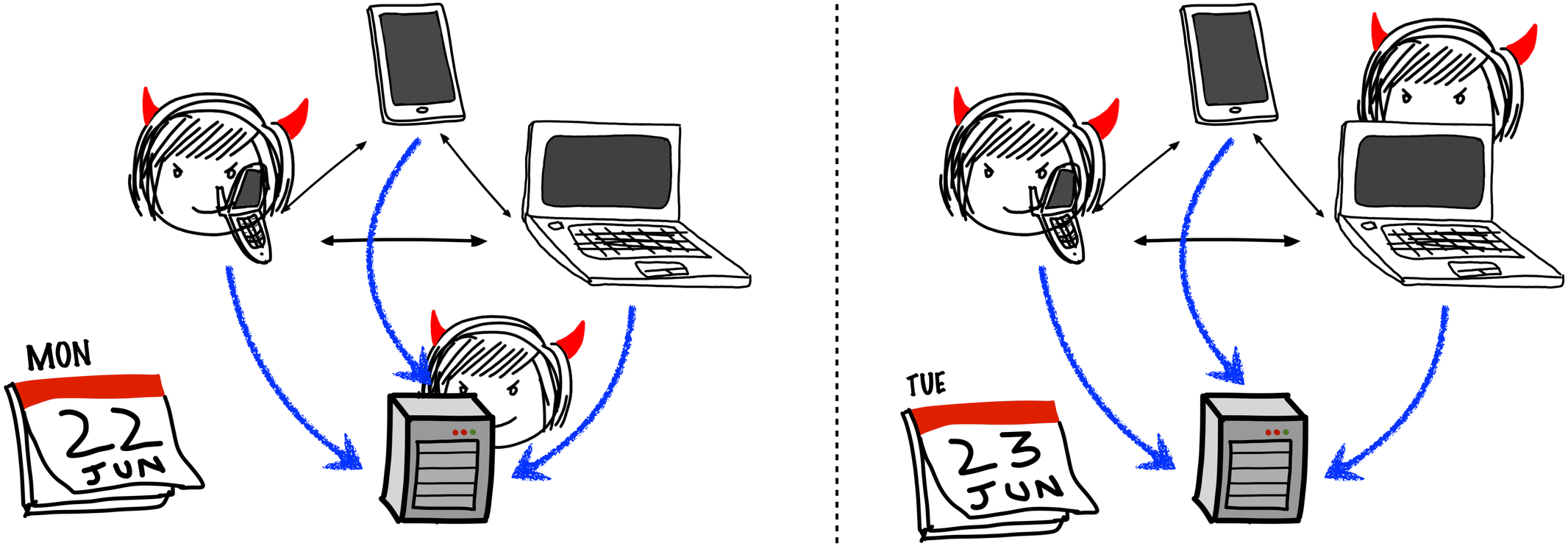


General Attack Strategy



Can derive state of  on TUE even ***after*** refresh

General Attack Strategy



Can derive state of  on TUE even ***after*** refresh

Two corrupt parties + 1 derived state = (3,4) broken

See the paper for…

See the paper for…

- Refreshing Multiplier/OT Extension state for ECDSA signing (hint: Beaver’s OT correlation trick)

See the paper for…

- Refreshing Multiplier/OT Extension state for ECDSA signing (hint: Beaver's OT correlation trick)
- Benchmarks of overhead added by (2,*n*) refresh to existing ECDSA implementation

Thanks Jack Doerner!

See the paper for…

- Refreshing Multiplier/OT Extension state for ECDSA signing (hint: Beaver's OT correlation trick)
- Benchmarks of overhead added by (2,*n*) refresh to existing ECDSA implementation

Thanks Jack Doerner!

- Discussions of definition, full proofs

Thanks!

eprint.iacr.org/2019/1328

Thanks **Eysa Lee** for