On The Complexity of Compressing Obfuscation

Gilad Asharov, Naomi Ephraim,
Ilan Komargodski, and Rafael Pass

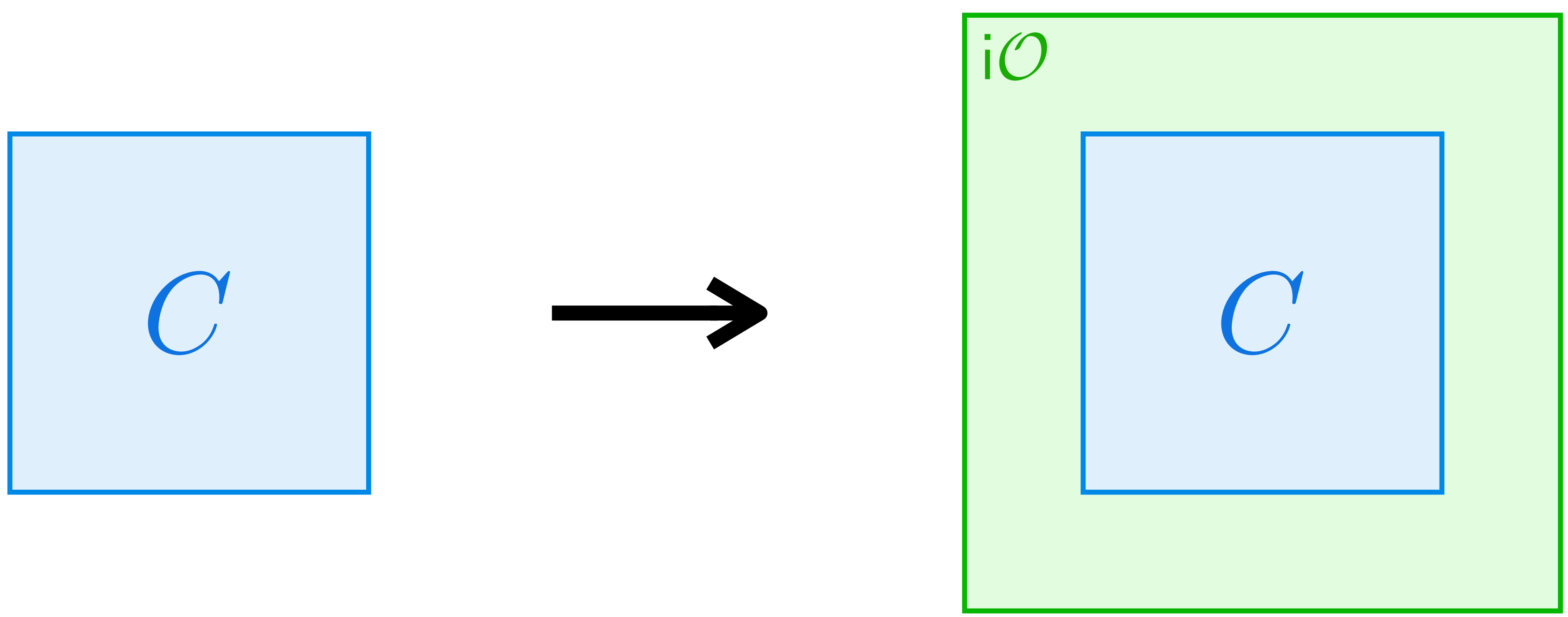
Cornell University and Cornell Tech

CRYPTO 2018

Indistinguishability Obfuscation (iO)

An obfuscator is a *compiler* which

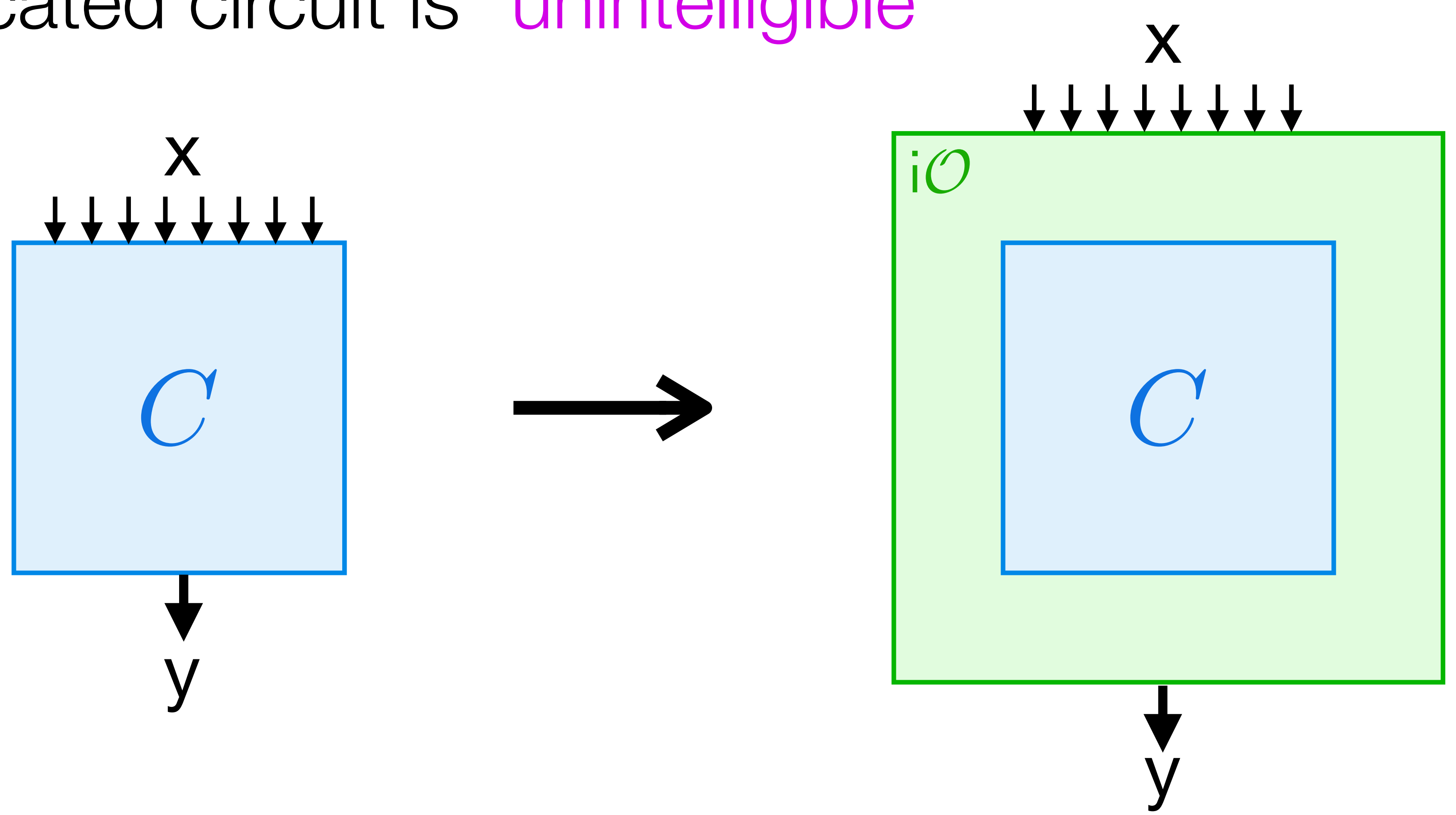
- preserves functionality
- obfuscated circuit is "unintelligible"



Indistinguishability Obfuscation (iO)

An obfuscator is a *compiler* which

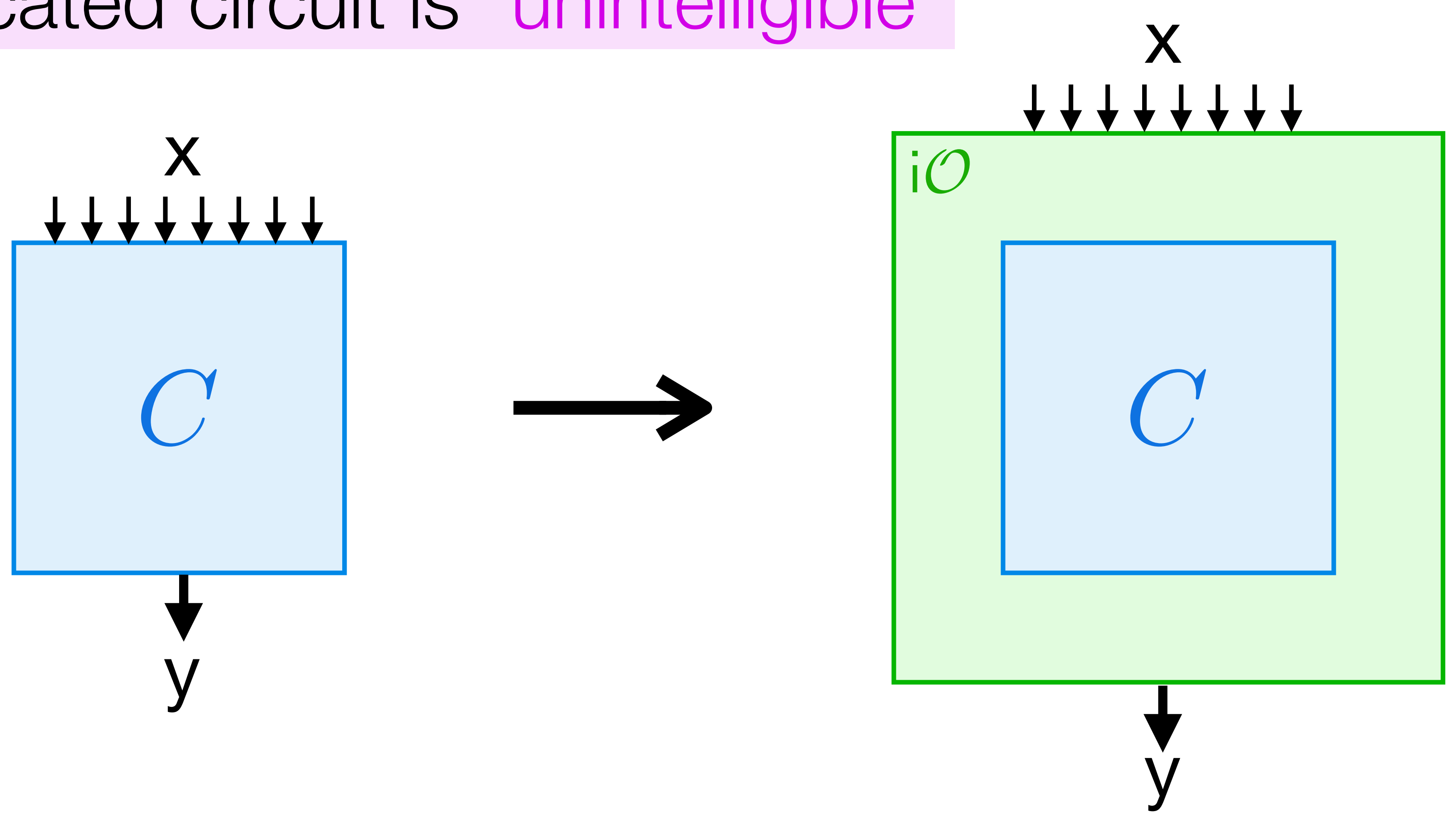
- preserves functionality
- obfuscated circuit is “unintelligible”



Indistinguishability Obfuscation (iO)

An obfuscator is a *compiler* which

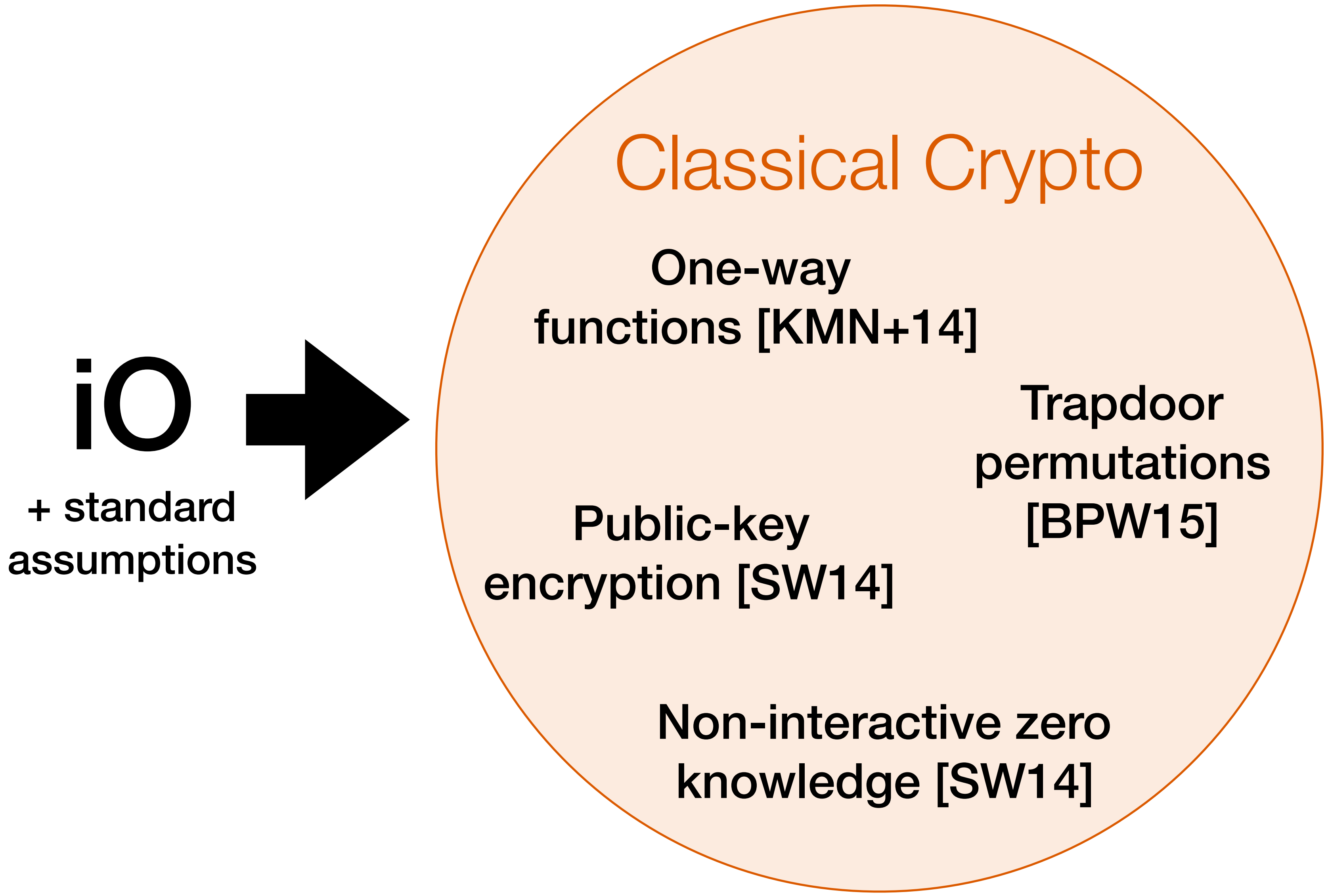
- preserves functionality
- obfuscated circuit is "unintelligible"



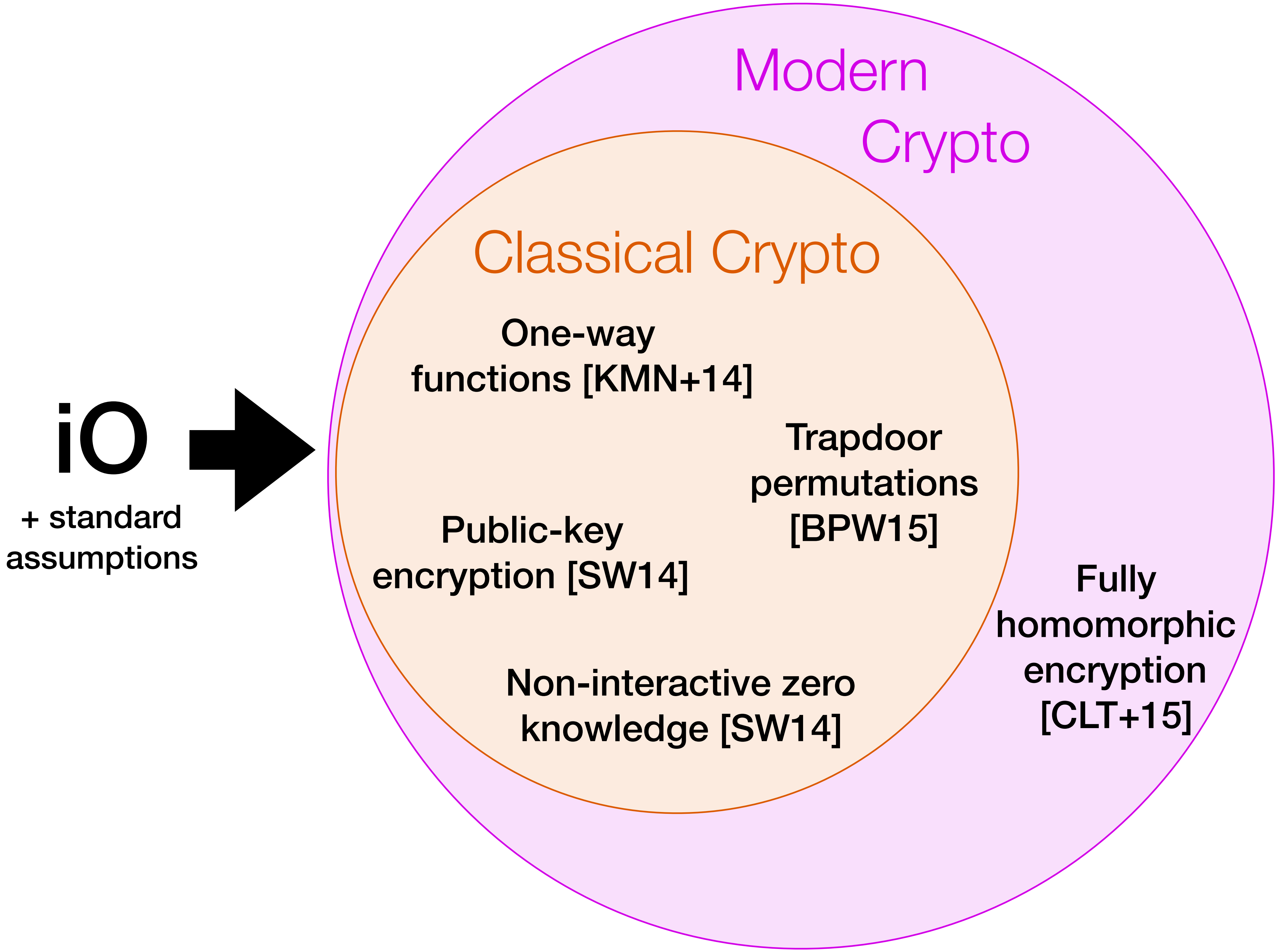
If C_0 and C_1 compute the same function and $|C_0|=|C_1|$, then $iO(C_0)$ and $iO(C_1)$ are hard to distinguish

Power of iO

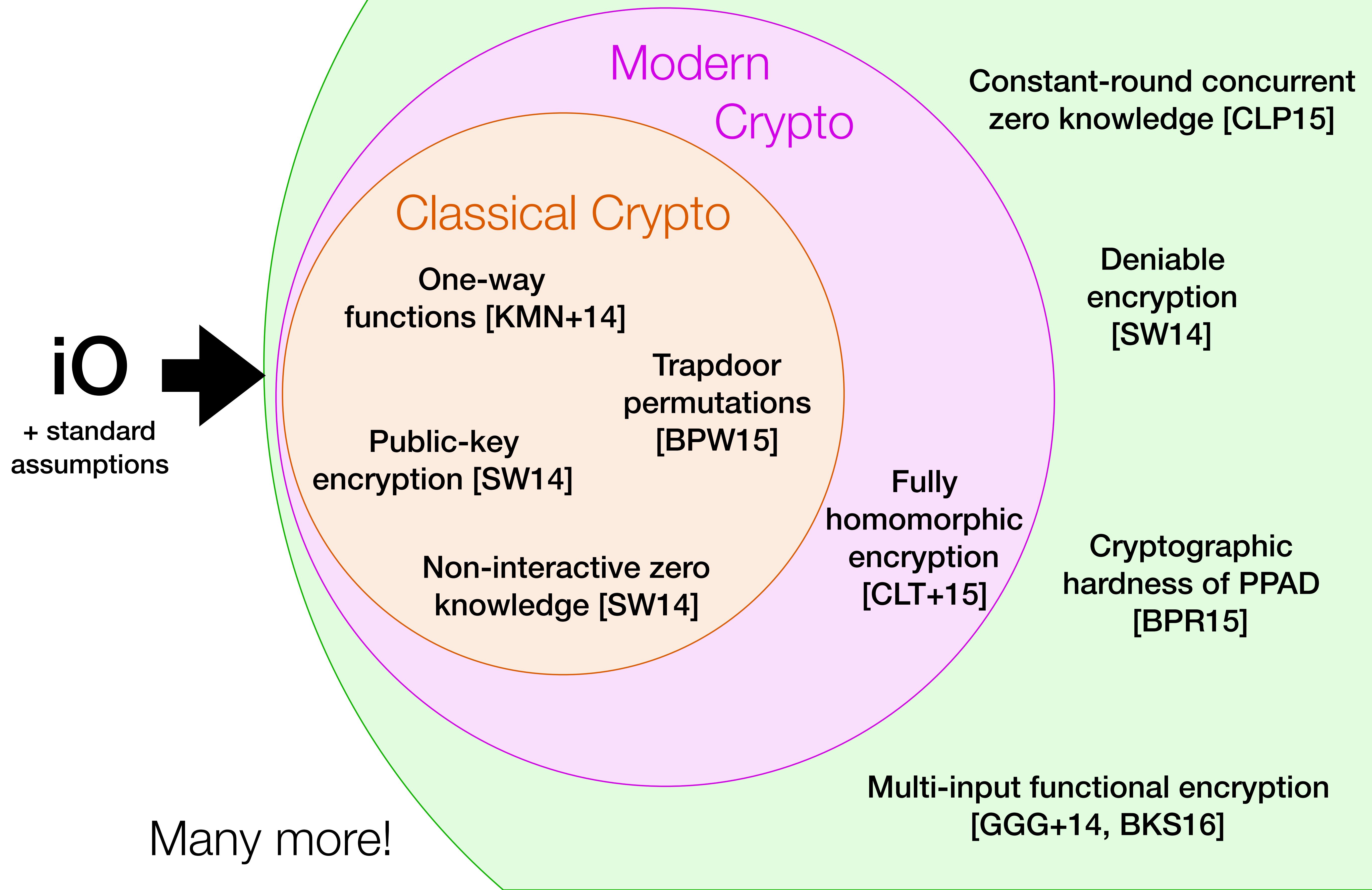
Power of iO


iO
+ standard
assumptions
Classical Crypto
One-way
functions [KMN+14]
Trapdoor
permutations
[BPW15]
Public-key
encryption [SW14]
Non-interactive zero
knowledge [SW14]

Power of iO


iO
+ standard assumptions
Modern Crypto
Classical Crypto
One-way functions [KMN+14]
Trapdoor permutations [BPW15]
Public-key encryption [SW14]
Non-interactive zero knowledge [SW14]
Fully homomorphic encryption [CLT+15]

Power of iO

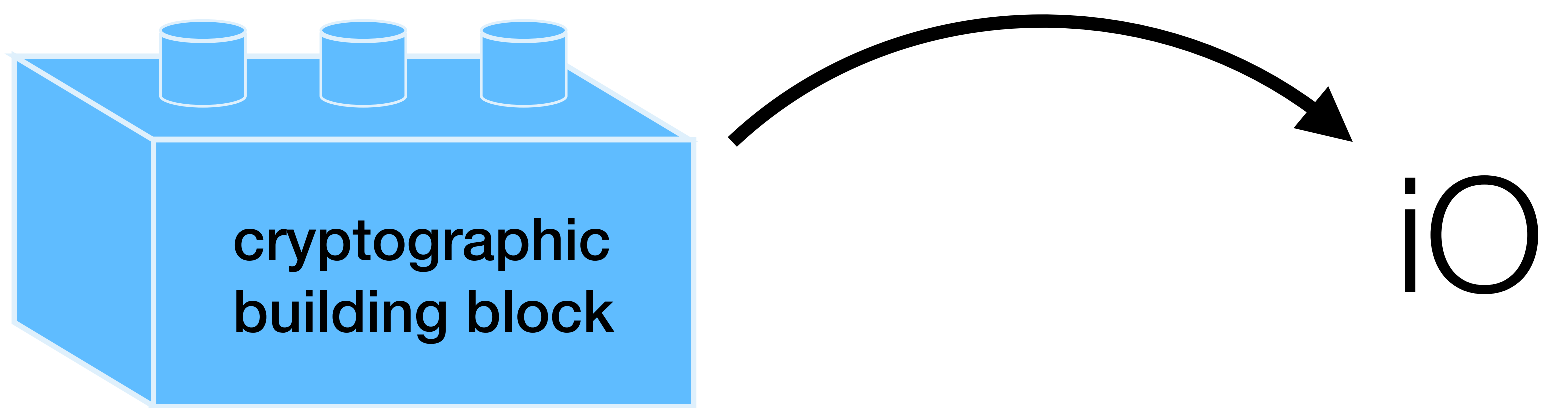

iO
+ standard assumptions
Modern Crypto
Classical Crypto
One-way functions [KMN+14]
Trapdoor permutations [BPW15]
Public-key encryption [SW14]
Non-interactive zero knowledge [SW14]
Fully homomorphic encryption [CLT+15]
Constant-round concurrent zero knowledge [CLP15]
Deniable encryption [SW14]
Cryptographic hardness of PPAD [BPR15]
Multi-input functional encryption [GGG+14, BKS16]
Many more!

Existence of iO

Reduce iO to seemingly weaker **building blocks**

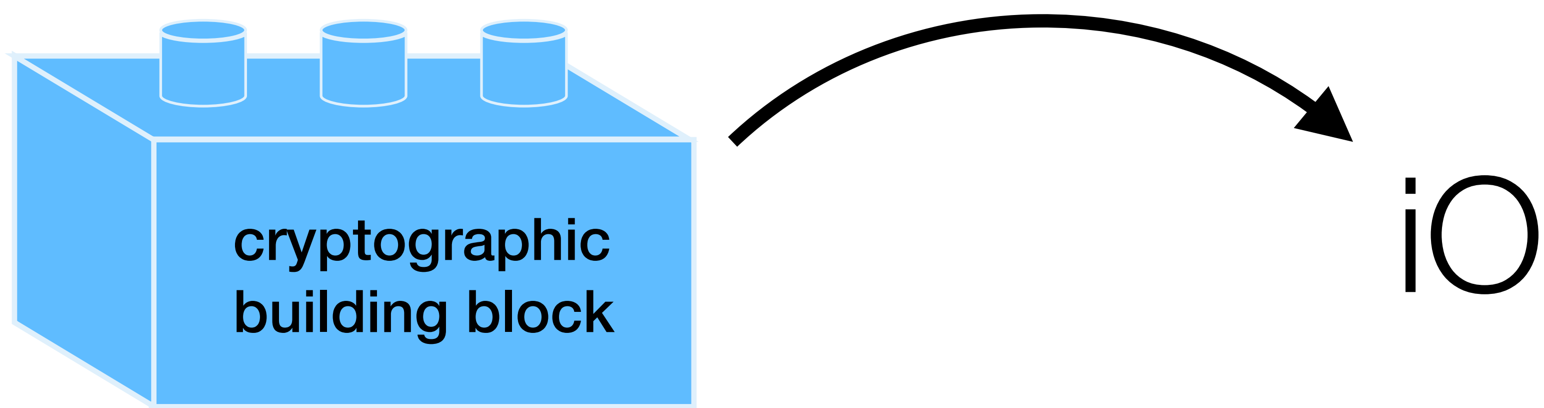
Existence of iO

Reduce iO to seemingly weaker **building blocks**



Existence of iO

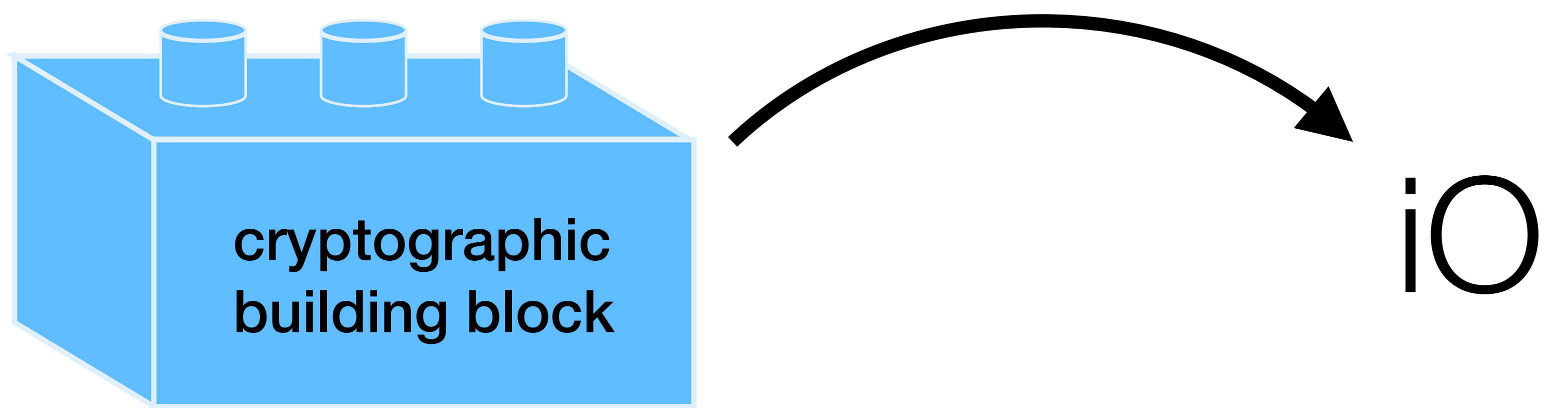
Reduce iO to seemingly weaker **building blocks**



Reduce the existence of iO to **new concrete assumptions**

Existence of iO

Reduce iO to seemingly weaker **building blocks**



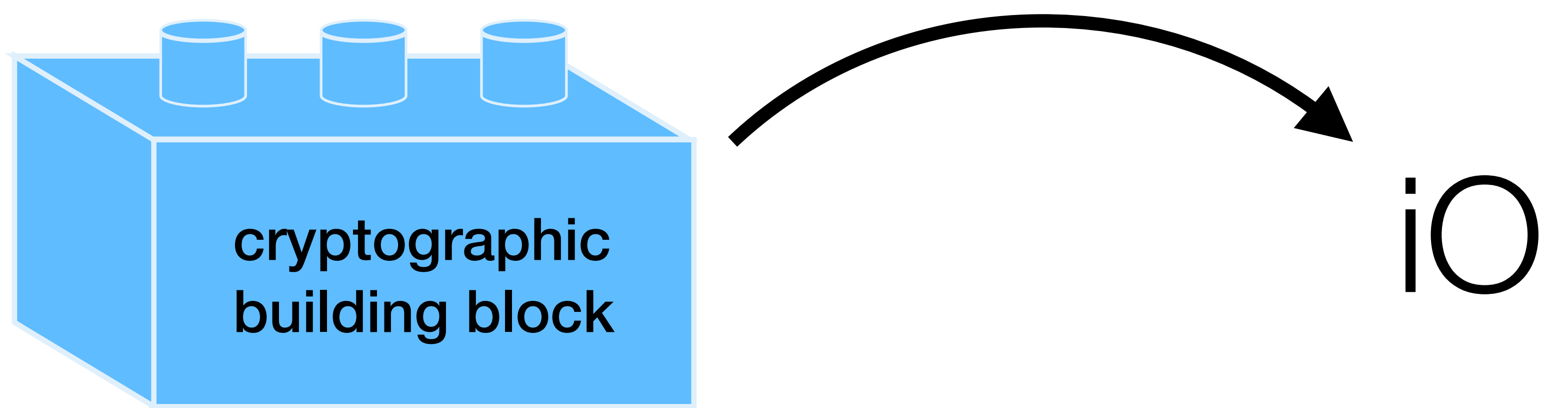
Reduce the existence of iO to **new concrete assumptions**

In all of these, the assumption is *nonstandard* and is vulnerable to attacks

[ADGM17,BBKK17,BWZ14,CGH17,CHLRS15,GHMS14,LV17,MSZ16]

Existence of iO

Reduce iO to seemingly weaker **building blocks**

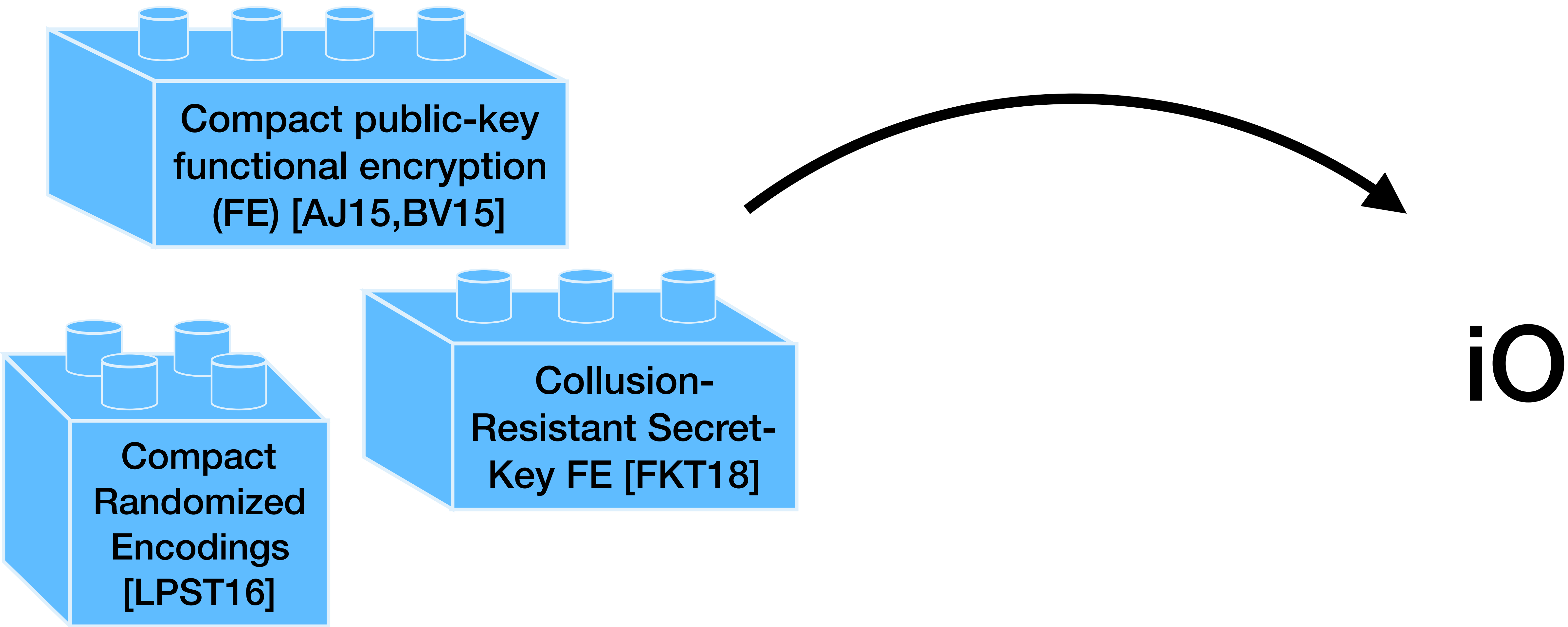


Reduce the existence of iO to **new concrete assumptions**

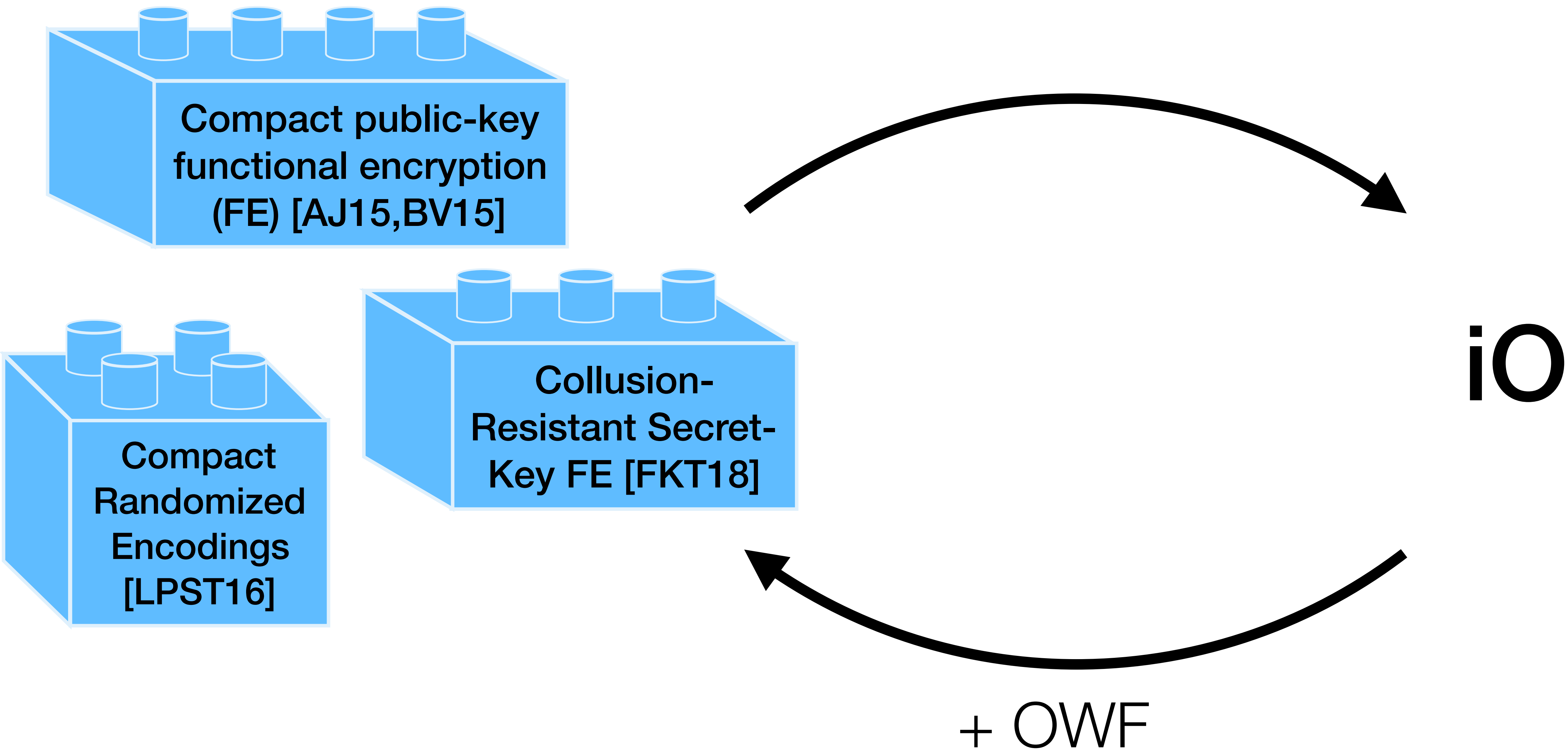
In all of these, the assumption is *nonstandard* and is vulnerable to attacks

[ADGM17,BBKK17,BWZ14,CGH17,CHLRS15,GHMS14,LV17,MSZ16]

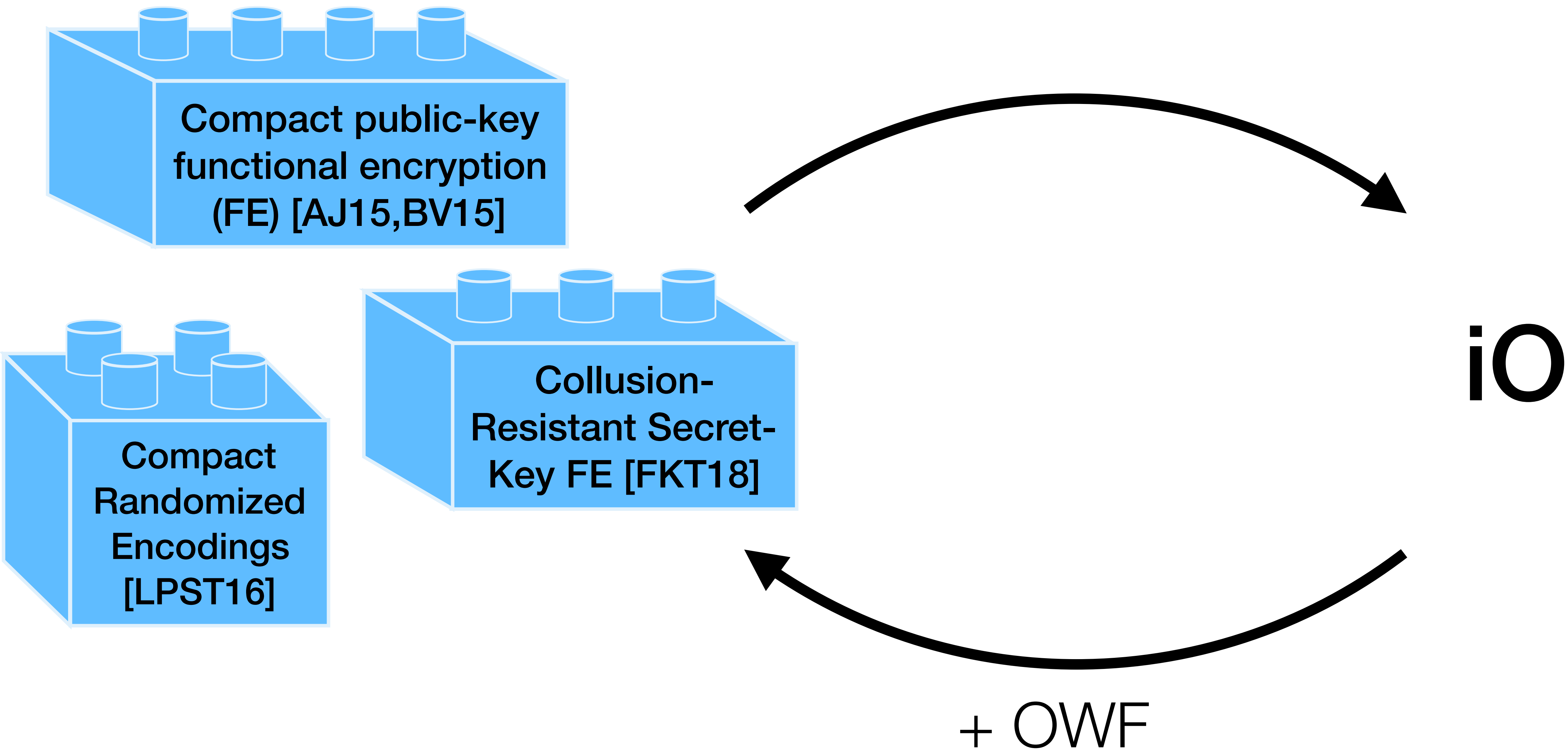
Existence of iO



Existence of iO

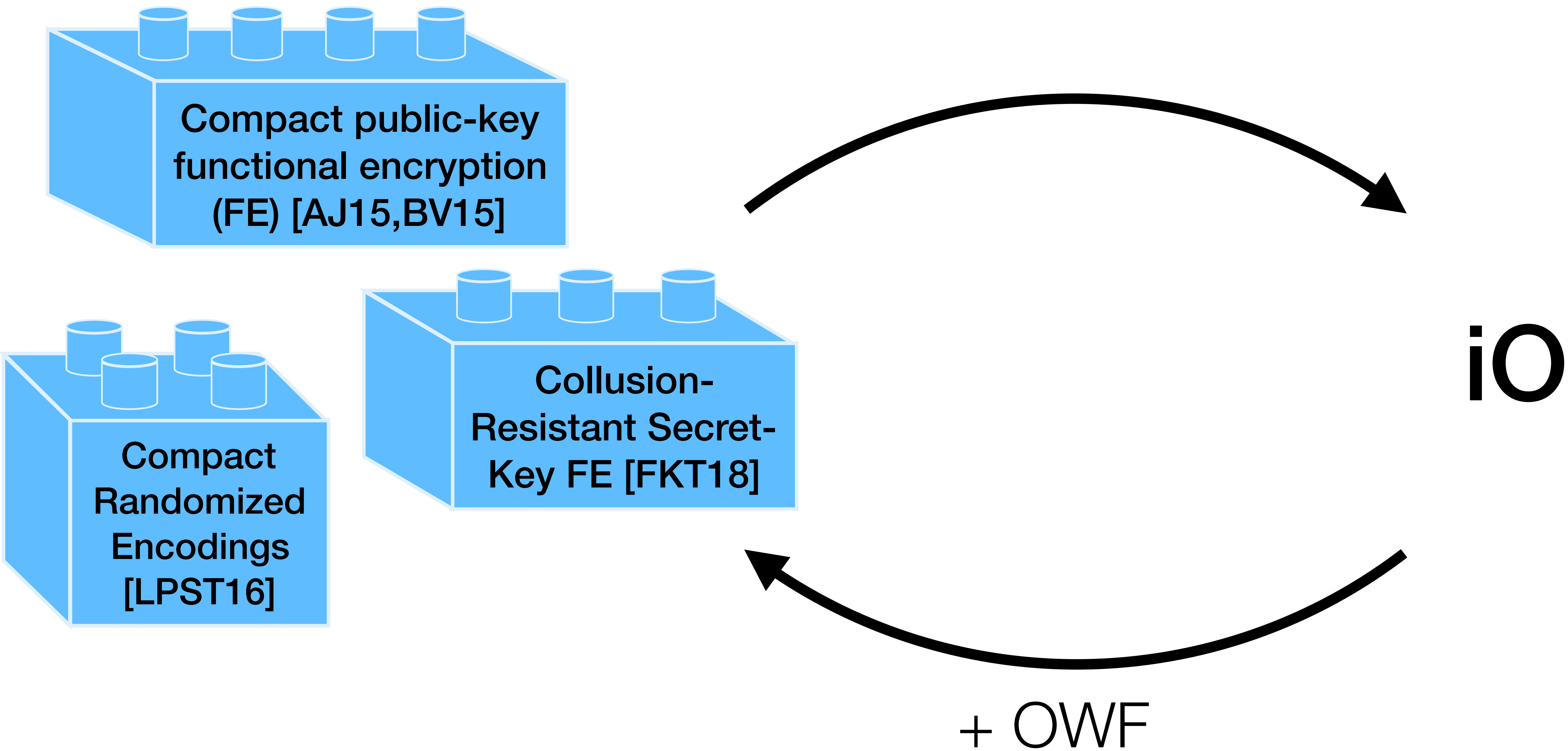


Existence of iO



What is the weakest building block that implies iO?

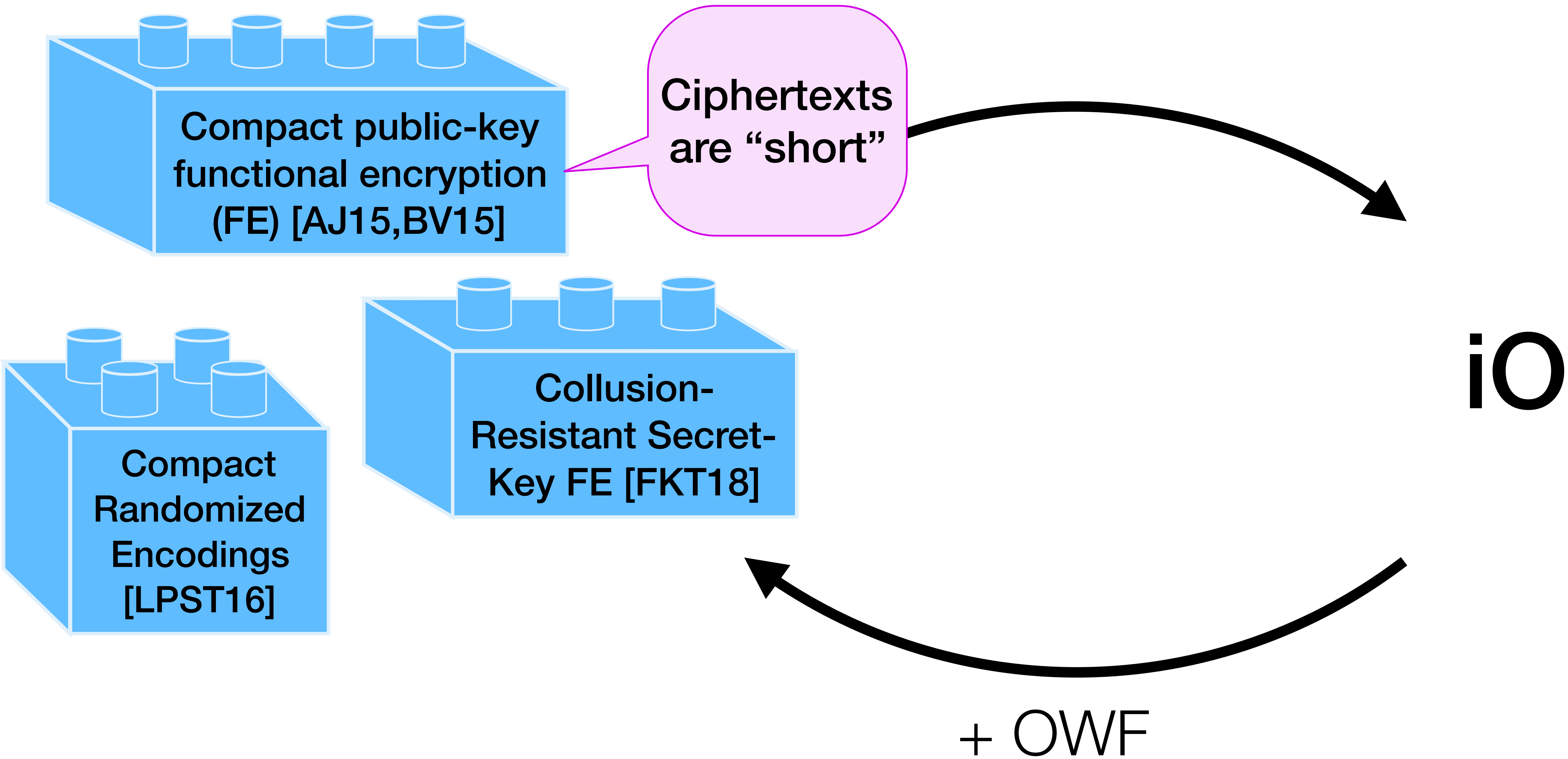
Existence of iO



What is the weakest building block that implies iO?

All building blocks require some form of **compression**

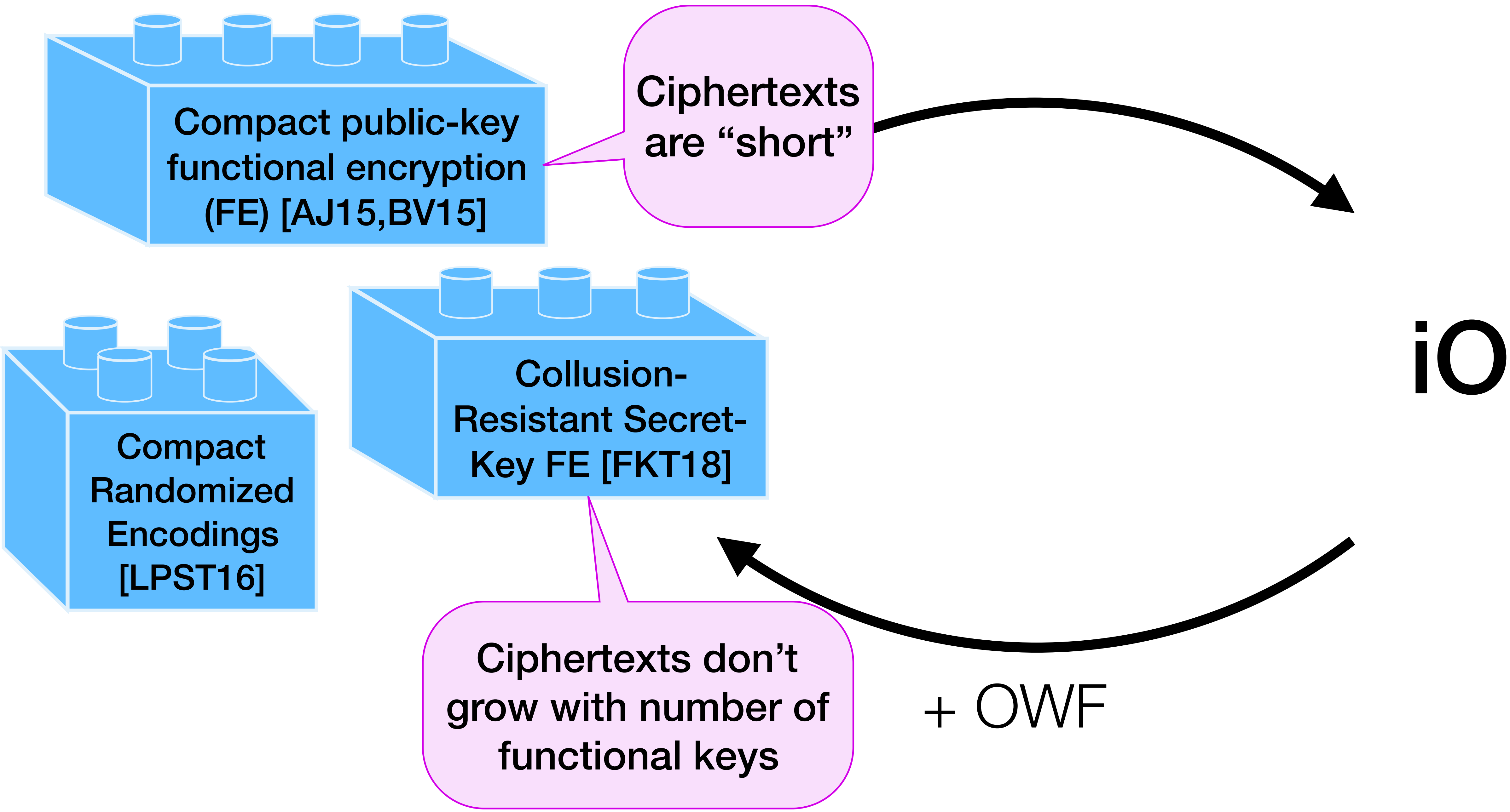
Existence of iO



What is the weakest building block that implies iO?

All building blocks require some form of **compression**

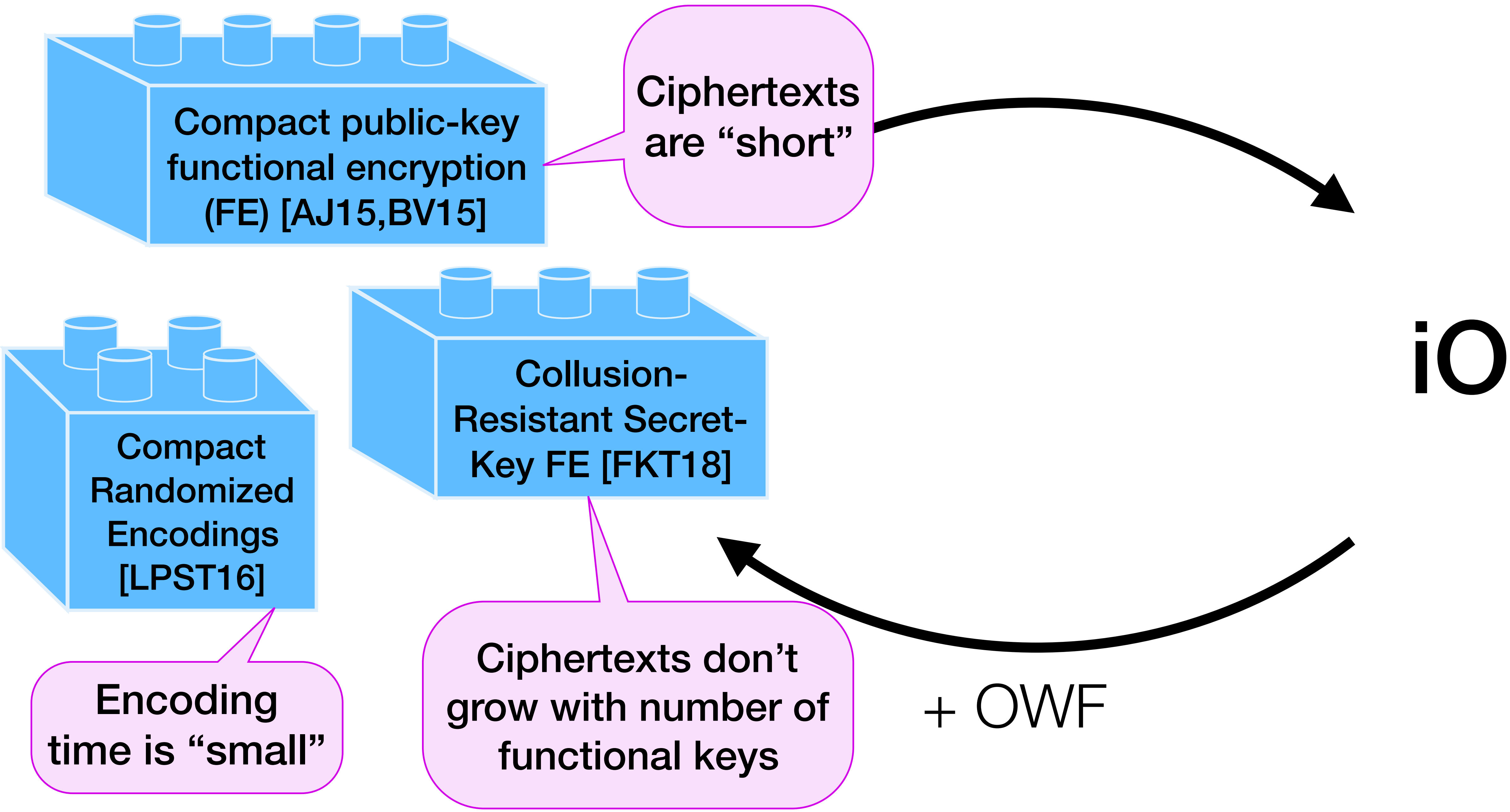
Existence of iO



What is the weakest building block that implies iO?

All building blocks require some form of compression

Existence of iO



What is the weakest building block that implies iO?

All building blocks require some form of compression

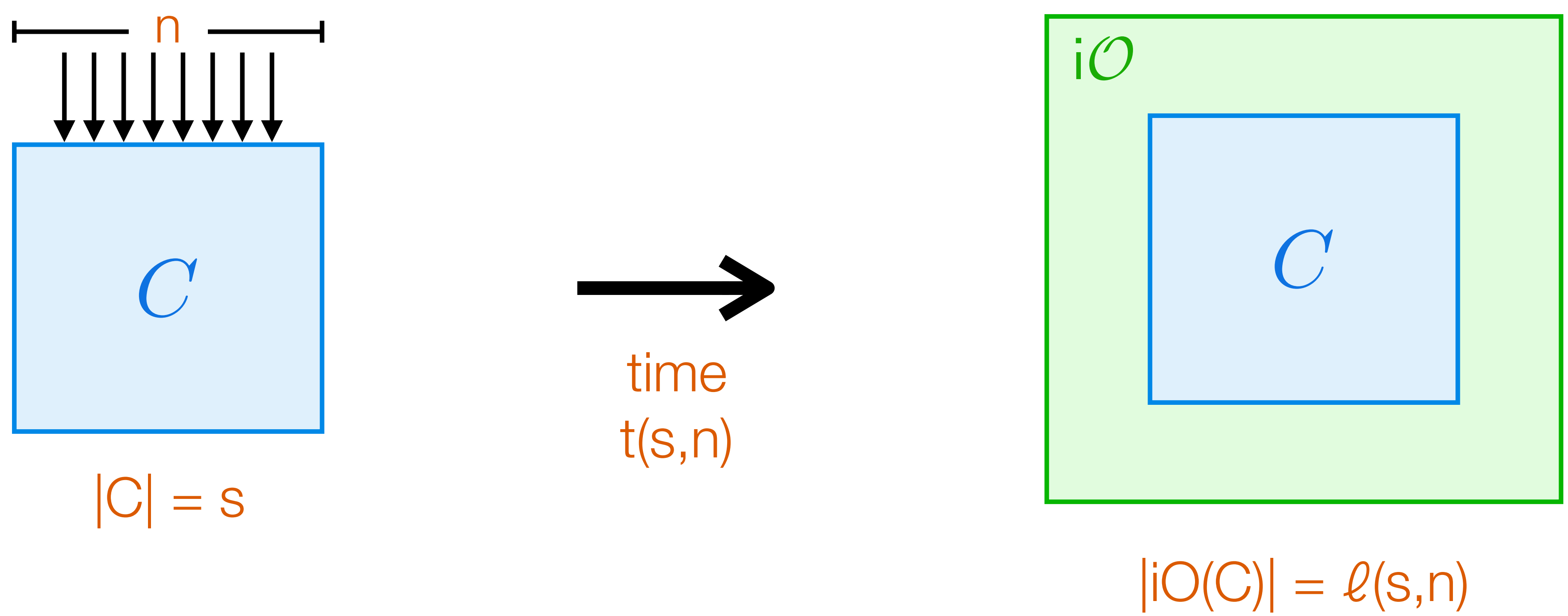
Compressing Obfuscation

A (t,ℓ)-compressing obfuscator has:

Time to obfuscate is t(s,n)

Size of the obfuscation is ℓ(s,n)

This talk: circuits C
- Size s
- input length n



Compressing Obfuscation

A (t,ℓ)-compressing obfuscator has:

Time to obfuscate is $t(s,n)$

Size of the obfuscation is $\ell(s,n)$

This talk: circuits C
- Size s
- input length n

$t(s,n)=$

$\ell(s,n)=$

Compressing Obfuscation

A (t,ℓ)-compressing obfuscator has:

Time to obfuscate is $t(s,n)$

Size of the obfuscation is $\ell(s,n)$

This talk: circuits C
- Size s
- input length n

iO

$t(s,n)=$ $\text{poly}(s)$

$\ell(s,n)=$ $\text{poly}(s)$

Compressing Obfuscation

A (t,ℓ)-compressing obfuscator has:

Time to obfuscate is t(s,n)

Size of the obfuscation is ℓ(s,n)

This talk: circuits C
- Size s
- input length n

	Trivial	iO
t(s,n)=	$2^n \cdot s$	$\mathrm{poly}(s)$
ℓ(s,n)=	2^n	$\mathrm{poly}(s)$

Compressing Obfuscation

A (t,ℓ)-compressing obfuscator has:

Time to obfuscate is t(s,n)

Size of the obfuscation is ℓ(s,n)

This talk: circuits C
- Size s
- input length n

time = |truth table|
size = smaller than truth table

	Trivial	XiO [LPST16]	iO
t(s,n)=	$2^n \cdot s$	$\text{poly}(2^n, s)$	$\text{poly}(s)$
ℓ(s,n)=	2^n	$2^{n(1-\epsilon)} \cdot \text{poly}(s)$	$\text{poly}(s)$

Compressing Obfuscation

A (t,ℓ)-compressing obfuscator has:

Time to obfuscate is t(s,n)

Size of the obfuscation is ℓ(s,n)

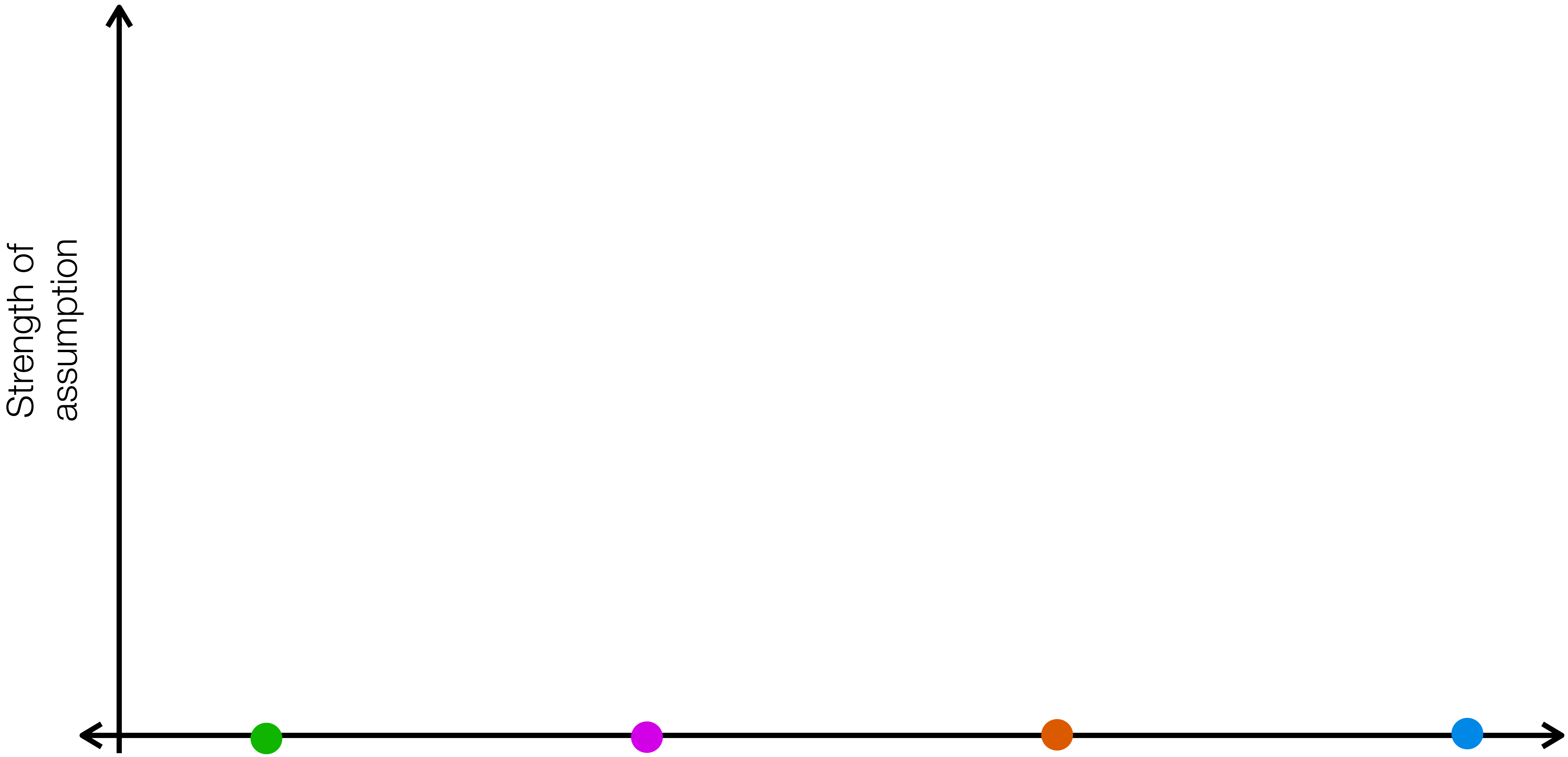
This talk: circuits C
- Size s
- input length n

time = |truth table|
size = smaller than truth table

time and **size** smaller than truth table

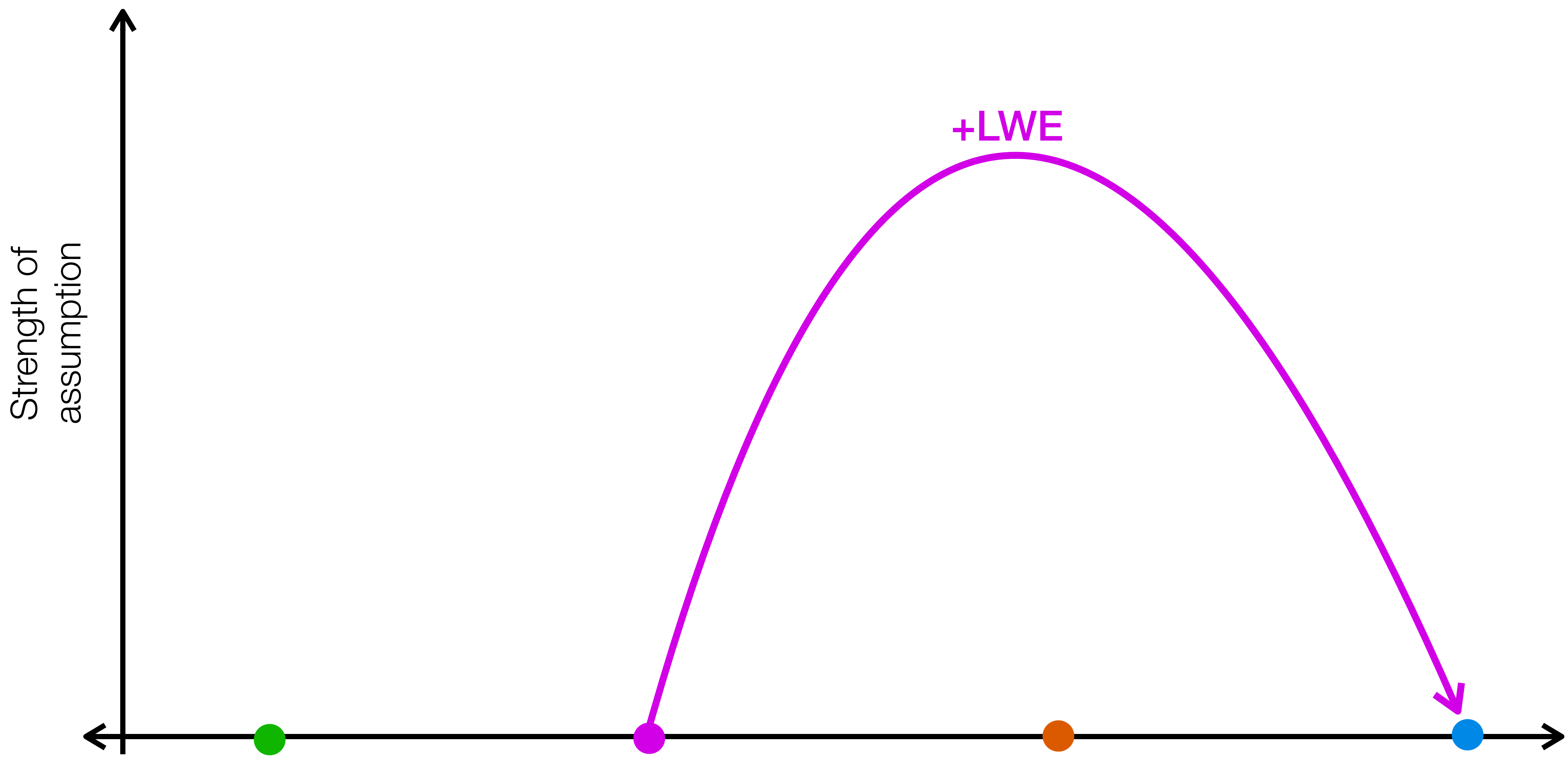
	Trivial	XiO [LPST16]	SXiO [BNPW16]	iO
t(s,n)=	$2^n \cdot s$	$\text{poly}(2^n, s)$	$2^{n(1-\epsilon)} \cdot \text{poly}(s)$	$\text{poly}(s)$
ℓ(s,n)=	2^n	$2^{n(1-\epsilon)} \cdot \text{poly}(s)$	$2^{n(1-\epsilon)} \cdot \text{poly}(s)$	$\text{poly}(s)$

Compression Hierarchy



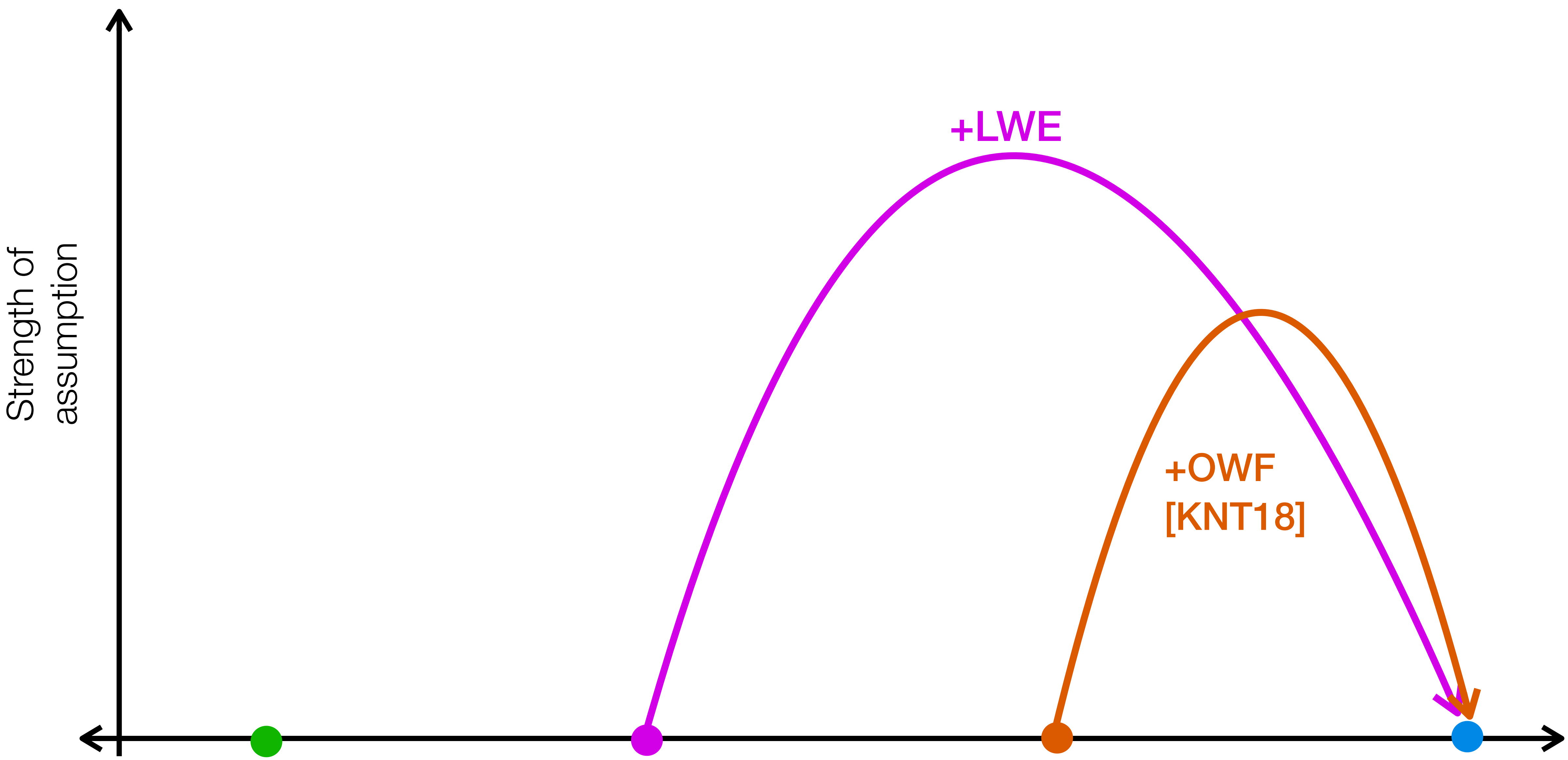
	Trivial	XiO [LPST16]	SXiO [BNPW16]	iO
t(s,n)=	$2^n \cdot s$	$\mathsf{poly}(2^n, s)$	$2^{n(1-\epsilon)} \cdot \mathsf{poly}(s)$	$\mathsf{poly}(s)$
ℓ(s,n)=	2^n	$2^{n(1-\epsilon)} \cdot \mathsf{poly}(s)$	$2^{n(1-\epsilon)} \cdot \mathsf{poly}(s)$	$\mathsf{poly}(s)$

Compression Hierarchy



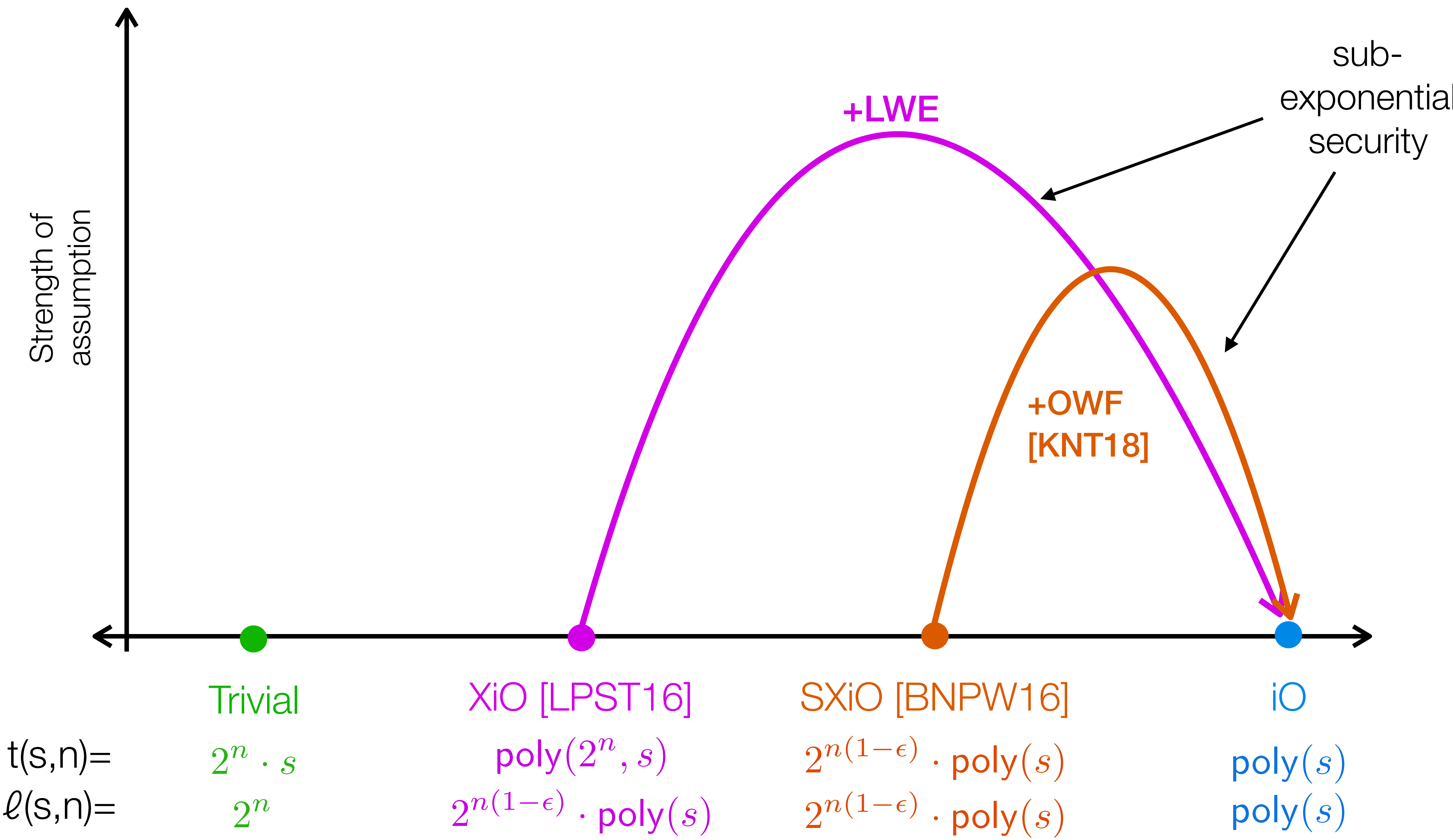
	Trivial	XiO [LPST16]	SXiO [BNPW16]	iO
t(s,n)=	$2^n \cdot s$	$\text{poly}(2^n, s)$	$2^{n(1-\epsilon)} \cdot \text{poly}(s)$	$\text{poly}(s)$
ℓ(s,n)=	2^n	$2^{n(1-\epsilon)} \cdot \text{poly}(s)$	$2^{n(1-\epsilon)} \cdot \text{poly}(s)$	$\text{poly}(s)$

Compression Hierarchy

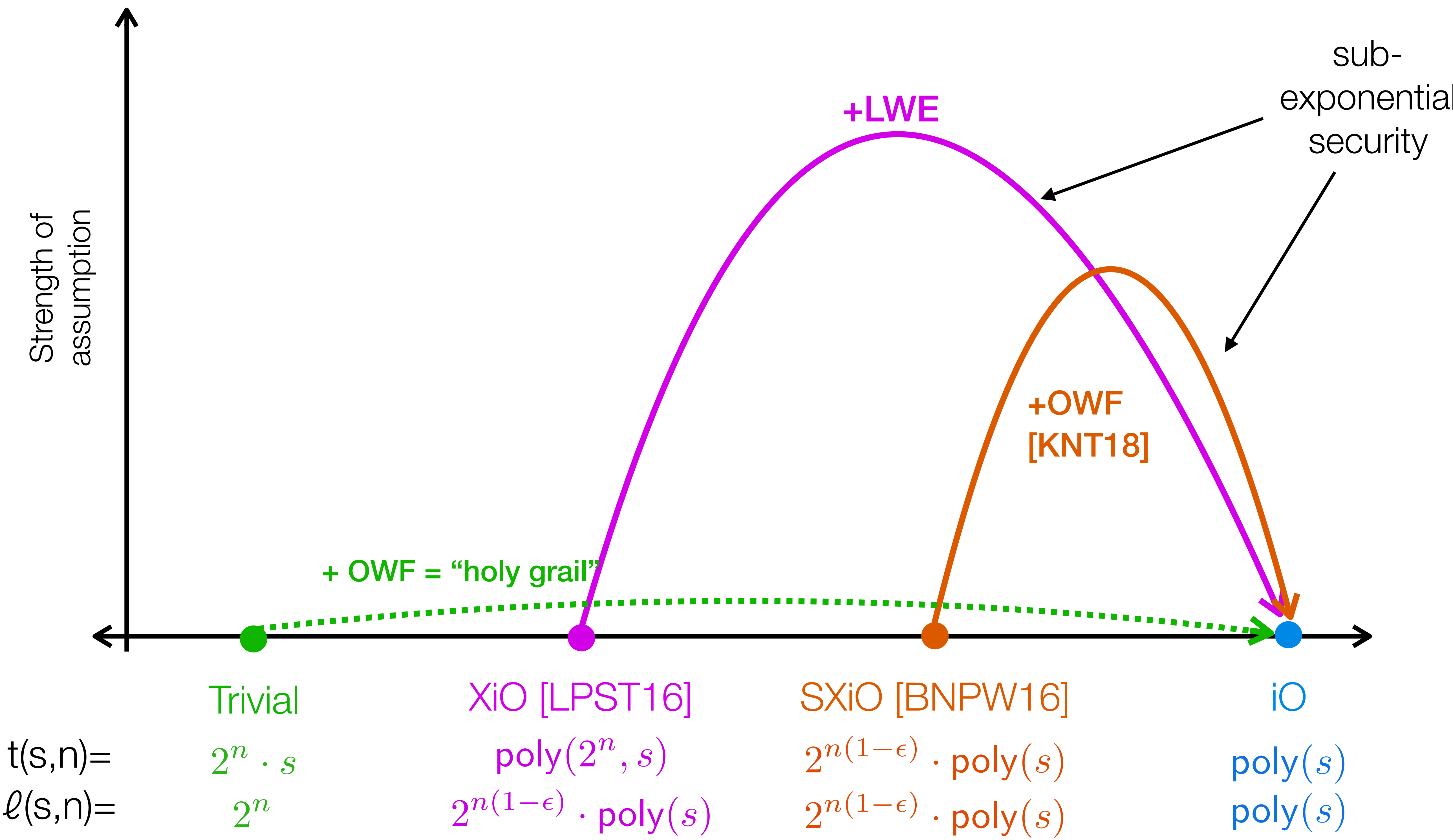


	Trivial	XiO [LPST16]	SXiO [BNPW16]	iO
t(s,n)=	$2^n \cdot s$	poly$(2^n, s)$	$2^{n(1-\epsilon)} \cdot$ poly(s)	poly(s)
ℓ(s,n)=	2^n	$2^{n(1-\epsilon)} \cdot$ poly(s)	$2^{n(1-\epsilon)} \cdot$ poly(s)	poly(s)

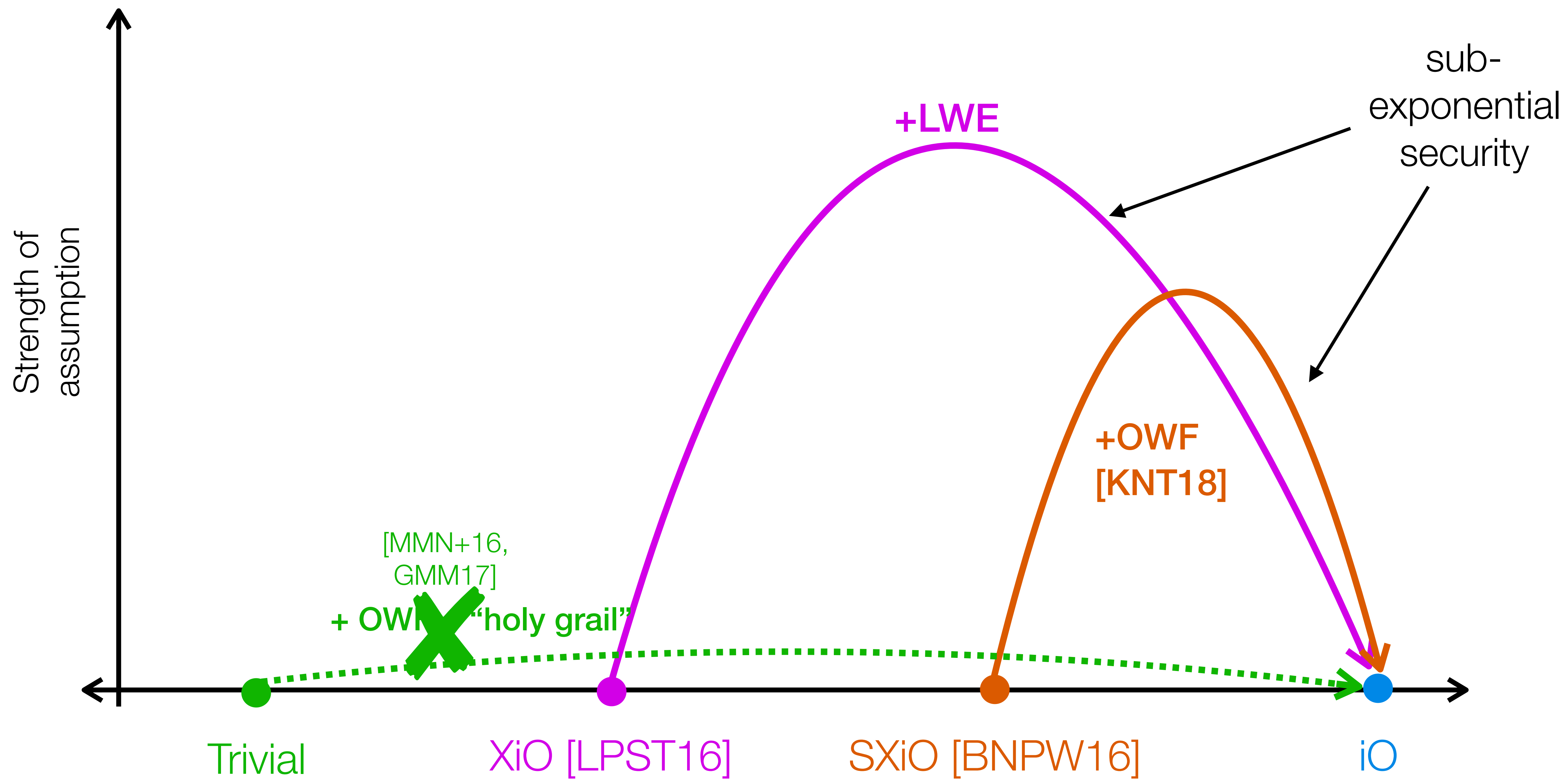
Compression Hierarchy



Compression Hierarchy

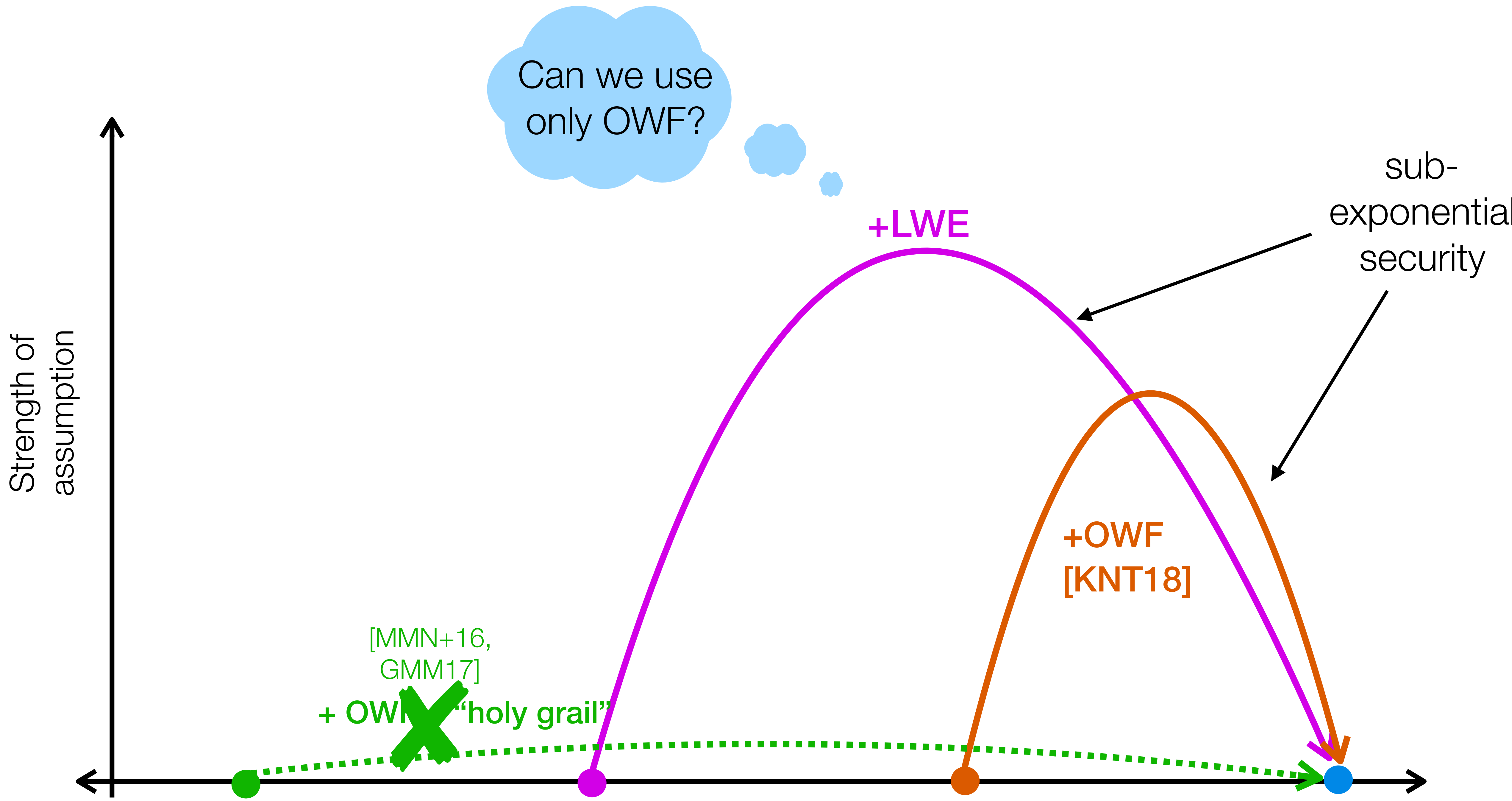


Compression Hierarchy



	Trivial	XiO [LPST16]	SXiO [BNPW16]	iO
t(s,n)=	$2^n \cdot s$	$\text{poly}(2^n, s)$	$2^{n(1-\epsilon)} \cdot \text{poly}(s)$	$\text{poly}(s)$
ℓ(s,n)=	2^n	$2^{n(1-\epsilon)} \cdot \text{poly}(s)$	$2^{n(1-\epsilon)} \cdot \text{poly}(s)$	$\text{poly}(s)$

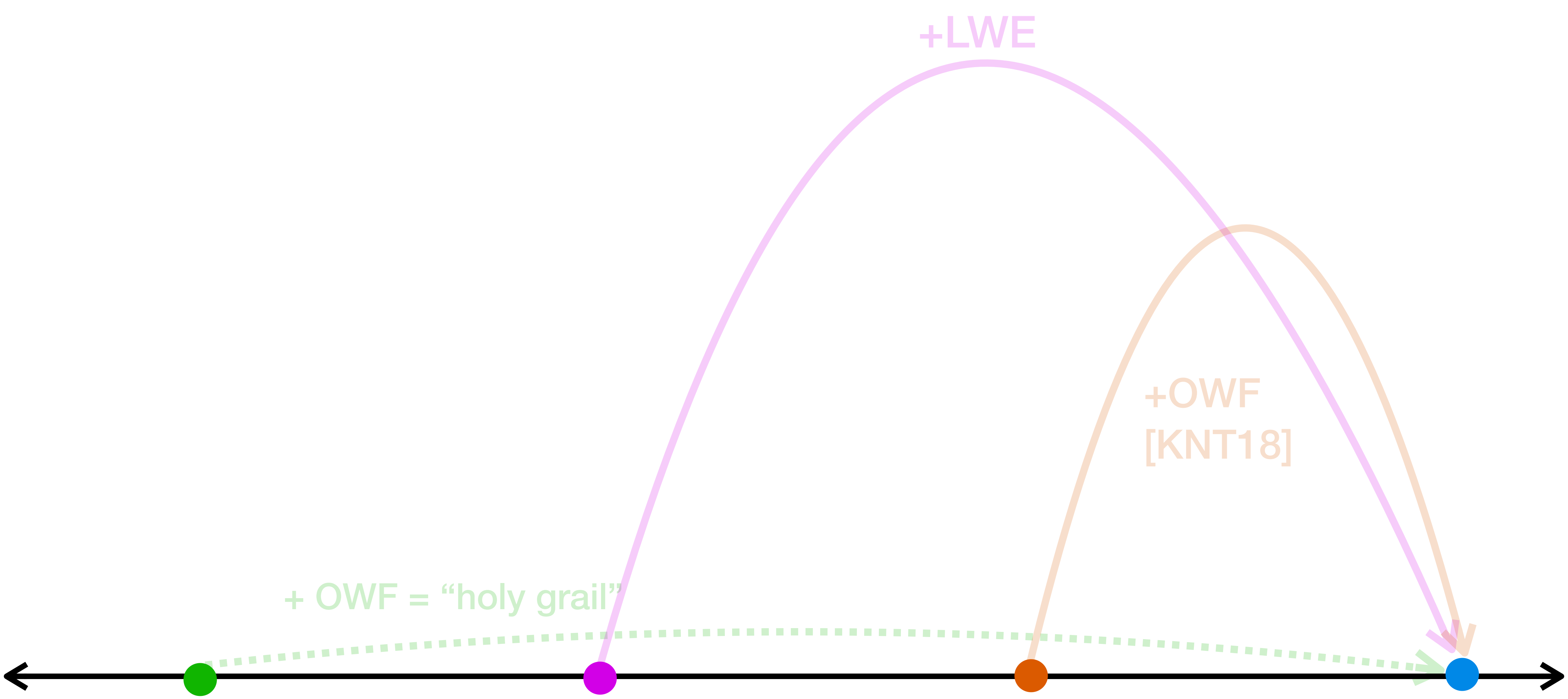
Compression Hierarchy



	Trivial	XiO [LPST16]	SXiO [BNPW16]	iO
t(s,n)=	$2^n \cdot s$	$\text{poly}(2^n, s)$	$2^{n(1-\epsilon)} \cdot \text{poly}(s)$	$\text{poly}(s)$
ℓ(s,n)=	2^n	$2^{n(1-\epsilon)} \cdot \text{poly}(s)$	$2^{n(1-\epsilon)} \cdot \text{poly}(s)$	$\text{poly}(s)$

Compression Hierarchy

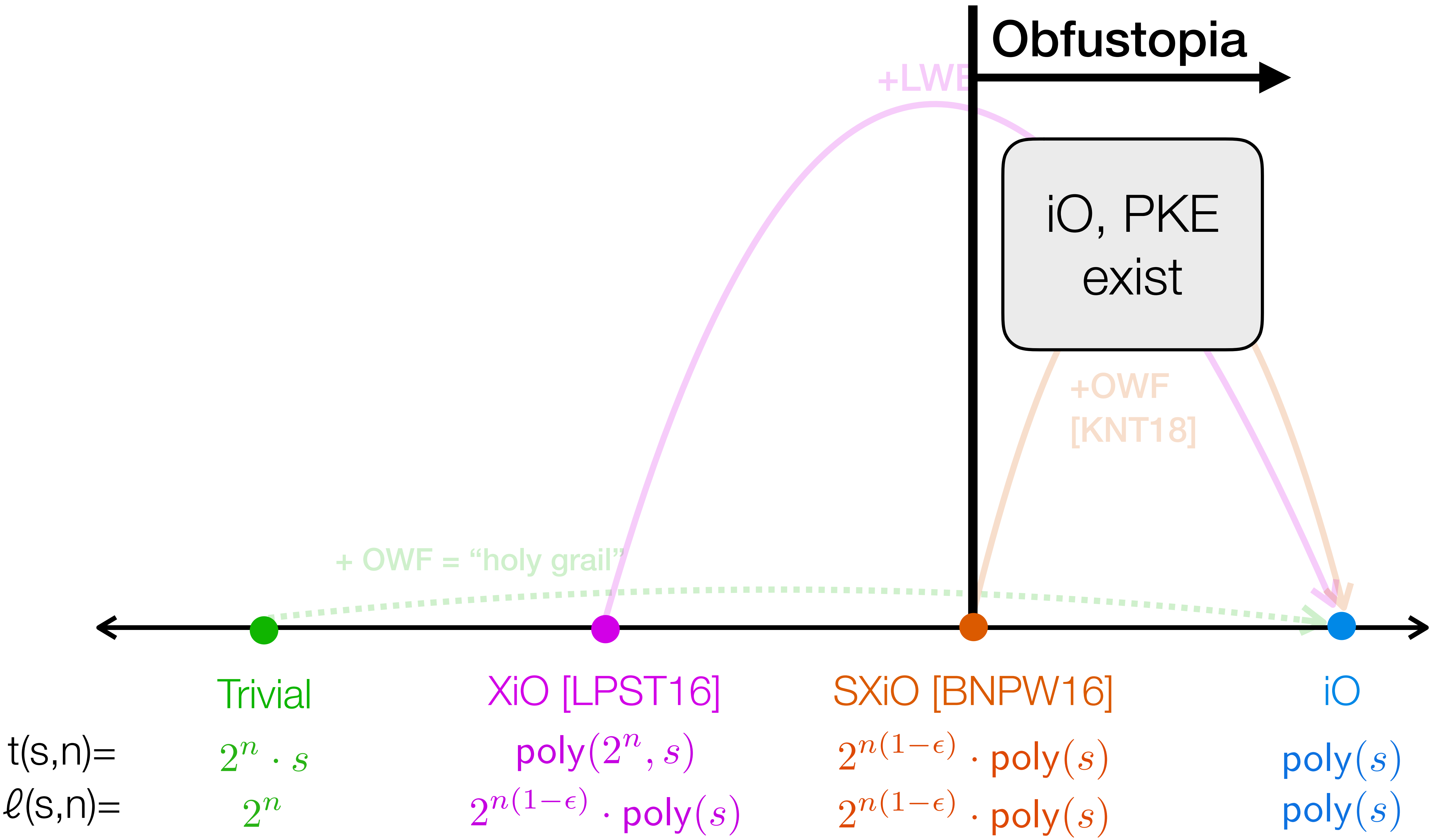
Assume sub-exponential OWF



	Trivial	XiO [LPST16]	SXiO [BNPW16]	iO
t(s,n)=	$2^n \cdot s$	$\text{poly}(2^n, s)$	$2^{n(1-\epsilon)} \cdot \text{poly}(s)$	$\text{poly}(s)$
ℓ(s,n)=	2^n	$2^{n(1-\epsilon)} \cdot \text{poly}(s)$	$2^{n(1-\epsilon)} \cdot \text{poly}(s)$	$\text{poly}(s)$

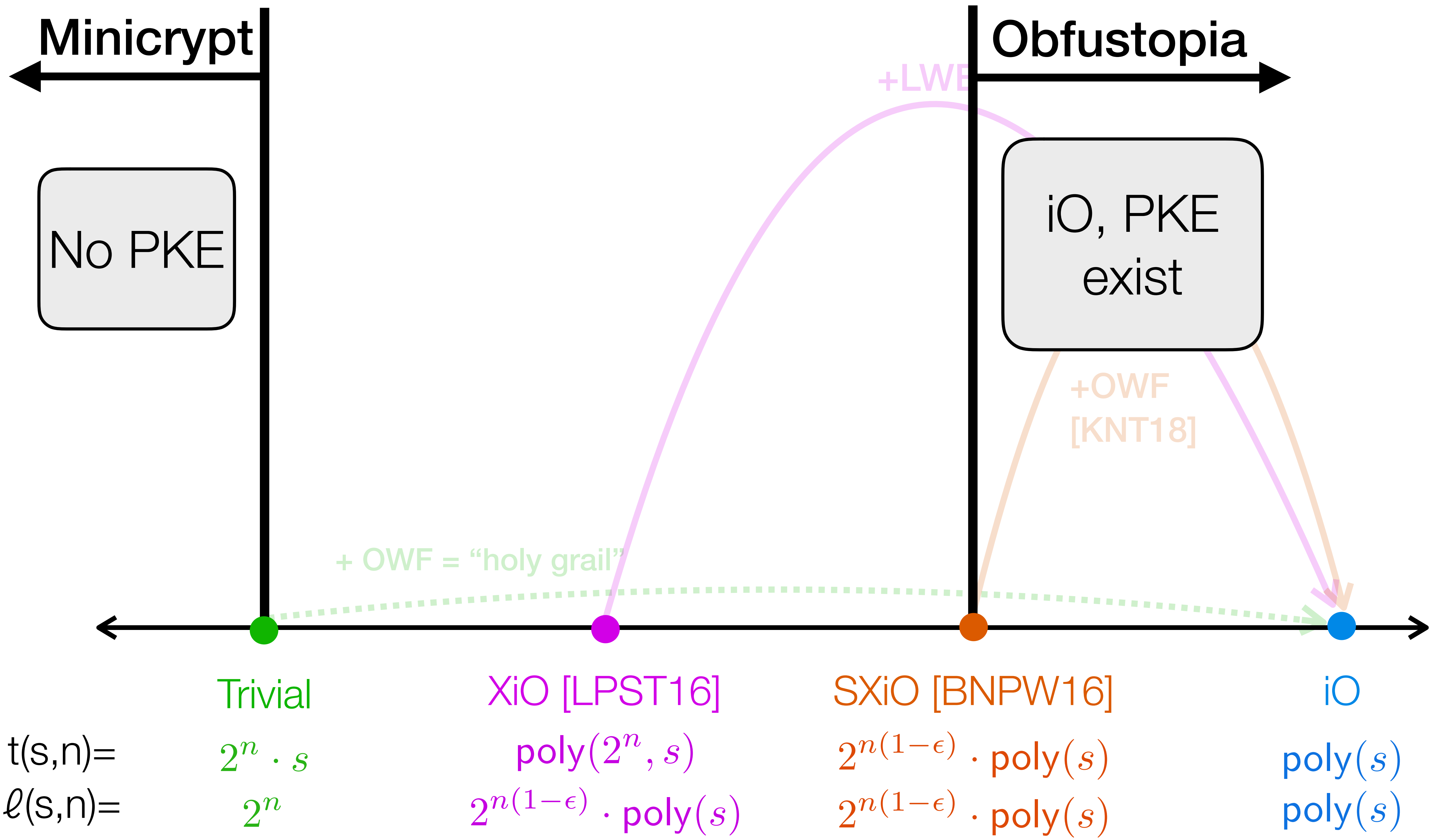
Compression Hierarchy

Assume sub-exponential OWF



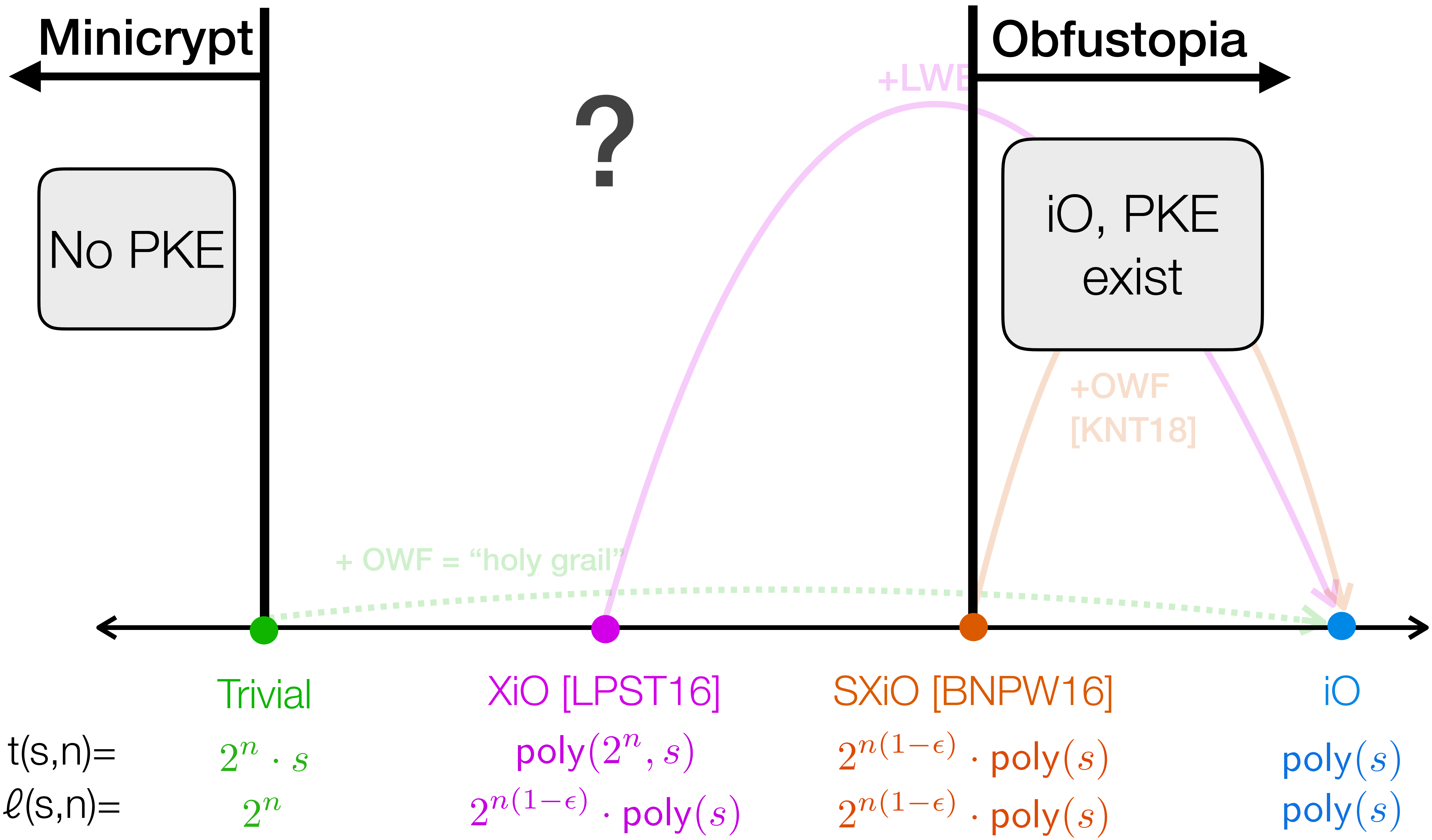
Compression Hierarchy

Assume sub-exponential OWF



Compression Hierarchy

Assume sub-exponential OWF



Our Results

Compressing obfuscation as an independent primitive

Our Results

Compressing obfuscation as an independent primitive

1. Power of compressing obfuscation

Our Results

Compressing obfuscation as an independent primitive

1. Power of compressing obfuscation

XiO + one-way functions $\not\Rightarrow$ public-key encryption in a black-box way

Our Results

Compressing obfuscation as an independent primitive

1. **Power of compressing obfuscation**

   XiO + one-way functions $\not\Rightarrow$ public-key encryption in a black-box way

2. **Existence with statistical security**

Our Results

Compressing obfuscation as an independent primitive

1. Power of compressing obfuscation

XiO + one-way functions $\not\Rightarrow$ public-key encryption in a black-box way

2. Existence with statistical security

- ✅ Constructions for "powerful" class of circuits (e.g., AC^0)
- ❌ Unlikely to exist with stronger compression

Our Results

Compressing obfuscation as an independent primitive

1. Power of compressing obfuscation

XiO + one-way functions $\not\Rightarrow$ public-key encryption in a black-box way

2. Existence with statistical security

☑ Constructions for "powerful" class of circuits (e.g., AC^0)

☒ Unlikely to exist with stronger compression

3. Existence under computational assumptions

Our Results

Compressing obfuscation as an independent primitive

1. Power of compressing obfuscation

XiO + one-way functions $\not\Rightarrow$ public-key encryption in a black-box way

2. Existence with statistical security

✅ Constructions for "powerful" class of circuits (e.g., AC^0)

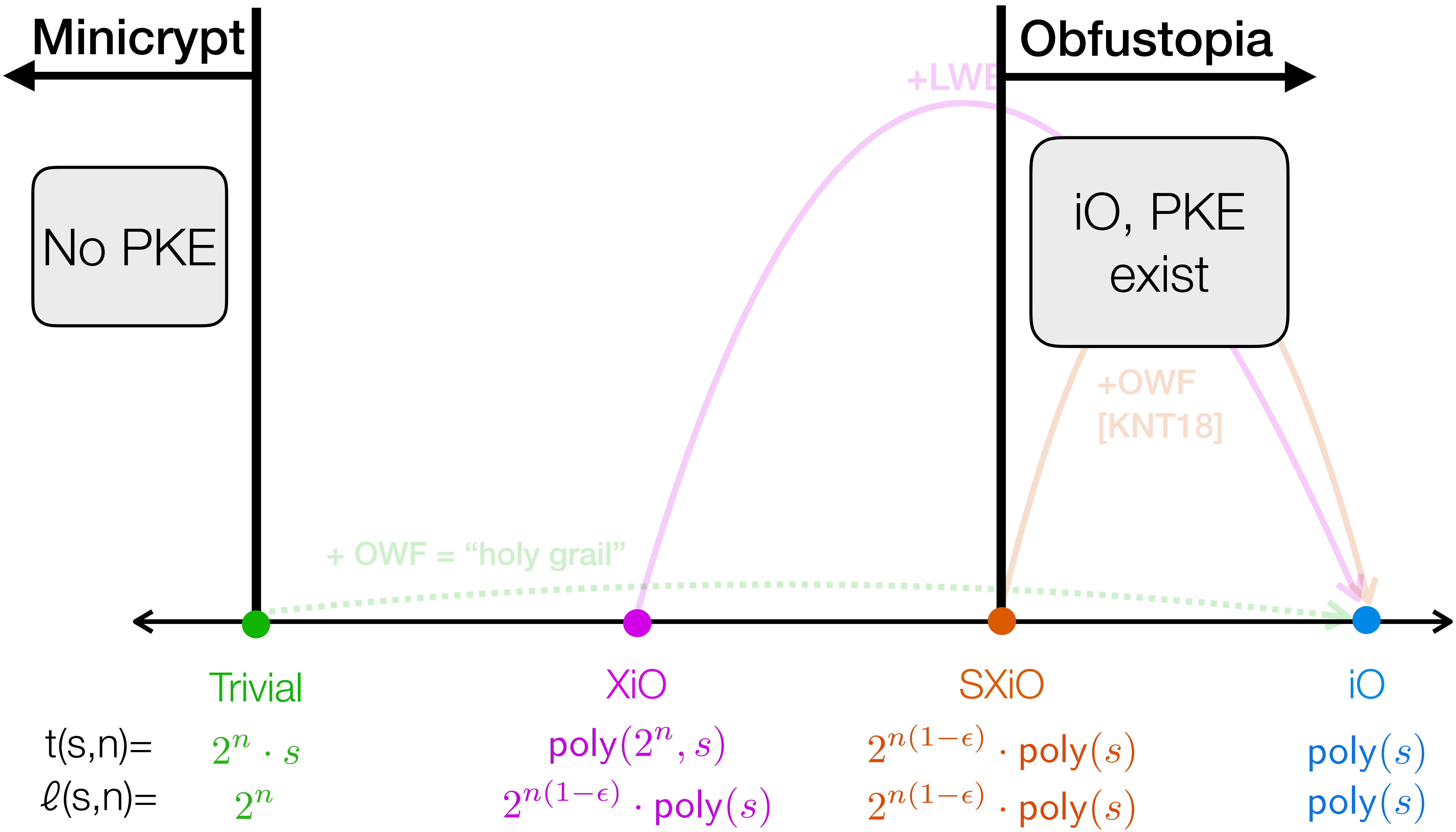
❎ Unlikely to exist with stronger compression

3. Existence under computational assumptions

Approximately-correct (S)XiO + polynomial LWE + NIZK $\Rightarrow$ correct (S)XiO

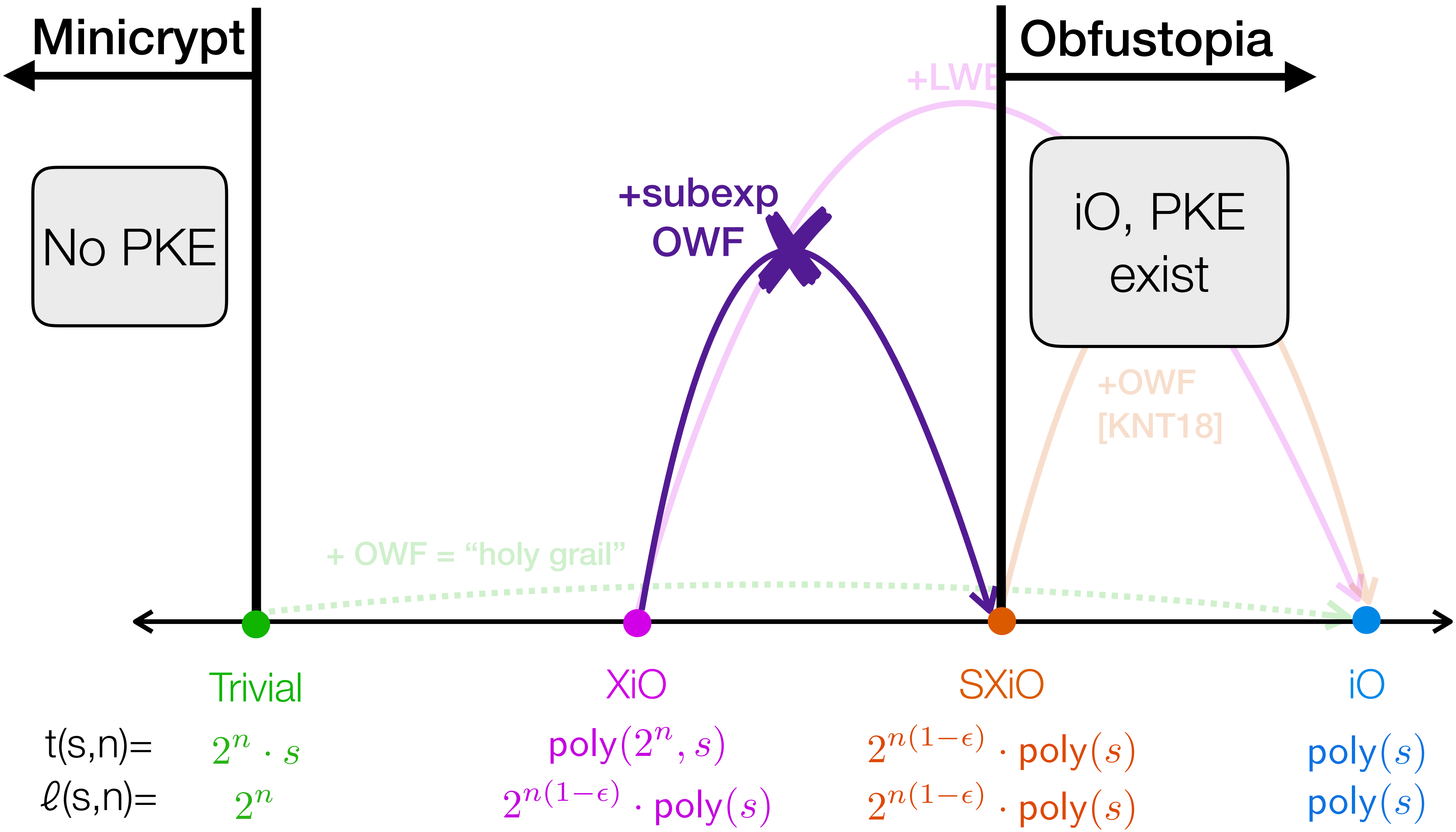
Impact of Results

Assume sub-exponential OWF



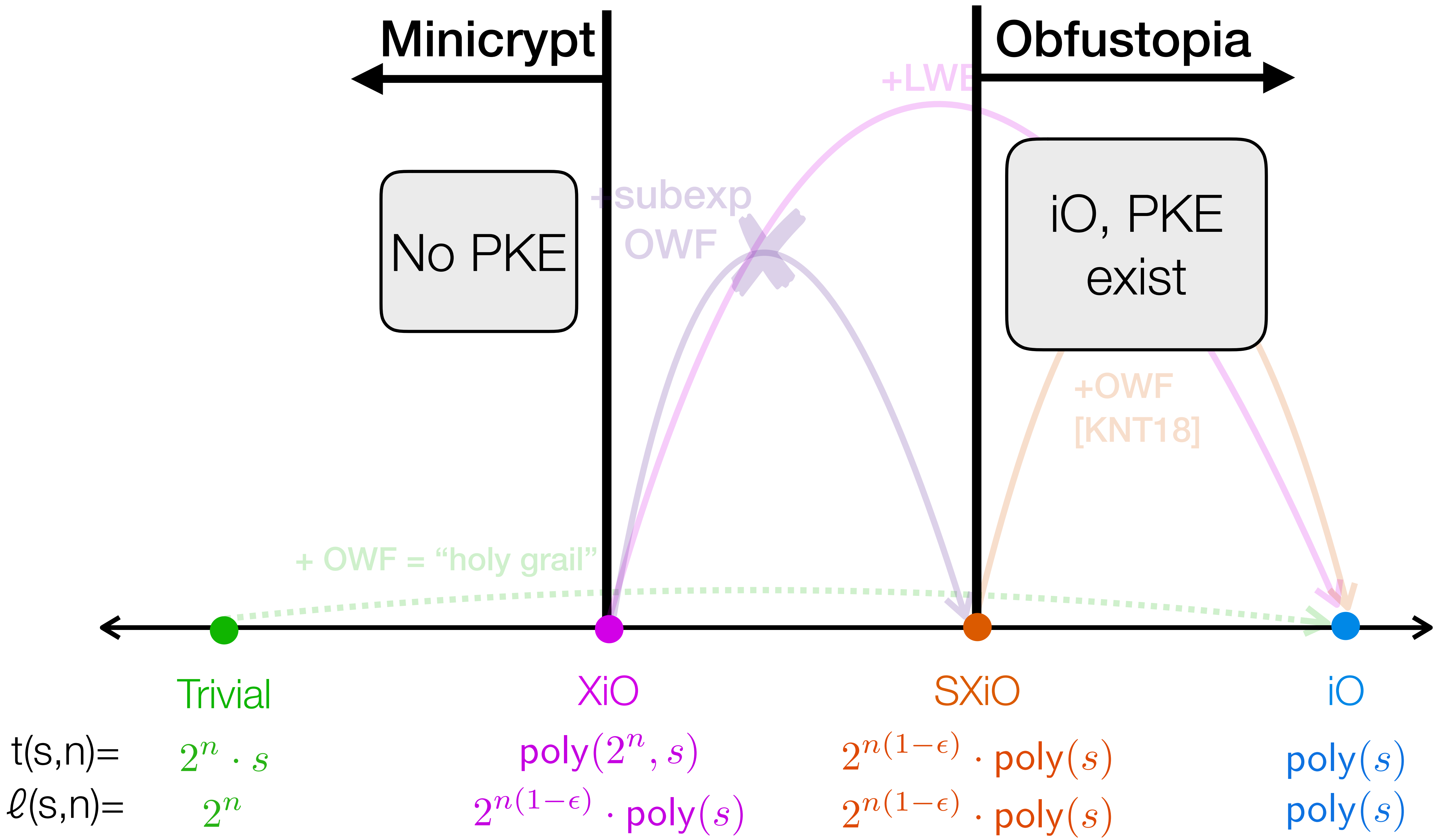
Impact of Results

Assume sub-exponential OWF



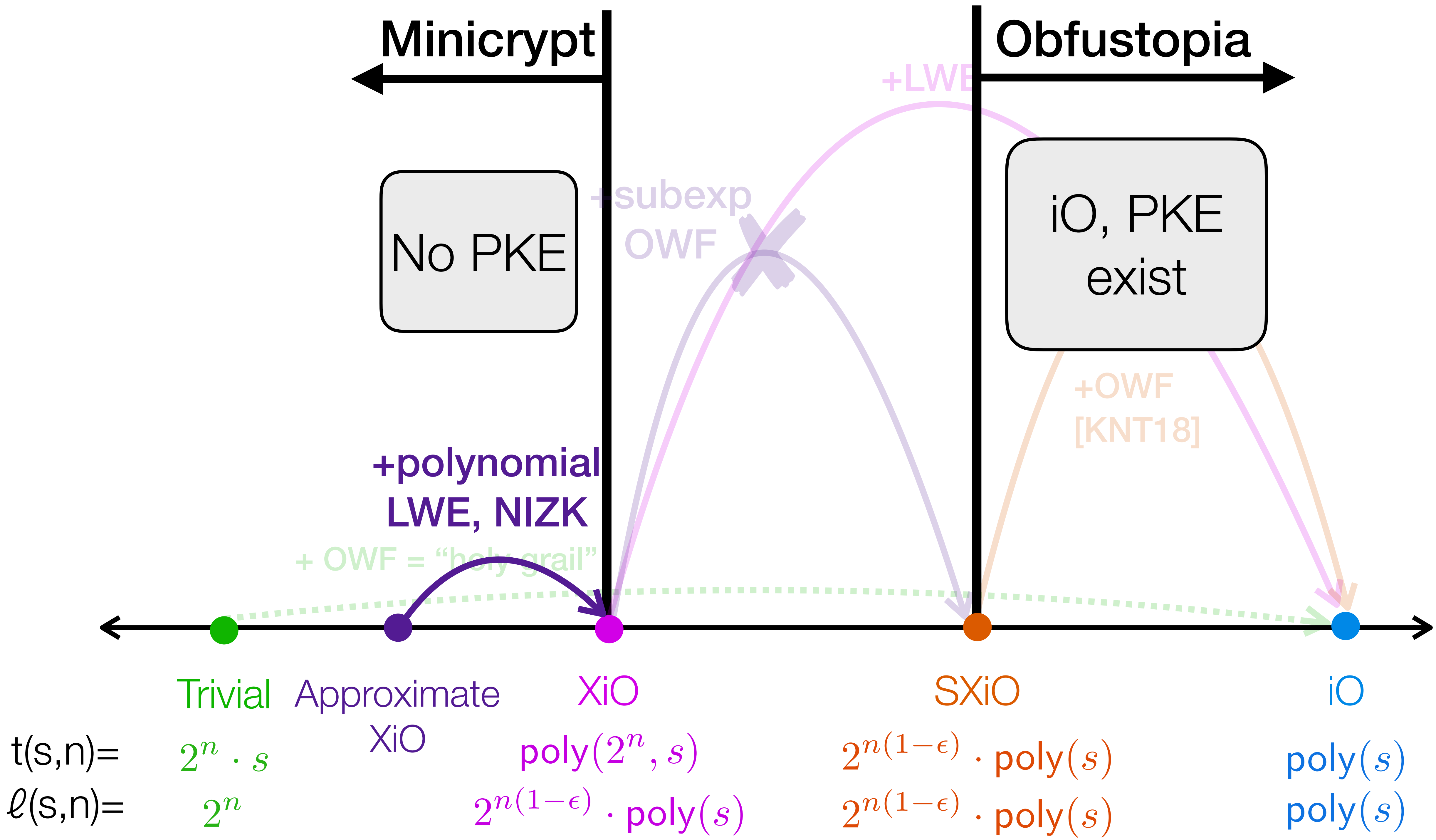
Impact of Results

Assume sub-exponential OWF



Impact of Results

Assume sub-exponential OWF



Our Results and Outline

1. Power of compressing obfuscation

XiO + one-way functions $\not\Rightarrow$ public-key encryption in a black-box way

2. Existence with statistical security

- ☑ Constructions for “powerful” class of circuits (e.g., AC^0)
- ☒ Unlikely to exist with stronger compression

3. Existence under computational assumptions

Approximately-correct (S)XiO + polynomial LWE + NIZK $\Rightarrow$ correct (S)XiO

Power of XiO

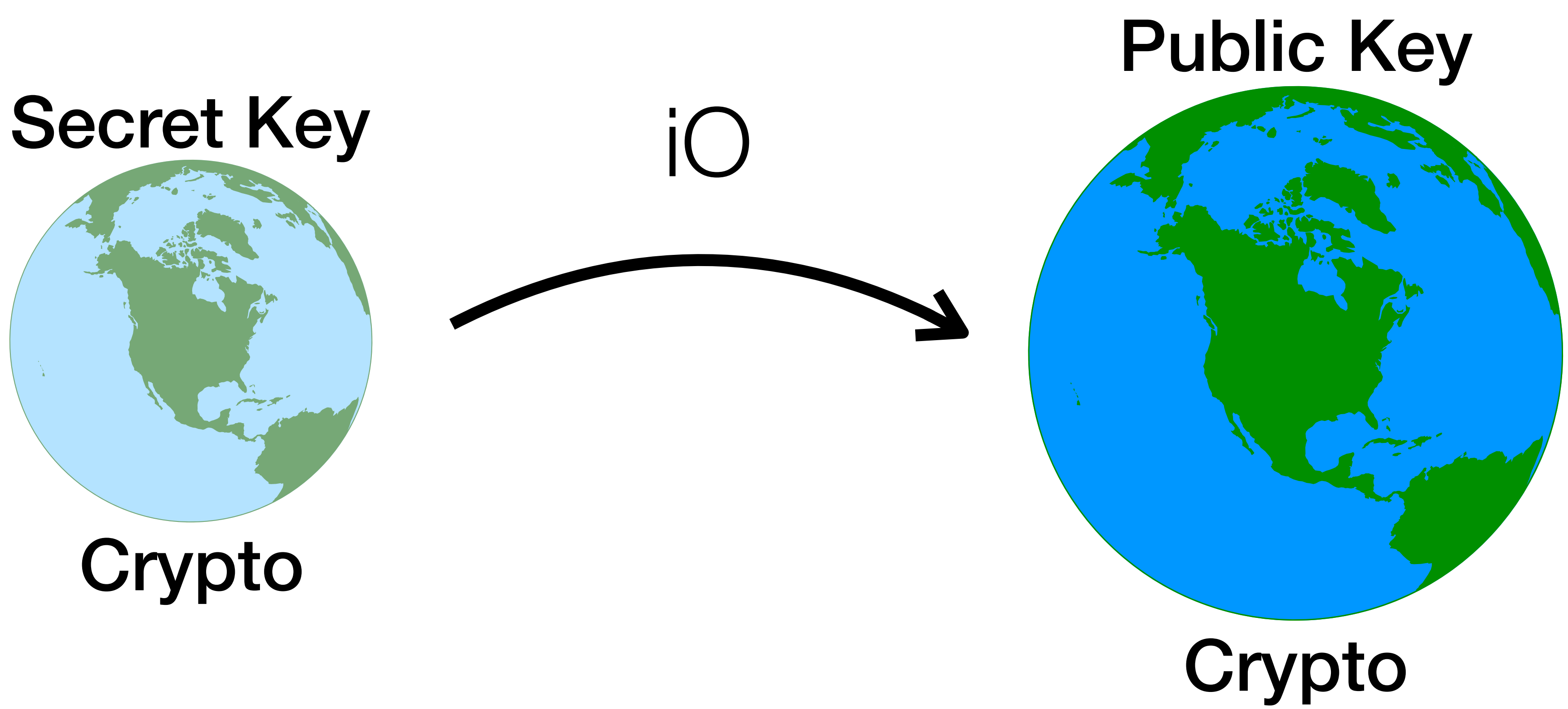
Recall: XiO + LWE $\Rightarrow$ iO

Is XiO useful without LWE?

Power of XiO

Recall: XiO + LWE $\Rightarrow$ iO

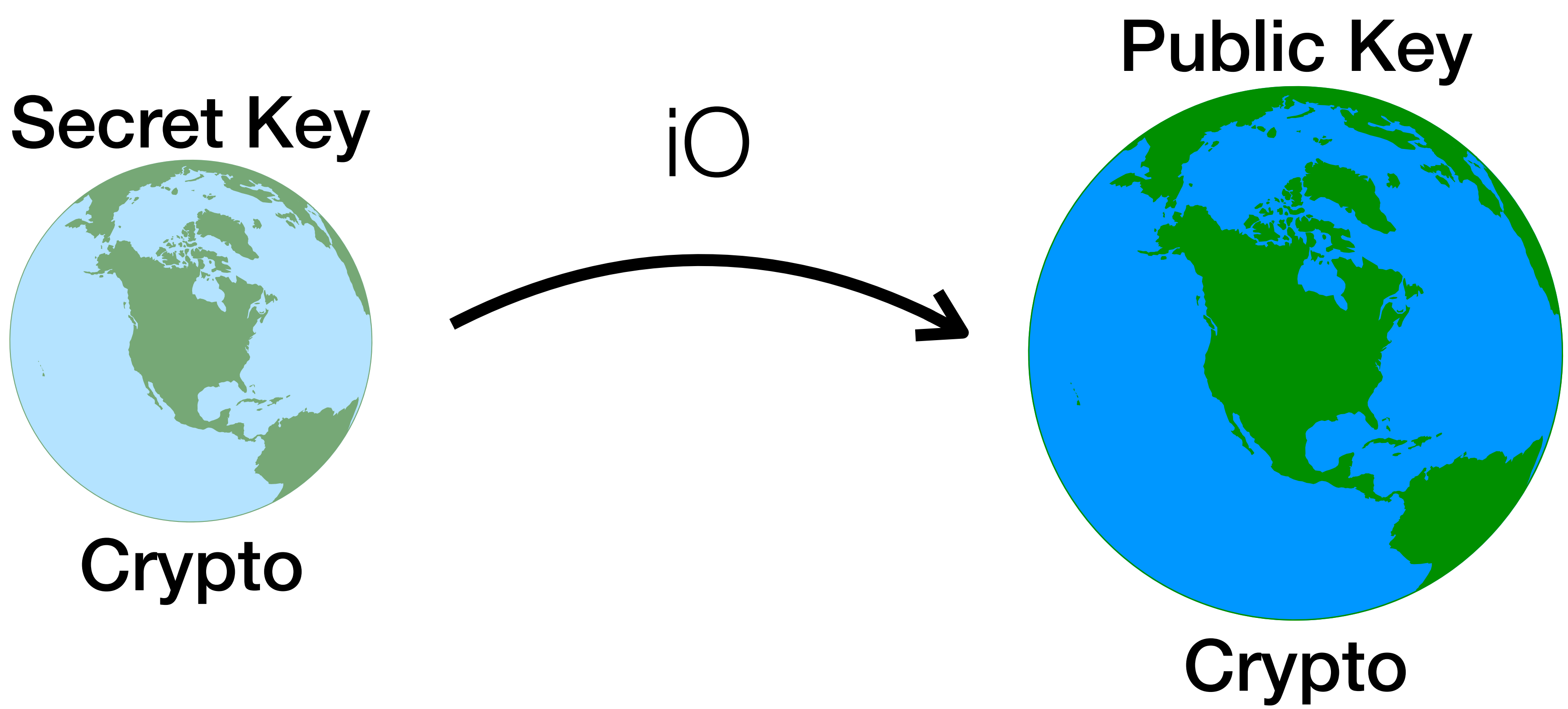
Is XiO useful without LWE?



Power of XiO

Recall: XiO + LWE $\Rightarrow$ iO

Is XiO useful without LWE?

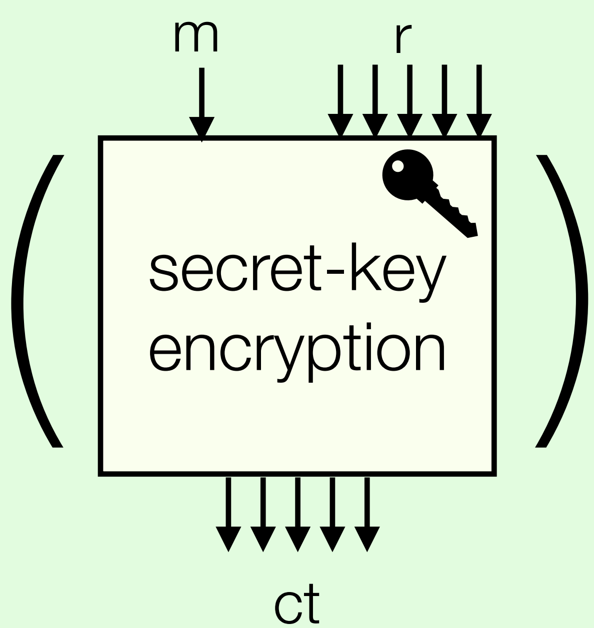


Theorem: **XiO** + **OWF** $\not\Rightarrow$ **PKE** in a black-box way

Intuition: PKE from iO + OWF

[SW14]

pk ← iO.Obf



Enc(pk,m):

Sample r and

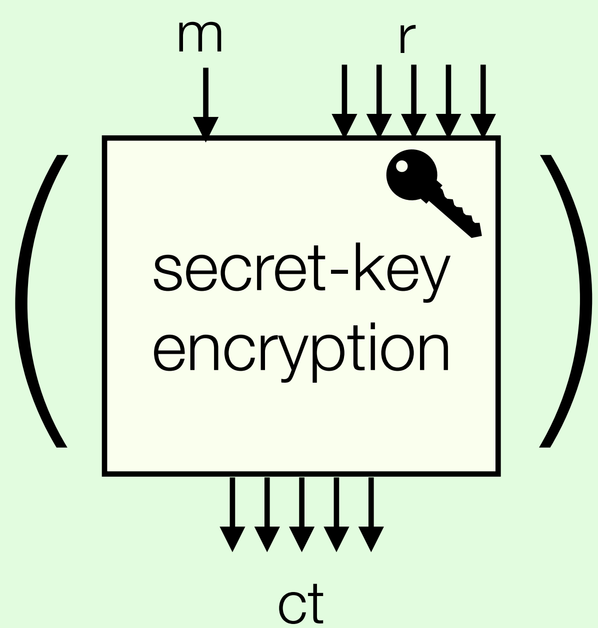
evaluate pk

on (m,r)

Intuition: PKE from iO + OWF

[SW14]

pk ← iO.Obf



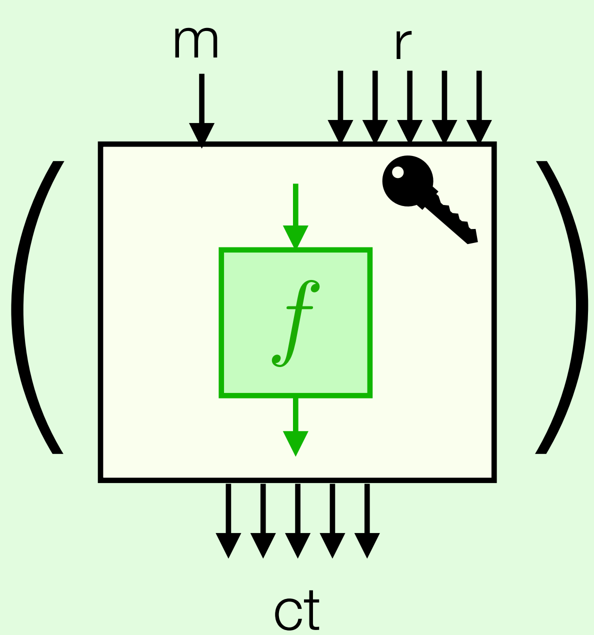
Enc(pk,m):

Sample r and evaluate pk on (m,r)

- With XiO, (m,r) must be short—Adversary can learn all possible ciphertexts!
- We show this is inherent for any construction from XiO

Intuition: PKE from iO + OWF

[SW14]

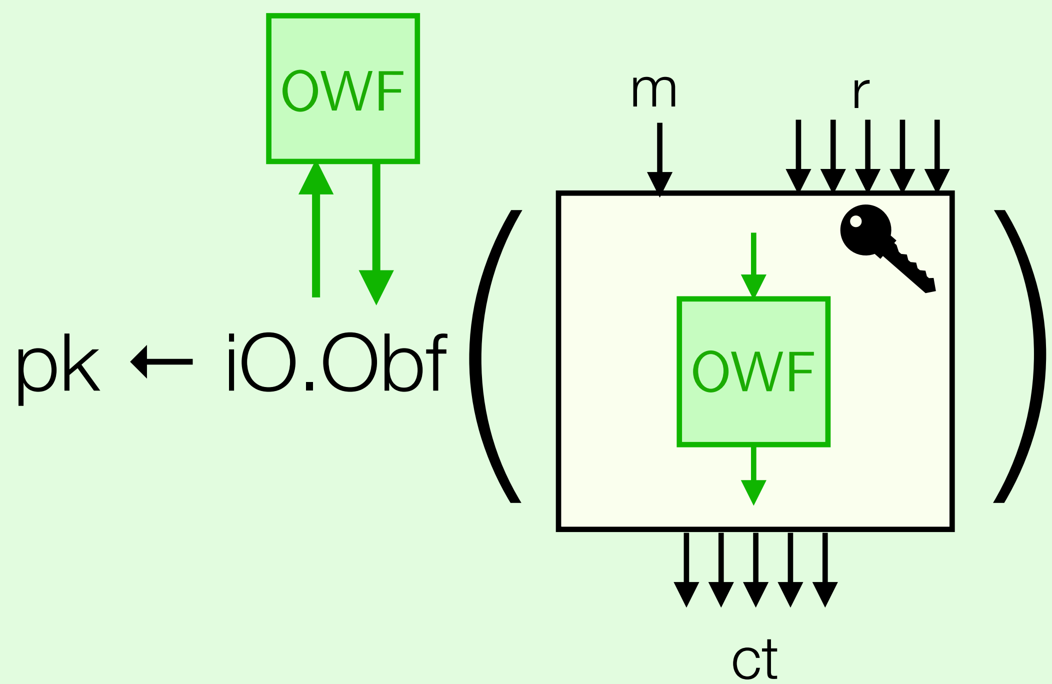
pk ← iO.Obf 

Enc(pk,m):
Sample r and
evaluate pk
on (m,r)

- With XiO, (m,r) must be short—Adversary can learn all possible ciphertexts!
- We show this is inherent for any construction from XiO

Intuition: PKE from iO + OWF

[SW14]



Enc(pk,m):

Sample r and evaluate pk on (m,r)

- With XiO, (m,r) must be short—Adversary can learn all possible ciphertexts!
- We show this is inherent for any construction from XiO

Black-Box Model

We consider XiO for *oracle-aided* circuits

Black-Box Model

We consider XiO for *oracle-aided* circuits

- First used for circuits with OWF gates [BKSY11, AS16]

Black-Box Model

We consider XiO for *oracle-aided* circuits

- First used for circuits with OWF gates [BKSY11, AS16]

  Problem: Separation overcome by new constructions (e.g., PKE from SXiO + OWFs [BNPW16])

Black-Box Model

We consider XiO for *oracle-aided* circuits

- First used for circuits with OWF gates [BKSY11, AS16]

  Problem: Separation overcome by new constructions (e.g., PKE from SXiO + OWFs [BNPW16])

- Extended to circuits with iO and OWF gates [GMM17]

Black-Box Model

We consider XiO for *oracle-aided* circuits

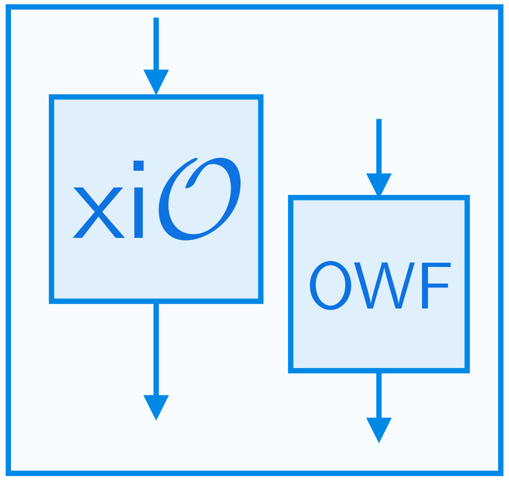
- First used for circuits with OWF gates [BKSY11, AS16]

  Problem: Separation overcome by new constructions (e.g., PKE from SXiO + OWFs [BNPW16])

- Extended to circuits with iO and OWF gates [GMM17]

Our result — extended model

Captures known techniques for iO, e.g., **"self-feeding"** techniques



Black-Box Model

We consider XiO for *oracle-aided* circuits

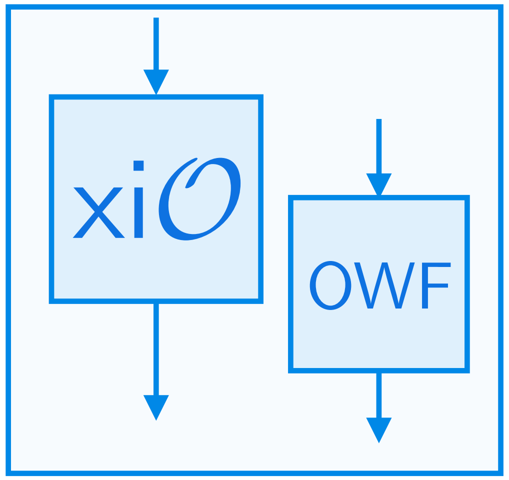
- First used for circuits with OWF gates [BKSY11, AS16]

  Problem: Separation overcome by new constructions (e.g., PKE from SXiO + OWFs [BNPW16])

- Extended to circuits with iO and OWF gates [GMM17]

Our result — extended model

Captures known techniques for iO, e.g., **“self-feeding”** techniques



Non-black-box extension of Impagliazzo-Rudich separation [IR89]

Our Results and Outline

1. Power of compressing obfuscation

XiO + one-way functions $\not\Rightarrow$ public-key encryption in a black-box way

2. Existence with statistical security

- ✅ Constructions for “powerful” class of circuits (e.g., AC^0)
- ❎ Unlikely to exist with stronger compression

3. Existence under computational assumptions

Approximately-correct (S)XiO + polynomial LWE + NIZK $\Rightarrow$ correct (S)XiO

Statistically Secure Compressing Obfuscation

Main idea: Take advantage of the running time of XiO

Statistically Secure Compressing Obfuscation

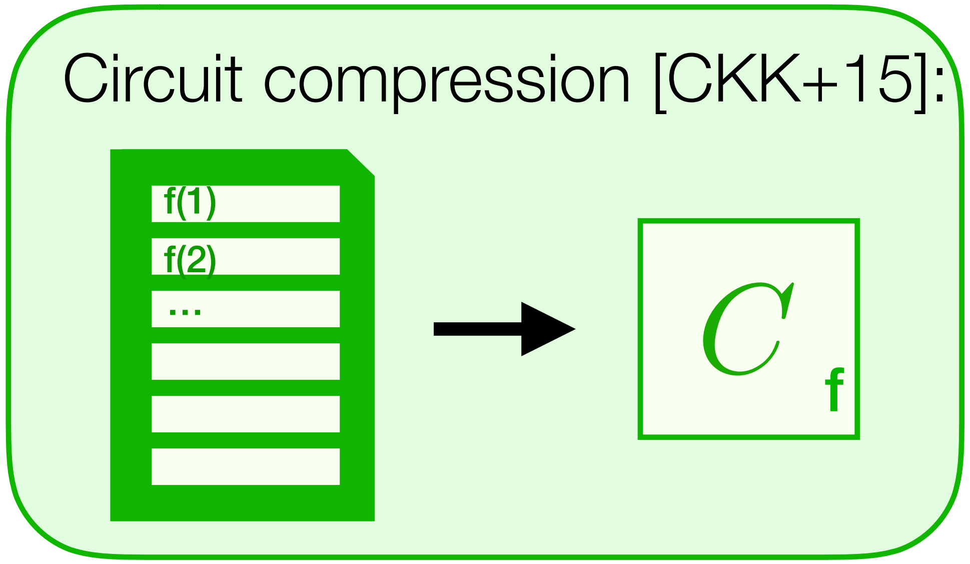
Main idea: Take advantage of the running time of XiO

Theorem: XiO with output length $2^{n(1-o(1))}$ exists for AC^0

Statistically Secure Compressing Obfuscation

Main idea: Take advantage of the running time of XiO

Theorem: XiO with output length $2^{n(1-o(1))}$ exists for AC^0

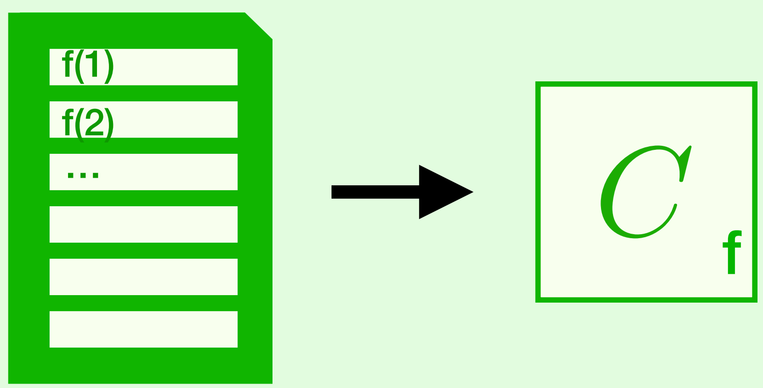


Statistically Secure Compressing Obfuscation

Main idea: Take advantage of the running time of XiO

Theorem: XiO with output length $2^{n(1-o(1))}$ exists for AC^0

Circuit compression [CKK+15]:

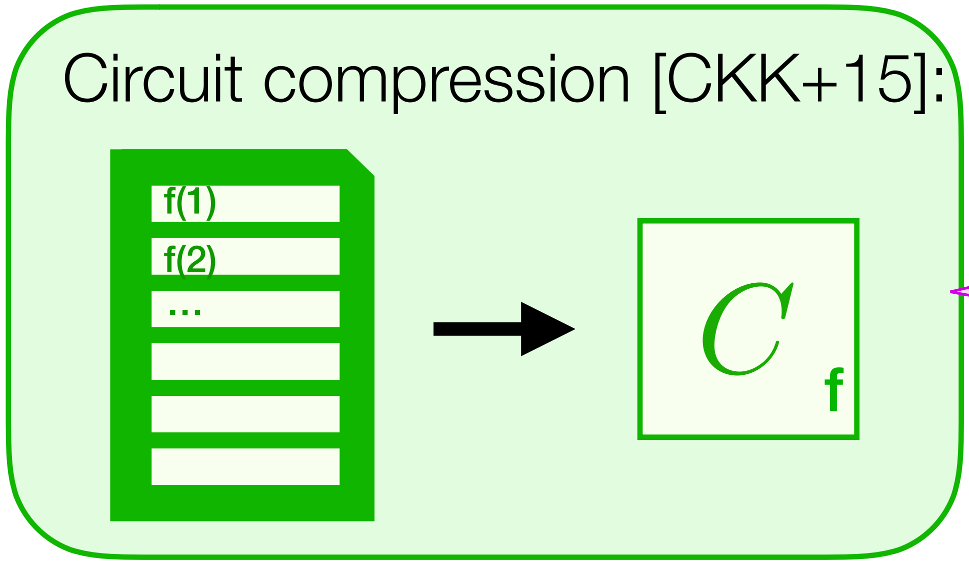


Construction is black-box, i.e., implies compressing VBB

Statistically Secure Compressing Obfuscation

Main idea: Take advantage of the running time of XiO

Theorem: XiO with output length $2^{n(1-o(1))}$ exists for AC^0



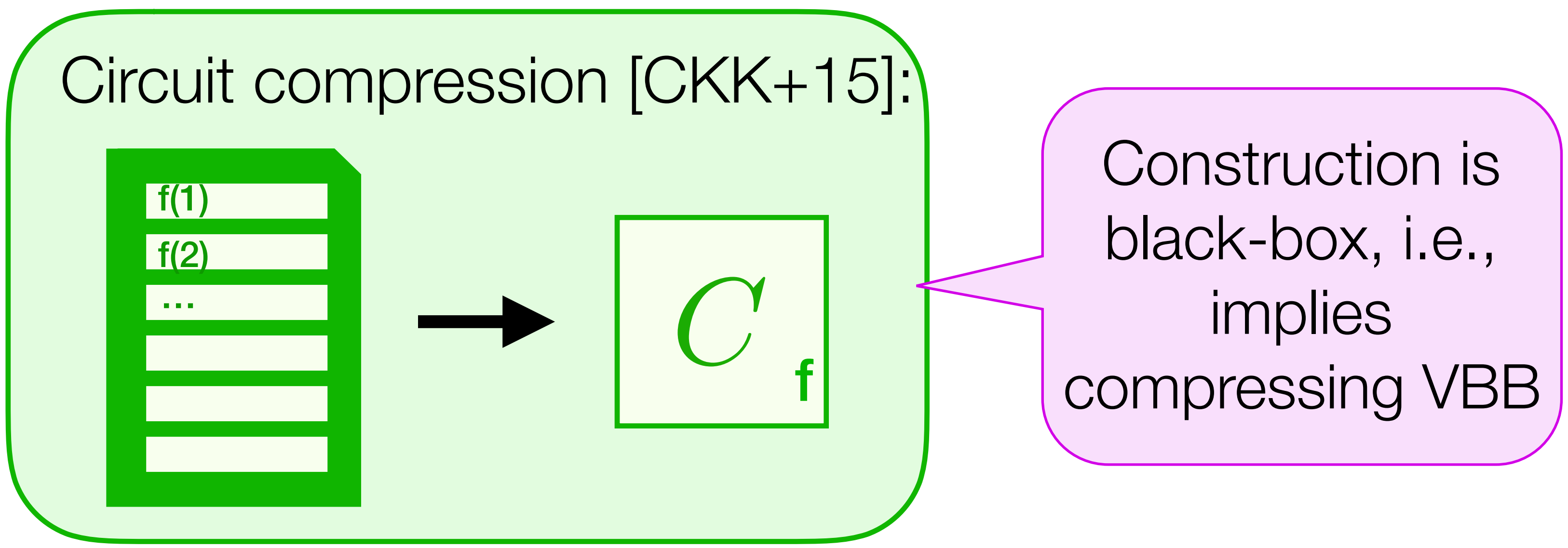
Construction is black-box, i.e., implies compressing VBB

Stronger compression implies nontrivial speedups for UNSAT

Statistically Secure Compressing Obfuscation

Main idea: Take advantage of the running time of XiO

Theorem: XiO with output length $2^{n(1-o(1))}$ exists for AC^0



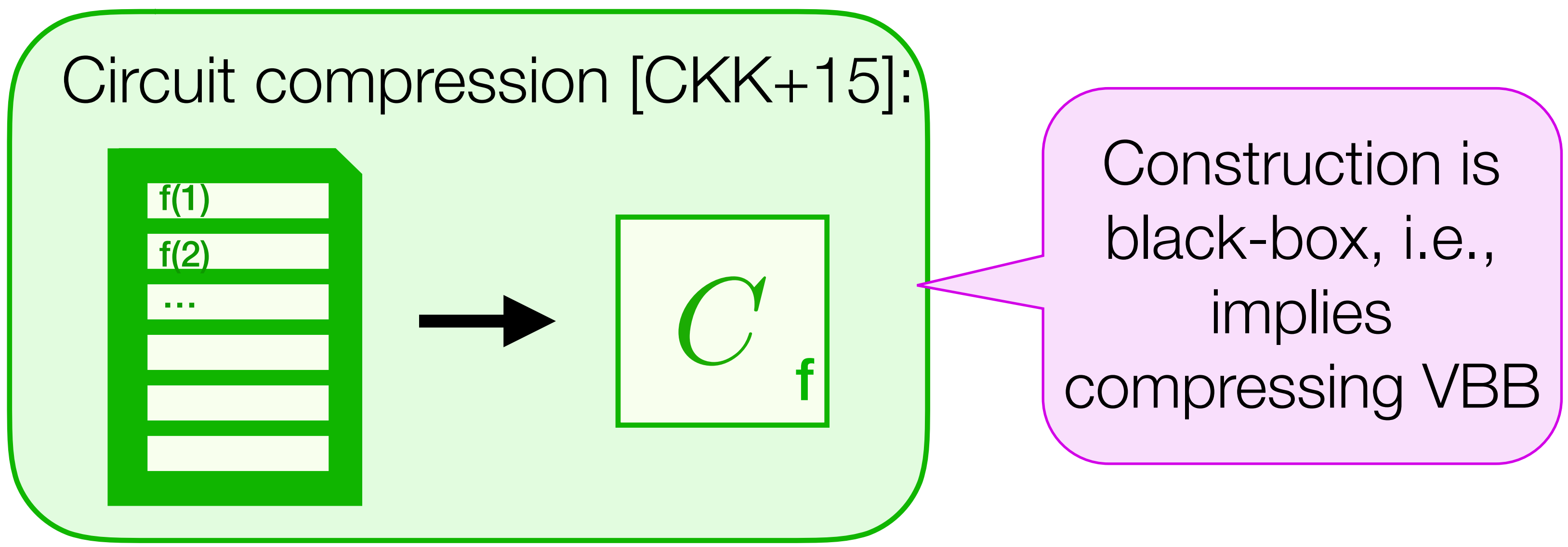
Stronger compression implies nontrivial speedups for UNSAT

Theorem: $(2^{\epsilon n}, 2^{\epsilon n})$-compressing obfuscation for depth 2 circuits implies UNSAT $\in$ AM$[2^{c\epsilon n}]$ for a constant c

Statistically Secure Compressing Obfuscation

Main idea: Take advantage of the running time of XiO

Theorem: XiO with output length $2^{n(1-o(1))}$ exists for AC⁰



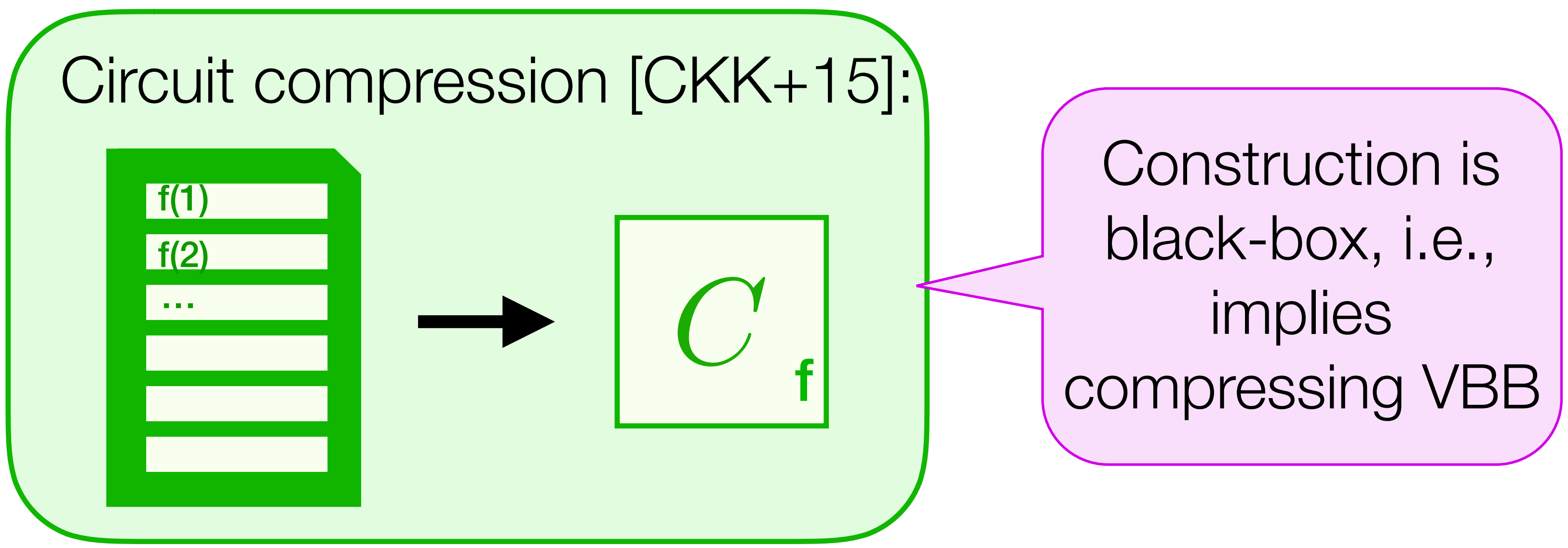
Stronger compression implies nontrivial speedups for UNSAT

Theorem: $(2^{\epsilon n}, 2^{\epsilon n})$-compressing obfuscation for depth 2 circuits implies $\text{UNSAT} \in \text{AM}[2^{c\epsilon n}]$ for a constant c

Statistically Secure Compressing Obfuscation

Main idea: Take advantage of the running time of XiO

Theorem: XiO with output length $2^{n(1-o(1))}$ exists for AC^0



Stronger compression implies nontrivial speedups for UNSAT

Theorem: $(2^{\epsilon n}, 2^{\epsilon n})$-compressing obfuscation for depth 2 circuits implies UNSAT $\in$ AM$[2^{c\epsilon n}]$ for a constant c

- Conclusion is true when ϵ = 1/2 [W16]
- Unknown for smaller ϵ

Conclusion

Compressing obfuscation is unusual!

Conclusion

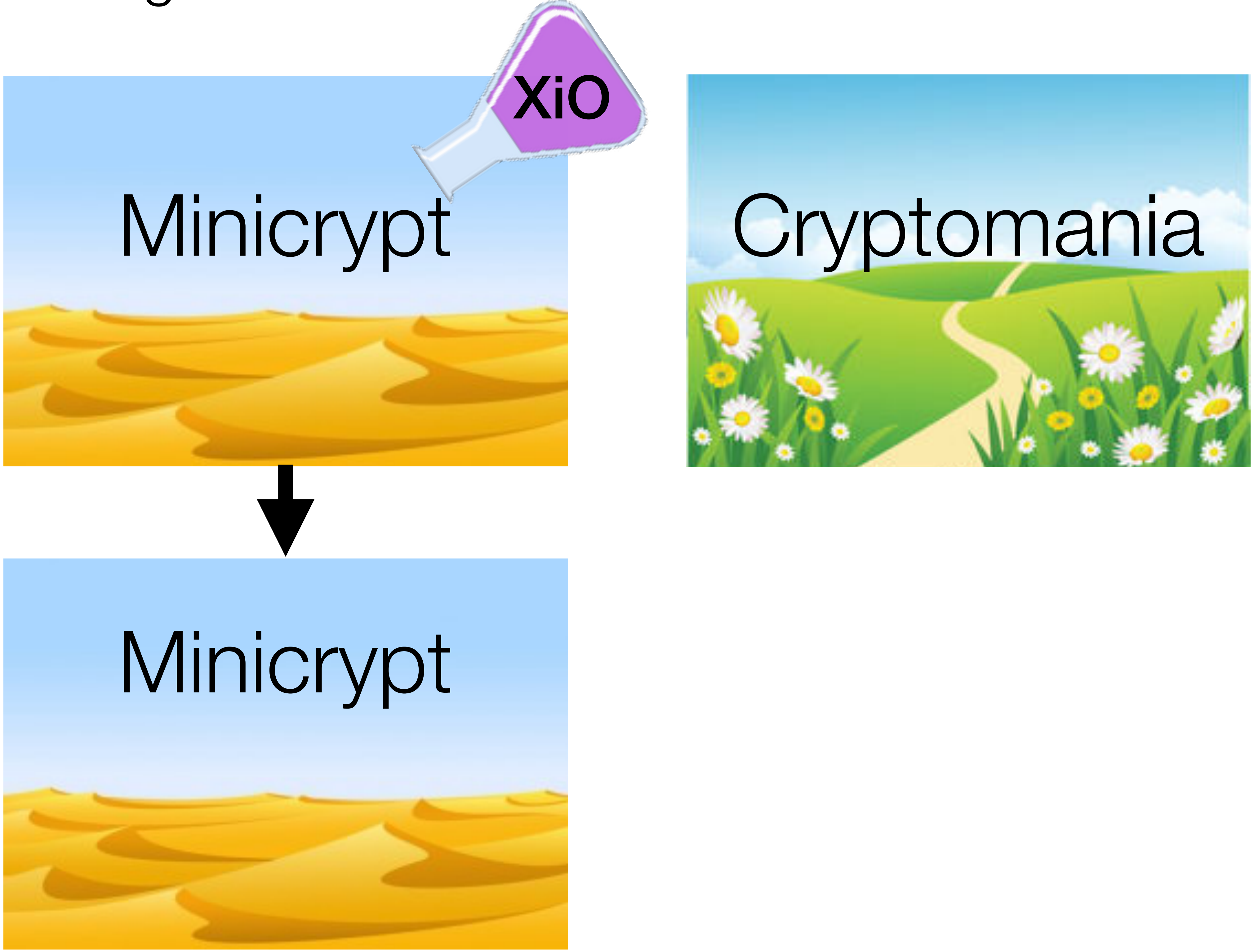
Compressing obfuscation is unusual!

Conclusion

Compressing obfuscation is unusual!

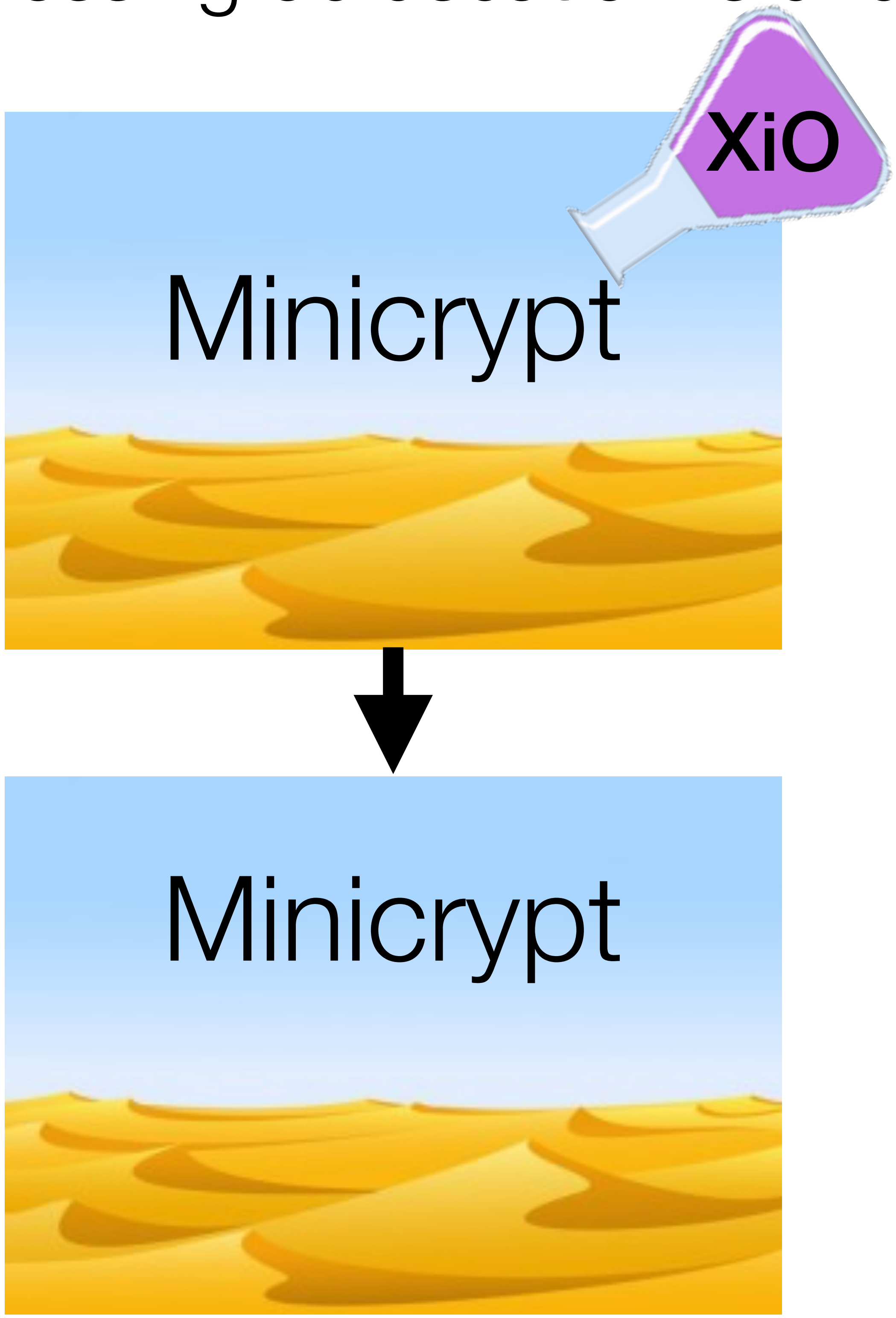
Conclusion

Compressing obfuscation is unusual!



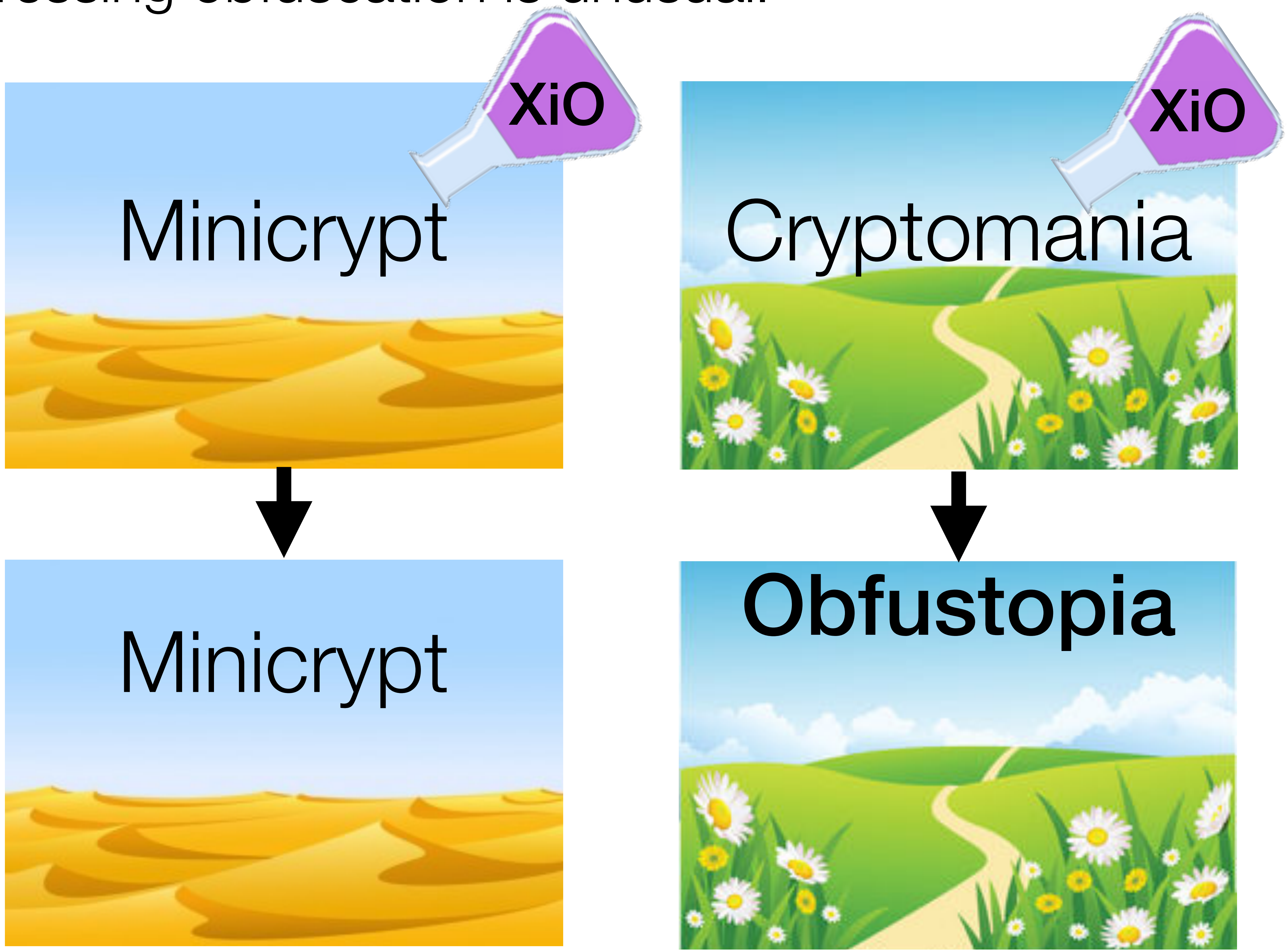
Conclusion

Compressing obfuscation is unusual!



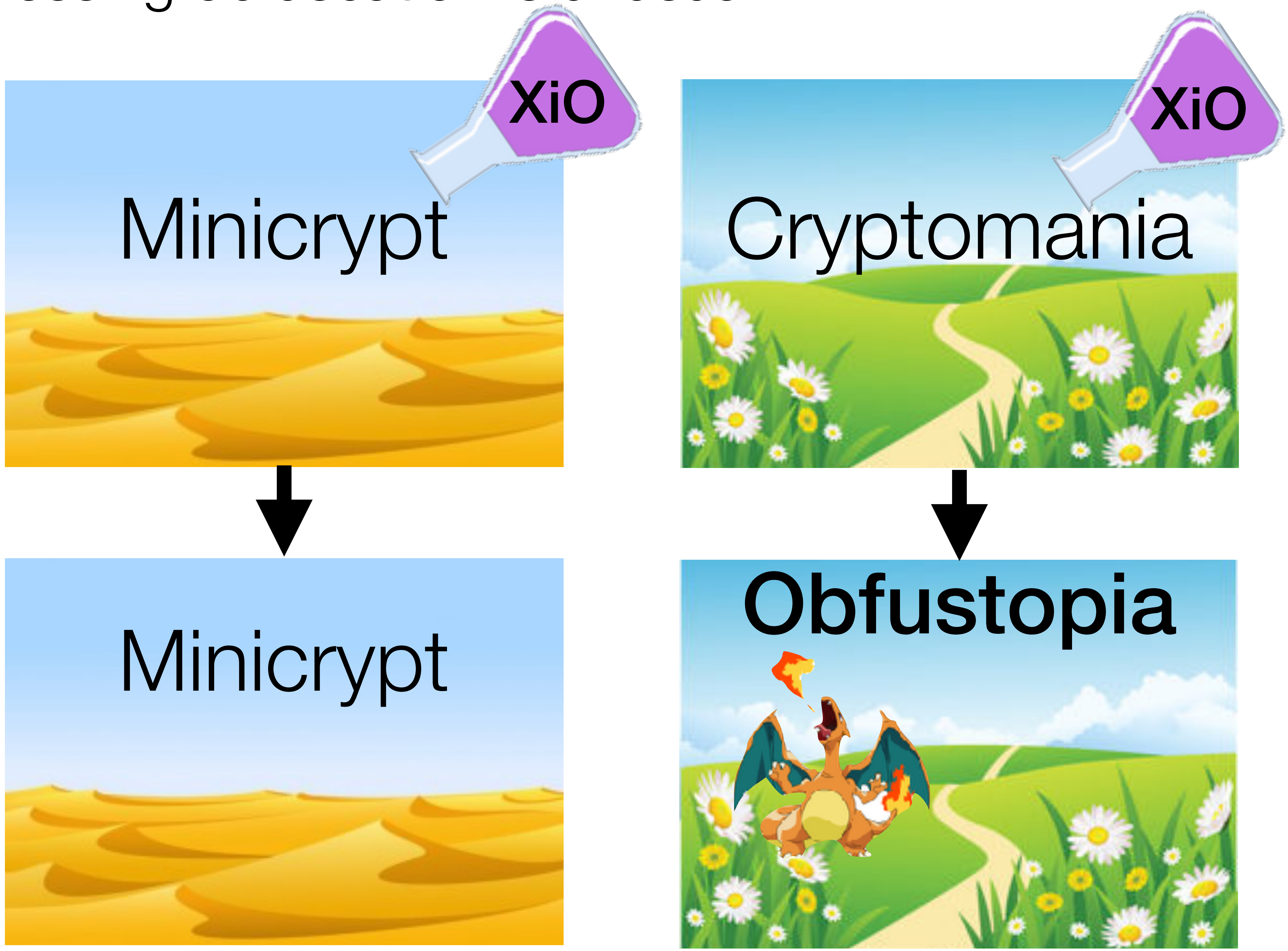
Conclusion

Compressing obfuscation is unusual!



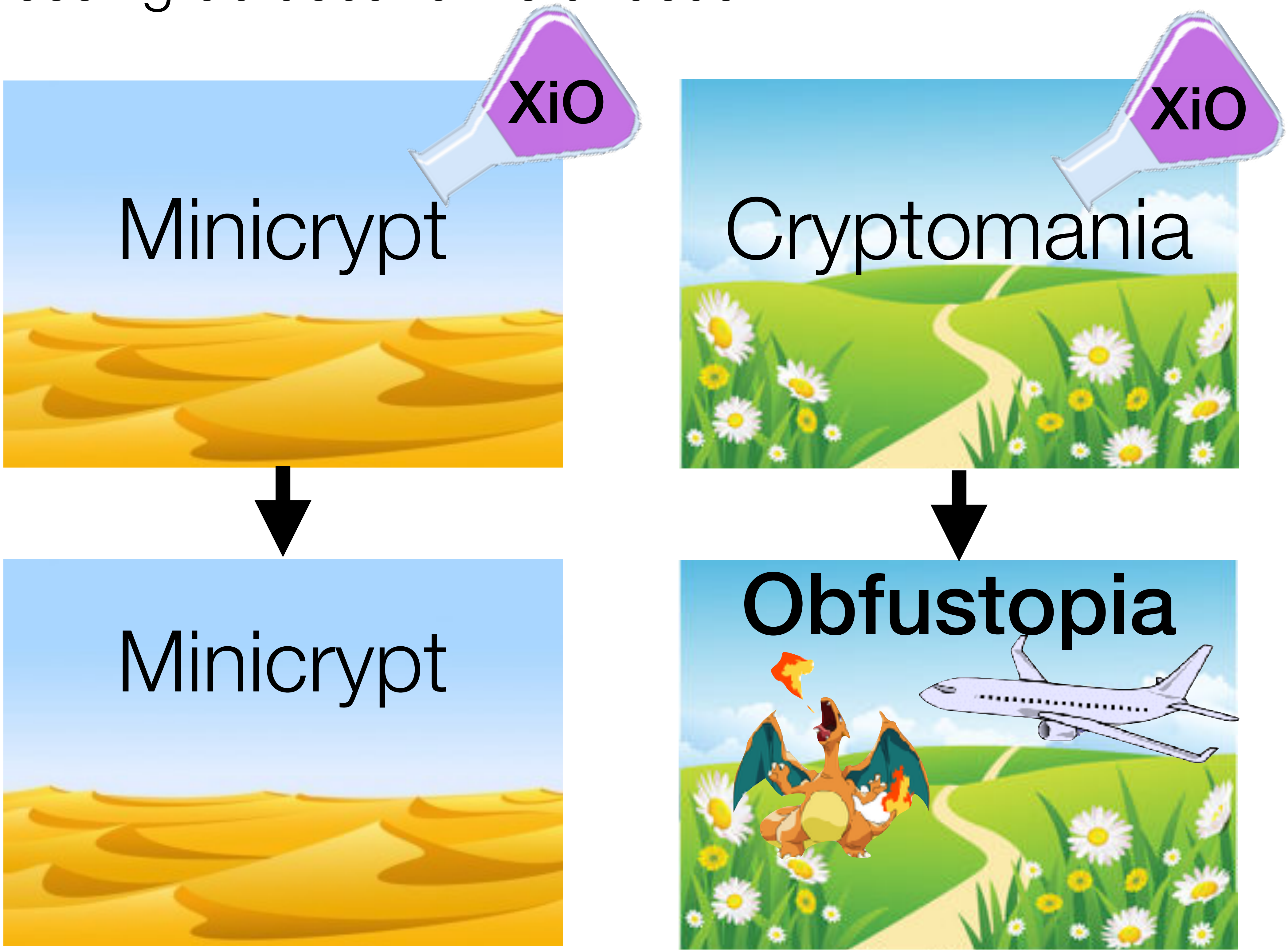
Conclusion

Compressing obfuscation is unusual!



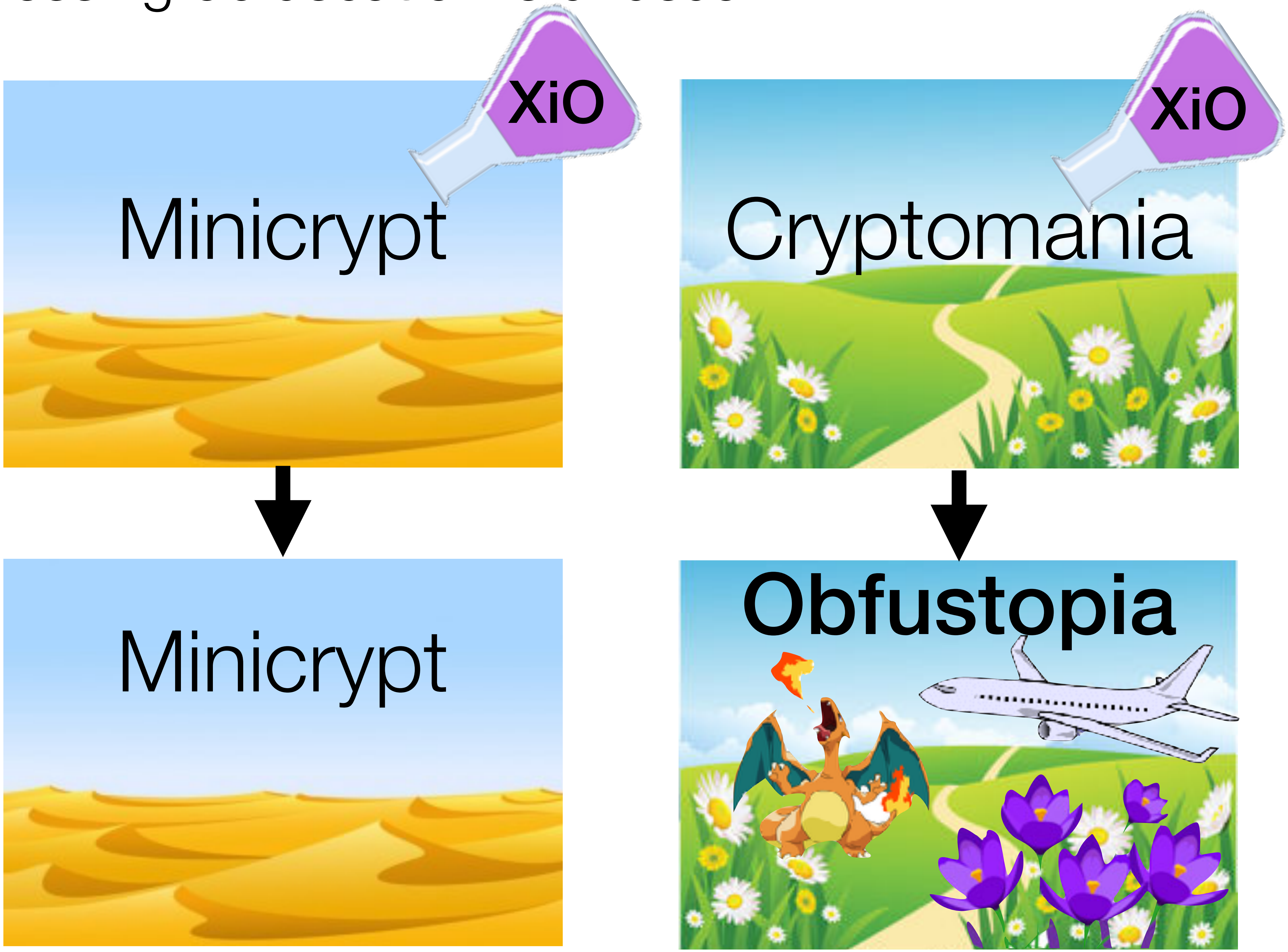
Conclusion

Compressing obfuscation is unusual!



Conclusion

Compressing obfuscation is unusual!

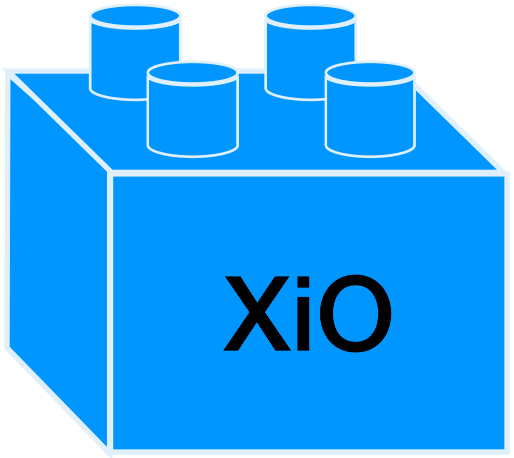


Conclusion

XiO is weak — cannot compress running time

Conclusion

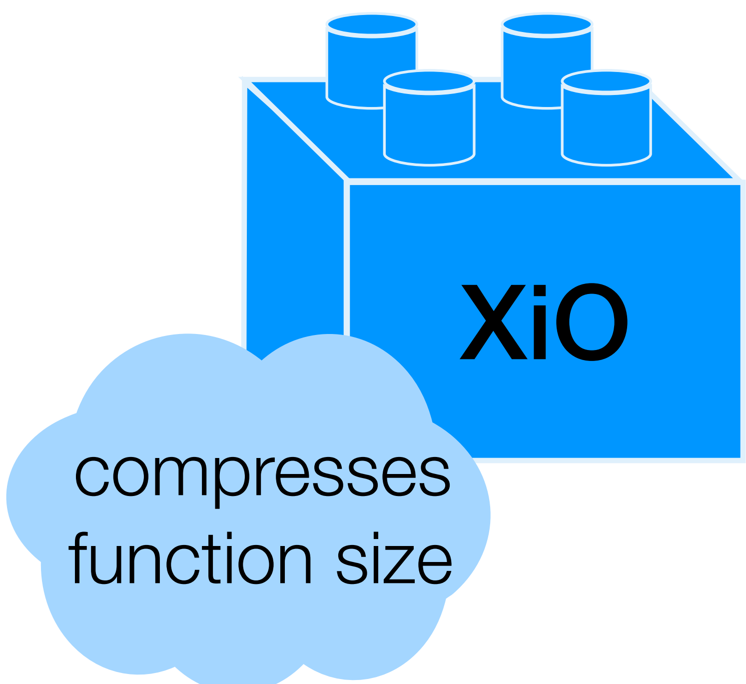
XiO is weak — cannot compress running time



+ LWE → iO

Conclusion

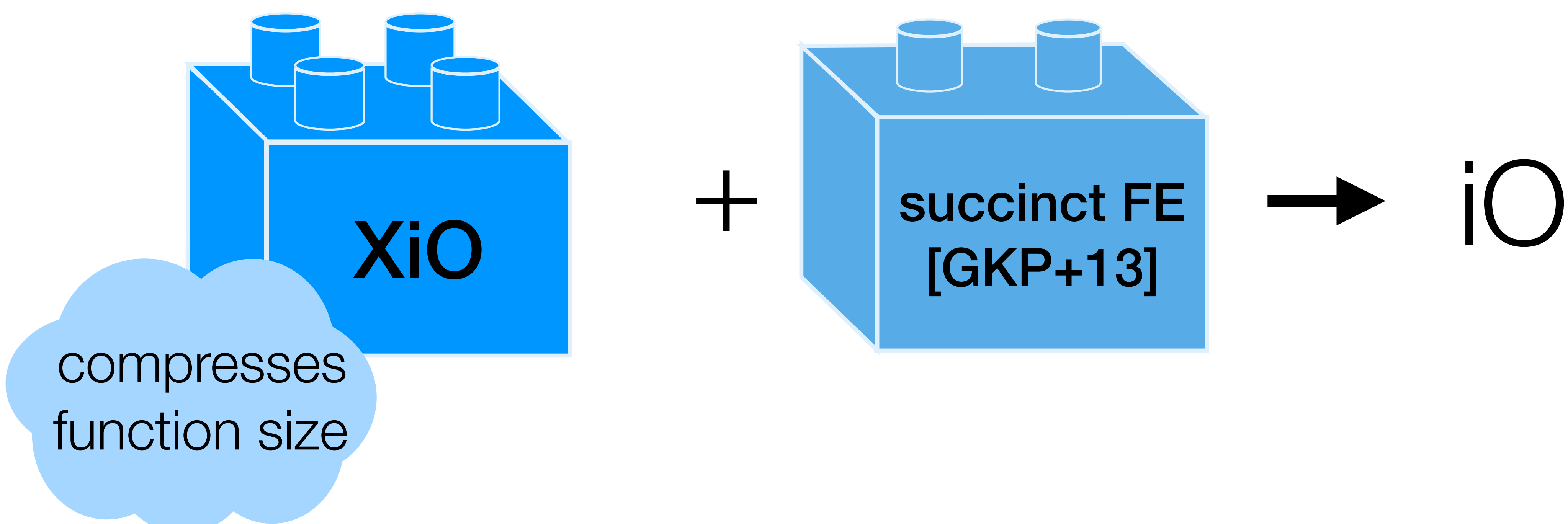
XiO is weak — cannot compress running time



\+ LWE → iO

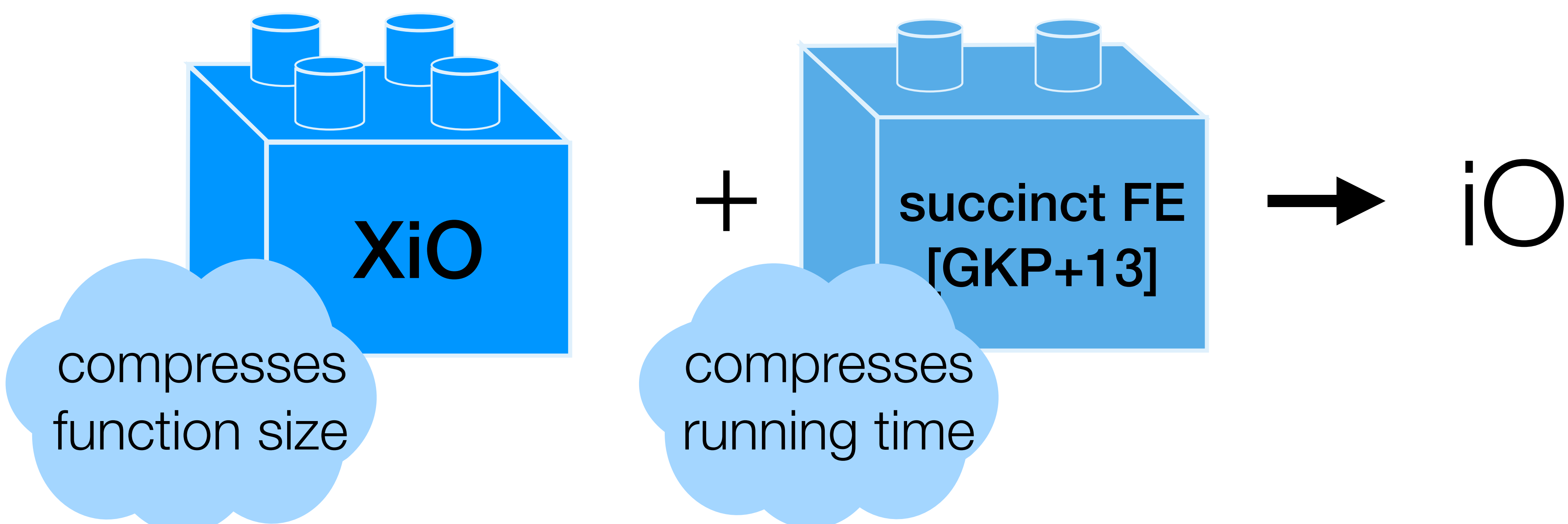
Conclusion

XiO is weak — cannot compress running time



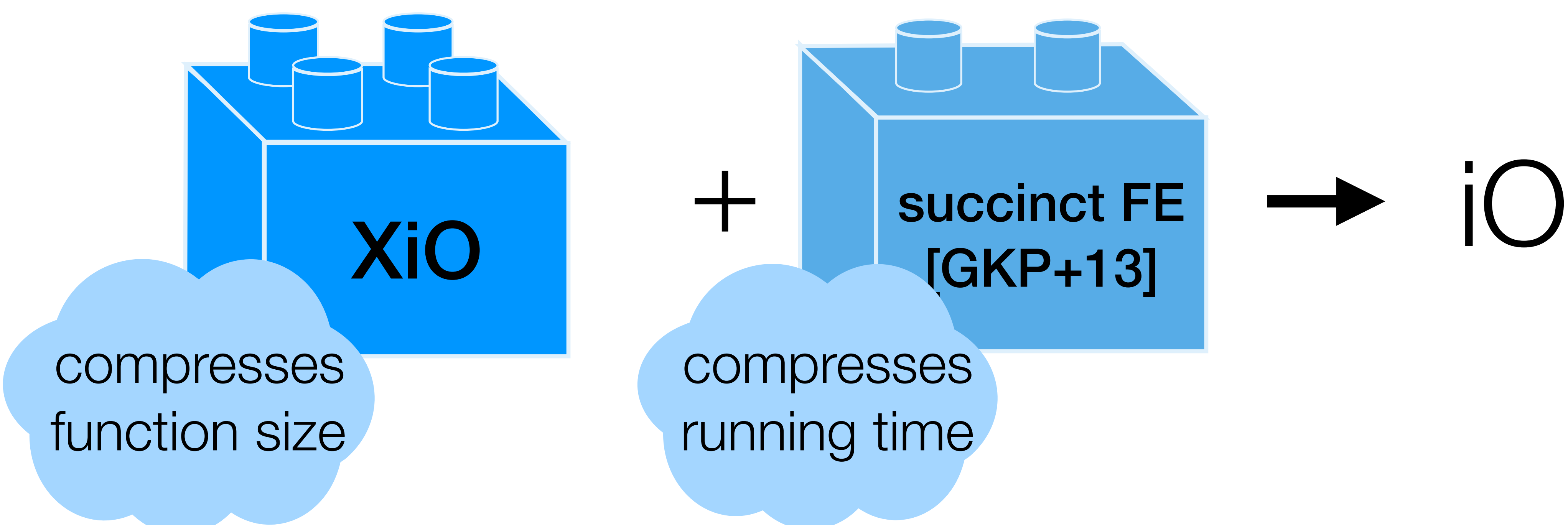
Conclusion

XiO is weak — cannot compress running time



Conclusion

XiO is weak — cannot compress running time



Thank you!