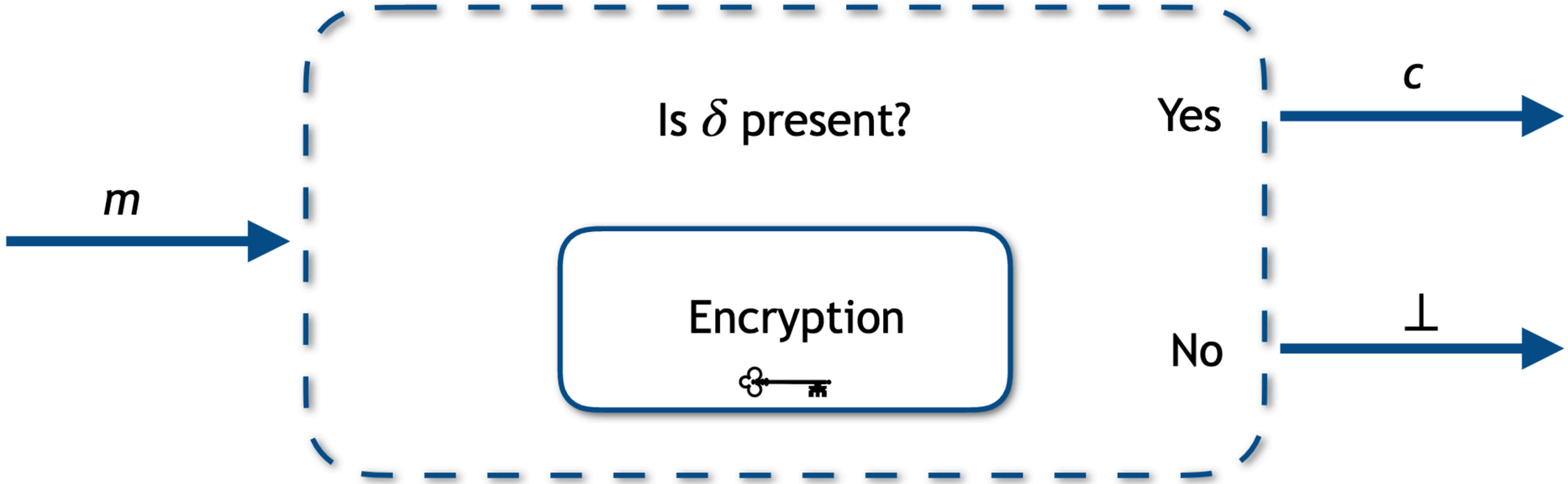
White-box cryptography with global device-binding from message-recoverable signatures and token-based obfuscation

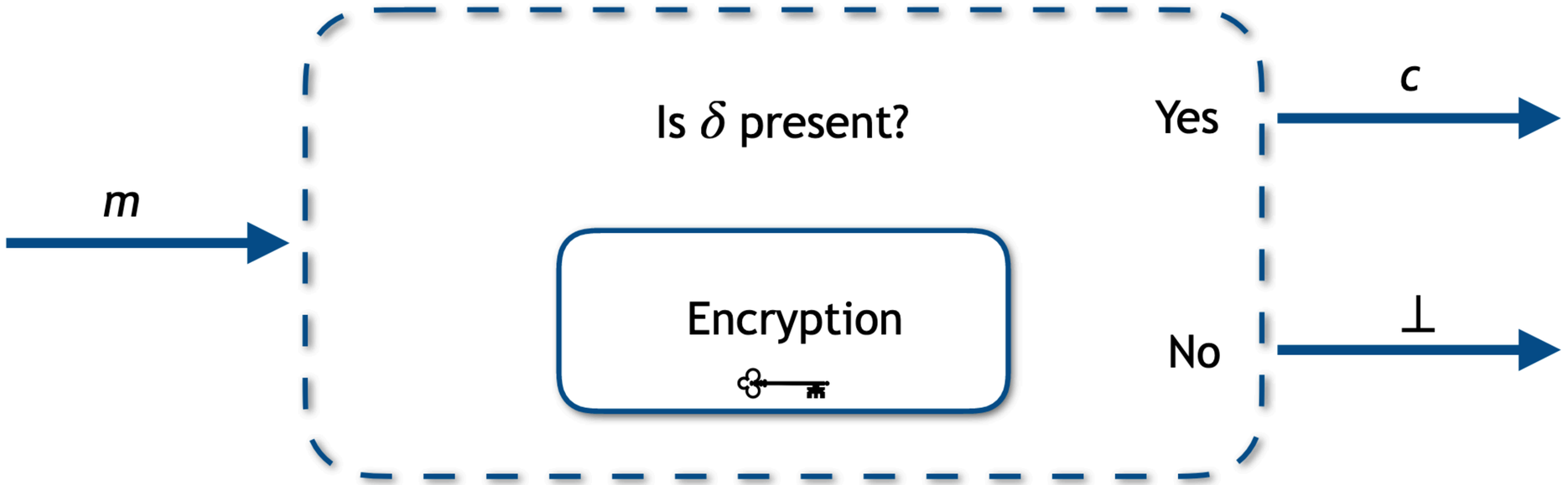
Estuardo Alpirez Bock
Xiphera LTD

Joint work with Shashank Agrawal, Yilei Chen and Gaven Watson while at Visa Research

White-box crypto with device-binding

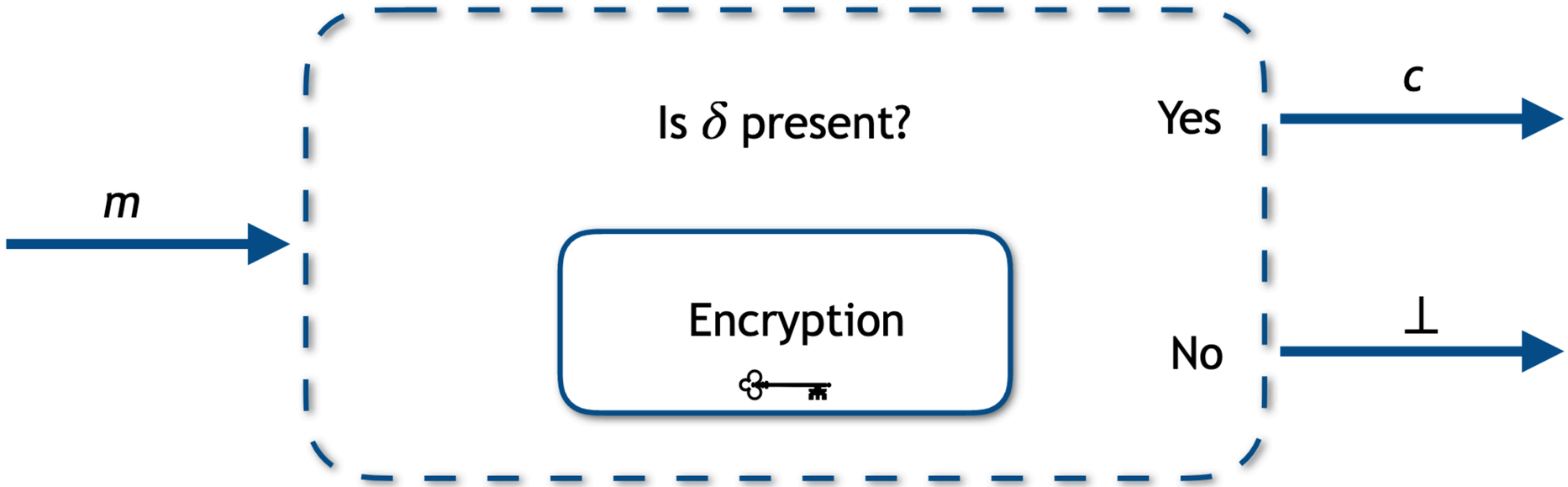


White-box crypto with device-binding



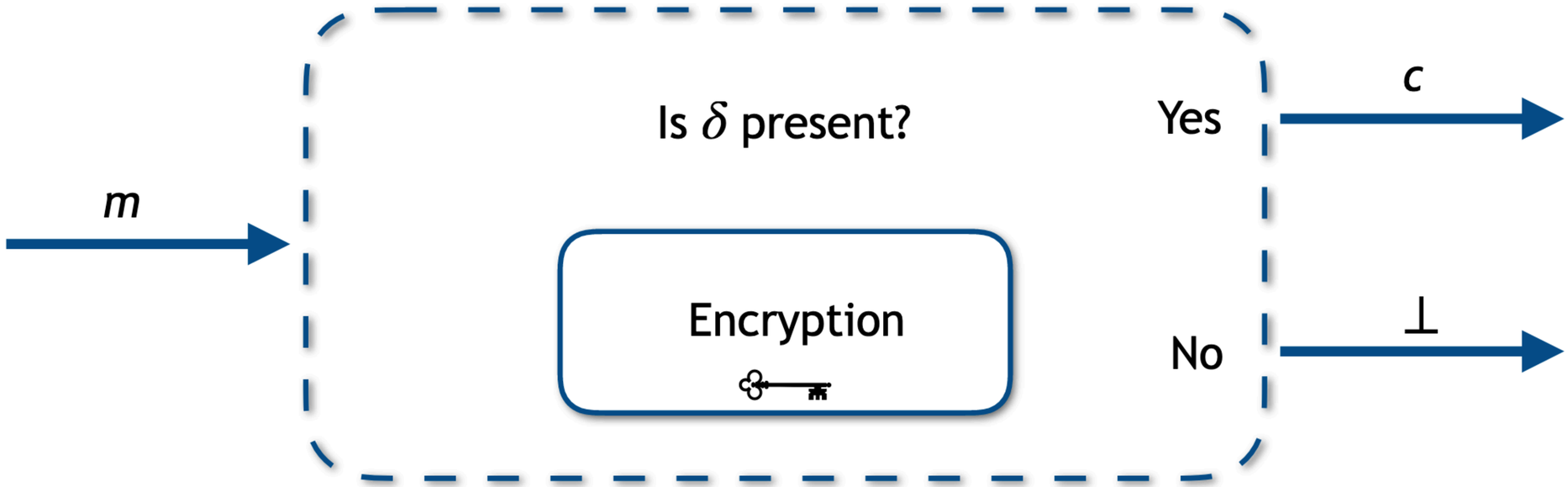
- The white-box program should only run correctly on one specific device (identified by δ)

White-box crypto with device-binding



- The white-box program should only run correctly on one specific device (identified by δ)
- Method for mitigating *code-lifting* attacks

White-box crypto with device-binding



- Definitional studies and feasibility results for white-box crypto
  - Construct white-box programs from established cryptographic primitives —> prove their security in the presence of a white-box adversary

Constructions achieving the property of device-binding

Our results

Constructions achieving the property of device-binding

Our results

- Introduce the concept of *global white-boxes*

Constructions achieving the property of device-binding

Our results

- Introduce the concept of *global white-boxes*
    - White-box programs compiled once, uploaded to the cloud. Can be downloaded and securely bound to a number of devices.

Constructions achieving the property of device-binding

Our results

- Introduce the concept of *global white-boxes*
    - White-box programs compiled once, uploaded to the cloud. Can be downloaded and securely bound to a number of devices.
    - Motivated by the use case of white-box crypto for protecting mobile payment applications

Constructions achieving the property of device-binding

Our results

- Introduce the concept of *global white-boxes*
    - White-box programs compiled once, uploaded to the cloud. Can be downloaded and securely bound to a number of devices.
    - Motivated by the use case of white-box crypto for protecting mobile payment applications
- Introduce security notions capturing security of such global white-boxes

Constructions achieving the property of device-binding

Our results

- Introduce the concept of *global white-boxes*
    - White-box programs compiled once, uploaded to the cloud. Can be downloaded and securely bound to a number of devices.
    - Motivated by the use case of white-box crypto for protecting mobile payment applications
- Introduce security notions capturing security of such global white-boxes
    - Different flavours of security for different types of GWBs

Constructions achieving the property of device-binding

Our results

- Introduce the concept of *global white-boxes*
    - White-box programs compiled once, uploaded to the cloud. Can be downloaded and securely bound to a number of devices.
    - Motivated by the use case of white-box crypto for protecting mobile payment applications
- Introduce security notions capturing security of such global white-boxes
    - Different flavours of security for different types of GWBs
- Present feasibility results via token-based obfuscation and message-recoverable signatures

Constructions achieving the property of device-binding

Our results

- Introduce the concept of *global white-boxes*
    - White-box programs compiled once, uploaded to the cloud. Can be downloaded and securely bound to a number of devices.
    - Motivated by the use case of white-box crypto for protecting mobile payment applications
- Introduce security notions capturing security of such global white-boxes
    - Different flavours of security for different types of GWBs
- Present feasibility results via token-based obfuscation and message-recoverable signatures
    - Introduce puncturable message-recoverable signatures

Constructions achieving the property of device-binding

Our results

- Introduce the concept of *global white-boxes*
    - White-box programs compiled once, uploaded to the cloud. Can be downloaded and securely bound to a number of devices.
    - Motivated by the use case of white-box crypto for protecting mobile payment applications
- Introduce security notions capturing security of such global white-boxes
    - Different flavours of security for different types of GWBs
- Present feasibility results via token-based obfuscation and message-recoverable signatures
    - Introduce puncturable message-recoverable signatures
    - Show new ways of constructing white-box crypto via token-based obfuscation (which can be constructed from LWE)

Previous work on Device-binding

- The device binding process is *personalised*

[1] Alpirez Bock, Amadori, Brzuska and Michiels: On the security goals of white-box cryptography. CHES 2020

[2] Alpirez Bock, Brzuska, Fischlin, Janson and Michiels: Security reductions for white-box key storage. ASIACRYPT 2020

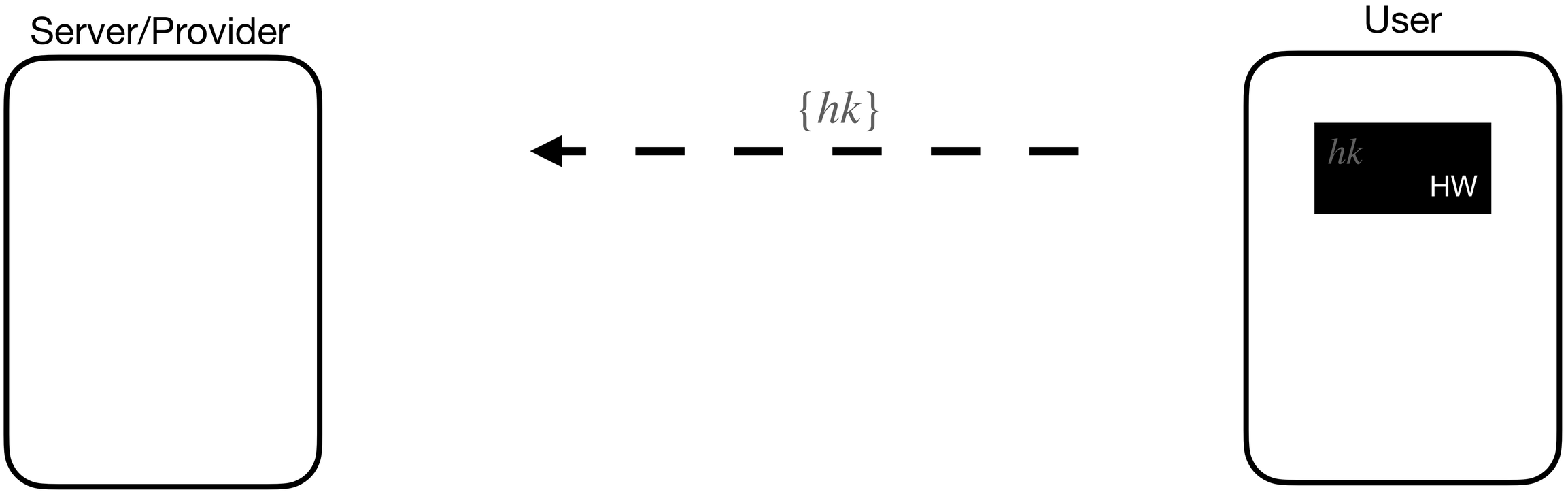
Previous work on Device-binding

- The device binding process is *personalised*

[1] Alpirez Bock, Amadori, Brzuska and Michiels: On the security goals of white-box cryptography. CHES 2020

[2] Alpirez Bock, Brzuska, Fischlin, Janson and Michiels: Security reductions for white-box key storage. ASIACRYPT 2020

Previous work on Device-binding

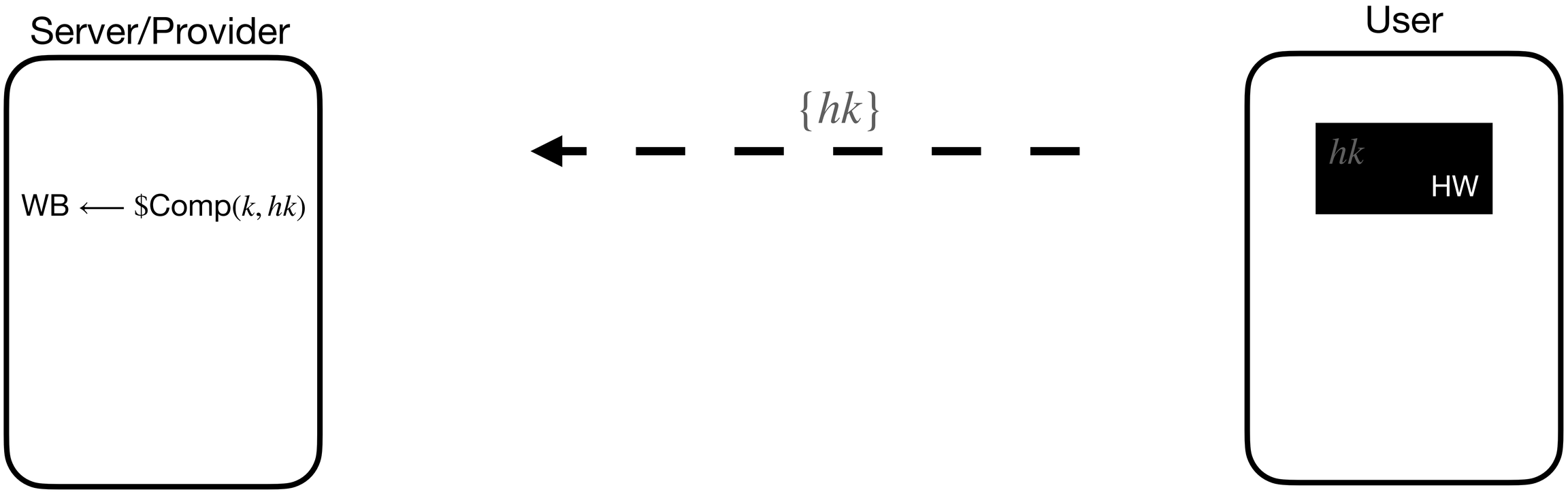


- The device binding process is *personalised*

[1] Alpirez Bock, Amadori, Brzuska and Michiels: On the security goals of white-box cryptography. CHES 2020

[2] Alpirez Bock, Brzuska, Fischlin, Janson and Michiels: Security reductions for white-box key storage. ASIACRYPT 2020

Previous work on Device-binding

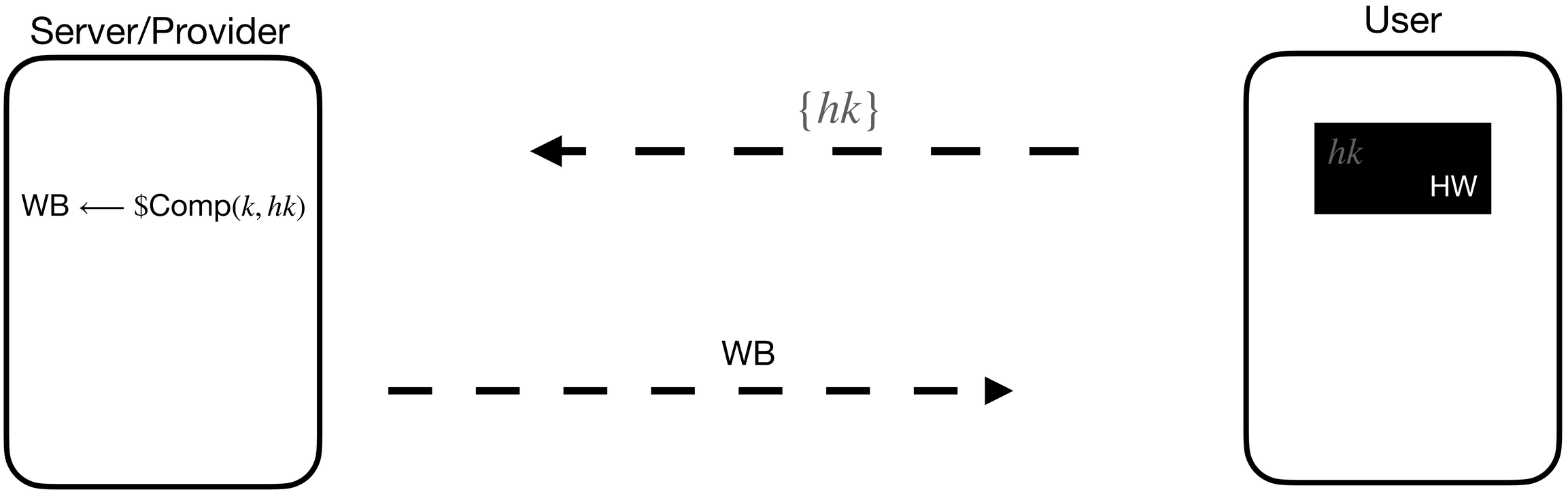


- The device binding process is *personalised*

[1] Alpirez Bock, Amadori, Brzuska and Michiels: On the security goals of white-box cryptography. CHES 2020

[2] Alpirez Bock, Brzuska, Fischlin, Janson and Michiels: Security reductions for white-box key storage. ASIACRYPT 2020

Previous work on Device-binding

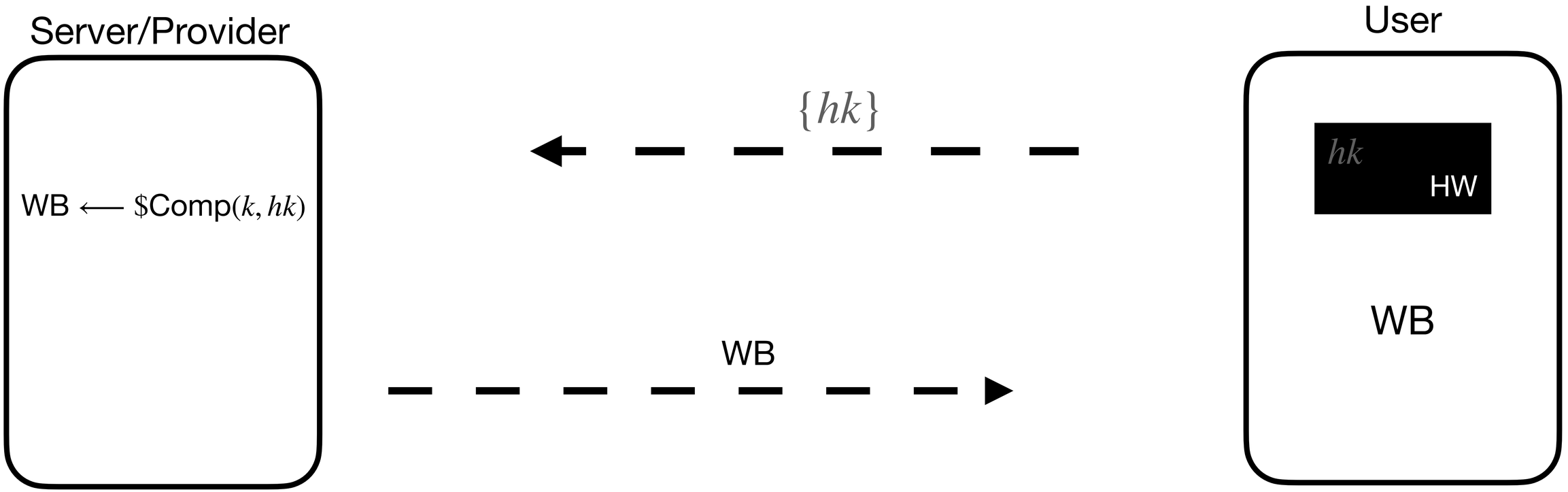


- The device binding process is *personalised*

[1] Alpirez Bock, Amadori, Brzuska and Michiels: On the security goals of white-box cryptography. CHES 2020

[2] Alpirez Bock, Brzuska, Fischlin, Janson and Michiels: Security reductions for white-box key storage. ASIACRYPT 2020

Previous work on Device-binding

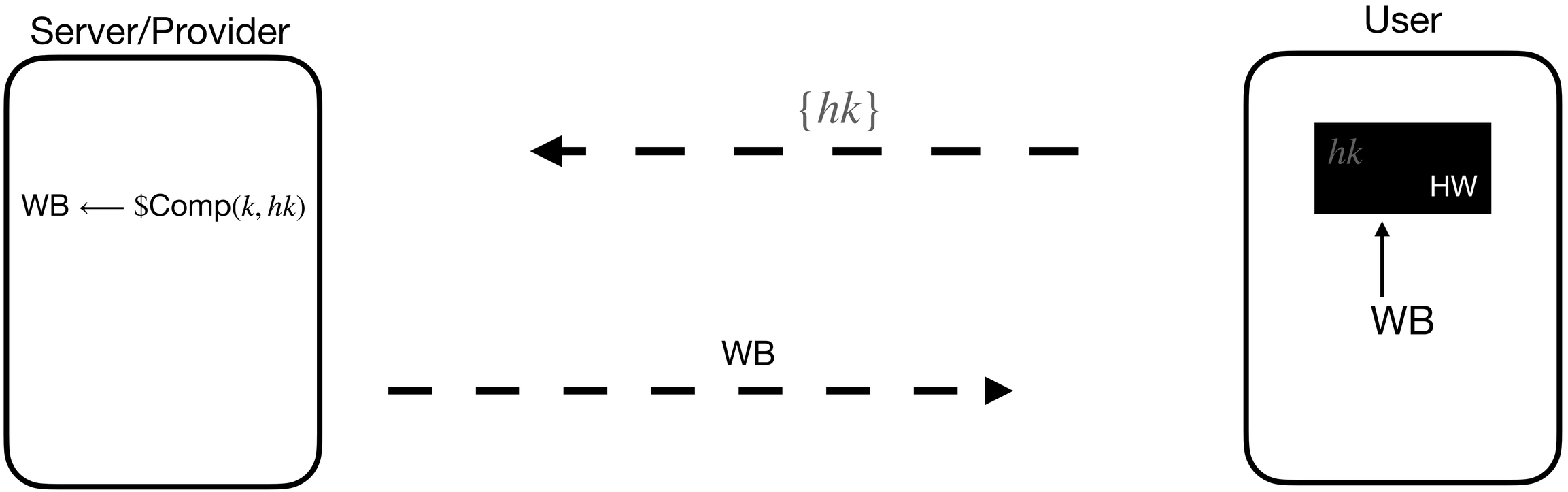


- The device binding process is *personalised*

[1] Alpirez Bock, Amadori, Brzuska and Michiels: On the security goals of white-box cryptography. CHES 2020

[2] Alpirez Bock, Brzuska, Fischlin, Janson and Michiels: Security reductions for white-box key storage. ASIACRYPT 2020

Previous work on Device-binding

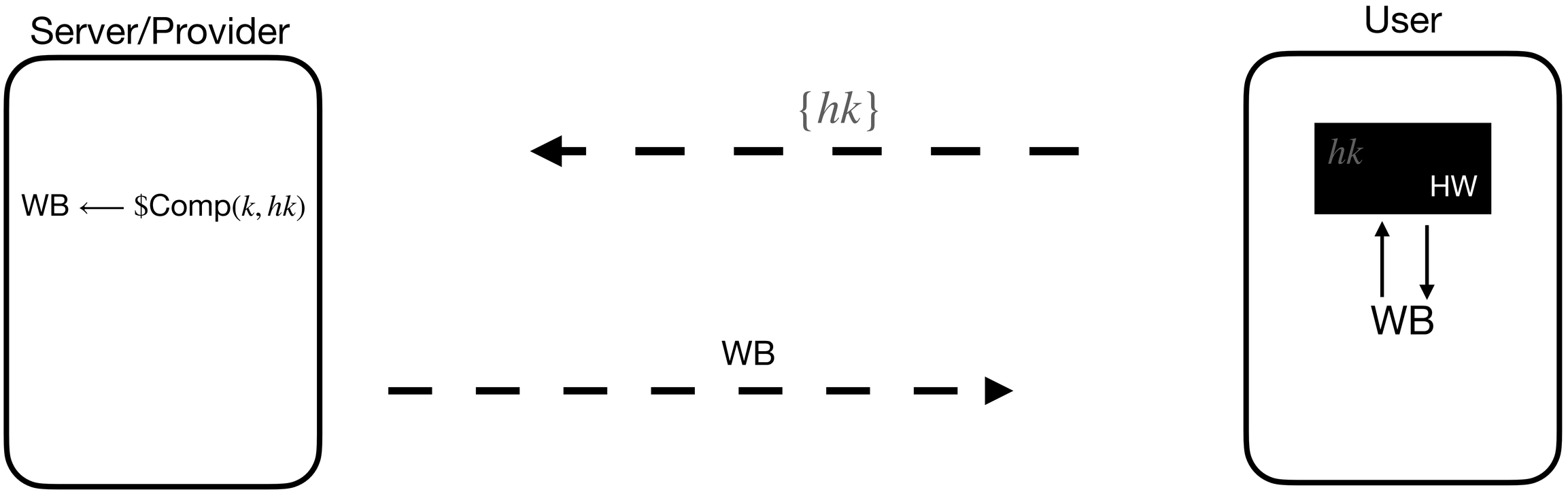


- The device binding process is *personalised*

[1] Alpirez Bock, Amadori, Brzuska and Michiels: On the security goals of white-box cryptography. CHES 2020

[2] Alpirez Bock, Brzuska, Fischlin, Janson and Michiels: Security reductions for white-box key storage. ASIACRYPT 2020

Previous work on Device-binding

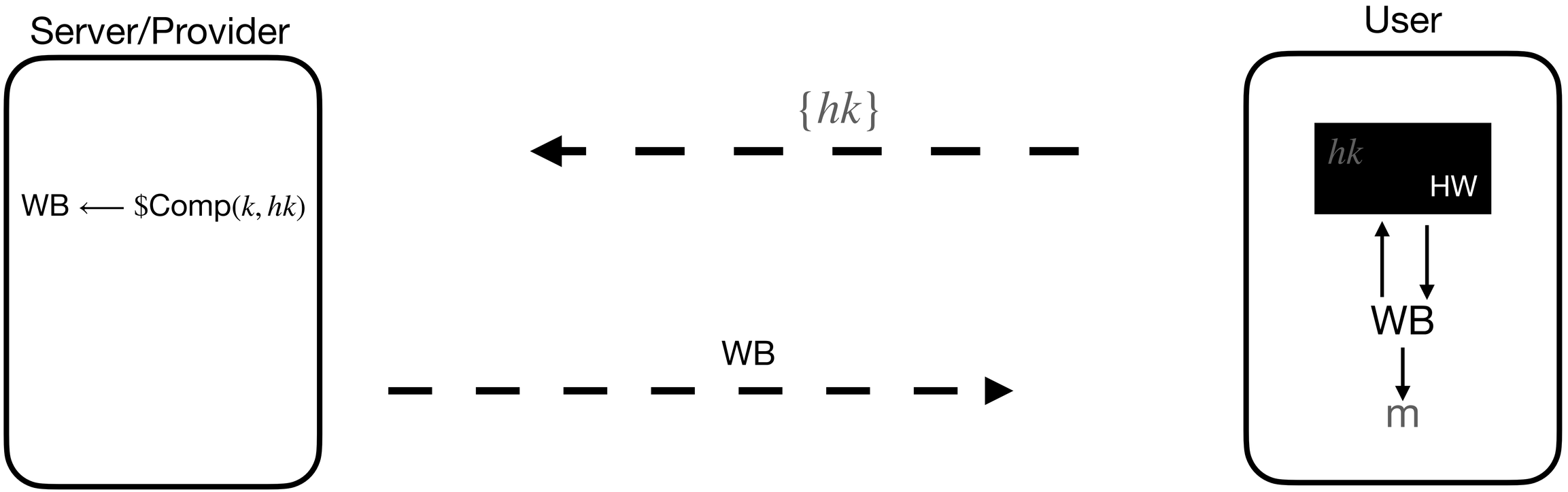


- The device binding process is *personalised*

[1] Alpirez Bock, Amadori, Brzuska and Michiels: On the security goals of white-box cryptography. CHES 2020

[2] Alpirez Bock, Brzuska, Fischlin, Janson and Michiels: Security reductions for white-box key storage. ASIACRYPT 2020

Previous work on Device-binding



- The device binding process is *personalised*

[1] Alpirez Bock, Amadori, Brzuska and Michiels: On the security goals of white-box cryptography. CHES 2020

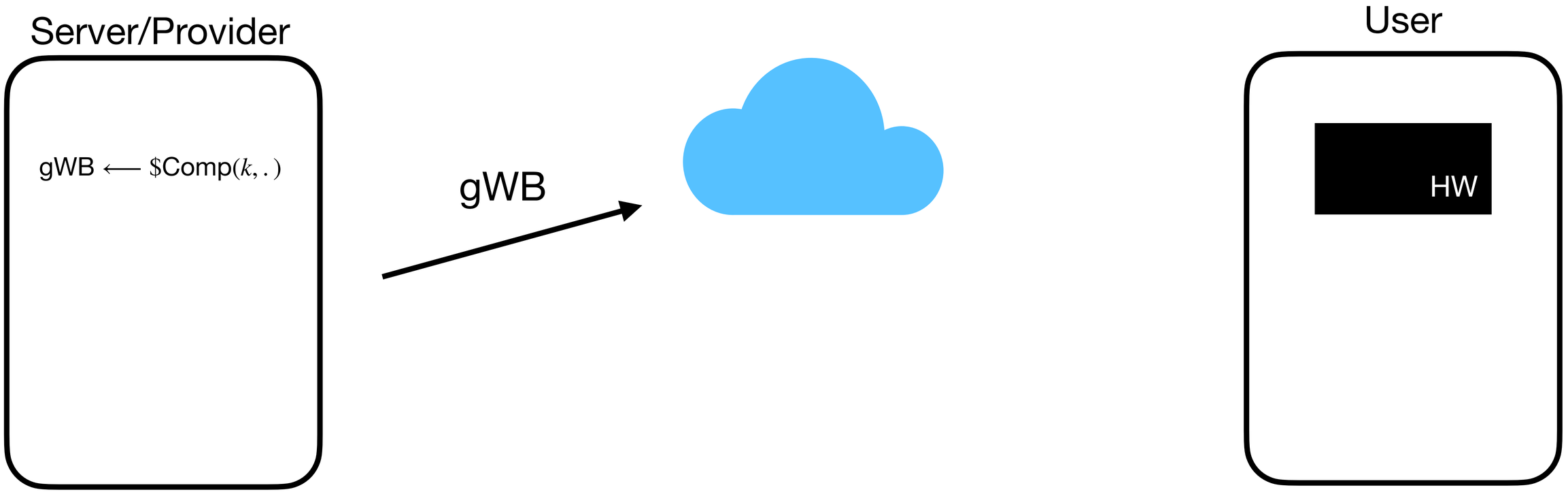
[2] Alpirez Bock, Brzuska, Fischlin, Janson and Michiels: Security reductions for white-box key storage. ASIACRYPT 2020

Global device-binding

Question: can we build a white-box program, which is compiled once, placed on the cloud, and securely bound to a number of devices?

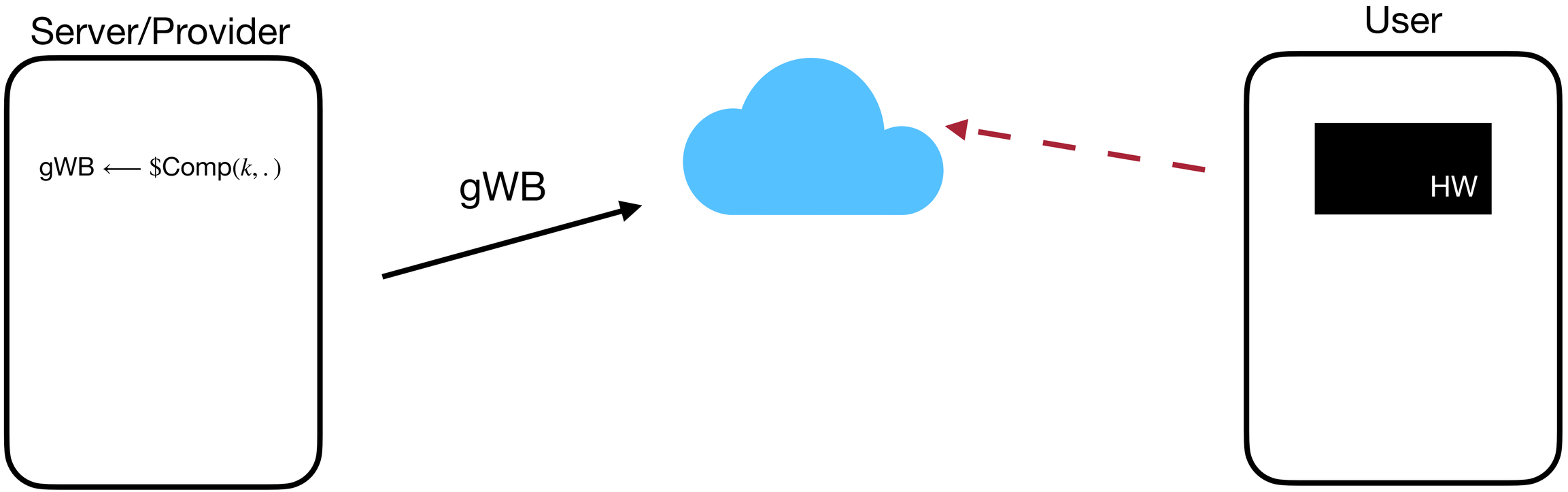
Global device-binding

Question: can we build a white-box program, which is compiled once, placed on the cloud, and securely bound to a number of devices?



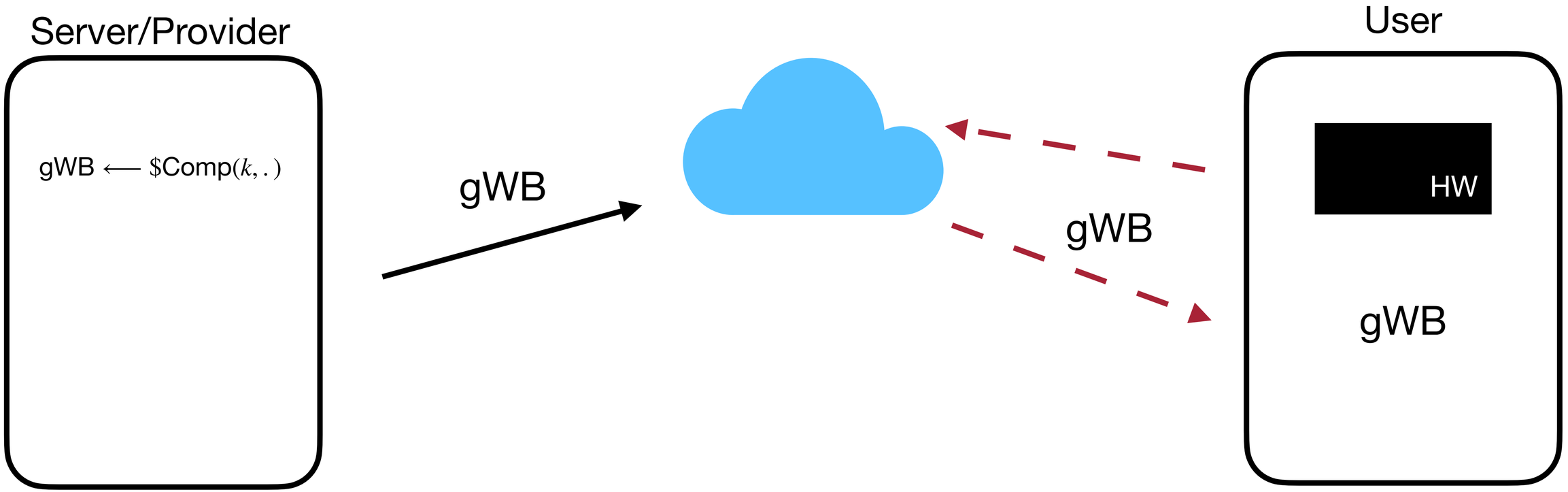
Global device-binding

Question: can we build a white-box program, which is compiled once, placed on the cloud, and securely bound to a number of devices?



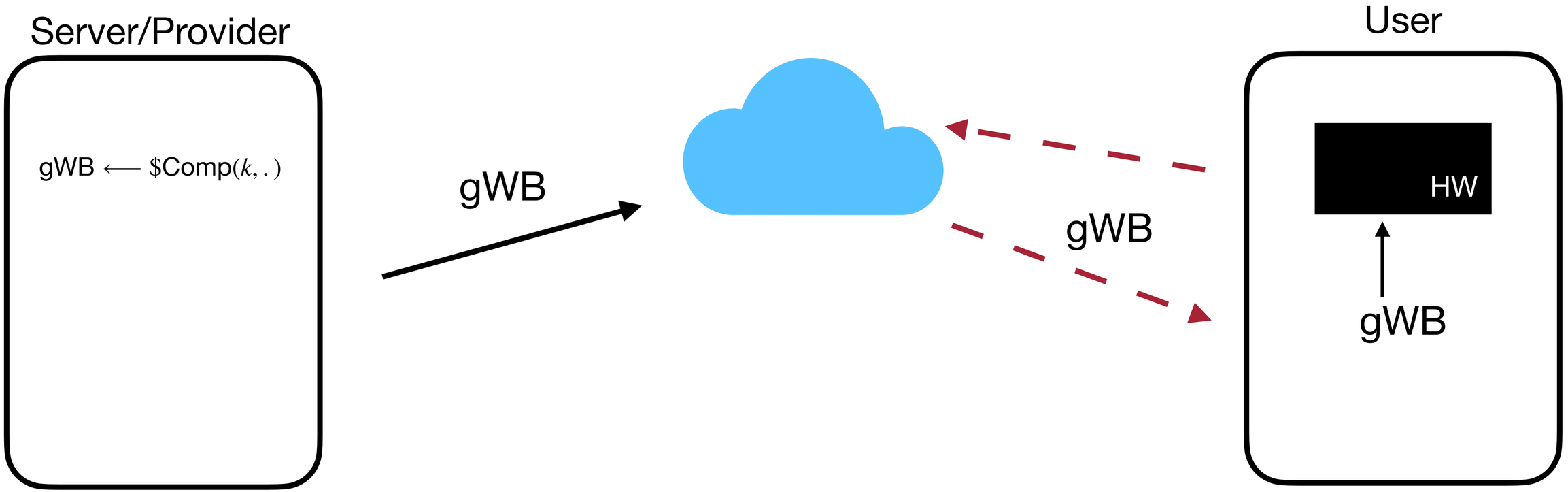
Global device-binding

Question: can we build a white-box program, which is compiled once, placed on the cloud, and securely bound to a number of devices?



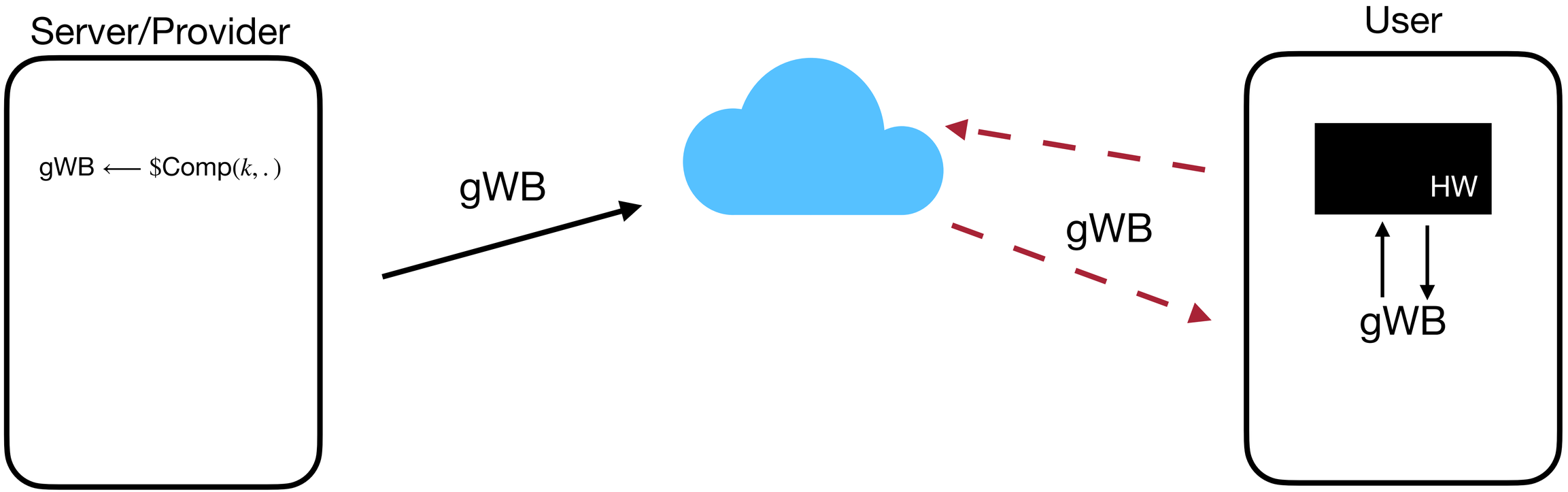
Global device-binding

Question: can we build a white-box program, which is compiled once, placed on the cloud, and securely bound to a number of devices?



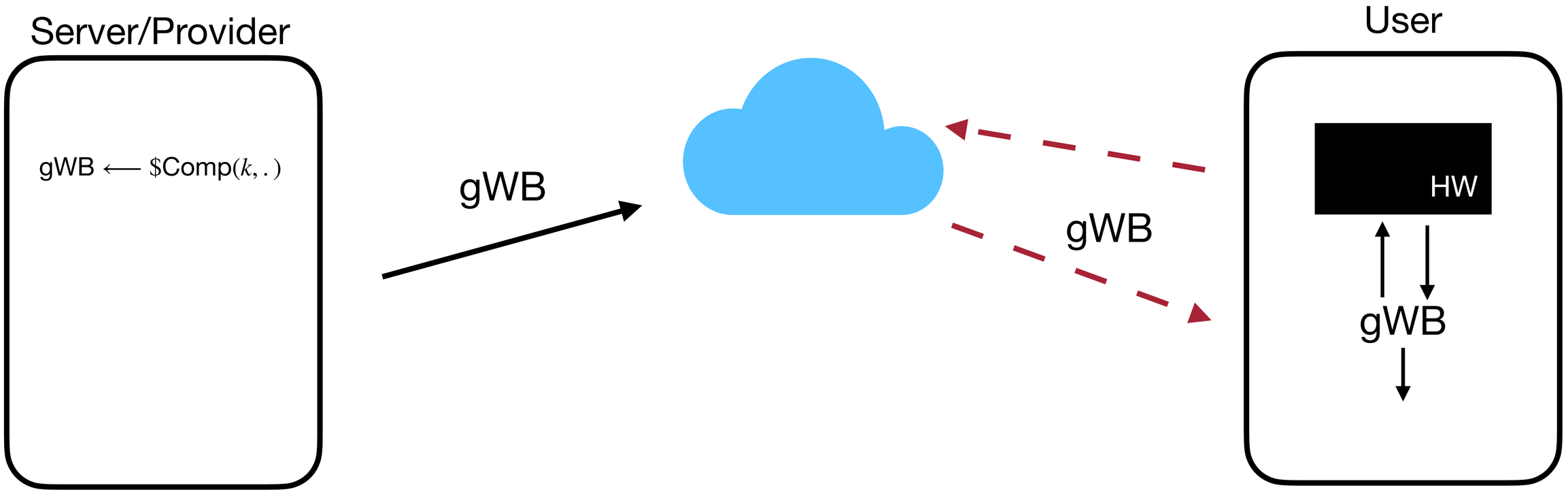
Global device-binding

Question: can we build a white-box program, which is compiled once, placed on the cloud, and securely bound to a number of devices?



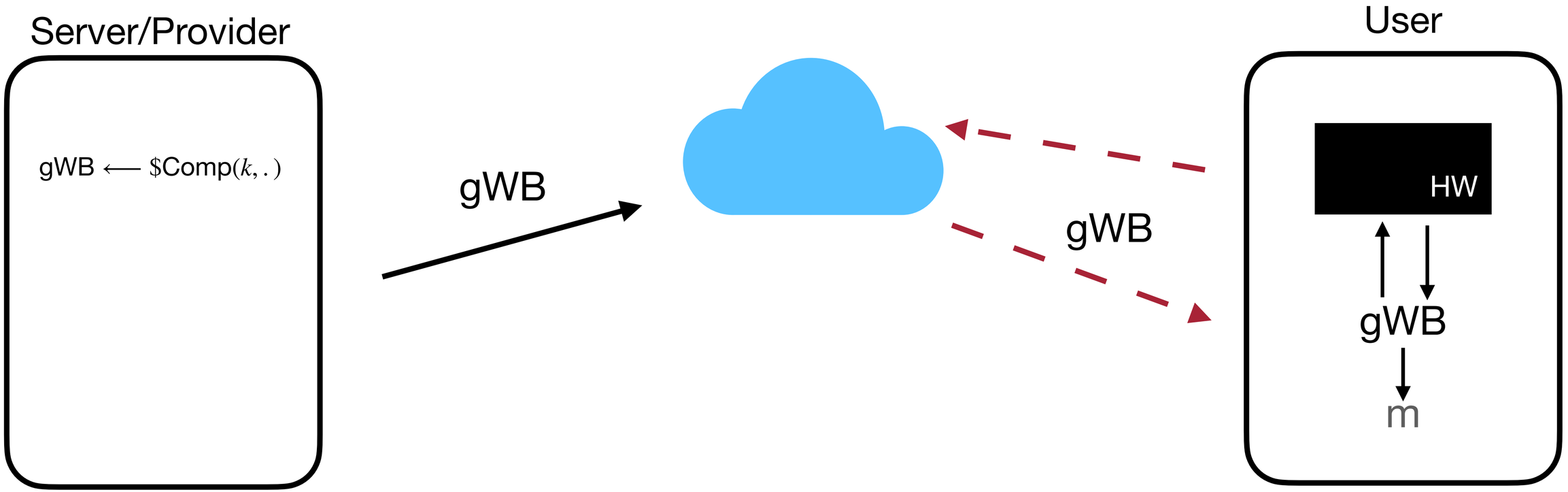
Global device-binding

Question: can we build a white-box program, which is compiled once, placed on the cloud, and securely bound to a number of devices?



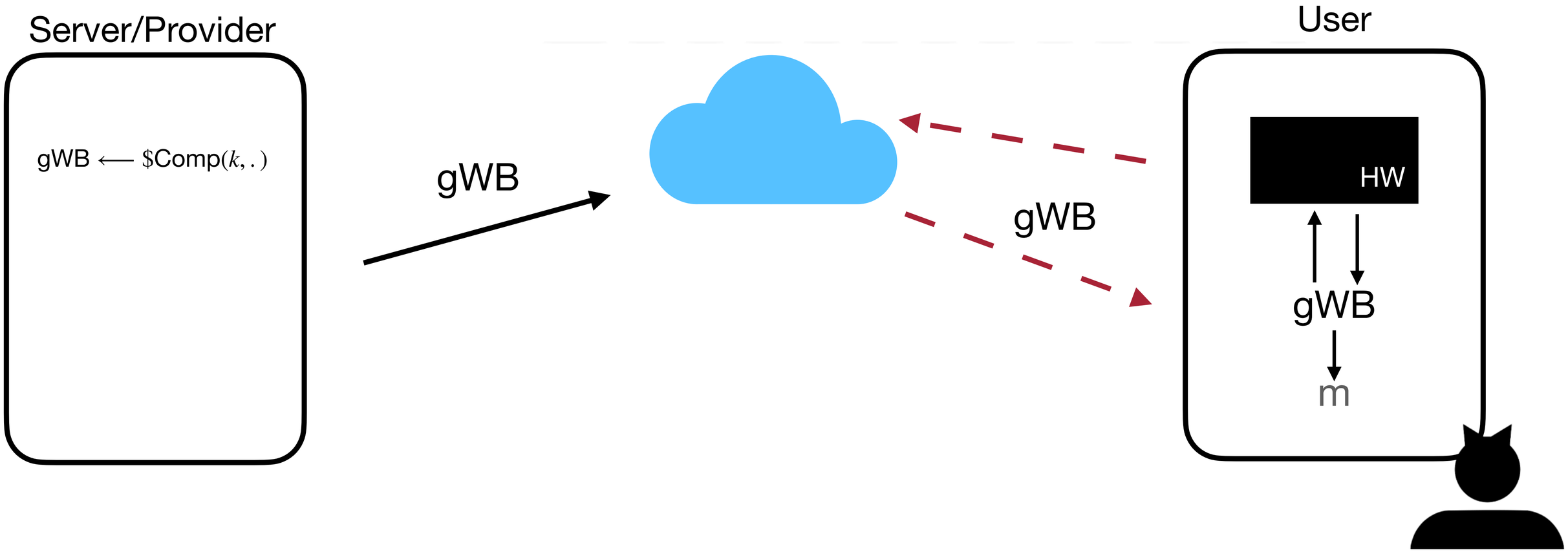
Global device-binding

Question: can we build a white-box program, which is compiled once, placed on the cloud, and securely bound to a number of devices?



Global device-binding

Question: can we build a white-box program, which is compiled once, placed on the cloud, and securely bound to a number of devices?



Idea: white-box on encoded inputs

Design the white-box s.t. it always computers on encoded inputs

$\mathsf{HW}(c)$

$\tilde{c} \leftarrow \mathsf{Encode}(e_s, c)$

return $\tilde{c}$

$\mathsf{WB}(\tilde{c})$

$c \leftarrow \mathsf{Decode}(d_s, \tilde{c})$

$m \leftarrow \mathsf{Dec}(k, c)$

return m

Idea: white-box on encoded inputs

Design the white-box s.t. it always computers on encoded inputs

$\mathsf{HW}(c)$

$\tilde{c} \leftarrow \mathsf{Encode}(e_s, c)$

return $\tilde{c}$

$\mathsf{WB}(\tilde{c})$

$c \leftarrow \mathsf{Decode}(d_s, \tilde{c})$

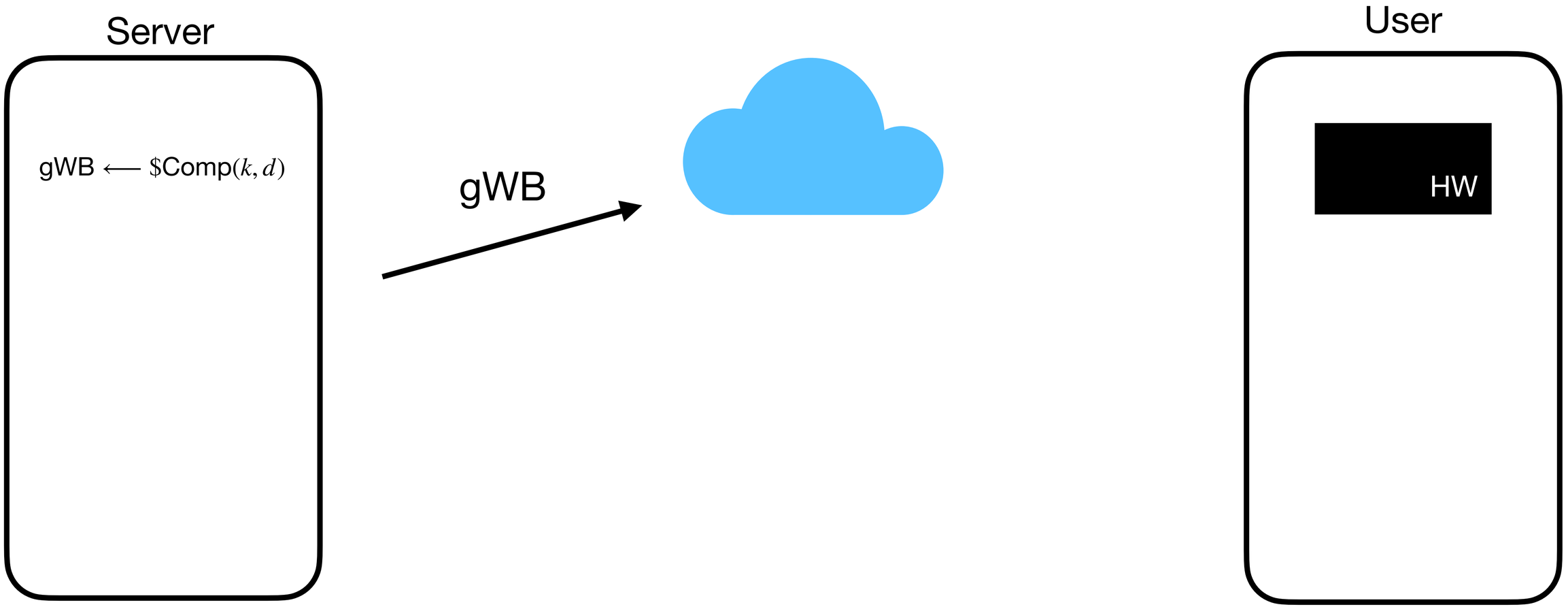
$m \leftarrow \mathsf{Dec}(k, c)$

return m

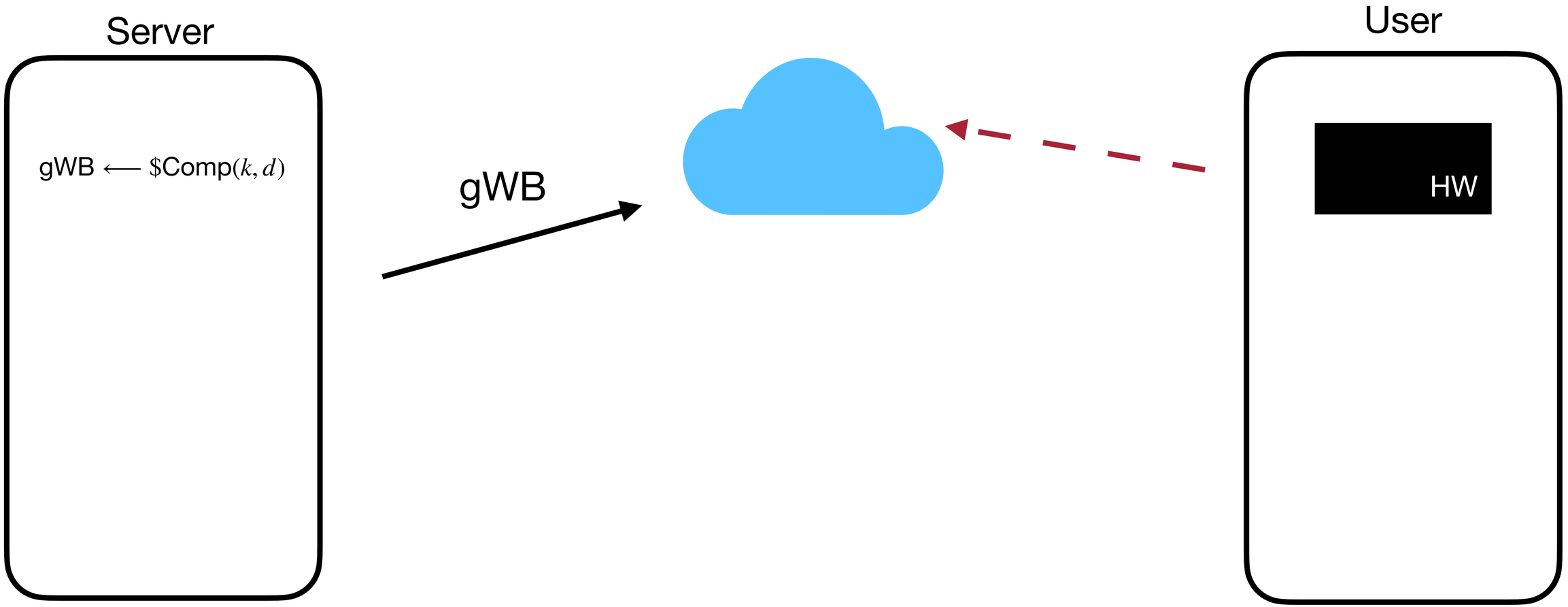
The encoding function will be available on *certified* devices

Global white-box

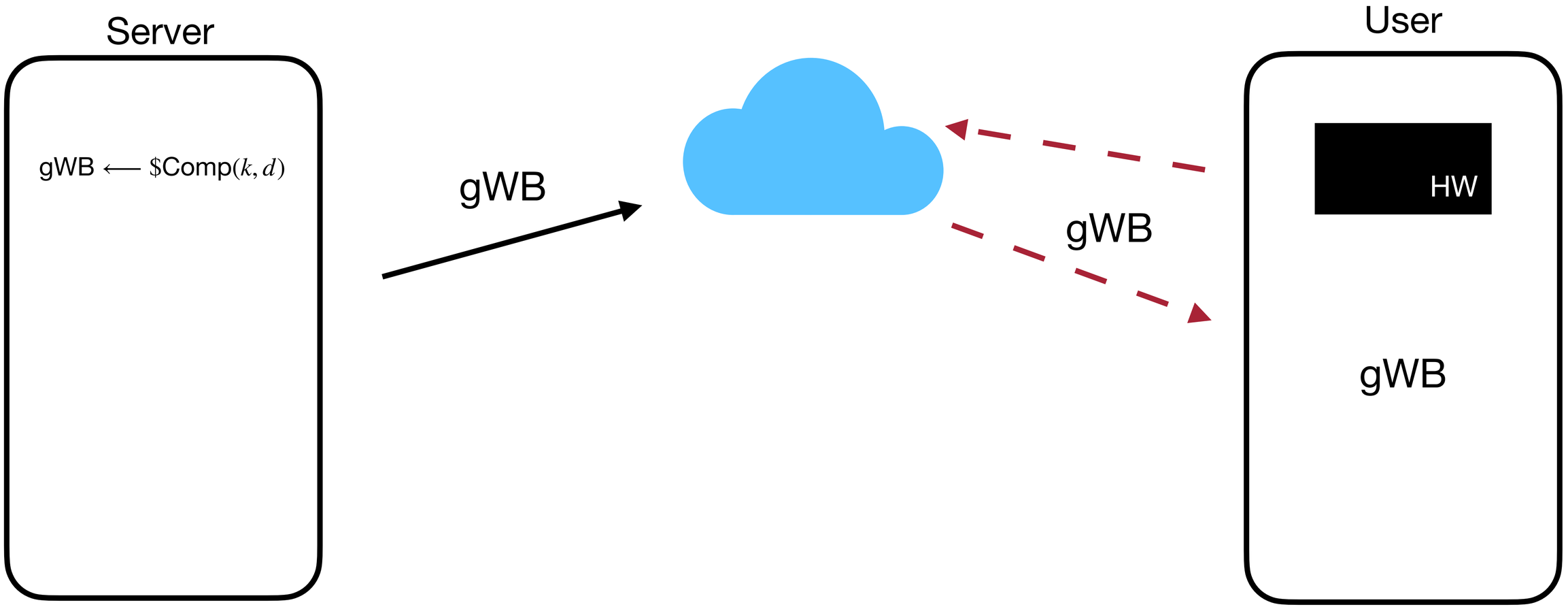
Global white-box



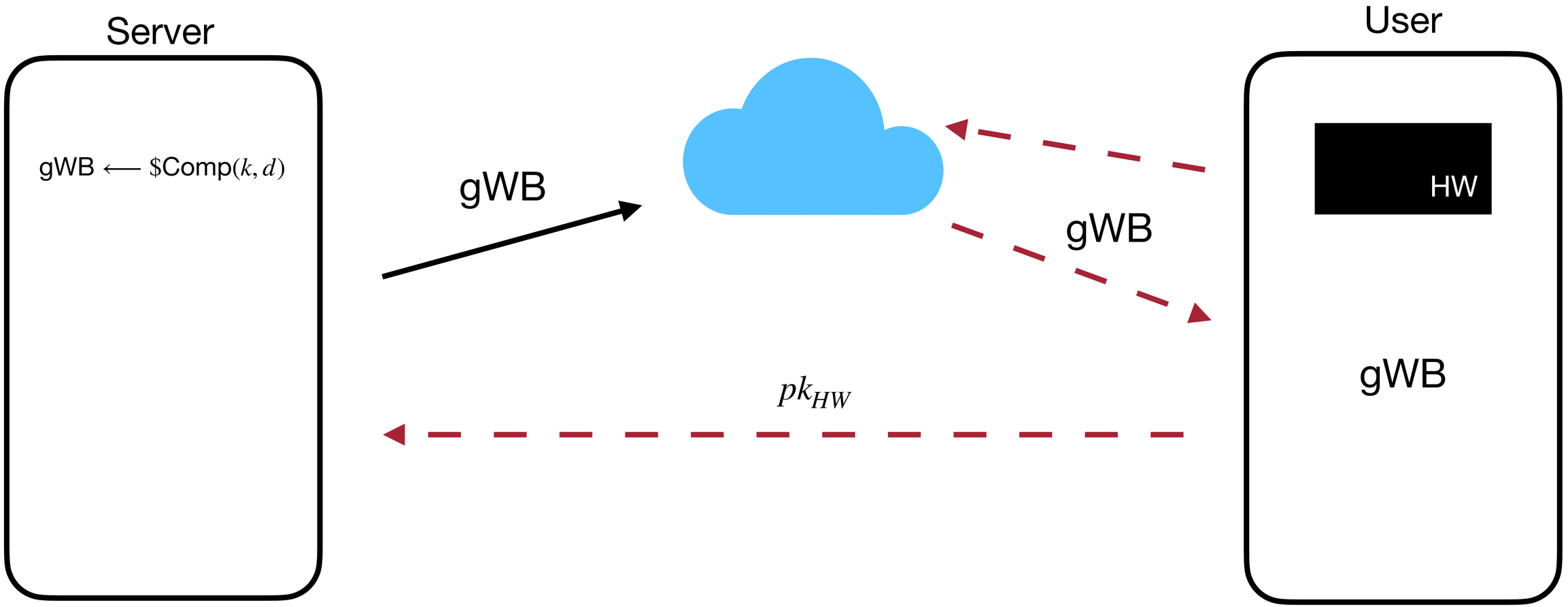
Global white-box



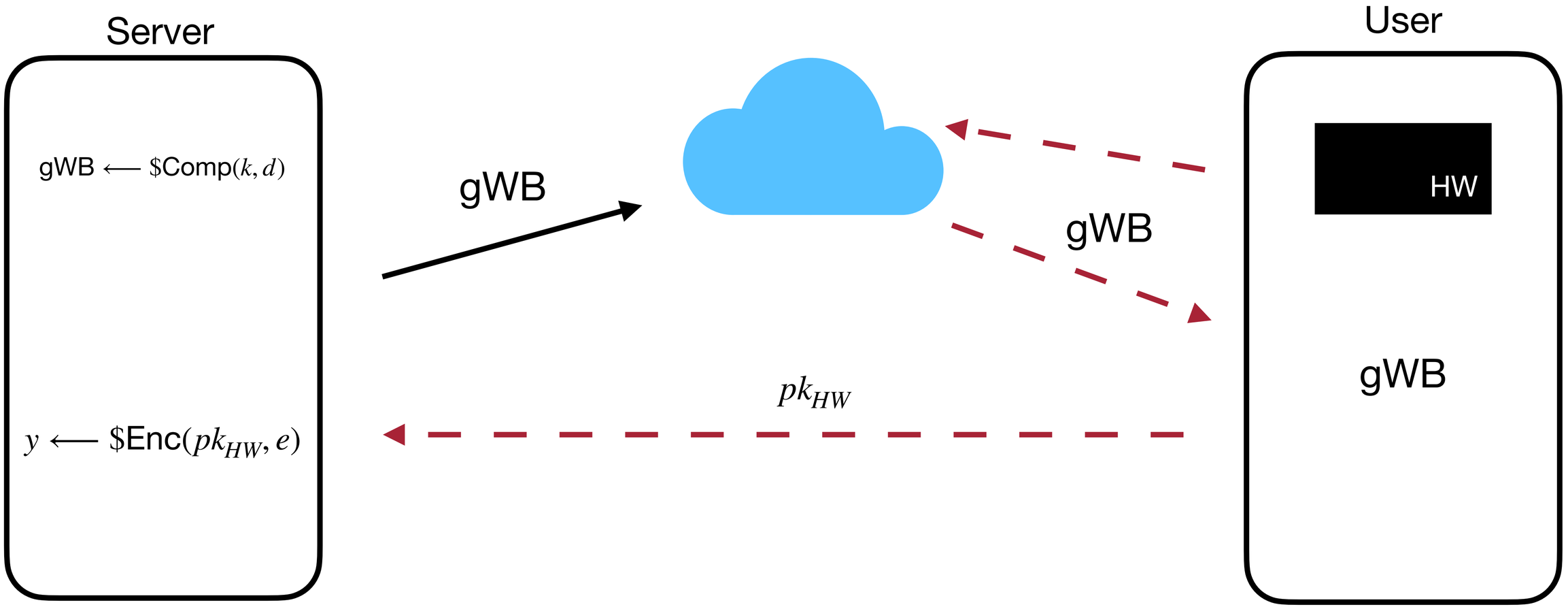
Global white-box



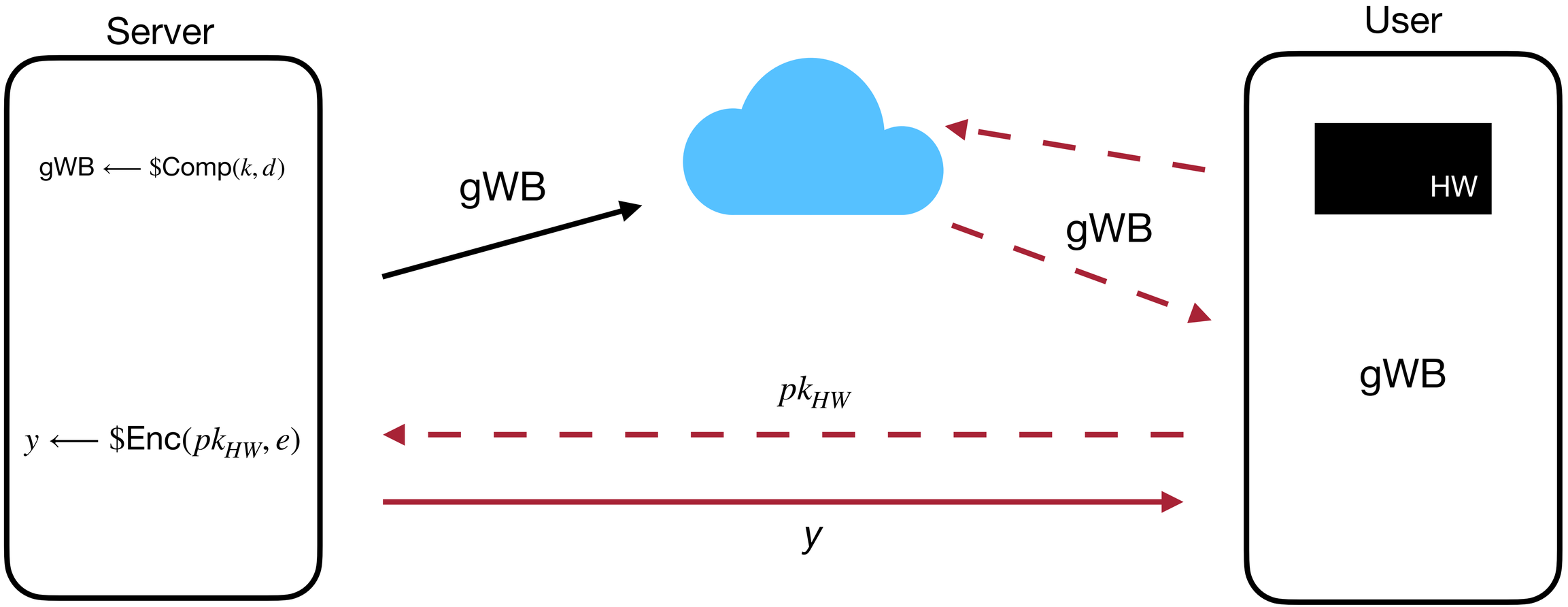
Global white-box



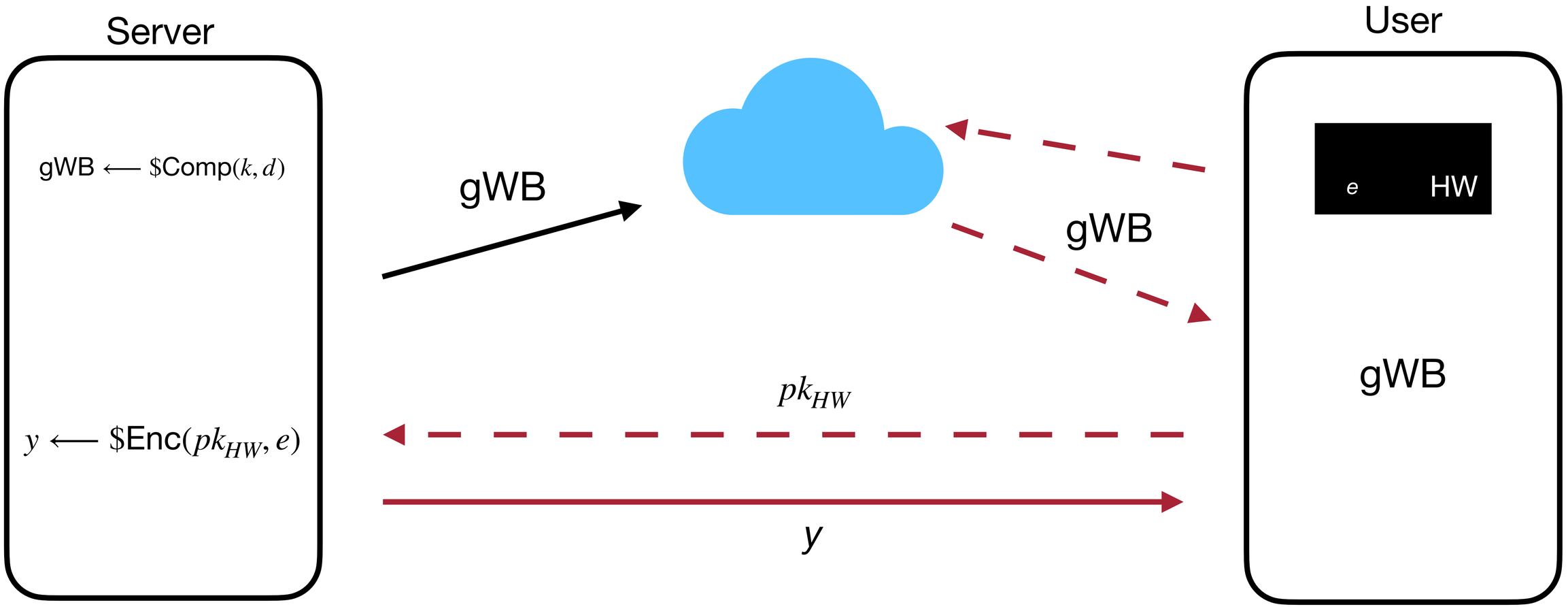
Global white-box



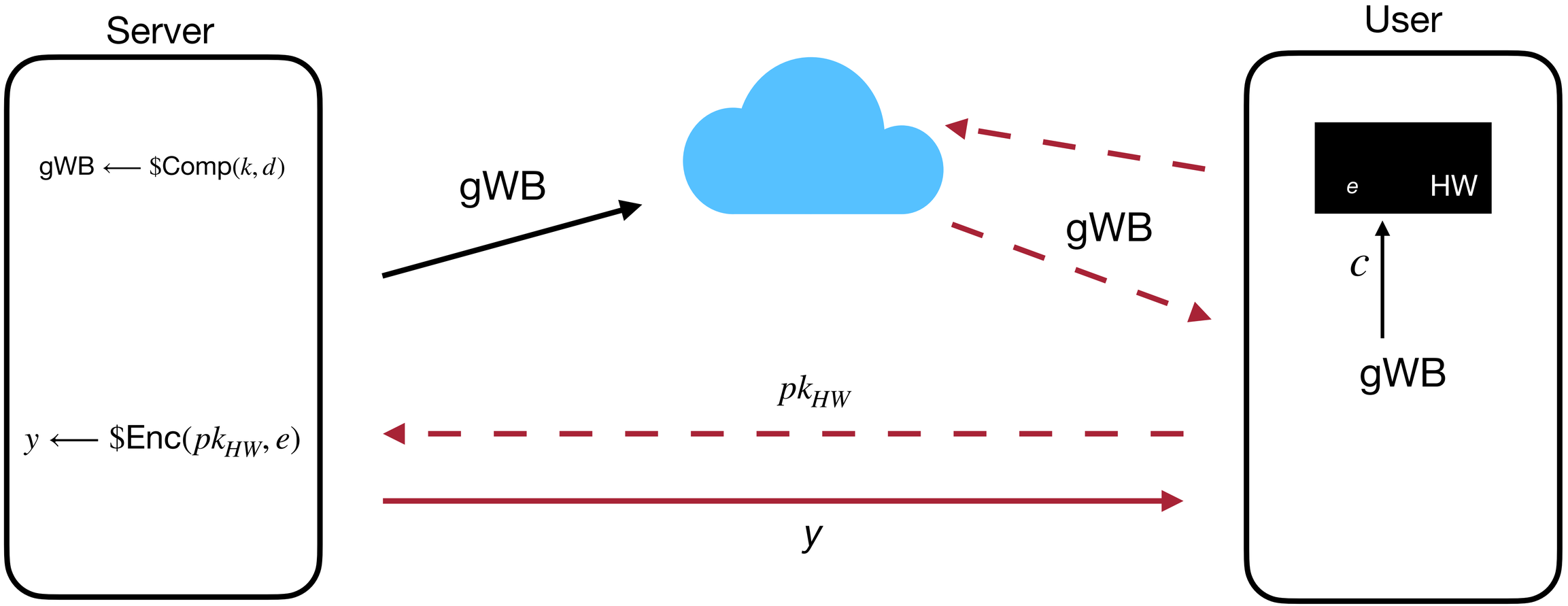
Global white-box



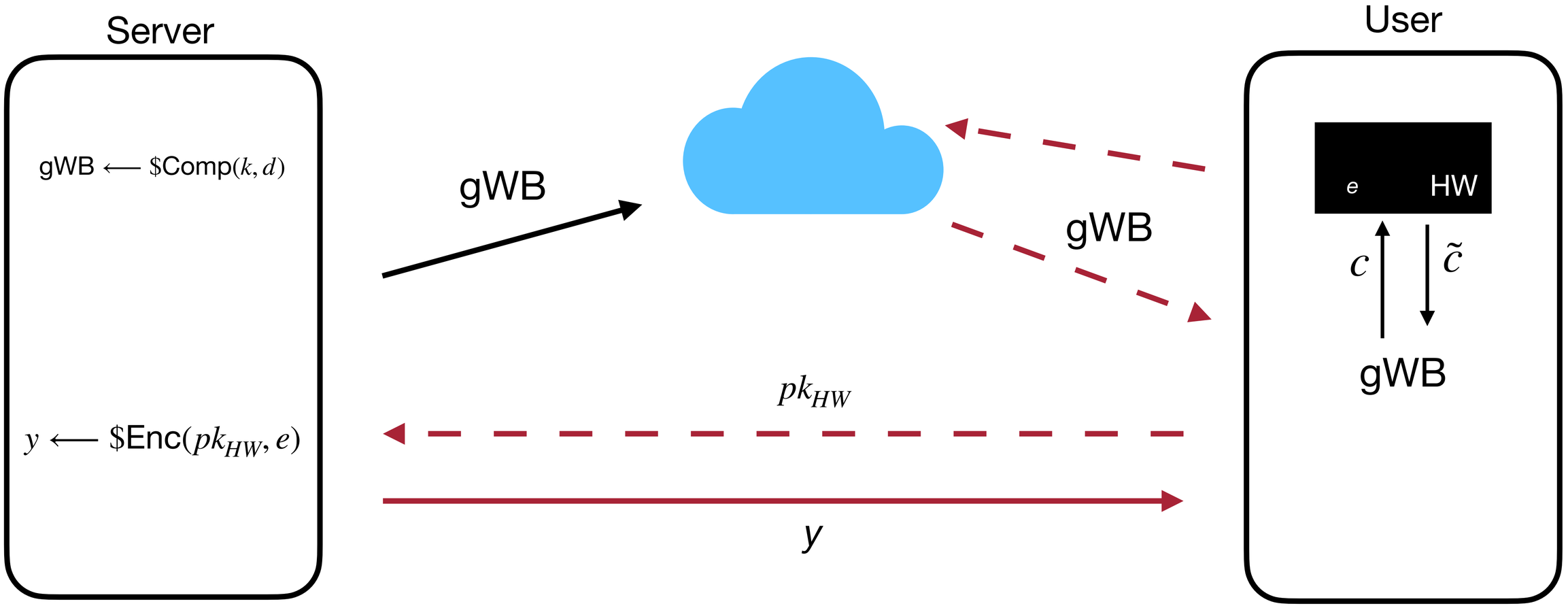
Global white-box



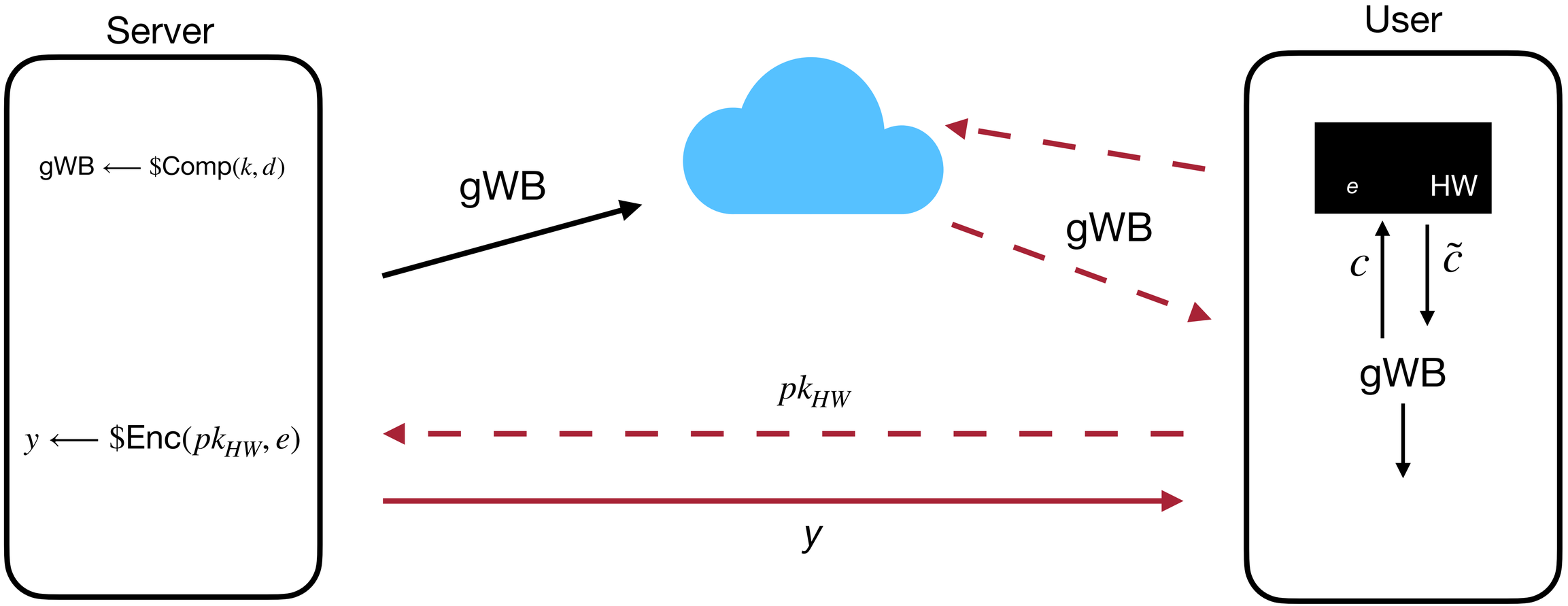
Global white-box



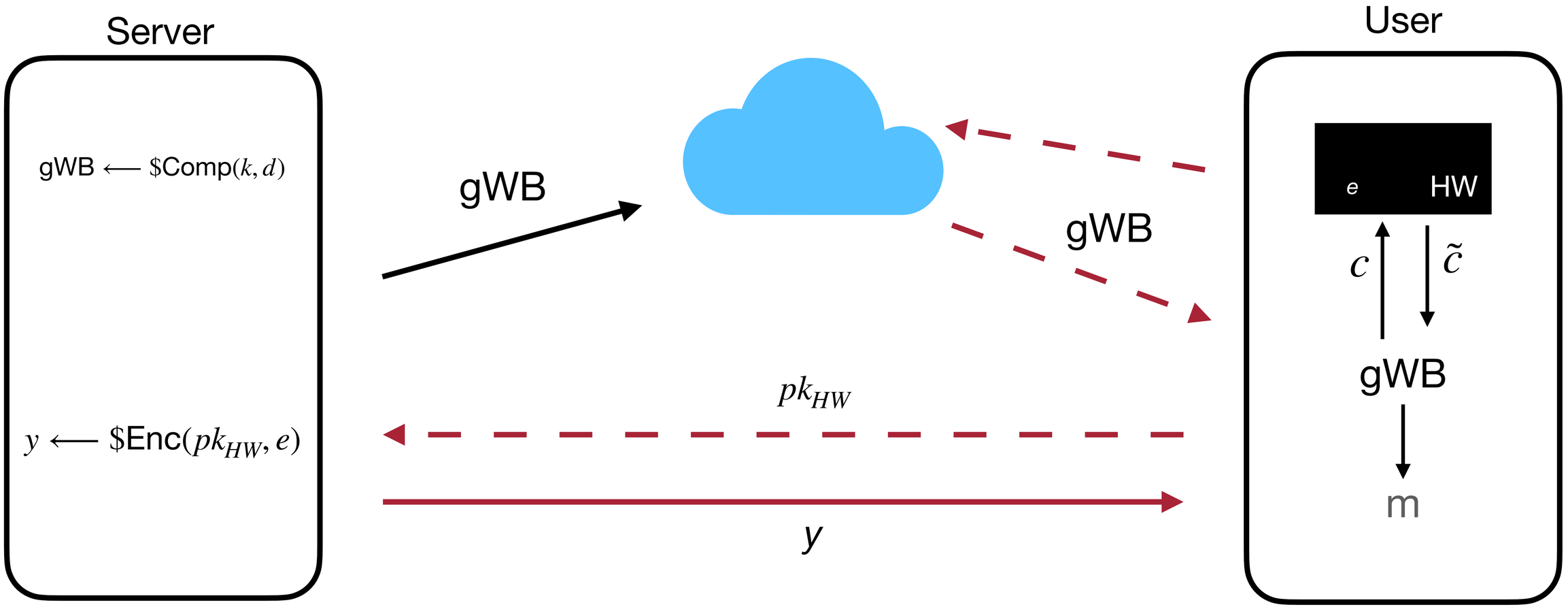
Global white-box



Global white-box



Global white-box



Global white-boxes via TBO

We can directly build such gWB via Token-based obfuscation [3]

$$\text{TBO.Obf}(C)\$ \longrightarrow \text{MSK}, O[C]$$

$$\text{TBO.InpGen}(\text{MSK}, x)\$ \longrightarrow \tilde{x}$$

$$O[C]\ (\text{InpGen}(\text{MSK}, m)) \ = \ C(m)$$

$$O[C]\ (\tilde{m}) \ = \ C(m)$$

—> Just place the token-generation algorithm in the secure hardware

[3] Goldwasser et al.: Reusable garbled circuits and succinct functional encryption. STOC 2013

Global white-boxes via TBO - drawbacks

Global white-boxes via TBO - drawbacks

We need to assume that each secure hardware can load a token-generation key

Global white-boxes via TBO - drawbacks

We need to assume that each secure hardware can load a token-generation key

> Note: this assumption is only needed for global white-boxes, not for individual ones in the style of previous works [1,2]

Global white-boxes via TBO - drawbacks

We need to assume that each secure hardware can load a token-generation key

Note: this assumption is only needed for global white-boxes, not for individual ones in the style of previous works [1,2]

In terms of feasibility, our construction improves previous ones as it is built from LWE, not iO, and can obfuscate *any* function

Global white-boxes via TBO - drawbacks

We need to assume that each secure hardware can load a token-generation key

Note: this assumption is only needed for global white-boxes, not for individual ones in the style of previous works [1,2]

In terms of feasibility, our construction improves previous ones as it is built from LWE, not iO, and can obfuscate *any* function

All devices store the same encoding key

Global white-boxes via TBO - drawbacks

We need to assume that each secure hardware can load a token-generation key

Note: this assumption is only needed for global white-boxes, not for individual ones in the style of previous works [1,2]

In terms of feasibility, our construction improves previous ones as it is built from LWE, not iO, and can obfuscate *any* function

All devices store the same encoding key

Extract *d* from one HW —> run any white-box you want

Global white-boxes via TBO - drawbacks

We need to assume that each secure hardware can load a token-generation key

Note: this assumption is only needed for global white-boxes, not for individual ones in the style of previous works [1,2]

In terms of feasibility, our construction improves previous ones as it is built from LWE, not iO, and can obfuscate *any* function

All devices store the same encoding key

Extract *d* from one HW —> run any white-box you want

Can we provide an alternative?

Strong global white-boxes via message-recoverable signatures [4]

[4] Bellare and Rogaway: The exact security of digital signatures - how to sign with RSA and Rabin. Eurocrypt 96

Strong global white-boxes via message-recoverable signatures [4]

$$\text{KGen\$} \longrightarrow sk, pk$$

[4] Bellare and Rogaway: The exact security of digital signatures - how to sign with RSA and Rabin. Eurocrypt 96

Strong global white-boxes via message-recoverable signatures [4]

$$\text{KGen}\$ \longrightarrow sk, pk$$

$$\text{Sign}(sk, m)\$ \longrightarrow \tilde{m}$$

[4] Bellare and Rogaway: The exact security of digital signatures - how to sign with RSA and Rabin. Eurocrypt 96

Strong global white-boxes via message-recoverable signatures [4]

$$\text{KGen}\$ \longrightarrow sk, pk$$

$$\text{Sign}(sk, m)\$ \longrightarrow \tilde{m}$$

$$\text{Rcvr}(pk, \tilde{m}) \longrightarrow m \mid \perp$$

[4] Bellare and Rogaway: The exact security of digital signatures - how to sign with RSA and Rabin. Eurocrypt 96

Strong global white-boxes via message-recoverable signatures [4]

$$\text{KGen}\$ \longrightarrow sk, pk$$

$$\text{Sign}(sk, m)\$ \longrightarrow \tilde{m}$$

$$\text{Rcvr}(pk, \tilde{m}) \longrightarrow m \mid \perp$$

$$\text{Rcvr}(pk, \text{Sign}(sk, m)) \longrightarrow m$$

[4] Bellare and Rogaway: The exact security of digital signatures - how to sign with RSA and Rabin. Eurocrypt 96

Strong global white-boxes via message-recoverable signatures [4]

$$\text{KGen}\$ \longrightarrow sk, pk$$

$$\text{Sign}(sk, m)\$ \longrightarrow \tilde{m}$$

$$\text{Rcvr}(pk, \tilde{m}) \longrightarrow m \mid \perp$$

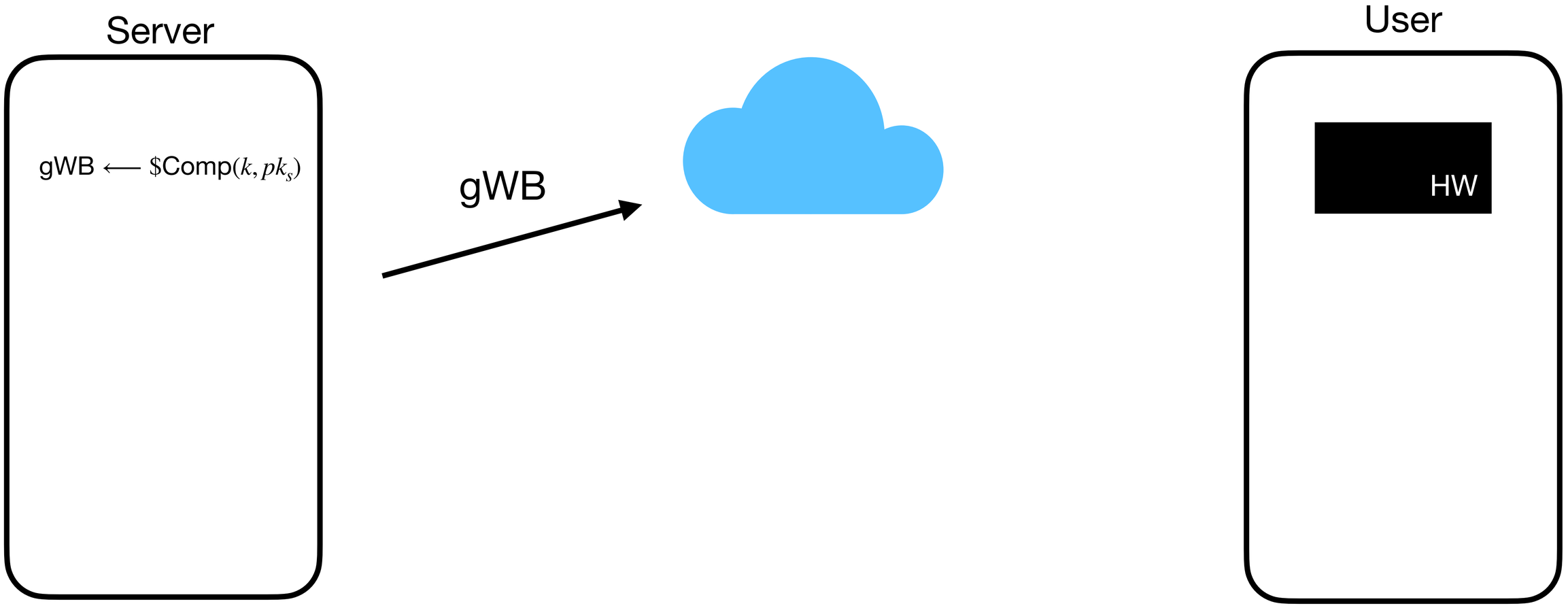
$$\text{Rcvr}(pk, \text{Sign}(sk, m)) \longrightarrow m$$

Use secret key for encoding and pk for decoding

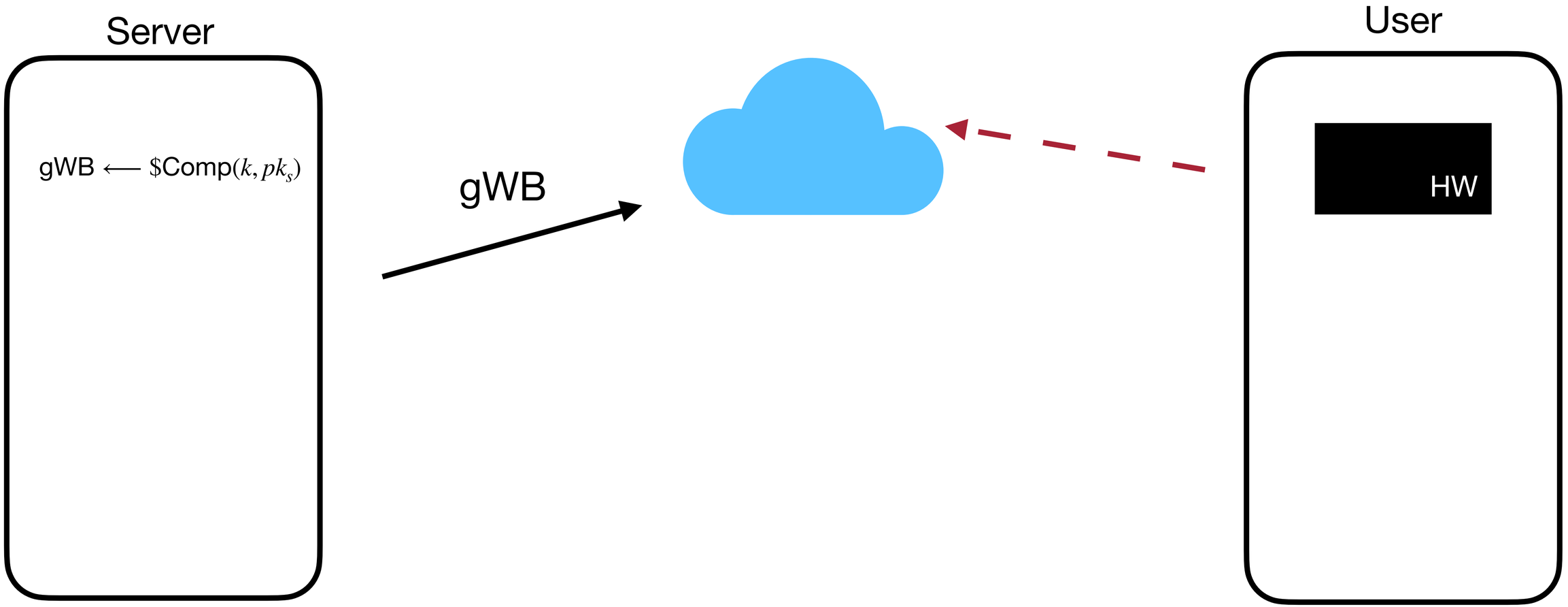
[4] Bellare and Rogaway: The exact security of digital signatures - how to sign with RSA and Rabin. Eurocrypt 96

Strong global white-boxes

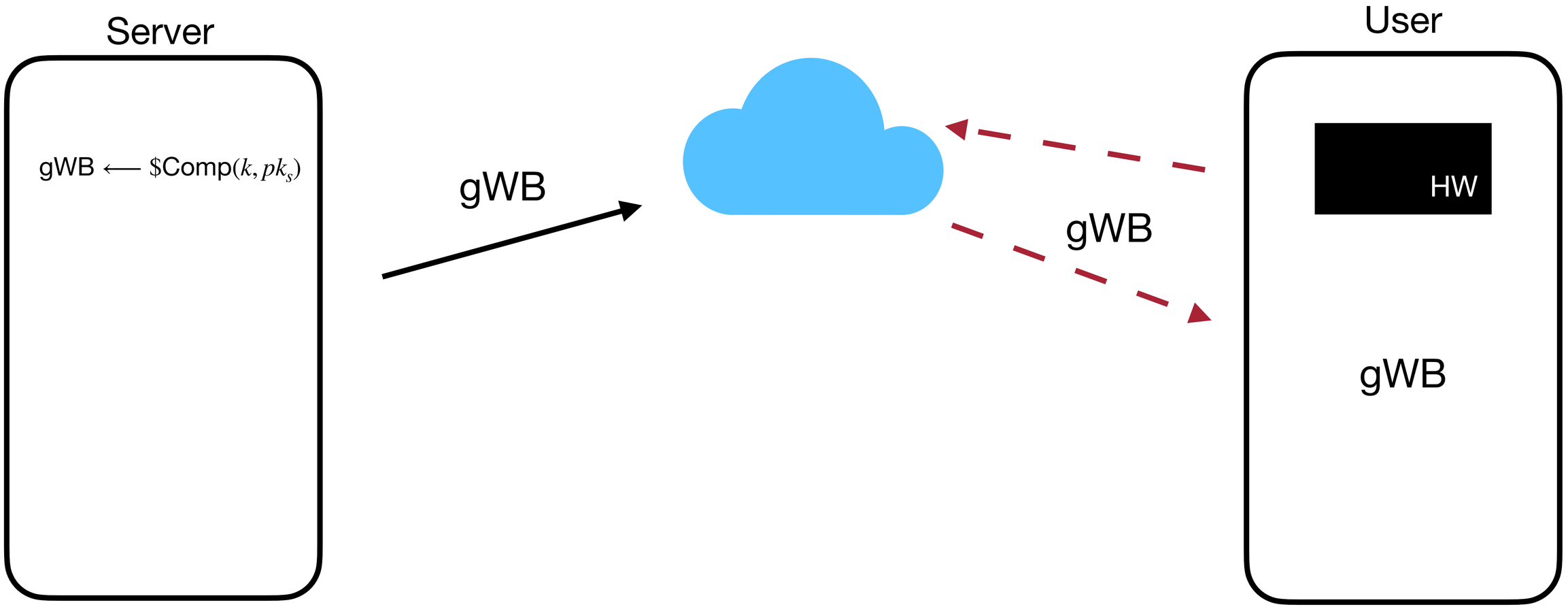
Strong global white-boxes



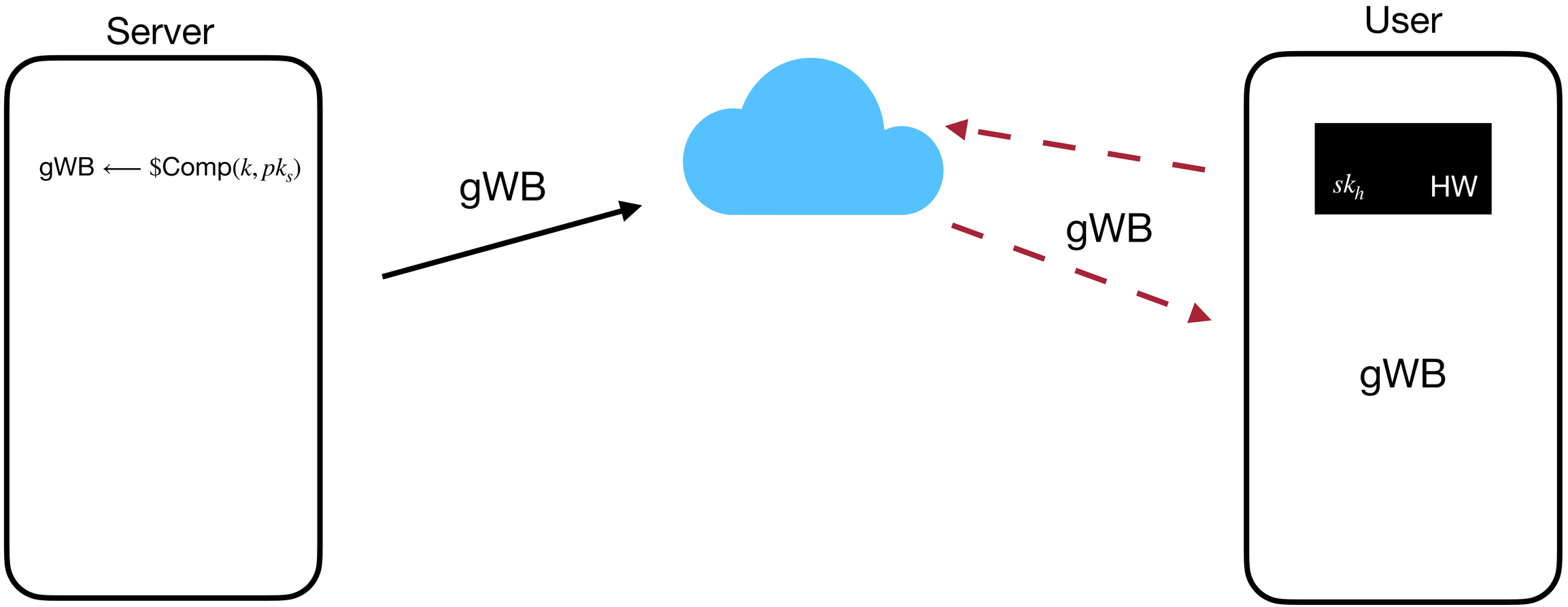
Strong global white-boxes



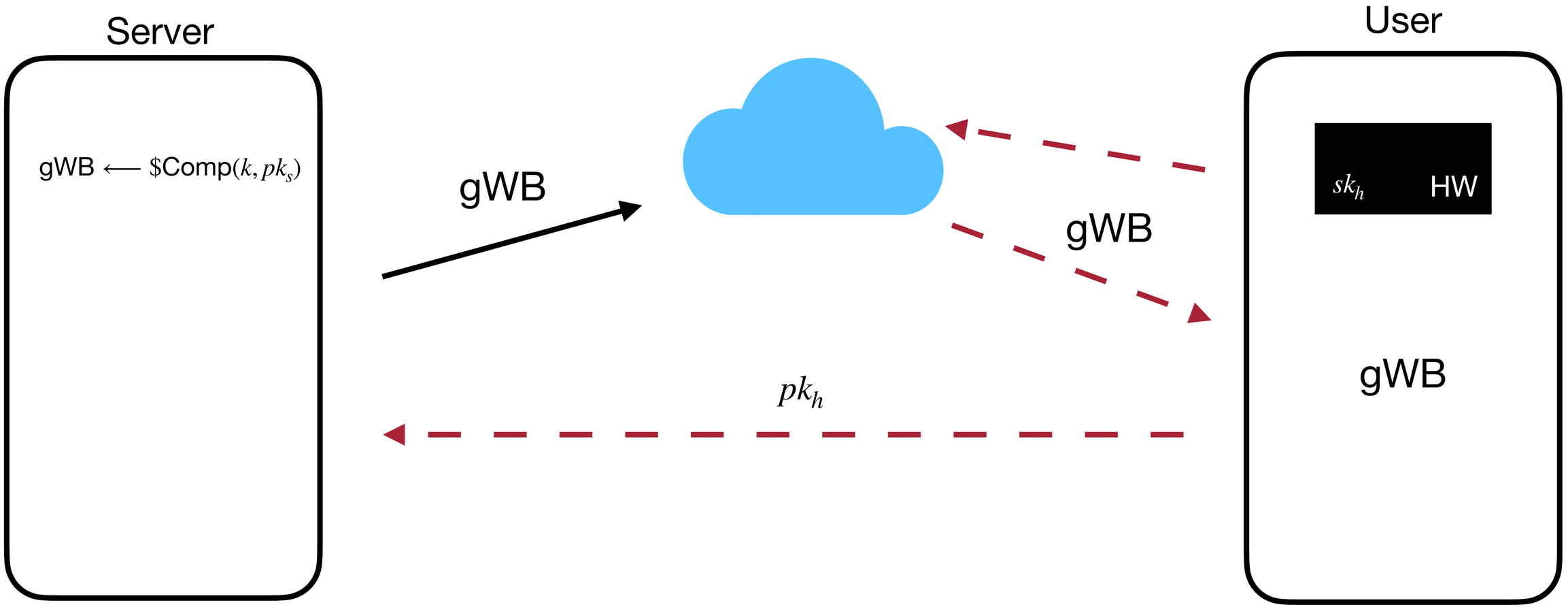
Strong global white-boxes



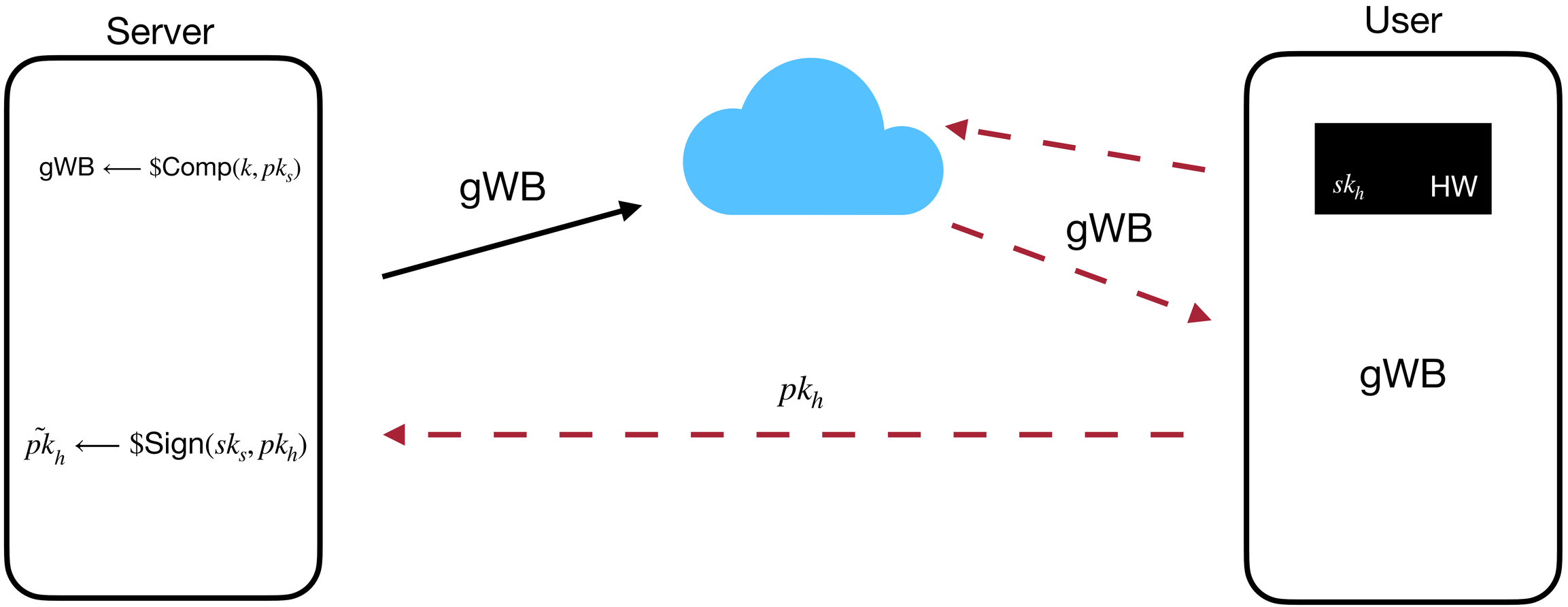
Strong global white-boxes



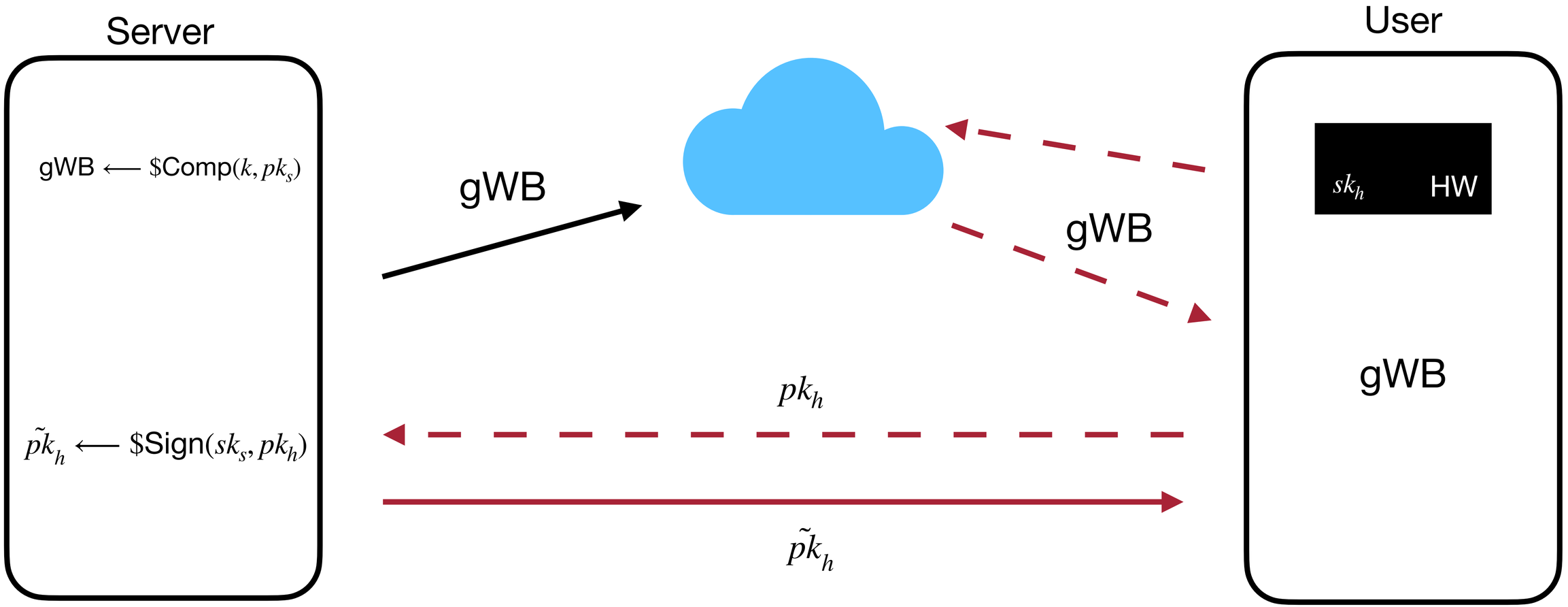
Strong global white-boxes



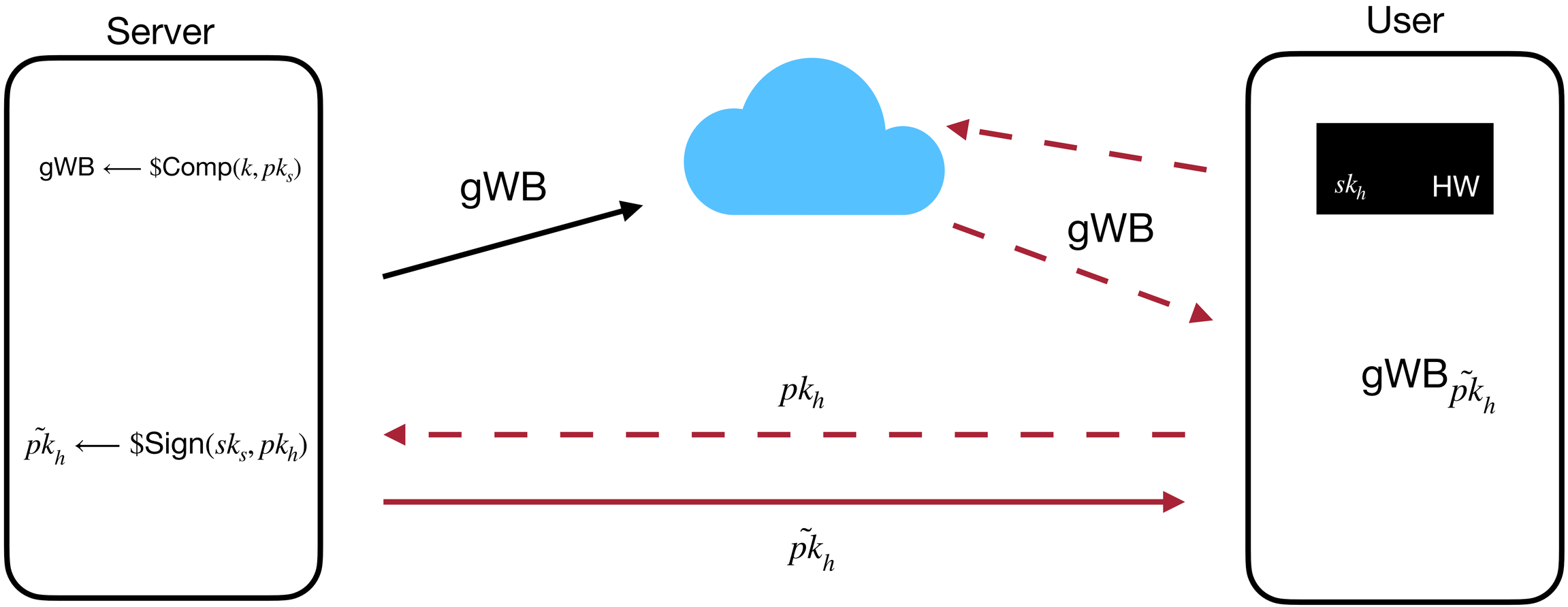
Strong global white-boxes



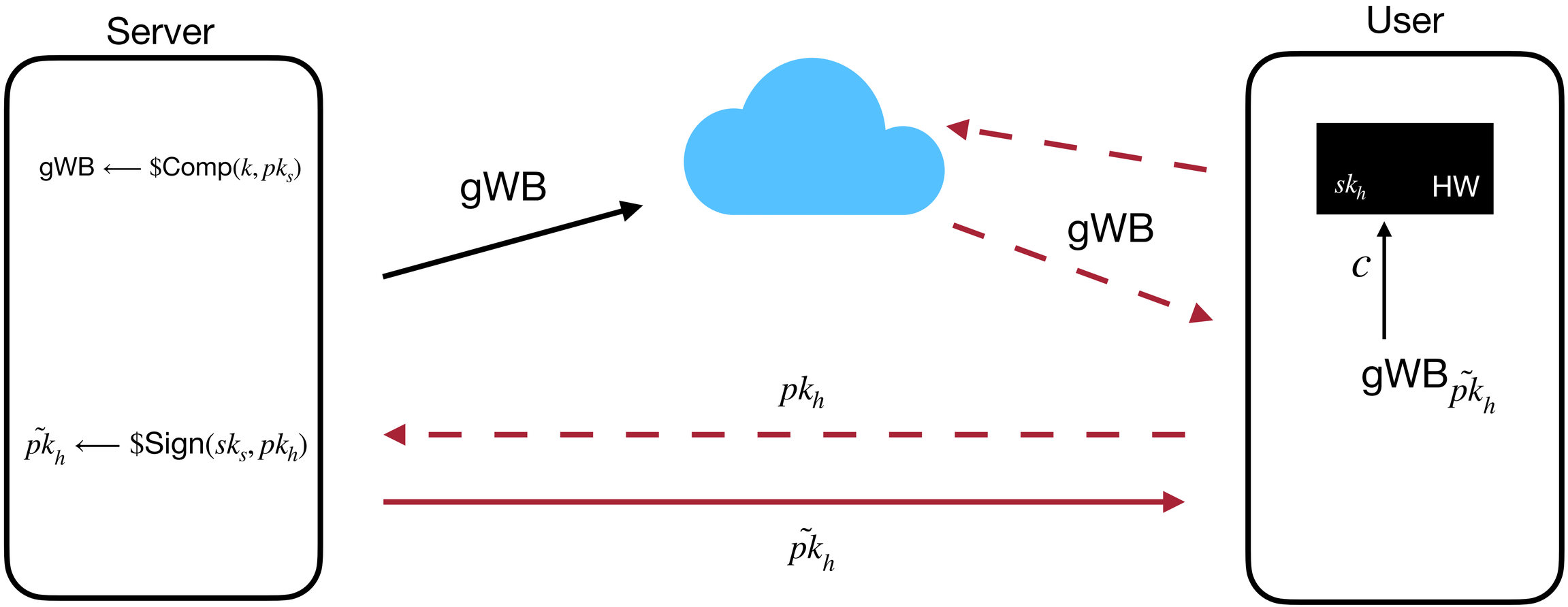
Strong global white-boxes



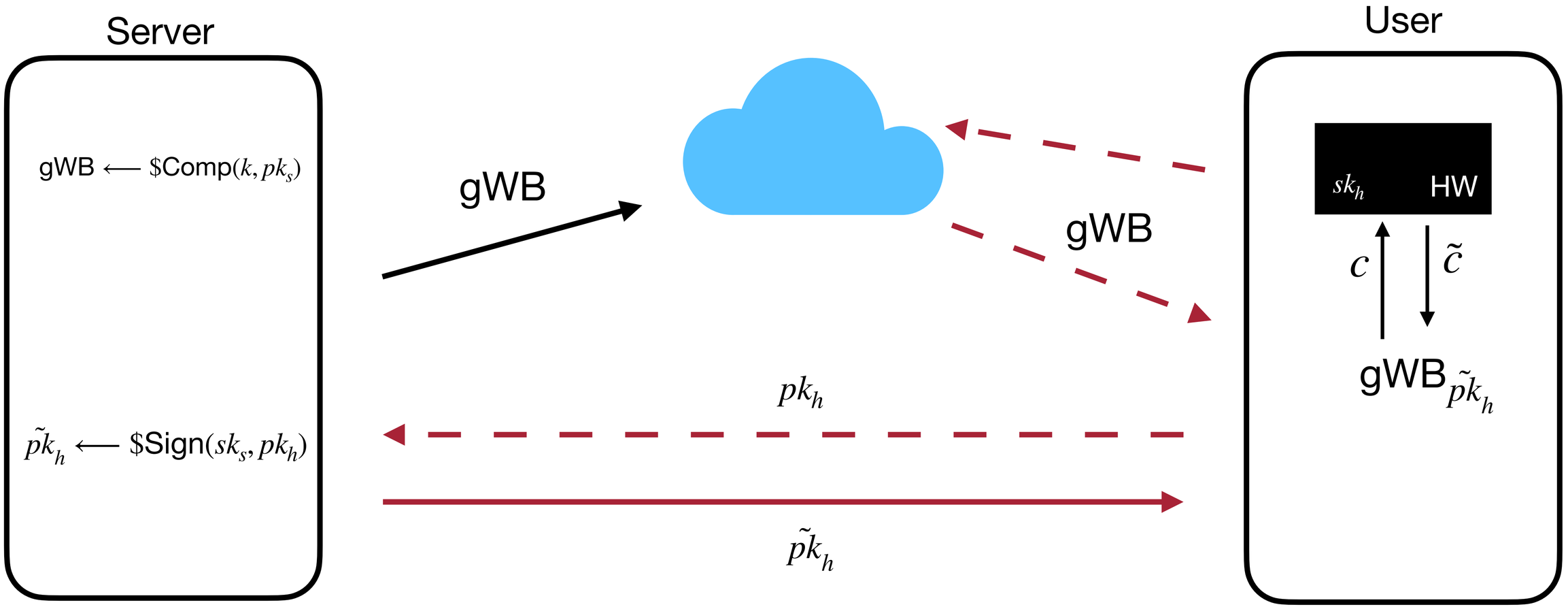
Strong global white-boxes



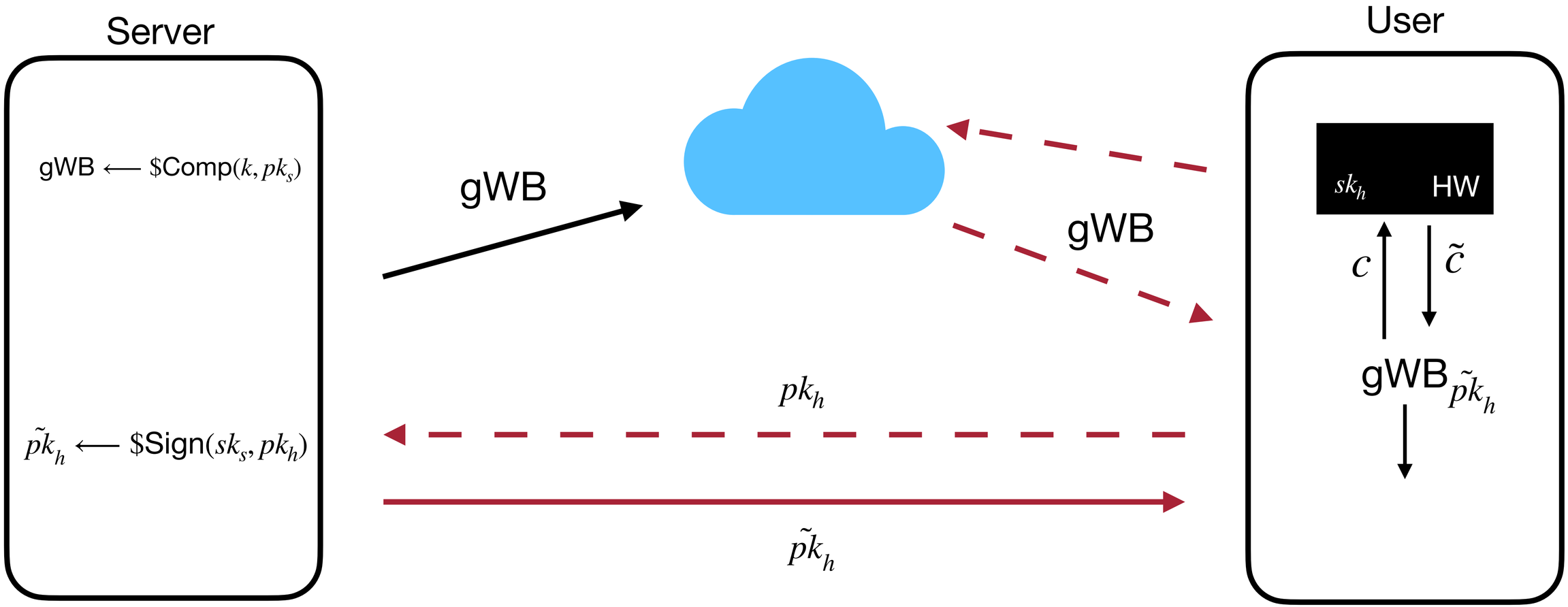
Strong global white-boxes



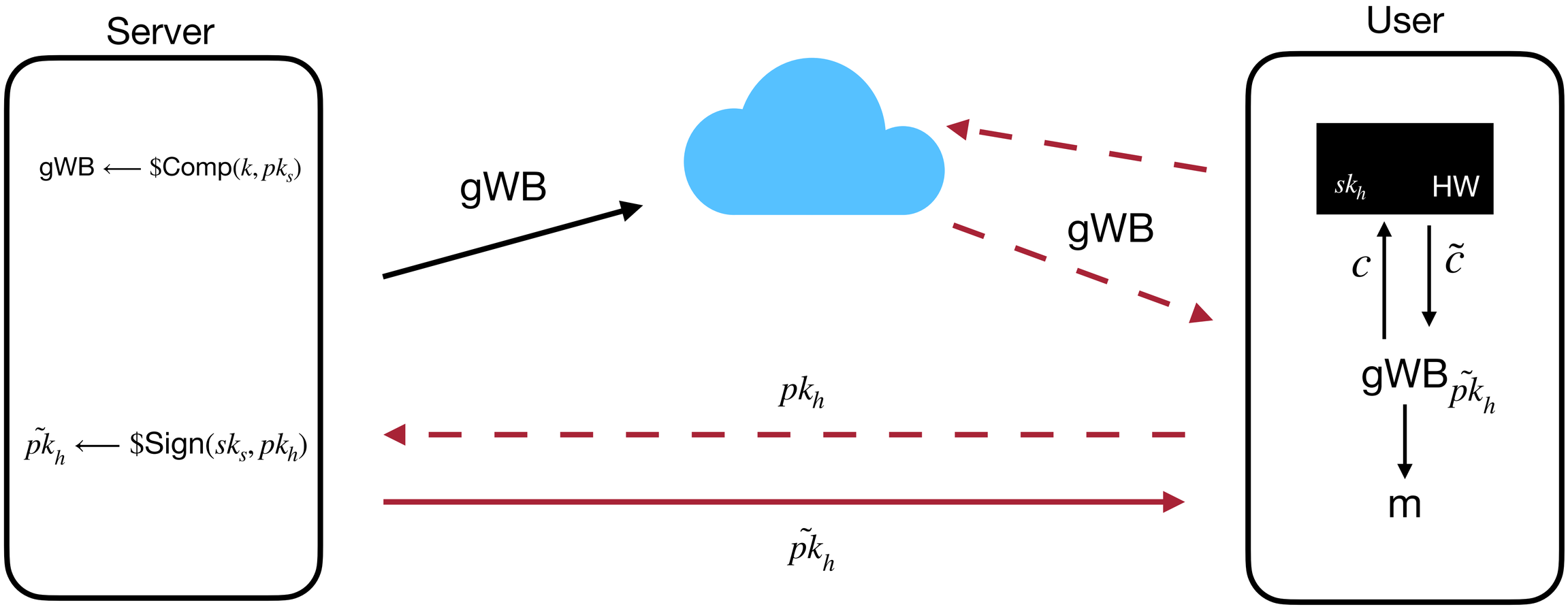
Strong global white-boxes



Strong global white-boxes



Strong global white-boxes



Strong global white-boxes

Design the white-box s.t. it always computers on encoded inputs

But additionally —> require the decoding key to be an *input* to the white-box

$\underline{\mathsf{Server}(d_h)}$

$\tilde{d_h} \leftarrow \mathsf{Encode}(e_s, d_h)$

return $\tilde{d_h}$

$\underline{\mathsf{HW}(c)}$

$\tilde{c} \leftarrow \mathsf{Encode}(e_h, c)$

return $\tilde{c}$

$\underline{\mathsf{WB}(\tilde{d_h}, \tilde{c})}$

$d_h \leftarrow \mathsf{Decode}(d_s, \tilde{d_h})$

$c \leftarrow \mathsf{Decode}(d_h, \tilde{c})$

$m \leftarrow \mathsf{Dec}(k, c)$

return m

The server encodes the personal encoding key

The white-box embeds a key for decoding the personal key

Each user can use their own key for encoding

Strong global white-boxes via message-recoverable signatures

Construct via puncturable primitives and indistinguishability obfuscation

—> construct *puncturable* MRS

—> obfuscate the circuit via iO

—> prove security via Sahai-Waters [5]

[5] Sahai and Waters: How to use indistinguishability obfuscation. STOC 2014

Strong global white-boxes vs *standard* ones

Positives for constructions of strong global white-boxes

Positives for constructions of standard global white-boxes

Strong global white-boxes vs *standard* ones

Positives for constructions of strong global white-boxes

Assumptions on the hw are lighter, no need of key loading

Positives for constructions of standard global white-boxes

Strong global white-boxes vs *standard* ones

Positives for constructions of strong global white-boxes

Assumptions on the hw are lighter, no need of key loading

No need to share secrets between provider and user

Positives for constructions of standard global white-boxes

Strong global white-boxes vs *standard* ones

Positives for constructions of strong global white-boxes

- Assumptions on the hw are lighter, no need of key loading
- No need to share secrets between provider and user
- Each user has their own encoding key

Positives for constructions of standard global white-boxes

Strong global white-boxes vs *standard* ones

Positives for constructions of strong global white-boxes

Assumptions on the hw are lighter, no need of key loading

No need to share secrets between provider and user

Each user has their own encoding key

Positives for constructions of standard global white-boxes

Allows us to obfuscate any program, including AES

Strong global white-boxes vs *standard* ones

Positives for constructions of strong global white-boxes

- Assumptions on the hw are lighter, no need of key loading
- No need to share secrets between provider and user
- Each user has their own encoding key

Positives for constructions of standard global white-boxes

- Allows us to obfuscate any program, including AES
- TBO can be achieved from more accepted assumptions than iO (LWE, garbled circuits)

Strong global white-boxes vs *standard* ones

Positives for constructions of strong global white-boxes

- Assumptions on the hw are lighter, no need of key loading
- No need to share secrets between provider and user
- Each user has their own encoding key

Positives for constructions of standard global white-boxes

- Allows us to obfuscate any program, including AES
- TBO can be achieved from more accepted assumptions than iO (LWE, garbled circuits)
- Faster, smaller white-boxes due to obfuscation means

Danke!

eprint: https://eprint.iacr.org/2021/767

Provider

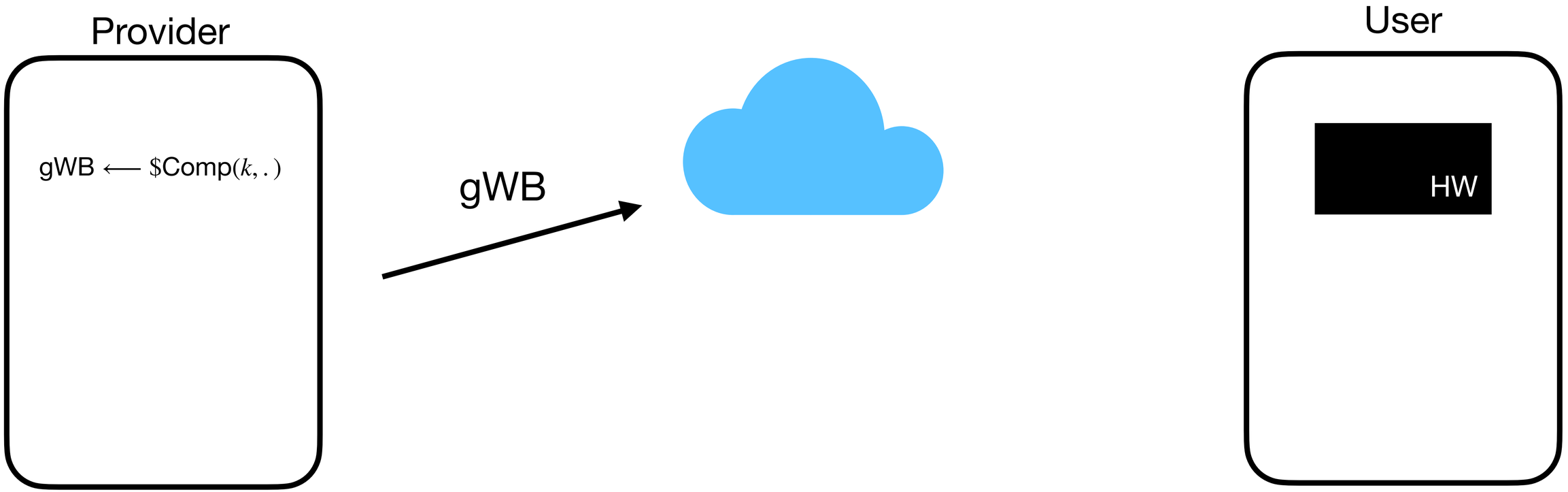
gWB ⟵ $Comp($k$, .)

User

HW

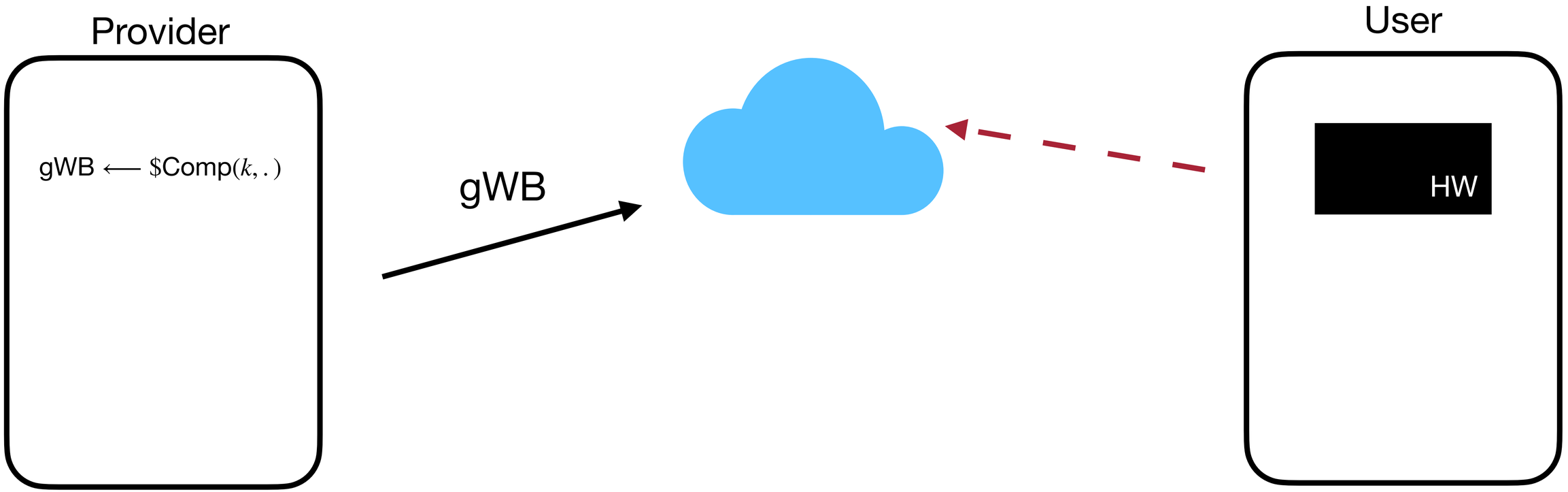
Danke!

eprint: https://eprint.iacr.org/2021/767



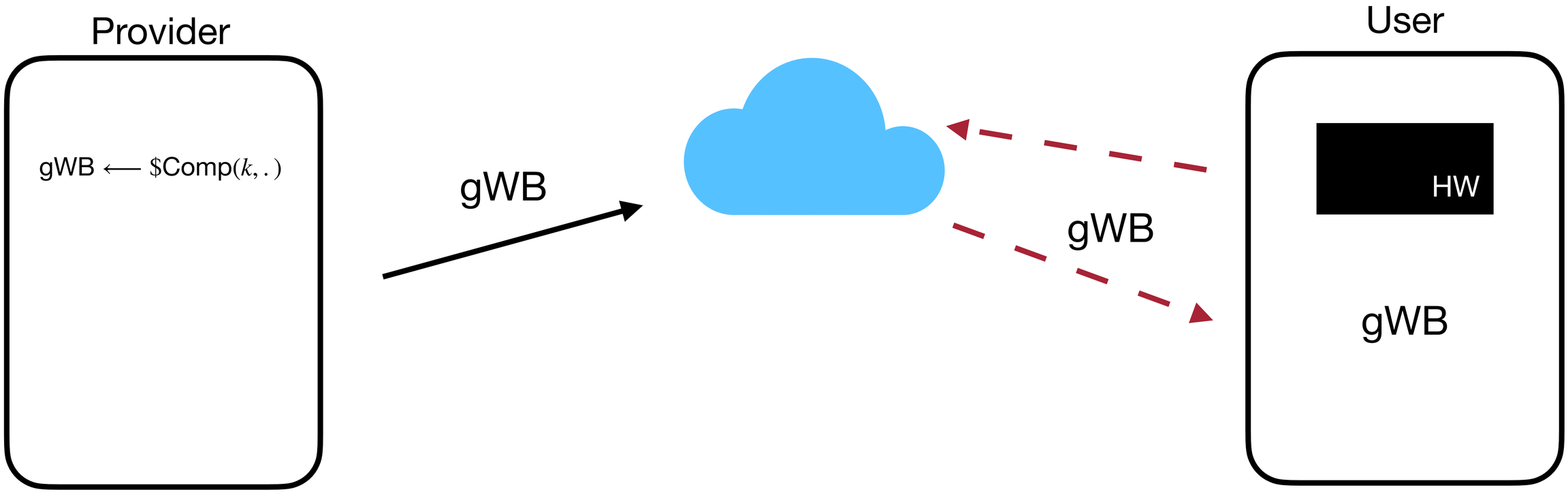
Danke!

eprint: https://eprint.iacr.org/2021/767



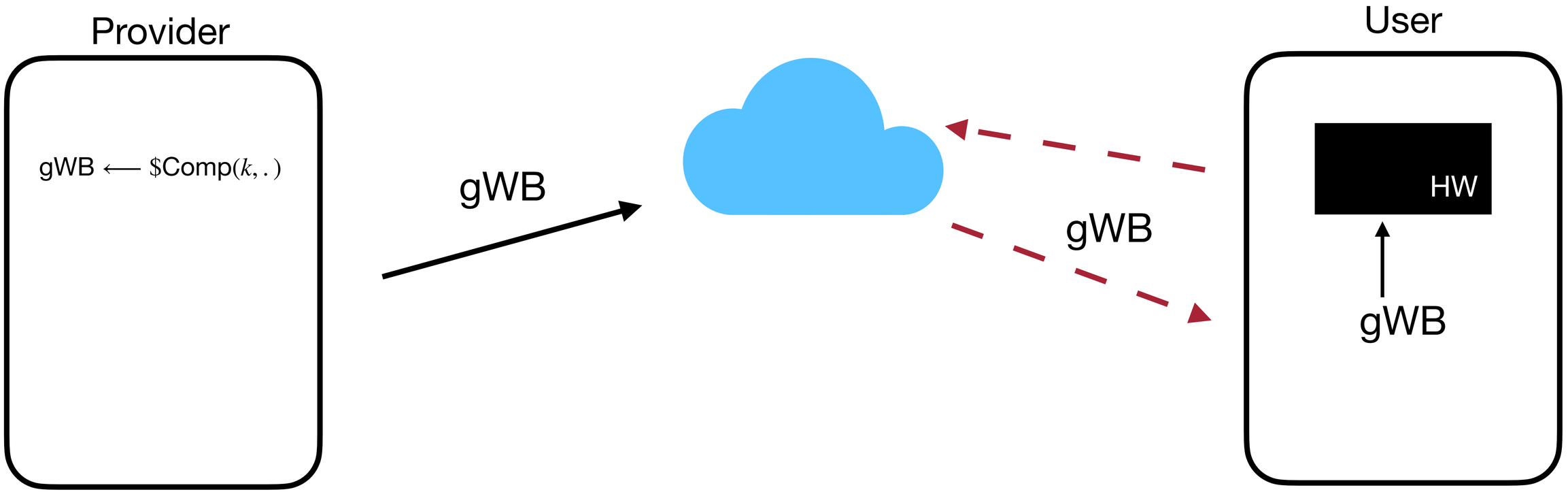
Danke!

eprint: https://eprint.iacr.org/2021/767



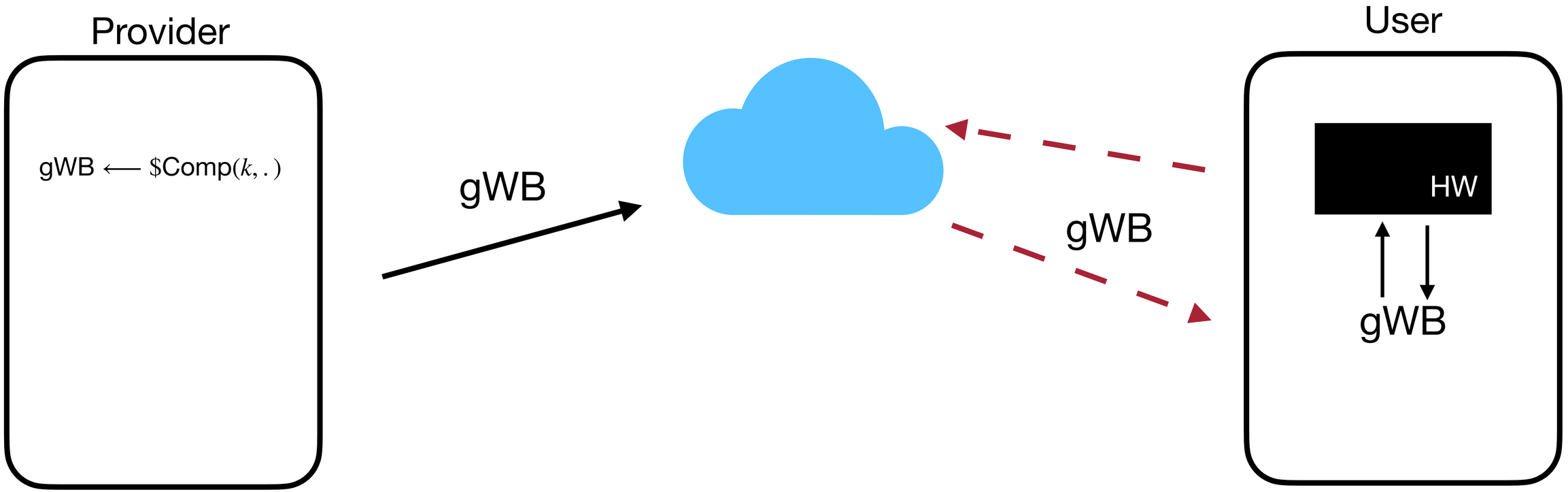
Danke!

eprint: https://eprint.iacr.org/2021/767



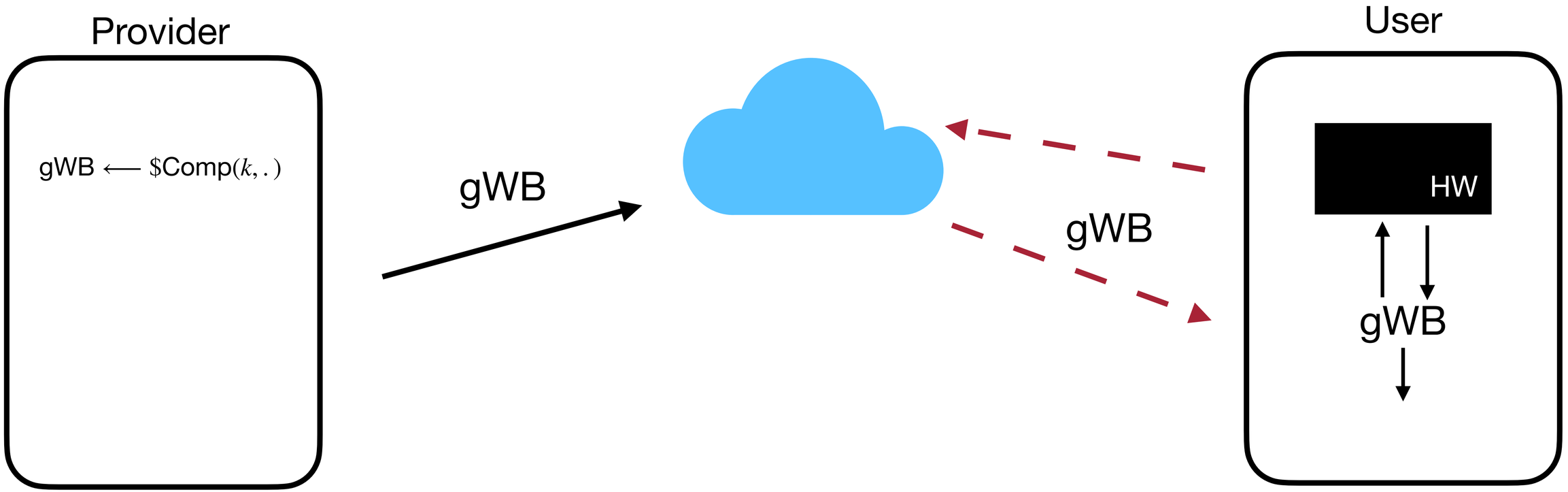
Danke!

eprint: https://eprint.iacr.org/2021/767



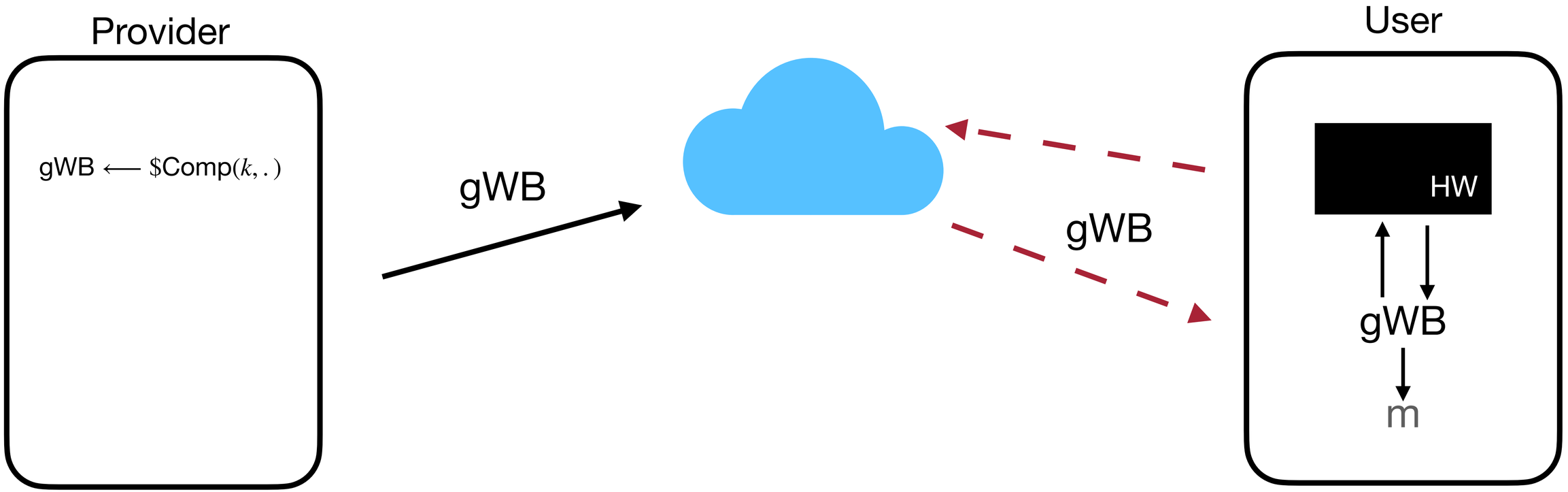
Danke!

eprint: https://eprint.iacr.org/2021/767



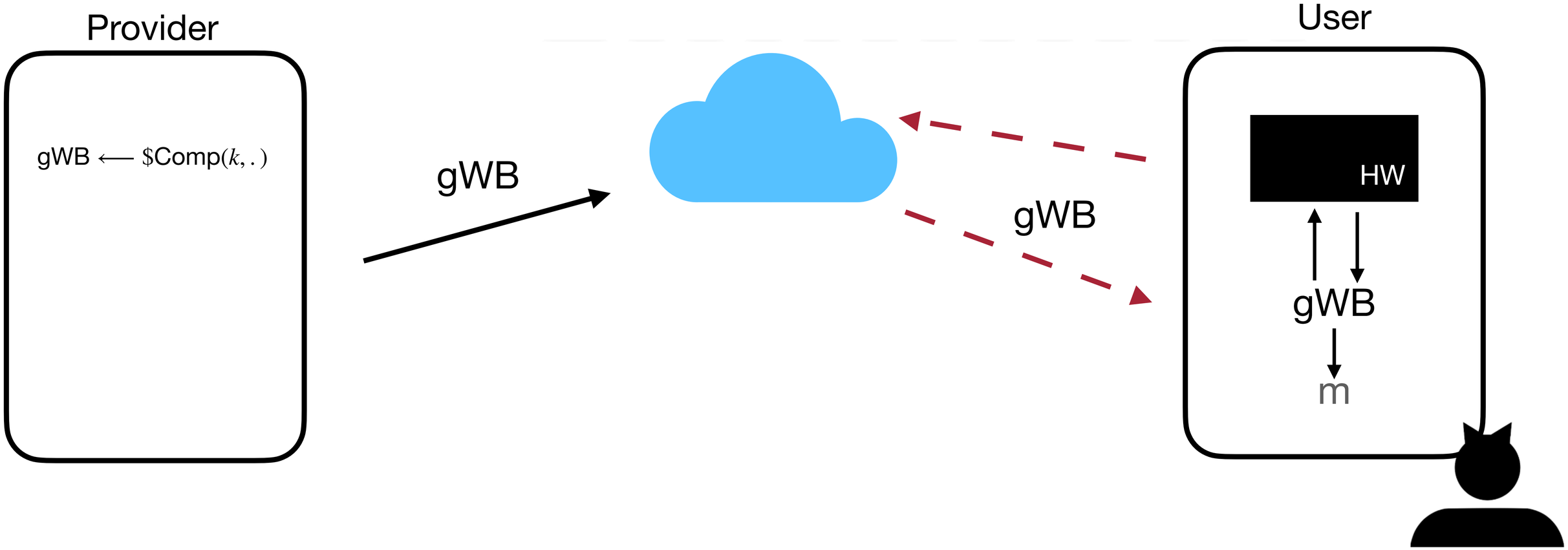
Danke!

eprint: https://eprint.iacr.org/2021/767



Danke!

eprint: https://eprint.iacr.org/2021/767



Summary

Summary

- Introduce the concept of *global white-boxes*

Summary

- Introduce the concept of *global white-boxes*
  - Aligning with commercial use of white-box crypto for protecting mobile payment applications

Summary

- Introduce the concept of *global white-boxes*
  - Aligning with commercial use of white-box crypto for protecting mobile payment applications
- Formally define the security of different flavours of global white-boxes

Summary

- Introduce the concept of *global white-boxes*
  - Aligning with commercial use of white-box crypto for protecting mobile payment applications
- Formally define the security of different flavours of global white-boxes
- Present provable secure constructions

Summary

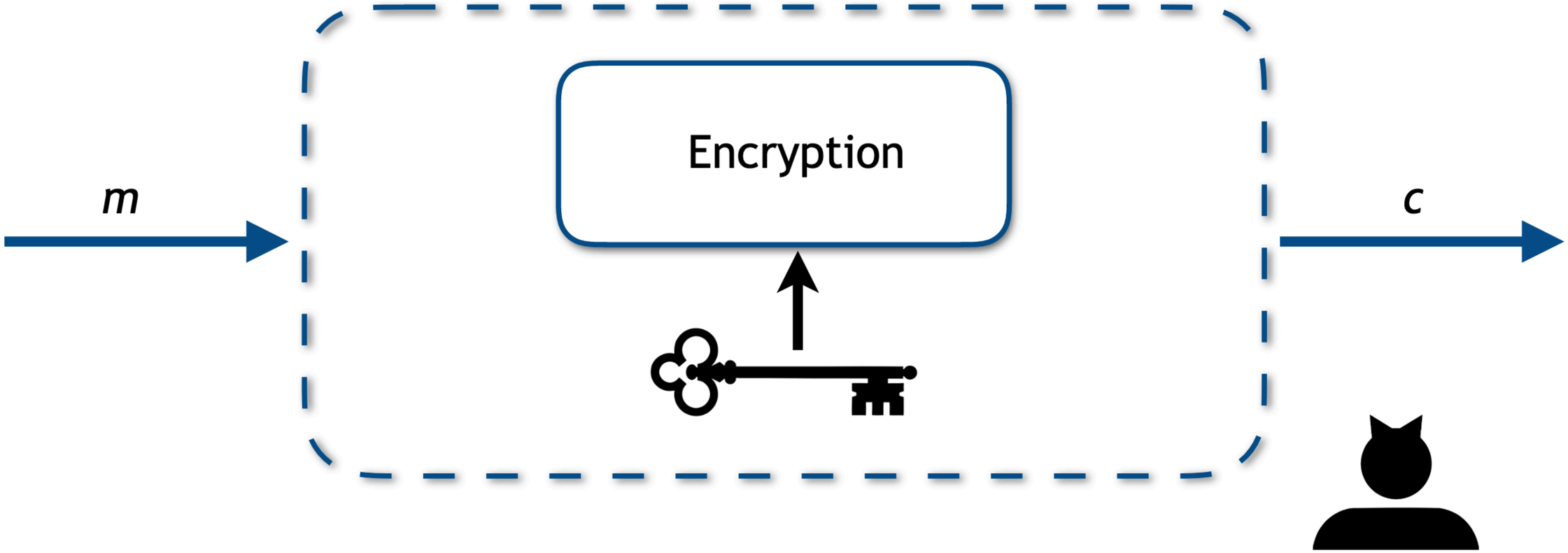
- Introduce the concept of *global white-boxes*
  - Aligning with commercial use of white-box crypto for protecting mobile payment applications
- Formally define the security of different flavours of global white-boxes
- Present provable secure constructions
  - Introduce puncturable message-recoverable signatures

Summary

- Introduce the concept of *global white-boxes*
  - Aligning with commercial use of white-box crypto for protecting mobile payment applications
- Formally define the security of different flavours of global white-boxes
- Present provable secure constructions
  - Introduce puncturable message-recoverable signatures
  - New feasibility results for white-box crypto via LWE. No need for iO for constructing secure white-box crypto!

Summary

- Introduce the concept of *global white-boxes*
  - Aligning with commercial use of white-box crypto for protecting mobile payment applications
- Formally define the security of different flavours of global white-boxes
- Present provable secure constructions
  - Introduce puncturable message-recoverable signatures
  - New feasibility results for white-box crypto via LWE. No need for iO for constructing secure white-box crypto!
- More details and further results on the extended version of our paper: eprint/2021



- Definitional studies and feasibility results for white-box cryptography
  - Construct white-box programs from established cryptographic primitives —> prove their security in the presence of a white-box adversary

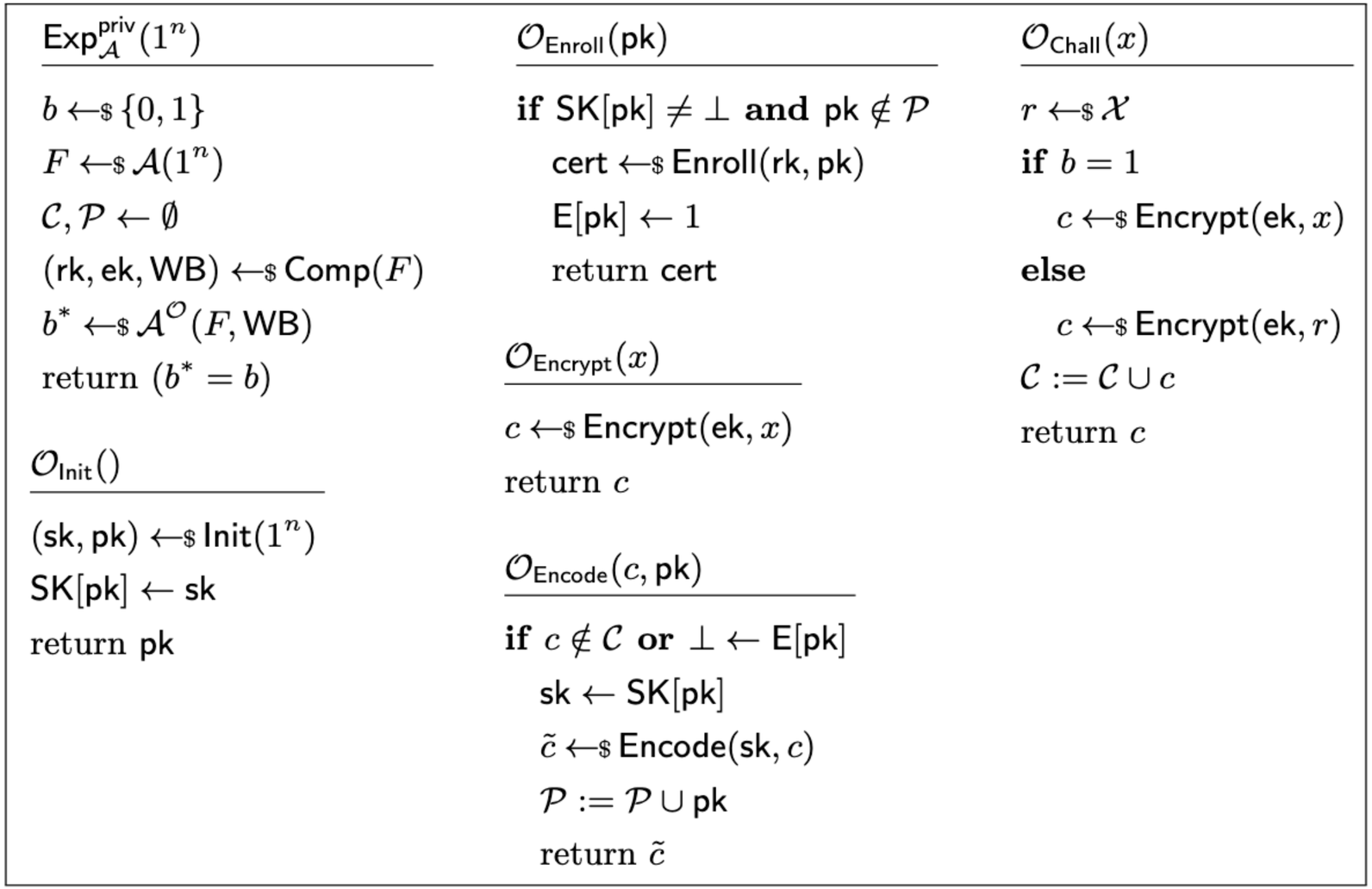


Fig. 2: Privacy $\mathsf{Exp}^{\mathsf{priv}}_{\mathcal{A}}(1^n)$ security game.

$\Gamma[\mathsf{pk}_2, \mathsf{ek}, F](\mathsf{cert}, \tilde{c}, y)$

$\mathsf{cert}||\mathsf{pk}_1 \leftarrow \mathsf{cert}$
$\tilde{c}||c \leftarrow \tilde{c}$
if $G(\mathsf{pk}_1) \neq G(\mathsf{Rcvr}(\mathsf{pk}_2, \mathsf{cert}))$
  return $\perp$
else if $G(c) \neq G(\mathsf{Rcvr}(\mathsf{pk}_1, \tilde{c}))$
  return $\perp$
else
  $t, r \leftarrow c$
  $v \leftarrow \mathtt{PPRF}(\mathsf{ek}, r)$
  $x \leftarrow t \oplus v$
  return $F(x, y)$
else *return* $\perp$

$\mathsf{Comp}(F)$

1 : $\mathsf{sk}_2, \mathsf{pk}_2 \leftarrow\!\!\$\ \mathsf{MRS.KGen}(1^\lambda)$
2 : $\mathsf{rk} \leftarrow \mathsf{sk}_2$
3 : $\mathsf{ek} \leftarrow\!\!\$\ \{0,1\}^\lambda$
4 : $\mathsf{WB} \leftarrow\!\!\$\ \mathtt{iO}(\Gamma[\mathsf{pk}_2, \mathsf{ek}, F])$
5 : *return* $\mathsf{rk}, \mathsf{ek}, \mathsf{WB}$

$\mathsf{Encrypt}(\mathsf{ek}, (x))$

1 : $r \leftarrow\!\!\$\ \{0,1\}^n$
2 : $v \leftarrow \mathtt{PPRF}(\mathsf{ek}, r)$
3 : $t \leftarrow x \oplus v$
4 : $c \leftarrow (t, r)$
5 : *return* c

$\mathsf{Enroll}(\mathsf{rk}, (\mathsf{pk}_1))$

1 : $\mathsf{cert} \leftarrow\!\!\$\ \mathsf{MRS.Sig}(\mathsf{rk}, \mathsf{pk}_1)$
2 : $\mathsf{cert} \leftarrow \mathsf{cert}||\mathsf{pk}_1$
3 : *return* cert

$\mathsf{Encode}(\mathsf{sk}_1, c))$

1 : $\tilde{c} \leftarrow\!\!\$\ \mathsf{MRS.Sig}(\mathsf{sk}_1, c)$
2 : $\tilde{c} \leftarrow \tilde{c}||c$
3 : *return* $\tilde{c}$

$\mathsf{Init}(1^\lambda)$

1 : $\mathsf{sk}, \mathsf{pk} \leftarrow\!\!\$\ \mathsf{MRS.KGen}(1^\lambda)$
2 : *return* $(\mathsf{sk}_1, \mathsf{pk}_1)$

Theorem 2. *Let* iO *be a an indistinguishability obfuscator and let* MRS *be a puncturable MRS scheme. Let PRG be a pseudorandom generator, and* PPRF *a secure puncturable pseudorandom function. Then Construction 5 is a privacy-secure* sGW *scheme.*