Intermundium-DL:

Assessing the Resilience of Current Schemes to Discrete-Log-Computation Attacks on Public Parameters

Mihir Bellare (UCSD)
Doreen Riepel (CISPA)
Laura Shea (UCSD)

PKC 2025

eprint 2025 / 663

INTERMUNDIUM-DL

A world in which

computing DLs in currently standardized groups is

Possible

but COSTLY

The adversary can compute a few discrete logarithms $\log_g(\,\cdot\,)$, but not many.

INTERMUNDIUM-DL

A world in which

computing DLs in currently standardized groups is

Possible

but COSTLY

Why?

Early Quantum Computers

The adversary can compute a few discrete logarithms $\log_g(\,\cdot\,)$, but not many.

INTERMUNDIUM-DL

A world in which

computing DLs in currently standardized groups is

Possible

but COSTLY

The adversary can compute a few discrete logarithms $\log_g(\,\cdot\,)$, but not many.

Why?

Early Quantum Computers

INTERMUNDIUM-DL

A world in which

computing DLs in currently standardized groups is

Possible

but COSTLY

The adversary can compute a few discrete logarithms $\log_g(\,\cdot\,)$, but not many.

How might an adversary best exploit this capability?

INTERMUNDIUM-DL

A world in which

computing DLs in currently standardized groups is

Possible

but COSTLY

The adversary can compute a few discrete logarithms $\log_g(\cdot)$, but not many.

How might an adversary best exploit this capability?

Our Answer: Attack schemes whose public parameters $\pi = (h_1, \ldots, h_w)$ consist of a few group elements:

- Compute $\log_g(h_1), \ldots, \log_g(h_w)$
- Hope thereby to easily compromise security of MANY users

INTERMUNDIUM-DL

A world in which
computing DLs in currently standardized groups is

Possible
but COSTLY

The adversary can compute a few discrete logarithms $\log_g(\cdot)$, but not many.

How might an adversary best exploit this capability?

Our Answer: Attack schemes whose public parameters $\pi = (h_1, \ldots, h_w)$ consist of a few group elements:

- Compute $\log_g(h_1), \ldots, \log_g(h_w)$
- Hope thereby to easily compromise security of MANY users

We accordingly investigate the security of current schemes in the setting where

the adversary knows $\log_g(h_1), \ldots, \log_g(h_w)$

The proofs typically assume that the adversary does NOT know these discrete logarithms.

So we might expect there to be attacks violating security in our setting.

INTERMUNDIUM-DL

A world in which
computing DLs in currently standardized groups is

Possible
but COSTLY

The adversary can compute a few discrete logarithms $\log_g(\,\cdot\,)$, but not many.

How might an adversary best exploit this capability?

Our Answer: Attack schemes whose public parameters $\pi = (h_1, \ldots, h_w)$ consist of a few group elements:

- Compute $\log_g(h_1), \ldots, \log_g(h_w)$
- Hope thereby to easily compromise security of MANY users

We accordingly investigate the security of current schemes in the setting where

the adversary knows $\log_g(h_1), \ldots, \log_g(h_w)$

The proofs typically assume that the adversary does NOT know these discrete logarithms.

So we might expect there to be attacks violating security in our setting.

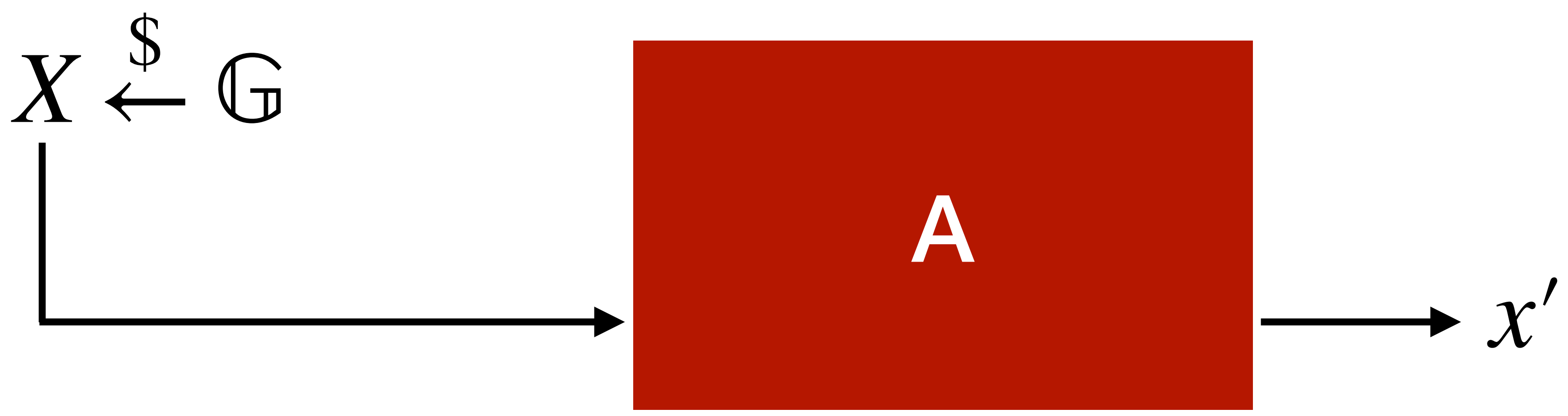
However we find surprising variations in security across schemes:

- Some fully retain security
- Some retain partial but meaningful security
- Some do break totally

THE DL (DISCRETE LOG) PROBLEM

Game $\mathrm{DL}_{\mathbb{G},p,g}$

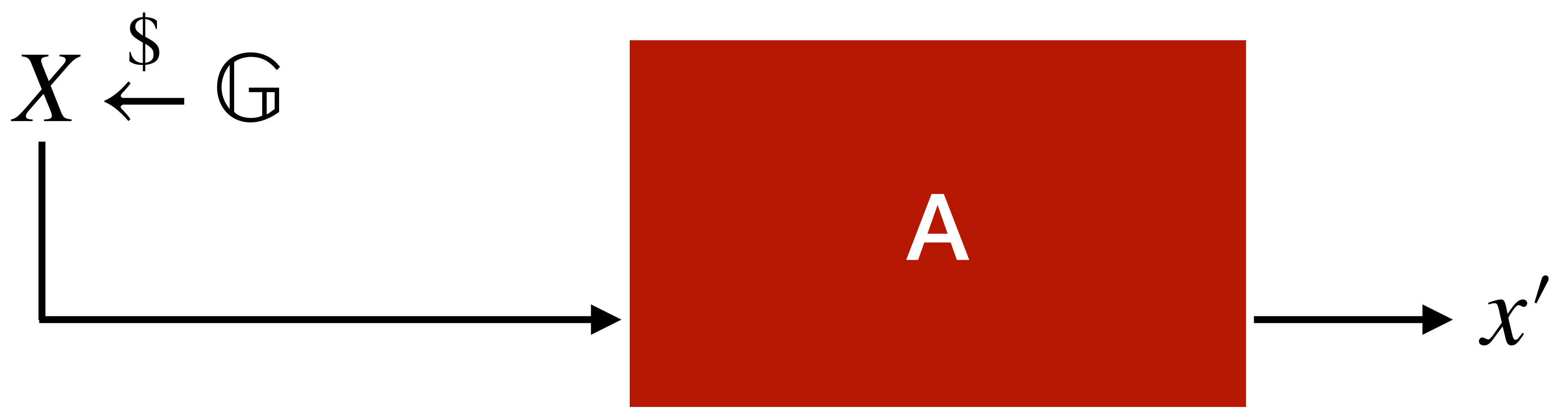
Group $\mathbb{G}$, of order p, with generator g



Adversary A wins game DL if $g^{x'} = X$.

THE DL (DISCRETE LOG) PROBLEM

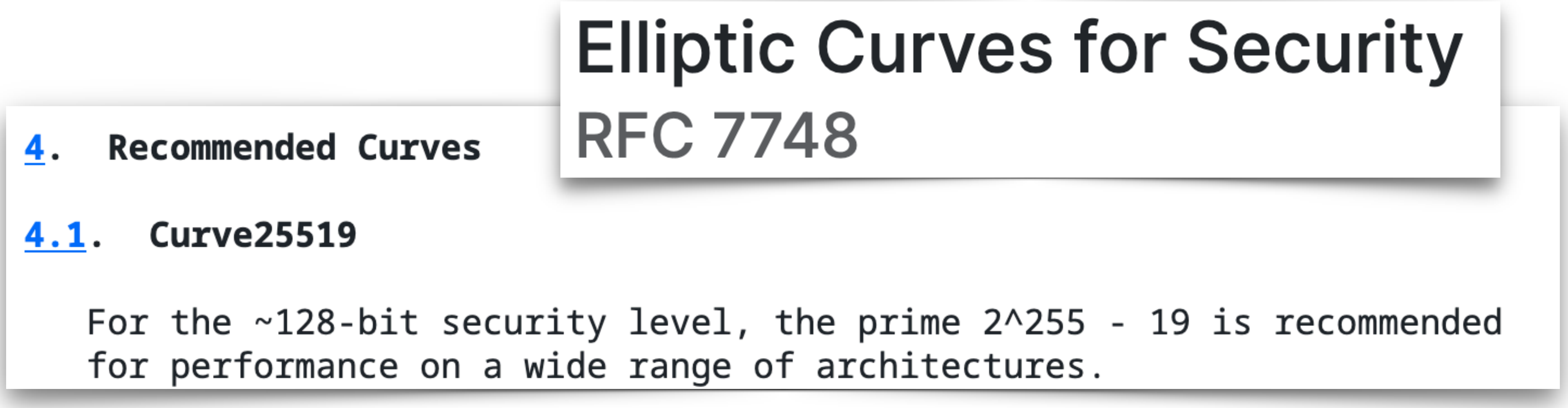
Game $\mathrm{DL}_{\mathbb{G},p,g}$ Group $\mathbb{G}$, of order p, with generator g



Adversary A wins game DL if $g^{x'} = X$.

What is $\mathbb{G}$?

Often, an elliptic-curve group of order $p \approx 2^{256}$.



Elliptic Curves for Security
RFC 7748

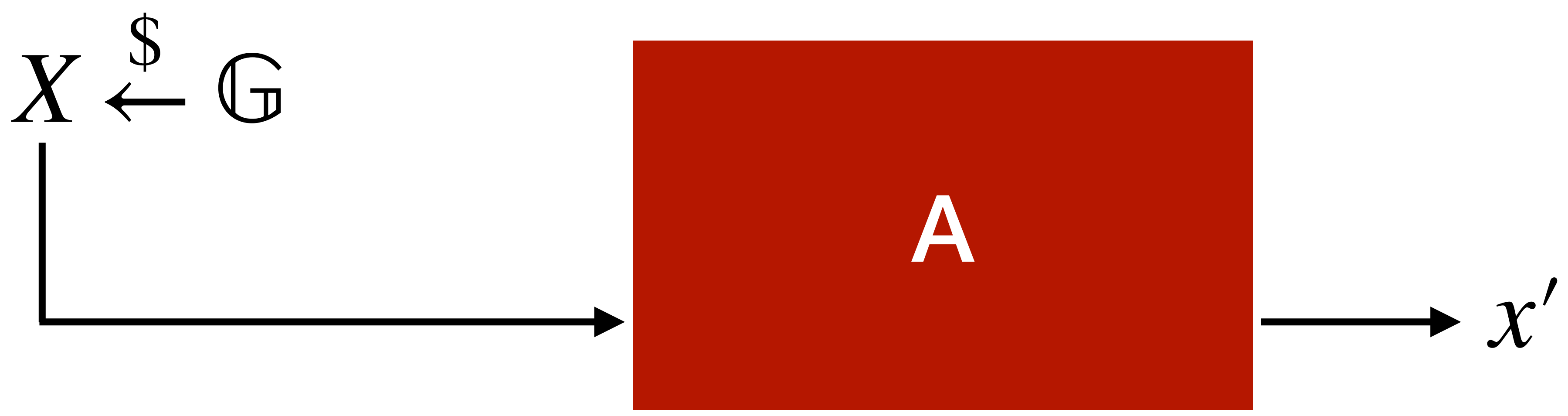
4. Recommended Curves

4.1. Curve25519

For the ~128-bit security level, the prime 2^255 - 19 is recommended for performance on a wide range of architectures.

THE DL (DISCRETE LOG) PROBLEM

Game $\mathrm{DL}_{\mathbb{G},p,g}$ | Group $\mathbb{G}$, of order p, with generator g



Adversary A wins game DL if $g^{x'} = X$.

What is $\mathbb{G}$?

Often, an elliptic-curve group of order $p \approx 2^{256}$.

Elliptic Curves for Security
RFC 7748

4. **Recommended Curves**

4.1. **Curve25519**

For the ~128-bit security level, the prime 2^255 - 19 is recommended for performance on a wide range of architectures.

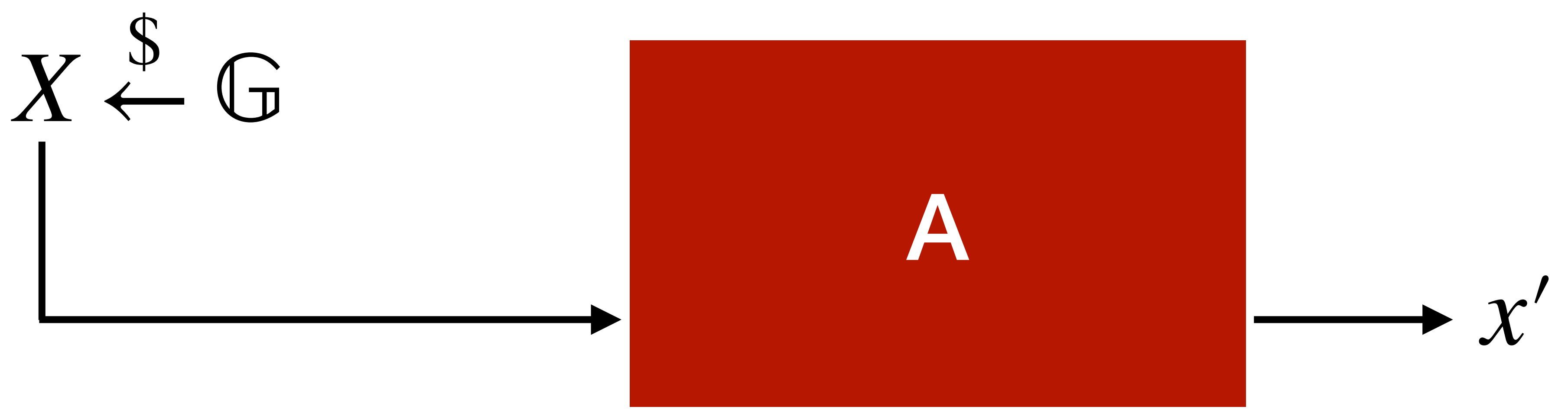
Q: How hard is it to win the DL game?

A: State-of-the-art cryptanalysis :

- For ANY group $\mathbb{G}$, there is a DL adversary of time $2^{(\log p)/2}$ [Pollard]
- With preprocessing, this can become $2^{(\log p)/3}$ [BL13, …]
- For SOME (non-EC) groups, there is a DL adversary of time about $2^{(\log p)^{1/3}}$ (NFS algorithm)

THE DL (DISCRETE LOG) PROBLEM

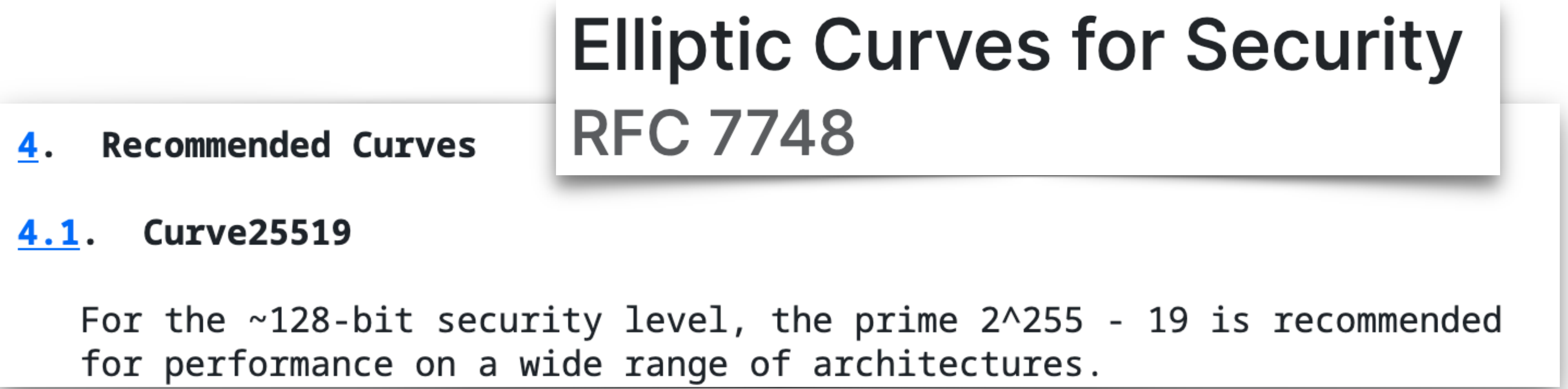
Game $\mathrm{DL}_{\mathbb{G},p,g}$ | Group $\mathbb{G}$, of order p, with generator g



Adversary A wins game DL if $g^{x'} = X$.

What is $\mathbb{G}$?

Often, an elliptic-curve group of order $p \approx 2^{256}$.


Elliptic Curves for Security
RFC 7748

4. Recommended Curves

4.1. Curve25519

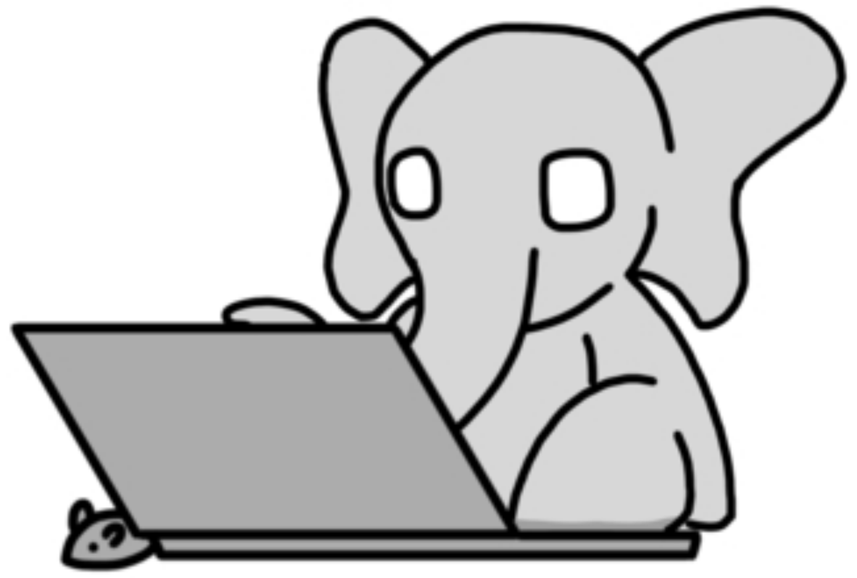
For the ~128-bit security level, the prime 2^255 - 19 is recommended for performance on a wide range of architectures.

Q: How hard is it to win the DL game?

A: State-of-the-art cryptanalysis :

- For ANY group $\mathbb{G}$, there is a DL adversary of time $2^{(\log p)/2}$ [Pollard]
- With preprocessing, this can become $2^{(\log p)/3}$ [BL13, …]
- For SOME (non-EC) groups, there is a DL adversary of time about $2^{(\log p)^{1/3}}$ (NFS algorithm)

There is a QUANTUM A that runs in time $\mathrm{poly}(\log p)$ [Shor]



XXX-krone QUESTION:
Will quantum computers run Shor’s algorithm on 256-bit elliptic curves, by year 20##?

(Choose your XXX, target year, and you can place a bet on PQC-forum.)

XXX-krone QUESTION:
Will quantum computers run Shor's algorithm on 256-bit elliptic curves, by year 20##?

(Choose your XXX, target year, and you can place a bet on PQC-forum.)

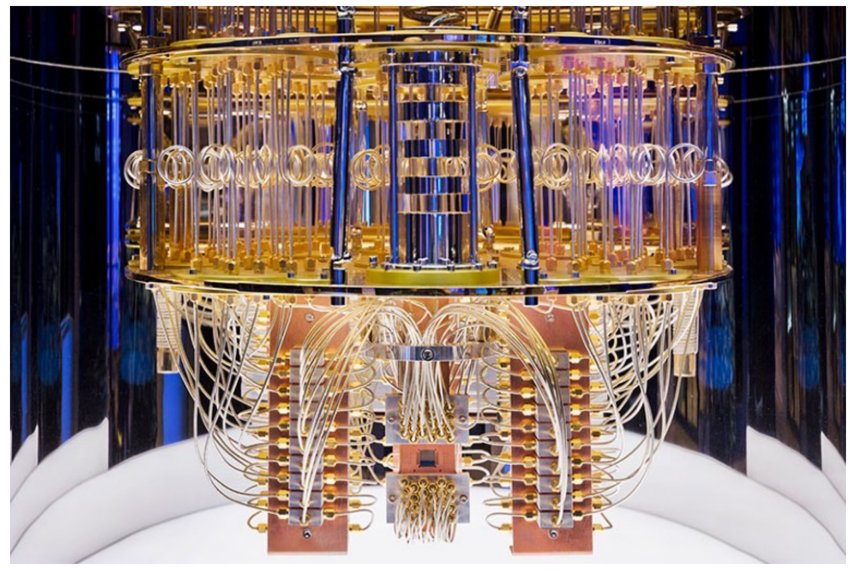
YES

WIRED

The Quantum Apocalypse Is Coming. Be Very Afraid

What happens when quantum computers can finally crack encryption and break into the world's best-kept secrets? It's called Q-Day—the worst...

1 month ago



NO

IEEE Spectrum

Quantum Computing's Hard, Cold Reality Check

Hype is everywhere, skeptics say, and practical applications are still far away.

Dec 22, 2023

Nature

Don't believe the hype — quantum tech can't yet solve real-world problems

Investors and the public should know what quantum devices can and, more importantly, can't do.

1 week ago

XXX-krone QUESTION:
Will quantum computers run Shor's algorithm on 256-bit elliptic curves, by year 20##?

(Choose your XXX, target year, and you can place a bet on PQC-forum.)

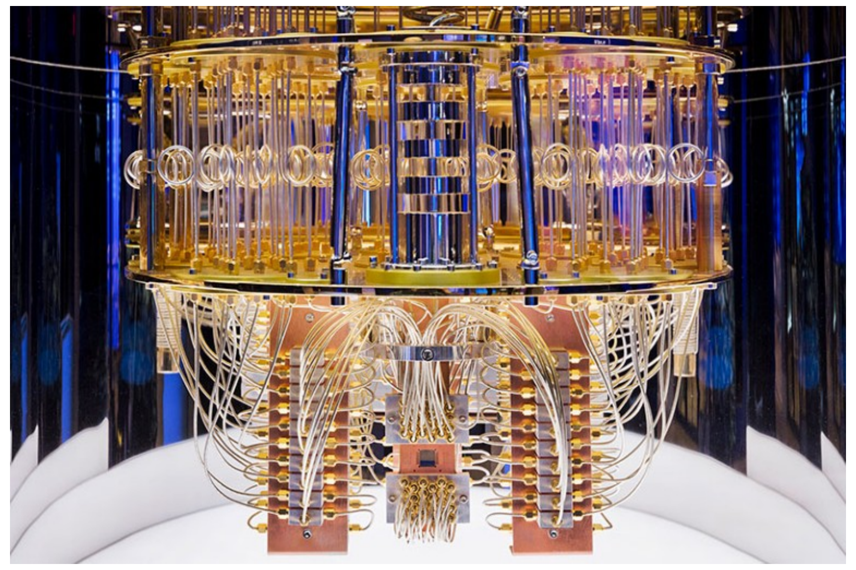
YES

WIRED

The Quantum Apocalypse Is Coming. Be Very Afraid

What happens when quantum computers can finally crack encryption and break into the world's best-kept secrets? It's called Q-Day—the worst...

1 month ago



NO

IEEE Spectrum

Quantum Computing's Hard, Cold Reality Check

Hype is everywhere, skeptics say, and practical applications are still far away.

Dec 22, 2023

Nature

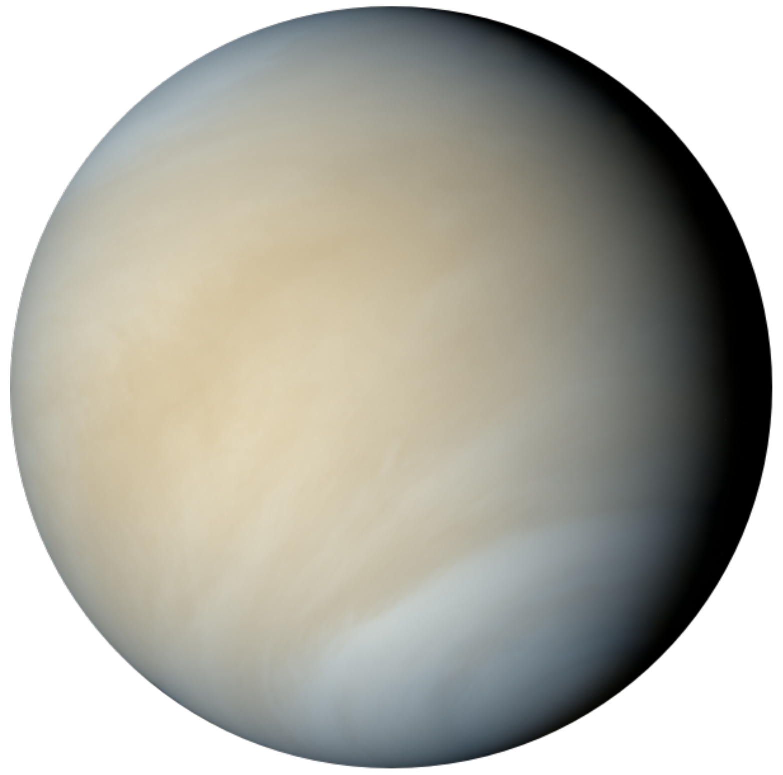
Don't believe the hype — quantum tech can't yet solve real-world problems

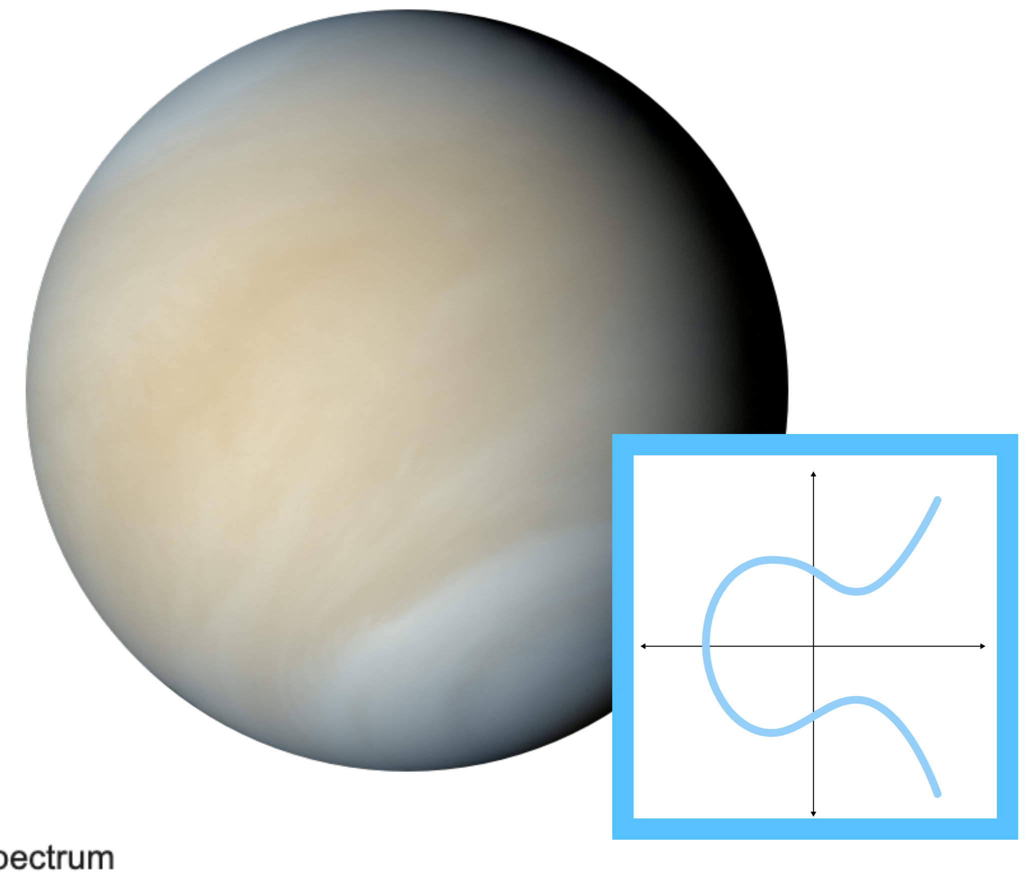
Investors and the public should know what quantum devices can and, more importantly, can't do.

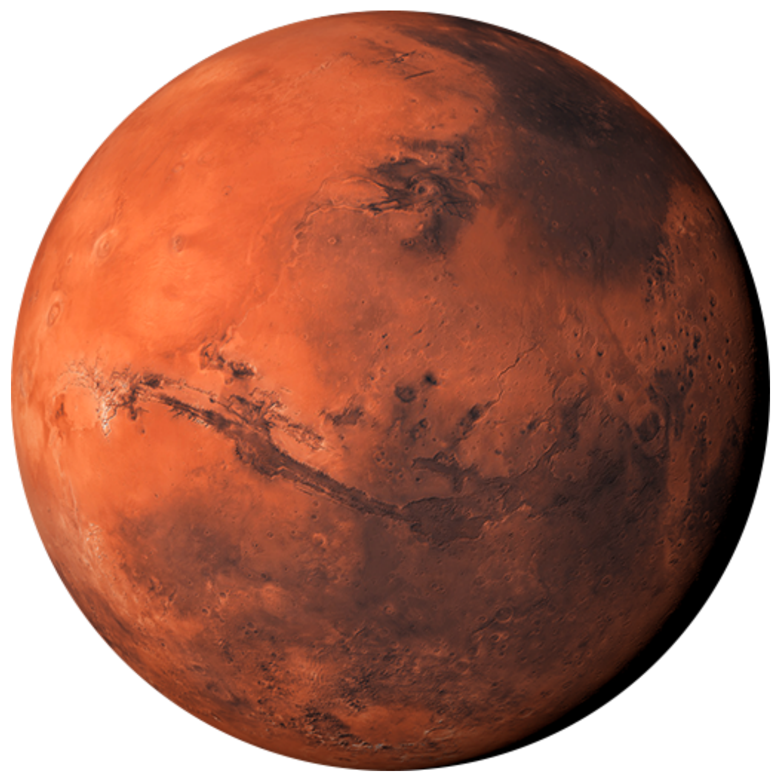
1 week ago

INTERMUNDIUM-DL

We're interested in a less binary answer.







IEEE Spectrum

Quantum Computing’s Hard, Cold Reality Check

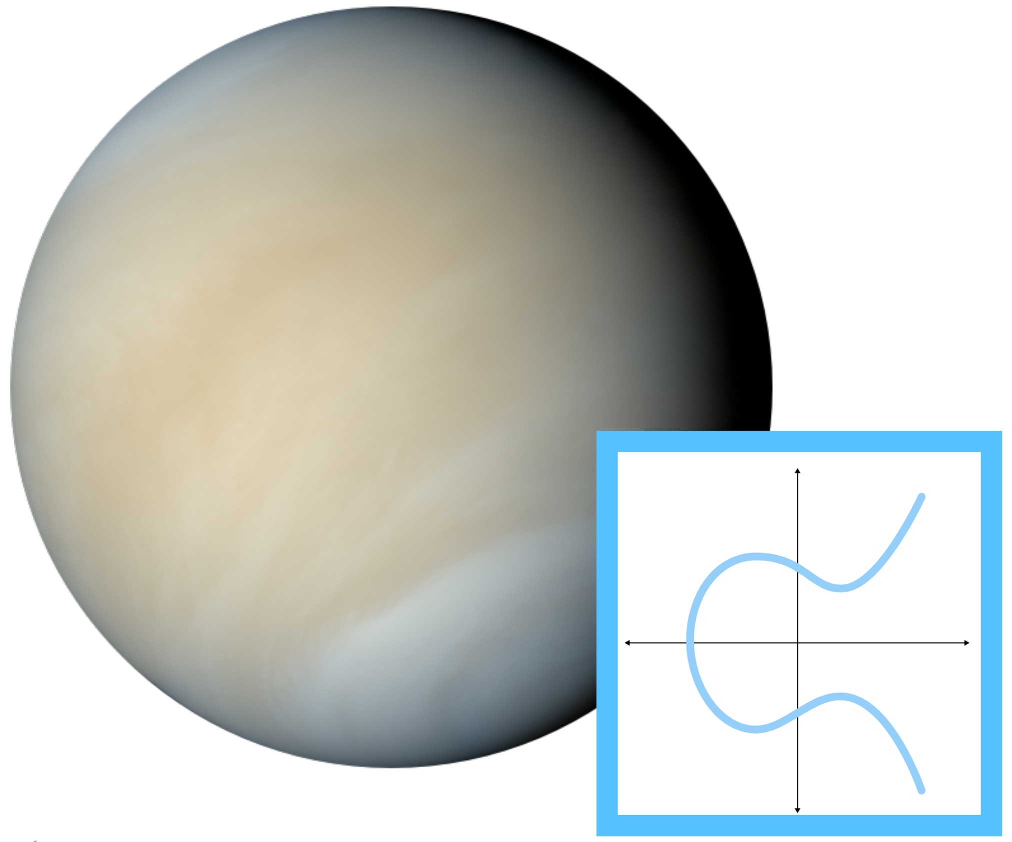
Hype is everywhere, skeptics say, and practical applications are still far away.

Dec 22, 2023

DL is hard

<u>Optimistic view:</u>

QCs running Shor’s are never built.
Classical cryptanalysis never improves.
DL-based crypto is totally safe.



IEEE Spectrum

Quantum Computing’s Hard, Cold Reality Check

Hype is everywhere, skeptics say, and practical applications are still far away.

Dec 22, 2023

WIRED

The Quantum Apocalypse Is Coming. Be Very Afraid

What happens when quantum computers can finally crack encryption and break into the world's best-kept secrets? It's called Q-Day—the worst...

1 month ago

DL is hard

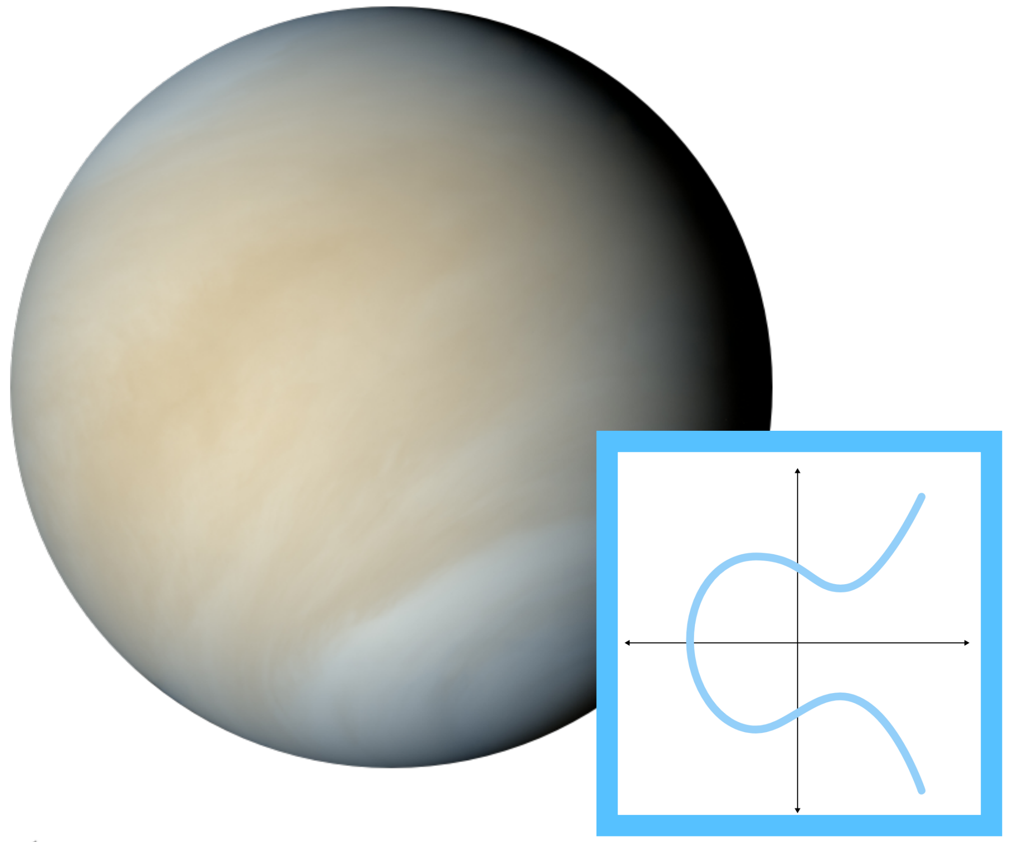
DL is easy

Optimistic view:

QCs running Shor’s are never built.
Classical cryptanalysis never improves.
DL-based crypto is totally safe.

Pessimistic view:

Blindingly fast QCs are right around the corner.
256-bit DL is easy and all DL-based crypto is forfeit.

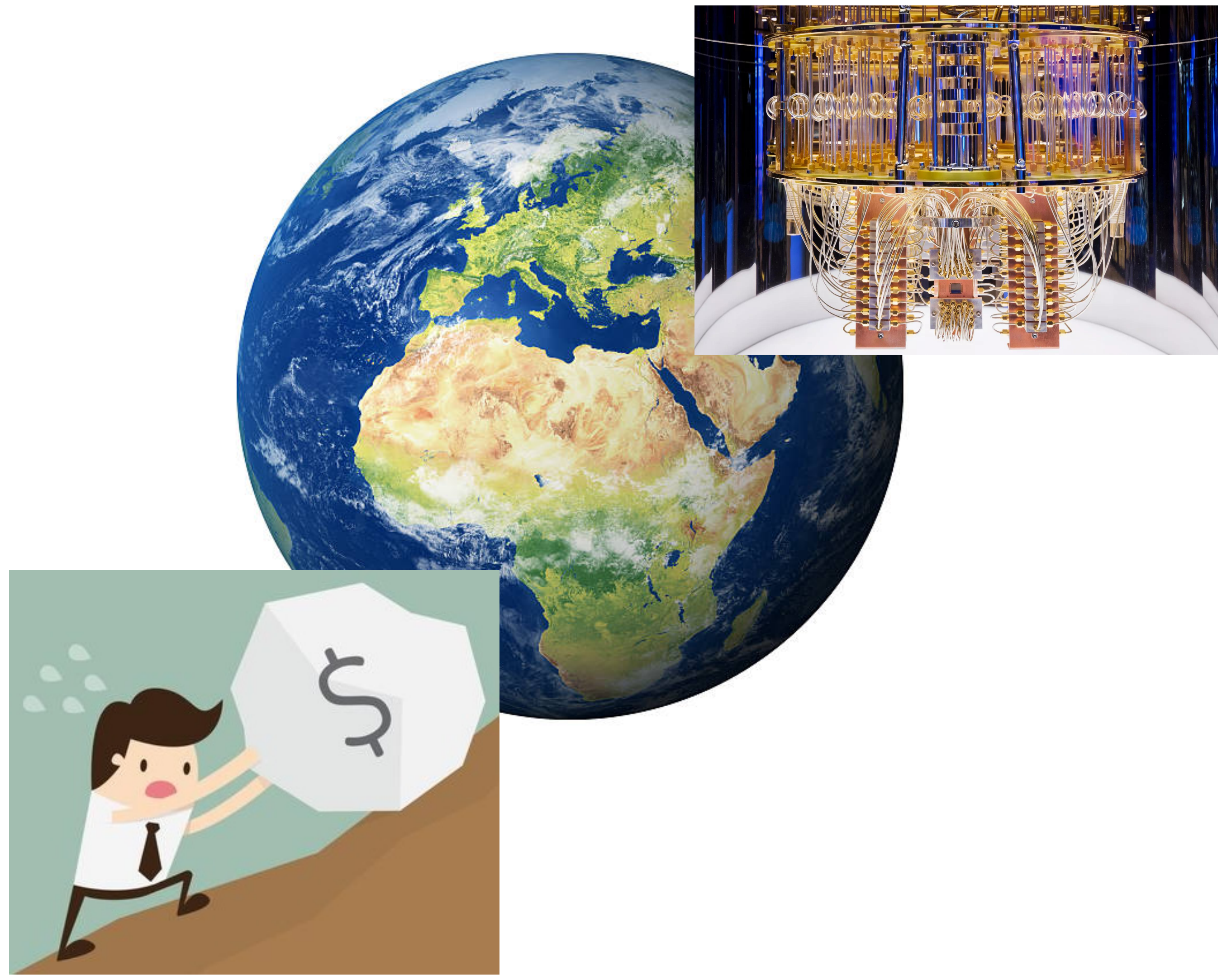


IEEE Spectrum

Quantum Computing’s Hard, Cold Reality Check

Hype is everywhere, skeptics say, and practical applications are still far away.

Dec 22, 2023



WIRED

The Quantum Apocalypse Is Coming. Be Very Afraid

What happens when quantum computers can finally crack encryption and break into the world's best-kept secrets? It's called Q-Day—the worst...

1 month ago

DL is hard

DL is feasible but expensive

DL is easy

Optimistic view:

QCs running Shor’s are never built.
Classical cryptanalysis never improves.
DL-based crypto is totally safe.

Intermundium-DL view:

QCs could run Shor, but at great cost. A rich adversary could compute a few DLs. But per-user DL computation is out of reach.

Pessimistic view:

Blindingly fast QCs are right around the corner.
256-bit DL is easy and all DL-based crypto is forfeit.

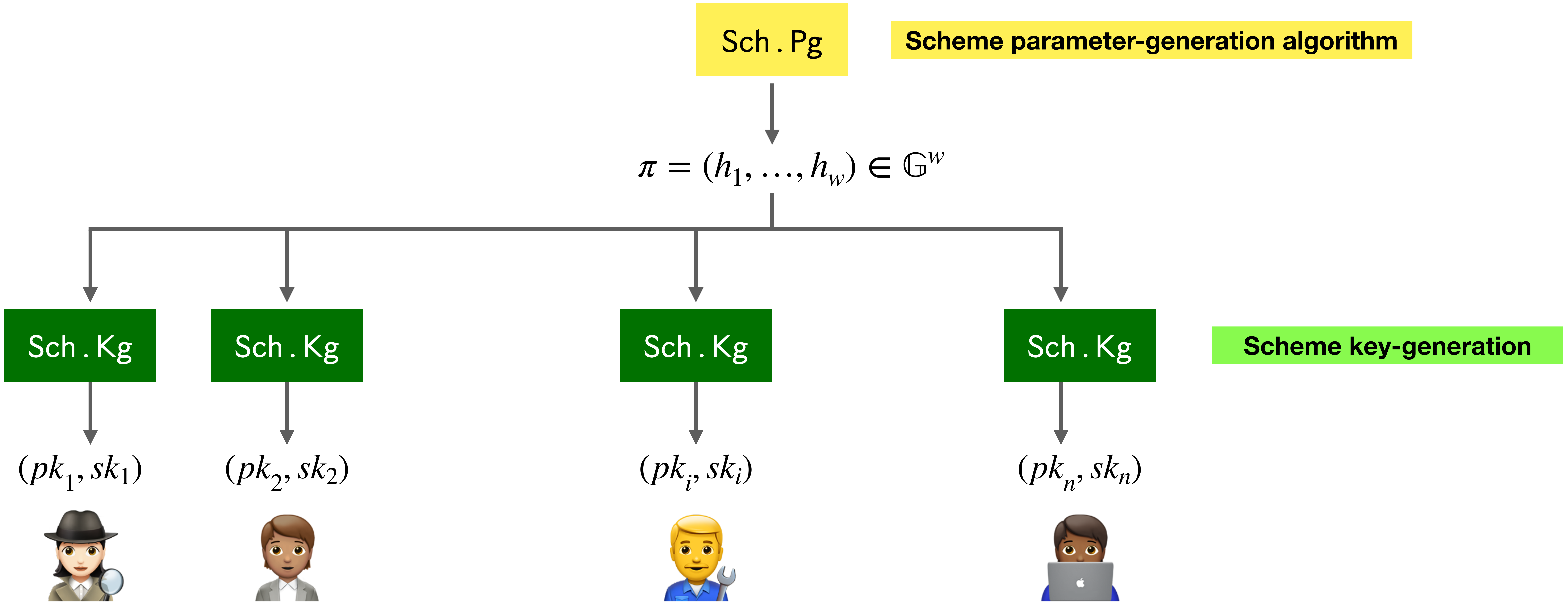
- [x] Setting the scene
- [ ] **Definitions: How to formalize security in Intermundium-DL?**

RESULTS

  - [ ] Signatures
  - [ ] Public-key encryption
  - [ ] Password-authenticated key exchange

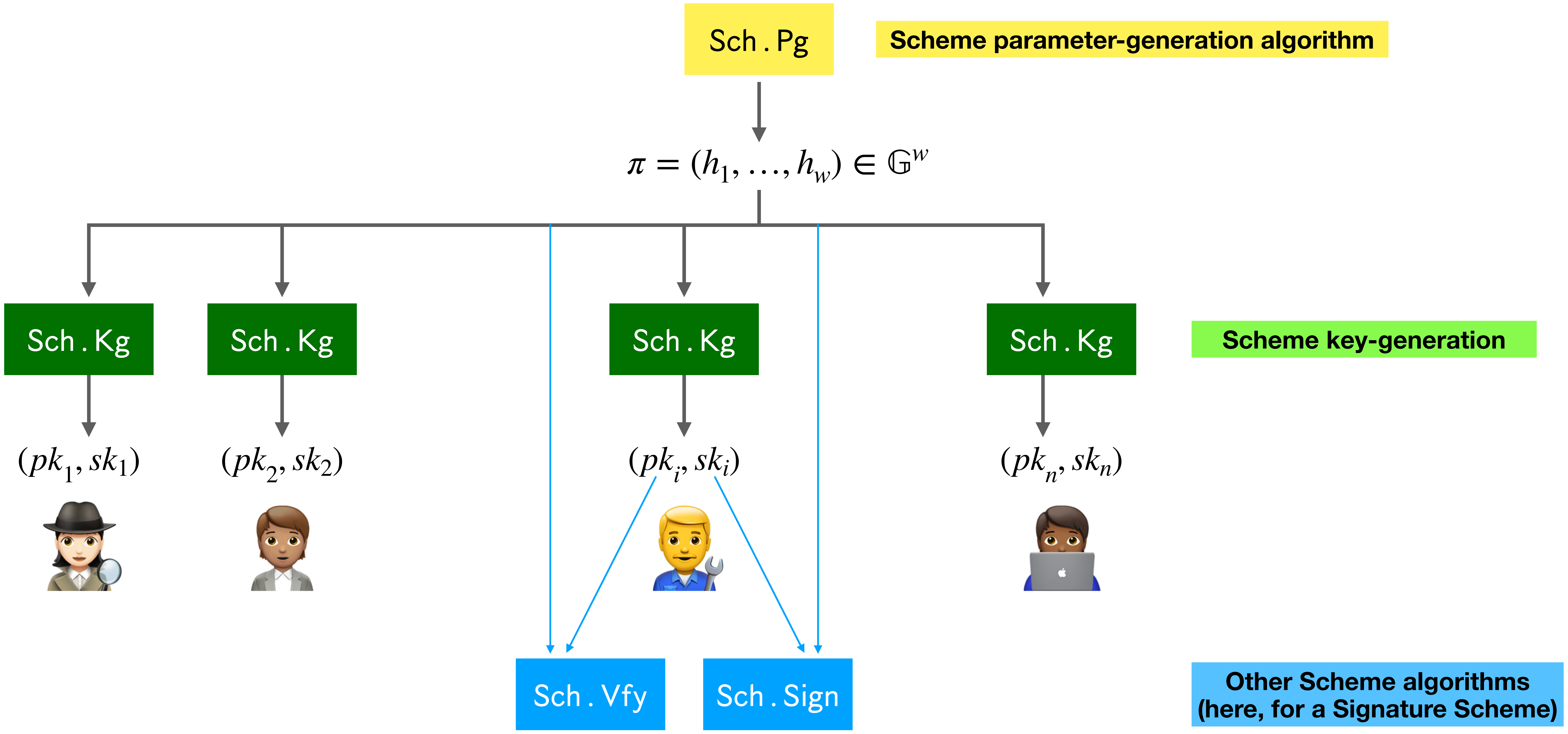
Group-Element-Parameter (GEP) Schemes

Let Sch be a width-w GEP scheme, over a fixed group described by $(\mathbb{G}, p, g)$.



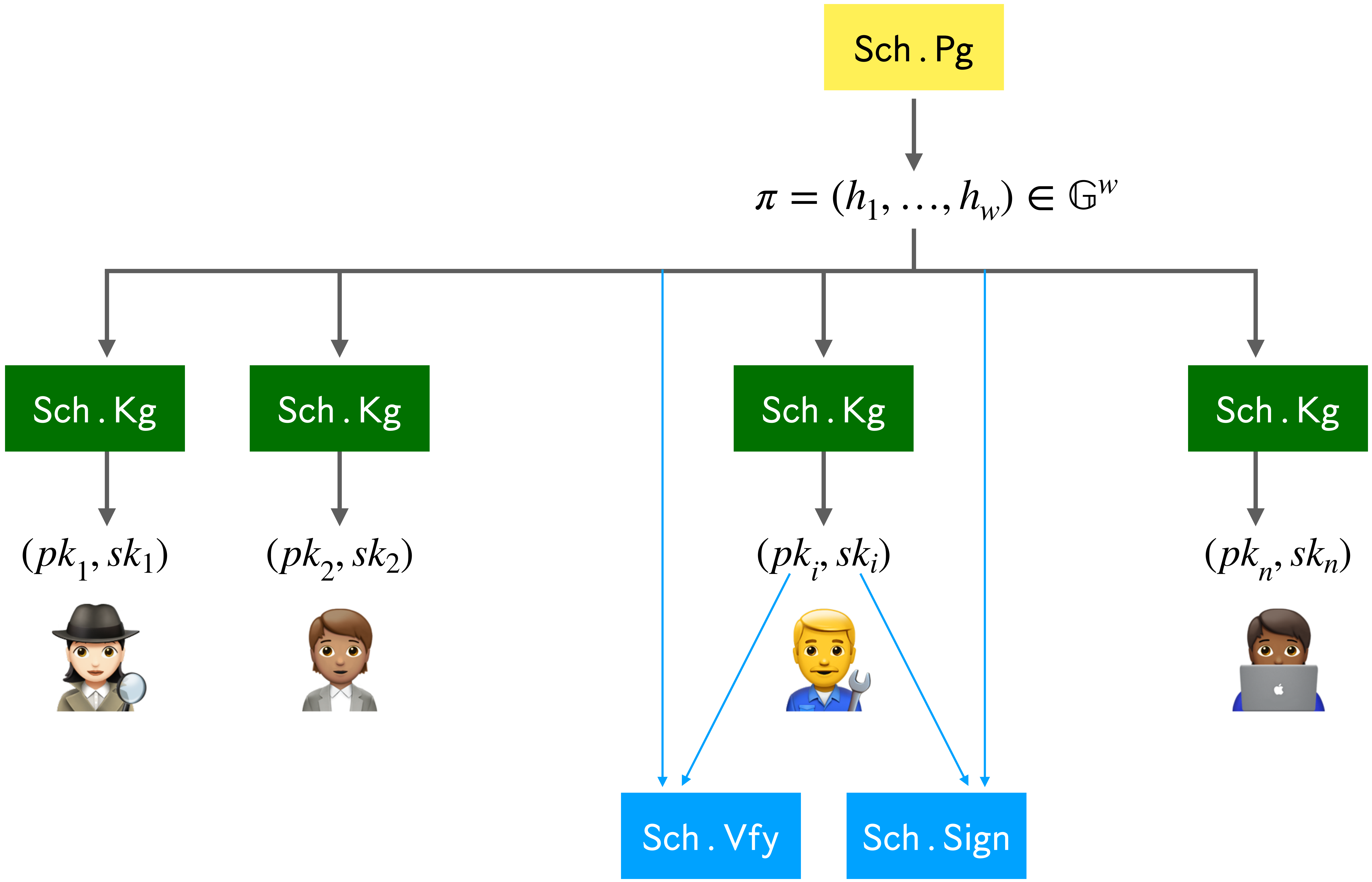
Group-Element-Parameter (GEP) Schemes

Let Sch be a width-w GEP scheme, over a fixed group described by $(\mathbb{G}, p, g)$.



Group-Element-Parameter (GEP) Schemes

Let Sch be a width-w GEP scheme, over a fixed group described by $(\mathbb{G}, p, g)$.



There are many such schemes, including:

w=1

Okamoto Signatures

Katz-Wang Signatures

Cramer-Shoup PKE

Dual EC PRG

w=2

SPAKE2

w=4

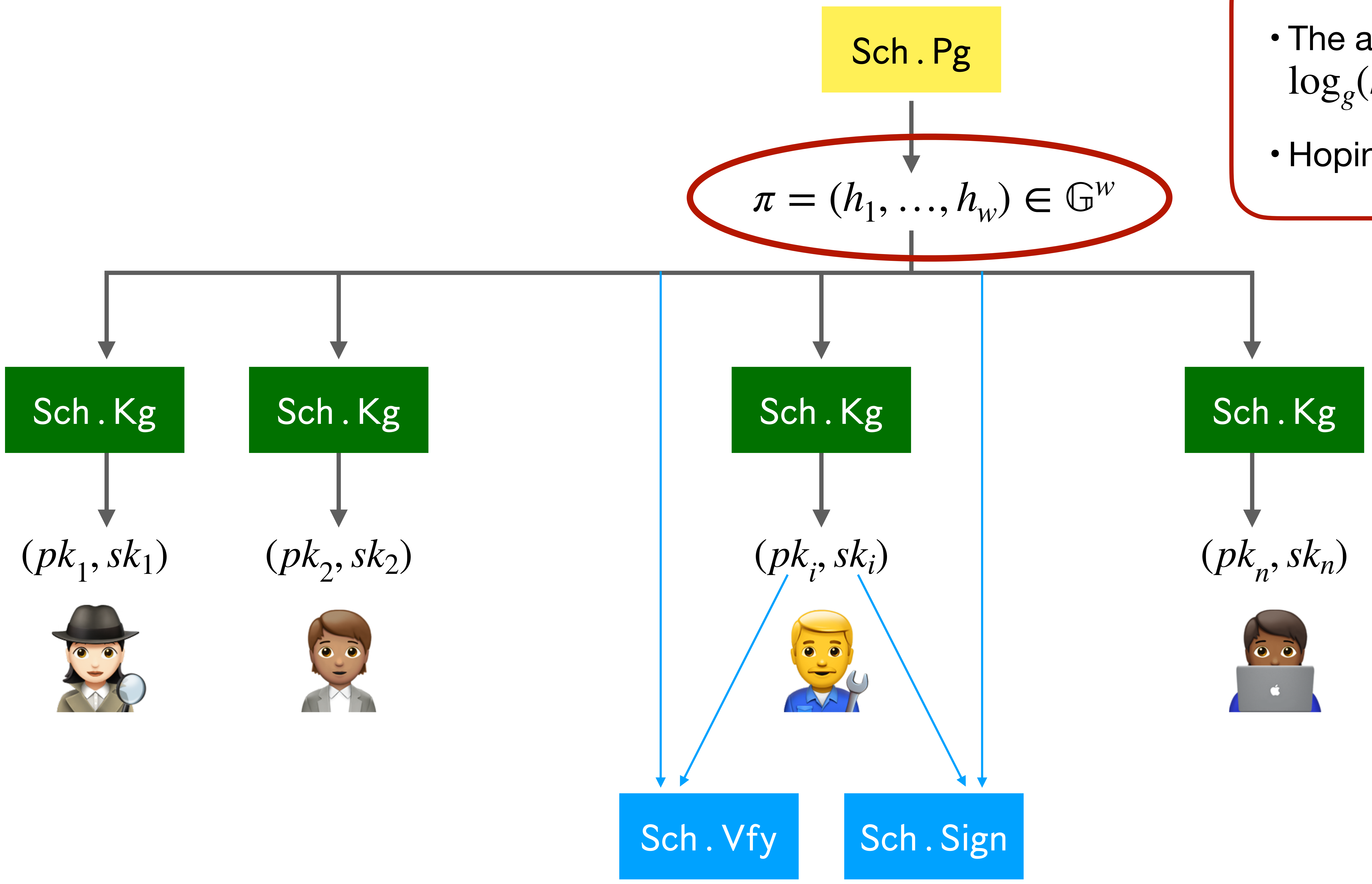
KOY PAKE

w=?

...

Group-Element-Parameter (GEP) Schemes

Let Sch be a width-w GEP scheme, over a fixed group described by $(\mathbb{G}, p, g)$.

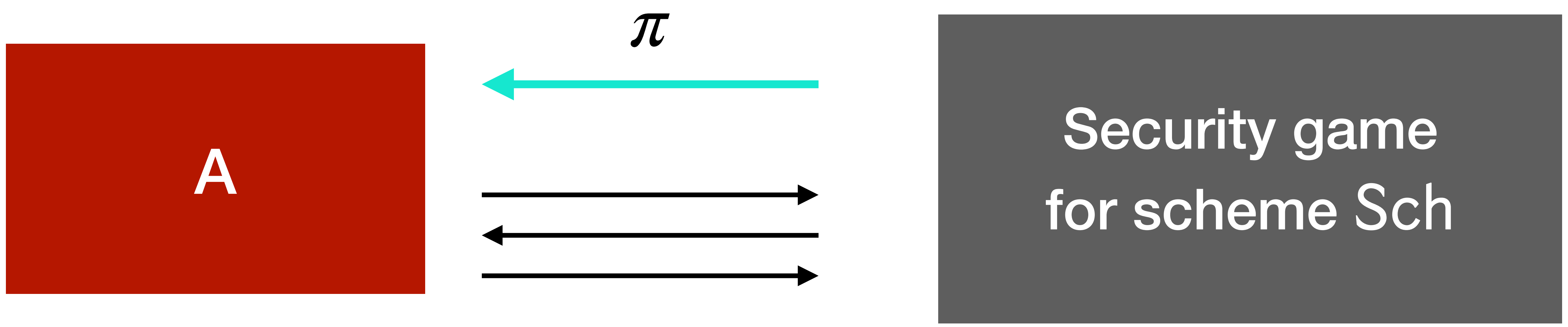


One natural Intermundium-DL concern is:

- The adversary computes w discrete logs $\log_g(h_1), \ldots, \log_g(h_w)$
- Hoping to compromise all n users!

$(w \ll n)$

Security games: One formalism of security goals

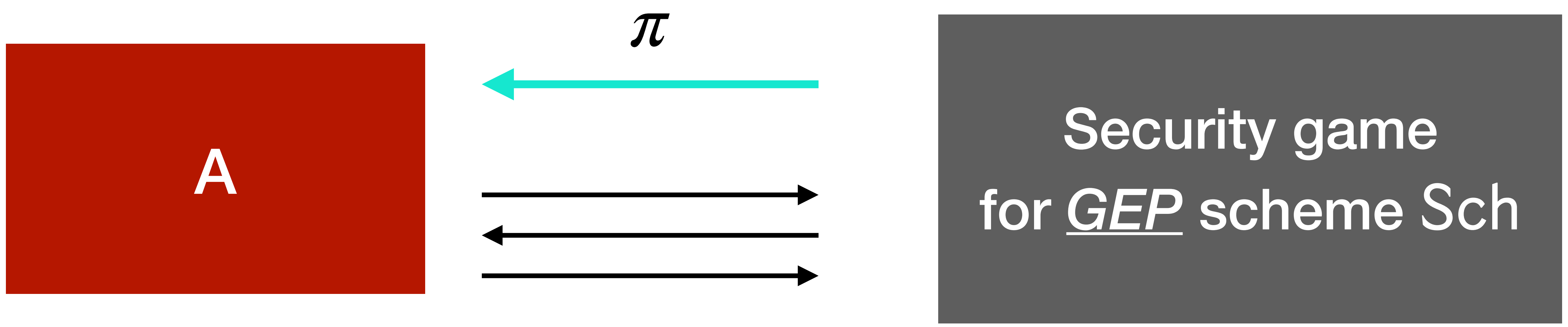


Chooses public parameters $\pi \xleftarrow{\$} \mathsf{Sch.Pg}$

Responds to oracle queries

Decides if A won the game.

Security games: One formalism of security goals



Chooses public parameters $\pi = (h_1, \ldots, h_w) \xleftarrow{\$} \mathsf{Sch.Pg}$

Responds to oracle queries

Decides if A won the game.

Security games: One formalism of security goals

A

π

Security game
for GEP scheme Sch

Chooses public parameters $\pi = (h_1, \ldots, h_w) \xleftarrow{\$} \mathsf{Sch.Pg}$

Responds to oracle queries

Decides if A won the game.

Security games, *With Advice*: Our approach to formalizing Intermundium-DL

A

Advice
Security game
for GEP scheme Sch

Security games: One formalism of security goals

A

π

Security game
for GEP scheme Sch

Chooses public parameters $\pi = (h_1, \ldots, h_w) \xleftarrow{\$} \mathsf{Sch.Pg}$

Responds to oracle queries

Decides if **A** won the game.

Security games, *With Advice*: Our approach to formalizing Intermundium-DL

A

$\pi, advice$

Advice
Security game
for GEP scheme Sch

Chooses public parameters **AND advice**

s.t. $\pi = (h_1, \ldots, h_w)$

$advice = (\beta_1, \ldots, \beta_w) = (\log_g(h_1), \ldots, \log_g(h_w))$

Security games: One formalism of security goals

A

π

Security game
for GEP scheme Sch

Chooses public parameters $\pi = (h_1, \ldots, h_w) \xleftarrow{\$} \mathsf{Sch.Pg}$

Responds to oracle queries

Decides if **A** won the game.

Security games, *With Advice*: Our approach to formalizing Intermundium-DL

A

$\pi, advice$

Advice
Security game
for GEP scheme Sch

Chooses public parameters **AND advice**

s.t. $\pi = (h_1, \ldots, h_w)$

$advice = (\beta_1, \ldots, \beta_w) = (\log_g(h_1), \ldots, \log_g(h_w))$

Responds to oracle queries

Decides if **A** won the game.

Security games: One formalism of security goals

A

π

Security game
for GEP scheme Sch

Chooses public parameters $\pi = (h_1, \ldots, h_w) \stackrel{\$}{\leftarrow} \mathsf{Sch.Pg}$

Responds to oracle queries

Decides if **A** won the game.

UFCMA | **CPA** | **PAKE** | **…**

A-UFCMA | **A-CPA** | **A-PAKE** | **…**

Security games, *With Advice*: Our approach to formalizing Intermundium-DL

A

$\pi, advice$

Advice
Security game
for GEP scheme Sch

Chooses public parameters **AND advice**

s.t. $\pi = (h_1, \ldots, h_w)$

$advice = (\beta_1, \ldots, \beta_w) = (\log_g(h_1), \ldots, \log_g(h_w))$

Responds to oracle queries

Decides if **A** won the game.

A

$\pi, advice$

Advice

Security game

for GEP scheme Sch

Chooses public parameters **AND advice**

s.t. $\pi = (h_1, \ldots, h_w)$

$$advice = (\beta_1, \ldots, \beta_w) = (\log_g(h_1), \ldots, \log_g(h_w))$$

Responds to oracle queries

Decides if A won the game.

These Advice-Security games also capture a natural backdooring strategy, as occurred with Dual EC.

A

π, *advice*

Advice
Security game
for GEP scheme Sch

Chooses public parameters **AND advice**

s.t. $\pi = (h_1, \ldots, h_w)$

$advice = (\beta_1, \ldots, \beta_w) = (\log_g(h_1), \ldots, \log_g(h_w))$

Responds to oracle queries

Decides if A won the game.

$\boxed{\beta} \xleftarrow{\$} \mathbb{Z}_p^*$; $h \leftarrow g^{\beta}$ $\longrightarrow$ $\pi = h$ NIST SP 800-90, ANSI X9.82

These Advice-Security games also capture a natural **backdooring** strategy, as occurred with Dual EC.

A

$\pi, advice$

Advice
Security game
for GEP scheme Sch

Chooses public parameters **AND advice**

s.t. $\pi = (h_1, \ldots, h_w)$

$advice = (\beta_1, \ldots, \beta_w) = (\log_g(h_1), \ldots, \log_g(h_w))$

Responds to oracle queries

Decides if A won the game.

$\boxed{\beta} \xleftarrow{\$} \mathbb{Z}_p^* \,;\, h \leftarrow g^{\beta}$ $\longrightarrow$ $\pi = h$

NIST SP 800-90, ANSI X9.82

These Advice-Security games also capture a natural **backdooring** strategy, as occurred with Dual EC.

Our results can also be seen as answering how resilient GEP schemes are to this natural backdoor.

A

π, $advice$

Advice
Security game
for GEP scheme Sch

Chooses public parameters **AND advice**

s.t. $\pi = (h_1, \ldots, h_w)$

$advice = (\beta_1, \ldots, \beta_w) = (\log_g(h_1), \ldots, \log_g(h_w))$

Responds to oracle queries

Decides if A won the game.

What happens to security of GEP schemes when an adversary has this advice?

Possible Questions:

1. Can we build schemes that are A-Secure (Advice-Secure)?

2. Are existing schemes A-Secure?

What happens to security of GEP schemes when an adversary has this advice?

Possible Questions:

[Not so interesting.
And, may incur other costs.]

1. Can we build schemes that are A-Secure (Advice-Secure)?

Yes, trivially. e.g. Don't have public parameters.

2. Are existing schemes A-Secure?

What happens to security of GEP schemes when an adversary has this advice?

Possible Questions:

1. Can we build schemes that are A-Secure (Advice-Secure)?

[Not so interesting.
And, may incur other costs.]

Yes, trivially. e.g. Don't have public parameters.

2. Are existing schemes A-Secure?

[Our question]

There are legacy systems, will they remain secure in Intermundium-DL?

What happens to security of ***existing*** GEP schemes when an adversary has this advice?

What happens to security of ***existing*** GEP schemes when an adversary has this advice?

We came across 3 categories:

- **A-INSECURE:** The scheme is broken!
- **A-SECURE:** The scheme is still completely secure! The public parameters didn't actually need a trusted setup.
- Something else? **PARTIALLY A-SECURE.**

What happens to security of ***existing*** GEP schemes when an adversary has this advice?

	w	Known results	Our results

What happens to security of ***existing*** GEP schemes when an adversary has this advice?

	w	Known results	Our results
Okamoto	1	UF-CMA under DL	A-UF-CMA under DL
Katz-Wang	1	UF-CMA under DDH	A-UF-CMA under DL

What happens to security of ***existing*** GEP schemes when an adversary has this advice?

	w	Known results	Our results
Okamoto	1	UF-CMA under DL	A-UF-CMA under DL
Katz-Wang	1	UF-CMA under DDH	A-UF-CMA under DL
Cramer-Shoup	1	CPA under DDH CCA-1, CCA-2 under DDH	A-CPA under DDH [Rosulek] A-CCA-1 under DT-DDH

What happens to security of *existing* GEP schemes when an adversary has this advice?

	w	Known results	Our results
Okamoto	1	UF-CMA under DL	A-UF-CMA under DL
Katz-Wang	1	UF-CMA under DDH	A-UF-CMA under DL
Cramer-Shoup	1	CPA under DDH CCA-1, CCA-2 under DDH	A-CPA under DDH [Rosulek] A-CCA-1 under DT-DDH
KOY	4	PAKE-secure under DDH	Broken!

What happens to security of *existing* GEP schemes when an adversary has this advice?

	w	Known results	Our results
Okamoto	1	UF-CMA under DL	A-UF-CMA under DL
Katz-Wang	1	UF-CMA under DDH	A-UF-CMA under DL
Cramer-Shoup	1	CPA under DDH CCA-1, CCA-2 under DDH	A-CPA under DDH [Rosulek] A-CCA-1 under DT-DDH
KOY	4	PAKE-secure under DDH	Broken!
SPAKE2	2	PAKE-secure under GapCDH	A-PAKE-secure under StrongCDH, assuming good passwords

☑ Setting the scene

☑ Definitions: How to formalize security in Intermundium-DL?

RESULTS

- ☐ **Signatures**
- ☐ Public-key encryption
- ☐ Password-authenticated key exchange

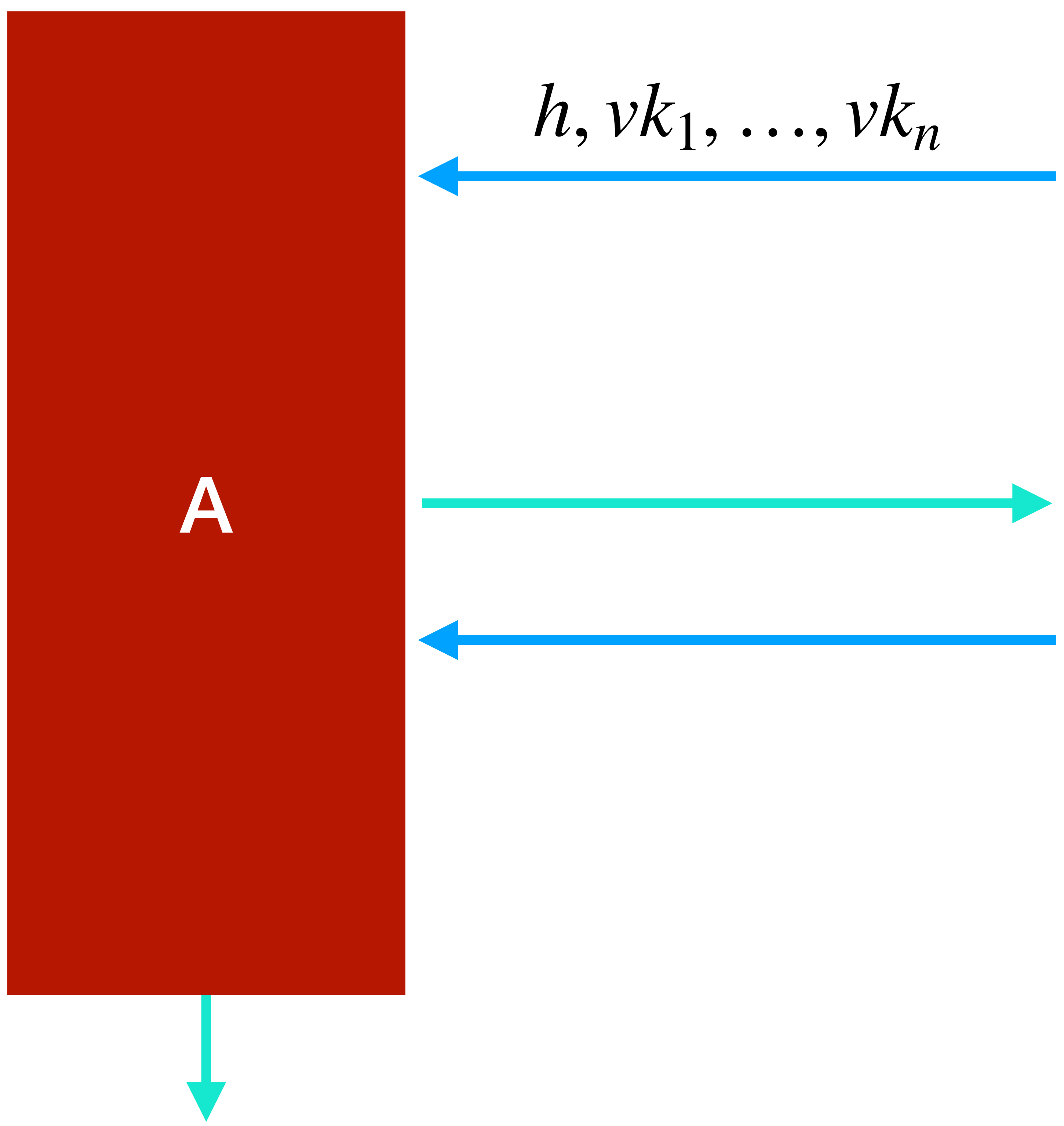
Signatures in Intermundium-DL:
UF-CMA

Simplification: The schemes in question all have width $w = 1$, so $\pi = h$.

Let S be a GEP signature scheme.

It has algorithms:

- S . Pg which outputs $\pi = h$
- S . Kg
- S . Sign
- S . Vfy



Forgery (i^*, m^*, σ^*)

UF-CMA game for GEP scheme S

$h \xleftarrow{\$}$ S . Pg

For $i = 1, \ldots, n$ do: $(vk_i, sk_i) \xleftarrow{\$}$ S . Kg(h)

Oracle $\mathtt{Sign}(i, m)$:

$Q \leftarrow Q \cup \{(i, m)\}$

Return $\sigma \xleftarrow{\$}$ S . Sign(sk_i, m)

A wins if:

$(i^*, m^*) \notin Q$ and S . Vfy$(vk_{i^*}, m^*, \sigma^*)$

Signatures in Intermundium-DL:

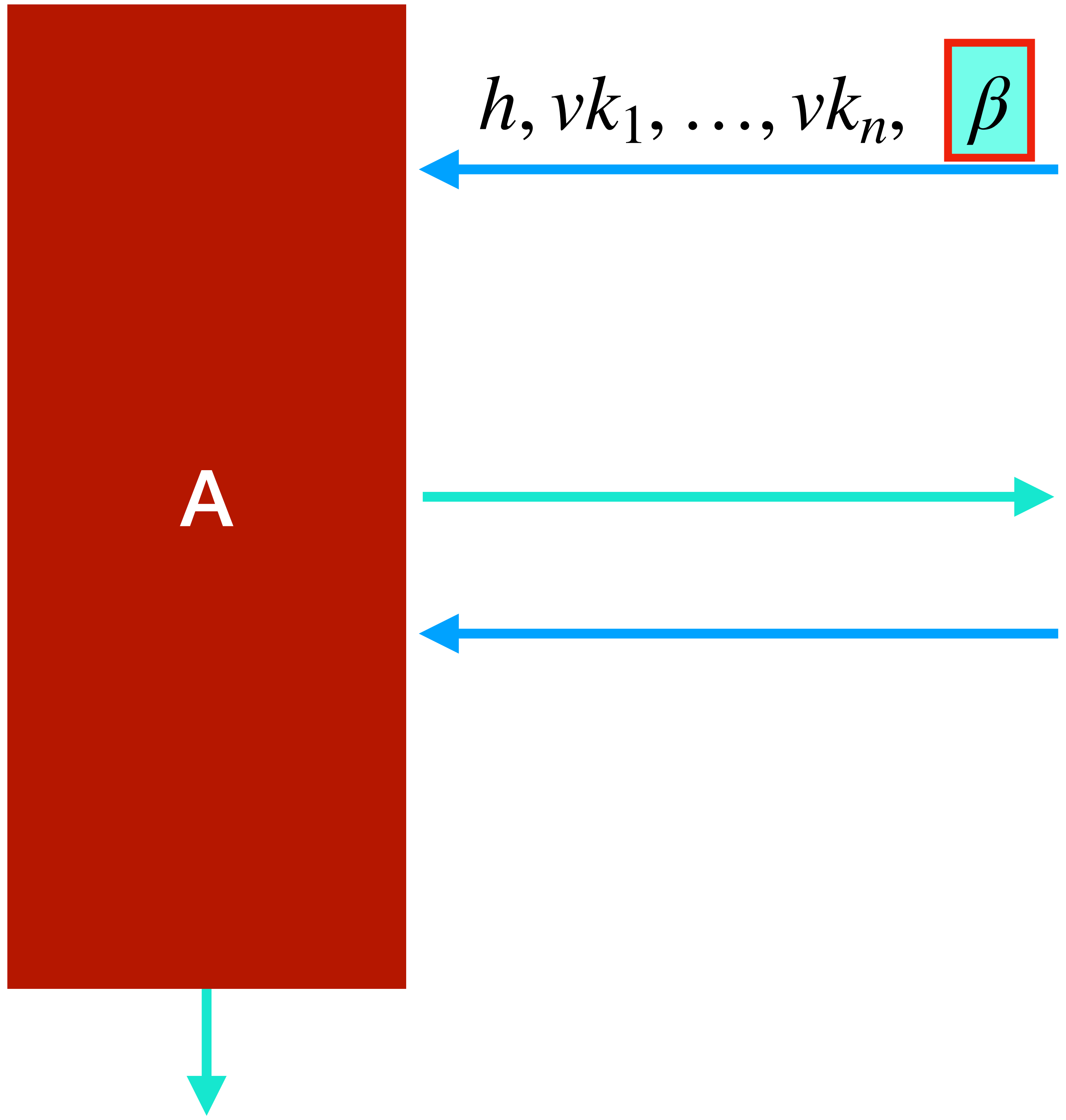
Advice-UF-CMA (A-UF-CMA)

Simplification: The schemes in question all have width $w = 1$, so $\pi = h$.

Let S be a GEP signature scheme.

It has algorithms:

- S . Pg which outputs $\pi = h$
- S . Kg
- S . Sign
- S . Vfy



Forgery (i^*, m^*, σ^*)

A-UF-CMA game for GEP scheme S

$h \xleftarrow{\$} \mathsf{S.Pg}$ $\quad \boxed{\beta \leftarrow \log_g(h)}$

For $i = 1, \ldots, n$ do: $(vk_i, sk_i) \xleftarrow{\$} \mathsf{S.Kg}(h)$

Oracle $\mathtt{Sign}(i, m)$:

$Q \leftarrow Q \cup \{(i, m)\}$

Return $\sigma \xleftarrow{\$} \mathsf{S.Sign}(sk_i, m)$

A wins if:

$(i^*, m^*) \notin Q$ and $\mathsf{S.Vfy}(vk_{i^*}, m^*, \sigma^*)$

Signatures in Intermundium-DL:
Old results

Prior UF-CMA results:

DL $\Rightarrow$ Okamoto UF-CMA

$\pi = h$

This reduction [O92] says:
Given a UF-CMA adversary breaking Okamoto,
we can build a DL adversary which, given h, finds $\log_g(h)$.

Signatures in Intermundium-DL:

Old results

Prior UF-CMA results:

DL $\Rightarrow$ Okamoto UF-CMA

$\pi = h$

This reduction [O92] says:
Given a UF-CMA adversary breaking Okamoto,
we can build a DL adversary which, given h, finds $\log_g(h)$.

DDH $\Rightarrow$ Katz-Wang UF-CMA

$\pi = h$

This reduction [KW03] says:
Given a UF-CMA adversary breaking Katz-Wang,
we can build a DDH adversary which, given (g, h, B, C), decides if $C = B^{\log_g(h)}$.

Signatures in Intermundium-DL:
Old results

Prior UF-CMA results:

Advice-UF-CMA results: ??

DL $\Rightarrow$ Okamoto UF-CMA

$\pi = h$

This reduction [O92] says:
Given a UF-CMA adversary breaking Okamoto,
we can build a DL adversary which, given h, finds $\boxed{\log_g(h)}$.

DDH $\Rightarrow$ Katz-Wang UF-CMA

$\pi = h$

This reduction [KW03] says:
Given a UF-CMA adversary breaking Katz-Wang,
we can build a DDH adversary which, given (g, h, B, C), decides if $C = B^{\boxed{\log_g(h)}}$

These reductions won't work for A-UF-CMA, since the advice $\log_g(h)$ must be revealed to the adversary.

Signatures in Intermundium-DL:
Old and new results

Prior UF-CMA results:

Our Advice-UF-CMA results:

DL $\Rightarrow$ Okamoto UF-CMA

$\pi = h$

DDH $\Rightarrow$ Katz-Wang UF-CMA

$\pi = h$

DL $\Rightarrow$ Schnorr UF-CMA

No public parameters

Signatures in Intermundium-DL:
Old and new results

Prior UF-CMA results:

DL ⇒ Okamoto UF-CMA

$\pi = h$

DDH ⇒ Katz-Wang UF-CMA

$\pi = h$

DL ⇒ Schnorr UF-CMA

No public parameters

Our Advice-UF-CMA results:

Schnorr **UF-CMA** ⇒ Okamoto **A-UF-CMA**

Schnorr **UF-CMA** ⇒ Katz-Wang **A-UF-CMA**

An illustration: Okamoto in Intermundium-DL

Theorem: Given an adversary *B* against A-UF-CMA of Okamoto, we can construct adversary *A* against UF-CMA of Schnorr.

Fixed group described by: $(\mathbb{G}, p, g)$

Fixed hash function: H

An illustration: Okamoto in Intermundium-DL

Theorem: Given an adversary *B* against A-UF-CMA of Okamoto, we can construct adversary *A* against UF-CMA of Schnorr.

Fixed group described by: $(\mathbb{G}, p, g)$

Fixed hash function: H

Okamoto Pg:

$\beta \leftarrow_{\$} \mathbb{Z}_p^* \; ; \; h \leftarrow g^{\beta}$

Return h

Okamoto Kg(h):

$s_1, s_2 \leftarrow_{\$} \mathbb{Z}_p$

$X \leftarrow g^{s_1} h^{s_2}$

Return $(X, (s_1, s_2))$

Okamoto Sign($h, X, (s_1, s_2), m$):

$r_1, r_2 \leftarrow_{\$} \mathbb{Z}_p$

$R \leftarrow g^{r_1} h^{r_2}$

$e \leftarrow \mathsf{H}(X, R, m)$

$y_1 \leftarrow (r_1 + e s_1) \bmod p$

$y_2 \leftarrow (r_2 + e s_2) \bmod p$

Return $\sigma \leftarrow (e, y_1, y_2)$

Schnorr Kg(ε):

$s \leftarrow_{\$} \mathbb{Z}_p$

$X \leftarrow g^{s}$

Return (X, s)

Schnorr Sign(ε, X, s, m):

$r \leftarrow_{\$} \mathbb{Z}_p$

$R \leftarrow g^{r}$

$e \leftarrow \mathsf{H}(X, R, m)$

$y \leftarrow (r + e s) \bmod p$

Return $\sigma \leftarrow (e, y)$

An illustration: Okamoto in Intermundium-DL

Theorem: Given an adversary *B* against A-UF-CMA of Okamoto, we can construct adversary *A* against UF-CMA of Schnorr.

Fixed group described by: $(\mathbb{G}, p, g)$ Fixed hash function: H

Okamoto Pg:

1 $\beta \leftarrow_{\$} \mathbb{Z}_p^*$; $h \leftarrow g^{\beta}$

2 Return h

Okamoto Kg(h):

3 $s_1, s_2 \leftarrow_{\$} \mathbb{Z}_p$

4 $X \leftarrow g^{s_1} h^{s_2}$

5 Return $(X, (s_1, s_2))$

Okamoto Sign$(h, X, (s_1, s_2), m)$:

6 $r_1, r_2 \leftarrow_{\$} \mathbb{Z}_p$

7 $R \leftarrow g^{r_1} h^{r_2}$

8 $e \leftarrow \mathsf{H}(X, R, m)$

9 $y_1 \leftarrow (r_1 + e s_1) \bmod p$

10 $y_2 \leftarrow (r_2 + e s_2) \bmod p$

11 Return $\sigma \leftarrow (e, y_1, y_2)$

Adversary *A* :

Inputs: Schnorr $vk = X$ and parameter $\pi = \varepsilon$

1. Select $\boxed{\beta \xleftarrow{\$} \mathbb{Z}_p^* ; h \leftarrow g^{\beta}}$

2. Run *B* with Okamoto $vk = X$, parameter h and advice $\boxed{\beta}$.

When *B* makes an Okamoto Sign(m) query:

A will...

(i) Make a Schnorr query $(e, y) \leftarrow \mathsf{Sign}(m)$

(ii) Select $y_2 \xleftarrow{\$} \mathbb{Z}_p$

(iii) Set $y_1 \leftarrow (y - \boxed{\beta} y_2)$

(iv) Return to *B* the signature (e, y_1, y_2).

3. When *B* outputs a forgery $(m, (e, y_1, y_2))$, *A* outputs Schnorr forgery $(m, (e, y_1 + \boxed{\beta} y_2))$.

Schnorr Kg(ε):

1 $s \leftarrow_{\$} \mathbb{Z}_p$

2 $X \leftarrow g^s$

3 Return (X, s)

Schnorr Sign(ε, X, s, m):

4 $r \leftarrow_{\$} \mathbb{Z}_p$

5 $R \leftarrow g^r$

6 $e \leftarrow \mathsf{H}(X, R, m)$

7 $y \leftarrow (r + es) \bmod p$

8 Return $\sigma \leftarrow (e, y)$

- [x] Setting the scene
- [x] Definitions: How to formalize security in Intermundium-DL?

RESULTS

  - [x] Signatures
  - [ ] **Public-key encryption**
  - [ ] Password-authenticated key exchange

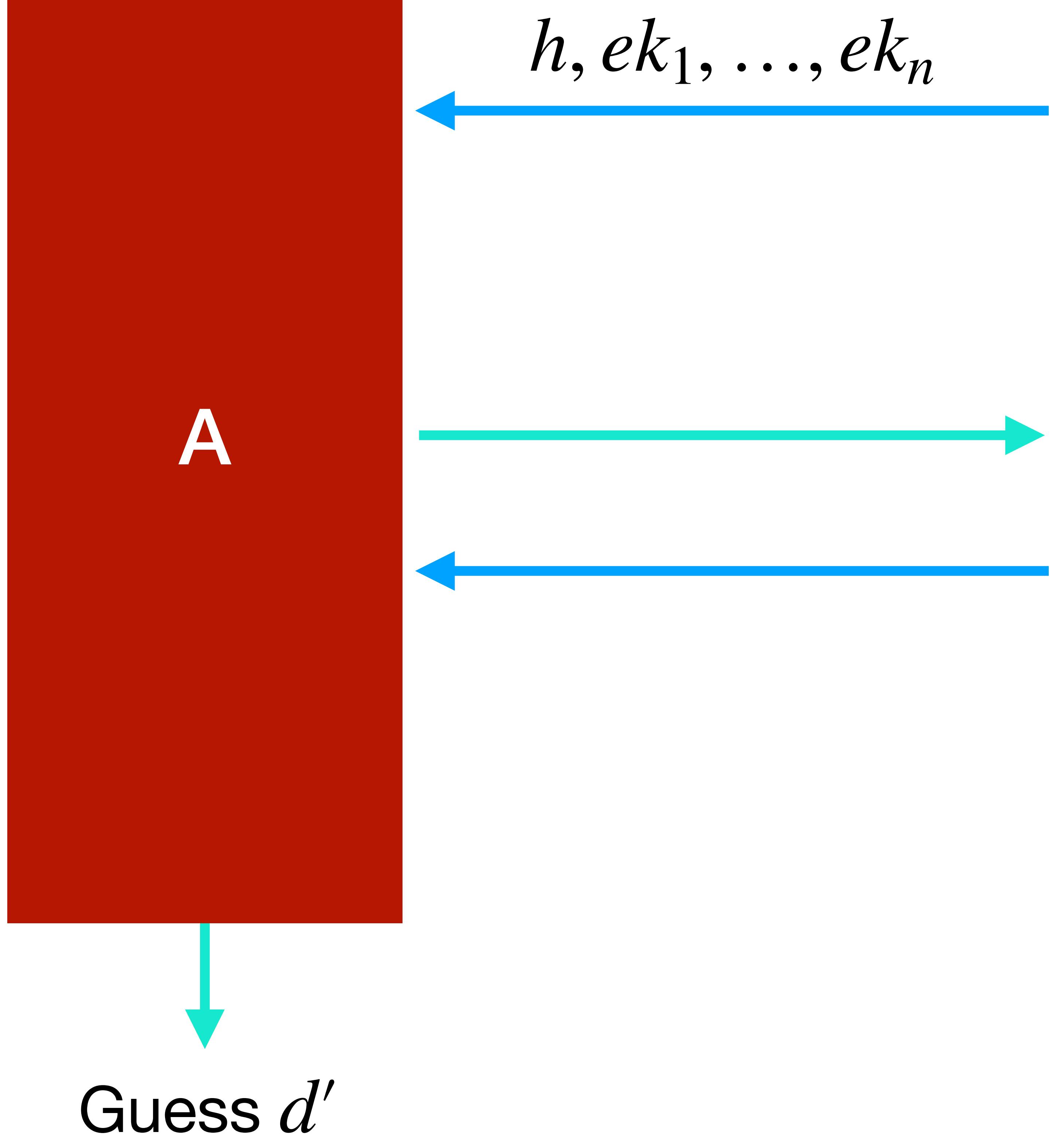
Encryption in Intermundium-DL:
CPA, CCA1, CCA2

Simplification: The schemes in question all have width $w = 1$, so $\pi = h$.

Let PKE be a GEP scheme.

It has algorithms:

- PKE . Pg which outputs $\pi = h$
- PKE . Kg
- PKE . Enc
- PKE . Dec



CPA, CCA1, CCA2 games for GEP scheme PKE

$h \xleftarrow{\$}$ PKE . Pg $\quad\quad d \xleftarrow{\$} \{0,1\}$

For $i = 1,\ldots,n$ do: $(ek_i, dk_i) \xleftarrow{\$}$ PKE . Kg(h)

Oracle $\mathtt{Enc}(i, m_0, m_1)$:

$C^* \xleftarrow{\$}$ PKE . Enc(h, ek_i, m_d)

Return C^*

Oracle $\mathtt{Dec}(i, C)$:

$M \leftarrow$ PKE . Dec(h, dk_i, C)

<u>If allowed</u>, Return M

<u>Dec queries:</u>

CPA: Never allowed

CCA1: Allowed *before* **Enc** queries

CCA2: Allowed at any time*

A wins if:

$d' = d$

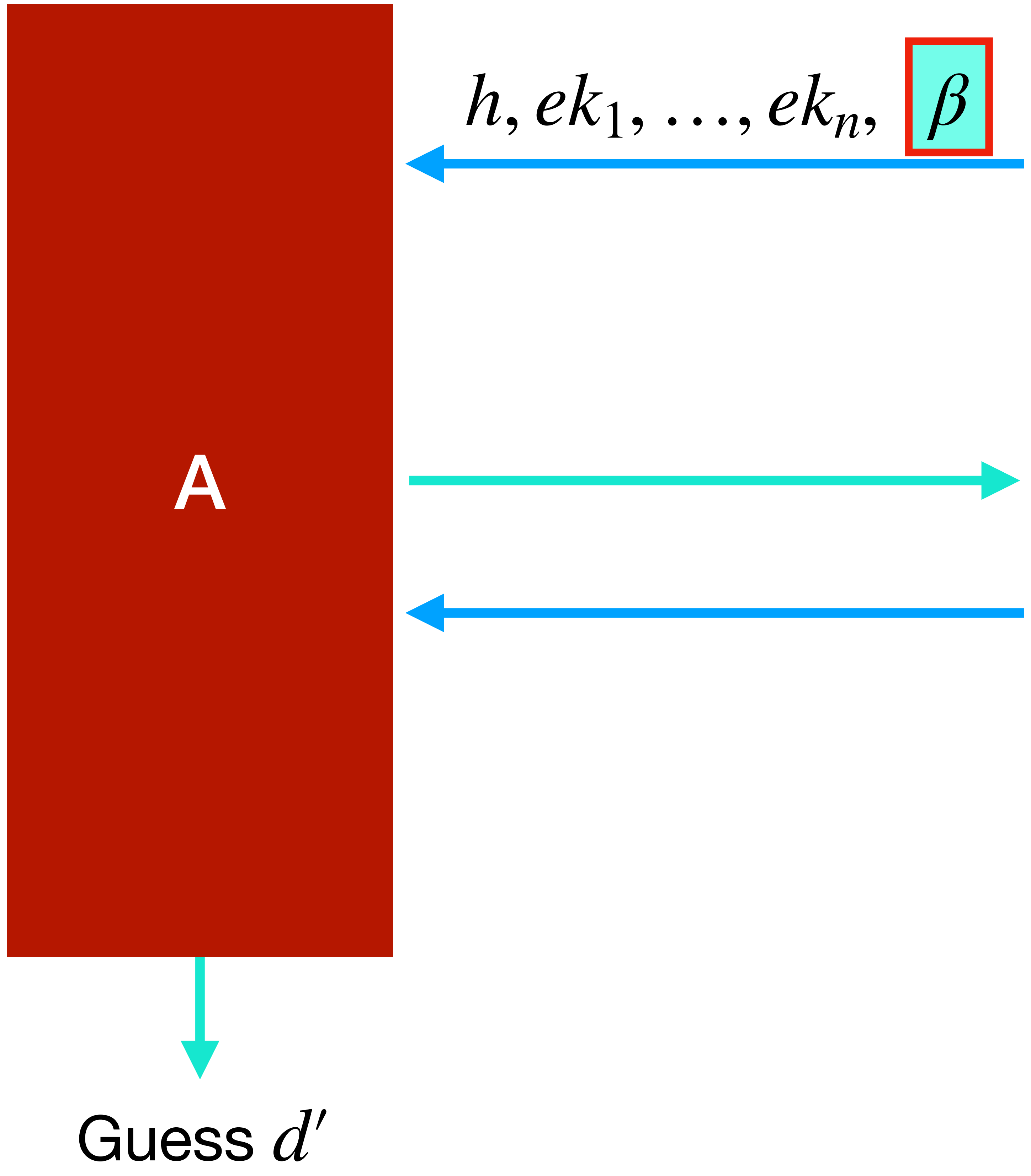
Encryption in Intermundium-DL:
Advice-{CPA, CCA1, CCA2}

Simplification: The schemes in question all have width $w = 1$, so $\pi = h$.

Let PKE be a GEP scheme.

It has algorithms:

- PKE . Pg which outputs $\pi = h$
- PKE . Kg
- PKE . Enc
- PKE . Dec



A-{CPA, CCA1, CCA2} for GEP scheme PKE

$h \xleftarrow{\$} \mathsf{PKE.Pg}$ $\quad \boxed{\beta \leftarrow \log_g(h)}$ $\quad d \xleftarrow{\$} \{0,1\}$

For $i = 1,\ldots,n$ do: $(ek_i, dk_i) \xleftarrow{\$} \mathsf{PKE.Kg}(h)$

Oracle $\mathtt{Enc}(i, m_0, m_1)$:

$C^* \xleftarrow{\$} \mathsf{PKE.Enc}(h, ek_i, m_d)$

Return C^*

Oracle $\mathtt{Dec}(i, C)$:

$M \leftarrow \mathsf{PKE.Dec}(h, dk_i, C)$

<u>If allowed</u>, Return M

<u>Dec queries:</u>

CPA: Never allowed

CCA1: Allowed *before* **Enc** queries

CCA2: Allowed at any time*

A wins if:

$d' = d$

Encryption in Intermundium-DL:
Old results

Prior results:

DDH ⇒ Cramer-Shoup **CPA**

DDH ⇒ Cramer-Shoup **CCA1**

DDH ⇒ Cramer-Shoup **CCA2**

[CS03 and others]

CS: $\pi = h$

Encryption in Intermundium-DL:
Old and new results

Prior results:

DDH ⇒ Cramer-Shoup **CPA**

DDH ⇒ Cramer-Shoup **CCA1**

DDH ⇒ Cramer-Shoup **CCA2**

[CS03 and others]

CS: $\pi = h$

Our Advice results:

DDH ⇒ Cramer-Shoup **A-CPA**

DT-DDH ⇒ Cramer-Shoup **A-CCA1**

??

The “Delayed-Target DDH” (DT-DDH) problem

$\boxed{\text{Game } \mathrm{DDH}_{\mathbb{G},p,g}}$ Group $\mathbb{G}$, of order p, with generator g

$$a, b, r \xleftarrow{\$} \mathbb{Z}_p^3$$

$$d \xleftarrow{\$} \{0,1\}$$

If $d = 1$ then return (g^a, g^b, g^{ab})

If $d = 0$ then return (g^a, g^b, g^r)

$\longrightarrow$ **A** $\longrightarrow d'$

Adversary **A** wins game DDH if $d' = d$.

The "Delayed-Target DDH" (DT-DDH) problem

$\boxed{\text{Game } \mathrm{DDH}_{\mathbb{G},p,g}}$ Group $\mathbb{G}$, of order p, with generator g

$$a, b, r \xleftarrow{\$} \mathbb{Z}_p^3$$

$$d \xleftarrow{\$} \{0,1\}$$

If $d = 1$ then return (g^a, g^b, g^{ab})

If $d = 0$ then return (g^a, g^b, g^r)

$\longrightarrow$ A $\longrightarrow d'$

Adversary A wins game DDH if $d' = d$.

$\boxed{\text{Game } \mathrm{DT\text{-}DDH}_{\mathbb{G},p,g}}$

$$a, b, r \xleftarrow{\$} \mathbb{Z}_p^3$$

$$d \xleftarrow{\$} \{0,1\}$$

Return $\boxed{g^a}$ $\longrightarrow$ A

The "Delayed-Target DDH" (DT-DDH) problem

Game $\mathrm{DDH}_{\mathbb{G},p,g}$

Group $\mathbb{G}$, of order p, with generator g

$a, b, r \xleftarrow{\$} \mathbb{Z}_p^3$

$d \xleftarrow{\$} \{0,1\}$

If $d = 1$ then return (g^a, g^b, g^{ab})

If $d = 0$ then return (g^a, g^b, g^r)

A

d'

Adversary A wins game DDH if $d' = d$.

Game $\mathrm{DT\text{-}DDH}_{\mathbb{G},p,g}$

$a, b, r \xleftarrow{\$} \mathbb{Z}_p^3$

$d \xleftarrow{\$} \{0,1\}$

Return $\boxed{g^a}$

A

Oracle $\mathrm{CDH}(Y)$: Return Y^a

The “Delayed-Target DDH” (DT-DDH) problem

Game $\mathrm{DDH}_{\mathbb{G},p,g}$

Group $\mathbb{G}$, of order p, with generator g

$a, b, r \xleftarrow{\$} \mathbb{Z}_p^3$

$d \xleftarrow{\$} \{0,1\}$

If $d = 1$ then return (g^a, g^b, g^{ab})

If $d = 0$ then return (g^a, g^b, g^r)

A

d'

Adversary A wins game DDH if $d' = d$.

Game $\mathrm{DT\text{-}DDH}_{\mathbb{G},p,g}$

$a, b, r \xleftarrow{\$} \mathbb{Z}_p^3$

$d \xleftarrow{\$} \{0,1\}$

Return $\boxed{g^a}$

A

Oracle $\cancel{\mathrm{CDH}(Y)}$: Return Y^a

Oracle “Request Target”:

The "Delayed-Target DDH" (DT-DDH) problem

Game $\mathrm{DDH}_{\mathbb{G},p,g}$

Group $\mathbb{G}$, of order p, with generator g

$a, b, r \xleftarrow{\$} \mathbb{Z}_p^3$

$d \xleftarrow{\$} \{0,1\}$

If $d = 1$ then return (g^a, g^b, g^{ab})

If $d = 0$ then return (g^a, g^b, g^r)

A

d'

Adversary A wins game DDH if $d' = d$.

Game $\mathrm{DT\text{-}DDH}_{\mathbb{G},p,g}$

$a, b, r \xleftarrow{\$} \mathbb{Z}_p^3$

$d \xleftarrow{\$} \{0,1\}$

Return $\boxed{g^a}$

A

Oracle $\cancel{\mathrm{CDH}(Y)}$: Return Y^a

Oracle "Request Target":

If $d = 1$ then return (g^b, g^{ab})

If $d = 0$ then return (g^b, g^r)

The “Delayed-Target DDH” (DT-DDH) problem

Game $\mathrm{DDH}_{\mathbb{G},p,g}$

Group $\mathbb{G}$, of order p, with generator g

$a, b, r \xleftarrow{\$} \mathbb{Z}_p^3$

$d \xleftarrow{\$} \{0,1\}$

If $d = 1$ then return (g^a, g^b, g^{ab})

If $d = 0$ then return (g^a, g^b, g^r)

A

d'

Adversary A wins game DDH if $d' = d$.

Game $\text{DT-DDH}_{\mathbb{G},p,g}$

$a, b, r \xleftarrow{\$} \mathbb{Z}_p^3$

$d \xleftarrow{\$} \{0,1\}$

Return $\boxed{g^a}$

A

d'

Oracle $\cancel{\mathrm{CDH}(Y)}$: Return Y^a

Oracle “Request Target”:

If $d = 1$ then return (g^b, g^{ab})

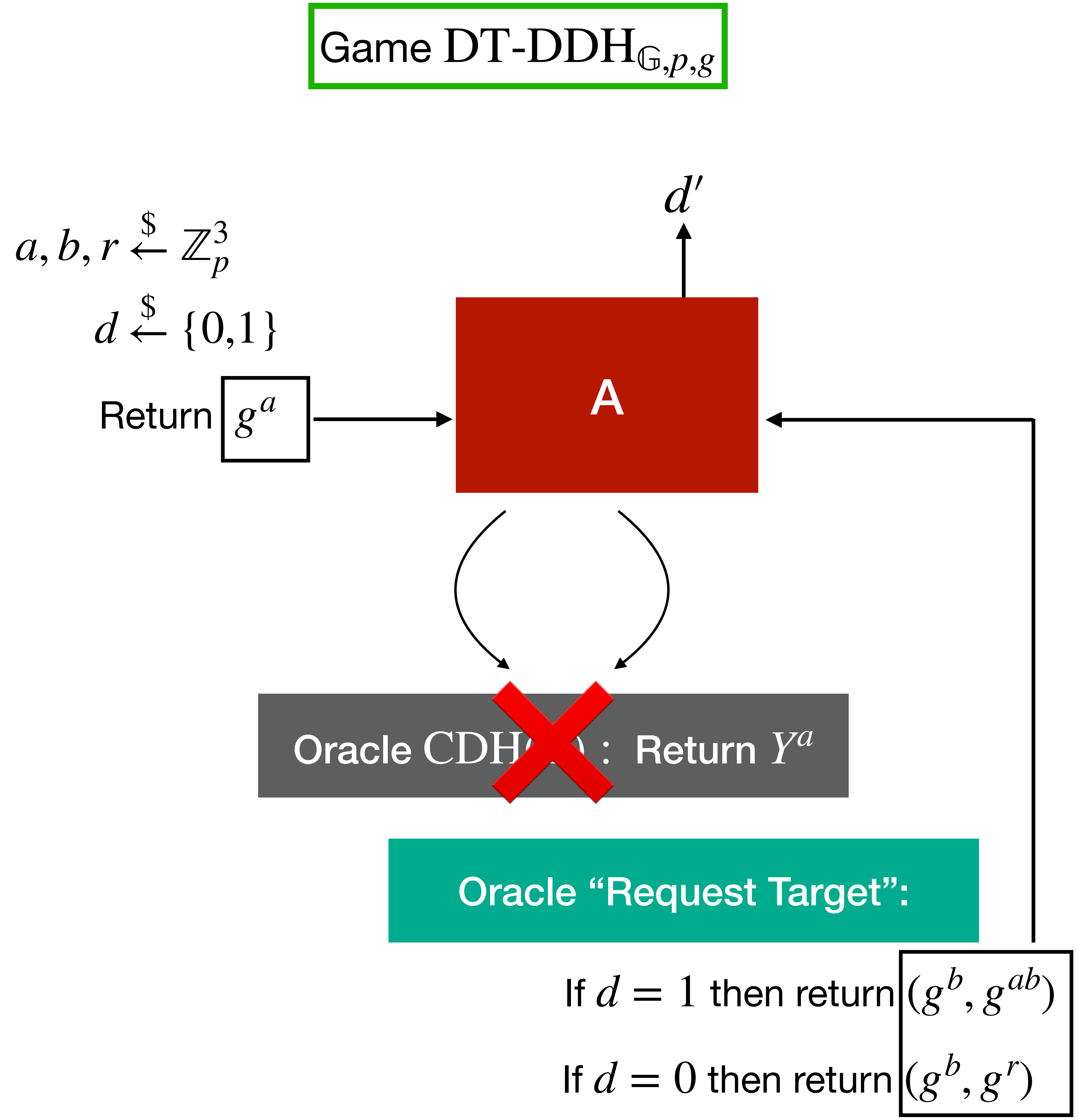
If $d = 0$ then return (g^b, g^r)

Adversary A wins game DT-DDH if $d' = d$.

The “Delayed-Target DDH” (DT-DDH) problem

Comments on DT-DDH:

- **DT-DDH** is from [L11] and CDH versions have been given as well [F05, KM08]
- Attacks (that are subexponential-time) exist for finite-field groups [JLNT09]



Adversary A wins game DT-DDH if $d' = d$.

Encryption in Intermundium-DL:
Old and new results

Prior results:

DDH $\Rightarrow$ Cramer-Shoup **CPA**

DDH $\Rightarrow$ Cramer-Shoup **CCA1**

DDH $\Rightarrow$ Cramer-Shoup **CCA2**

Our Advice results:

DDH $\Rightarrow$ Cramer-Shoup **A-CPA**

DT-DDH $\Rightarrow$ Cramer-Shoup **A-CCA1**

Encryption in Intermundium-DL:
Old and new results

Prior results:

DDH ⇒ Cramer-Shoup **CPA**

DDH ⇒ Cramer-Shoup **CCA1**

DDH ⇒ Cramer-Shoup **CCA2**

Our Advice results:

DDH ⇒ Cramer-Shoup **A-CPA**

DT-DDH ⇒ Cramer-Shoup **A-CCA1**

A sketch of the proof difference:

CS.Enc(h, ek, m):

$k \leftarrow_\$ \mathbb{Z}_p$

$(c, d, f) \leftarrow ek$ // Parse ek as

$u_1 \leftarrow g^k$; $u_2 \leftarrow h^k$

$e \leftarrow f^k m$

$\alpha \leftarrow \mathsf{H}(u_1, u_2, e)$

$v \leftarrow c^k d^{k\alpha}$

Return (u_1, u_2, e, v)

Given (DT-)DDH challenge (g, g^a, g^b, C)...

Encryption in Intermundium-DL:

Old and new results

Prior results:

DDH ⇒ Cramer-Shoup **CPA**

DDH ⇒ Cramer-Shoup **CCA1**

DDH ⇒ Cramer-Shoup **CCA2**

Our Advice results:

DDH ⇒ Cramer-Shoup **A-CPA**

DT-DDH ⇒ Cramer-Shoup **A-CCA1**

A sketch of the proof difference:

Given (DT-)DDH challenge (g, g^a, g^b, C)...

CS.Enc(h, ek, m):

$k \leftarrow_{\$} \mathbb{Z}_p$

$(c, d, f) \leftarrow ek$ // Parse ek as

$u_1 \leftarrow g^k$; $u_2 \leftarrow h^k$

$e \leftarrow f^k m$

$\alpha \leftarrow \mathsf{H}(u_1, u_2, e)$

$v \leftarrow c^k d^{k\alpha}$

Return (u_1, u_2, e, v)

In prior proofs,
C is embedded as u_2

Encryption in Intermundium-DL:

Old and new results

Prior results:

DDH ⇒ Cramer-Shoup **CPA**

DDH ⇒ Cramer-Shoup **CCA1**

DDH ⇒ Cramer-Shoup **CCA2**

Our Advice results:

DDH ⇒ Cramer-Shoup **A-CPA**

DT-DDH ⇒ Cramer-Shoup **A-CCA1**

A sketch of the proof difference:

Given (DT-)DDH challenge (g, g^a, g^b, C)...

CS.Enc(h, ek, m):

$k \leftarrow_{\$} \mathbb{Z}_p$

$(c, d, f) \leftarrow ek$ // Parse ek as

$u_1 \leftarrow g^k$; $u_2 \leftarrow h^k$

$e \leftarrow f^k m$

$\alpha \leftarrow \mathsf{H}(u_1, u_2, e)$

$v \leftarrow c^k d^{k\alpha}$

Return (u_1, u_2, e, v)

In prior proofs,
C is embedded as u_2

In ours,
C is embedded as f^k

$\boxed{\beta}$

And **Dec** queries are answered with the **CDH** oracle in **DT-DDH**

☑ Setting the scene

☑ Definitions: How to formalize security in Intermundium-DL?

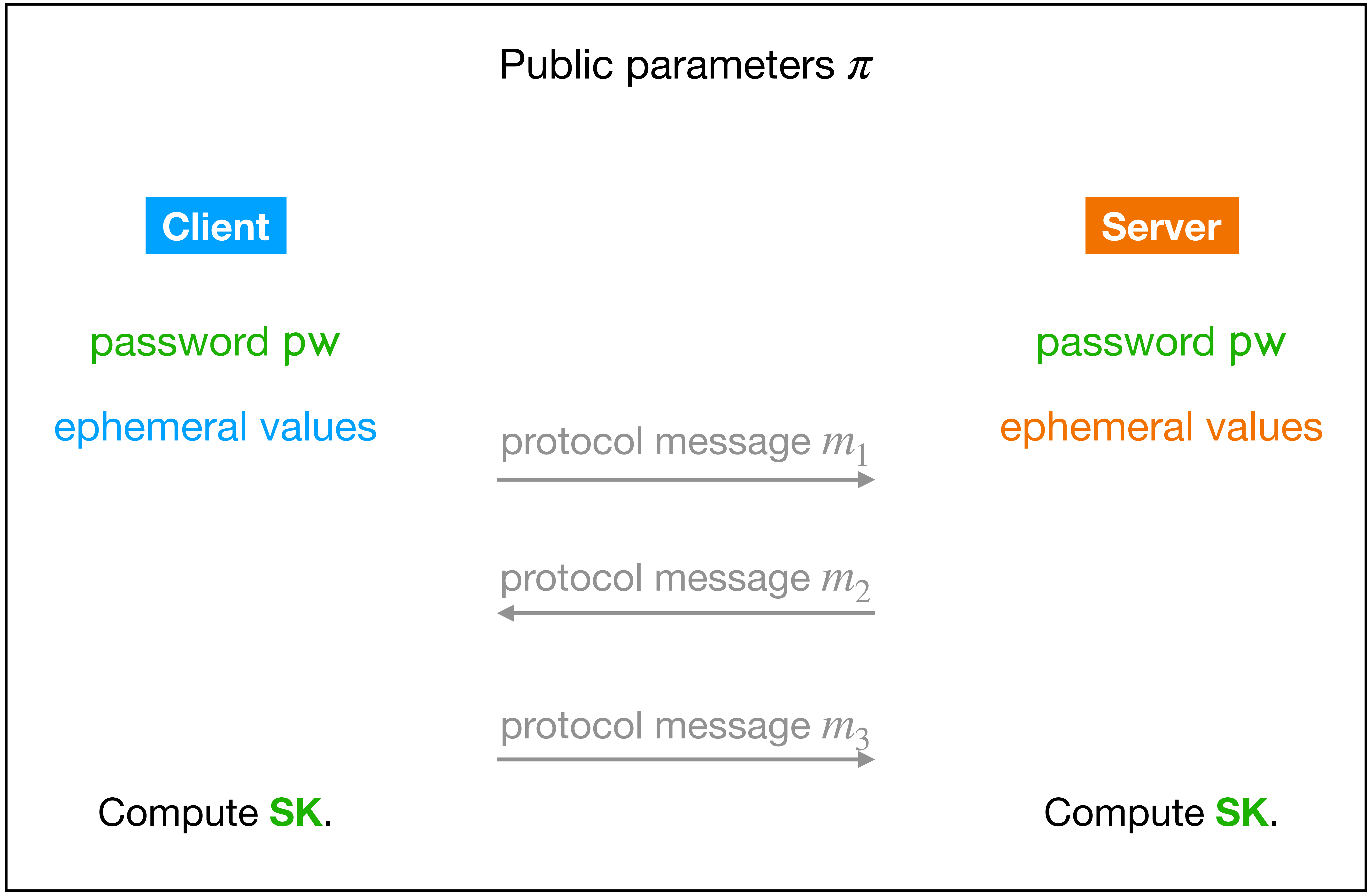
RESULTS

- ☑ Signatures
- ☑ Public-key encryption
- ☐ **Password-authenticated key exchange**

PAKEs (Password-Authenticated Key Exchange) in Intermundium-DL

What is a PAKE?

Short answer: A protocol through which, a **client** and **server** sharing a short *password*, compute a shared key.



Note: There are many clients, servers and sessions!

PAKEs (Password-Authenticated Key Exchange) in Intermundium-DL

What is a PAKE?

Short answer: A protocol through which, a **client** and **server** sharing a short *password*, compute a shared key.

Public parameters π

Client

password pw

ephemeral values

Compute **SK**.

Server

password pw

ephemeral values

Compute **SK**.

protocol message m_1 $\rightarrow$

protocol message m_2 $\leftarrow$

protocol message m_3 $\rightarrow$

Note: There are many clients, servers and sessions!

Usual PAKE security game*

An adversary ***A*** tries to distinguish between **SK** and a random key, given oracles to:

- Passively observe protocol messages
- Learn a pw or SK
- Send protocol messages
- Query a hash function (if relevant)

And is given $\pi = (h_1, \ldots, h_w)$.

* For game-based definitions. We use that of [AB19]: "*Key indistinguishability with weak forward secrecy*."

PAKEs (Password-Authenticated Key Exchange) in Intermundium-DL

What is a PAKE?

Short answer: A protocol through which, a **client** and **server** sharing a short *password*, compute a shared key.

Public parameters π

Client		**Server**
password pw		password pw
ephemeral values	protocol message m_1 $\longrightarrow$	ephemeral values
	$\longleftarrow$ protocol message m_2	
	protocol message m_3 $\longrightarrow$	
Compute **SK**.		Compute **SK**.

Note: There are many clients, servers and sessions!

Usual PAKE security game*

An adversary ***A*** tries to distinguish between **SK** and a random key, given oracles to:

- Passively observe protocol messages
- Learn a pw or SK
- Send protocol messages
- Query a hash function (if relevant)

And is given $\pi = (h_1, \ldots, h_w)$.

Advice A-PAKE security game

The same, but the adversary ***A*** is given both π and advice $\log_g(h_1), \ldots, \log_g(h_w)$.

* For game-based definitions. We use that of [AB19]: "*Key indistinguishability with weak forward secrecy.*"

The KOY protocol [Katz, Ostrovsky, Yung 09]

Public Parameters: $(g_2, h, c, d) \leftarrow_{\$} \mathsf{Gens}(\mathbb{G})^4$

Client C		**Server S**
Input: $\mathsf{pw}_{\mathsf{CS}} \in \mathbb{Z}_p$		Input: $\mathsf{pw}_{\mathsf{CS}} \in \mathbb{Z}_p$
$(vk, sk) \leftarrow_{\$} \mathsf{Sig.Kg}$ $r_1 \leftarrow_{\$} \mathbb{Z}_p$; $A \leftarrow g_1^{r_1}$; $B \leftarrow g_2^{r_1}$ $C \leftarrow h^{r_1} g_1^{\mathsf{pw}_{\mathsf{CS}}}$ $\alpha \leftarrow \mathsf{H}(\mathsf{C}, vk, A, B, C)$ $D \leftarrow (cd^{\alpha})^{r_1}$	$\xrightarrow{\mathsf{C}, vk, A, B, C, D}$	$x_2, y_2, z_2, w_2, r_2 \leftarrow_{\$} \mathbb{Z}_p$ $\alpha' \leftarrow \mathsf{H}(\mathsf{C}, vk, A, B, C)$ $E \leftarrow g_1^{x_2} g_2^{y_2} h^{z_2} (cd^{\alpha'})^{w_2}$ $F \leftarrow g_1^{r_2}$; $G \leftarrow g_2^{r_2}$; $I \leftarrow h^{r_2} g_1^{\mathsf{pw}_{\mathsf{CS}}}$ $\gamma \leftarrow \mathsf{H}(\mathsf{S}, E, F, G, I)$; $J \leftarrow (cd^{\gamma})^{r_2}$
$x_1, y_1, z_1, w_1 \leftarrow_{\$} \mathbb{Z}_p$ $\gamma' \leftarrow \mathsf{H}(\mathsf{S}, E, F, G, I)$ $K \leftarrow g_1^{x_1} g_2^{y_1} h^{z_1} (cd^{\gamma'})^{w_1}$ $\sigma \leftarrow_{\$} \mathsf{Sig.Sign}(sk, (\gamma', K))$	$\xleftarrow{\mathsf{S}, E, F, G, I, J}$ $\xrightarrow{K, \sigma}$	If $\mathsf{Sig.Vfy}(vk, (\gamma, K), \sigma) = 0$ $\mathsf{SK}' \leftarrow_{\$} \mathbb{G}$ Else: $C' \leftarrow C / g_1^{\mathsf{pw}_{\mathsf{CS}}}$ $\mathsf{SK}' \leftarrow K^{r_2} A^{x_2} B^{y_2} (C')^{z_2} D^{w_2}$
$I' \leftarrow I / g_1^{\mathsf{pw}_{\mathsf{CS}}}$ $\mathsf{SK} \leftarrow E^{r_1} F^{x_1} G^{y_1} (I')^{z_1} J^{w_1}$		
Output: SK		Output: SK'

The KOY protocol [Katz, Ostrovsky, Yung 09]

Public Parameters: $(g_2, h, c, d) \leftarrow_\$ \mathsf{Gens}(\mathbb{G})^4$

Client C

Input: $\mathsf{pw}_{\mathsf{CS}} \in \mathbb{Z}_p$

$A \leftarrow g^r$

$C \leftarrow h^r \cdot g^{\mathsf{pw}}$

$\xrightarrow{\mathsf{C}, vk, A, B, C, D}$

$\xleftarrow{\mathsf{S}, E, F, G, I, J}$

$\xrightarrow{K, \sigma}$

$I' \leftarrow I / g^{\mathsf{pw}}$

Output: SK

Server S

Input: $\mathsf{pw}_{\mathsf{CS}} \in \mathbb{Z}_p$

$x_2, y_2, z_2, w_2, r_2 \leftarrow_\$ \mathbb{Z}_p$

$\alpha' \leftarrow \mathsf{H}(\mathsf{C}, vk, A, B, C)$

$E \leftarrow g_1^{x_2} g_2^{y_2} h^{z_2} (cd^{\alpha'})^{w_2}$

$F \leftarrow g_1^{r_2}$; $G \leftarrow g_2^{r_2}$; $I \leftarrow h^{r_2} g_1^{\mathsf{pw}_{\mathsf{CS}}}$

$\gamma \leftarrow \mathsf{H}(\mathsf{S}, E, F, G, I)$; $J \leftarrow (cd^{\gamma})^{r_2}$

If $\mathsf{Sig.Vfy}(vk, (\gamma, K), \sigma) = 0$

$\quad \mathsf{SK}' \leftarrow_\$ \mathbb{G}$

Else:

$\quad C' \leftarrow C / g_1^{\mathsf{pw}_{\mathsf{CS}}}$

$\quad \mathsf{SK}' \leftarrow K^{r_2} A^{x_2} B^{y_2} (C')^{z_2} D^{w_2}$

Output: SK'

Where is the password actually used?

Notice: To break PAKE security, it suffices to learn g^{pw}.

The KOY protocol [Katz, Ostrovsky, Yung 09]

Public Parameters: $(g_2, h, c, d) \leftarrow_{\$} \mathsf{Gens}(\mathbb{G})^4$

Client C		**Server S**
Input: $\mathsf{pw}_{\mathsf{CS}} \in \mathbb{Z}_p$		Input: $\mathsf{pw}_{\mathsf{CS}} \in \mathbb{Z}_p$
$A \leftarrow g^r$ $C \leftarrow h^r \cdot g^{\mathsf{pw}}$	$\xrightarrow{\mathsf{C}, vk, A, B, C, D}$	$x_2, y_2, z_2, w_2, r_2 \leftarrow_{\$} \mathbb{Z}_p$ $\alpha' \leftarrow \mathsf{H}(\mathsf{C}, vk, A, B, C)$ $E \leftarrow g_1^{x_2} g_2^{y_2} h^{z_2} (cd^{\alpha'})^{w_2}$ $F \leftarrow g_1^{r_2}$; $G \leftarrow g_2^{r_2}$; $I \leftarrow h^{r_2} g_1^{\mathsf{pw}_{\mathsf{CS}}}$ $\gamma \leftarrow \mathsf{H}(\mathsf{S}, E, F, G, I)$; $J \leftarrow (cd^{\gamma})^{r_2}$
	$\xleftarrow{\mathsf{S}, E, F, G, I, J}$	
$I' \leftarrow I / g^{\mathsf{pw}}$	$\xrightarrow{K, \sigma}$	If $\mathsf{Sig.Vfy}(vk, (\gamma, K), \sigma) = 0$ $\mathsf{SK}' \leftarrow_{\$} \mathbb{G}$ Else: $C' \leftarrow C / g_1^{\mathsf{pw}_{\mathsf{CS}}}$ $\mathsf{SK}' \leftarrow K^{r_2} A^{x_2} B^{y_2} (C')^{z_2} D^{w_2}$
Output: SK		Output: SK'

Where is the password actually used?

Notice: To break PAKE security, it suffices to learn g^{pw}.

Claim: There is an adversary in the A-PAKE game with advantage close to 1.

Given input:

...

$\boxed{\beta = \log_g(h)}$

....

Passively observe a protocol execution.

Compute $C \cdot \boxed{A^{-\beta}}$.

This is: $h^r \cdot g^{\mathsf{pw}} \cdot g^{-r\beta}$
$= g^{\mathsf{pw}}$

Use g^{pw} to start a new session, and learn the session key! (Without pw.)

The KOY protocol [Katz, Ostrovsky, Yung 09]

Conclusion: KOY is A-INSECURE in Intermundium-DL.

Public Parameters: $(g_2, h, c, d) \leftarrow_\$ \mathsf{Gens}(\mathbb{G})^4$

Client C		**Server S**
Input: $\mathsf{pw}_{\mathsf{CS}} \in \mathbb{Z}_p$		Input: $\mathsf{pw}_{\mathsf{CS}} \in \mathbb{Z}_p$
$A \leftarrow g^r$		
$C \leftarrow h^r \cdot g^{\mathsf{pw}}$	$\xrightarrow{\mathsf{C}, vk, A, B, C, D}$	$x_2, y_2, z_2, w_2, r_2 \leftarrow_\$ \mathbb{Z}_p$ $\alpha' \leftarrow \mathsf{H}(\mathsf{C}, vk, A, B, C)$ $E \leftarrow g_1^{x_2} g_2^{y_2} h^{z_2} (cd^{\alpha'})^{w_2}$
	$\xleftarrow{\mathsf{S}, E, F, G, I, J}$	$F \leftarrow g_1^{r_2}$; $G \leftarrow g_2^{r_2}$; $I \leftarrow h^{r_2} g_1^{\mathsf{pw}_{\mathsf{CS}}}$ $\gamma \leftarrow \mathsf{H}(\mathsf{S}, E, F, G, I)$; $J \leftarrow (cd^{\gamma})^{r_2}$
$I' \leftarrow I / g^{\mathsf{pw}}$	$\xrightarrow{K, \sigma}$	If $\mathsf{Sig.Vfy}(vk, (\gamma, K), \sigma) = 0$ $\mathsf{SK}' \leftarrow_\$ \mathbb{G}$ Else: $C' \leftarrow C / g_1^{\mathsf{pw}_{\mathsf{CS}}}$ $\mathsf{SK}' \leftarrow K^{r_2} A^{x_2} B^{y_2} (C')^{z_2} D^{w_2}$
Output: SK		Output: SK'

Where is the password actually used?

Notice: To break PAKE security, it suffices to learn g^{pw}.

Claim: There is an adversary in the A-PAKE game with advantage close to 1.

Given input:

...

$\boxed{\beta = \log_g(h)}$

....

Passively observe a protocol execution.

Compute $C \cdot \boxed{A^{-\beta}}$.

This is: $h^r \cdot g^{\mathsf{pw}} \cdot g^{-r\beta}$
$= g^{\mathsf{pw}}$

Use g^{pw} to start a new session, and learn the session key! (Without pw.)

The SPAKE2 protocol

Public Parameters: $(M, N) \leftarrow_\$ \mathbb{G}^2$

Client C

Input: $\mathsf{pw}_{\mathsf{CS}} \in \mathbb{Z}_p$

$x \leftarrow_\$ \mathbb{Z}_p$

$X^* \leftarrow g^x \cdot M^{\mathsf{pw}_{\mathsf{CS}}}$

$$\xrightarrow{\mathsf{C}, X^*}$$

$$\xleftarrow{\mathsf{S}, Y^*}$$

$Y \leftarrow Y^*/N^{\mathsf{pw}_{\mathsf{CS}}}$; $Z \leftarrow Y^x$

$\mathsf{SK} \leftarrow \mathsf{H}(\mathsf{C}, \mathsf{S}, \mathsf{pw}_{\mathsf{CS}}, X^*, Y^*, Z)$

Output: SK

Server S

Input: $\mathsf{pw}_{\mathsf{CS}} \in \mathbb{Z}_p$

$y \leftarrow_\$ \mathbb{Z}_p$

$Y^* \leftarrow g^y \cdot N^{\mathsf{pw}_{\mathsf{CS}}}$

$X \leftarrow X^*/M^{\mathsf{pw}_{\mathsf{CS}}}$; $Z' \leftarrow X^y$

$\mathsf{SK}' \leftarrow \mathsf{H}(\mathsf{C}, \mathsf{S}, \mathsf{pw}_{\mathsf{CS}}, X^*, Y^*, Z')$

Output: SK'

The SPAKE2 protocol

Public Parameters: $(M, N) \leftarrow_{\$} \mathbb{G}^2$

Client C		**Server S**
Input: $\mathsf{pw}_{\mathsf{CS}} \in \mathbb{Z}_p$		Input: $\mathsf{pw}_{\mathsf{CS}} \in \mathbb{Z}_p$
$x \leftarrow_{\$} \mathbb{Z}_p$		$y \leftarrow_{\$} \mathbb{Z}_p$
$X^* \leftarrow g^x \cdot M^{\mathsf{pw}_{\mathsf{CS}}}$	$\xrightarrow{\mathsf{C}, X^*}$	$Y^* \leftarrow g^y \cdot N^{\mathsf{pw}_{\mathsf{CS}}}$
	$\xleftarrow{\mathsf{S}, Y^*}$	
$Y \leftarrow Y^*/N^{\mathsf{pw}_{\mathsf{CS}}}$; $Z \leftarrow Y^x$		$X \leftarrow X^*/M^{\mathsf{pw}_{\mathsf{CS}}}$; $Z' \leftarrow X^y$
$\mathsf{SK} \leftarrow \mathsf{H}(\mathsf{C}, \mathsf{S}, \mathsf{pw}_{\mathsf{CS}}, X^*, Y^*, Z)$		$\mathsf{SK}' \leftarrow \mathsf{H}(\mathsf{C}, \mathsf{S}, \mathsf{pw}_{\mathsf{CS}}, X^*, Y^*, Z')$
Output: SK		Output: SK'

- **SPAKE2** was proposed in 2005 [AP05]
- **Has a 2023 RFC**: *Given existing use of variants in Kerberos and other applications, it was felt that publication was beneficial.*
- Achieves **PAKE security under GapCDH** in game-based [AB19] and UC models [AB+20]

MIT Kerberos Documentation

CONTENTS | PREVIOUS | NEXT | INDEX | SEARCH | FEEDBACK

SPAKE Preauthentication

Stream:	
RFC:	9382
Category:	Informational
Published:	September 2023
ISSN:	2070-1721
Author:	W. Ladd *Akamai*

RFC 9382

SPAKE2, a Password-Authenticated Key Exchange

SPAKE2 in Intermundium-DL

Prior result [AB19]: SPAKE2 achieves **PAKE** security under GapCDH, assuming MEDIUM quality passwords.

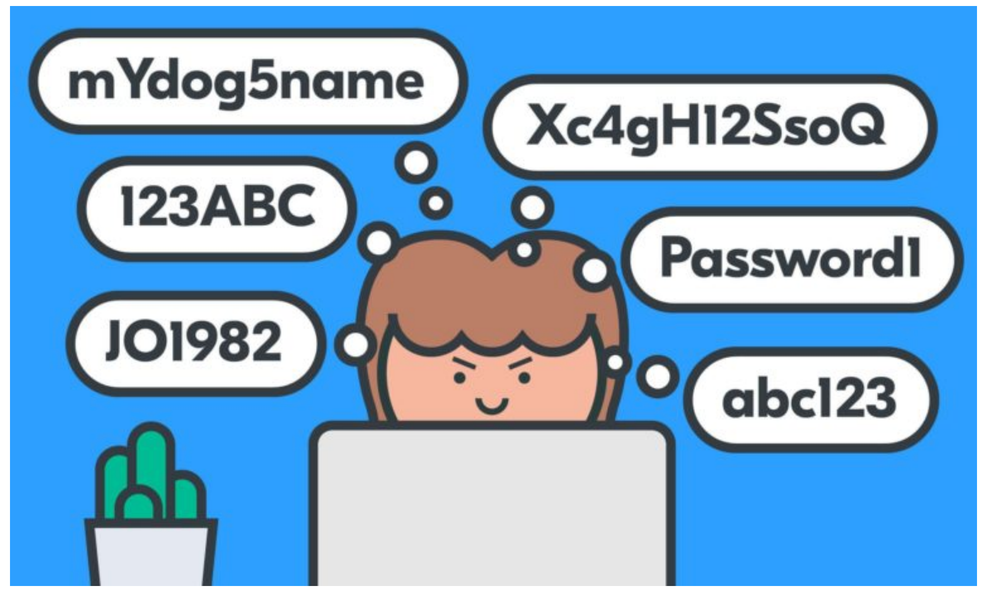
Our result: SPAKE2 achieves **A-PAKE** security under StrongCDH, assuming HIGH quality passwords.

SPAKE2 in Intermundium-DL

Prior result [AB19]: SPAKE2 achieves **PAKE** security under GapCDH, assuming <u>MEDIUM quality passwords</u>.

Our result: SPAKE2 achieves **A-PAKE** security under StrongCDH, assuming <u>HIGH quality passwords</u>.

Password strength	
LOW: Attackable with online queries	
MEDIUM: Attackable with offline queries; prohibitive online	
HIGH: Prohibitive offline and online	

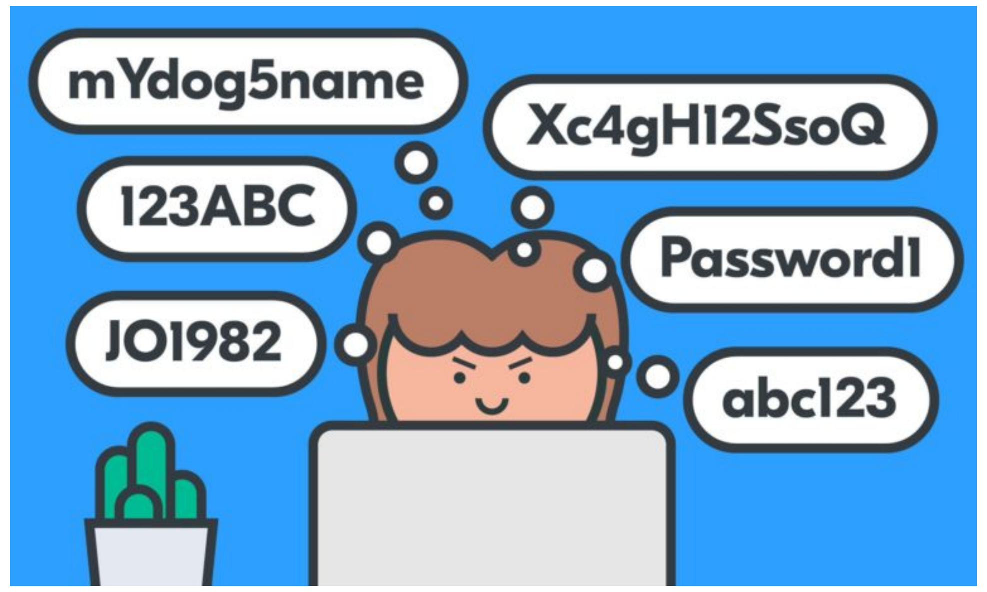


SPAKE2 in Intermundium-DL

Prior result [AB19]: SPAKE2 achieves **PAKE** security under GapCDH, assuming MEDIUM quality passwords.

Our result: SPAKE2 achieves **A-PAKE** security under StrongCDH, assuming HIGH quality passwords.

Password strength	Does SPAKE2 offer PAKE security?	Does SPAKE2 offer Advice-PAKE security?
LOW: Attackable with online queries		
MEDIUM: Attackable with offline queries; prohibitive online		
HIGH: Prohibitive offline and online		

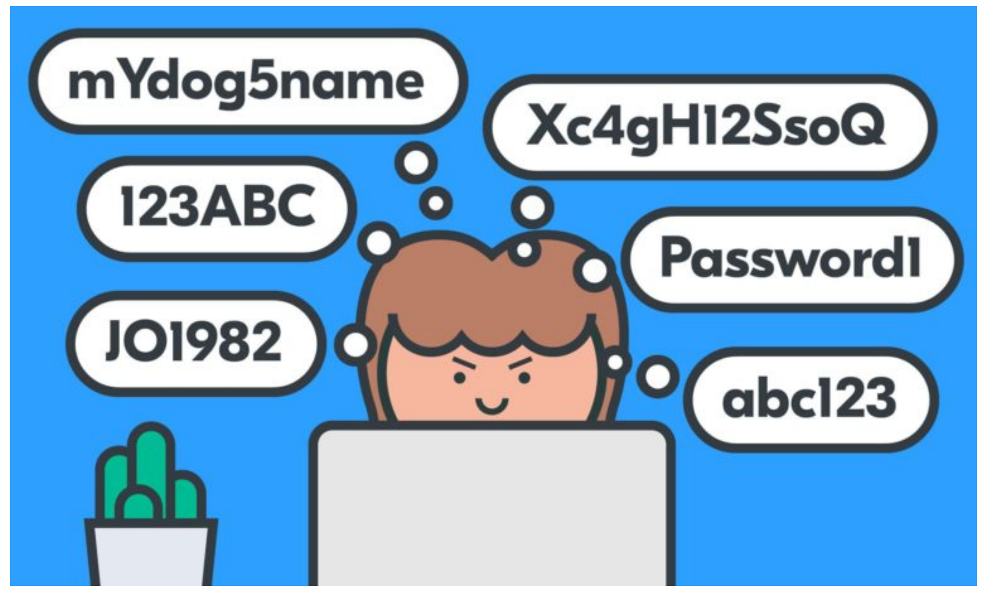


SPAKE2 in Intermundium-DL

Prior result [AB19]: SPAKE2 achieves **PAKE** security under GapCDH, assuming MEDIUM quality passwords.

Our result: SPAKE2 achieves **A-PAKE** security under StrongCDH, assuming HIGH quality passwords.

Password strength	Does SPAKE2 offer PAKE security?	Does SPAKE2 offer Advice-PAKE security?
LOW: Attackable with online queries	❌	
MEDIUM: Attackable with offline queries; prohibitive online	✅	
HIGH: Prohibitive offline and online	✅	

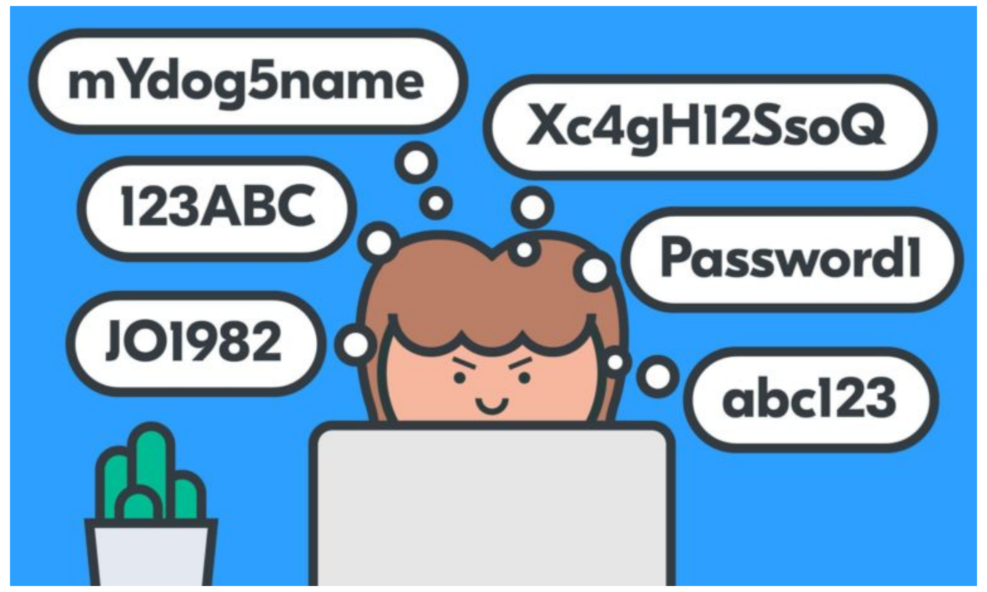


SPAKE2 in Intermundium-DL

Prior result [AB19]: SPAKE2 achieves **PAKE** security under GapCDH, assuming <u>MEDIUM quality passwords</u>.

Our result: SPAKE2 achieves **A-PAKE** security under StrongCDH, assuming <u>HIGH quality passwords</u>.

Password strength	Does SPAKE2 offer PAKE security?	Does SPAKE2 offer Advice-PAKE security?
LOW: Attackable with online queries	❌	❌
MEDIUM: Attackable with offline queries; prohibitive online	✅	❌
HIGH: Prohibitive offline and online	✅	✅

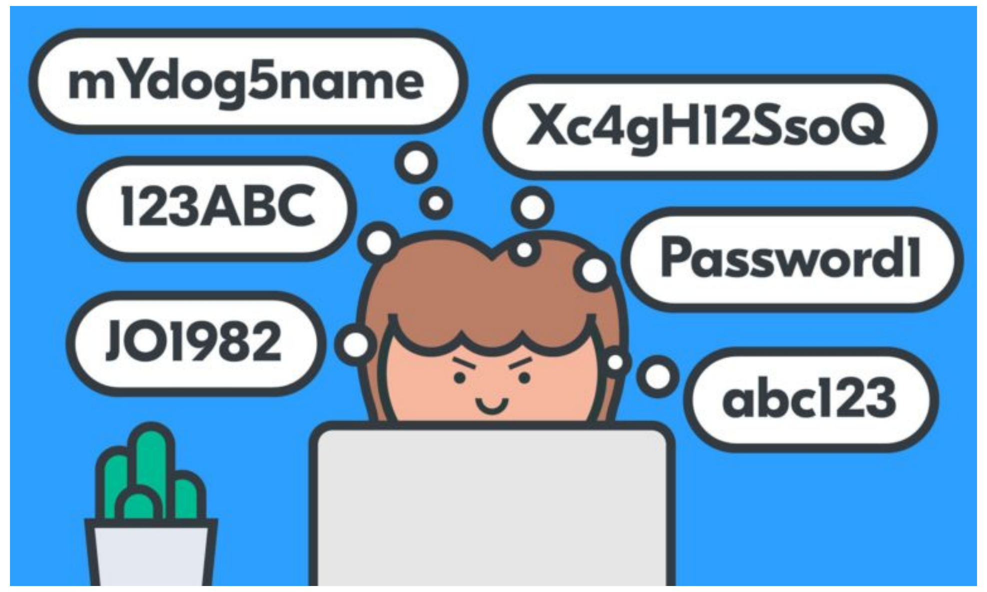


SPAKE2 in Intermundium-DL

Prior result [AB19]: SPAKE2 achieves **PAKE** security under GapCDH, assuming <u>MEDIUM quality passwords</u>.

Our result: SPAKE2 achieves **A-PAKE** security under StrongCDH, assuming <u>HIGH quality passwords</u>.

Password strength	Does SPAKE2 offer PAKE security?	Does SPAKE2 offer Advice-PAKE security?
LOW: Attackable with online queries	❌	❌
MEDIUM: Attackable with offline queries; prohibitive online	✅	❌
HIGH: Prohibitive offline and online	✅	✅



Many people do use HIGH quality passwords, and they retain security in Intermundium-DL.

Open questions

- **Some immediate questions:**

  Are there other positive results about Advice-Security?

  What about A-CCA2 of Cramer-Shoup? What about Okamoto-inspired recent signature schemes?

  Delayed-Target DDH: An interesting target for cryptanalysis.

Open questions

- **Some immediate questions:**

  Are there other positive results about Advice-Security?

  What about A-CCA2 of Cramer-Shoup? What about Okamoto-inspired recent signature schemes?

  Delayed-Target DDH: An interesting target for cryptanalysis.

- **If you are designing a new GEP scheme with trusted setup, perhaps check:** Is it necessary?

Open questions

- **Some immediate questions:**

  Are there other positive results about Advice-Security?

  What about A-CCA2 of Cramer-Shoup? What about Okamoto-inspired recent signature schemes?

  Delayed-Target DDH: An interesting target for cryptanalysis.

- **If you are designing a new GEP scheme with trusted setup, perhaps check:** Is it necessary?

- **Some questions about our model:**

  Our Advice-Security notion is pragmatic.

  But, it doesn't capture all the ways an attacker could utilize an expensive DL solver, nor all DL backdoors.

  So, is there a different way to model this?

INTERMUNDIUM-DL

A world in which
computing DLs in currently standardized groups is

Possible
but COSTLY

The adversary can compute a few discrete logarithms $\log_g(\,\cdot\,)$, but not many.

How might an adversary best exploit this capability?

Our Answer: Attack schemes whose public parameters $\pi = (h_1, \ldots, h_w)$ consist of a few group elements:

- Compute $\log_g(h_1), \ldots, \log_g(h_w)$
- Hope thereby to easily compromise security of MANY users

We accordingly investigate the security of current schemes in the setting where
the adversary knows $\log_g(h_1), \ldots, \log_g(h_w)$

The proofs typically assume that the adversary does NOT know these discrete logarithms.

So we might expect there to be attacks violating security in our setting.

However we find surprising variations in security across schemes:

- Some fully retain security
- Some retain partial but meaningful security
- Some do break totally

INTERMUNDIUM-DL

A world in which
computing DLs in currently standardized groups is

Possible
but COSTLY

The adversary can compute a few discrete logarithms $\log_g(\cdot)$, but not many.

How might an adversary best exploit this capability?

Our Answer: Attack schemes whose public parameters $\pi = (h_1, \ldots, h_w)$ consist of a few group elements:

- Compute $\log_g(h_1), \ldots, \log_g(h_w)$
- Hope thereby to easily compromise security of MANY users

We accordingly investigate the security of current schemes in the setting where

the adversary knows $\log_g(h_1), \ldots, \log_g(h_w)$

...pically assume that the adversary does ...now these discrete logarithms.

So we might expect there to be attacks violating security in our setting.

Thanks for listening!

Questions?

eprint 2025 / 663

However we find surprising variations in security across schemes:

- Some fully retain security
- Some retain partial but meaningful security
- Some do break totally