Zero Knowledge Succinct Arguments: an Introduction

Alessandro Chiesa
UC Berkeley

cryptography is a powerful tool
for building secure systems

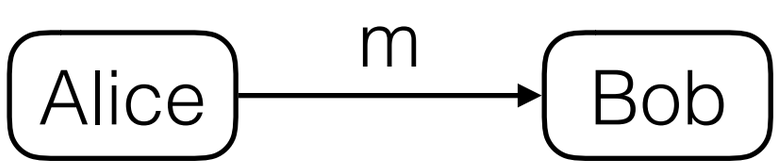
cryptography is a powerful tool
for building secure systems

much of the cryptography used today
offers security properties for **data**

cryptography is a powerful tool
for building secure systems

much of the cryptography used today
offers security properties for **data**

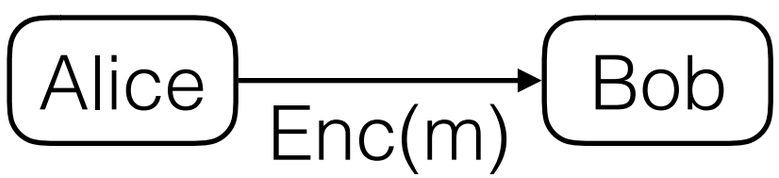
confidentiality



cryptography is a powerful tool
for building secure systems

much of the cryptography used today
offers security properties for **data**

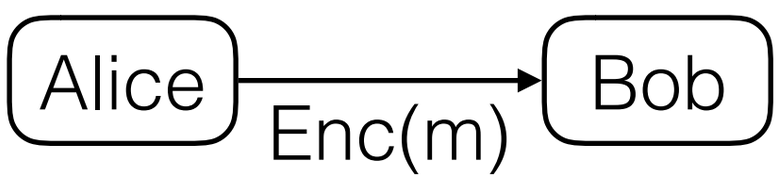
confidentiality



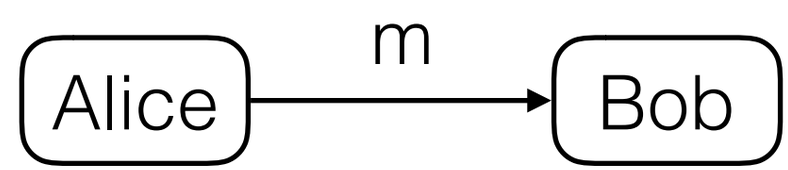
cryptography is a powerful tool
for building secure systems

much of the cryptography used today
offers security properties for **data**

confidentiality



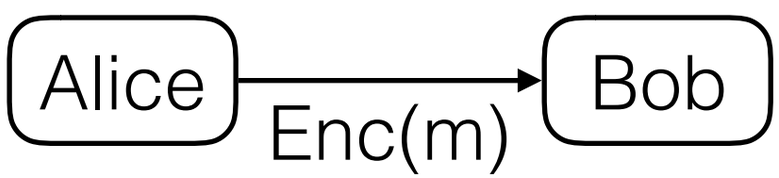
authenticity



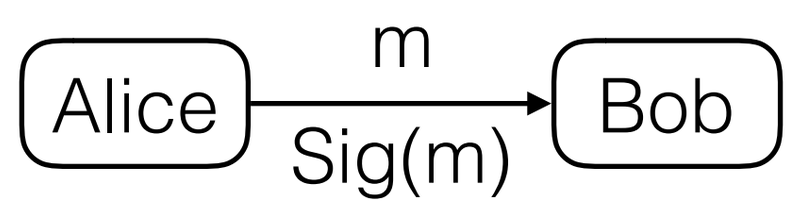
cryptography is a powerful tool
for building secure systems

much of the cryptography used today
offers security properties for **data**

confidentiality



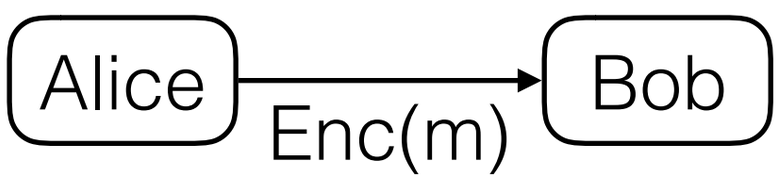
authenticity



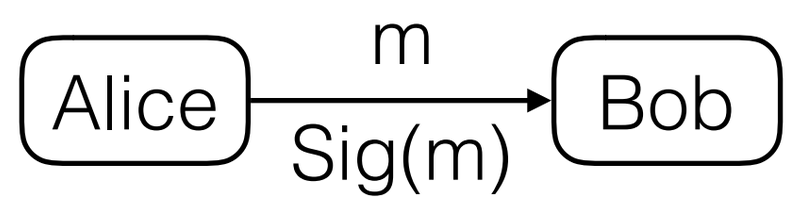
cryptography is a powerful tool
for building secure systems

much of the cryptography used today
offers security properties for **data**

confidentiality



authenticity

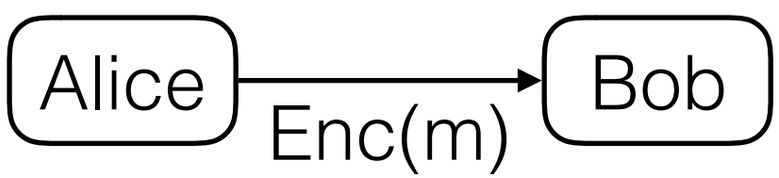


what about security properties for **computation**?

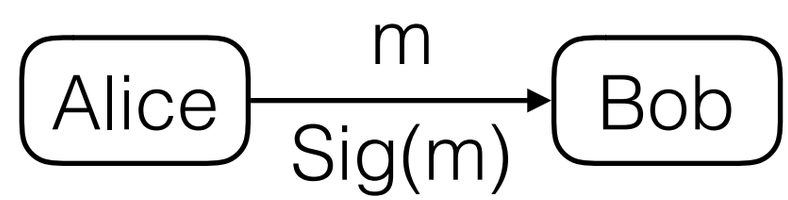
cryptography is a powerful tool
for building secure systems

much of the cryptography used today
offers security properties for **data**

confidentiality



authenticity



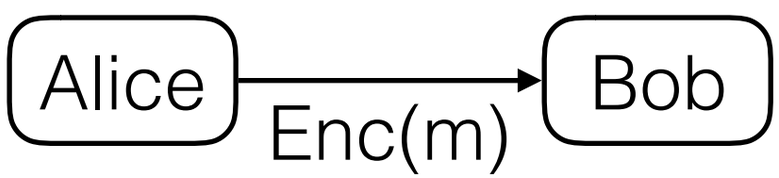
what about security properties for **computation**?

cryptographic proofs offer
privacy-preserving integrity for computation

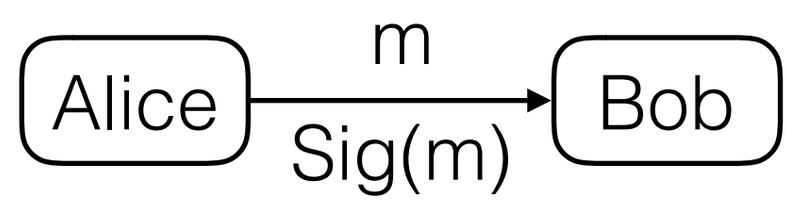
cryptography is a powerful tool
for building secure systems

much of the cryptography used today
offers security properties for **data**

confidentiality



authenticity



what about security properties for **computation**?

cryptographic proofs offer
privacy-preserving integrity for computation

one of the exciting crypto deployment frontiers today

Cryptographic Proofs

Cryptographic Proofs

a powerful defense against malicious behavior
especially, in **distributed protocols**

Cryptographic Proofs

a powerful defense against malicious behavior
especially, in **distributed protocols**

1980s securely compute $\boldsymbol{y}=\boldsymbol{F}(\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n)$
via a multi-party protocol

Cryptographic Proofs

a powerful defense against malicious behavior
especially, in **distributed protocols**

1980s securely compute $y=F(x_1,\ldots,x_n)$
via a multi-party protocol

passive security

active security

Cryptographic Proofs

a powerful defense against malicious behavior
especially, in **distributed protocols**

1980s securely compute $y=F(x_1,\ldots,x_n)$
via a multi-party protocol

passive security

active security

Key properties

- zero knowledge
- proof of knowledge

Cryptographic Proofs

a powerful defense against malicious behavior
especially, in **distributed protocols**

1980s securely compute $y=F(x_1,\ldots,x_n)$
via a multi-party protocol

passive security active security

Key properties

- zero knowledge
- proof of knowledge

2010s blockchain technology

Cryptographic Proofs

a powerful defense against malicious behavior
especially, in **distributed protocols**

1980s securely compute $y=F(x_1,\ldots,x_n)$
via a multi-party protocol

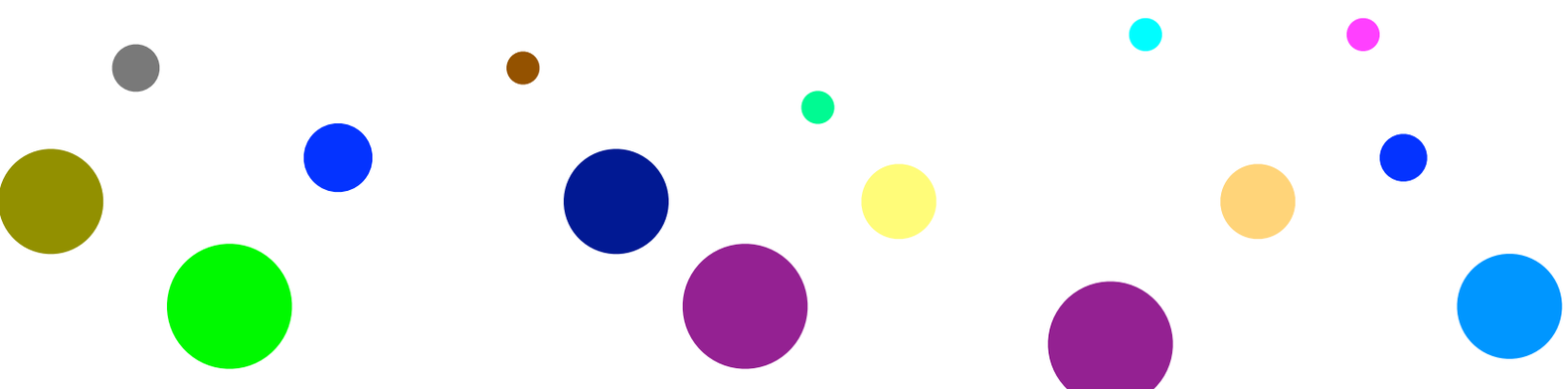
passive security

active security

Key properties

- zero knowledge
- proof of knowledge

2010s blockchain technology



Cryptographic Proofs

a powerful defense against malicious behavior
especially, in **distributed protocols**

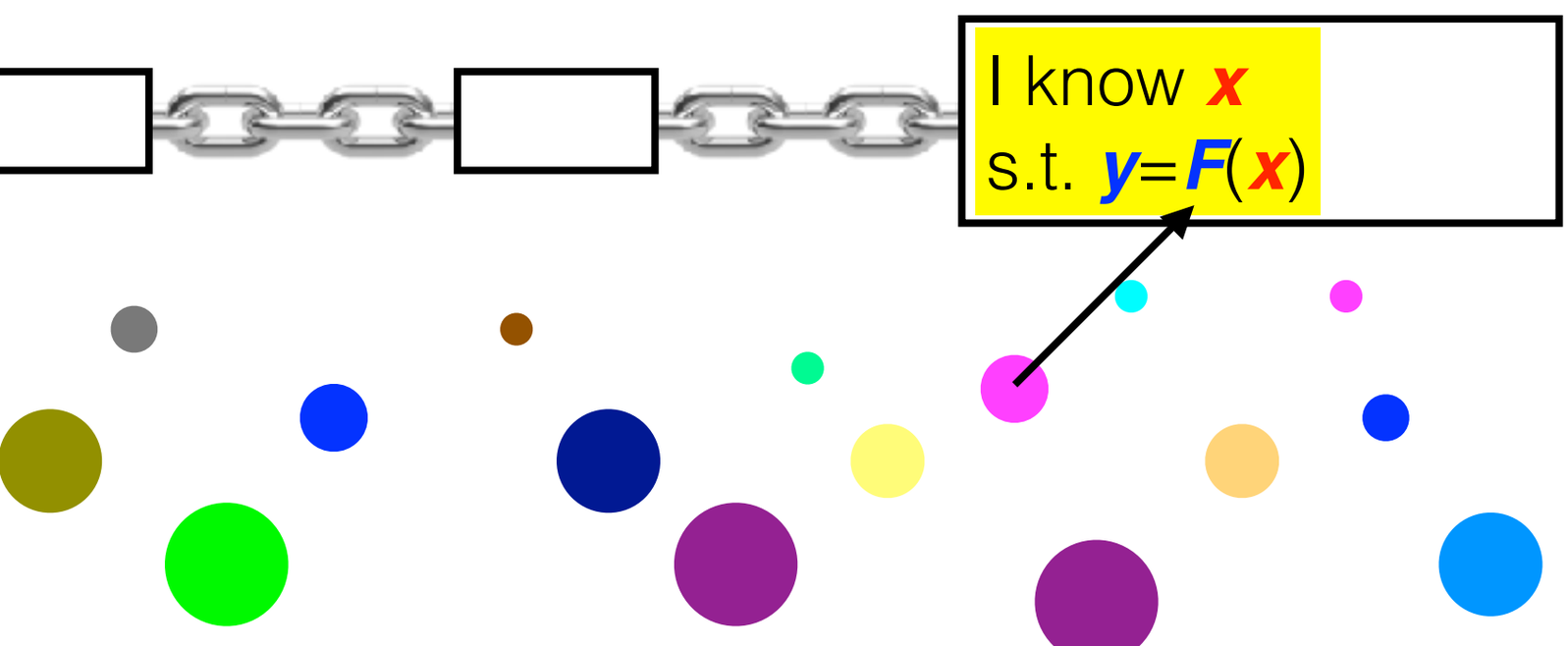
1980s securely compute $y=F(x_1,\ldots,x_n)$
via a multi-party protocol

passive security active security

Key properties

- zero knowledge
- proof of knowledge

2010s blockchain technology



Cryptographic Proofs

a powerful defense against malicious behavior
especially, in **distributed protocols**

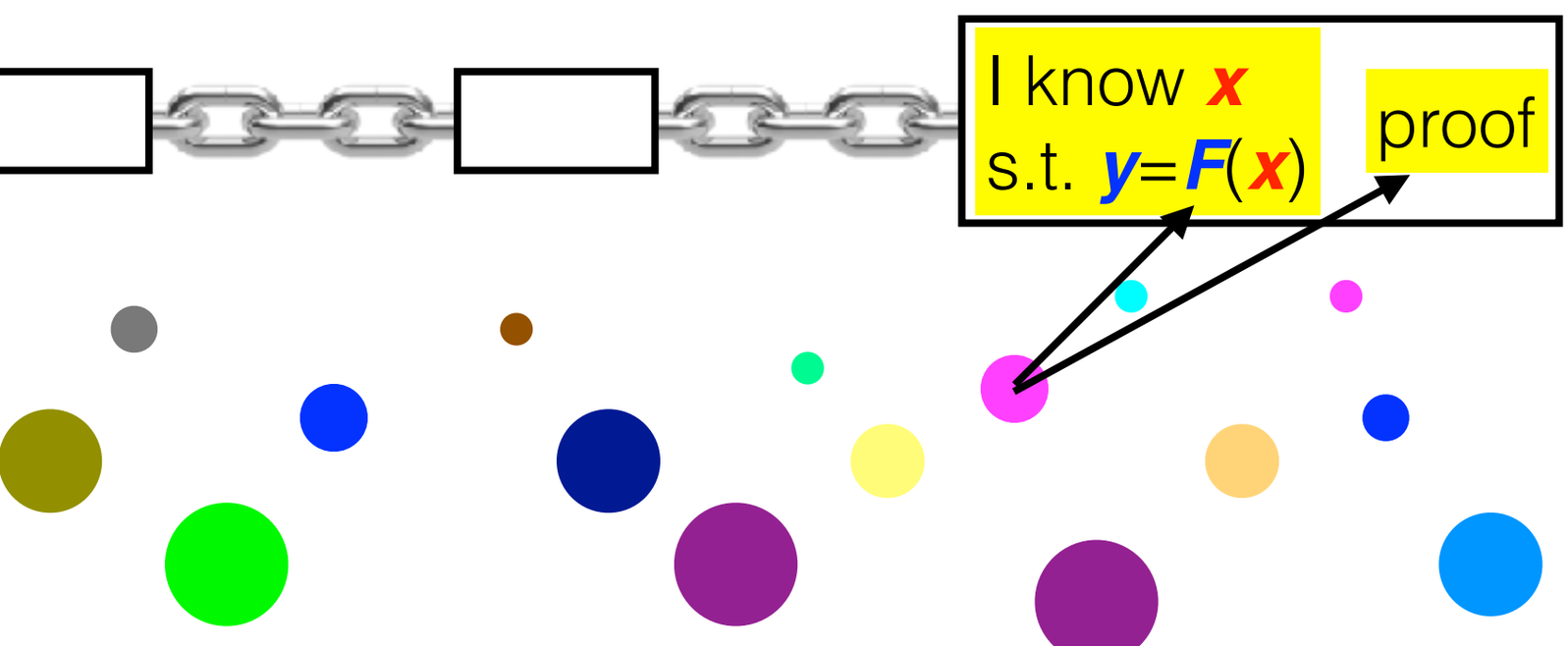
1980s securely compute $y=F(x_1,\ldots,x_n)$
via a multi-party protocol

passive security active security

Key properties

- zero knowledge
- proof of knowledge

2010s blockchain technology



Cryptographic Proofs

a powerful defense against malicious behavior
especially, in **distributed protocols**

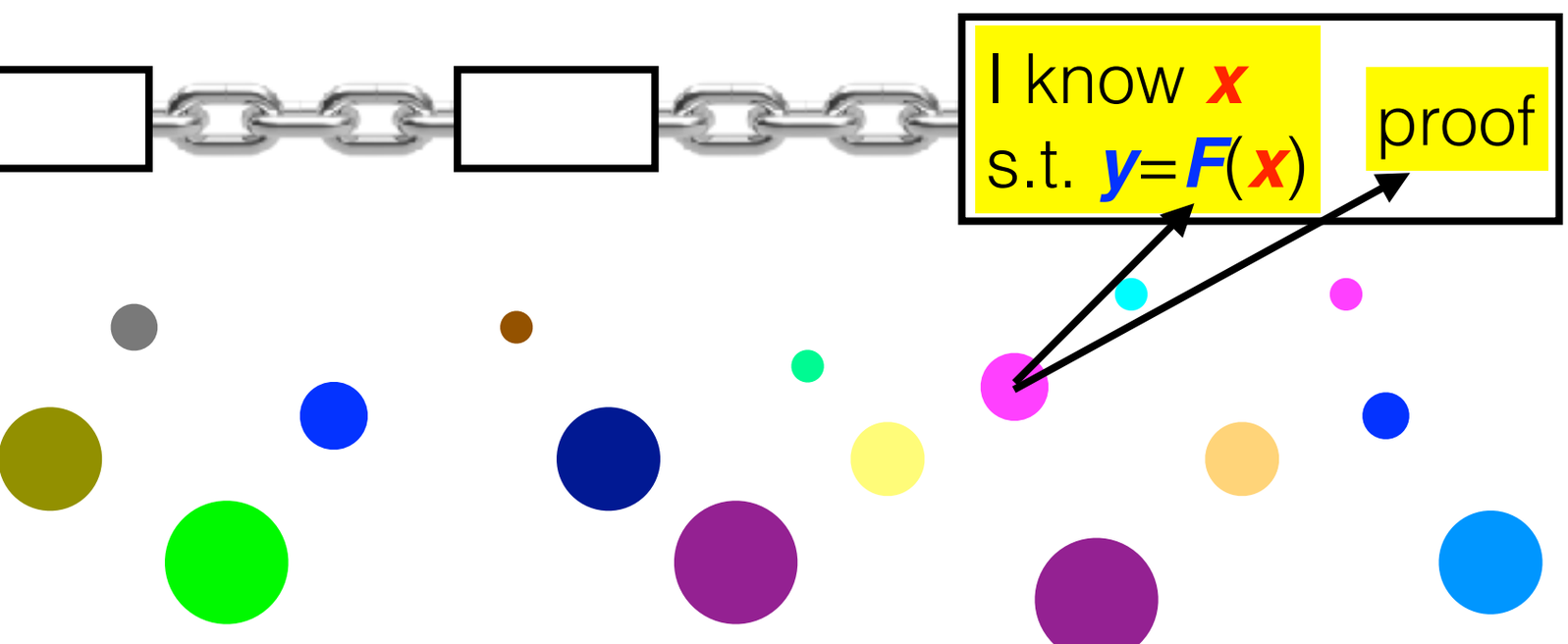
1980s securely compute $y=F(x_1,\ldots,x_n)$
via a multi-party protocol

passive security active security

Key properties

- zero knowledge
- proof of knowledge

2010s blockchain technology



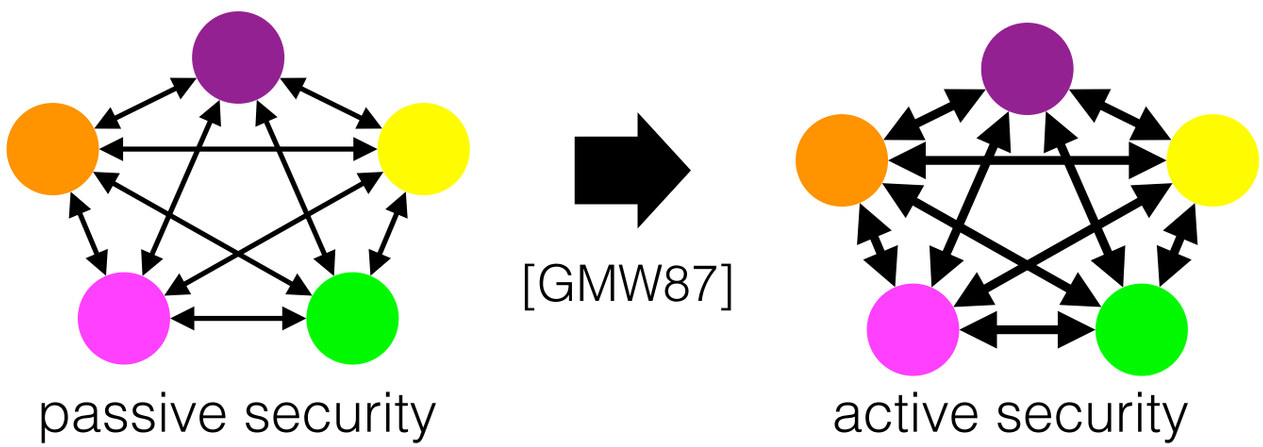
Additional key properties

- non-interactive
- publicly verifiable
- succinct

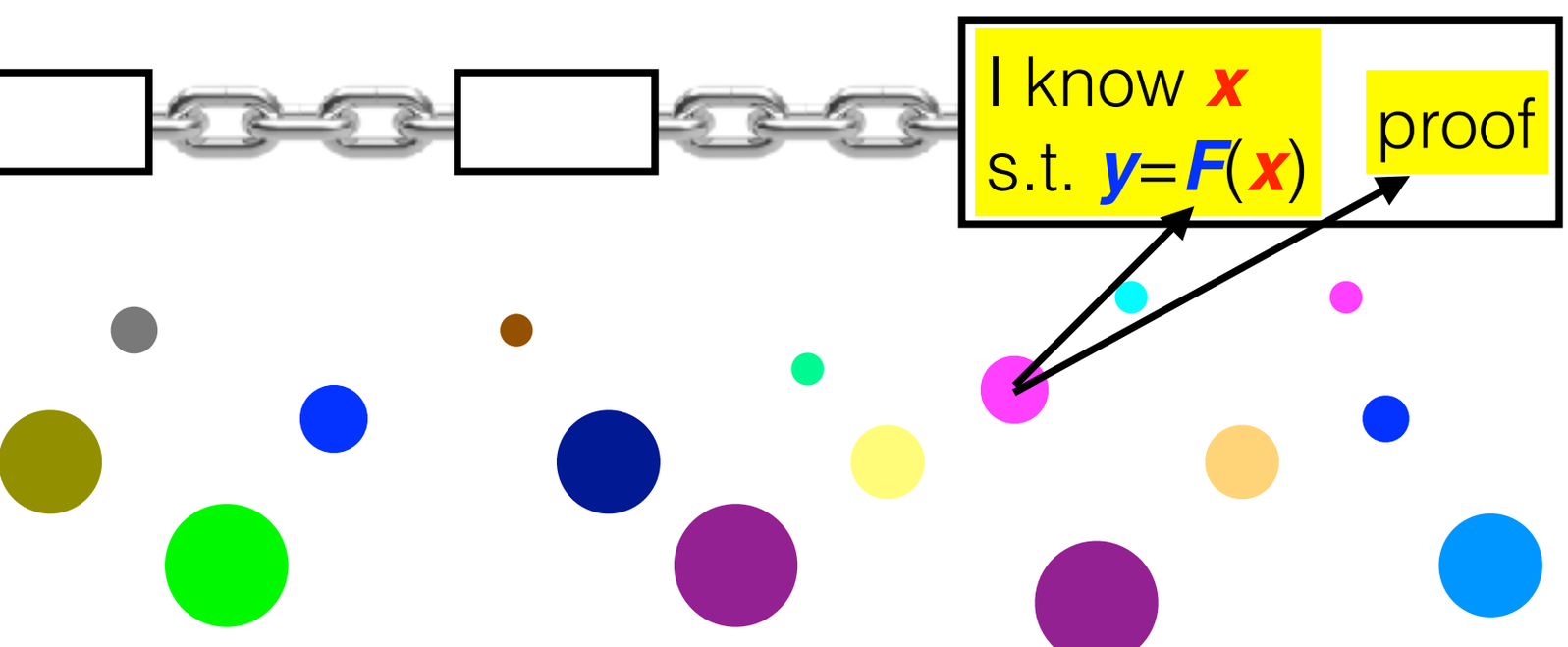
Cryptographic Proofs

a powerful defense against malicious behavior
especially, in **distributed protocols**

1980s securely compute $y=F(x_1,\ldots,x_n)$
via a multi-party protocol



2010s blockchain technology



zk-SNARK

Key properties

- zero knowledge
- proof of knowledge

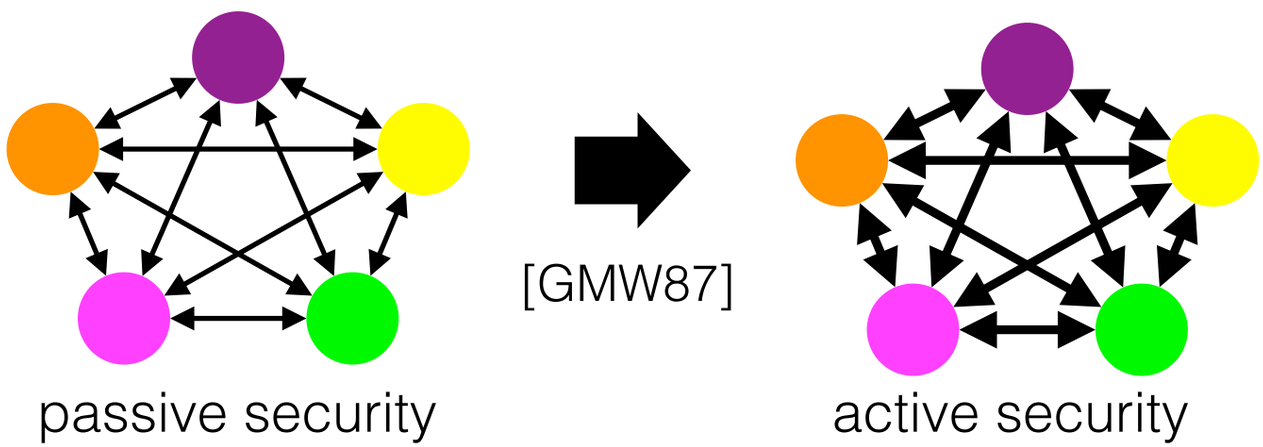
Additional key properties

- non-interactive
- publicly verifiable
- succinct

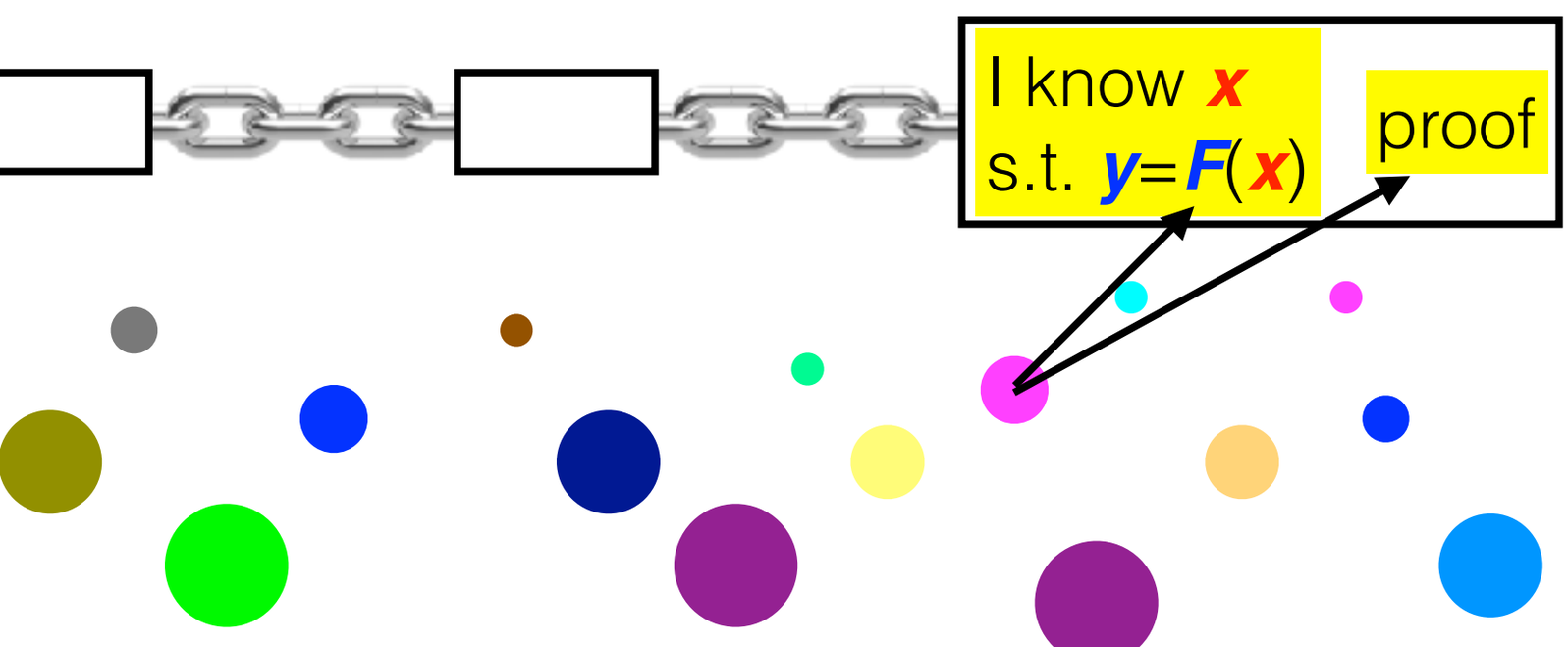
Cryptographic Proofs

a powerful defense against malicious behavior
especially, in **distributed protocols**

1980s securely compute $y=F(x_1,\ldots,x_n)$
via a multi-party protocol



2010s blockchain technology



zk-SNARK

Key properties

- zero knowledge
- ~~proof of knowledge~~

Additional key properties

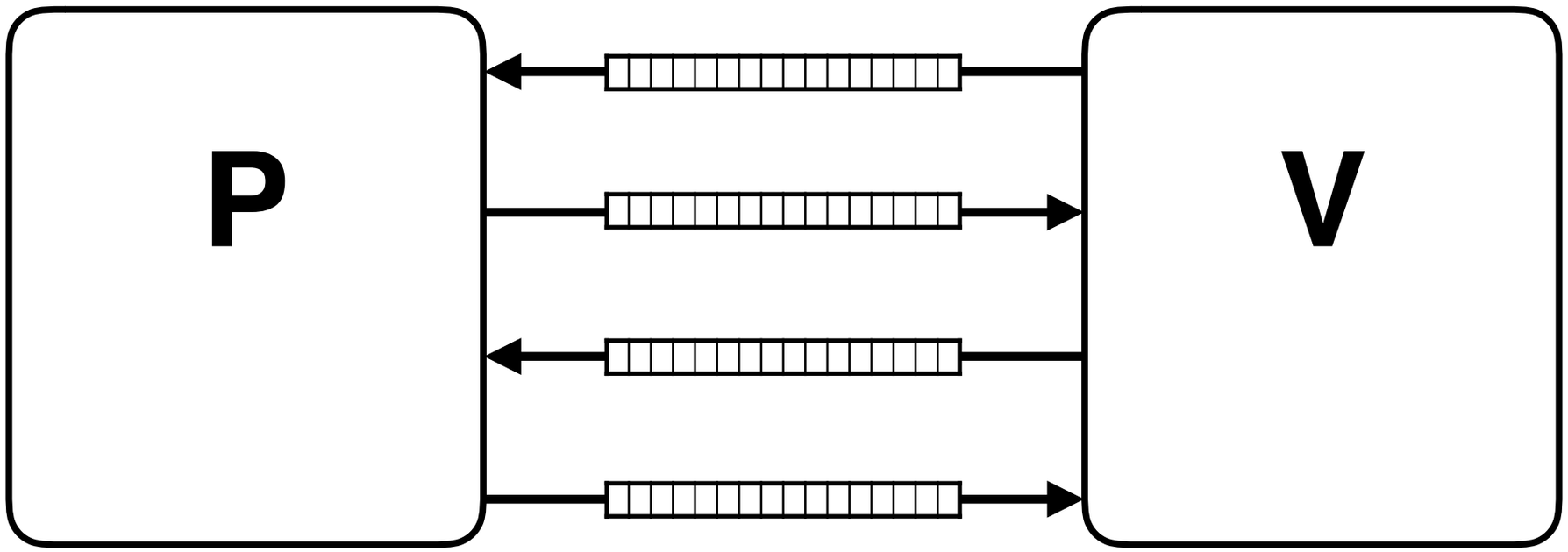
- non-interactive
- publicly verifiable
- succinct

Zero Knowledge Proofs

[GMR85]

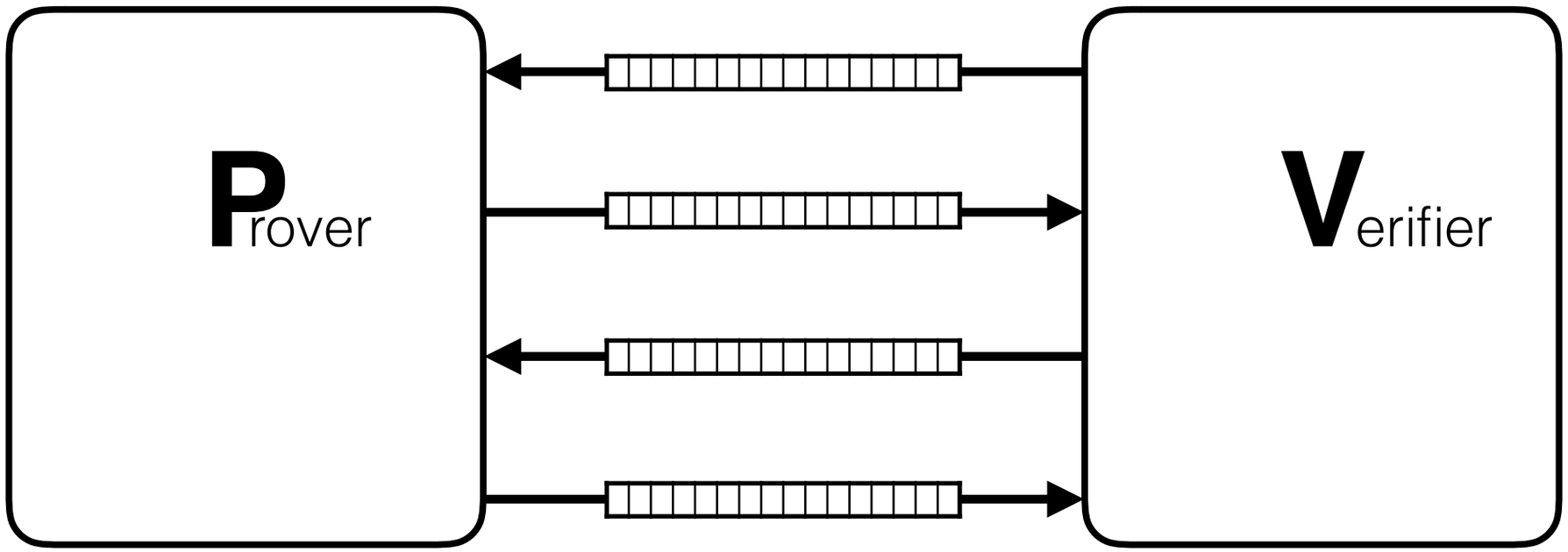
Zero Knowledge Proofs

[GMR85]



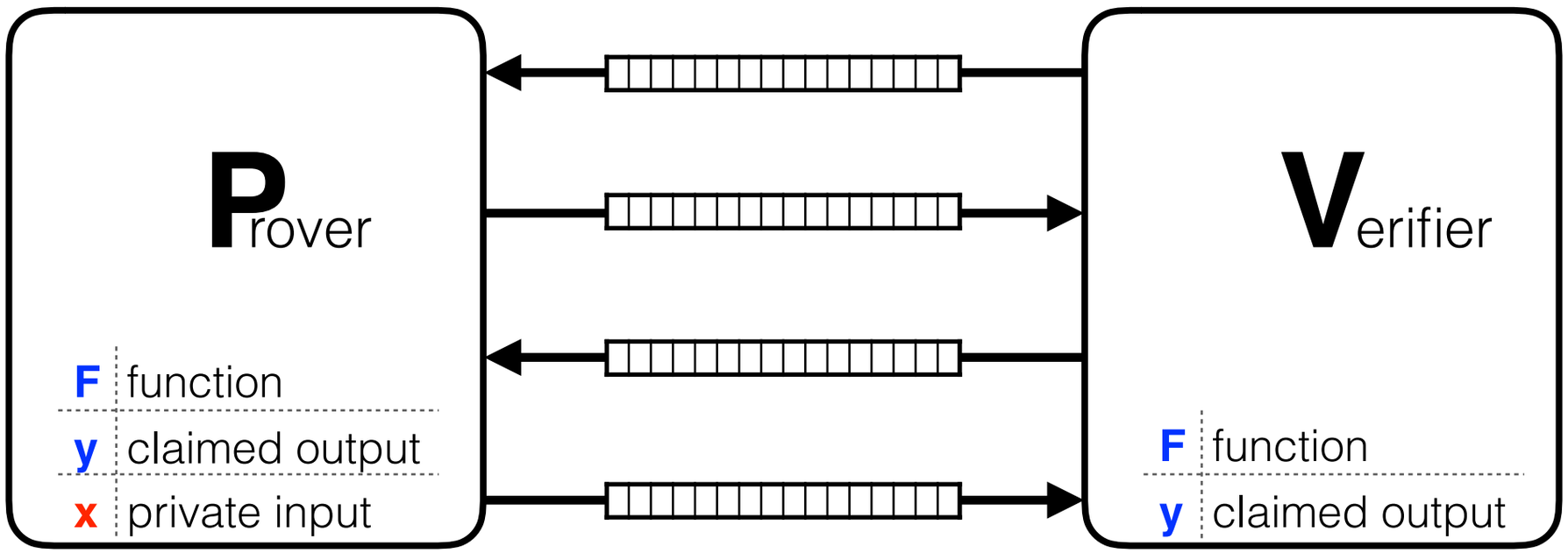
Zero Knowledge Proofs

[GMR85]



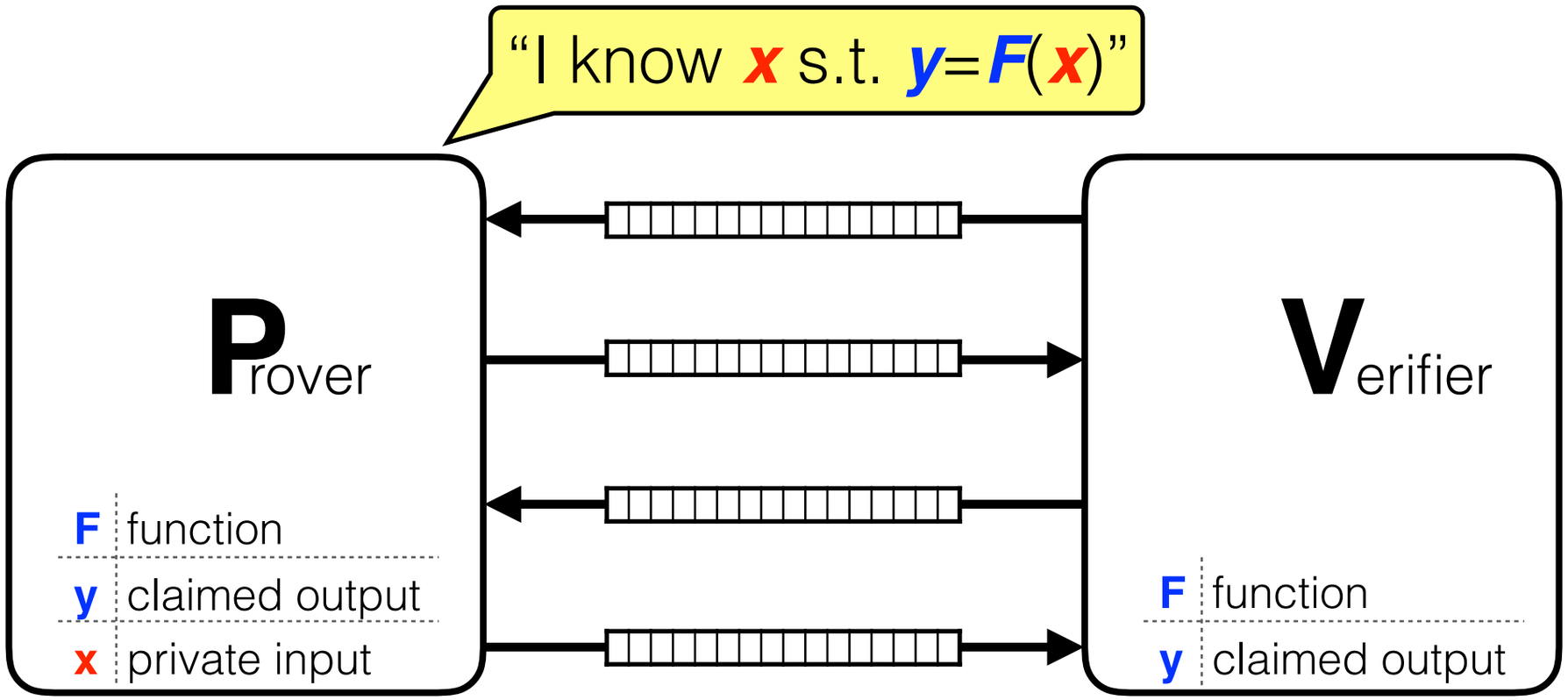
Zero Knowledge Proofs

[GMR85]



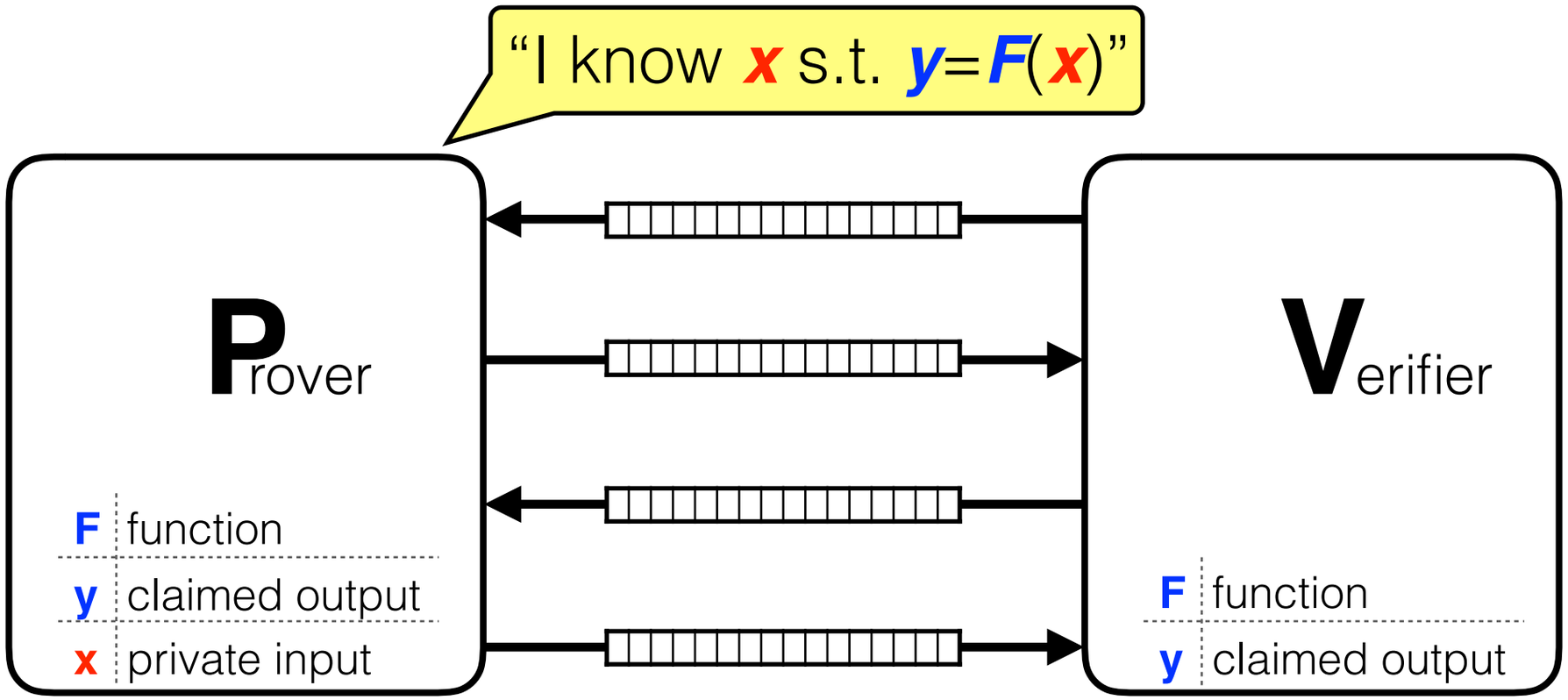
Zero Knowledge Proofs

[GMR85]



Zero Knowledge Proofs

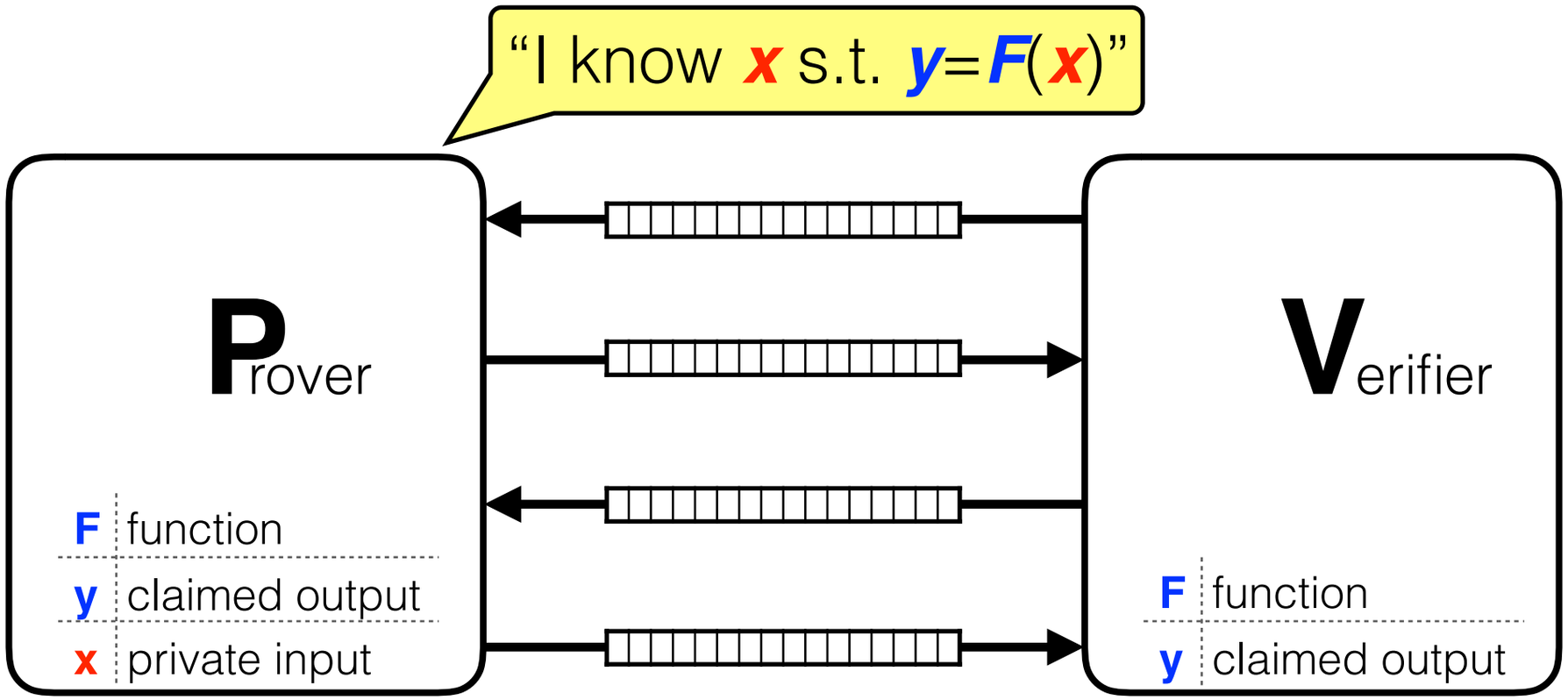
[GMR85]



completeness $\exists\, \boldsymbol{x}: \boldsymbol{y}=\boldsymbol{F}(\boldsymbol{x}) \rightarrow \Pr[\mathbf{P}(\boldsymbol{F},\boldsymbol{y},\boldsymbol{x}) \text{ convinces } \mathbf{V}(\boldsymbol{F},\boldsymbol{y})]=1$

Zero Knowledge Proofs

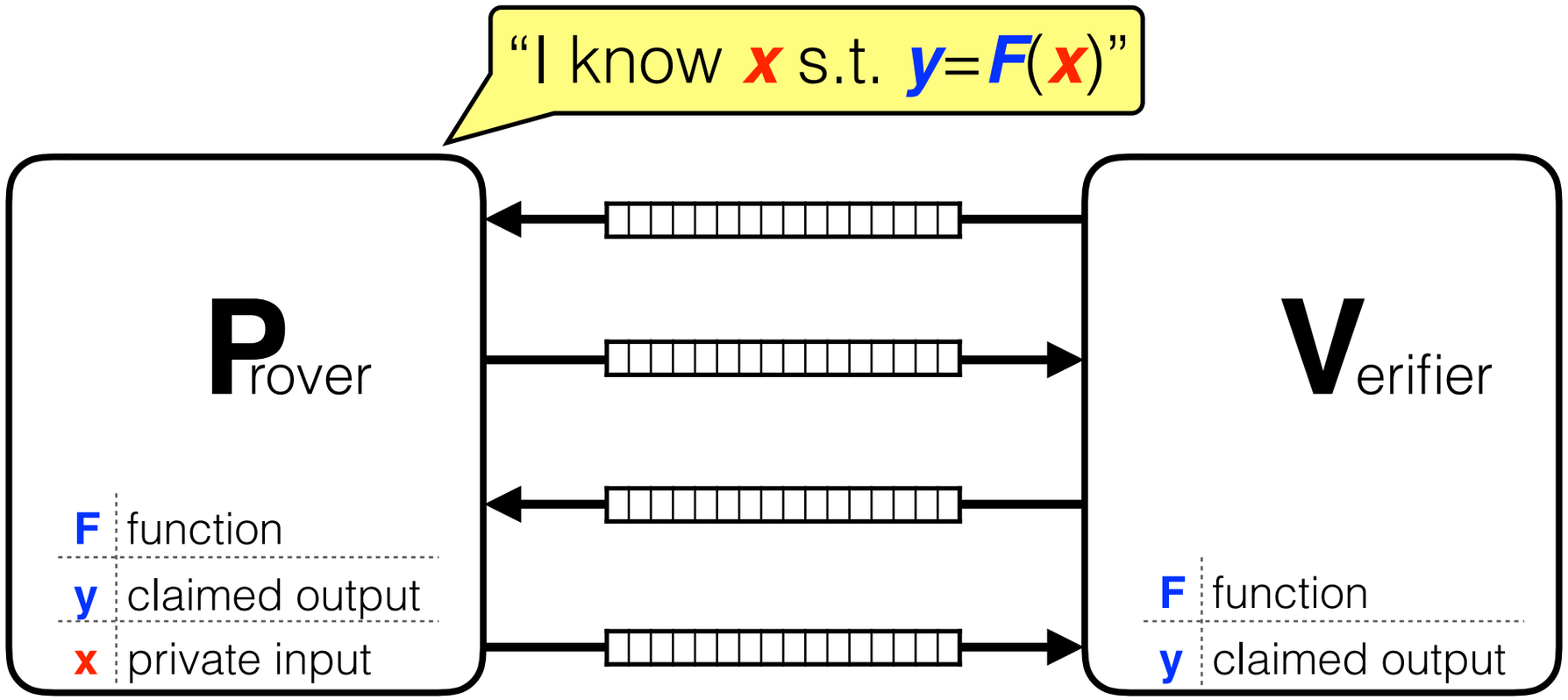
[GMR85]



completeness	$\exists\, \boldsymbol{x}$: $\boldsymbol{y}=\boldsymbol{F}(\boldsymbol{x}) \rightarrow \Pr[\mathbf{P}(\boldsymbol{F},\boldsymbol{y},\boldsymbol{x})$ convinces $\mathbf{V}(\boldsymbol{F},\boldsymbol{y})]=1$
soundness	$\nexists\, \boldsymbol{x}$: $\boldsymbol{y}=\boldsymbol{F}(\boldsymbol{x}) \rightarrow \forall\, \mathbf{P'}$, $\Pr[\mathbf{P'}$ convinces $\mathbf{V}(\boldsymbol{F},\boldsymbol{y})]\simeq 0$

Zero Knowledge Proofs

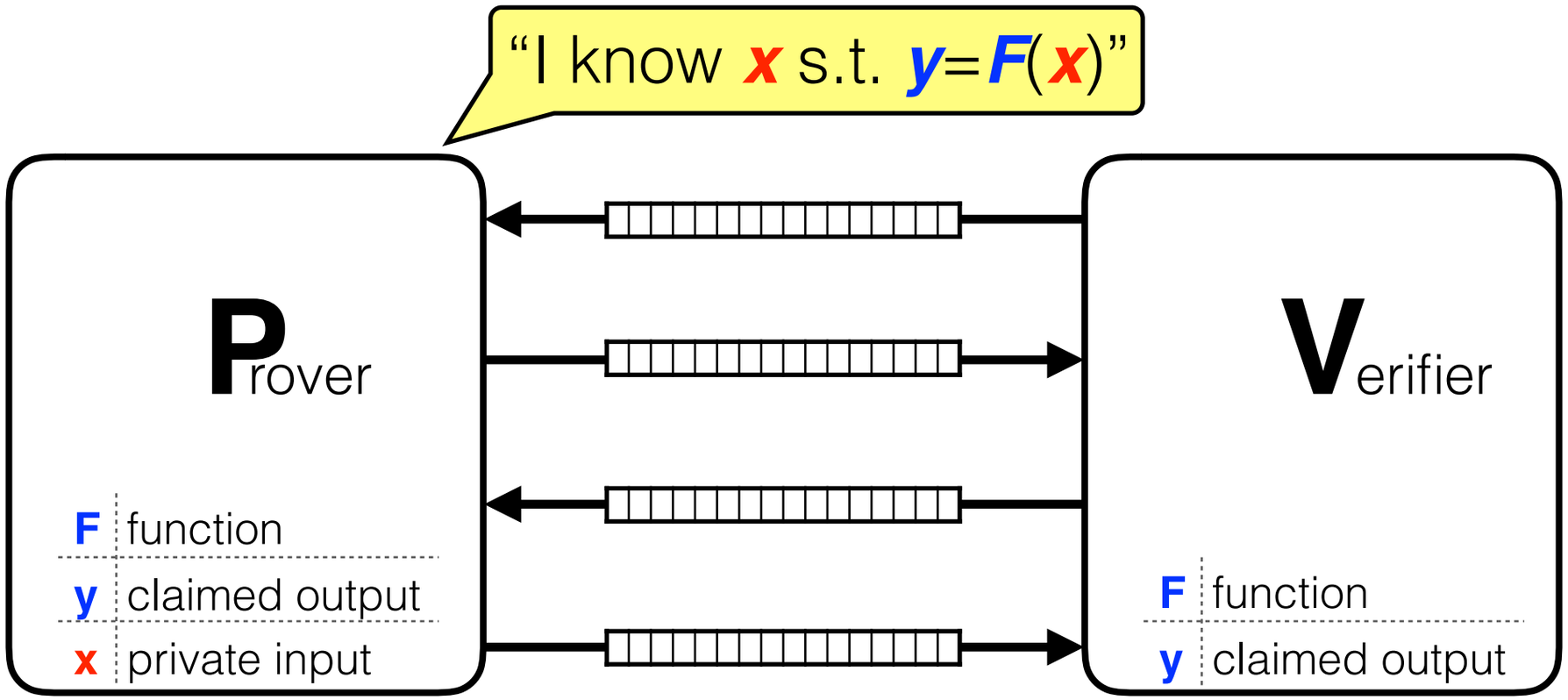
[GMR85]



completeness	$\exists\, \boldsymbol{x}$: $\boldsymbol{y}=\boldsymbol{F}(\boldsymbol{x}) \rightarrow \Pr[\mathbf{P}(\boldsymbol{F},\boldsymbol{y},\boldsymbol{x})$ convinces $\mathbf{V}(\boldsymbol{F},\boldsymbol{y})]=1$
soundness	$\nexists\, \boldsymbol{x}$: $\boldsymbol{y}=\boldsymbol{F}(\boldsymbol{x}) \rightarrow \forall\, \mathbf{P'}$, $\Pr[\mathbf{P'}$ convinces $\mathbf{V}(\boldsymbol{F},\boldsymbol{y})]\simeq 0$
zero knowledge	$\exists\, \boldsymbol{x}$: $\boldsymbol{y}=\boldsymbol{F}(\boldsymbol{x}) \rightarrow \forall\, \mathbf{V'}$, $\mathbf{S}(\mathbf{V'},\boldsymbol{F},\boldsymbol{y}) \simeq$ view of $\mathbf{V'}$ with $\mathbf{P}(\boldsymbol{F},\boldsymbol{y},\boldsymbol{x})$

Zero Knowledge Proofs

[GMR85]

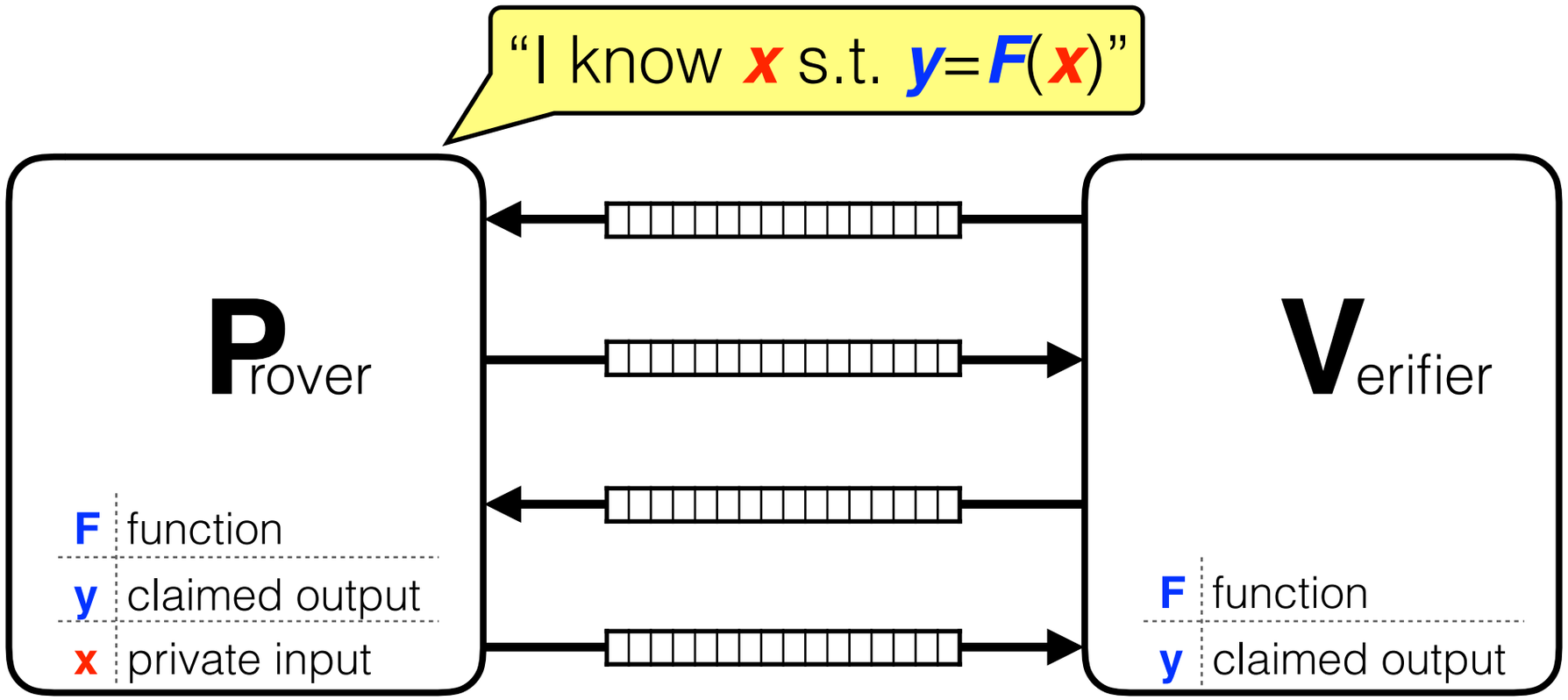


completeness	$\exists\, \boldsymbol{x}$: $\boldsymbol{y}=\boldsymbol{F}(\boldsymbol{x}) \rightarrow \Pr[\mathbf{P}(\boldsymbol{F},\boldsymbol{y},\boldsymbol{x})$ convinces $\mathbf{V}(\boldsymbol{F},\boldsymbol{y})]=1$
soundness	$\nexists\, \boldsymbol{x}$: $\boldsymbol{y}=\boldsymbol{F}(\boldsymbol{x}) \rightarrow \forall\, \mathbf{P'}$, $\Pr[\mathbf{P'}$ convinces $\mathbf{V}(\boldsymbol{F},\boldsymbol{y})]\simeq 0$
zero knowledge	$\exists\, \boldsymbol{x}$: $\boldsymbol{y}=\boldsymbol{F}(\boldsymbol{x}) \rightarrow \forall\, \mathbf{V'}$, $\mathbf{S}(\mathbf{V'},\boldsymbol{F},\boldsymbol{y}) \simeq$ view of $\mathbf{V'}$ with $\mathbf{P}(\boldsymbol{F},\boldsymbol{y},\boldsymbol{x})$

simulator

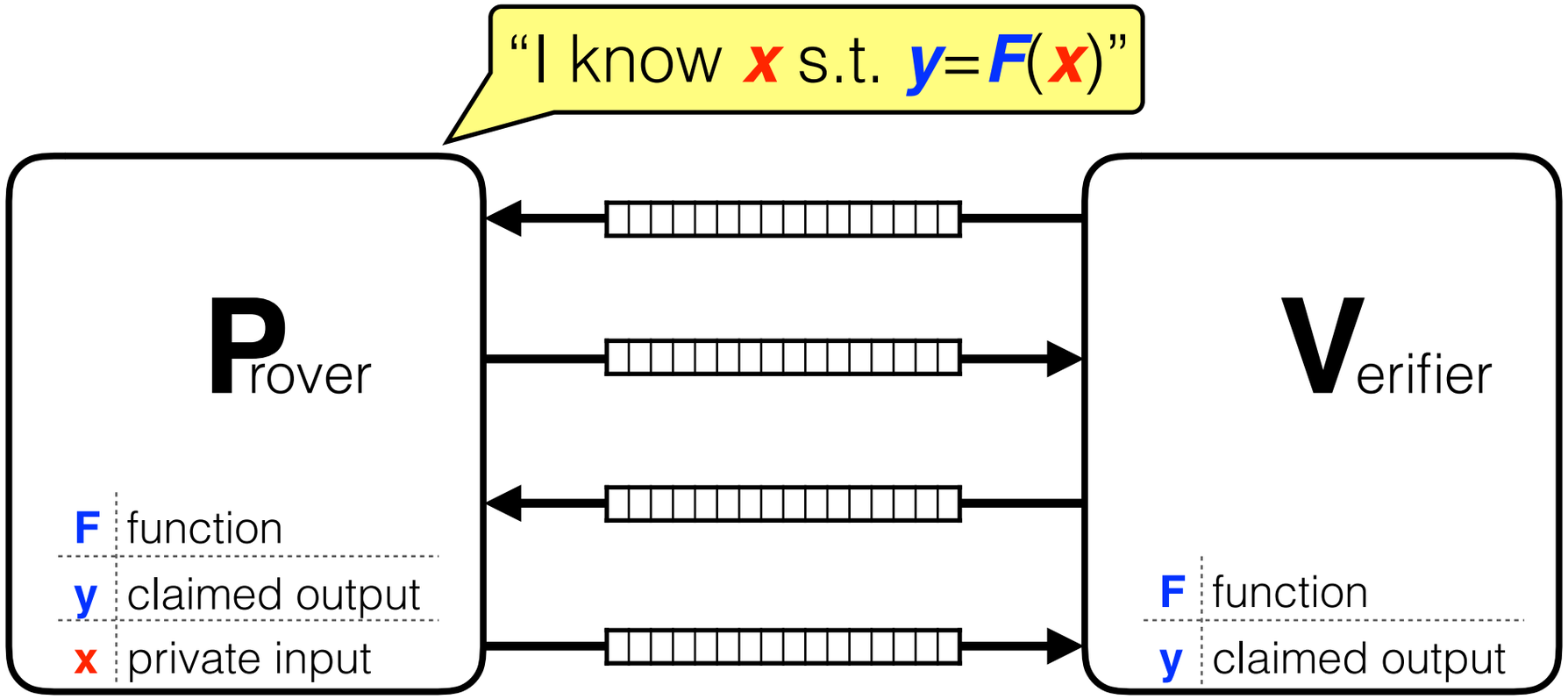
Zero Knowledge Proofs

[GMR85]



Zero Knowledge Proofs

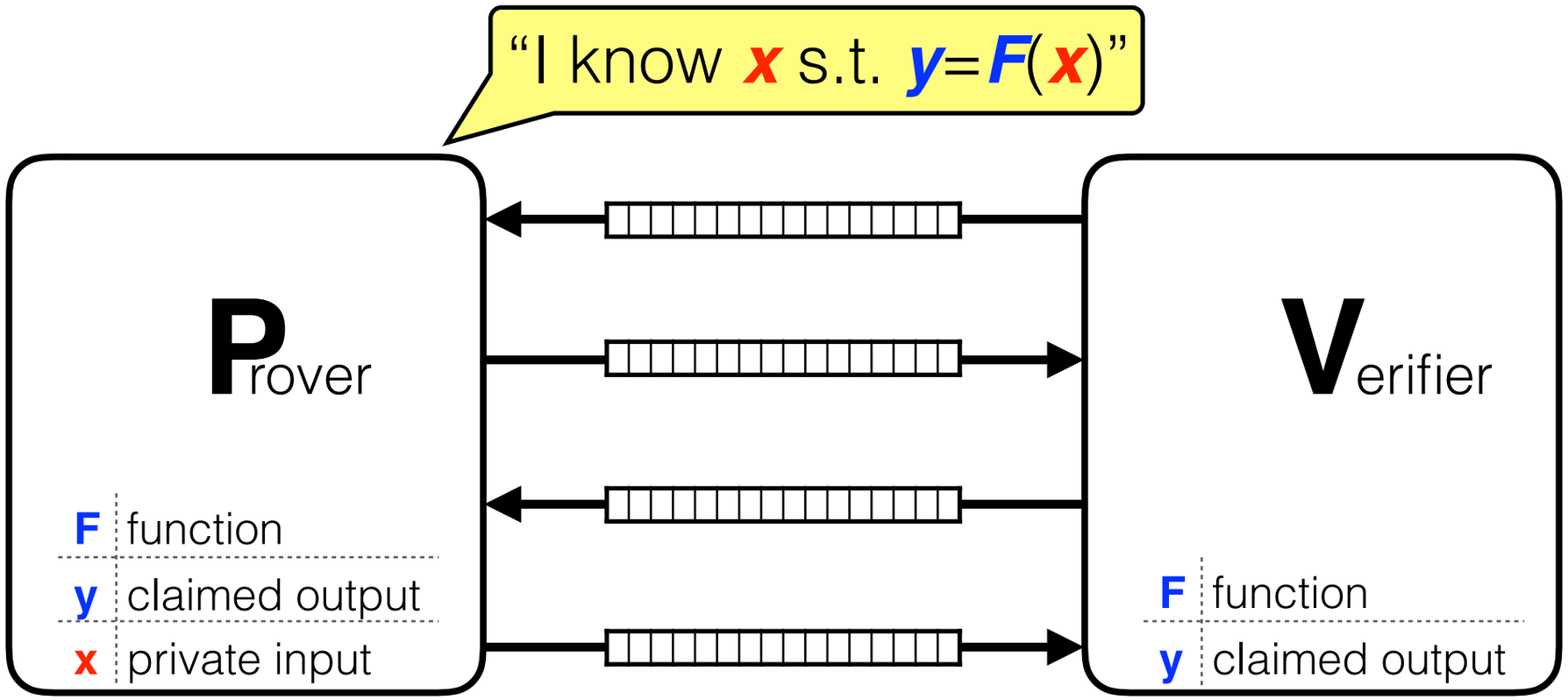
[GMR85]



[GMR85]: ZKPs for certain number-theoretic problems (QR,QNR)

Zero Knowledge Proofs

[GMR85]

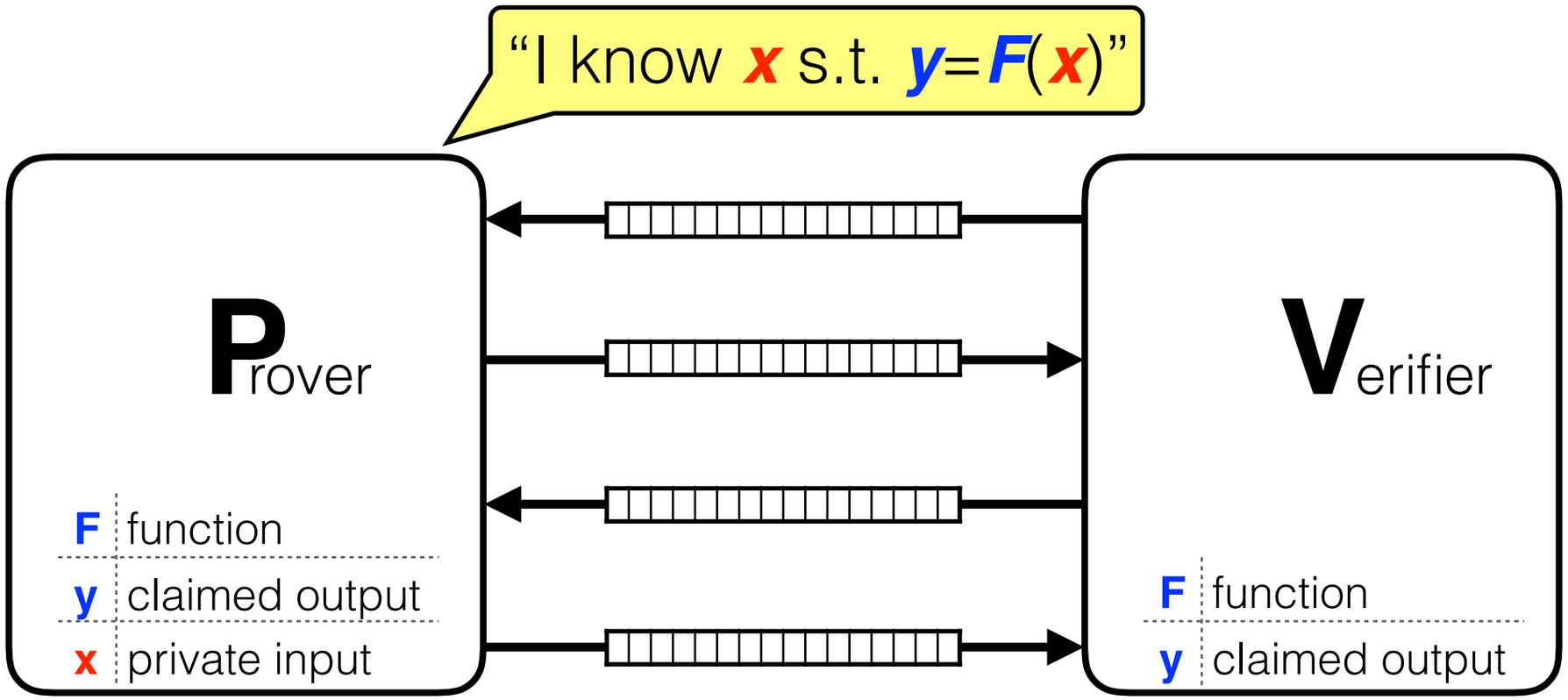


[GMR85]: ZKPs for certain number-theoretic problems (QR,QNR)

If one-way functions exist:

Zero Knowledge Proofs

[GMR85]



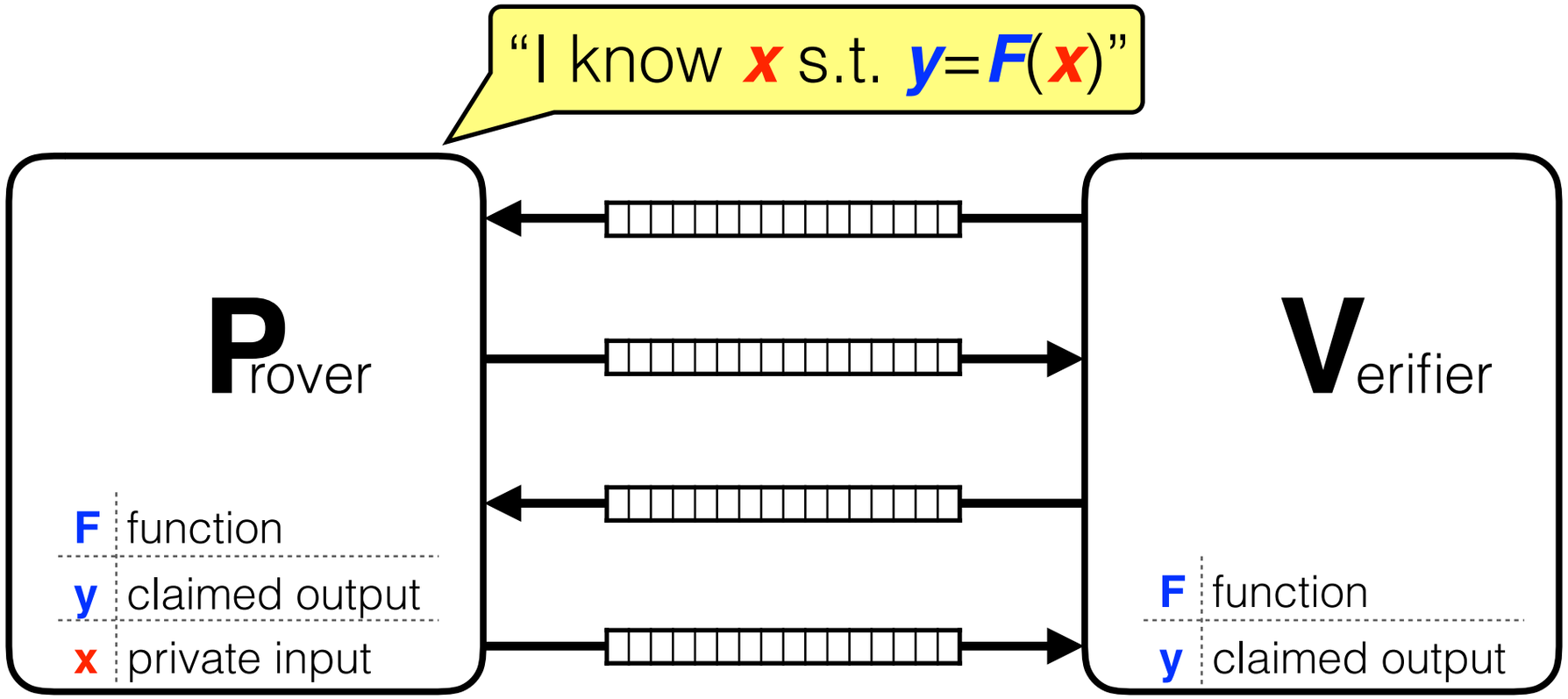
[GMR85]: ZKPs for certain number-theoretic problems (QR,QNR)

If one-way functions exist:

[GMW86]: ZKPs for all poly-**time** computable functions **F**

Zero Knowledge Proofs

[GMR85]



[GMR85]: ZKPs for certain number-theoretic problems (QR,QNR)

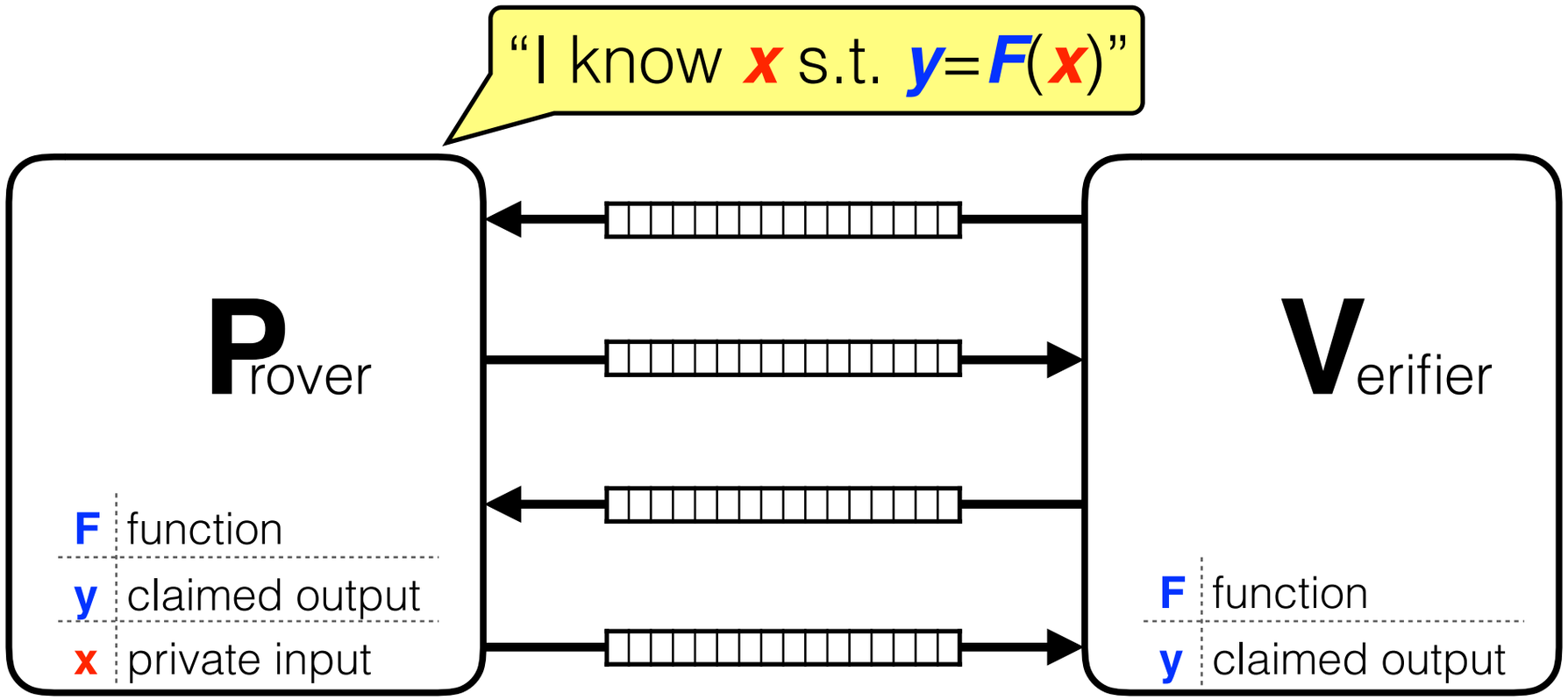
If one-way functions exist:

[GMW86]: ZKPs for all poly-**time** computable functions **F**

[BGGHKMR88]: ZKPs for all poly-**space** computable functions **F**

Zero Knowledge Proofs

[GMR85]



[GMR85]: ZKPs for certain number-theoretic problems (QR,QNR)

If one-way functions exist:

[GMW86]: ZKPs for all poly-**time** computable functions **F**

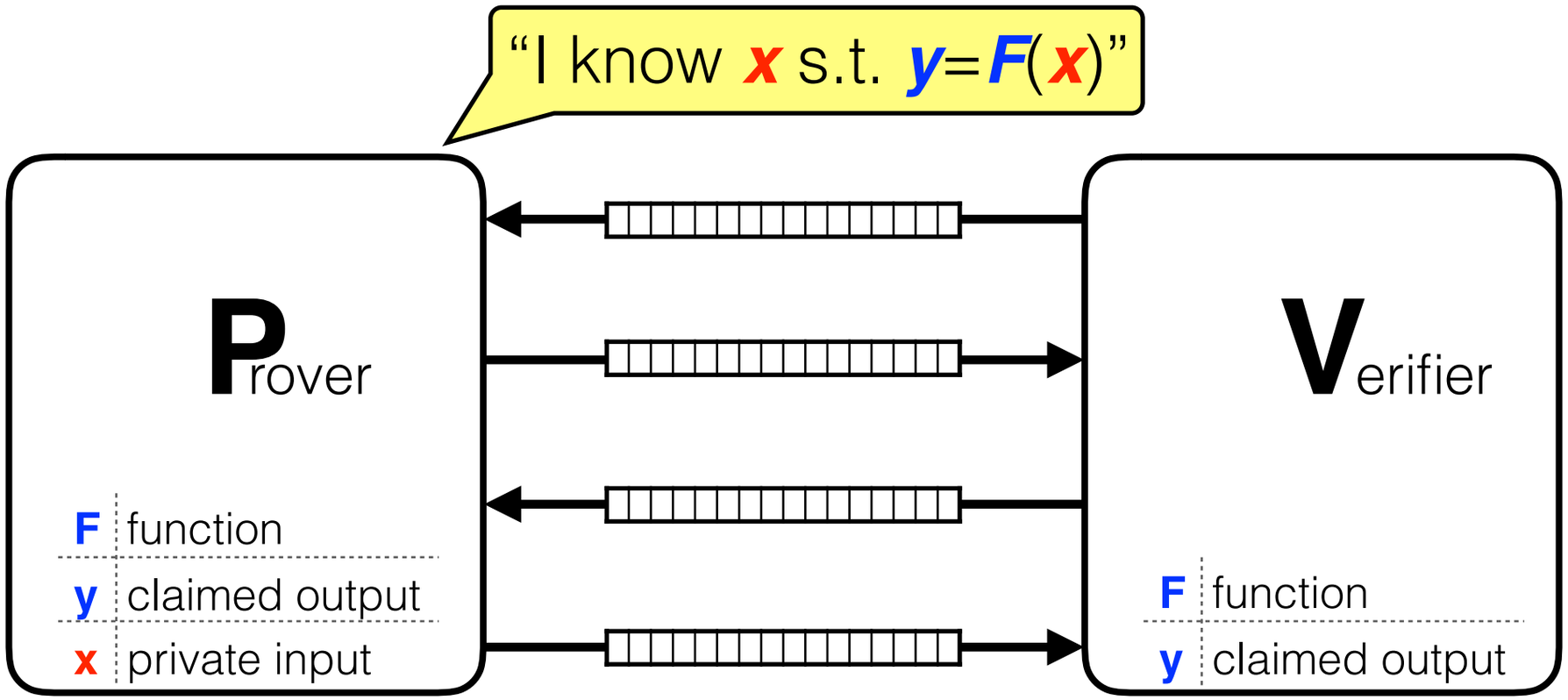
[BGGHKMR88]: ZKPs for all poly-**space** computable functions **F**

Everything Provable is Provable in Zero-Knowledge

Michael Ben-Or	Hebrew University
Oded Goldreich	Technion – Israel Institute of Technology
Shafi Goldwasser	M.I.T. Laboratory for Computer Science
Johan Håstad	Royal Institute of Technology, Sweden
Joe Kilian	M.I.T. Laboratory for Computer Science
Silvio Micali	M.I.T. Laboratory for Computer Science
Phillip Rogaway	M.I.T. Laboratory for Computer Science

Zero Knowledge Proofs

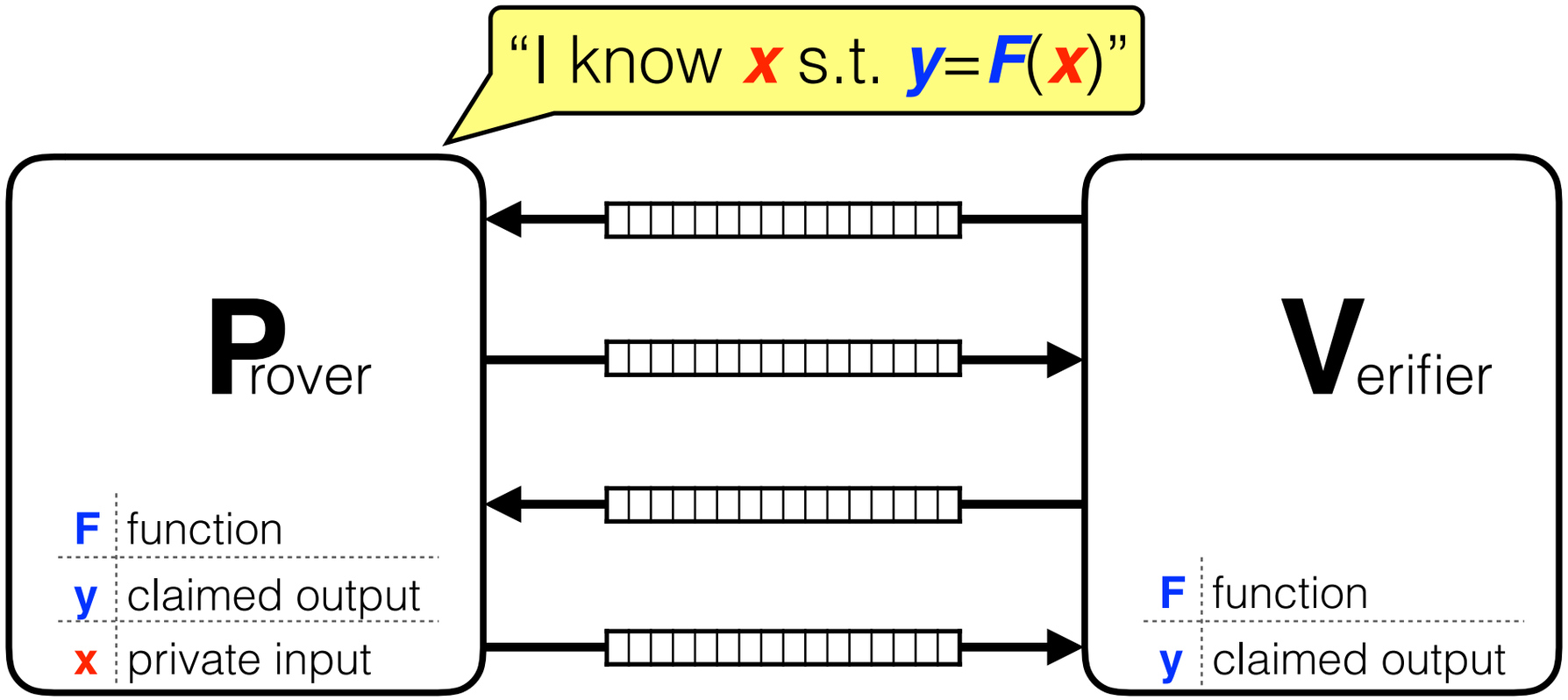
[GMR85]



Powerful cryptographic primitive.

Zero Knowledge Proofs

[GMR85]

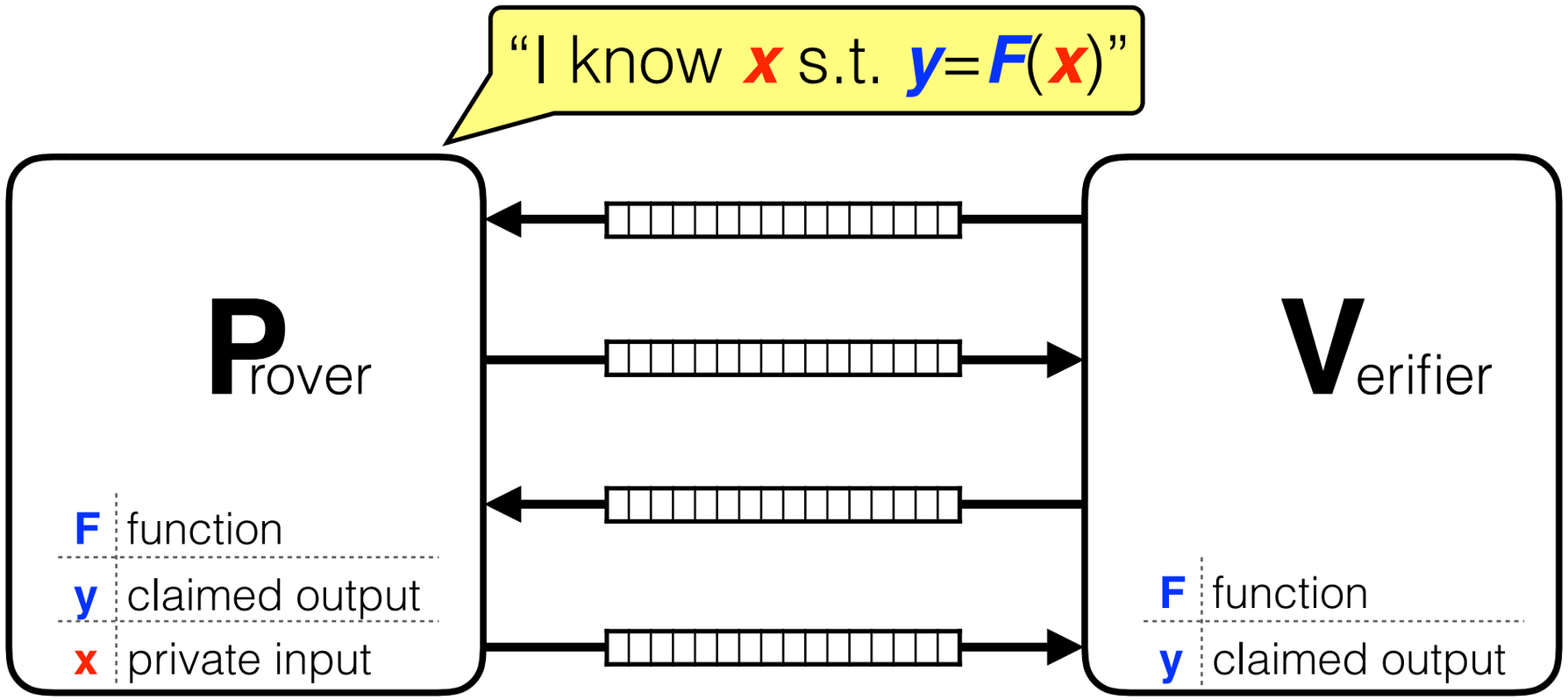


Powerful cryptographic primitive.

BUT

Zero Knowledge Proofs

[GMR85]



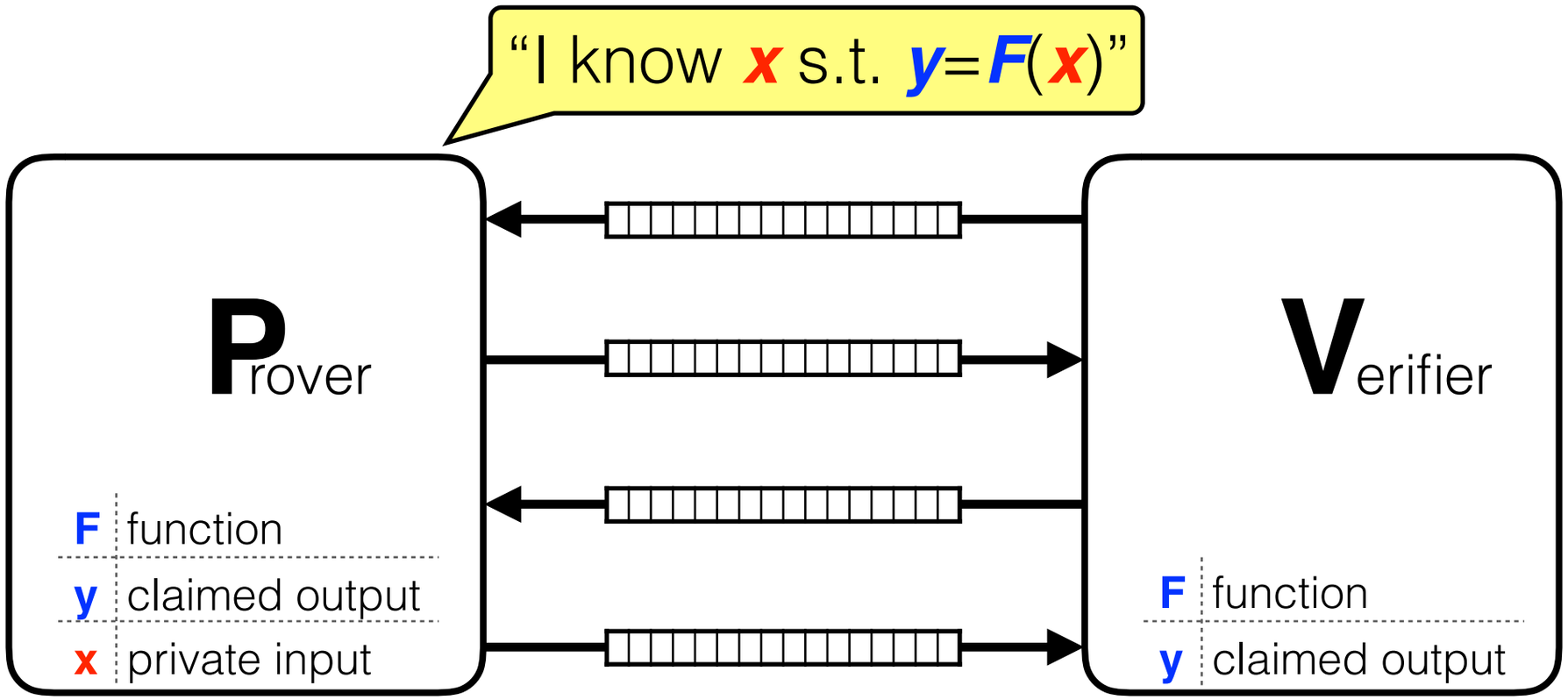
Powerful cryptographic primitive.

BUT

interactive

Zero Knowledge Proofs

[GMR85]



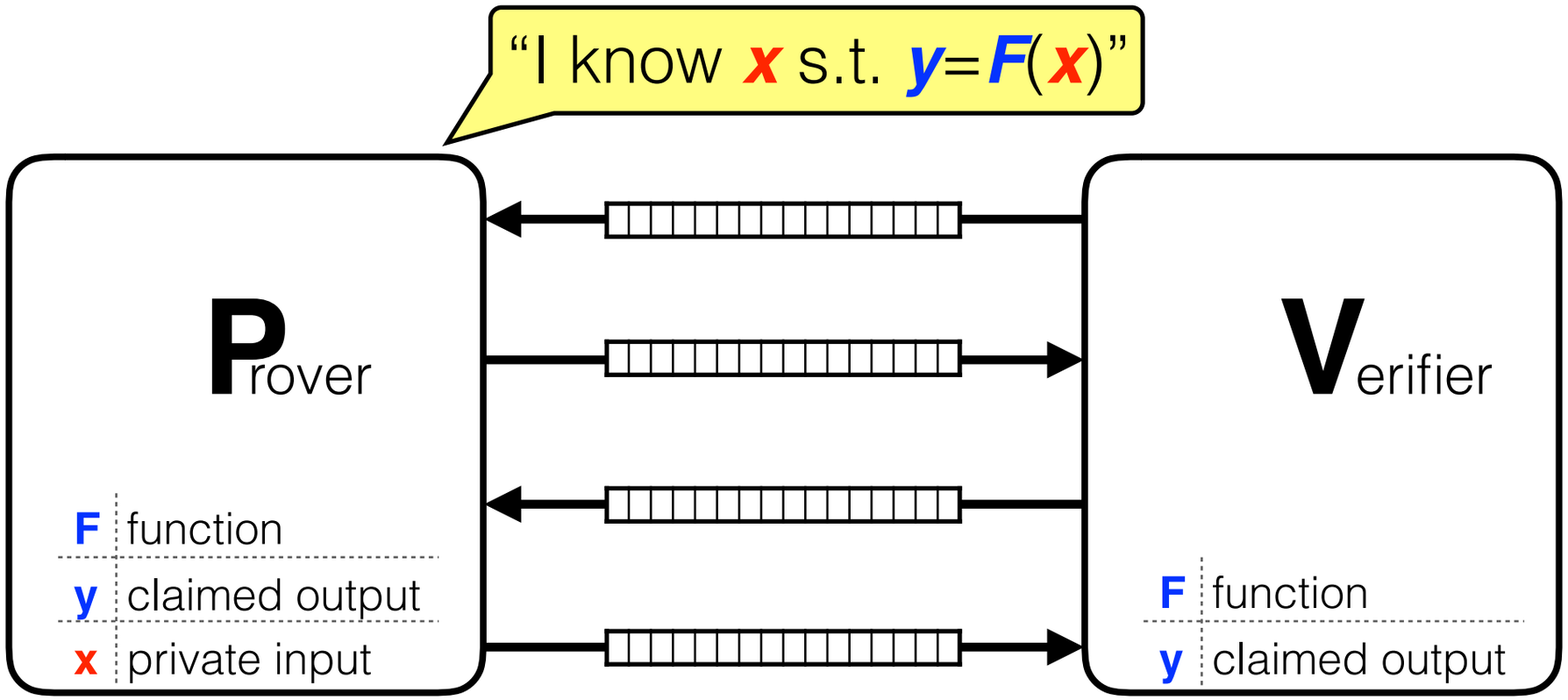
Powerful cryptographic primitive.

BUT

interactive **not succinct**

Zero Knowledge Proofs

[GMR85]



Powerful cryptographic primitive.

BUT

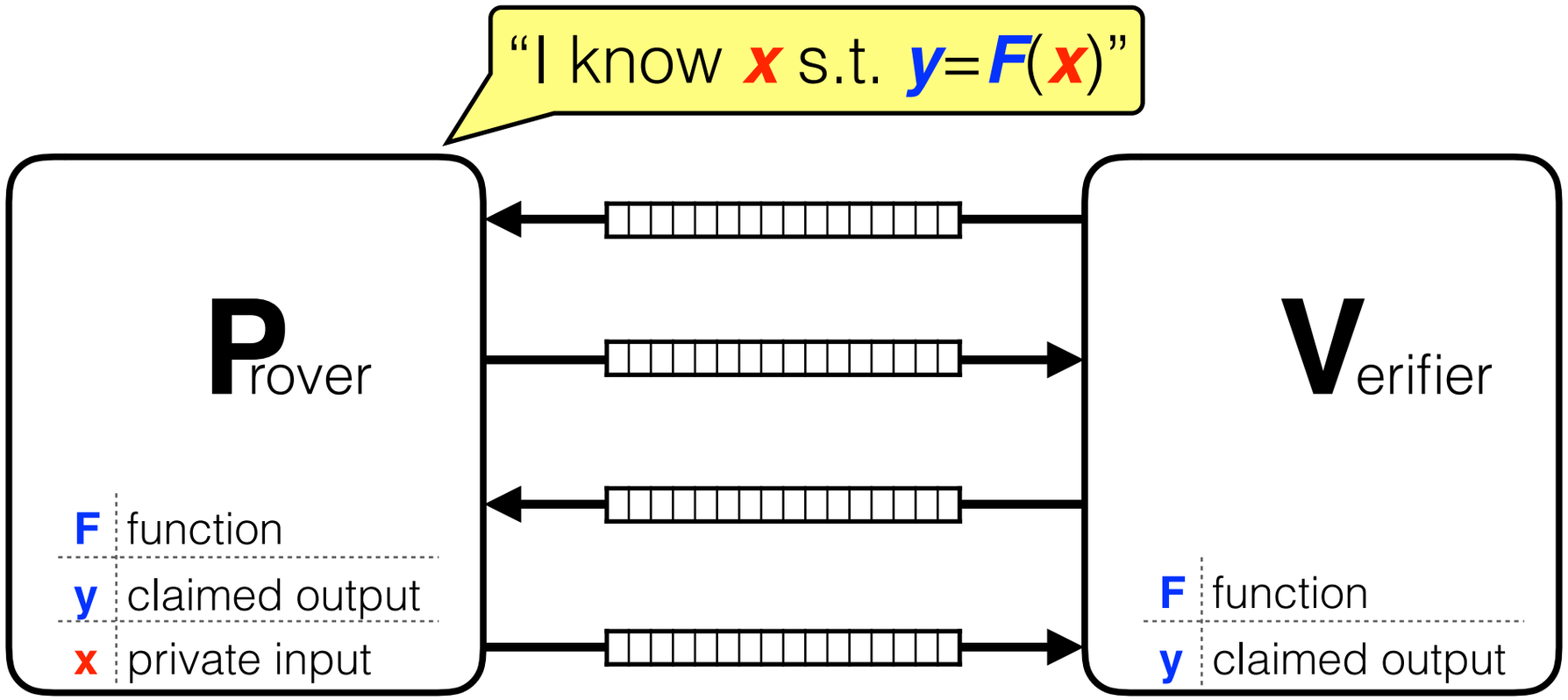
interactive

not succinct

communication complexity
& verification complexity
are proportional to time(**F**)

Zero Knowledge Proofs

[GMR85]



Powerful cryptographic primitive.

BUT

interactive **not succinct**

communication complexity & verification complexity are proportional to time(**F**)

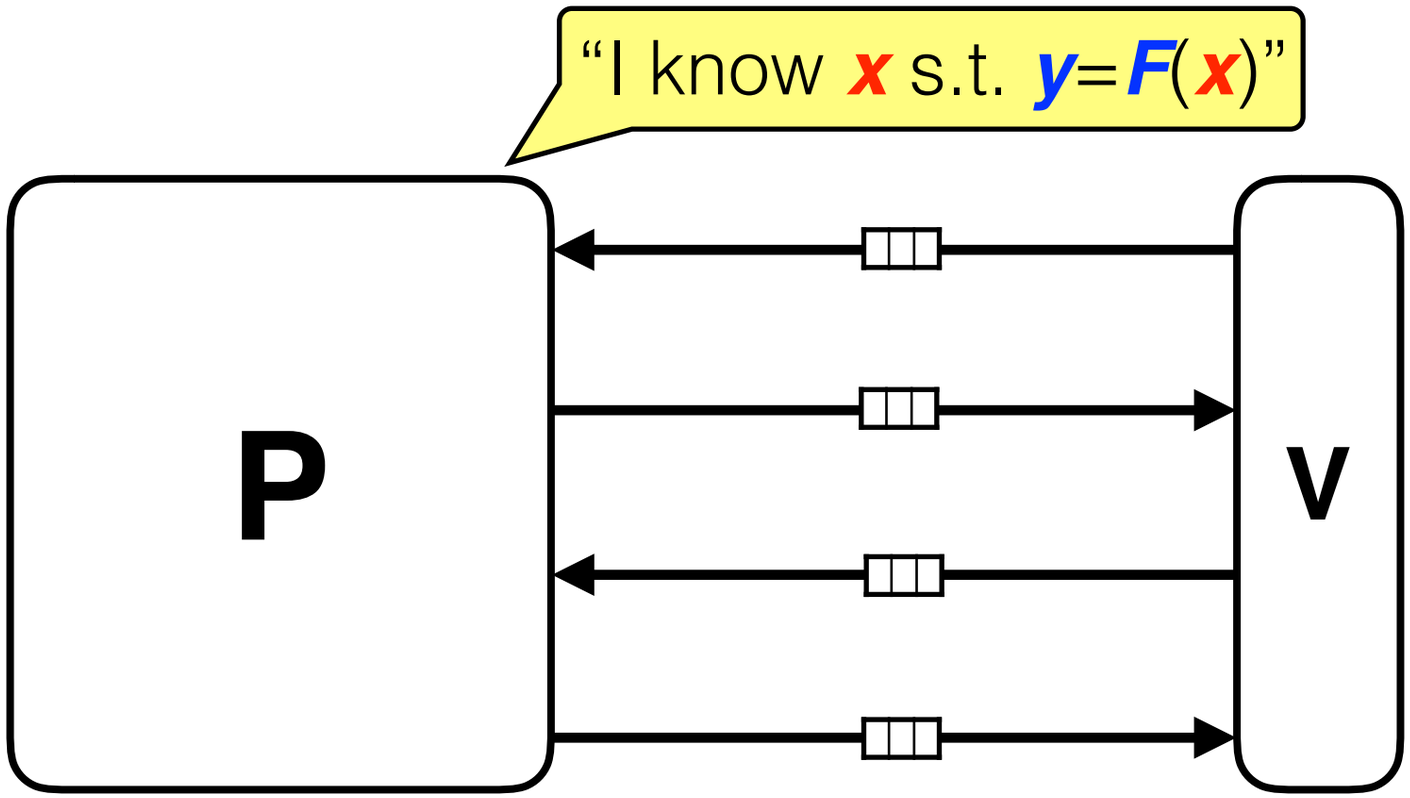
for typical **F** size(**F**) ≪ time(**F**)

Zero Knowledge **Succinct** Proofs

[Kilian92][Micali94]

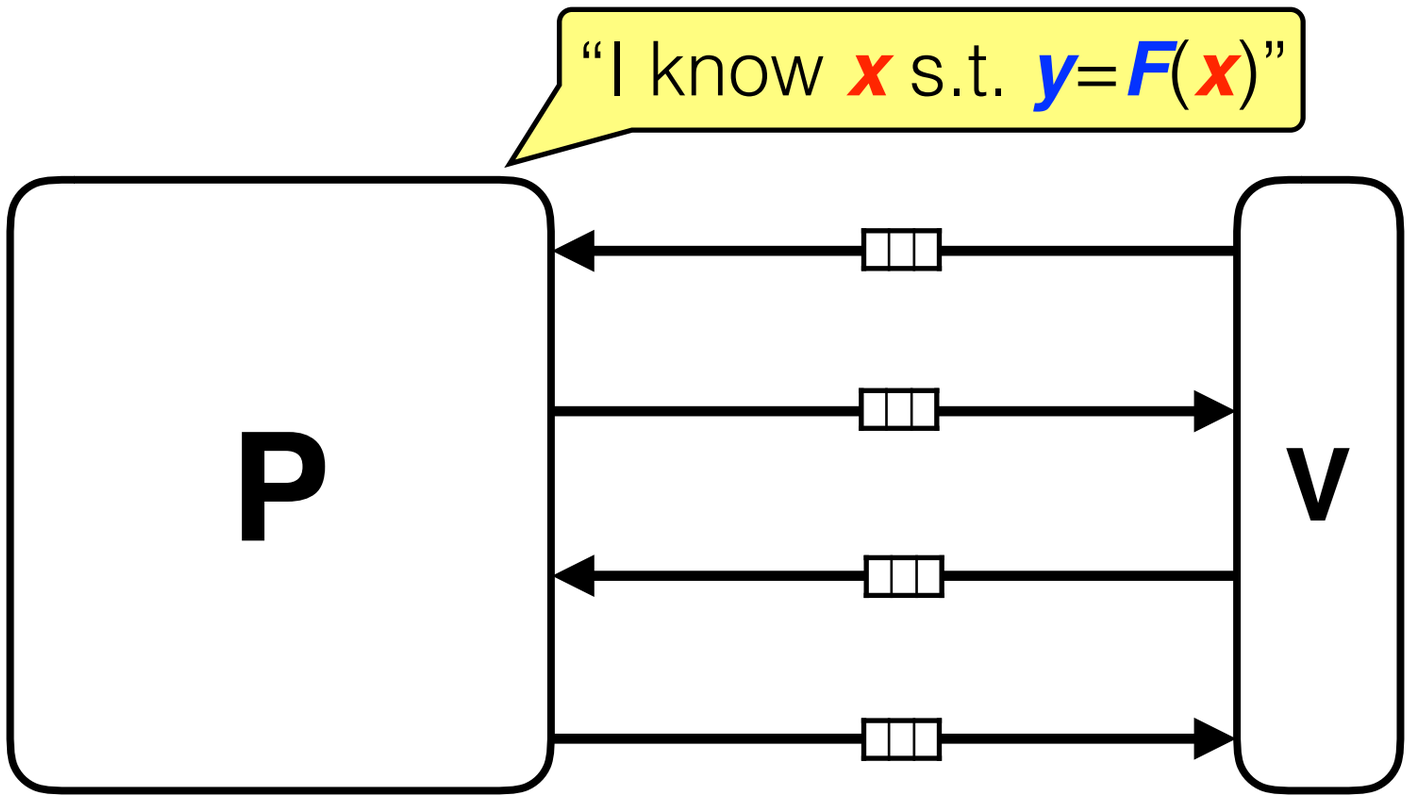
Zero Knowledge **Succinct** Proofs

[Kilian92][Micali94]



Zero Knowledge **Succinct** Proofs

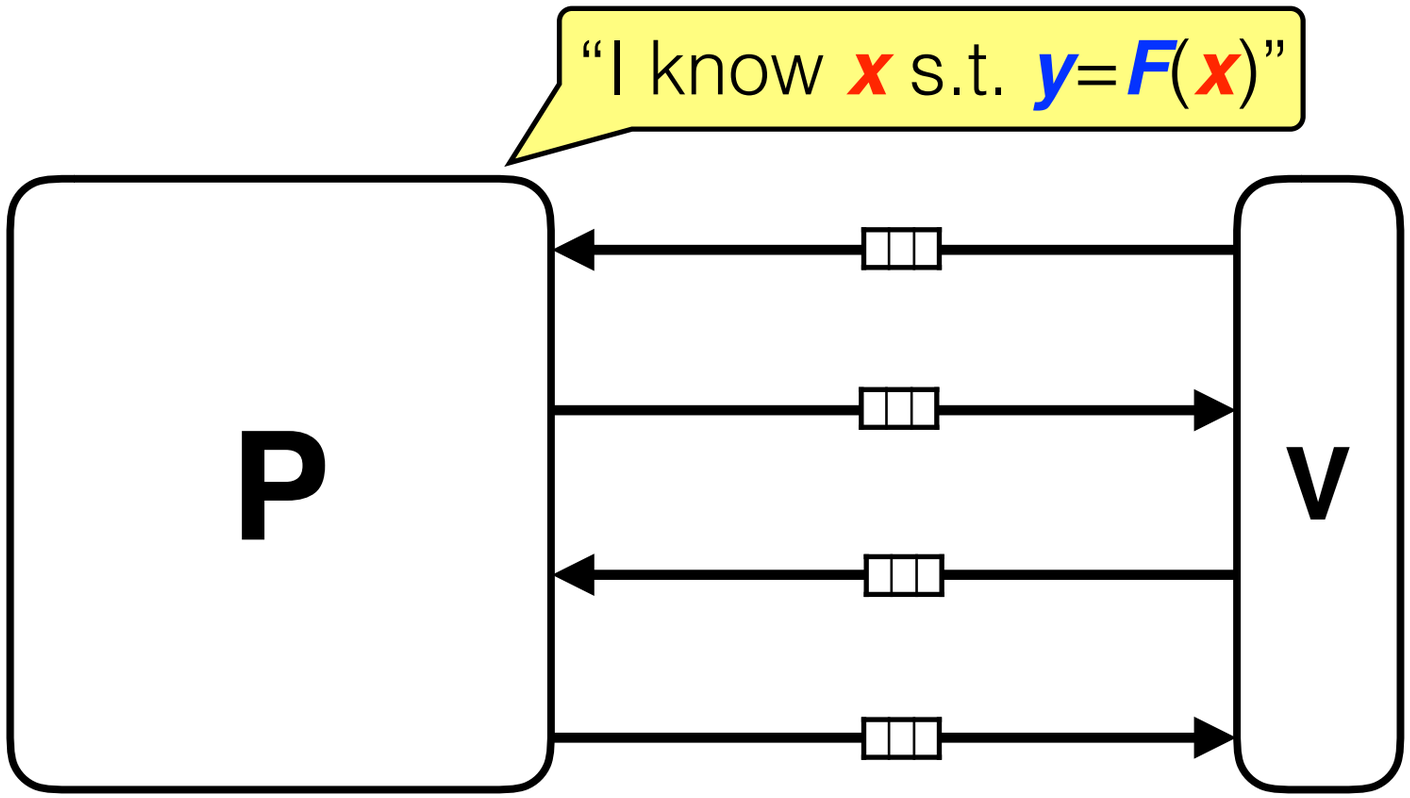
[Kilian92][Micali94]



completeness	$\exists\, \boldsymbol{x}$: $\boldsymbol{y}=\boldsymbol{F}(\boldsymbol{x}) \rightarrow \Pr[\mathbf{P}(\boldsymbol{F},\boldsymbol{y},\boldsymbol{x})$ convinces $\mathbf{V}(\boldsymbol{F},\boldsymbol{y})]=1$
soundness	$\nexists\, \boldsymbol{x}$: $\boldsymbol{y}=\boldsymbol{F}(\boldsymbol{x}) \rightarrow \forall\, \mathbf{P'}$ $\Pr[\mathbf{P'}$ convinces $\mathbf{V}(\boldsymbol{F},\boldsymbol{y})]\simeq 0$
zero knowledge	$\exists\, \boldsymbol{x}$: $\boldsymbol{y}=\boldsymbol{F}(\boldsymbol{x}) \rightarrow \forall\, \mathbf{V'}$, $\mathbf{S}(\mathbf{V'},\boldsymbol{F},\boldsymbol{y}) \simeq$ view of $\mathbf{V'}$ with $\mathbf{P}(\boldsymbol{F},\boldsymbol{y},\boldsymbol{x})$
succinctness	$\mathbf{V}(\boldsymbol{F},\boldsymbol{y})$ runs in time proportional to $\lvert\boldsymbol{F}\rvert+\lvert\boldsymbol{y}\rvert$ (not time$(\boldsymbol{F})+\lvert\boldsymbol{y}\rvert$)

Zero Knowledge **Succinct** Proofs

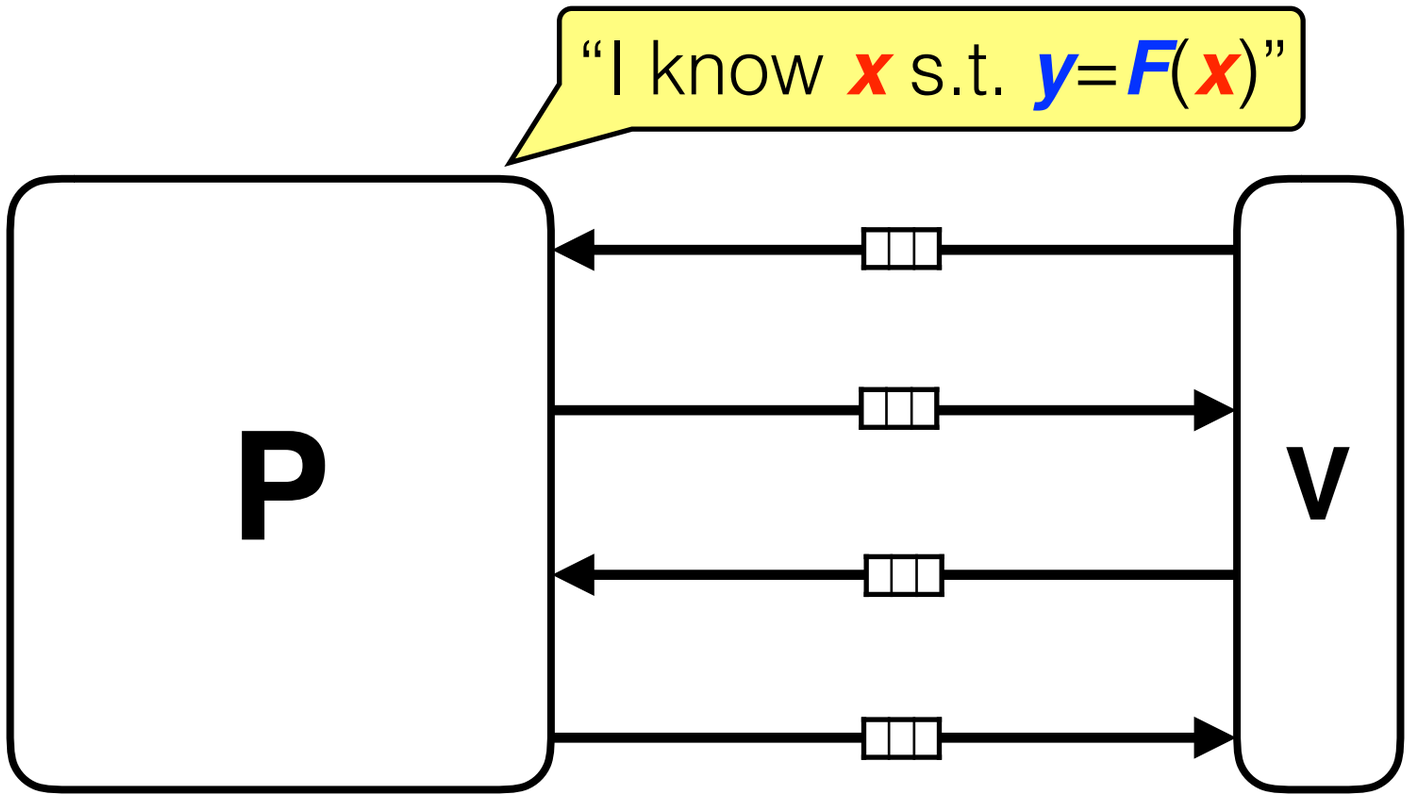
[Kilian92][Micali94]



completeness	$\exists\, \boldsymbol{x}$: $\boldsymbol{y}=\boldsymbol{F}(\boldsymbol{x}) \rightarrow \Pr[\mathbf{P}(\boldsymbol{F},\boldsymbol{y},\boldsymbol{x})$ convinces $\mathbf{V}(\boldsymbol{F},\boldsymbol{y})]=1$
soundness*	$\nexists\, \boldsymbol{x}$: $\boldsymbol{y}=\boldsymbol{F}(\boldsymbol{x}) \rightarrow \forall\, \mathbf{P'}$ $\Pr[\mathbf{P'}$ convinces $\mathbf{V}(\boldsymbol{F},\boldsymbol{y})]\simeq 0$
zero knowledge	$\exists\, \boldsymbol{x}$: $\boldsymbol{y}=\boldsymbol{F}(\boldsymbol{x}) \rightarrow \forall\, \mathbf{V'}$, $\mathbf{S}(\mathbf{V'},\boldsymbol{F},\boldsymbol{y}) \simeq$ view of $\mathbf{V'}$ with $\mathbf{P}(\boldsymbol{F},\boldsymbol{y},\boldsymbol{x})$
succinctness	$\mathbf{V}(\boldsymbol{F},\boldsymbol{y})$ runs in time proportional to $\lvert\boldsymbol{F}\rvert+\lvert\boldsymbol{y}\rvert$ (not time$(\boldsymbol{F})+\lvert\boldsymbol{y}\rvert$)

Zero Knowledge **Succinct** Proofs

[Kilian92][Micali94]

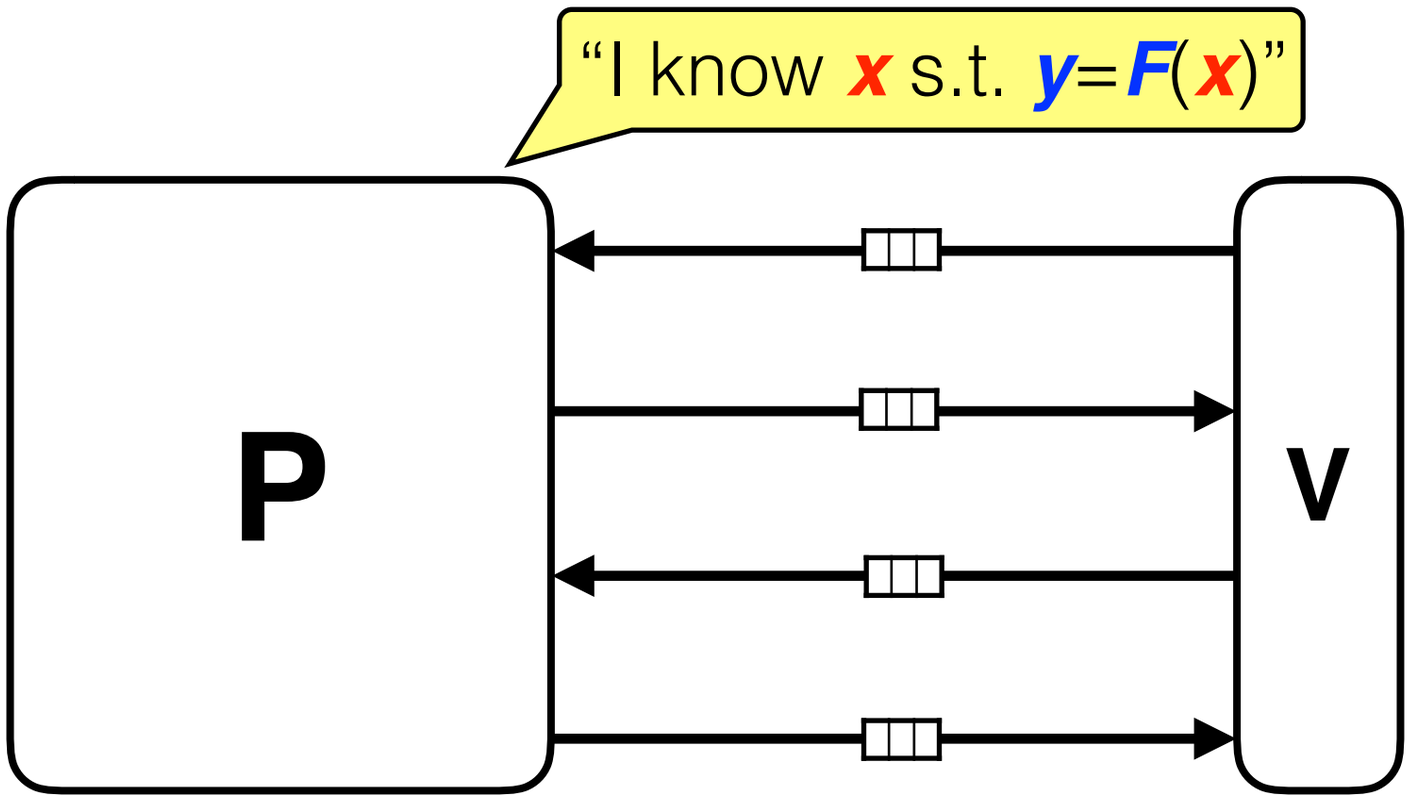


completeness	$\exists\, \boldsymbol{x}$: $\boldsymbol{y}=\boldsymbol{F}(\boldsymbol{x}) \rightarrow \Pr[\mathbf{P}(\boldsymbol{F},\boldsymbol{y},\boldsymbol{x})$ convinces $\mathbf{V}(\boldsymbol{F},\boldsymbol{y})]=1$
soundness*	$\nexists\, \boldsymbol{x}$: $\boldsymbol{y}=\boldsymbol{F}(\boldsymbol{x}) \rightarrow \forall\, \mathbf{P'}$ $\Pr[\mathbf{P'}$ convinces $\mathbf{V}(\boldsymbol{F},\boldsymbol{y})]\simeq 0$
zero knowledge	$\exists\, \boldsymbol{x}$: $\boldsymbol{y}=\boldsymbol{F}(\boldsymbol{x}) \rightarrow \forall\, \mathbf{V'}$, $\mathbf{S}(\mathbf{V'},\boldsymbol{F},\boldsymbol{y}) \simeq$ view of $\mathbf{V'}$ with $\mathbf{P}(\boldsymbol{F},\boldsymbol{y},\boldsymbol{x})$
succinctness	$\mathbf{V}(\boldsymbol{F},\boldsymbol{y})$ runs in time proportional to $\lvert\boldsymbol{F}\rvert+\lvert\boldsymbol{y}\rvert$ (not time($\boldsymbol{F}$)+$\lvert\boldsymbol{y}\rvert$)

* must relax to *computational* soundness: ∀ PPT **P'** … [GH98]

Zero Knowledge **Succinct** ~~Proofs~~ Arguments

[Kilian92][Micali94]



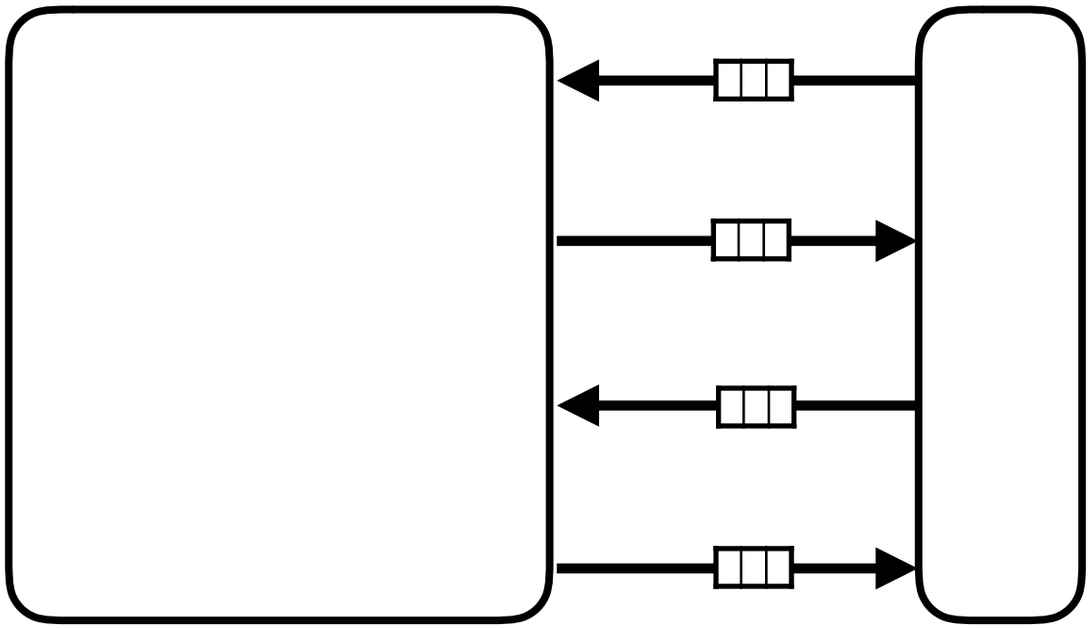
completeness	$\exists\, \boldsymbol{x}$: $\boldsymbol{y}=\boldsymbol{F}(\boldsymbol{x}) \rightarrow \Pr[\mathbf{P}(\boldsymbol{F},\boldsymbol{y},\boldsymbol{x})$ convinces $\mathbf{V}(\boldsymbol{F},\boldsymbol{y})]=1$
soundness*	$\nexists\, \boldsymbol{x}$: $\boldsymbol{y}=\boldsymbol{F}(\boldsymbol{x}) \rightarrow \forall\, \mathbf{P'}$ $\Pr[\mathbf{P'}$ convinces $\mathbf{V}(\boldsymbol{F},\boldsymbol{y})]\simeq 0$
zero knowledge	$\exists\, \boldsymbol{x}$: $\boldsymbol{y}=\boldsymbol{F}(\boldsymbol{x}) \rightarrow \forall\, \mathbf{V'}$, $\mathbf{S}(\mathbf{V'},\boldsymbol{F},\boldsymbol{y}) \simeq$ view of $\mathbf{V'}$ with $\mathbf{P}(\boldsymbol{F},\boldsymbol{y},\boldsymbol{x})$
succinctness	$\mathbf{V}(\boldsymbol{F},\boldsymbol{y})$ runs in time proportional to $\lvert\boldsymbol{F}\rvert+\lvert\boldsymbol{y}\rvert$ (not time($\boldsymbol{F}$)+$\lvert\boldsymbol{y}\rvert$)

* must relax to *computational* soundness: ∀ PPT **P'** … [GH98]

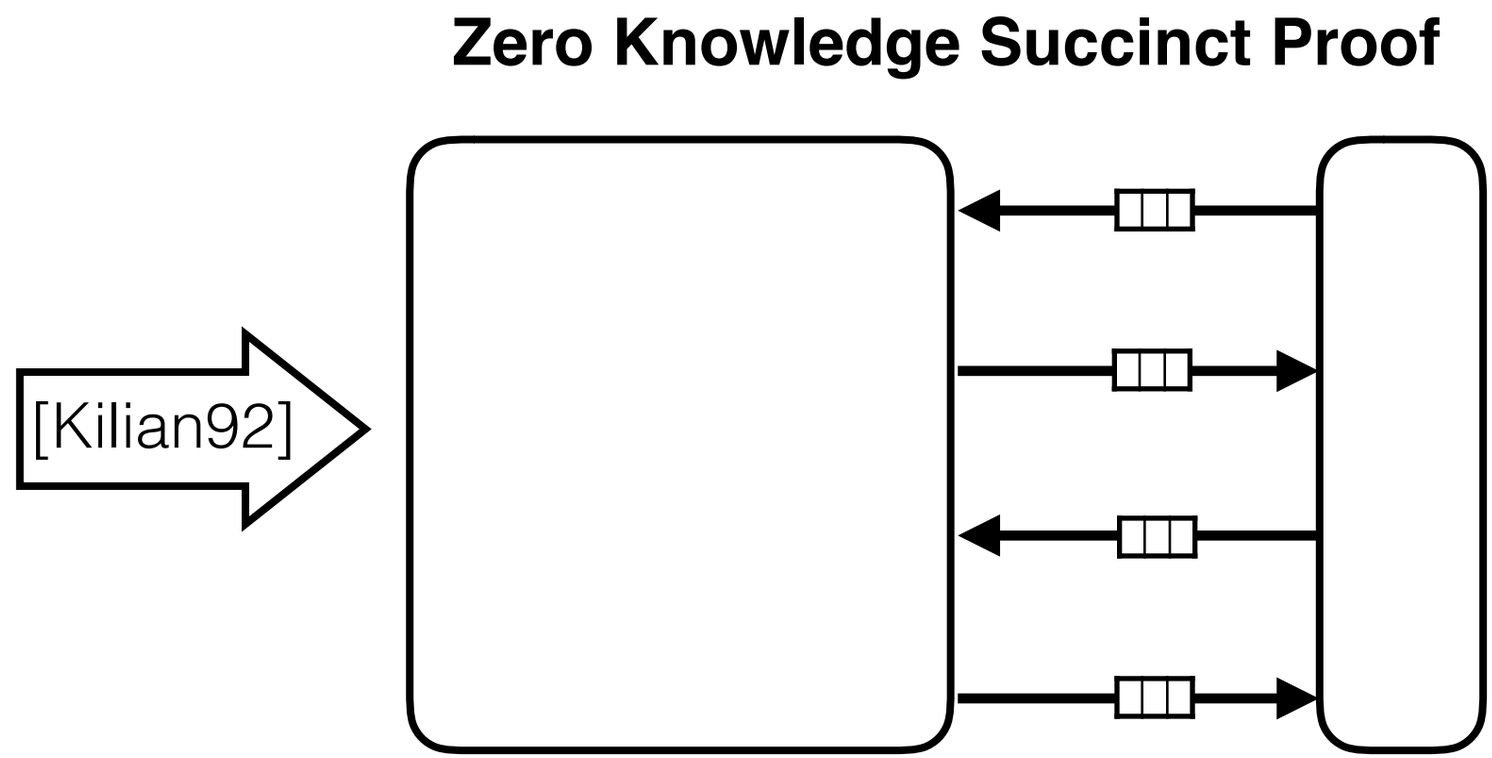
Achieving Succinctness

Achieving Succinctness

Zero Knowledge Succinct Proof



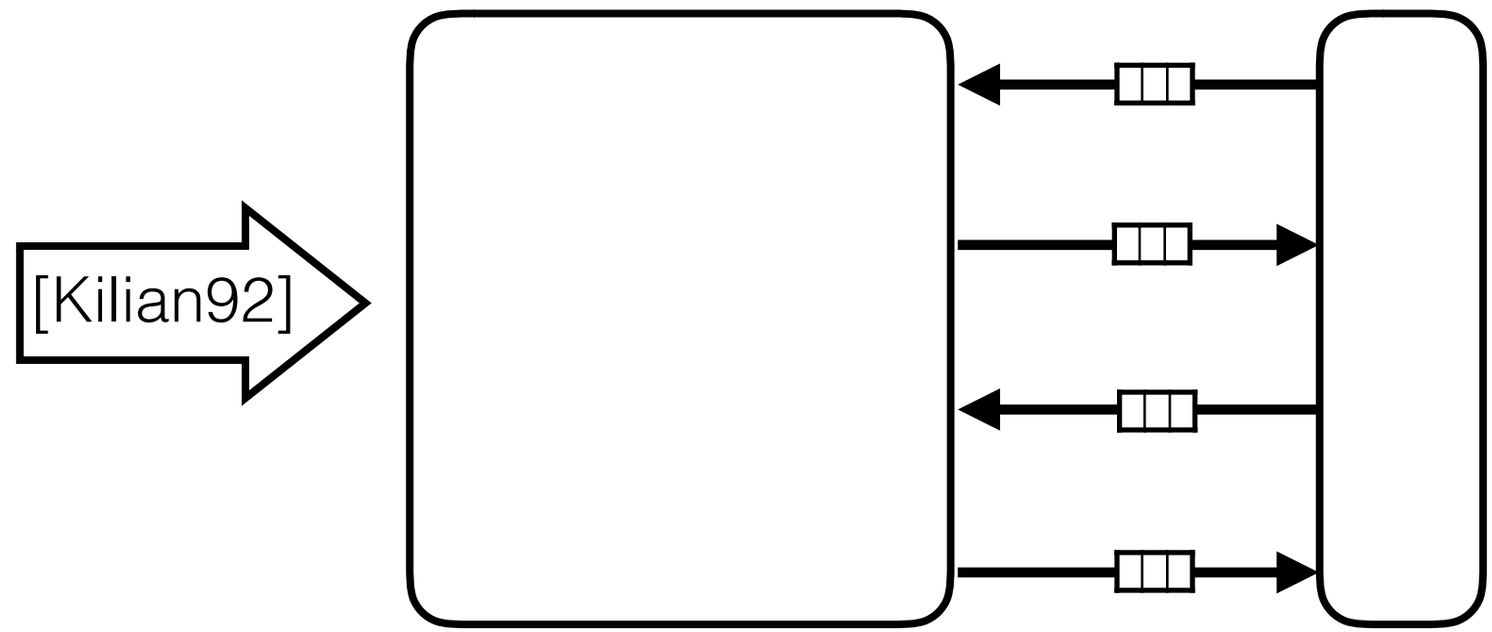
Achieving Succinctness



Achieving Succinctness

Probabilistically Checkable Proof
[BFLS91][FGLSS96][AS92][ALMSS92]

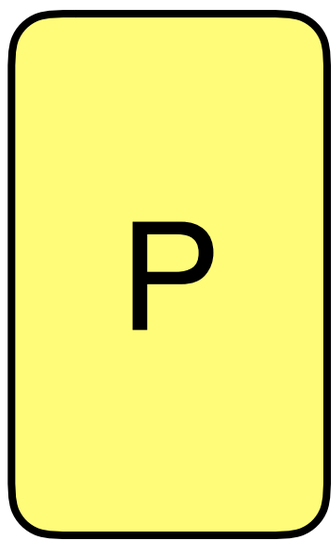
Zero Knowledge Succinct Proof

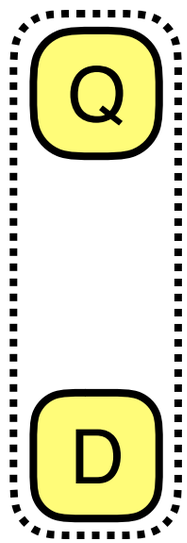


Achieving Succinctness

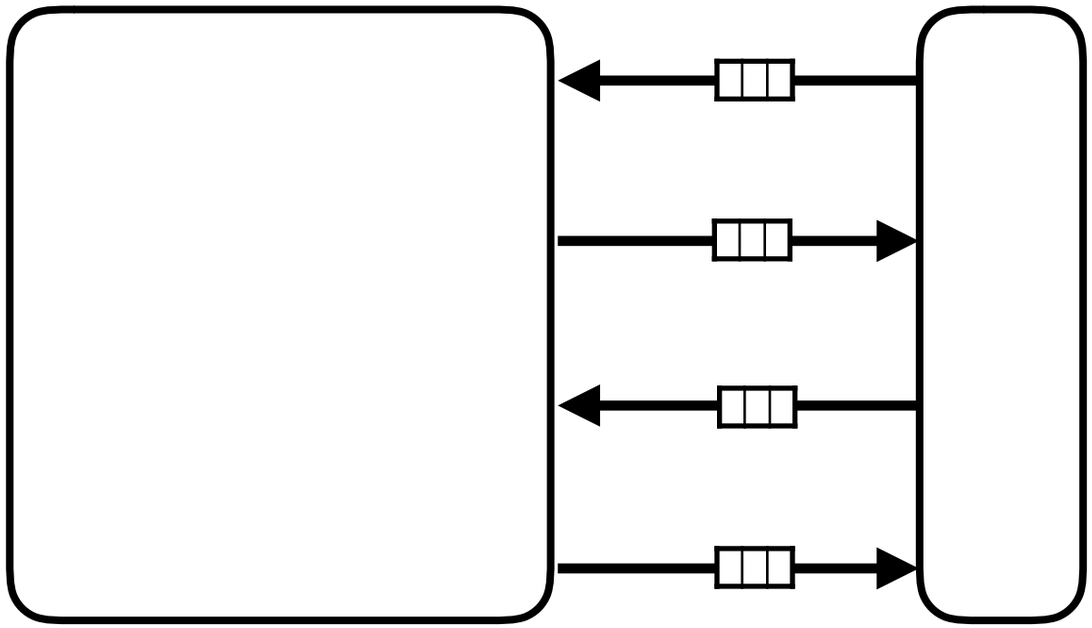
Probabilistically Checkable Proof

[BFLS91][FGLSS96][AS92][ALMSS92]





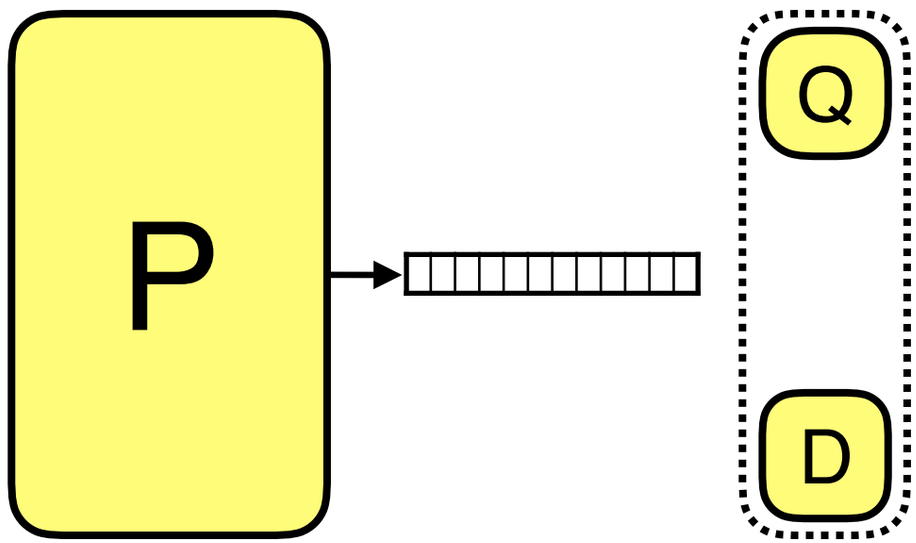
Zero Knowledge Succinct Proof



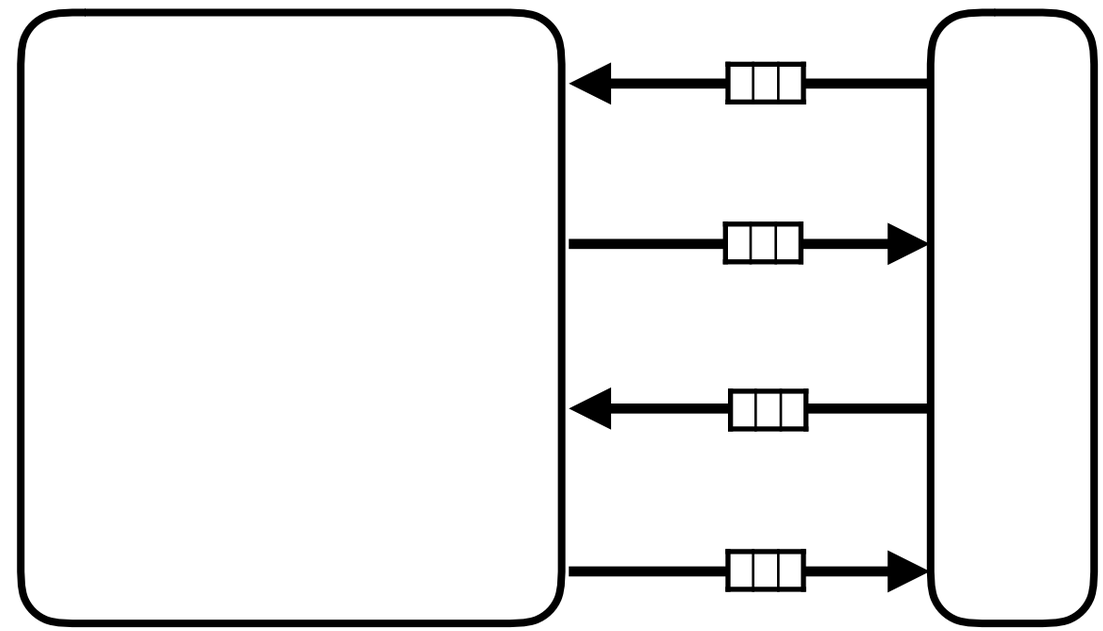
Achieving Succinctness

Probabilistically Checkable Proof

[BFLS91][FGLSS96][AS92][ALMSS92]



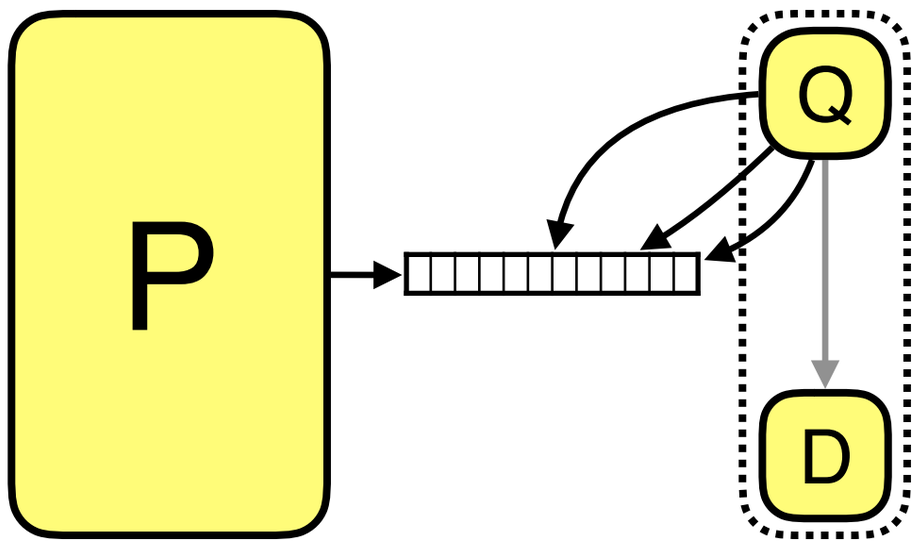
Zero Knowledge Succinct Proof



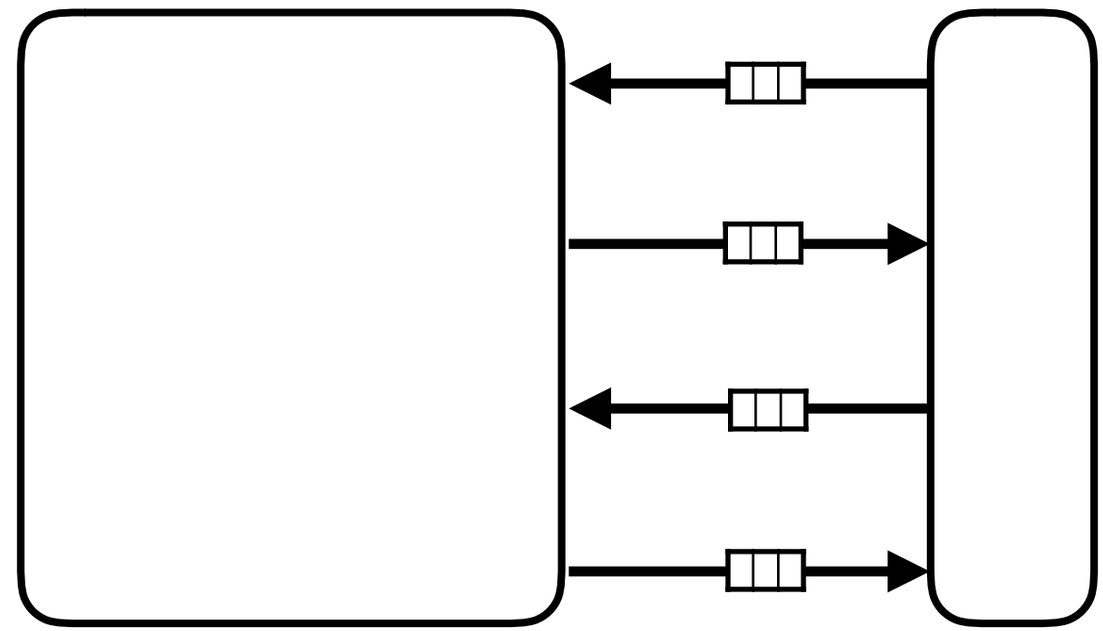
Achieving Succinctness

Probabilistically Checkable Proof

[BFLS91][FGLSS96][AS92][ALMSS92]



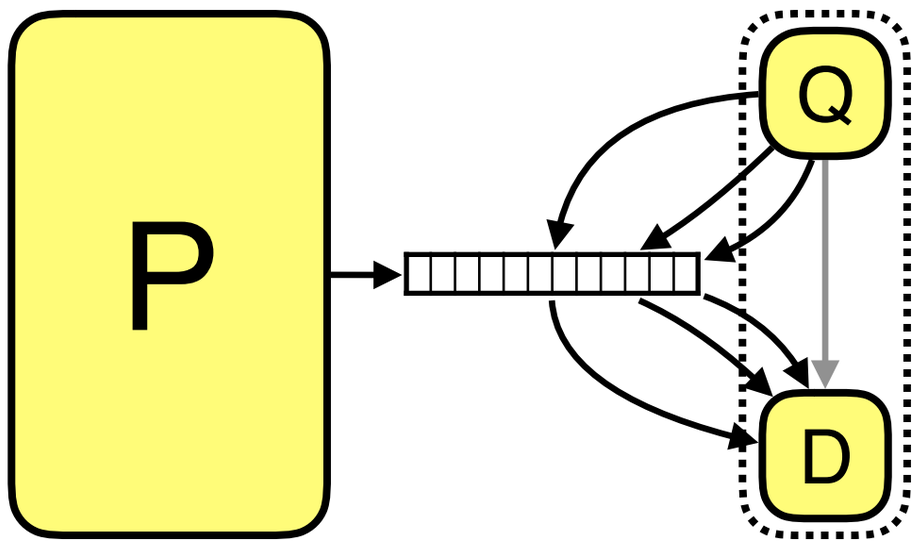
Zero Knowledge Succinct Proof



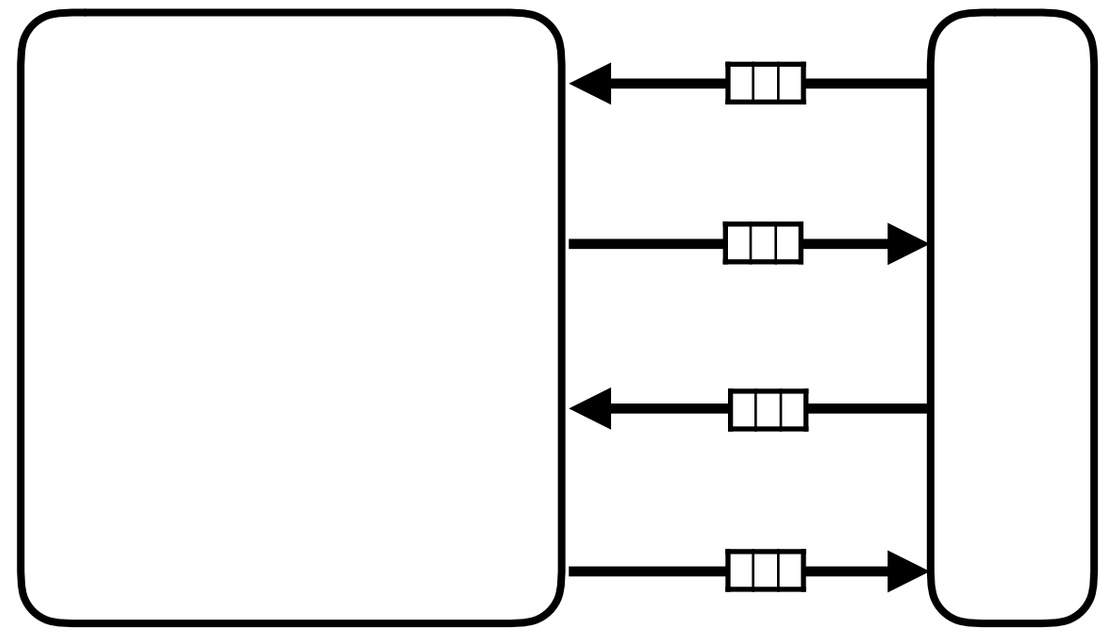
Achieving Succinctness

Probabilistically Checkable Proof

[BFLS91][FGLSS96][AS92][ALMSS92]



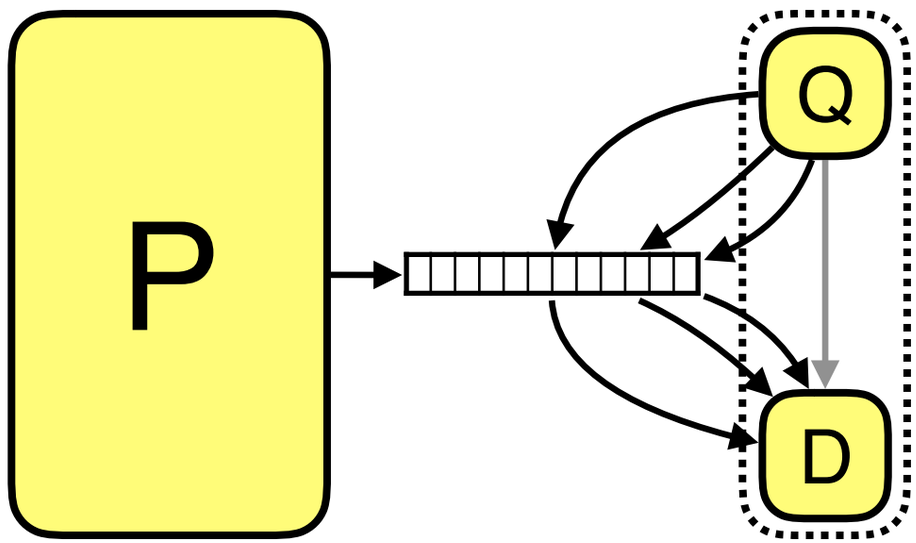
Zero Knowledge Succinct Proof



Achieving Succinctness

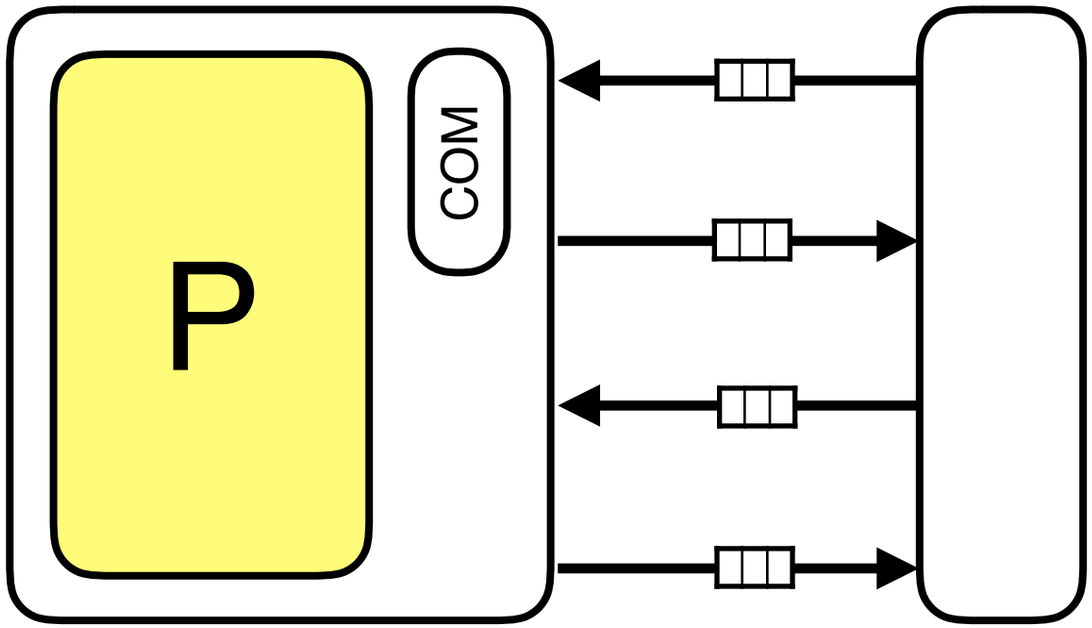
Probabilistically Checkable Proof

[BFLS91][FGLSS96][AS92][ALMSS92]



[Kilian92]

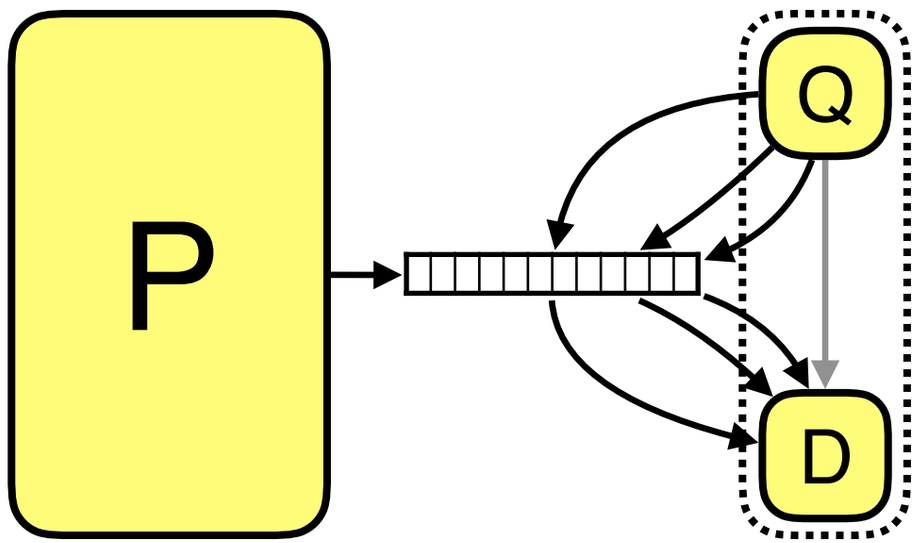
Zero Knowledge Succinct Proof



Achieving Succinctness

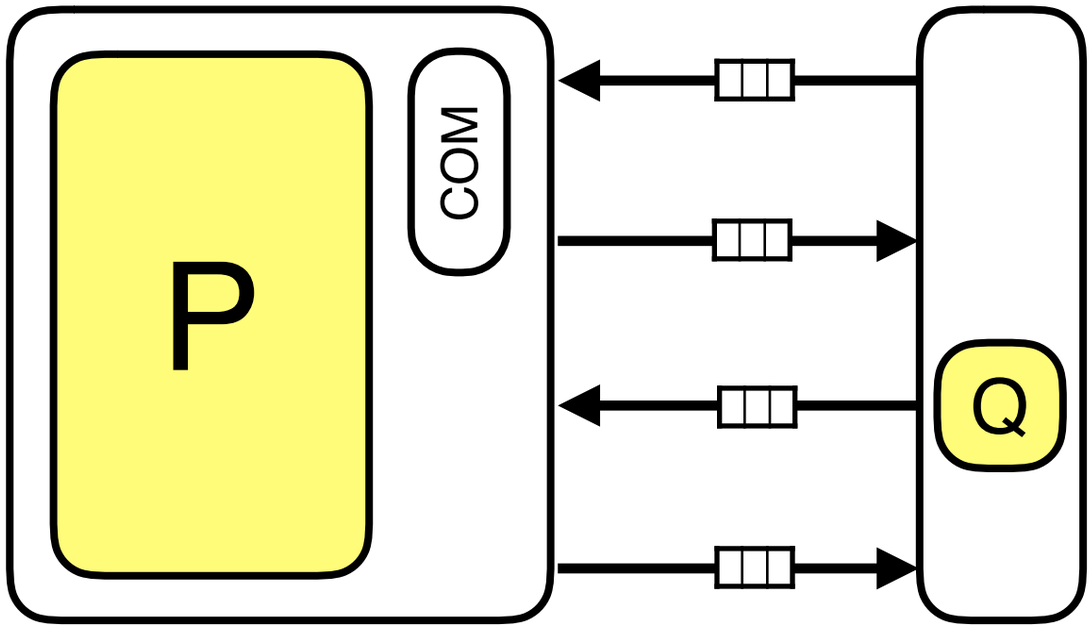
Probabilistically Checkable Proof

[BFLS91][FGLSS96][AS92][ALMSS92]



[Kilian92]

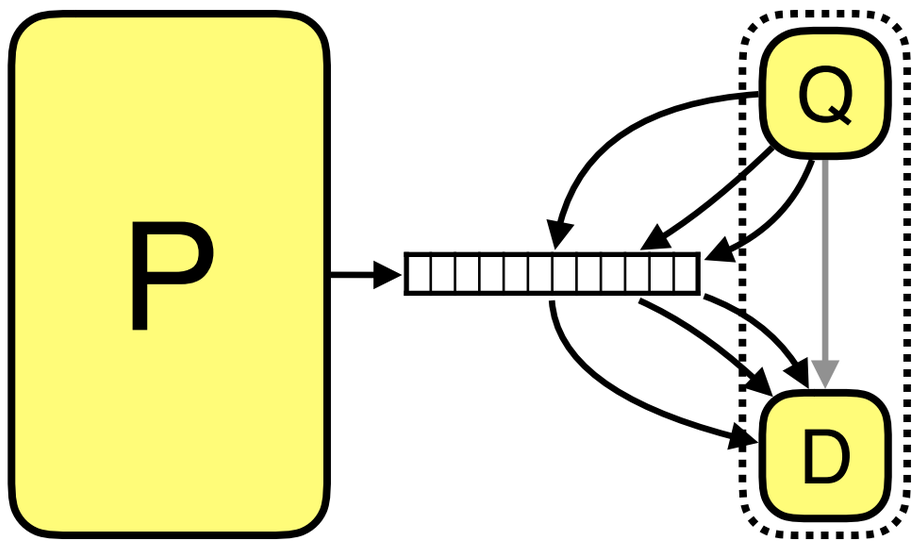
Zero Knowledge Succinct Proof



Achieving Succinctness

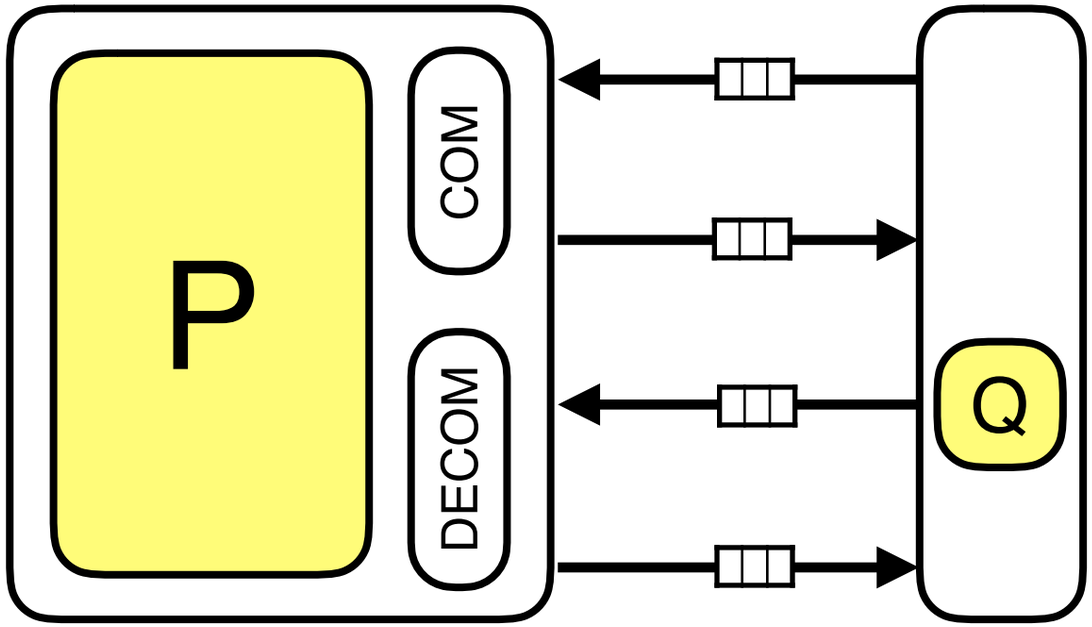
Probabilistically Checkable Proof

[BFLS91][FGLSS96][AS92][ALMSS92]



[Kilian92]

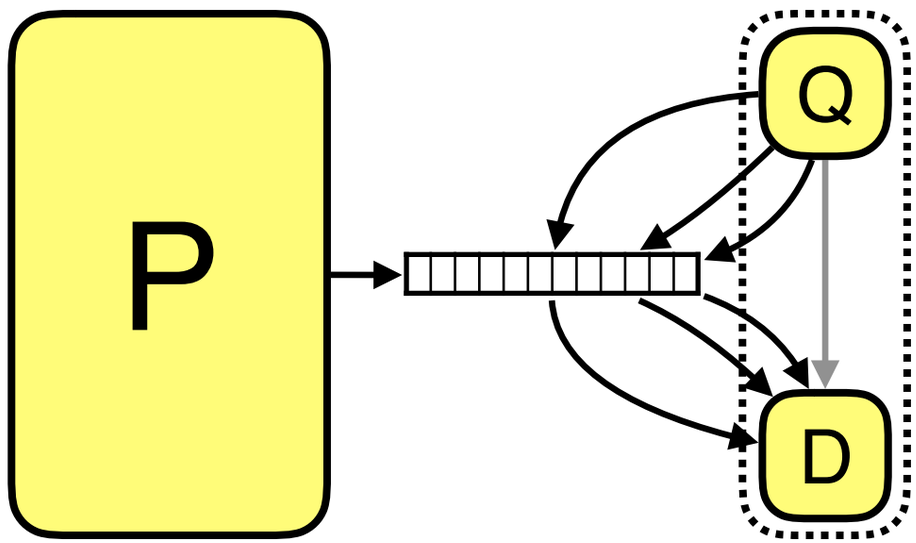
Zero Knowledge Succinct Proof



Achieving Succinctness

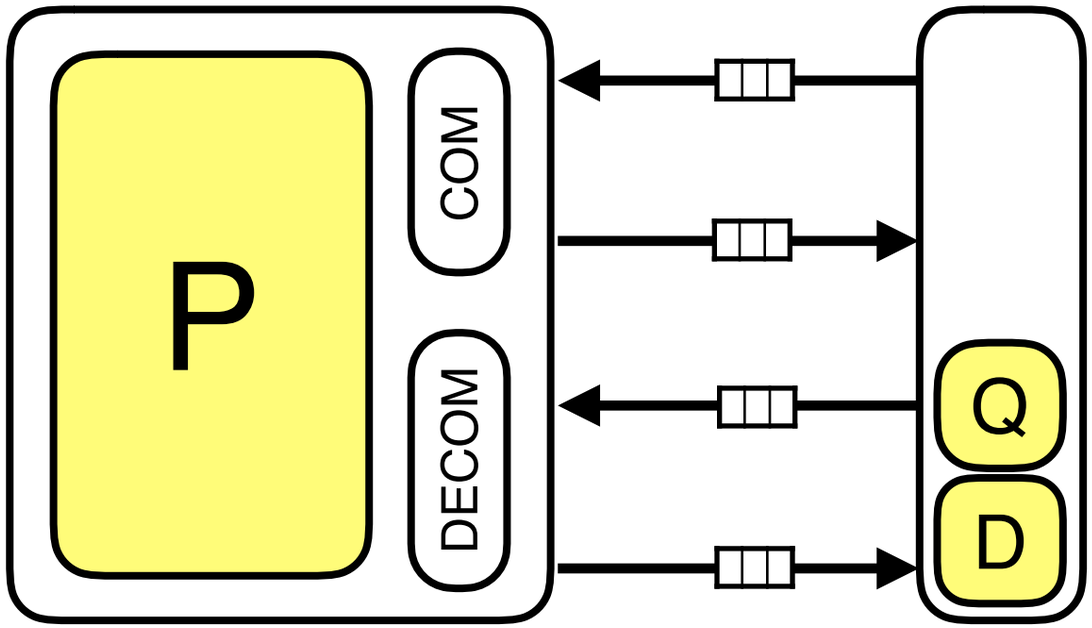
Probabilistically Checkable Proof

[BFLS91][FGLSS96][AS92][ALMSS92]



[Kilian92]

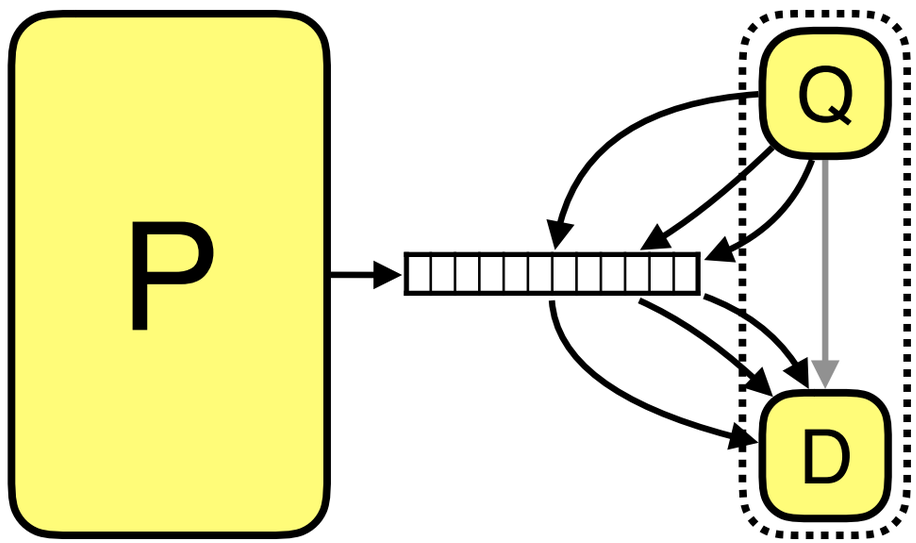
Zero Knowledge Succinct Proof



Achieving Succinctness

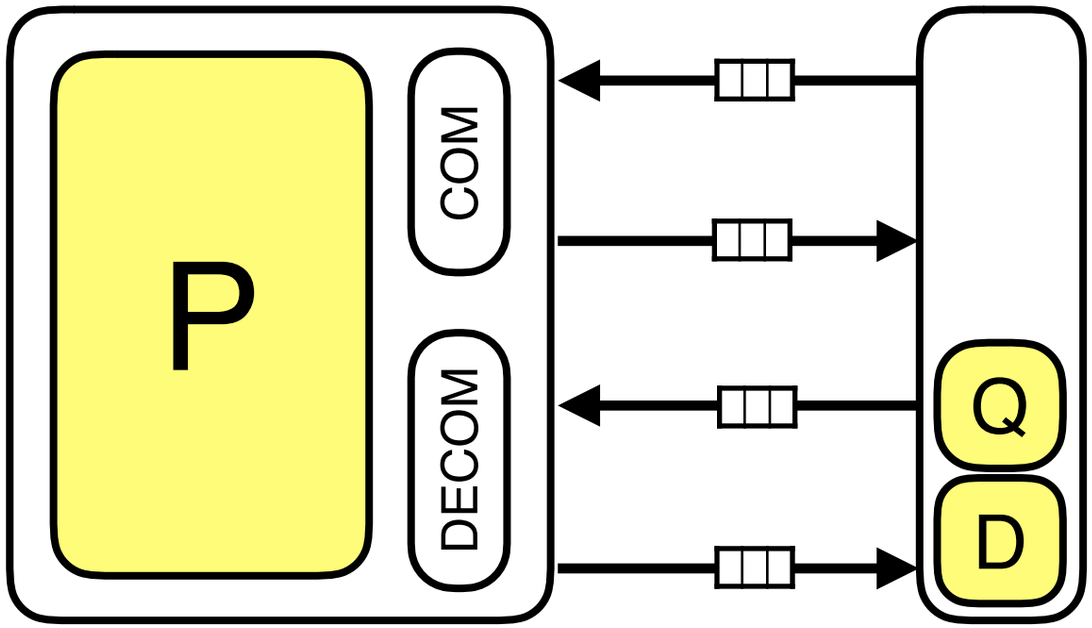
Probabilistically Checkable Proof

[BFLS91][FGLSS96][AS92][ALMSS92]



[Kilian92]

Zero Knowledge Succinct Proof



TOFIX

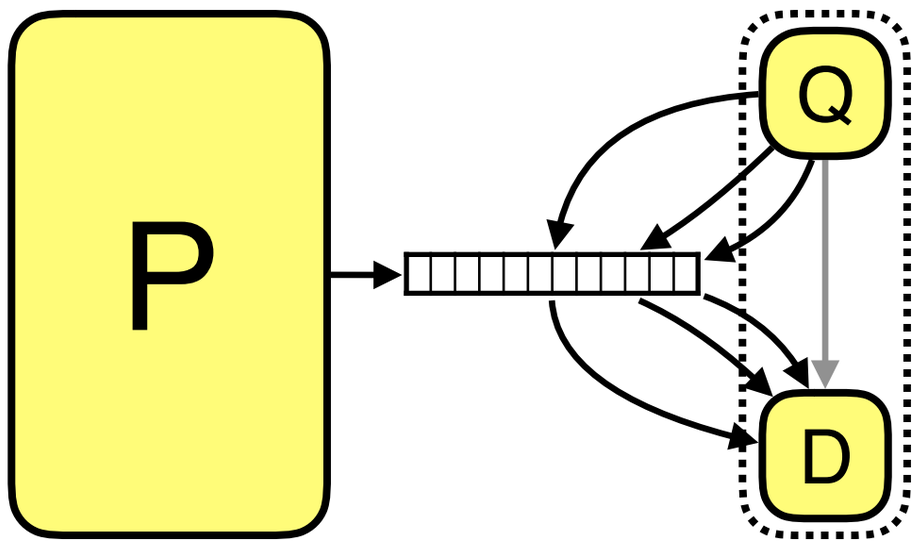
interactive

not succinct

Achieving Succinctness

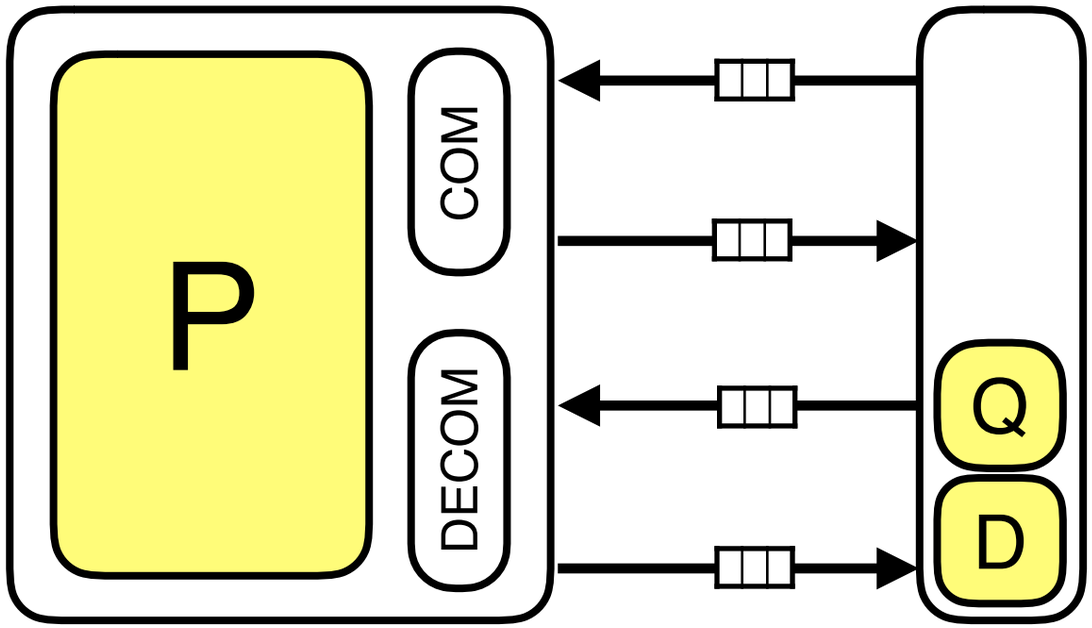
Probabilistically Checkable Proof

[BFLS91][FGLSS96][AS92][ALMSS92]



[Kilian92]

Zero Knowledge Succinct Proof



TOFIX

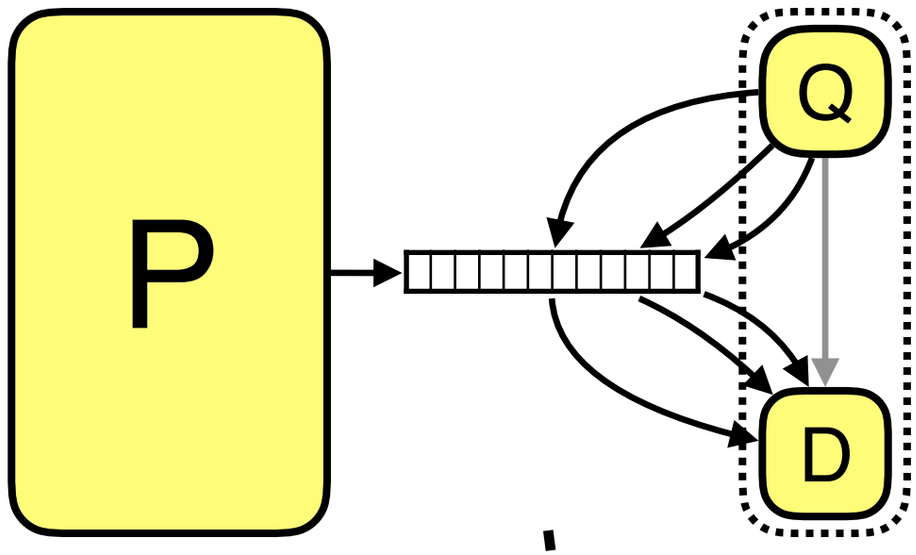
interactive

~~not succinct~~

Achieving Succinctness

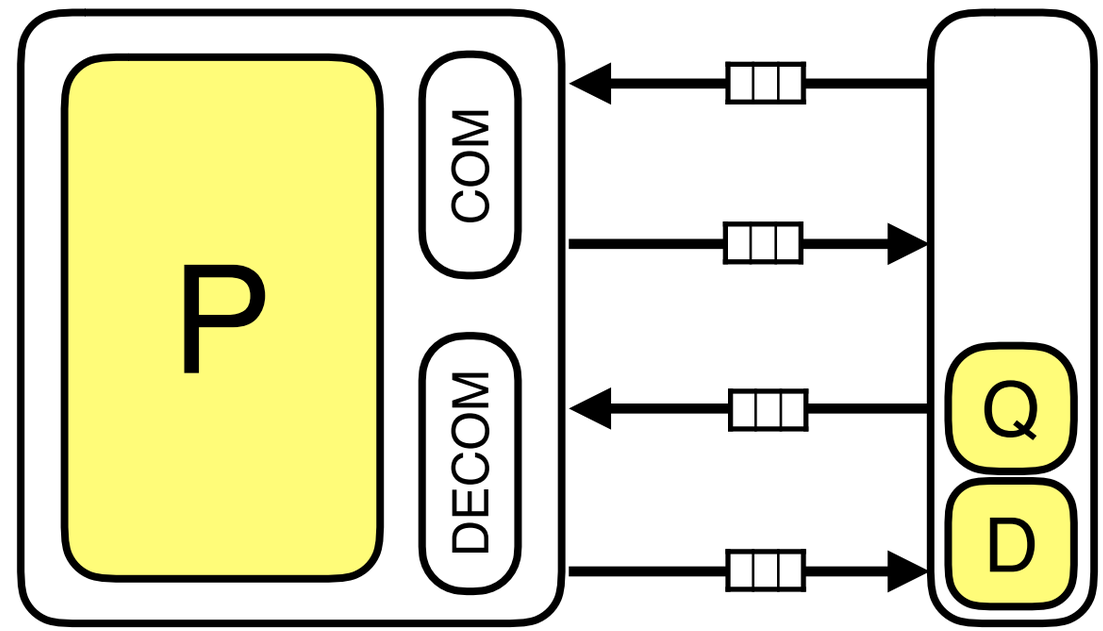
Probabilistically Checkable Proof

[BFLS91][FGLSS96][AS92][ALMSS92]



[Kilian92]

Zero Knowledge Succinct Proof



TOFIX

interactive

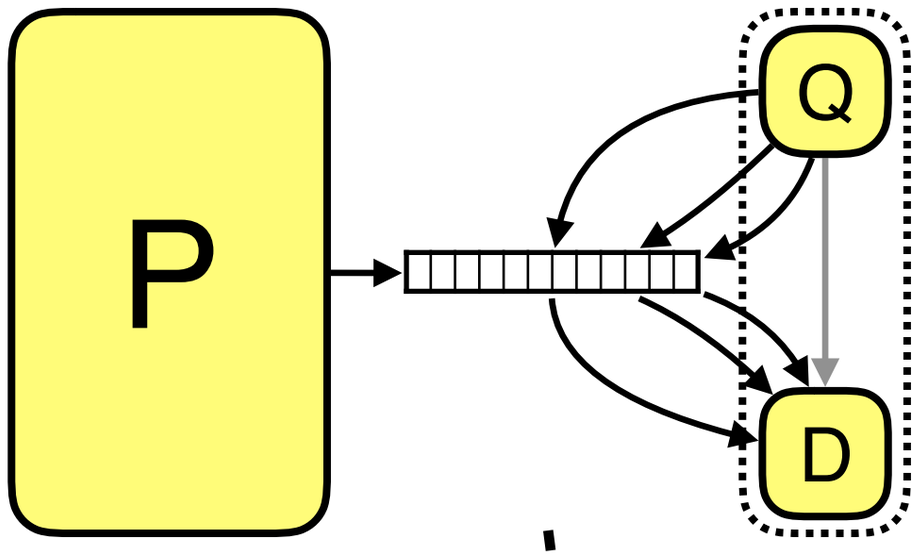
~~not succinct~~

bad concrete efficiency

Achieving Non-Interactivity

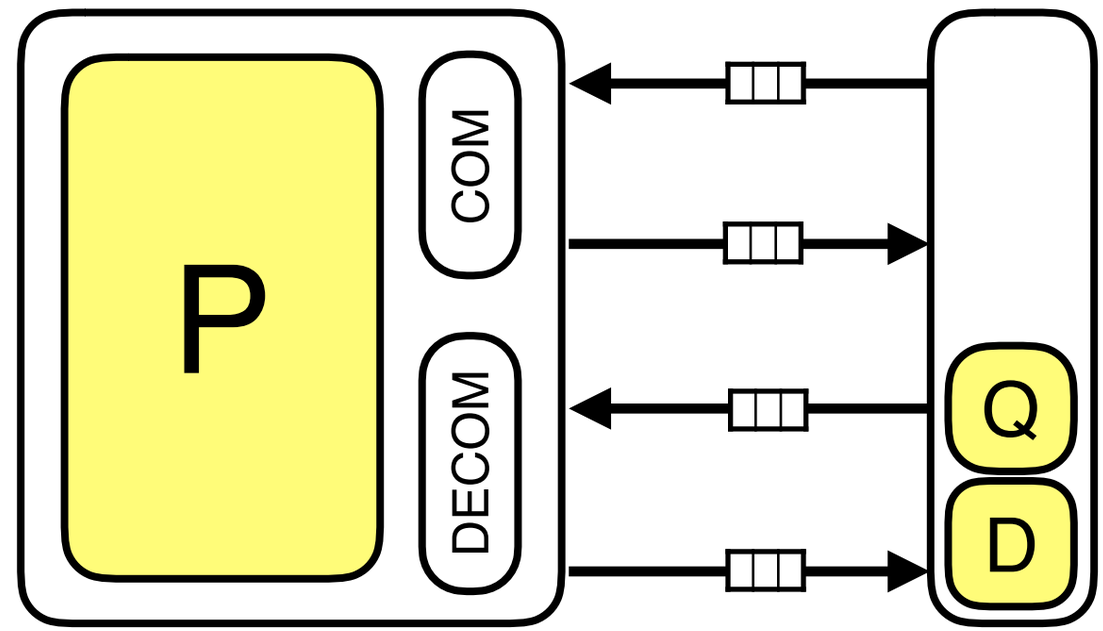
Probabilistically Checkable Proof

[BFLS91][FGLSS96][AS92][ALMSS92]



[Kilian92]

Zero Knowledge Succinct Proof



TOFIX

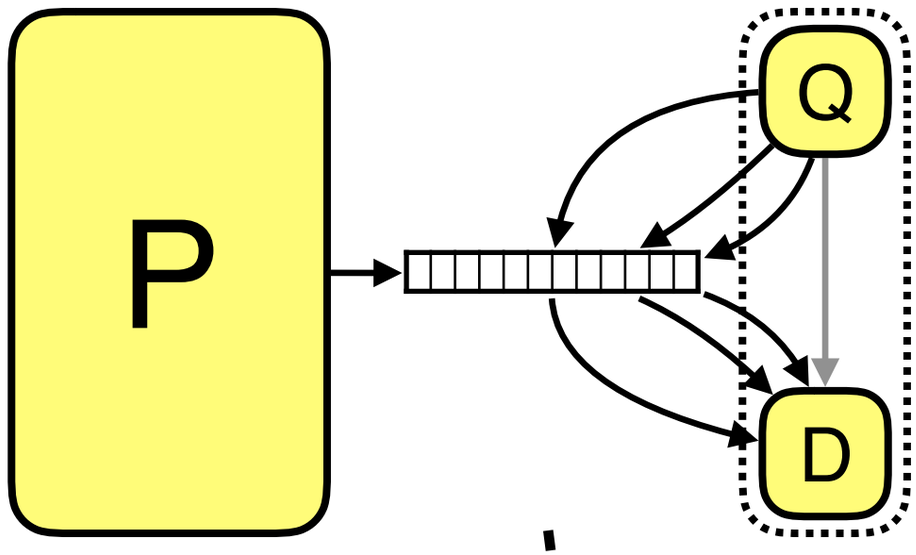
interactive

~~not succinct~~

bad concrete efficiency

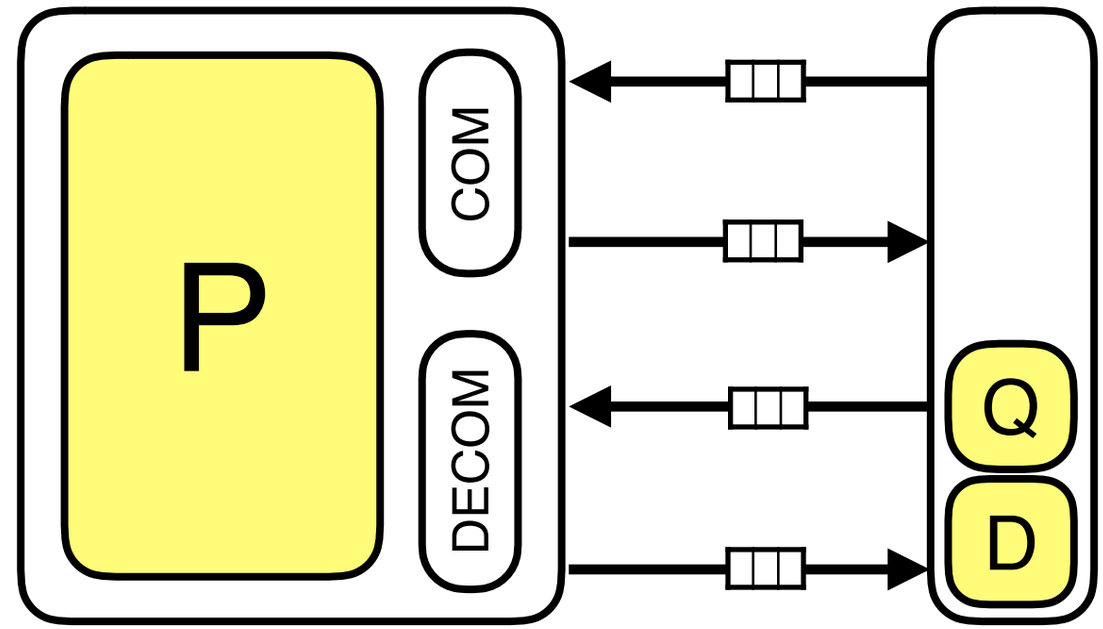
Achieving Non-Interactivity

Probabilistically Checkable Proof
[BFLS91][FGLSS96][AS92][ALMSS92]



[Kilian92]

Zero Knowledge Succinct Proof



[Micali94]

TOFIX

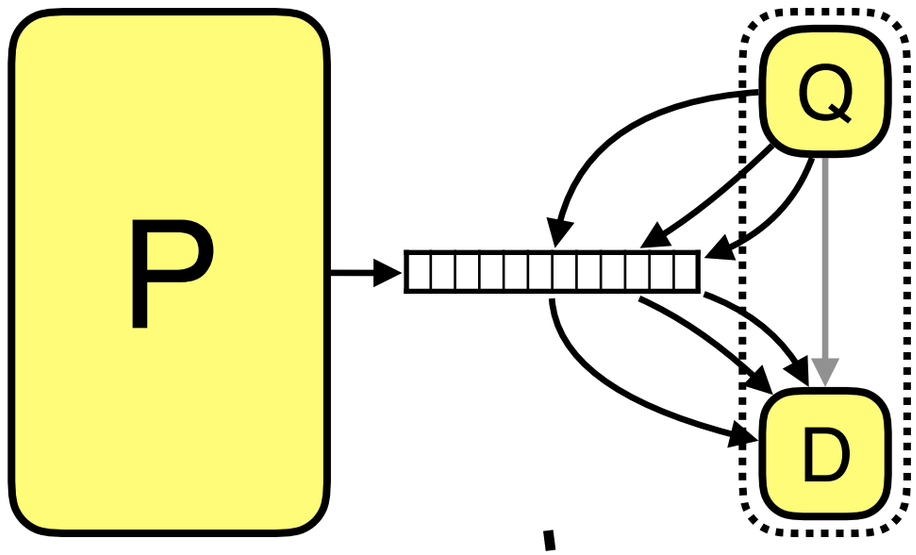
interactive

~~not succinct~~

bad concrete efficiency

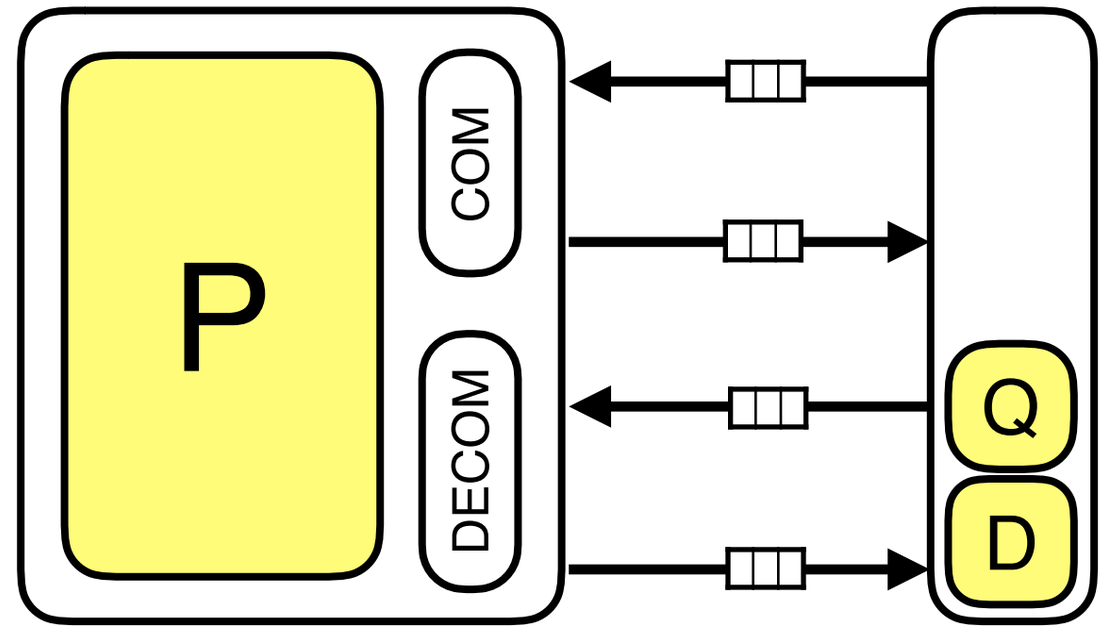
Achieving Non-Interactivity

Probabilistically Checkable Proof
[BFLS91][FGLSS96][AS92][ALMSS92]



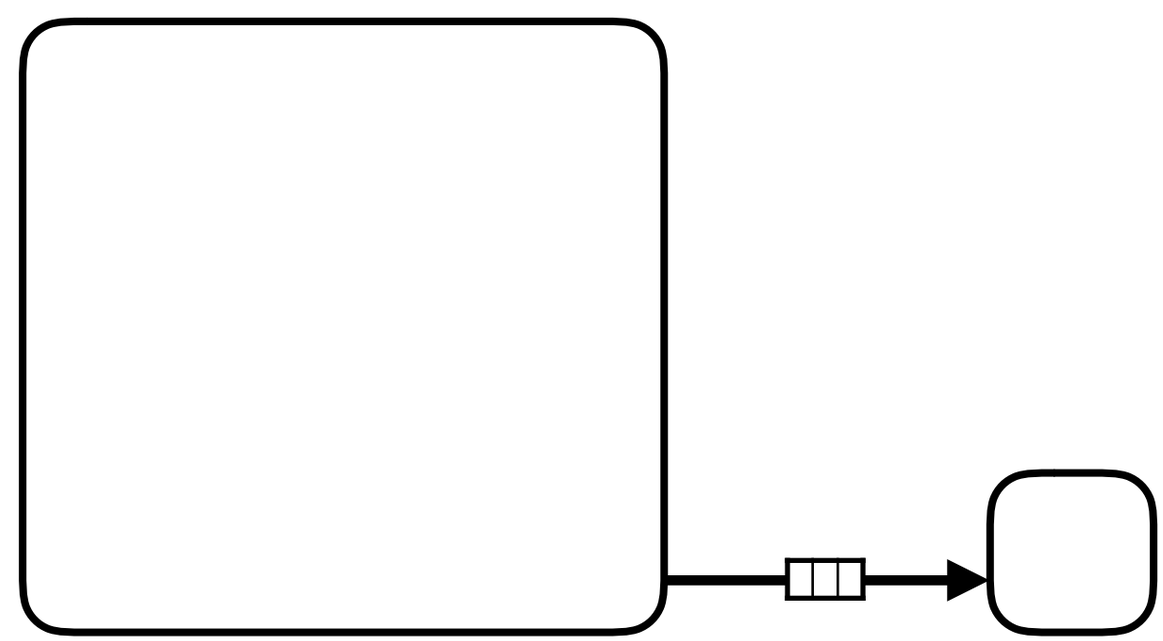
[Kilian92]

Zero Knowledge Succinct Proof



[Micali94]

(the first)
Zero Knowledge SNARK



TOFIX

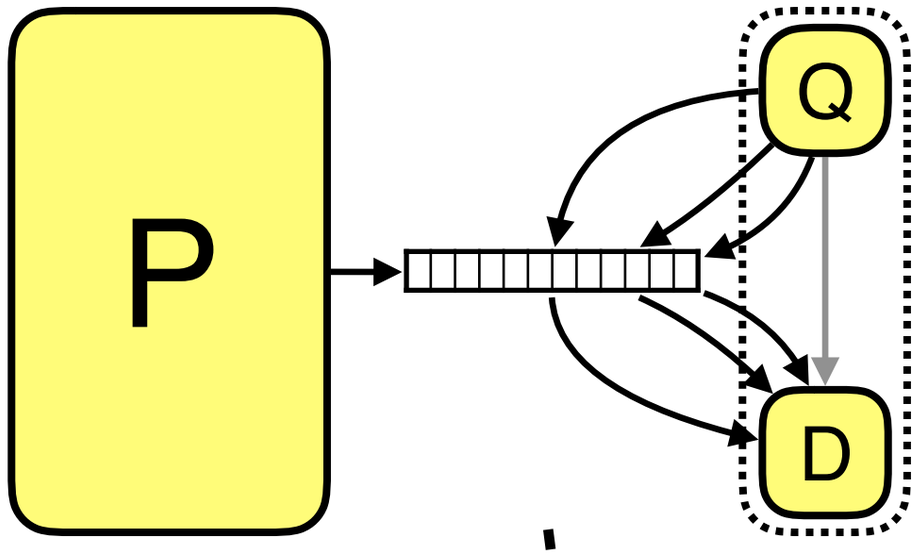
interactive

~~not succinct~~

bad concrete efficiency

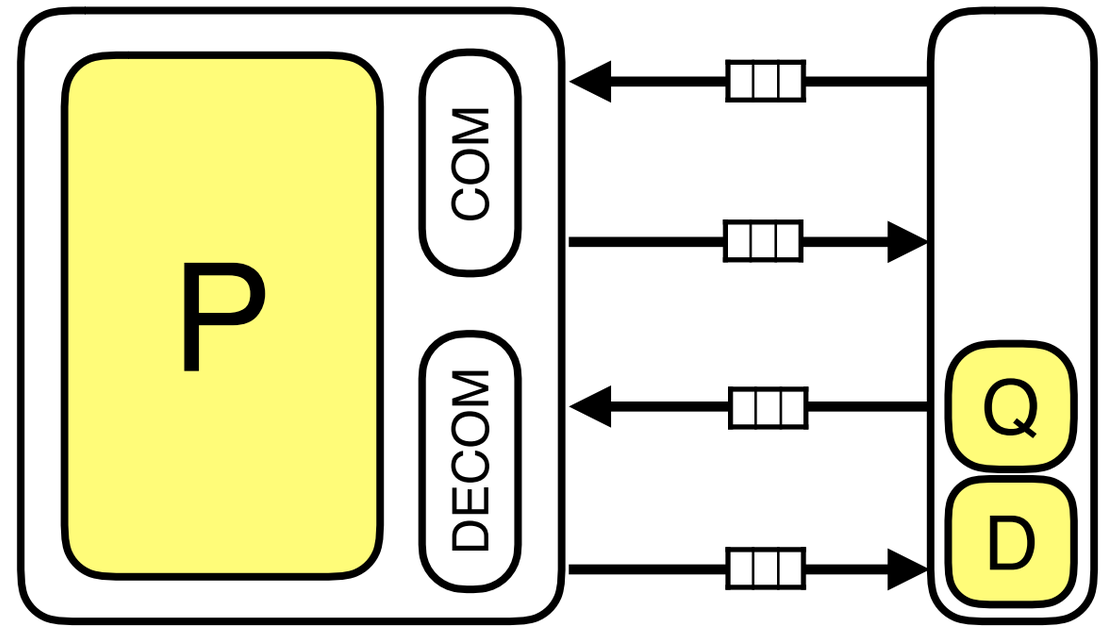
Achieving Non-Interactivity

Probabilistically Checkable Proof
[BFLS91][FGLSS96][AS92][ALMSS92]



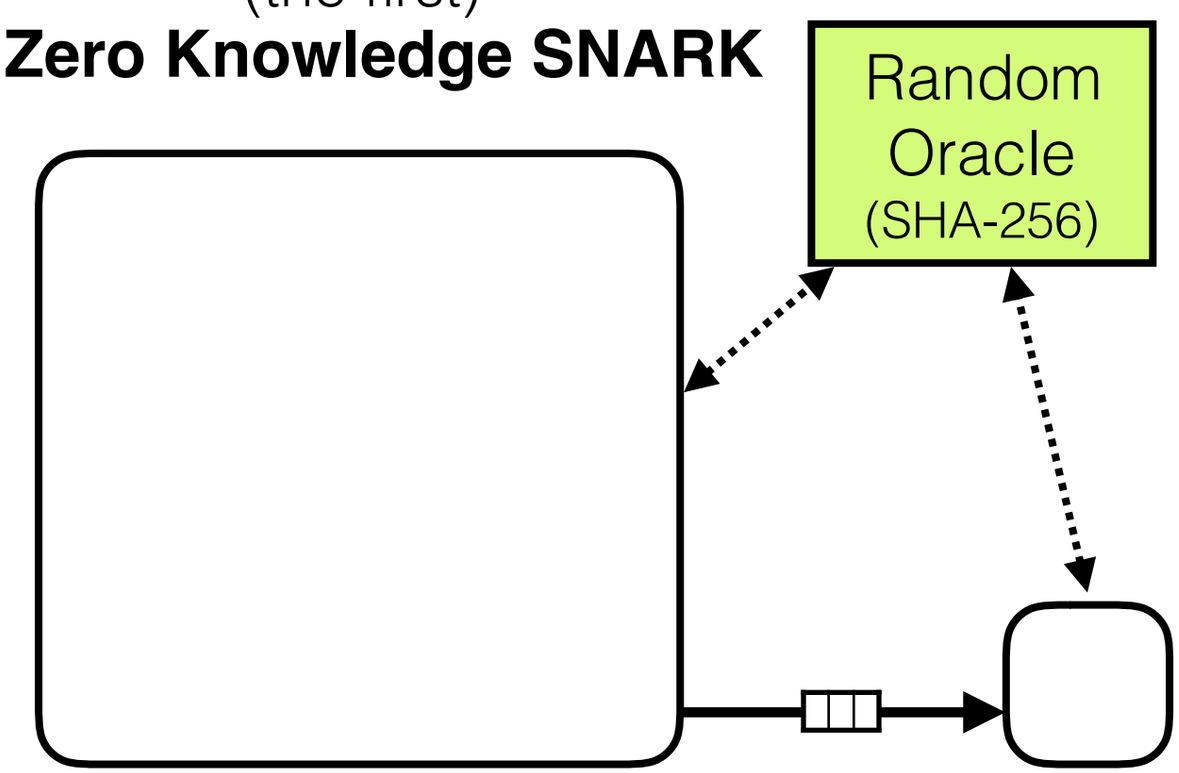
[Kilian92]

Zero Knowledge Succinct Proof



[Micali94]

(the first)
Zero Knowledge SNARK



TOFIX

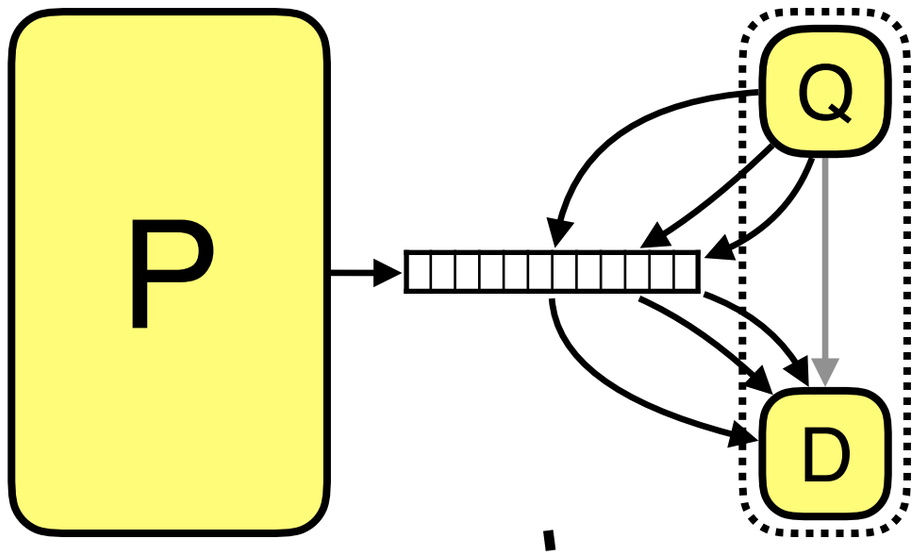
interactive

~~not succinct~~

bad concrete efficiency

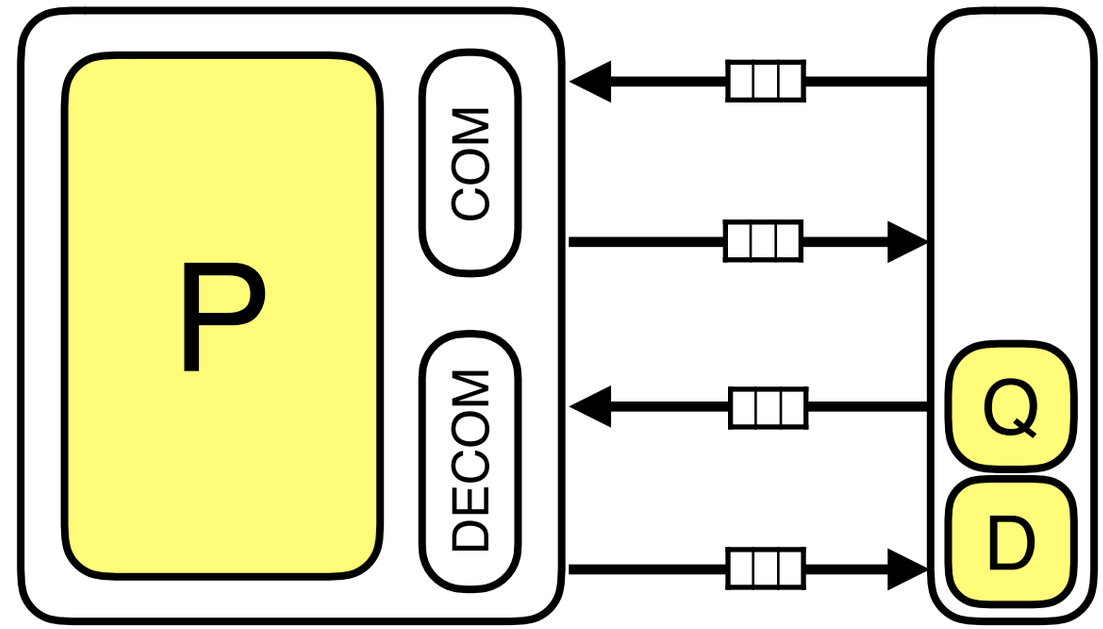
Achieving Non-Interactivity

Probabilistically Checkable Proof
[BFLS91][FGLSS96][AS92][ALMSS92]



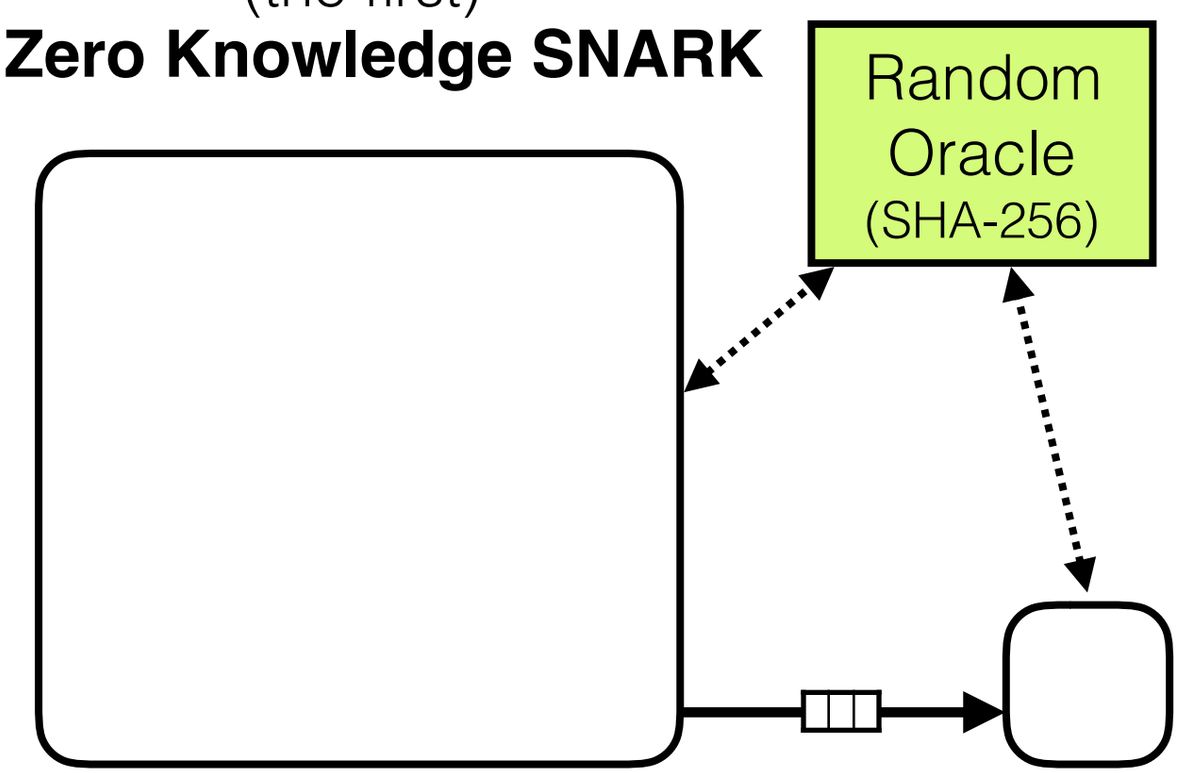
[Kilian92]

Zero Knowledge Succinct Proof



[Micali94]

(the first)
Zero Knowledge SNARK



TOFIX

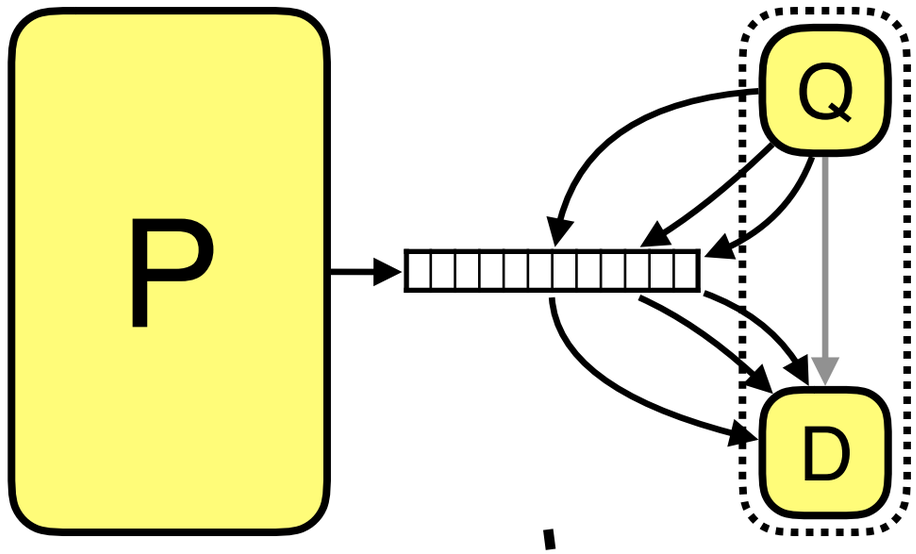
~~**interactive**~~

~~**not succinct**~~

bad concrete efficiency

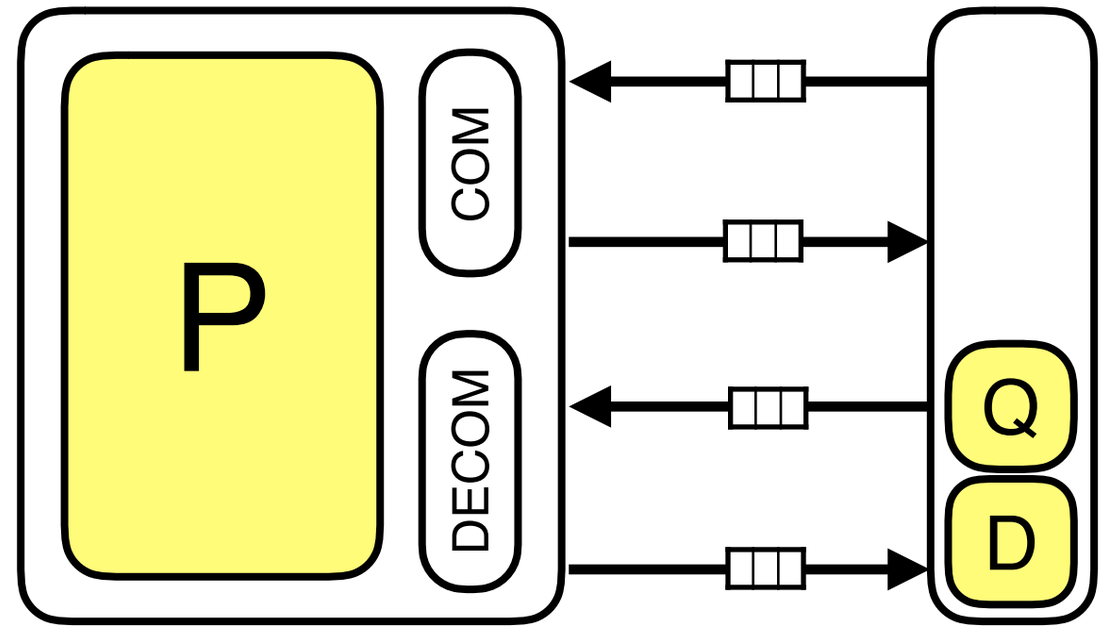
Achieving Non-Interactivity

Probabilistically Checkable Proof
[BFLS91][FGLSS96][AS92][ALMSS92]



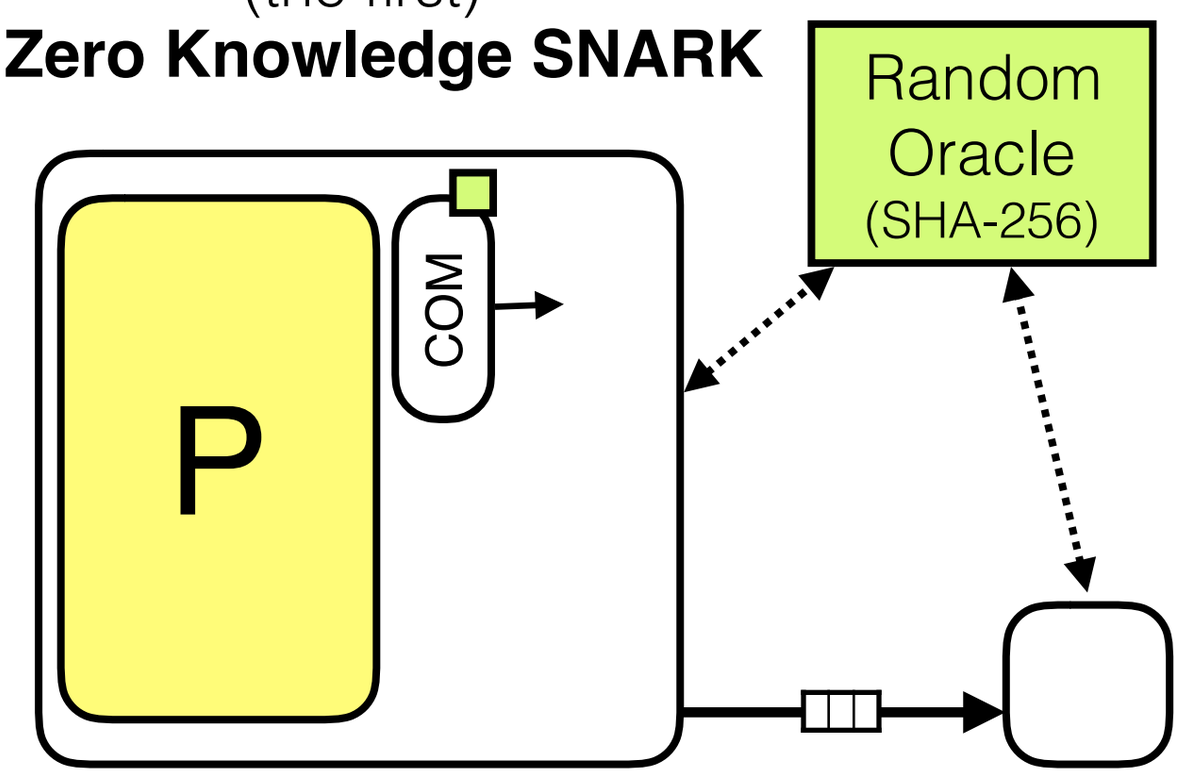
[Kilian92]

Zero Knowledge Succinct Proof



[Micali94]

(the first)
Zero Knowledge SNARK



TOFIX

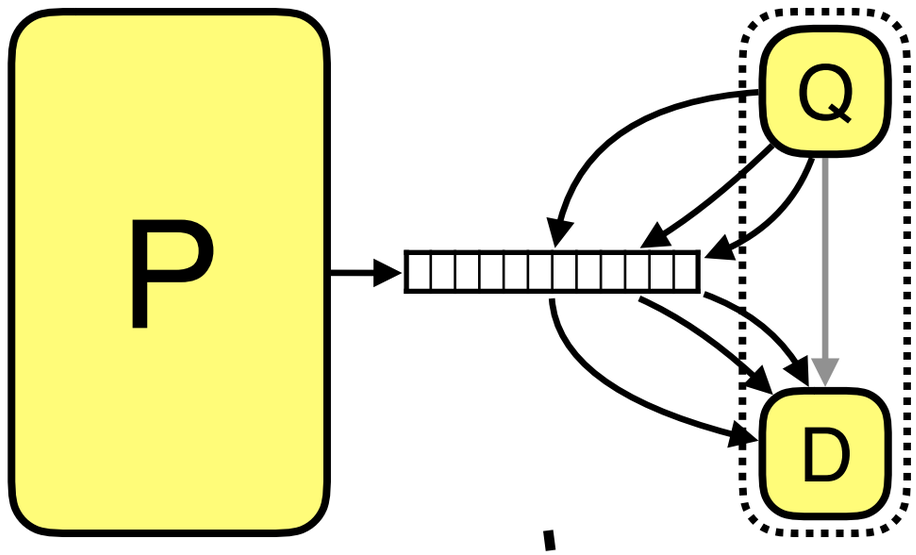
~~**interactive**~~

~~**not succinct**~~

bad concrete efficiency

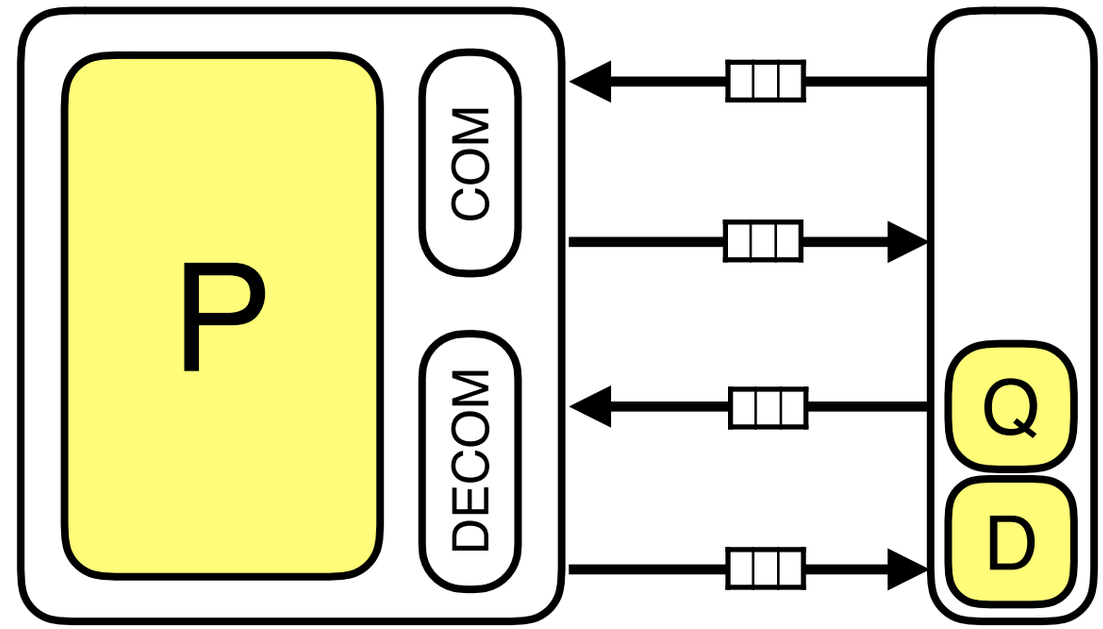
Achieving Non-Interactivity

Probabilistically Checkable Proof
[BFLS91][FGLSS96][AS92][ALMSS92]



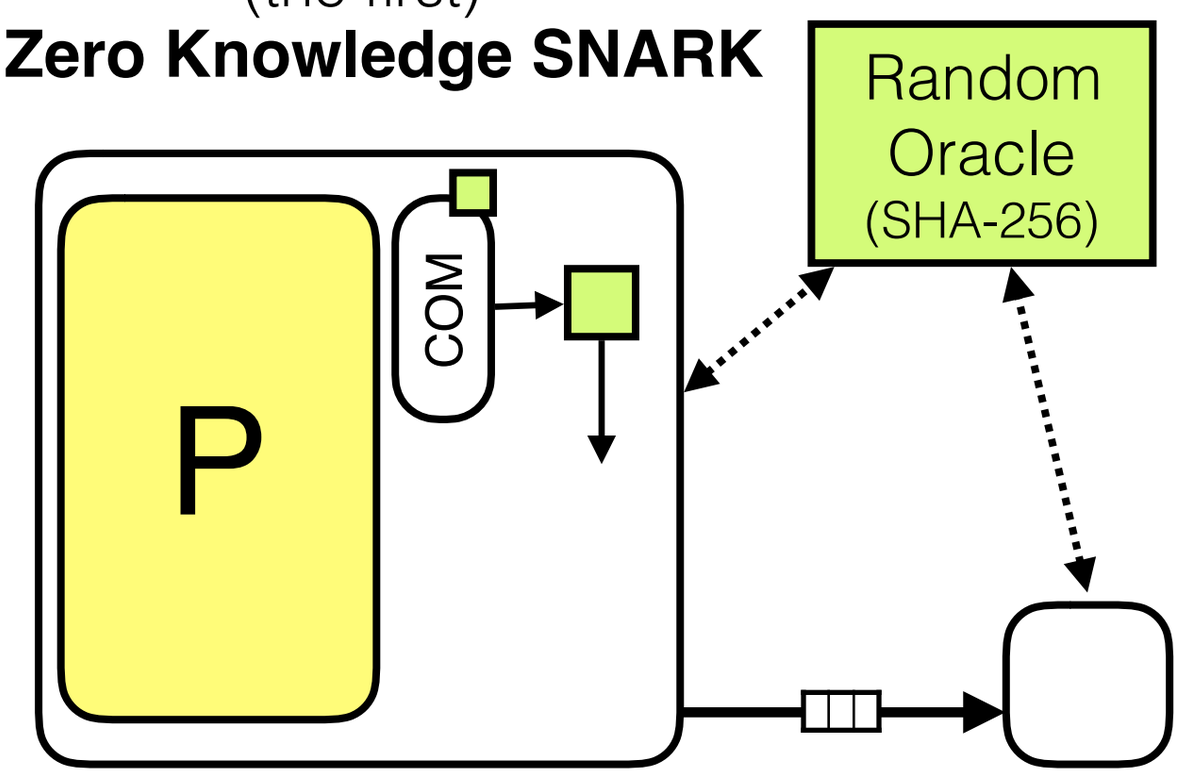
[Kilian92]

Zero Knowledge Succinct Proof



[Micali94]

(the first)
Zero Knowledge SNARK



TOFIX

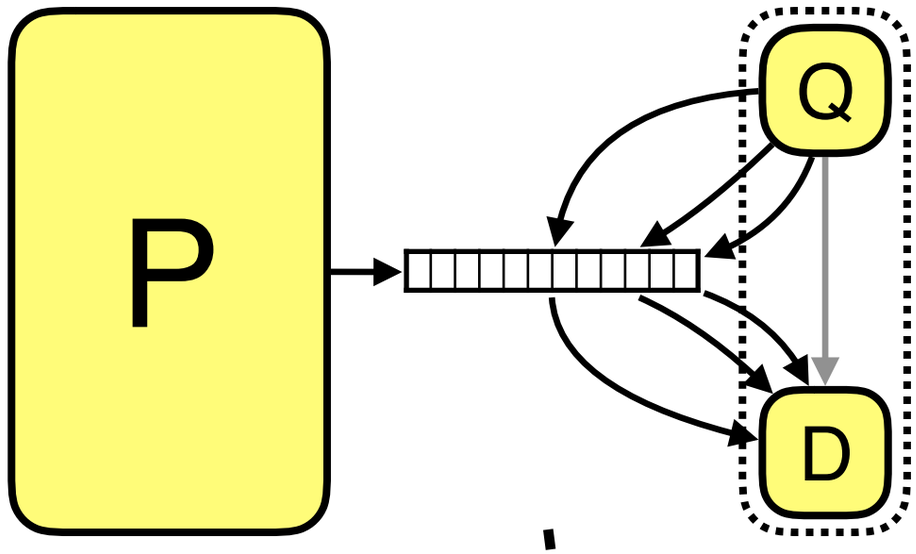
~~**interactive**~~

~~**not succinct**~~

bad concrete efficiency

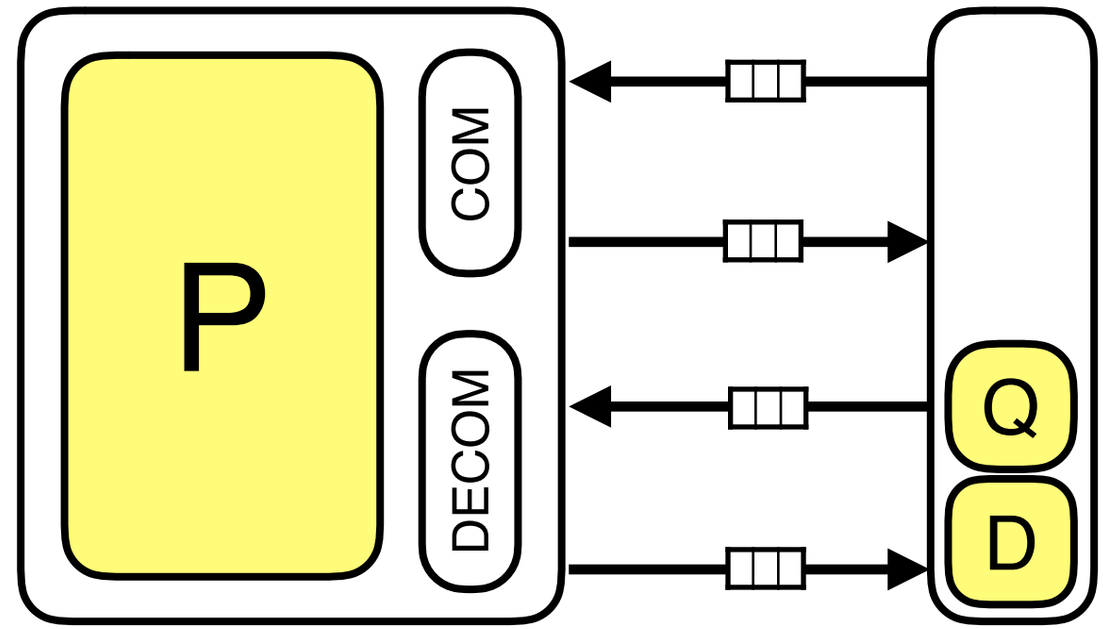
Achieving Non-Interactivity

Probabilistically Checkable Proof
[BFLS91][FGLSS96][AS92][ALMSS92]



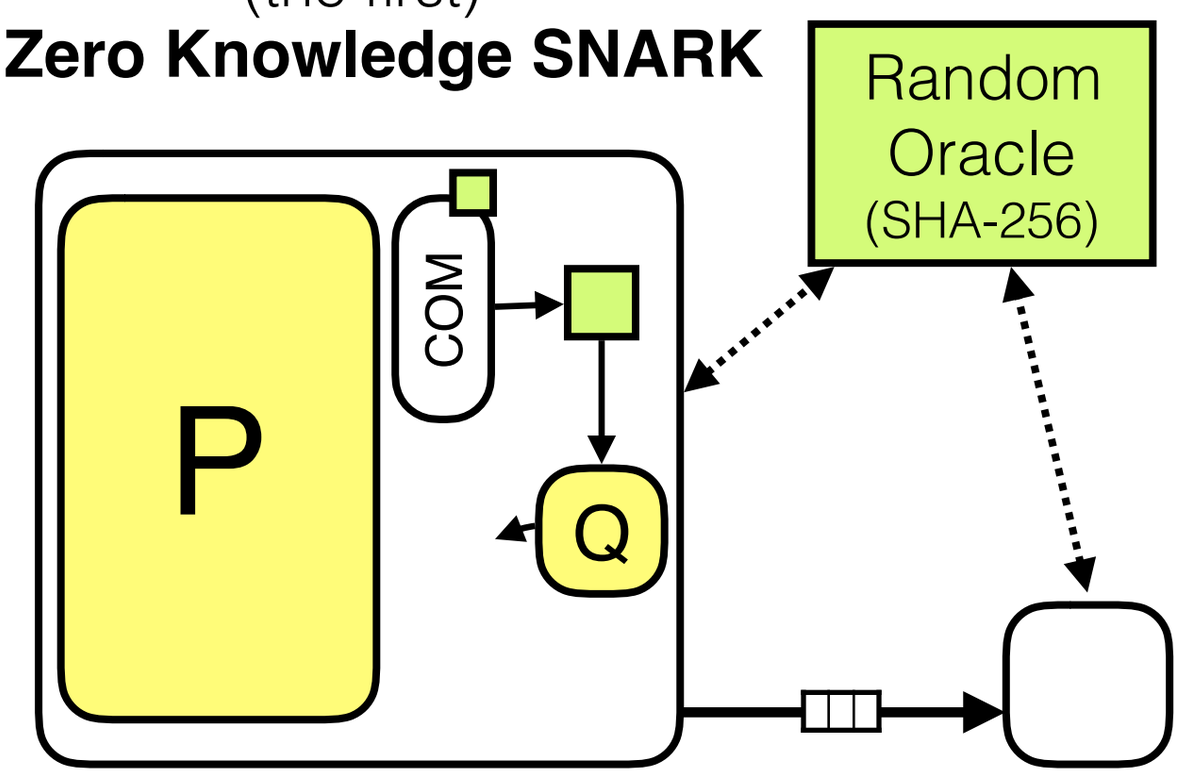
[Kilian92]

Zero Knowledge Succinct Proof



[Micali94]

(the first)
Zero Knowledge SNARK



TOFIX

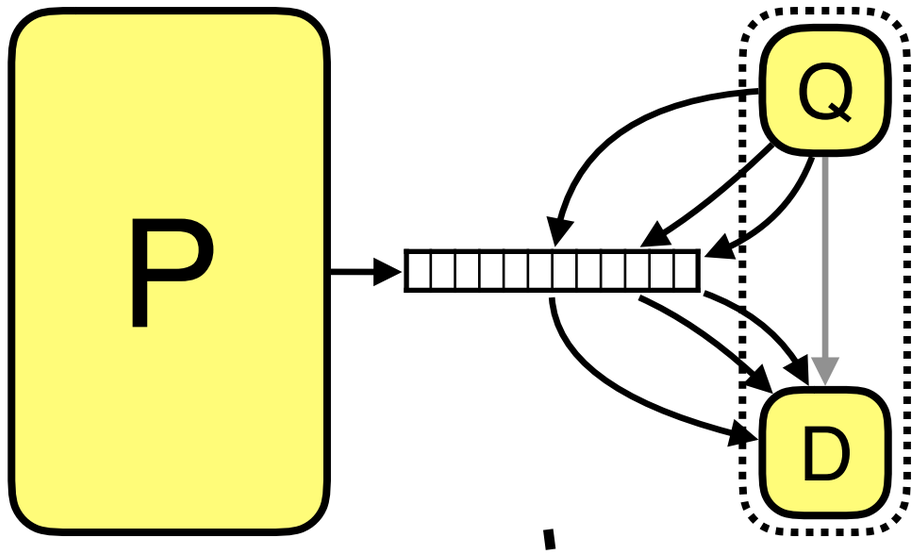
~~**interactive**~~

~~**not succinct**~~

bad concrete efficiency

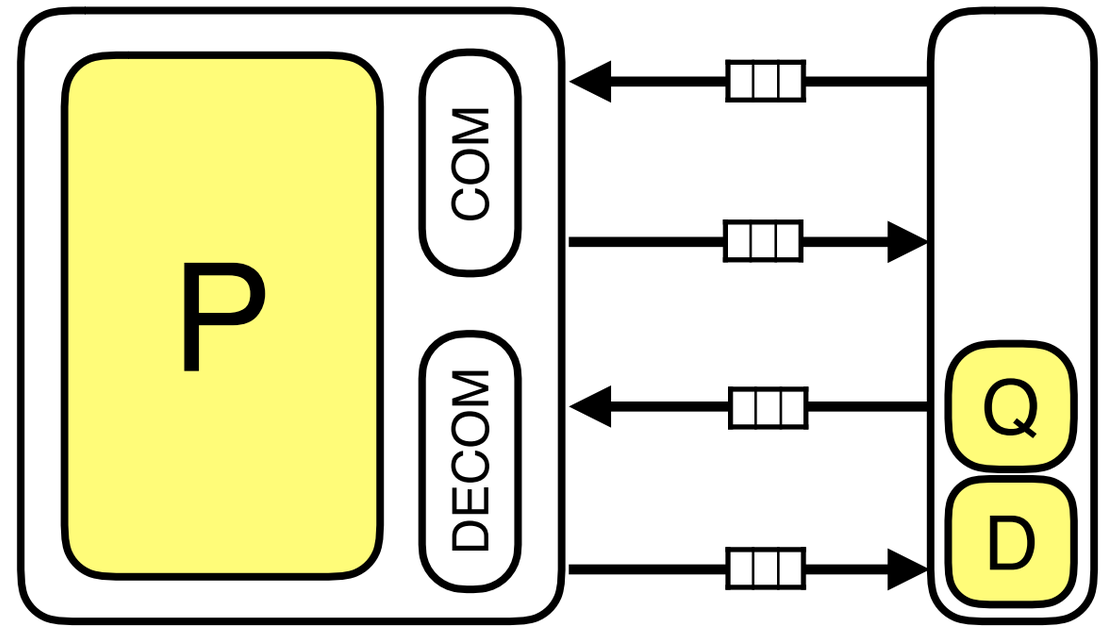
Achieving Non-Interactivity

Probabilistically Checkable Proof
[BFLS91][FGLSS96][AS92][ALMSS92]



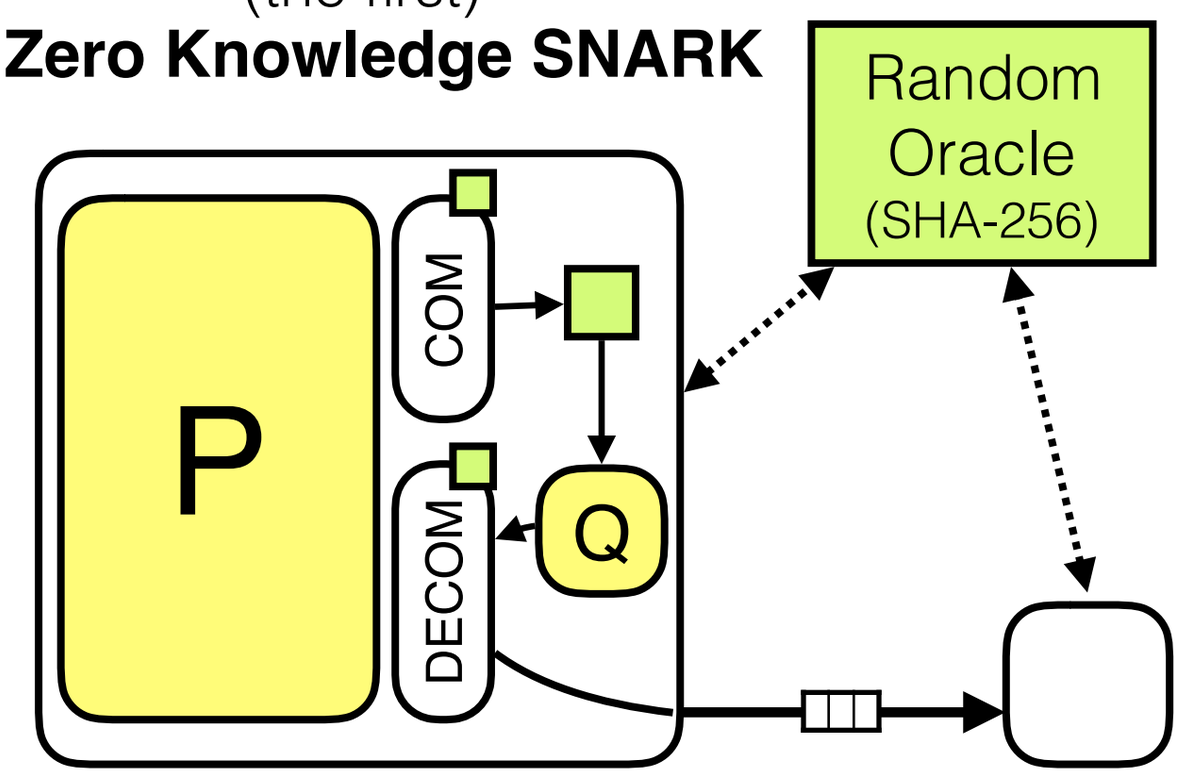
[Kilian92]

Zero Knowledge Succinct Proof



[Micali94]

(the first)
Zero Knowledge SNARK



TOFIX

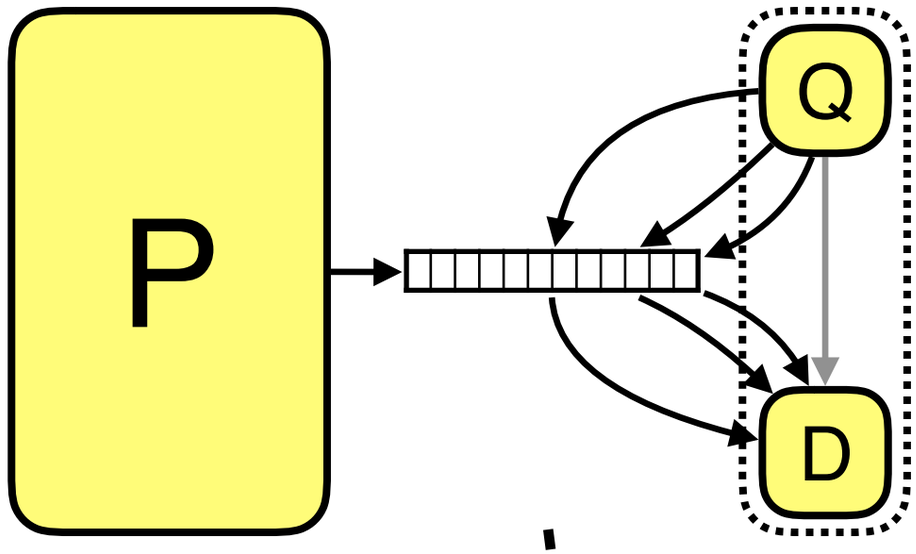
~~**interactive**~~

~~**not succinct**~~

bad concrete efficiency

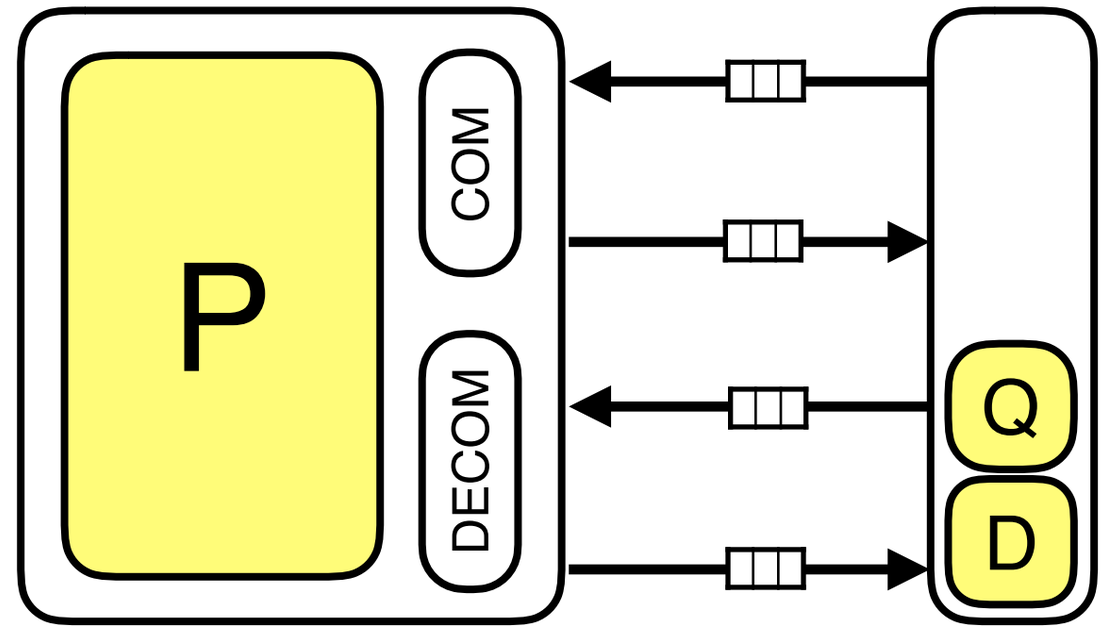
Achieving Non-Interactivity

Probabilistically Checkable Proof
[BFLS91][FGLSS96][AS92][ALMSS92]



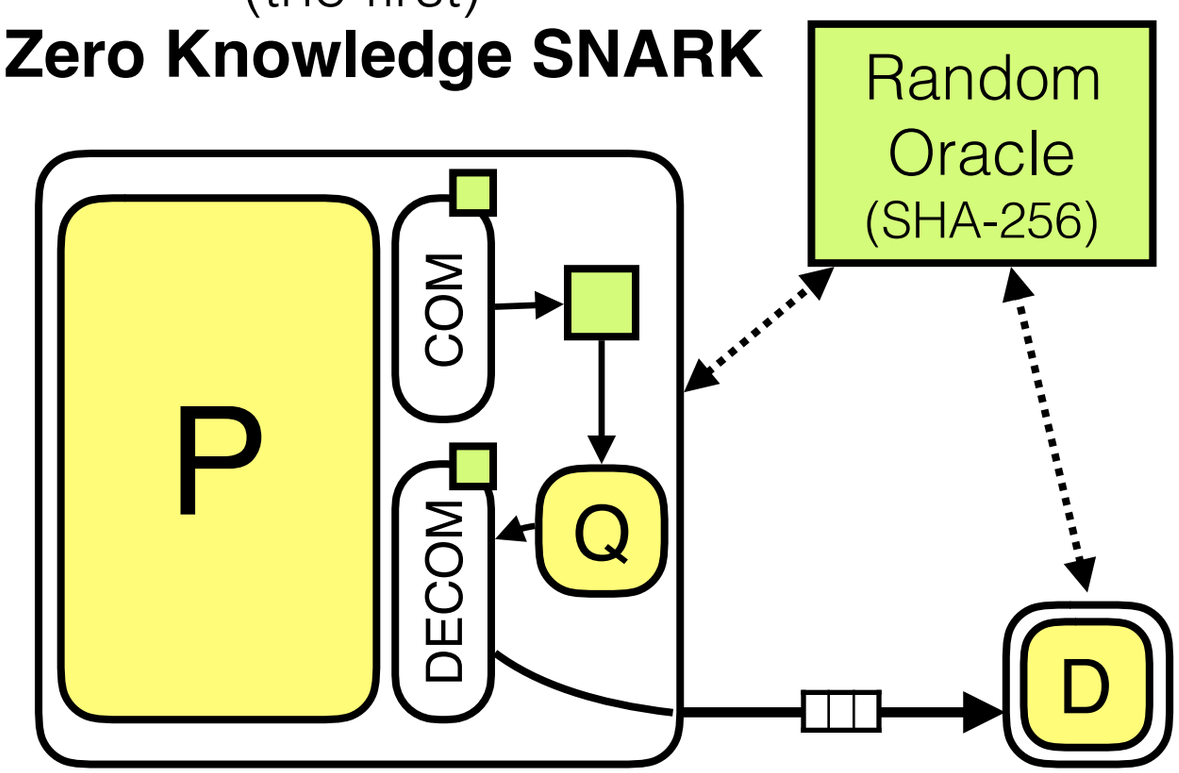
[Kilian92]

Zero Knowledge Succinct Proof



[Micali94]

(the first)
Zero Knowledge SNARK



TOFIX

~~**interactive**~~

~~**not succinct**~~

bad concrete efficiency

The Quest for ZK-SNARKs without Random Oracles

The Quest for ZK-SNARKs without Random Oracles

Negative result: constructing them "requires strong assumptions" [GW11]

The Quest for ZK-SNARKs without Random Oracles

Negative result: constructing them "requires strong assumptions" [GW11]

Positive results (under strong assumptions):

The Quest for ZK-SNARKs without Random Oracles

Negative result: constructing them "requires strong assumptions" [GW11]

Positive results (under strong assumptions):

Knowledge of Exponent
[D 92]

The Quest for ZK-SNARKs without Random Oracles

Negative result: constructing them "requires strong assumptions" [GW11]

Positive results (under strong assumptions):

Knowledge of Exponent
[D 92]

The Quest for ZK-SNARKs without Random Oracles

Negative result: constructing them "requires strong assumptions" [GW11]

Positive results (under strong assumptions):

Knowledge of Exponent
[D 92]

Extractable Hash Functions

The Quest for ZK-SNARKs without Random Oracles

Negative result: constructing them "requires strong assumptions" [GW11]

Positive results (under strong assumptions):

Knowledge of Exponent
[D 92]

Extractable Hash Functions

[BC**C**T 12]
[DFH 12]
[GLR 12]
[B**C** 12]
[BC**C**T 13]
[BC**C**GLRT 16]

The Quest for ZK-SNARKs without Random Oracles

Negative result: constructing them "requires strong assumptions" [GW11]

Positive results (under strong assumptions):

Knowledge of Exponent
[D 92]

Extractable Hash Functions

[BC**C**T 12]
[DFH 12]
[GLR 12]
[B**C** 12]
[BC**C**T 13]
[BC**C**GLRT 16]

based on **PCPs**

The Quest for ZK-SNARKs without Random Oracles

Negative result: constructing them "requires strong assumptions" [GW11]

Positive results (under strong assumptions):

Knowledge of Exponent
[D 92]

Extractable Hash Functions

[BC**C**T 12]
[DFH 12]
[GLR 12]
[B**C** 12]
[BC**C**T 13]
[BC**C**GLRT 16]

based on **PCPs**

Linear-Only Encryption/Encoding

The Quest for ZK-SNARKs without Random Oracles

Negative result: constructing them "requires strong assumptions" [GW11]

Positive results (under strong assumptions):

Knowledge of Exponent
[D 92]

Extractable Hash Functions

[BC**C**T 12]
[DFH 12]
[GLR 12]
[B**C** 12]
[BC**C**T 13]
[BC**C**GLRT 16]

based on **PCPs**

Linear-Only Encryption/Encoding

[G 10]
[L 12]
[B**C**IOP 13]
[GGPR 13]
[PGHR 13]
[B**C**GTV 13]
[B**C**TV 14a]
[DFGK 14]

[WSRBW 15]
[BBFR 15]
[B**C**TV 14b]
[**C**TV 15]
[CFHKKNPZ 15]
[BISW 17]
⋮

The Quest for ZK-SNARKs without Random Oracles

Negative result: constructing them "requires strong assumptions" [GW11]

Positive results (under strong assumptions):

Generic Group Model
[G 16]

Knowledge of Exponent
[D 92]

Extractable Hash Functions

[BC**C**T 12]
[DFH 12]
[GLR 12]
[B**C** 12]
[BC**C**T 13]
[BC**C**GLRT 16]

based on **PCPs**

Linear-Only Encryption/Encoding

[G 10]
[L 12]
[B**C**IOP 13]
[GGPR 13]
[PGHR 13]
[B**C**GTV 13]
[B**C**TV 14a]
[DFGK 14]

[WSRBW 15]
[BBFR 15]
[B**C**TV 14b]
[**C**TV 15]
[CFHKKNPZ 15]
[BISW 17]
⋮

The Quest for ZK-SNARKs without Random Oracles

Negative result: constructing them "requires strong assumptions" [GW11]

Positive results (under strong assumptions):

Generic Group Model
[G 16]

Knowledge of Exponent
[D 92]

Extractable Hash Functions

[BC**C**T 12]
[DFH 12]
[GLR 12]
[B**C** 12]
[BC**C**T 13]
[BC**C**GLRT 16]

based on **PCPs**

Linear-Only Encryption/Encoding

[G 10]
[L 12]
[B**C**IOP 13]
[GGPR 13]
[PGHR 13]
[B**C**GTV 13]
[B**C**TV 14a]
[DFGK 14]
[WSRBW 15]
[BBFR 15]
[B**C**TV 14b]
[**C**TV 15]
[CFHKKNPZ 15]
[BISW 17]
⋮

based on **linear PCPs**

The Quest for ZK-SNARKs without Random Oracles

Negative result: constructing them "requires strong assumptions" [GW11]

Positive results (under strong assumptions):

Generic Group Model
[G 16]

Knowledge of Exponent
[D 92]

Extractable Hash Functions

[BC**C**T 12]
[DFH 12]
[GLR 12]
[B**C** 12]
[BC**C**T 13]
[BC**C**GLRT 16]

based on **PCPs**

Linear-Only Encryption/Encoding

[G 10]
[L 12]
[B**C**IOP 13]
[GGPR 13]
[PGHR 13]
[B**C**GTV 13]
[B**C**TV 14a]
[DFGK 14]
[WSRBW 15]
[BBFR 15]
[B**C**TV 14b]
[**C**TV 15]
[CFHKKNPZ 15]
[BISW 17]
⋮

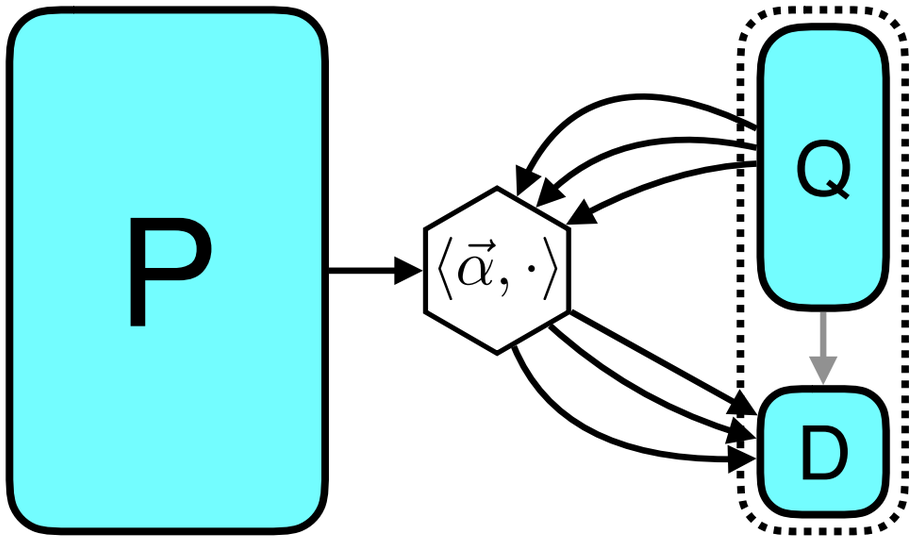
based on **linear PCPs**

strike a different tradeoff…

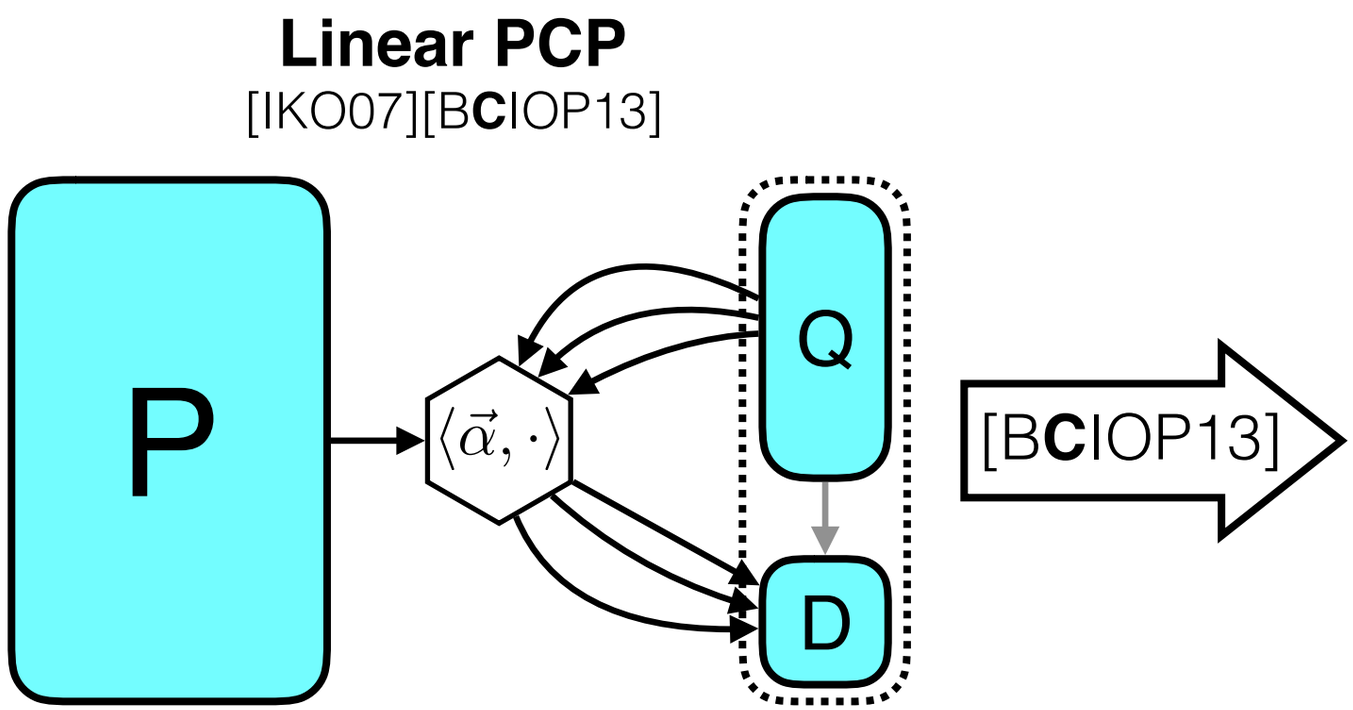
ZK-SNARKs from Linear PCPs

ZK-SNARKs from Linear PCPs

Linear PCP
[IKO07][B**C**IOP13]

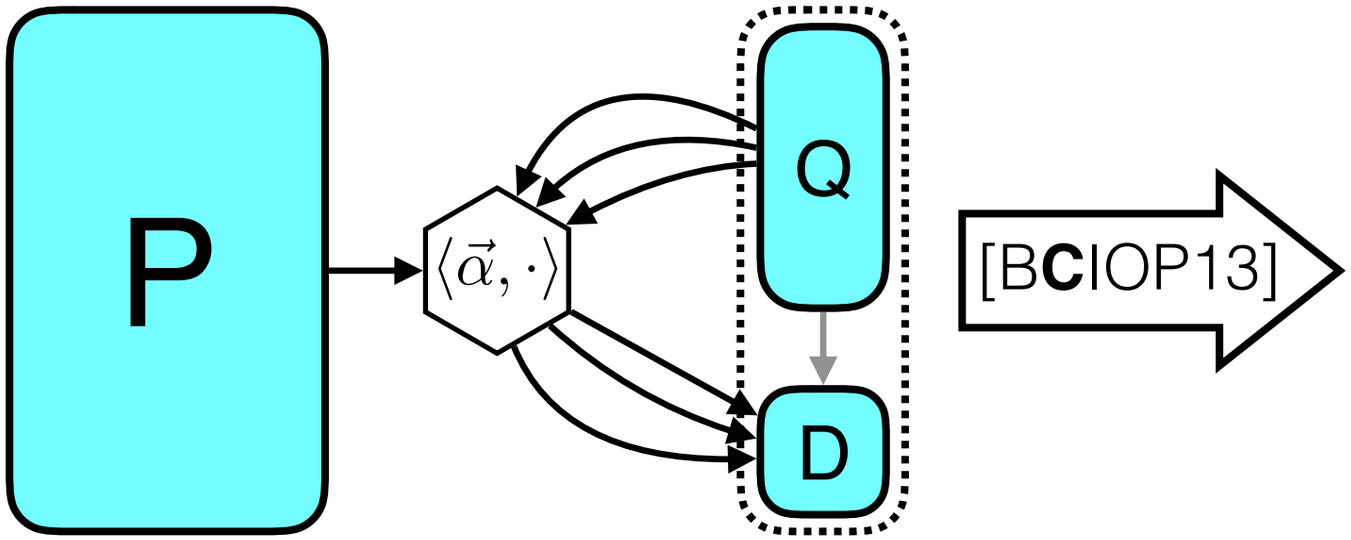


ZK-SNARKs from Linear PCPs



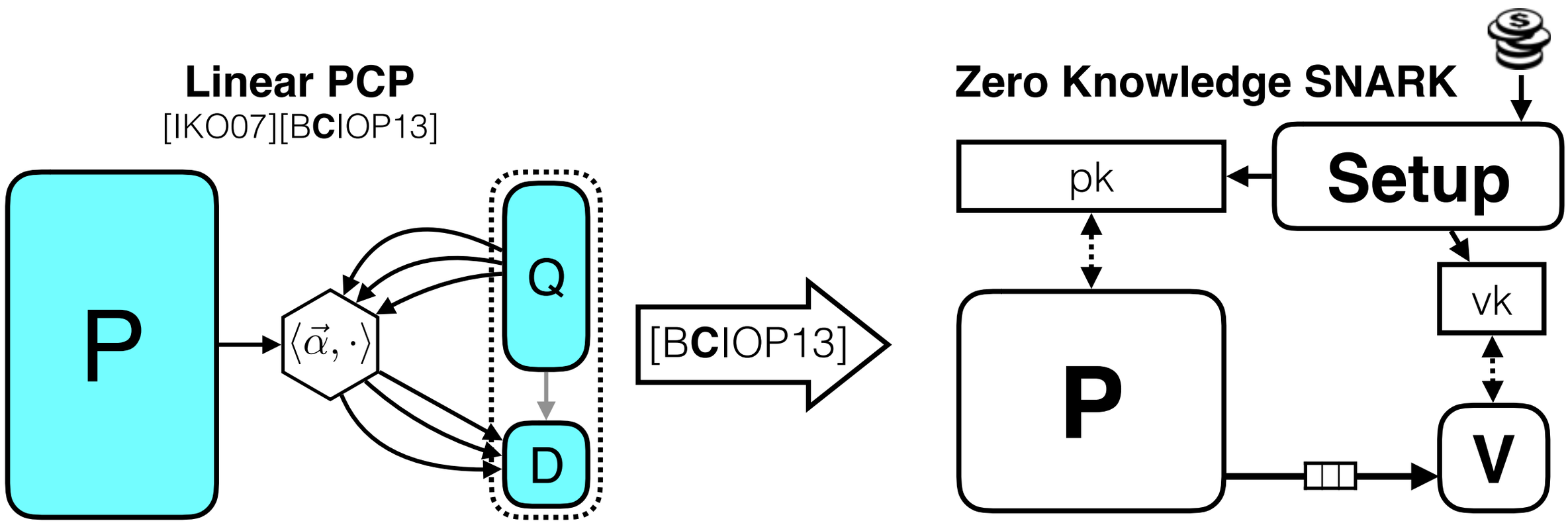
ZK-SNARKs from Linear PCPs

Linear PCP
[IKO07][B**C**IOP13]

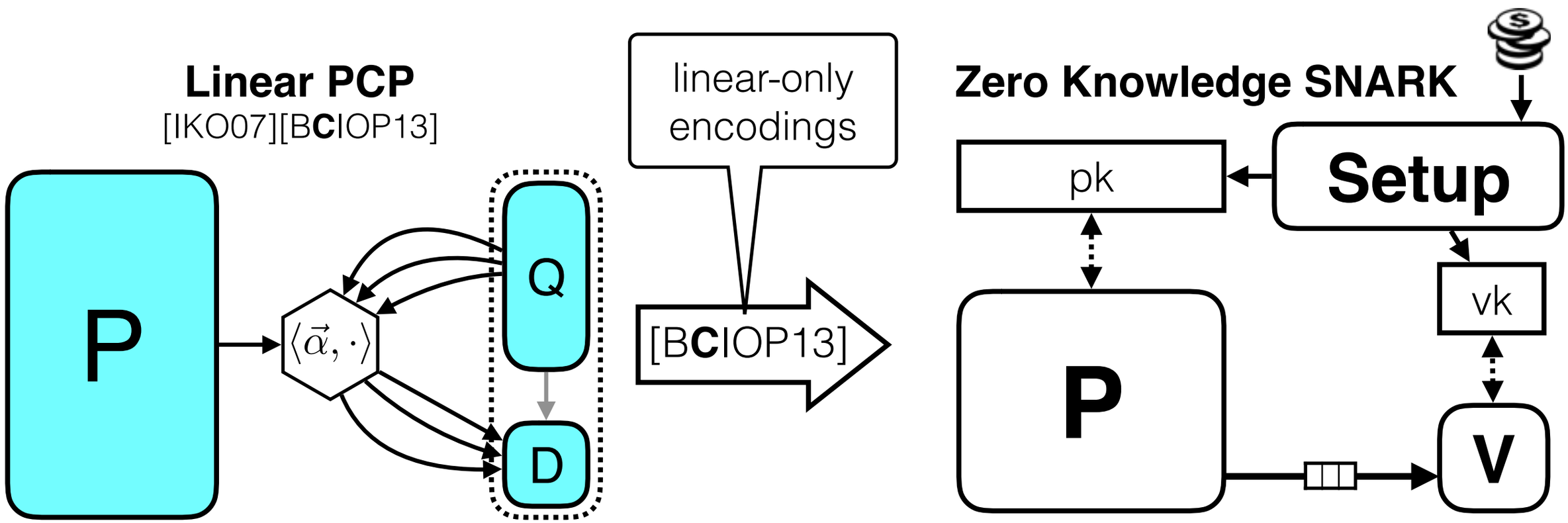


Zero Knowledge SNARK

ZK-SNARKs from Linear PCPs

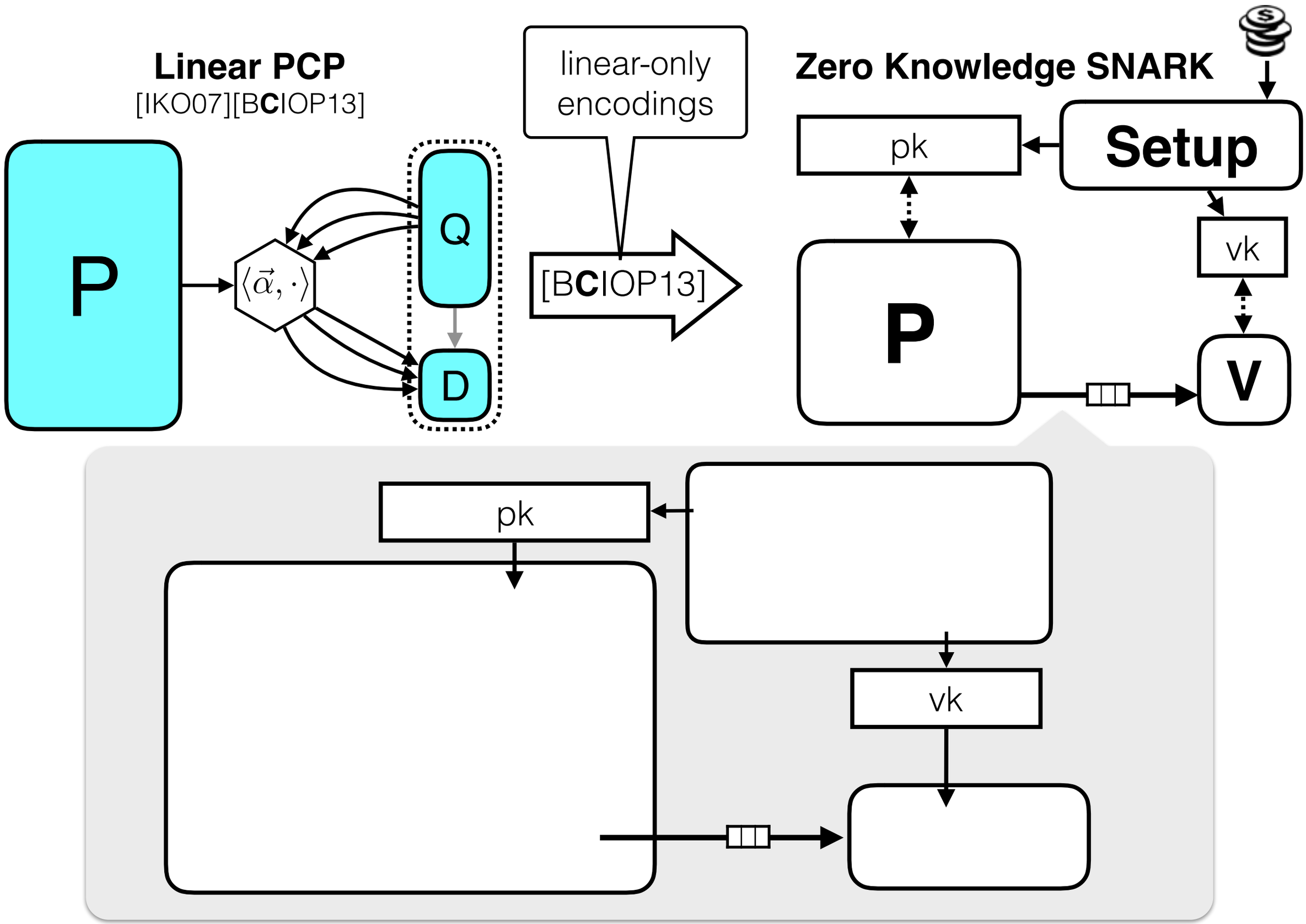


ZK-SNARKs from Linear PCPs

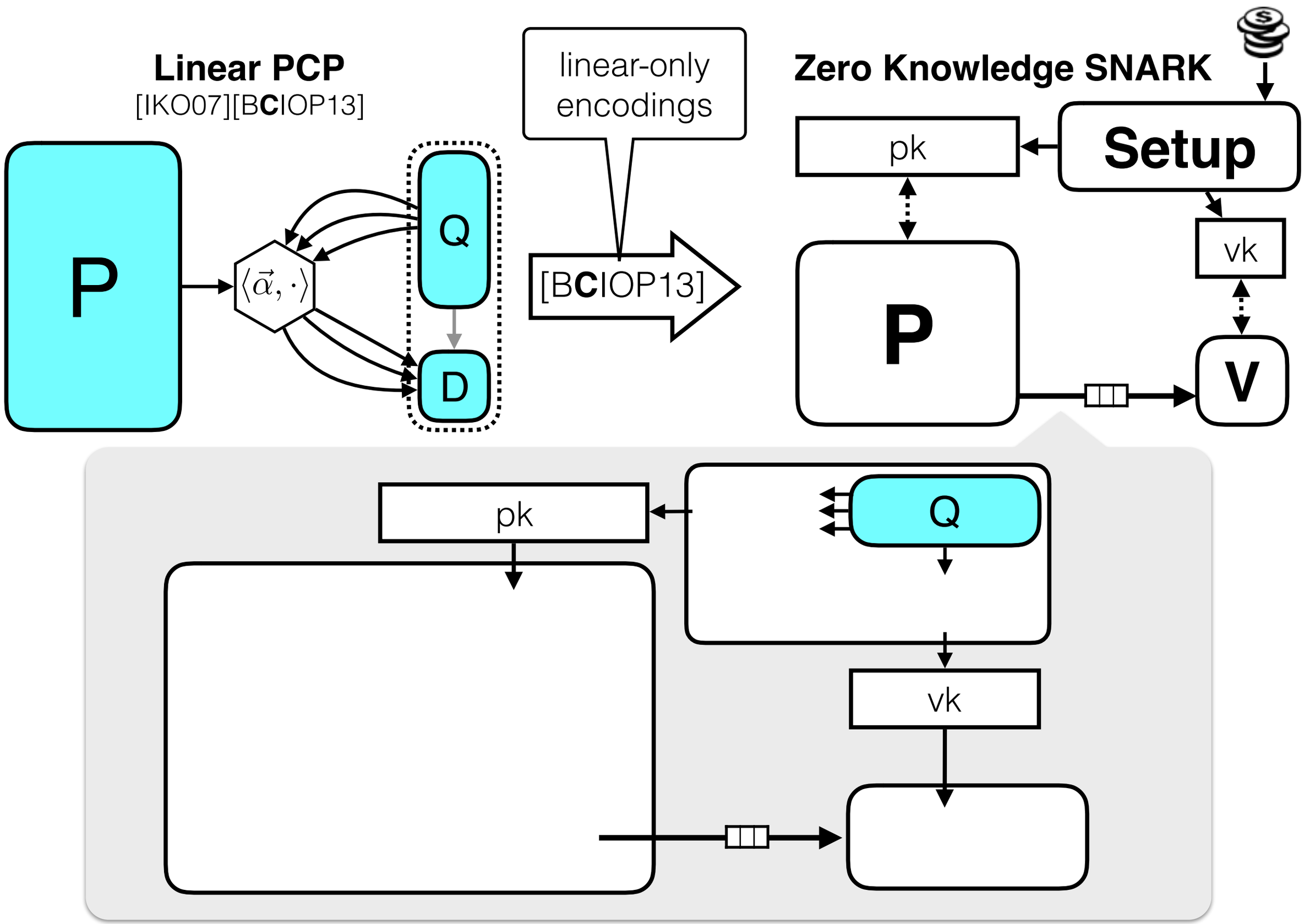


ZK-SNARKs from Linear PCPs

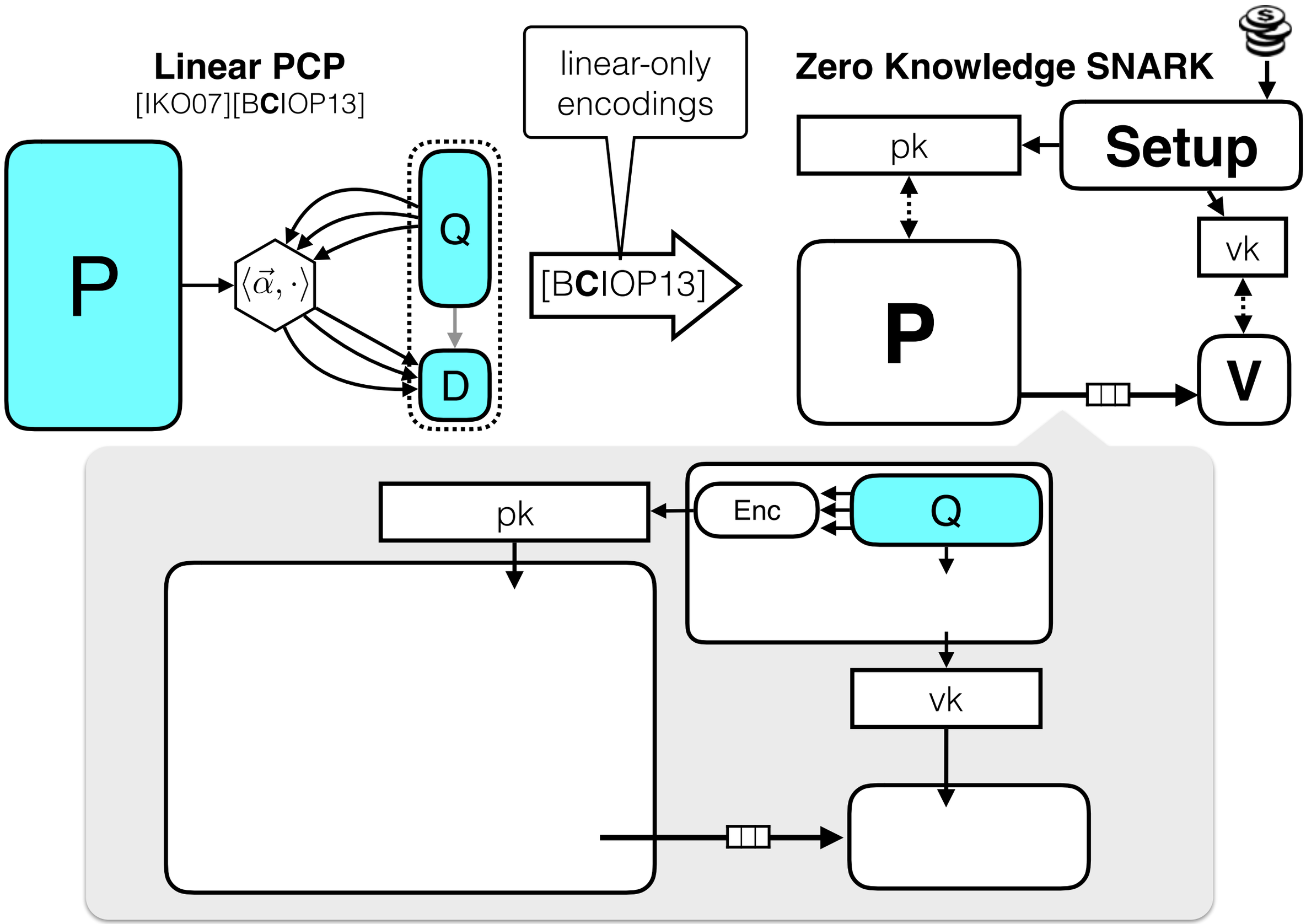
ZK-SNARKs from Linear PCPs



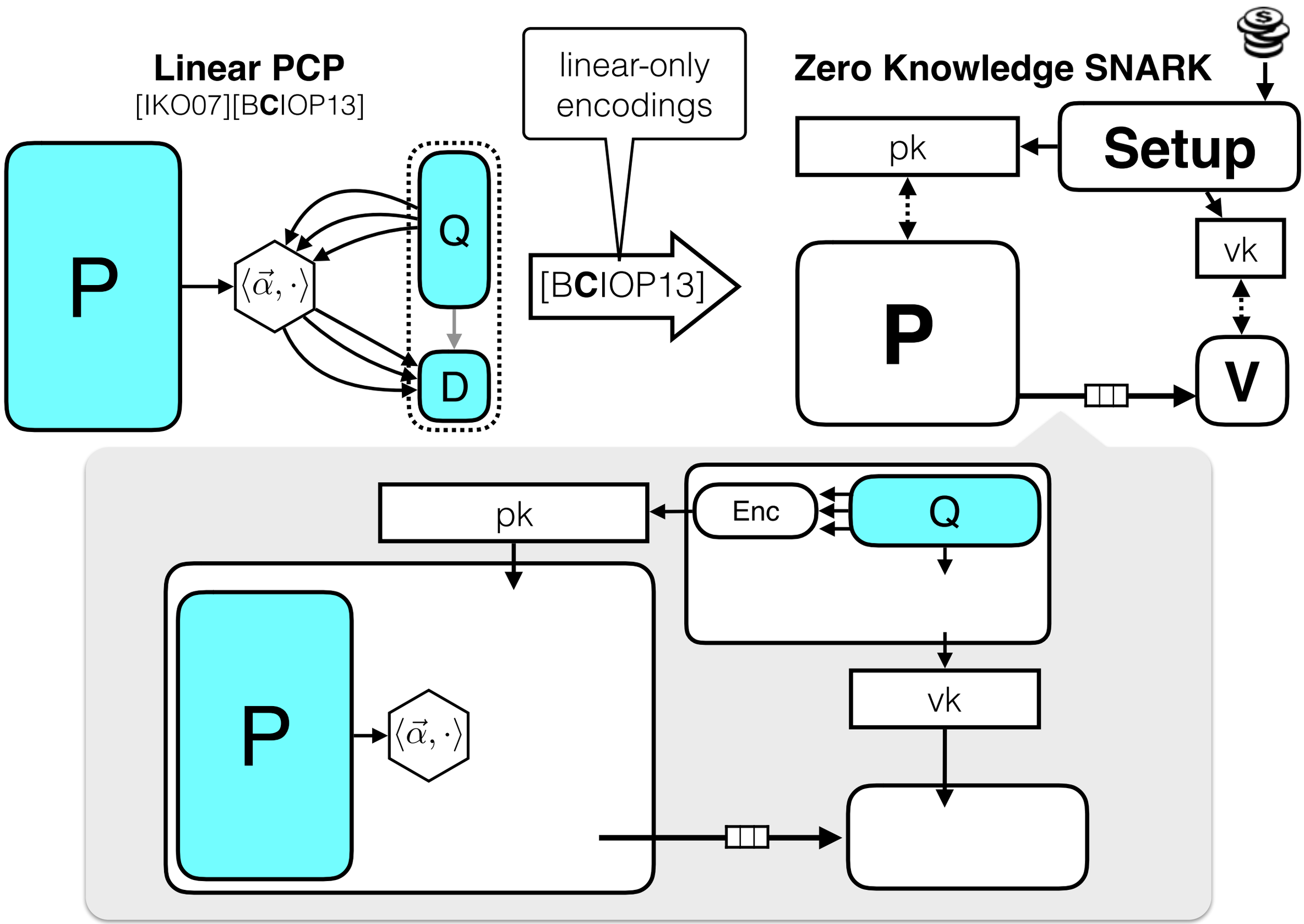
ZK-SNARKs from Linear PCPs



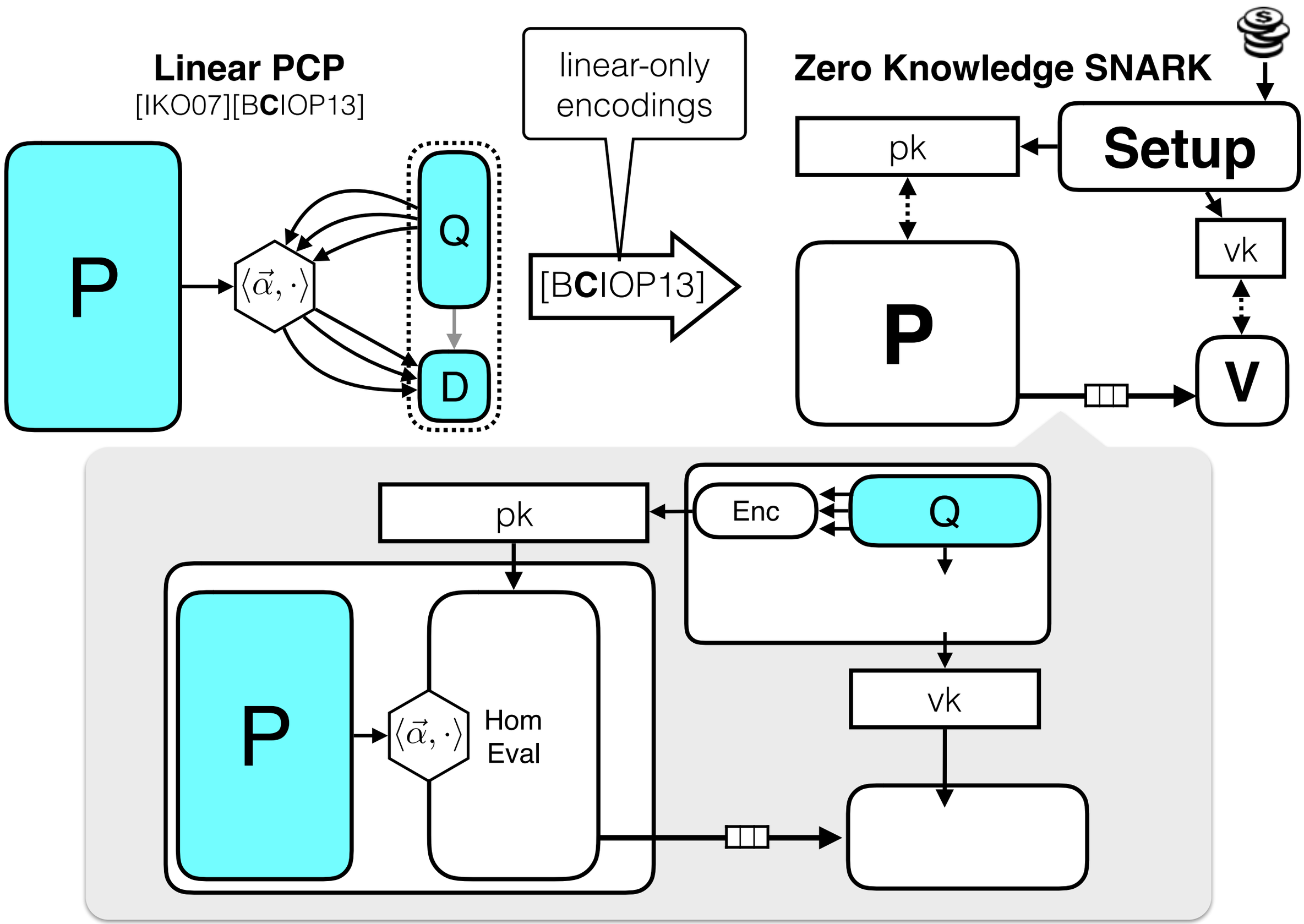
ZK-SNARKs from Linear PCPs



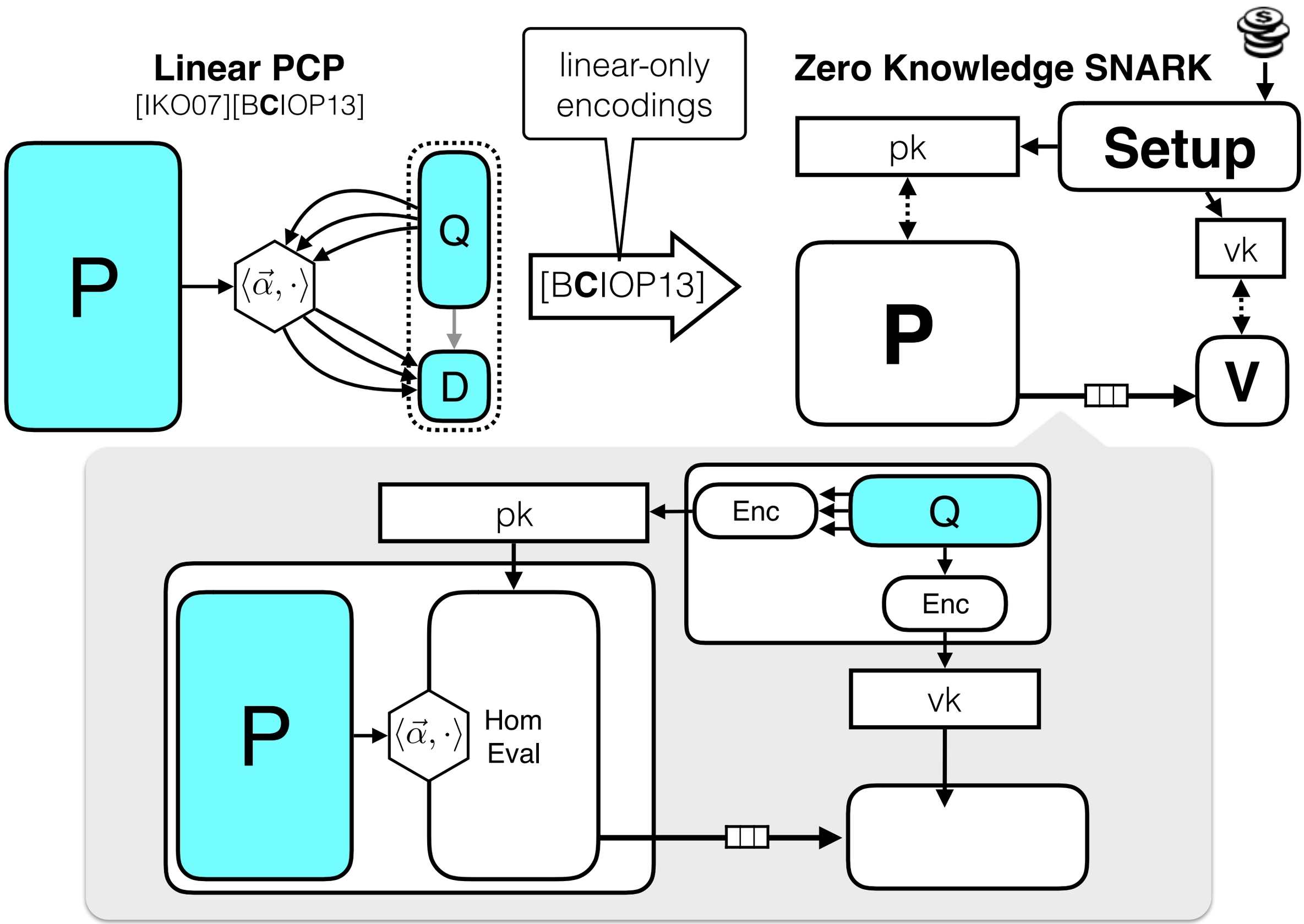
ZK-SNARKs from Linear PCPs



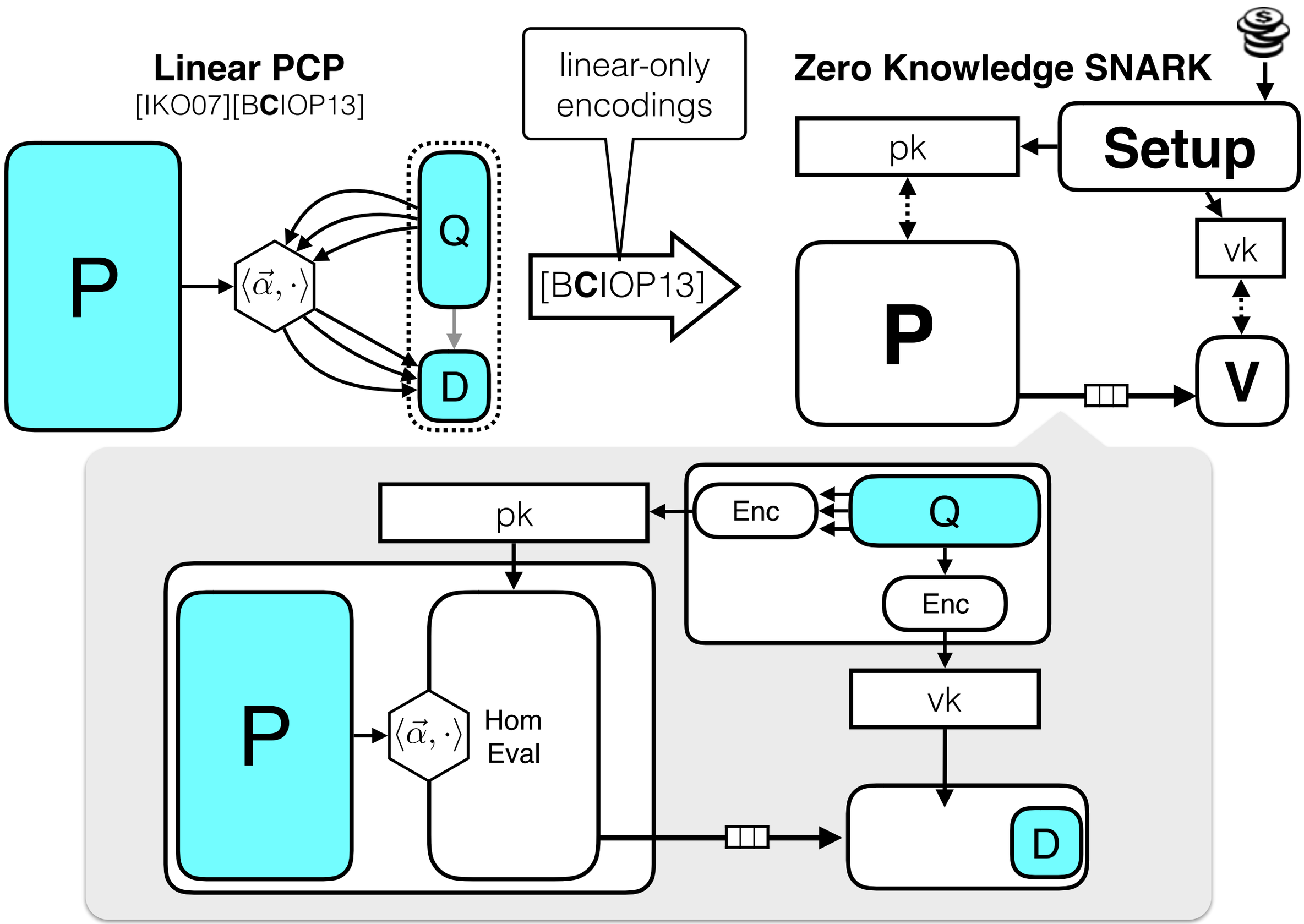
ZK-SNARKs from Linear PCPs



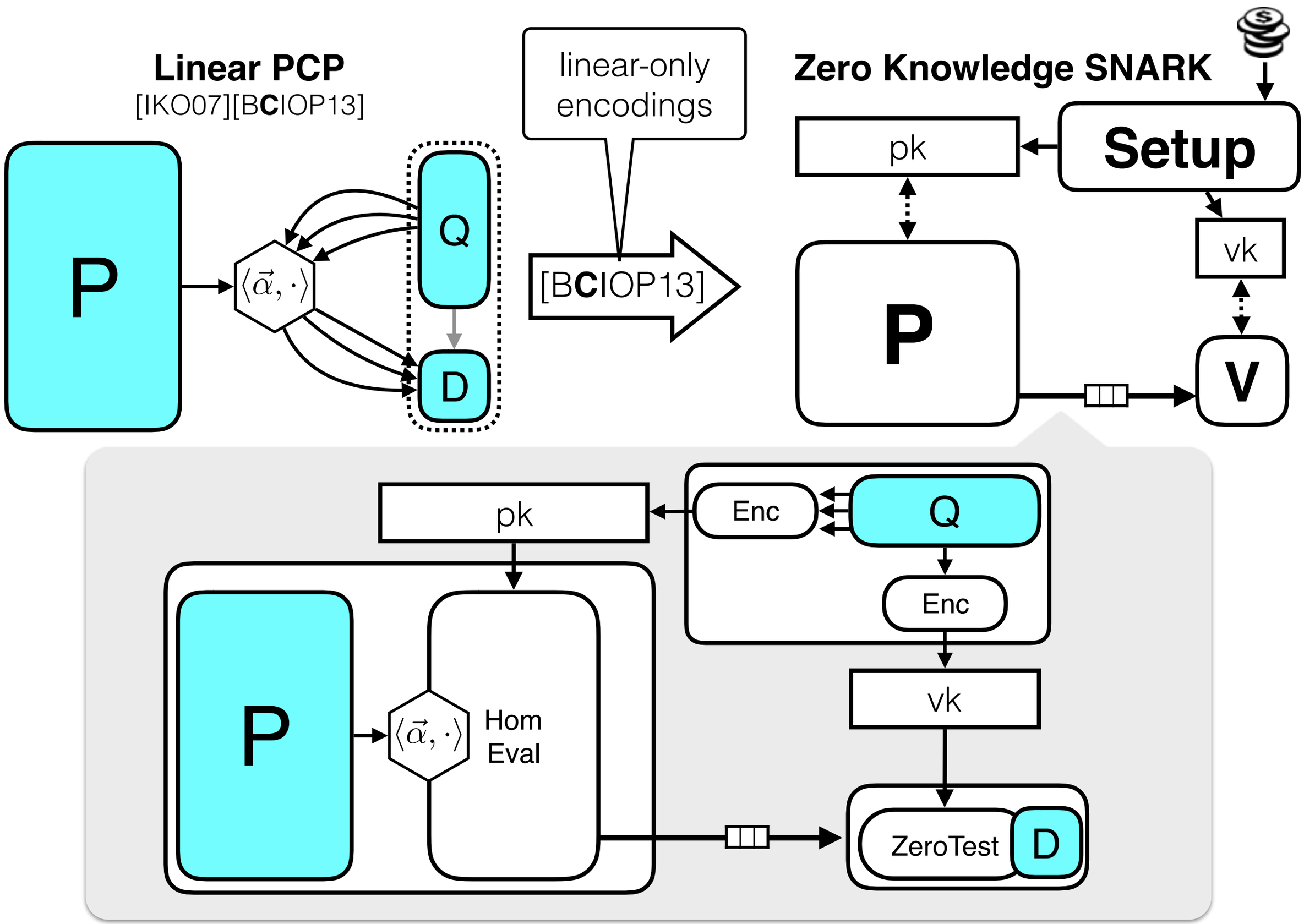
ZK-SNARKs from Linear PCPs



ZK-SNARKs from Linear PCPs

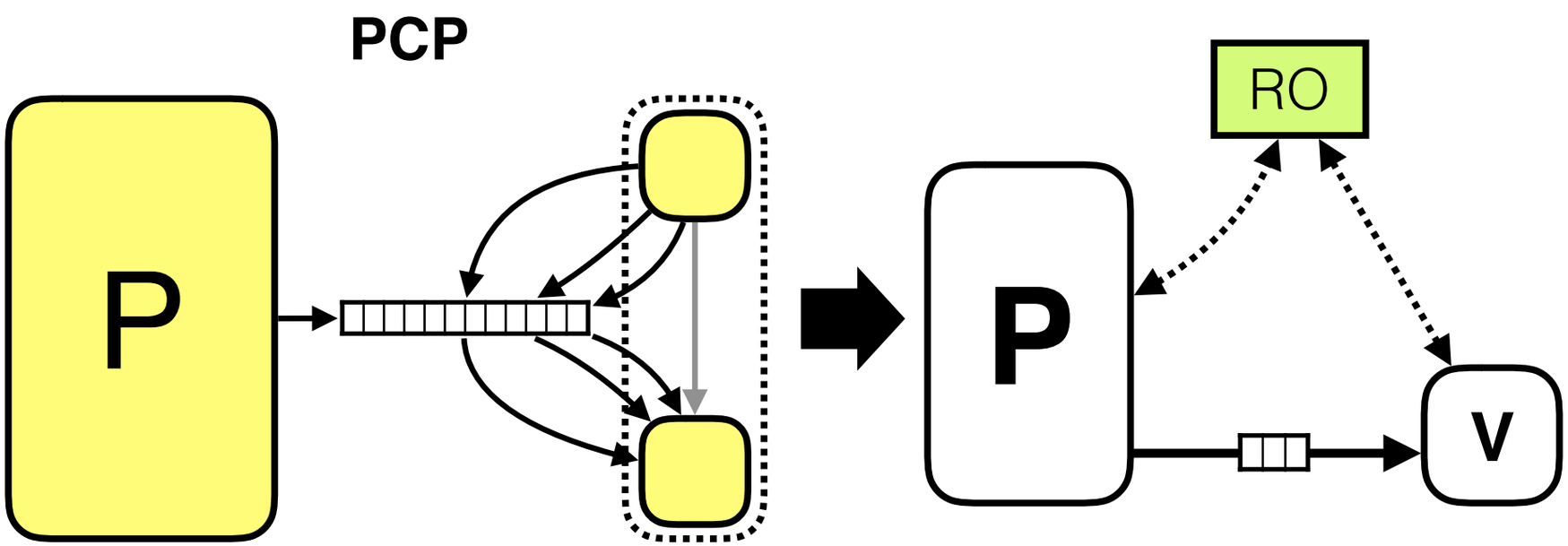


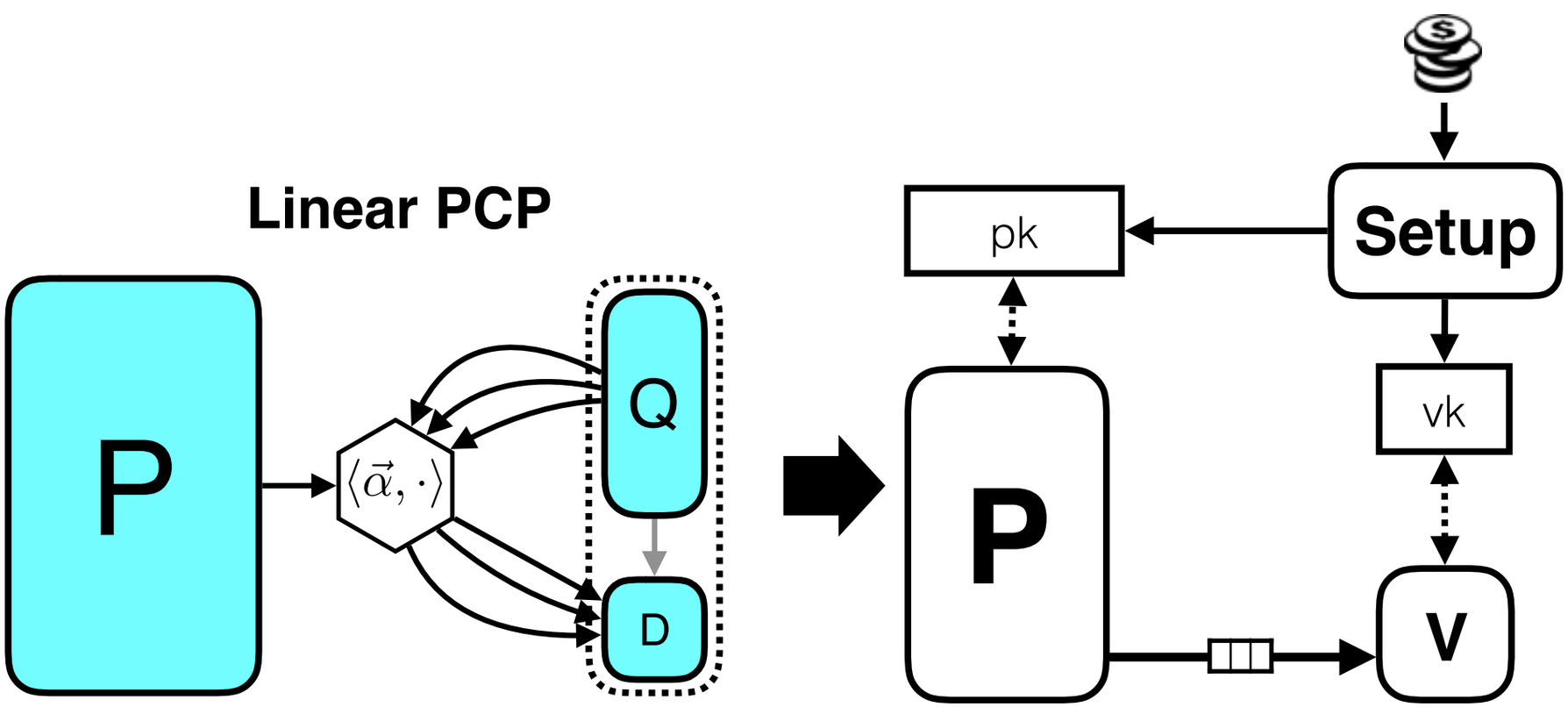
ZK-SNARKs from Linear PCPs



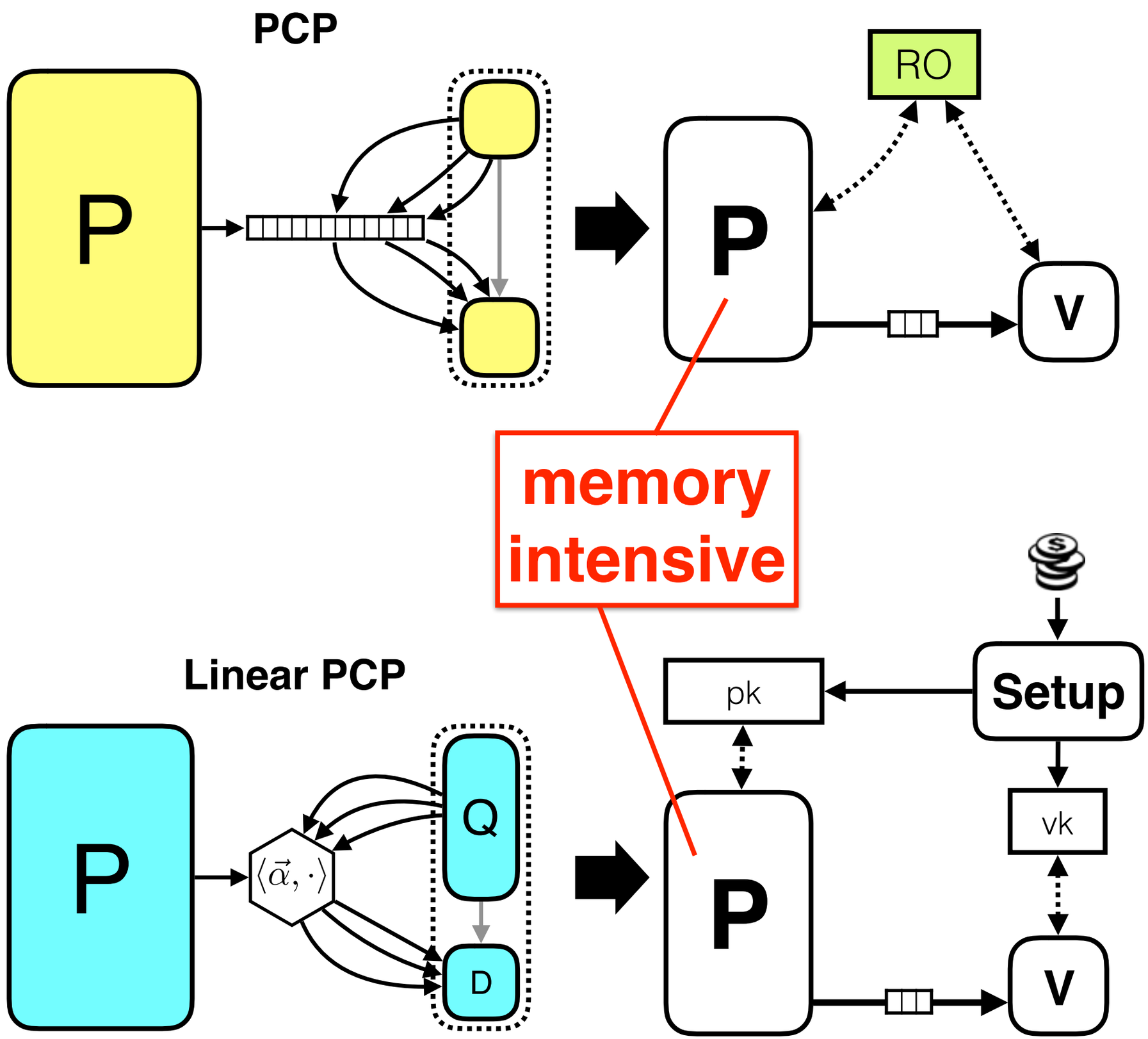
Which approach is better?

Which approach is better?

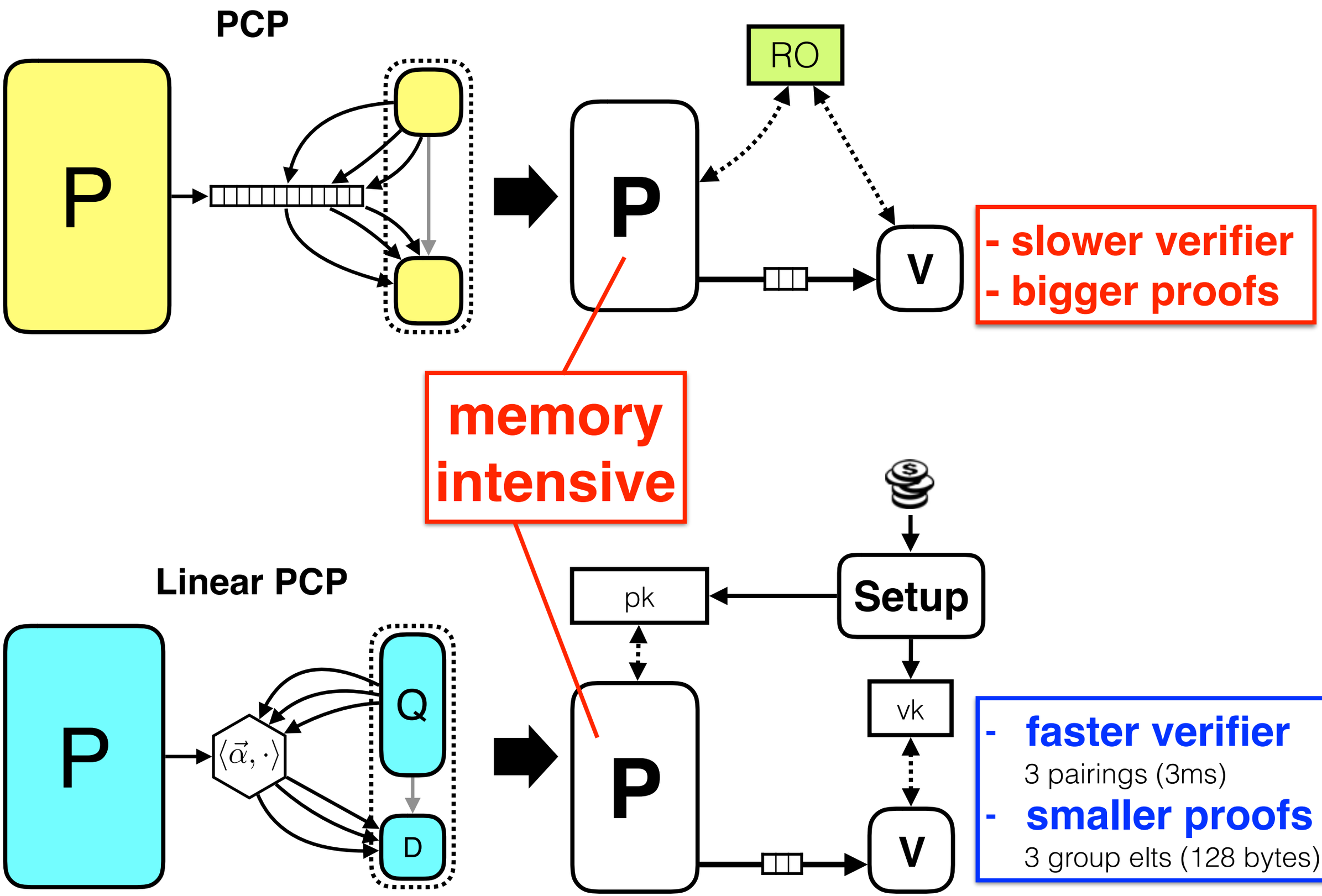




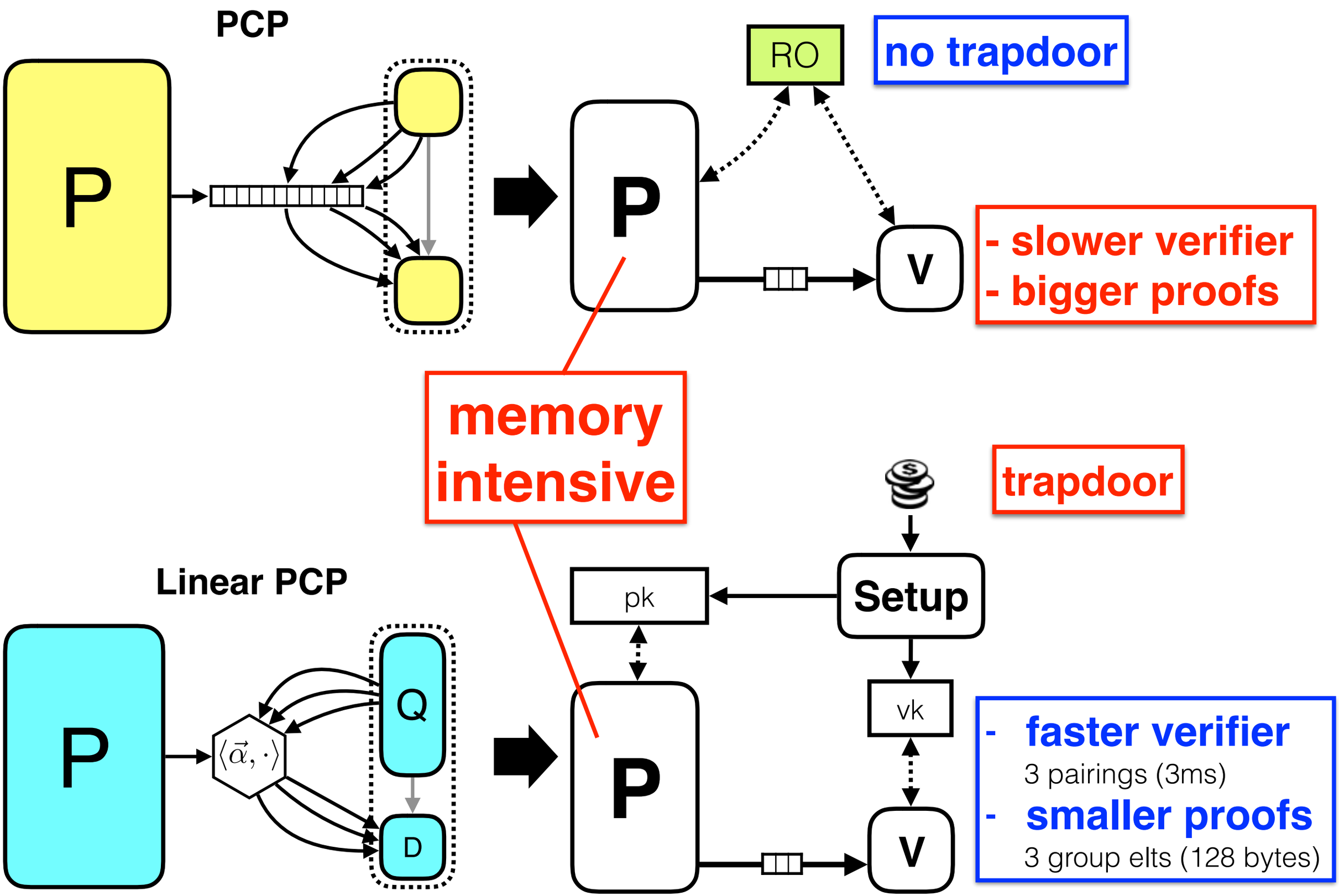
Which approach is better?



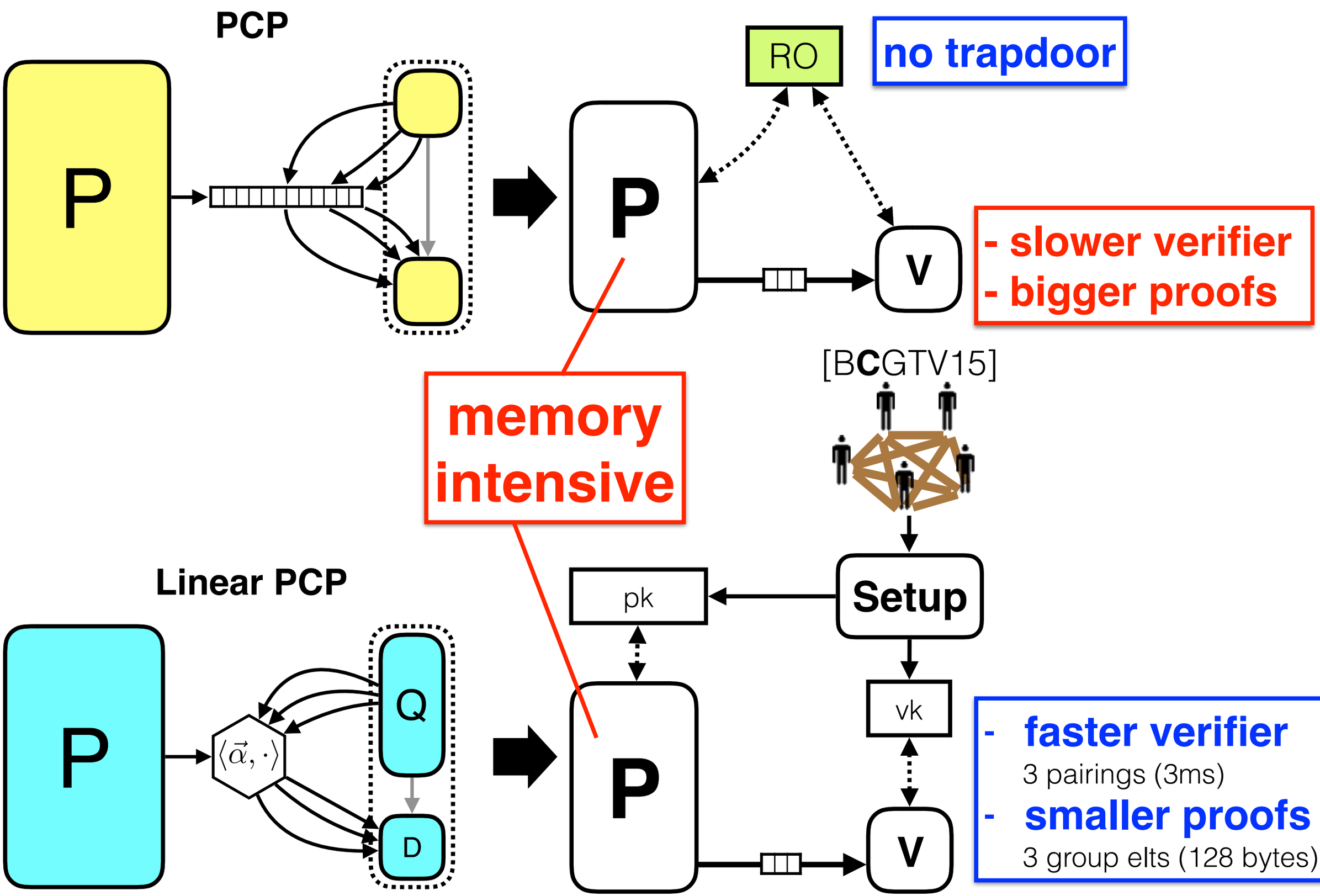
Which approach is better?



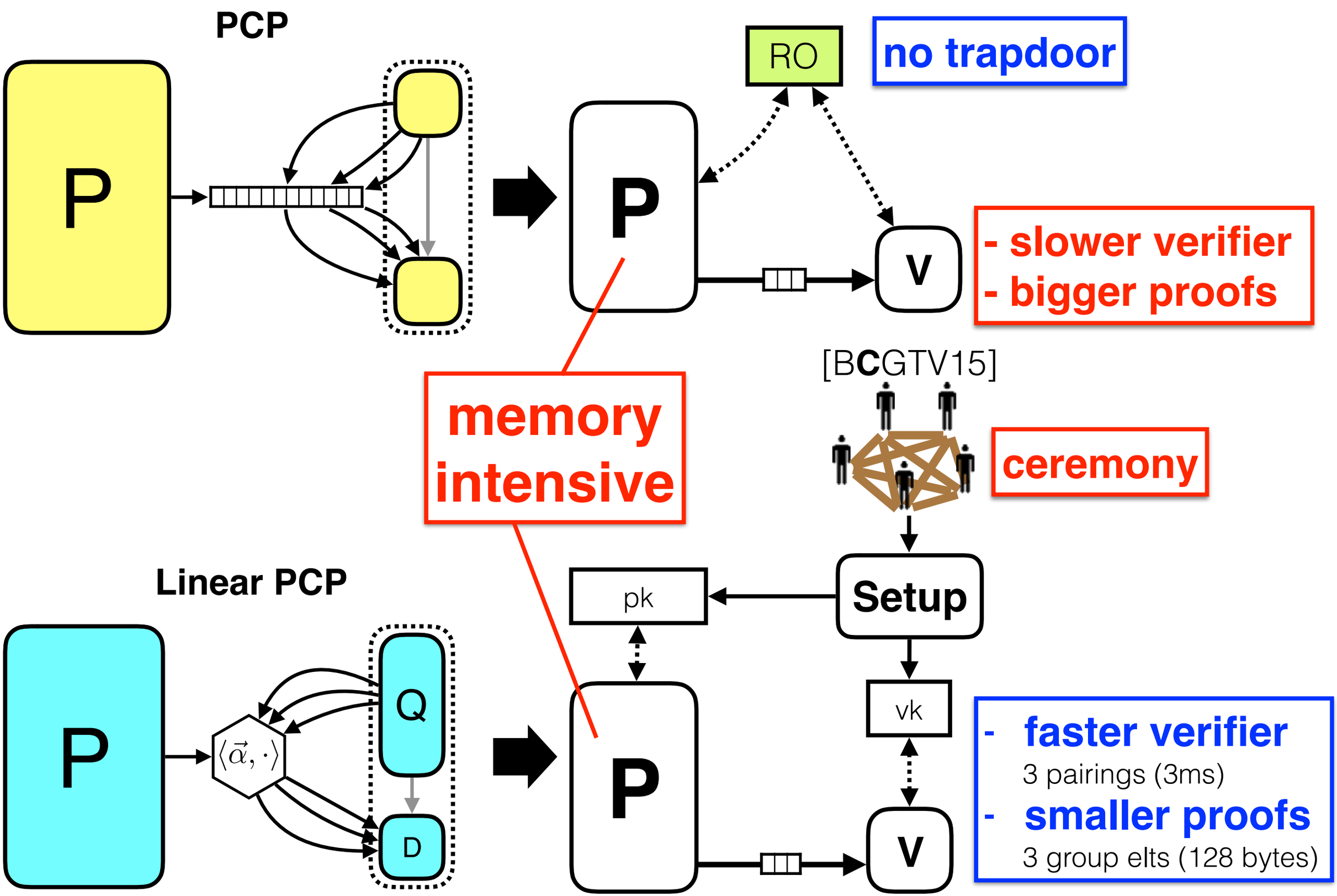
Which approach is better?



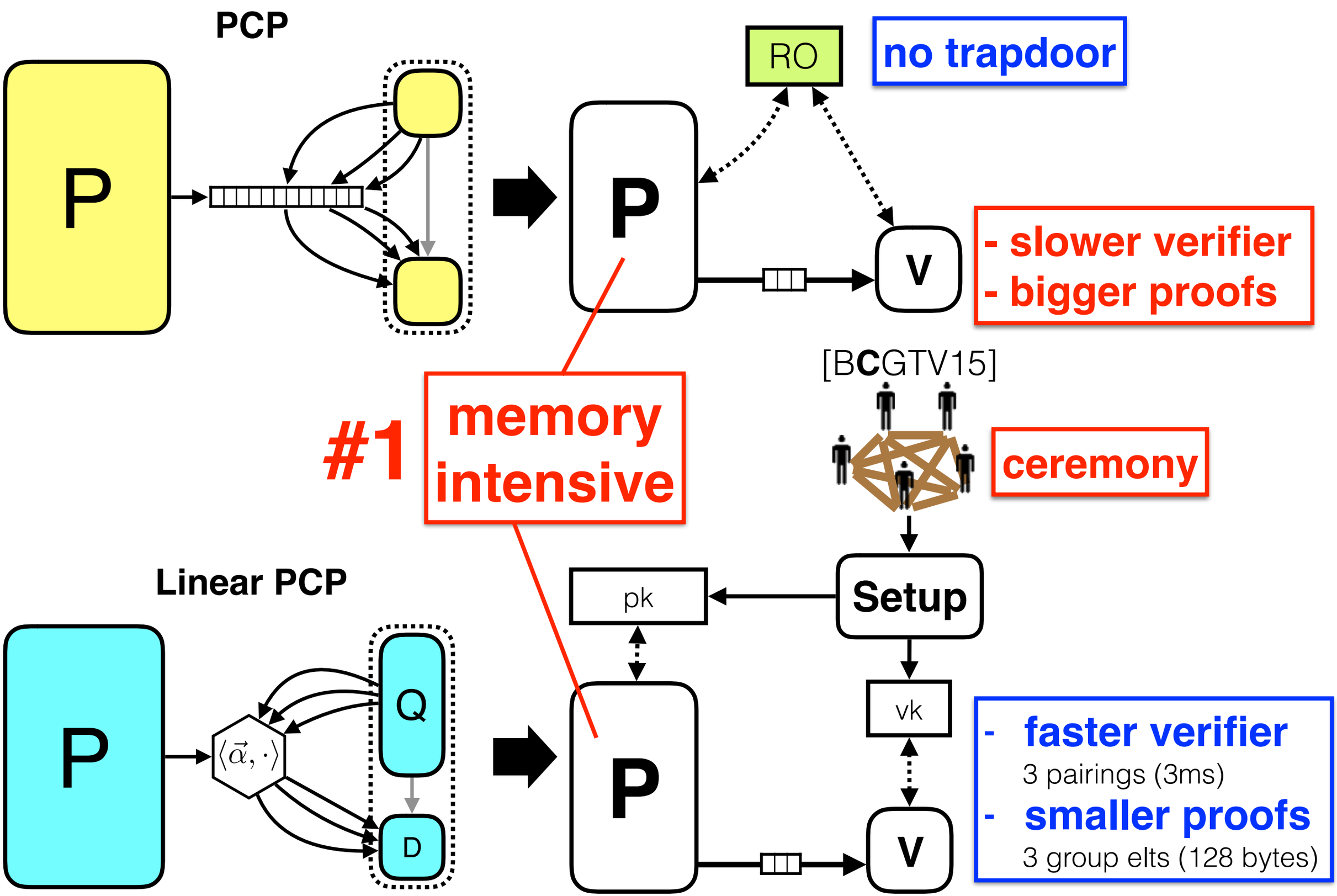
Which approach is better?



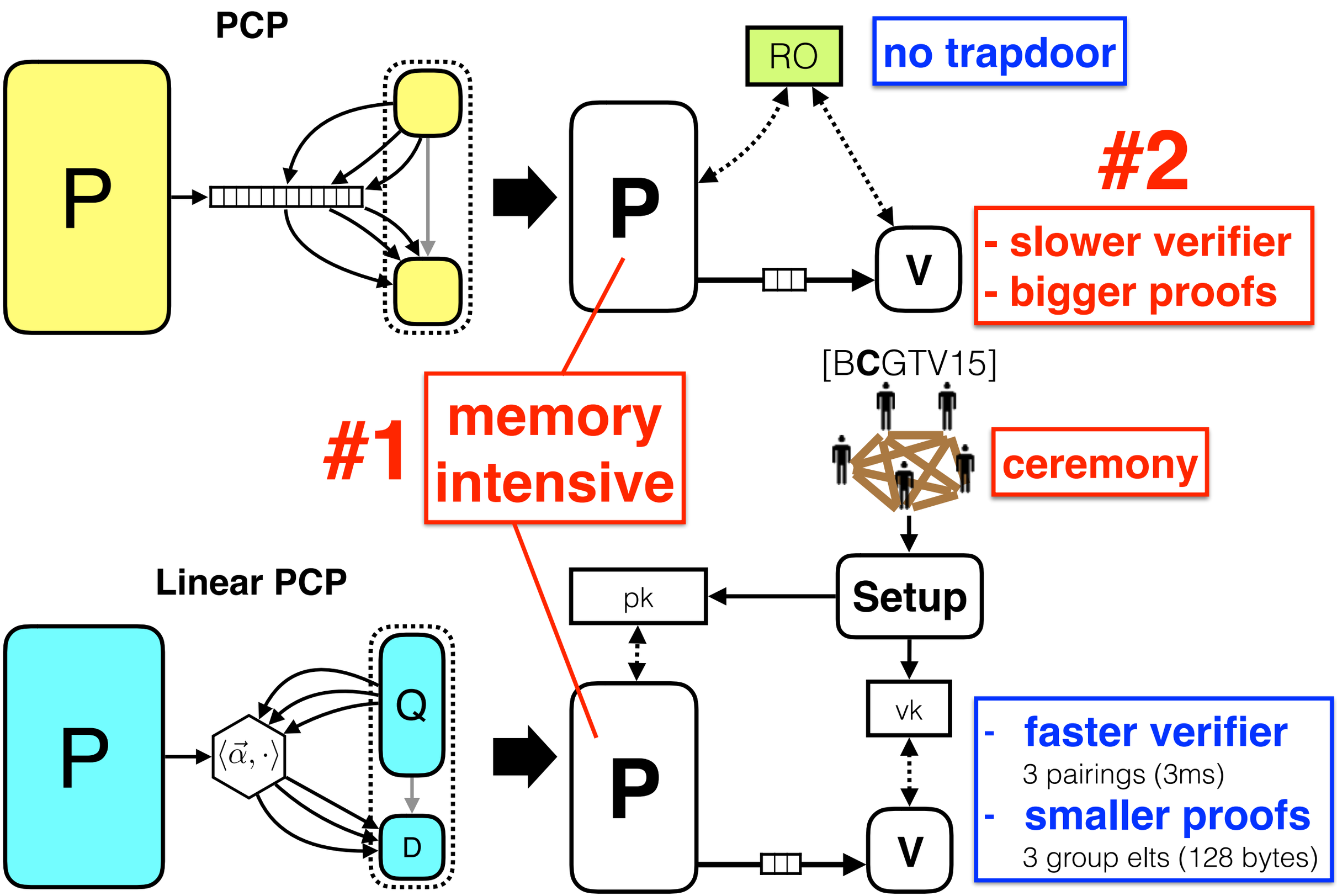
Which approach is better?



Which approach is better?



Which approach is better?



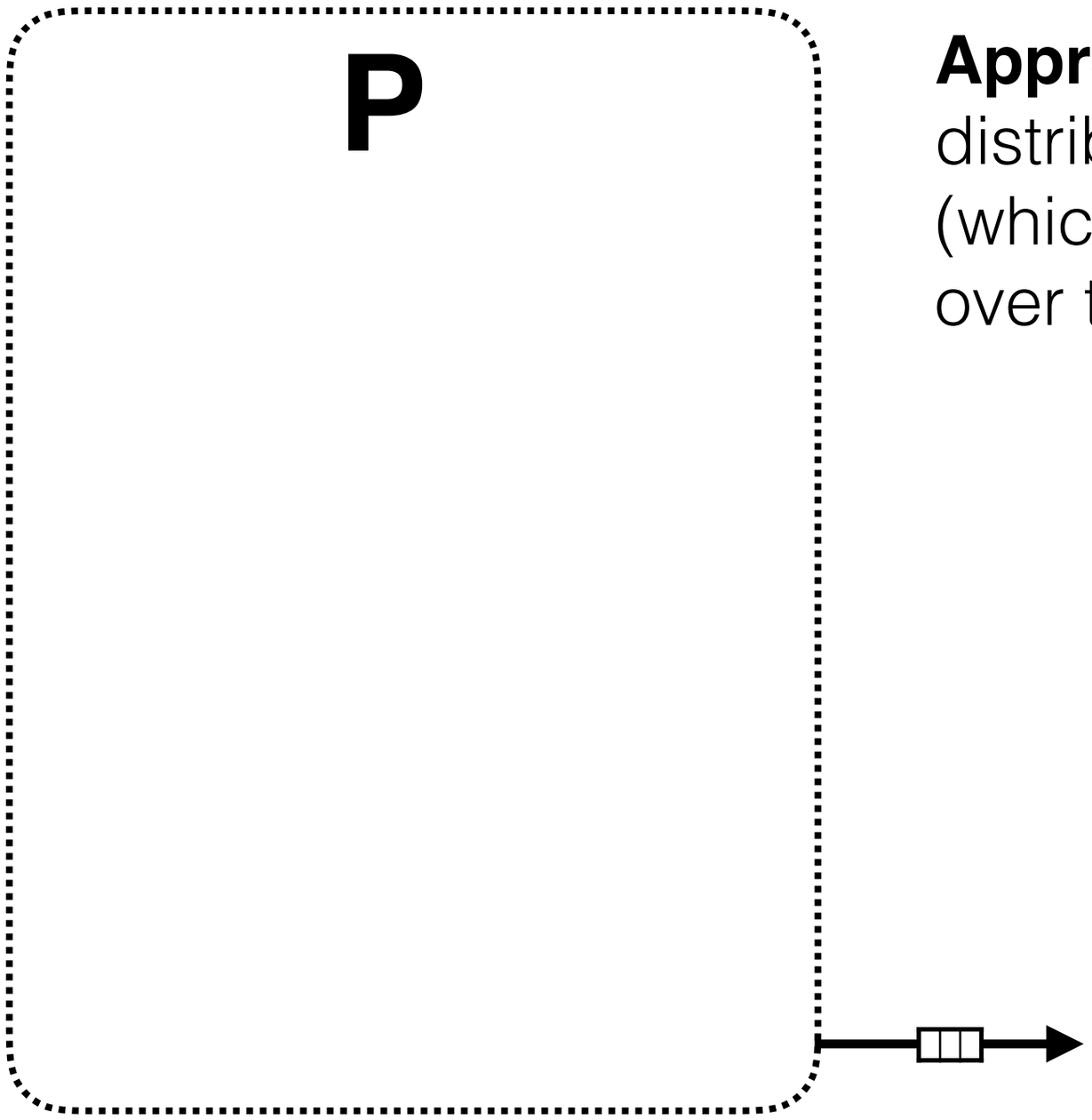
Tackling Problem #1

Distributed Proof Generation

Tackling Problem #1
Distributed Proof Generation

Approach:
distribute the prover algorithm
(which generates a small proof)
over the nodes in a computer cluster

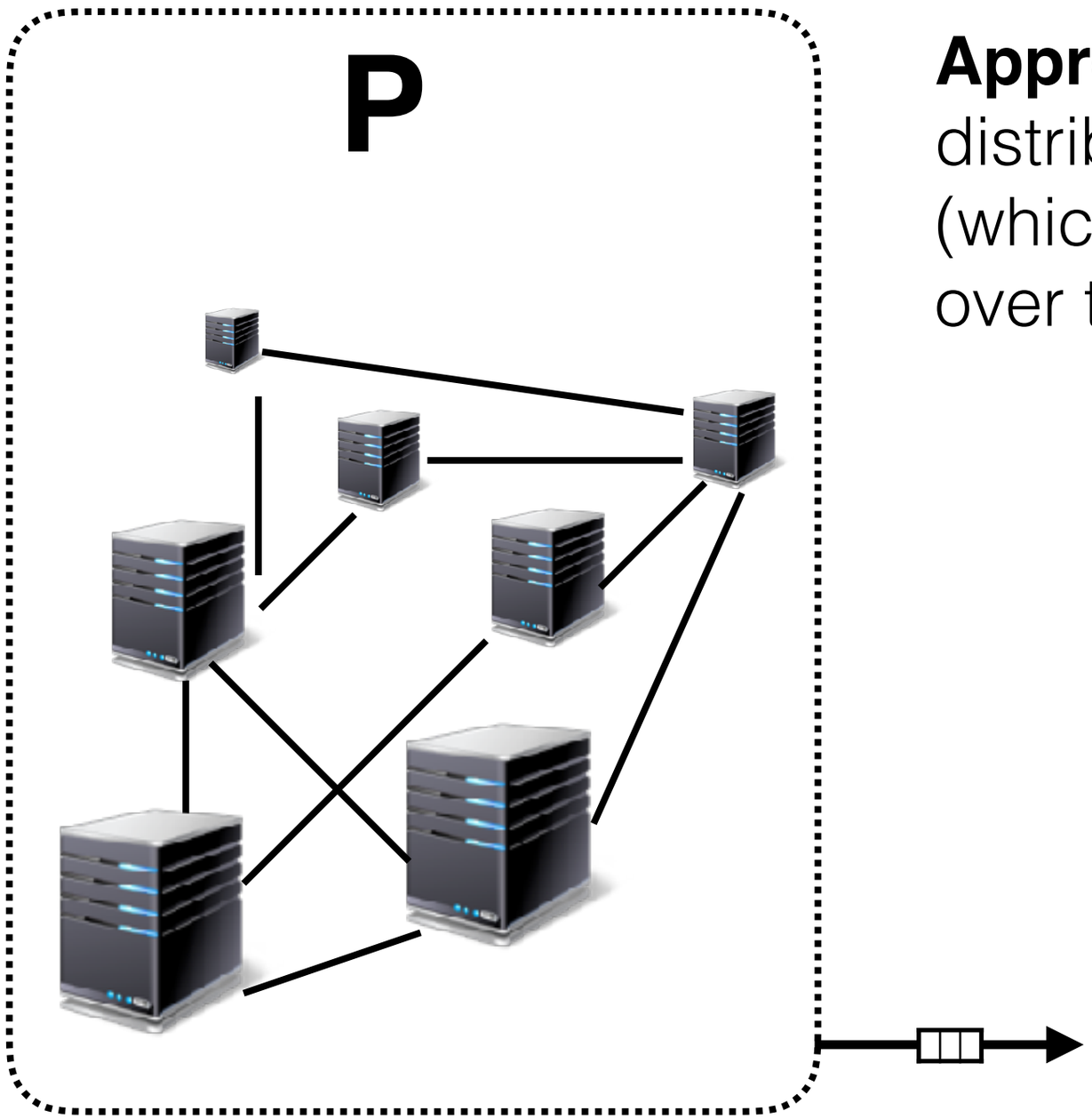
Tackling Problem #1
Distributed Proof Generation



Approach:
distribute the prover algorithm
(which generates a small proof)
over the nodes in a computer cluster

Tackling Problem #1

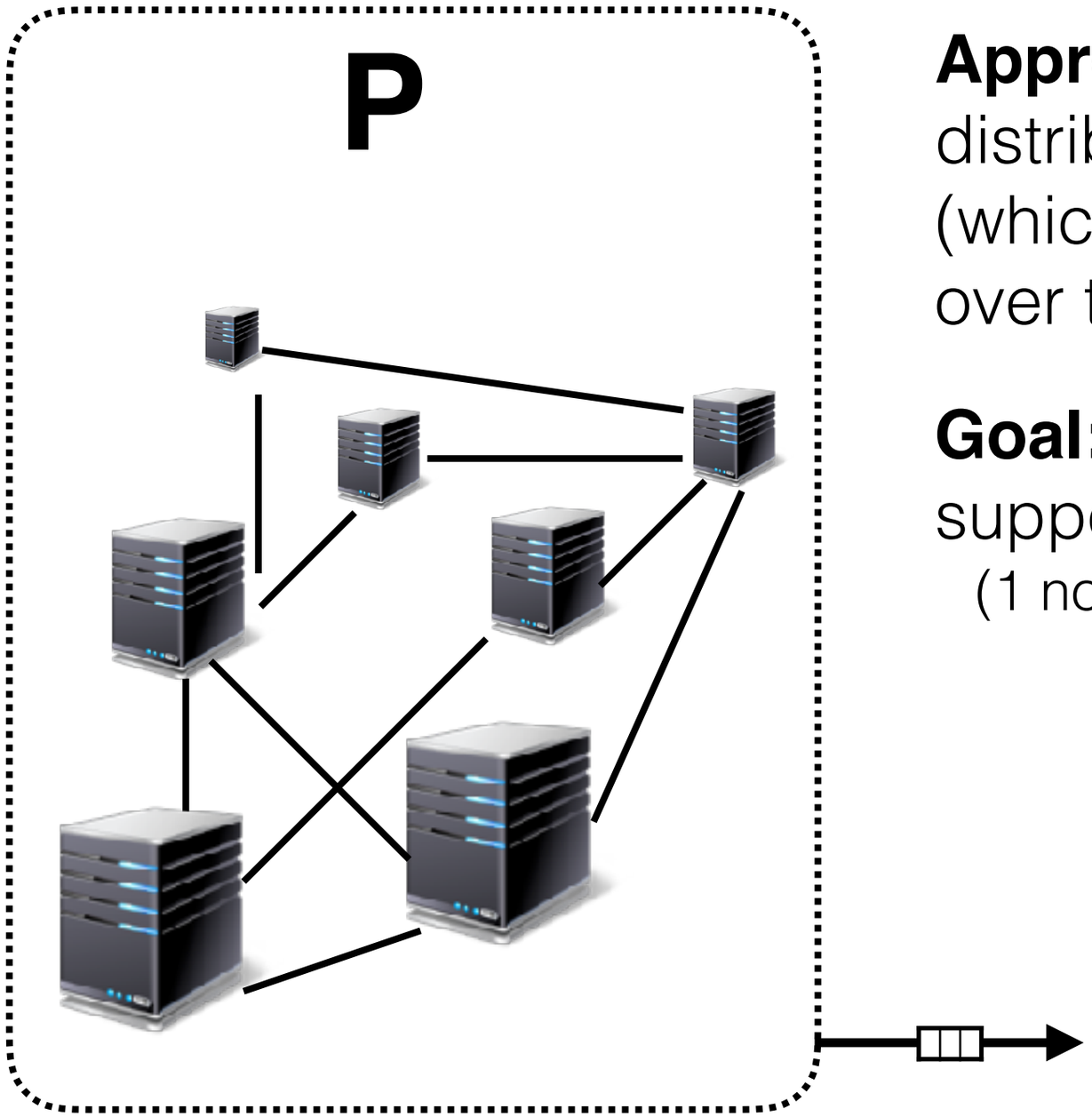
Distributed Proof Generation



Approach:
distribute the prover algorithm
(which generates a small proof)
over the nodes in a computer cluster

Tackling Problem #1

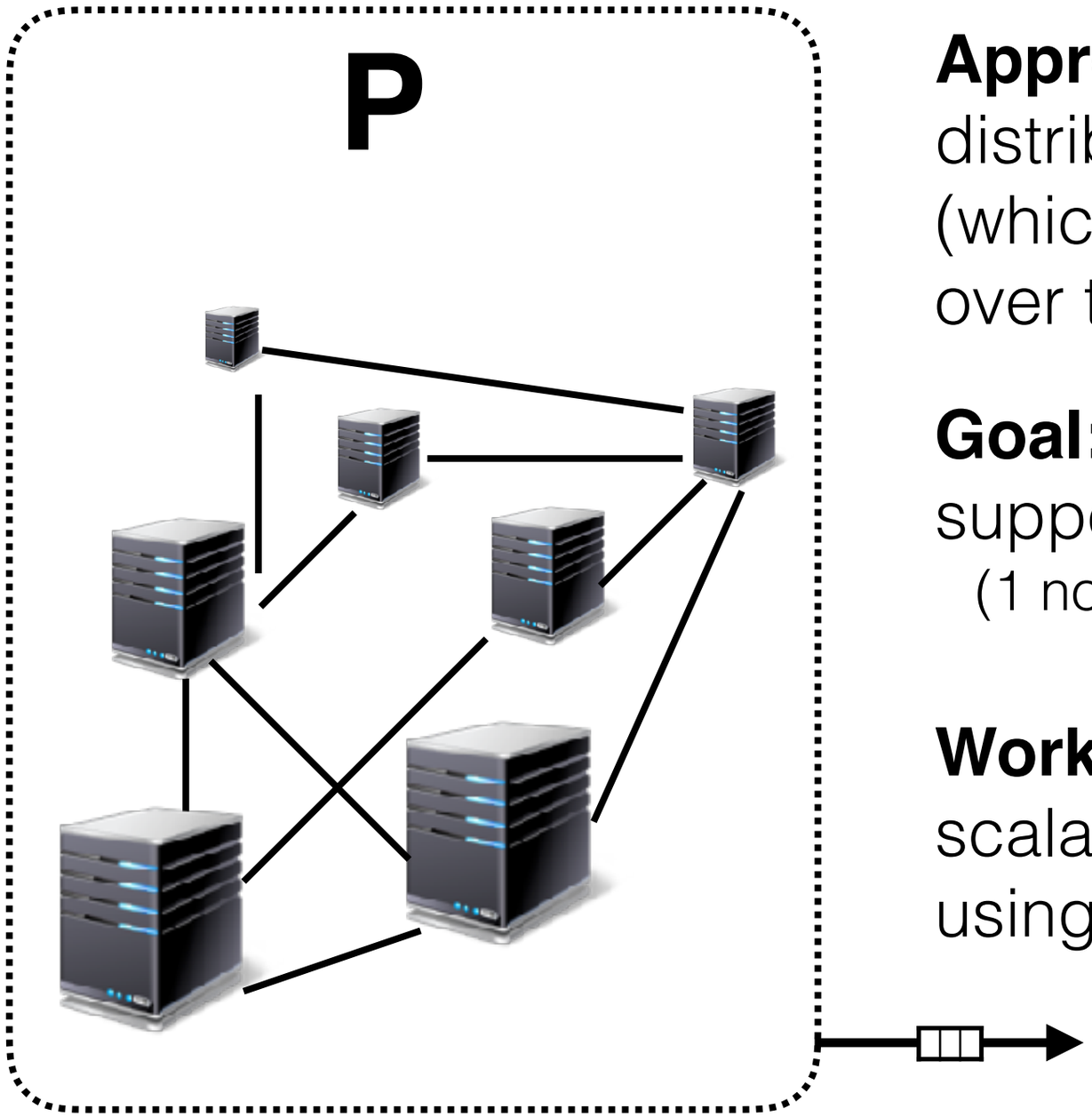
Distributed Proof Generation



Approach:
distribute the prover algorithm
(which generates a small proof)
over the nodes in a computer cluster

Goal:
support circuits of ~10^9/10^{10} gates
(1 node gets stuck at only ~10^7 gates)

Tackling Problem #1
Distributed Proof Generation

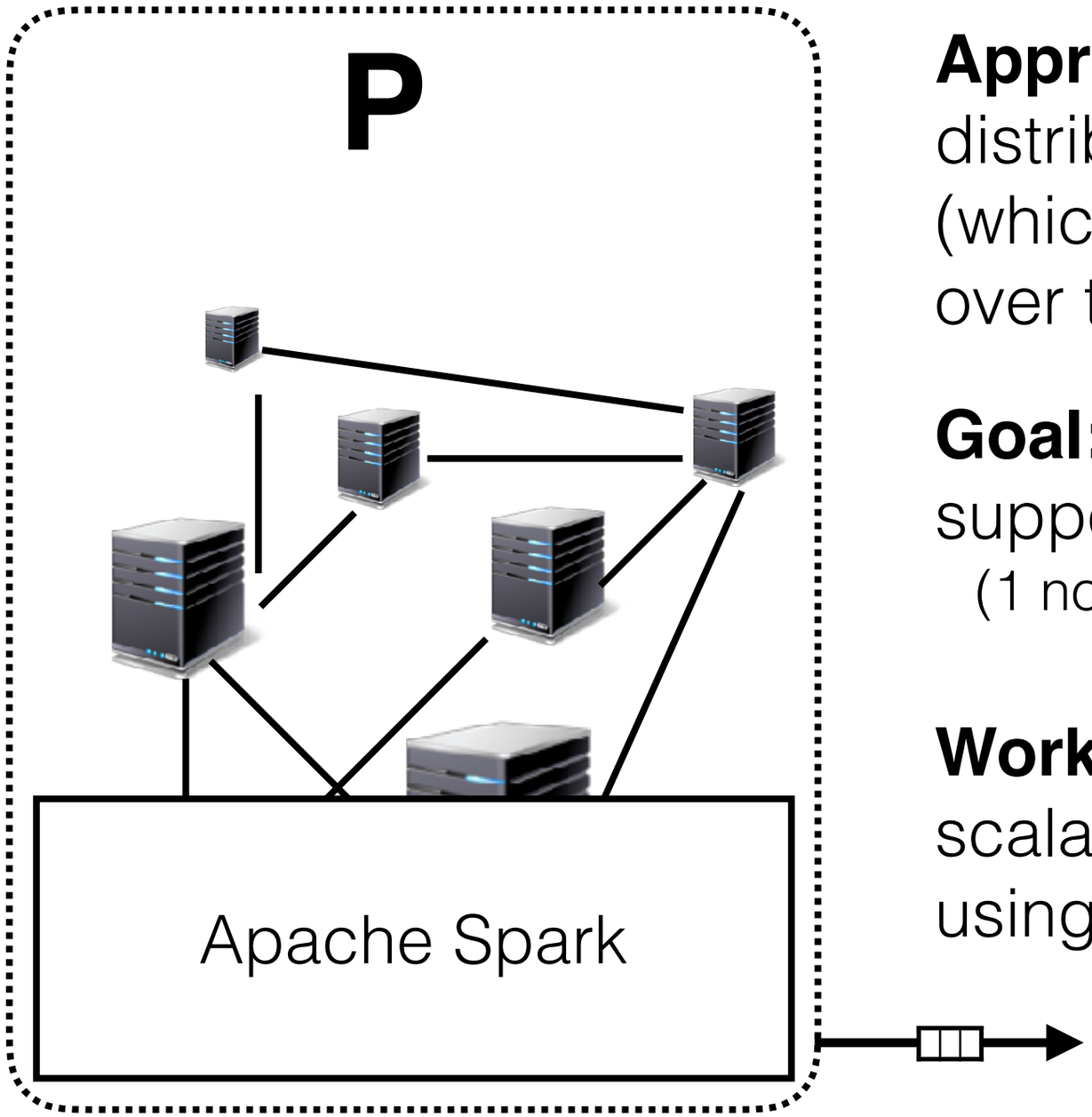


Approach:
distribute the prover algorithm
(which generates a small proof)
over the nodes in a computer cluster

Goal:
support circuits of ~10^9/10^{10} gates
(1 node gets stuck at only ~10^7 gates)

Work in progress:
scalable ZK-SNARK implementation
using Apache Spark

Tackling Problem #1
Distributed Proof Generation

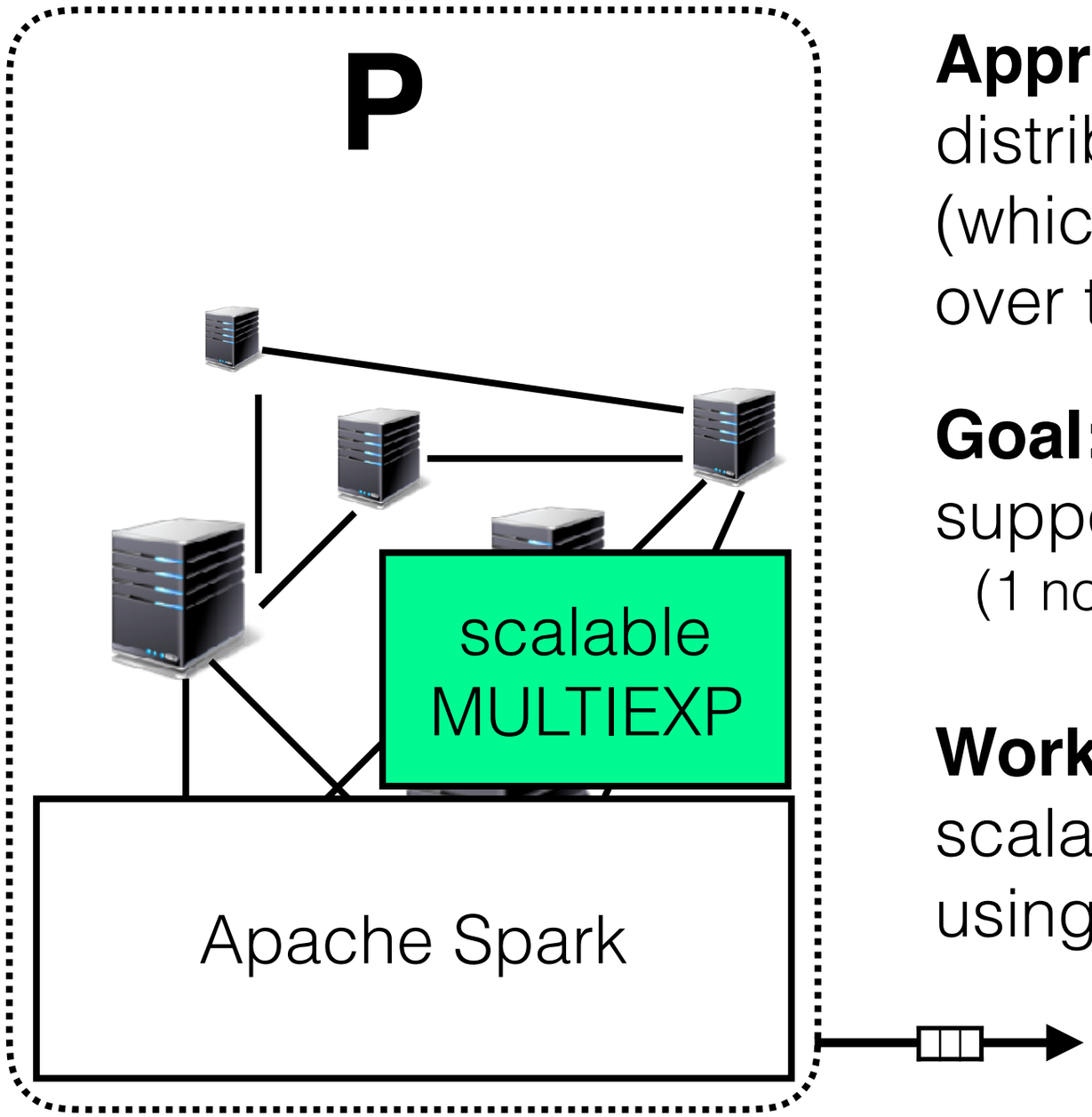


Approach:
distribute the prover algorithm
(which generates a small proof)
over the nodes in a computer cluster

Goal:
support circuits of ~10^9/10^{10} gates
(1 node gets stuck at only ~10^7 gates)

Work in progress:
scalable ZK-SNARK implementation
using Apache Spark

Tackling Problem #1
Distributed Proof Generation

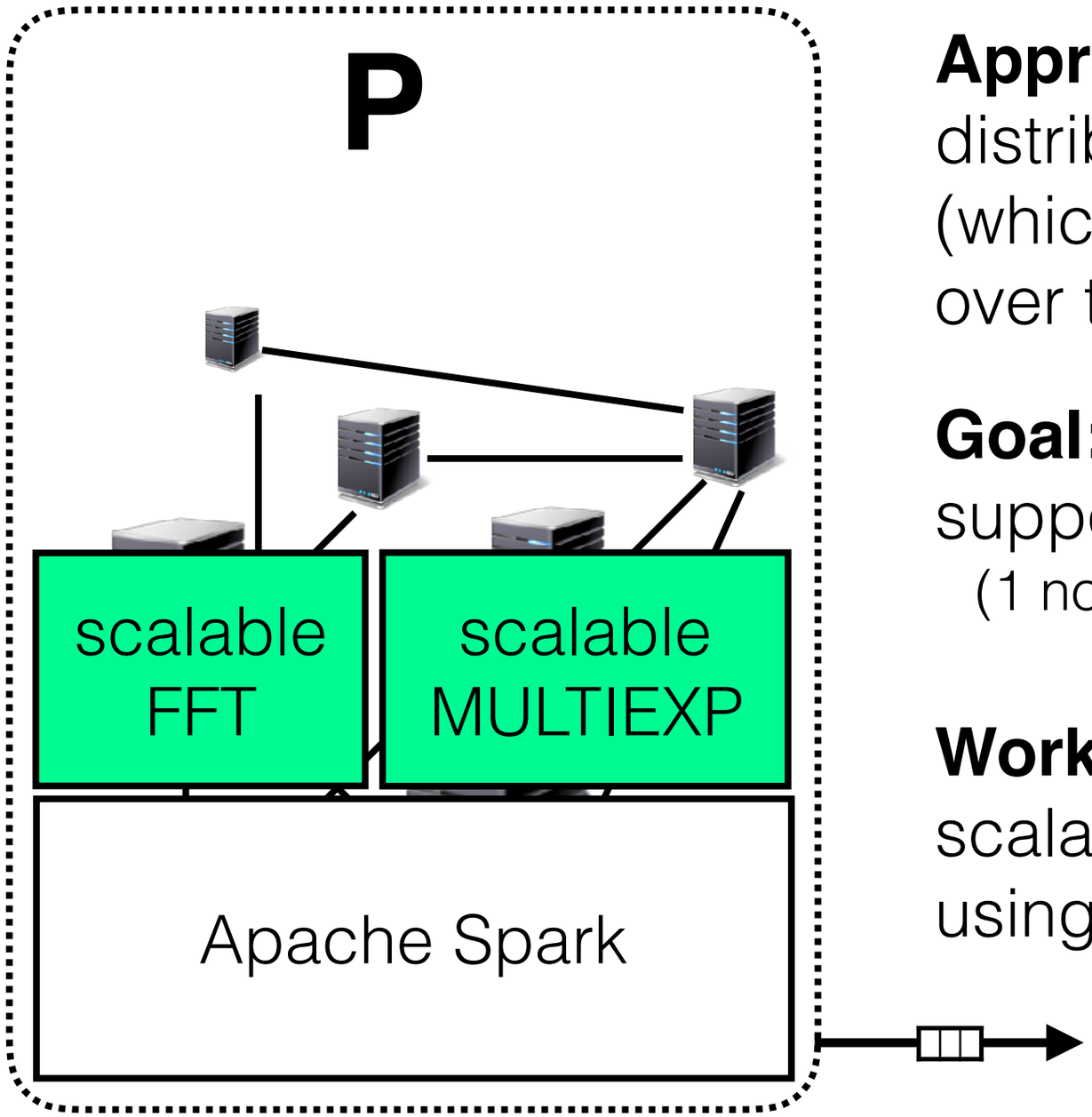


Approach:
distribute the prover algorithm
(which generates a small proof)
over the nodes in a computer cluster

Goal:
support circuits of ~10^9/10^{10} gates
(1 node gets stuck at only ~10^7 gates)

Work in progress:
scalable ZK-SNARK implementation
using Apache Spark

Tackling Problem #1
Distributed Proof Generation

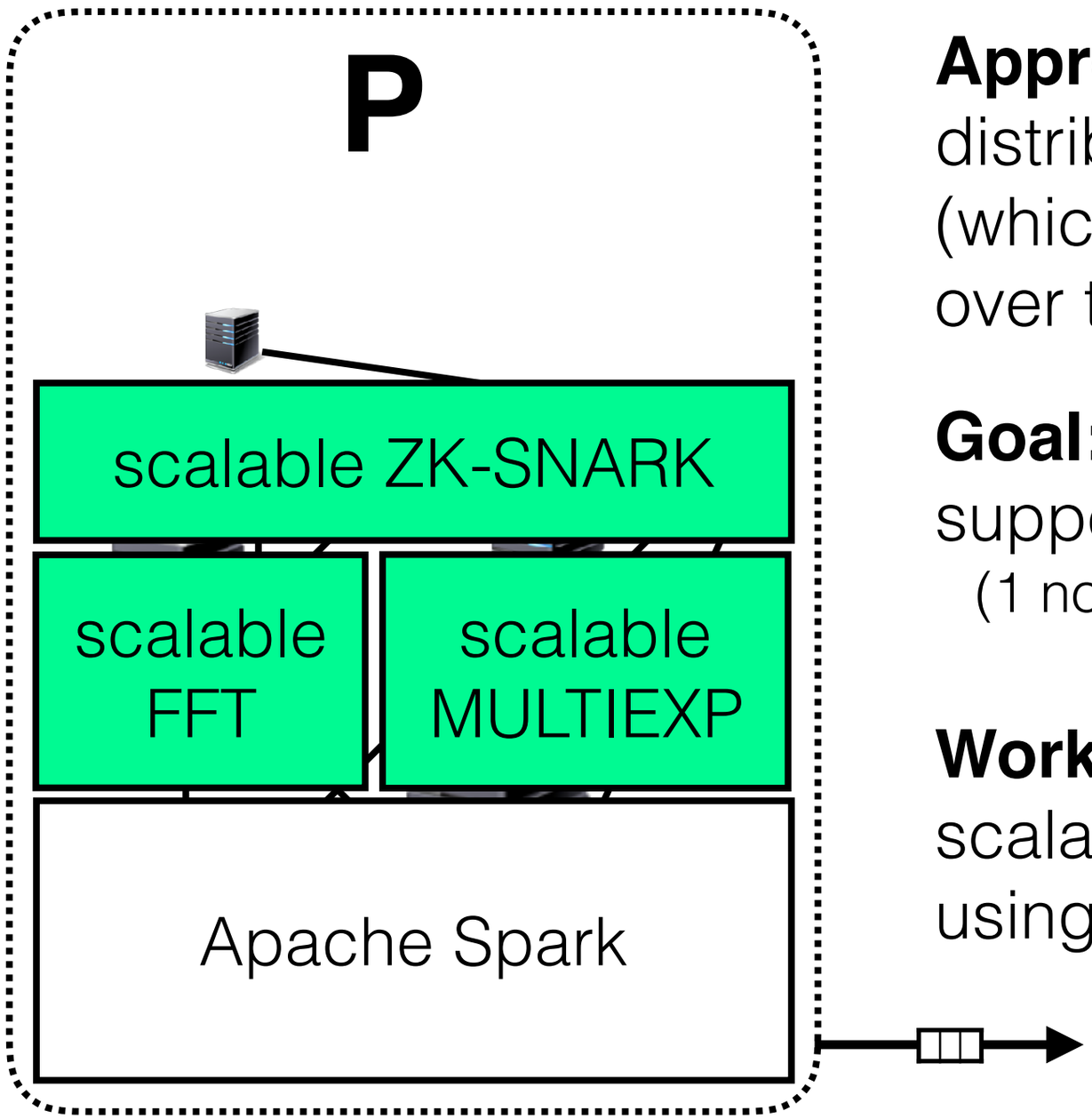


Approach:
distribute the prover algorithm (which generates a small proof) over the nodes in a computer cluster

Goal:
support circuits of ~10^9/10^{10} gates
(1 node gets stuck at only ~10^7 gates)

Work in progress:
scalable ZK-SNARK implementation using Apache Spark

Tackling Problem #1
Distributed Proof Generation



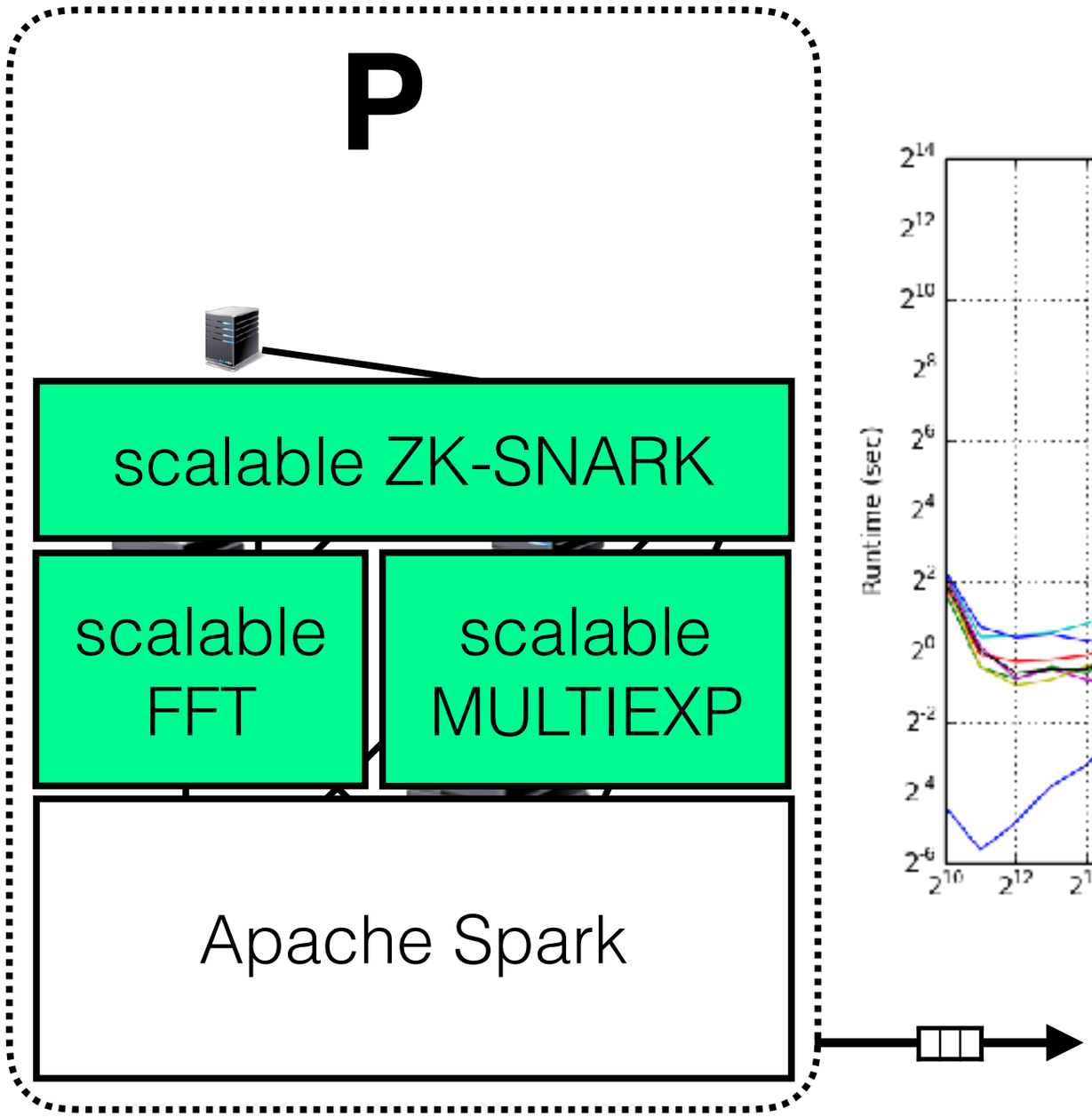
Approach:
distribute the prover algorithm (which generates a small proof) over the nodes in a computer cluster

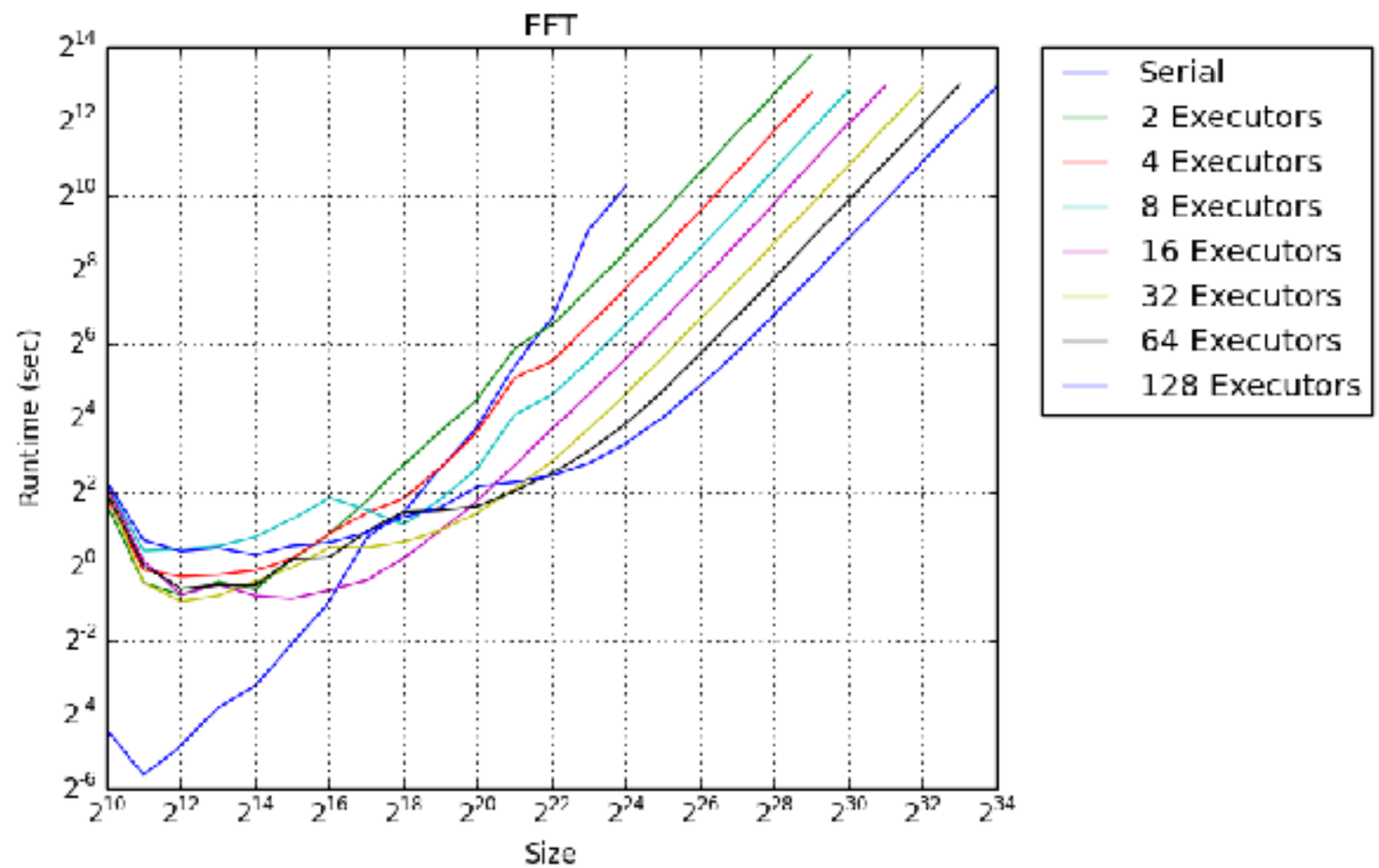
Goal:
support circuits of ~10^9/10^{10} gates
(1 node gets stuck at only ~10^7 gates)

Work in progress:
scalable ZK-SNARK implementation using Apache Spark

Tackling Problem #1

Distributed Proof Generation



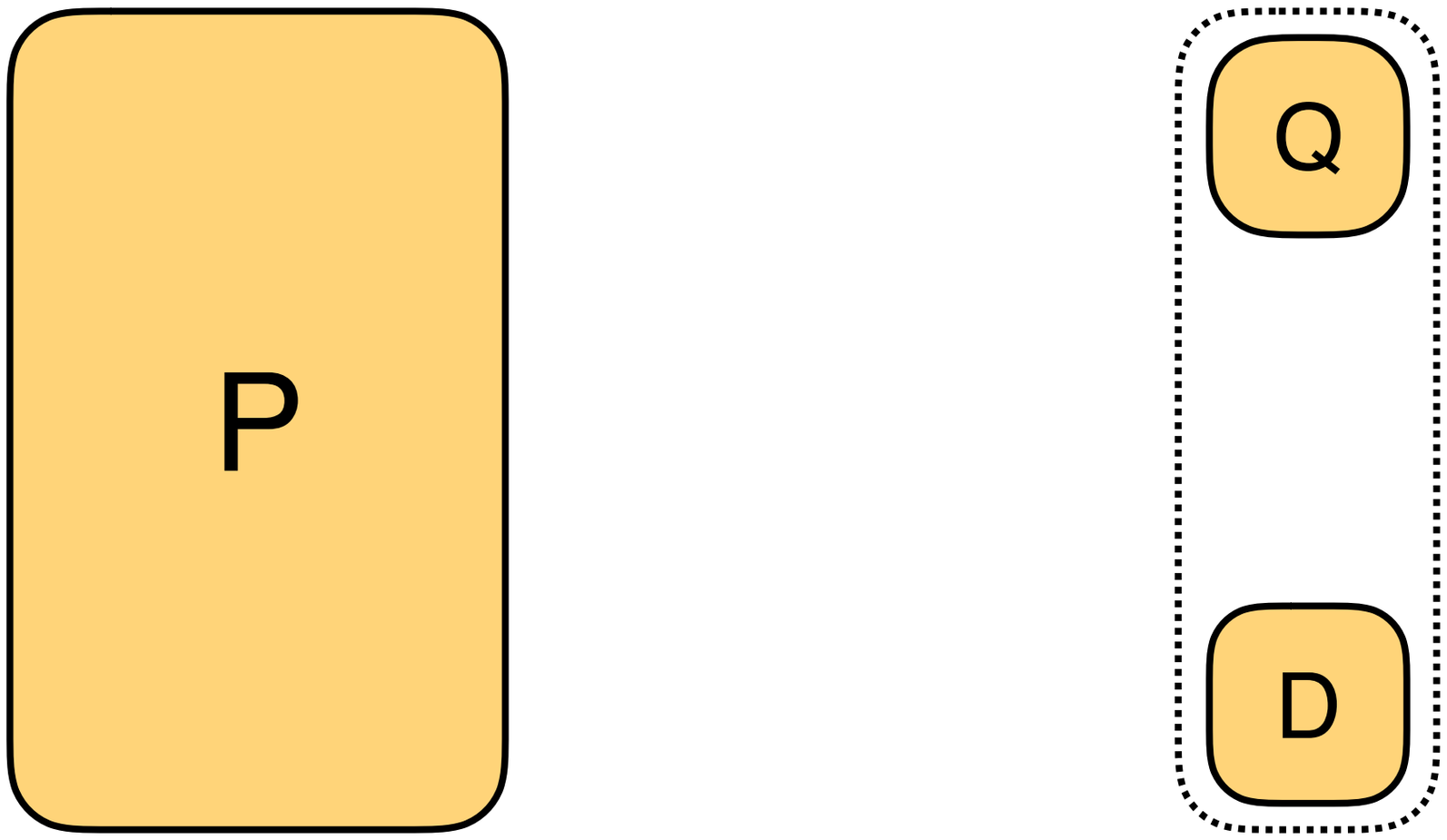


Tackling Problem #2:
Interactive Oracle Proofs

[B**C**S16][RRR16]

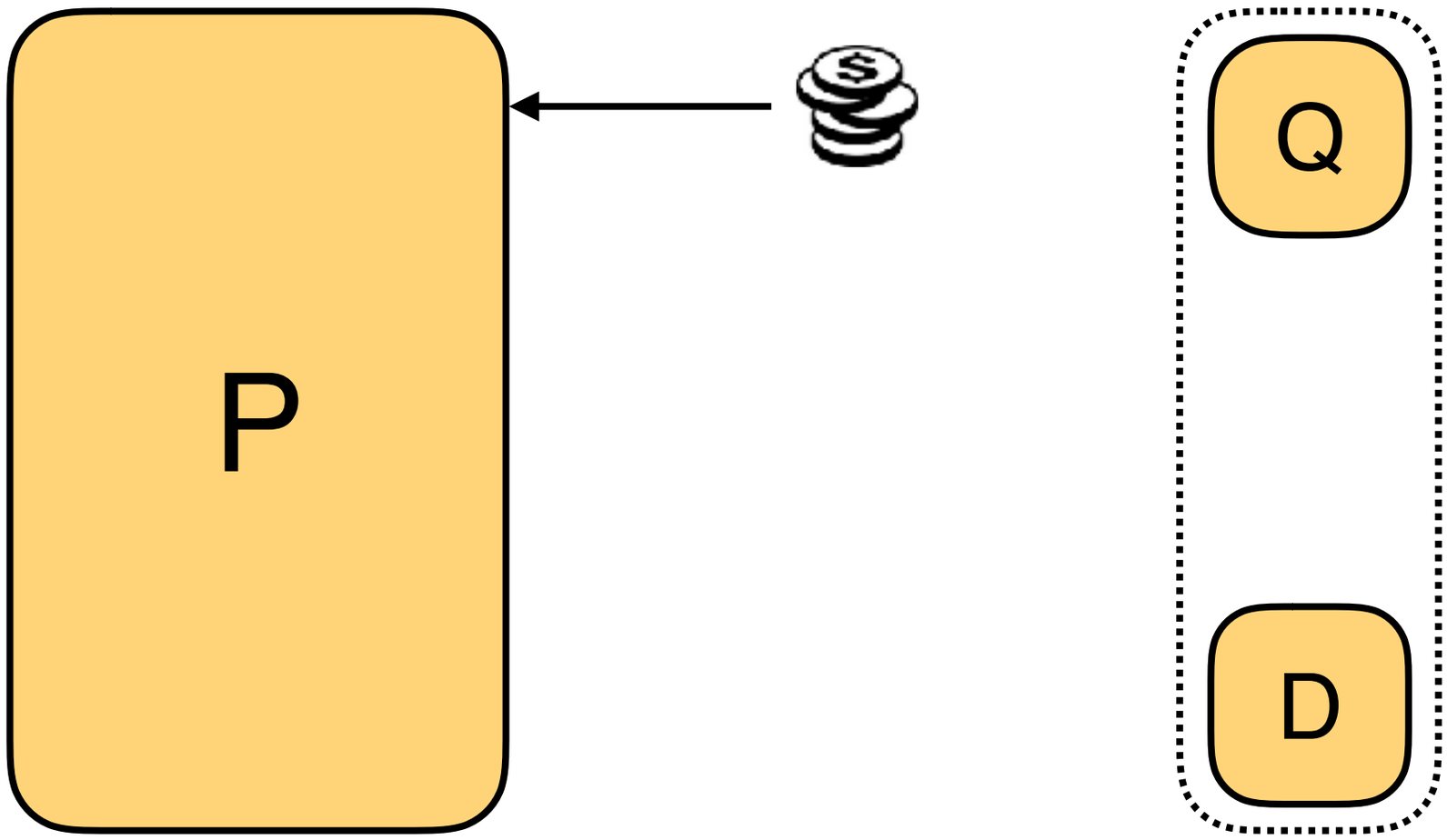
Tackling Problem #2:
Interactive Oracle Proofs

[B**C**S16][RRR16]



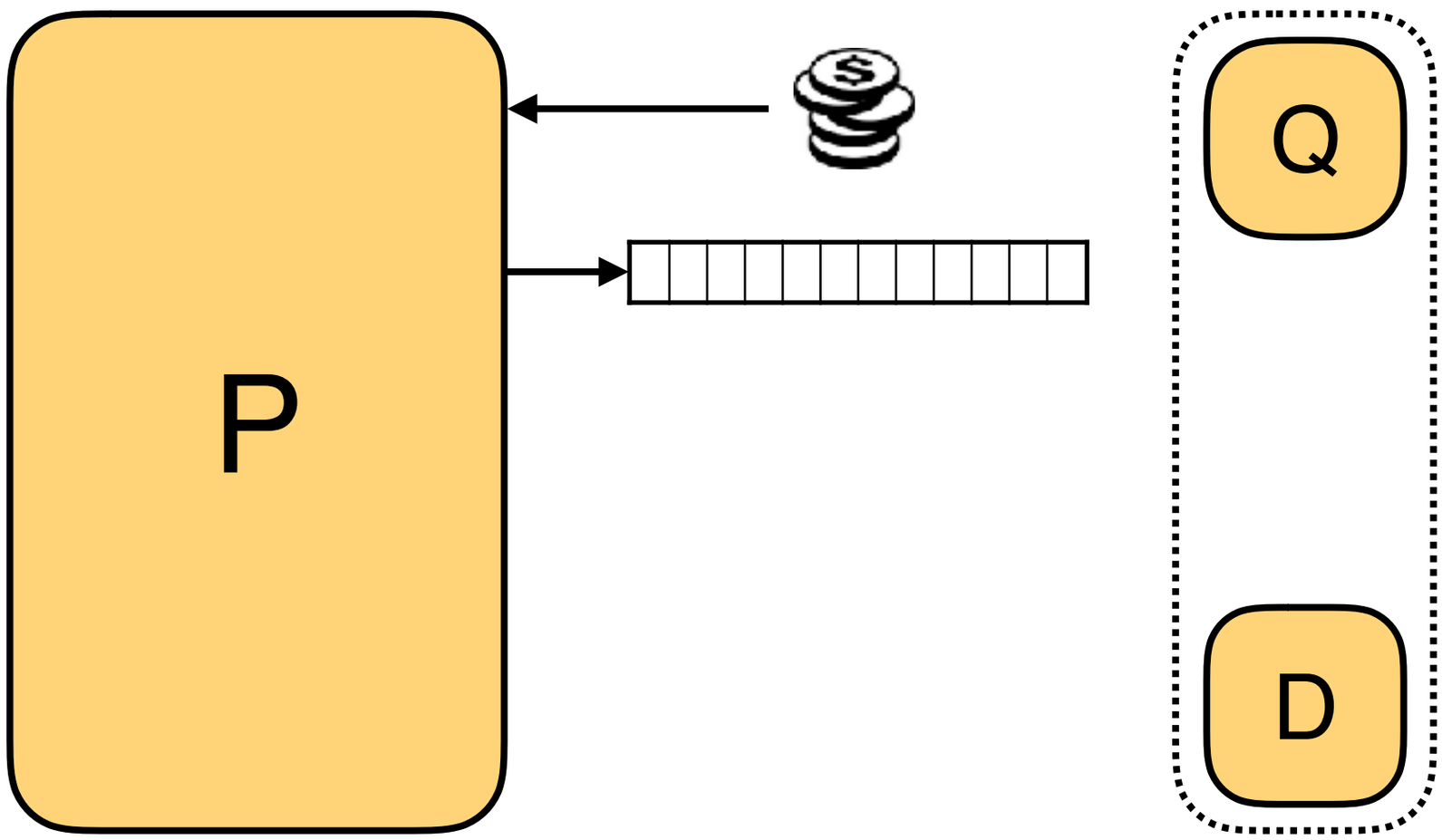
Tackling Problem #2:
Interactive Oracle Proofs

[B**C**S16][RRR16]



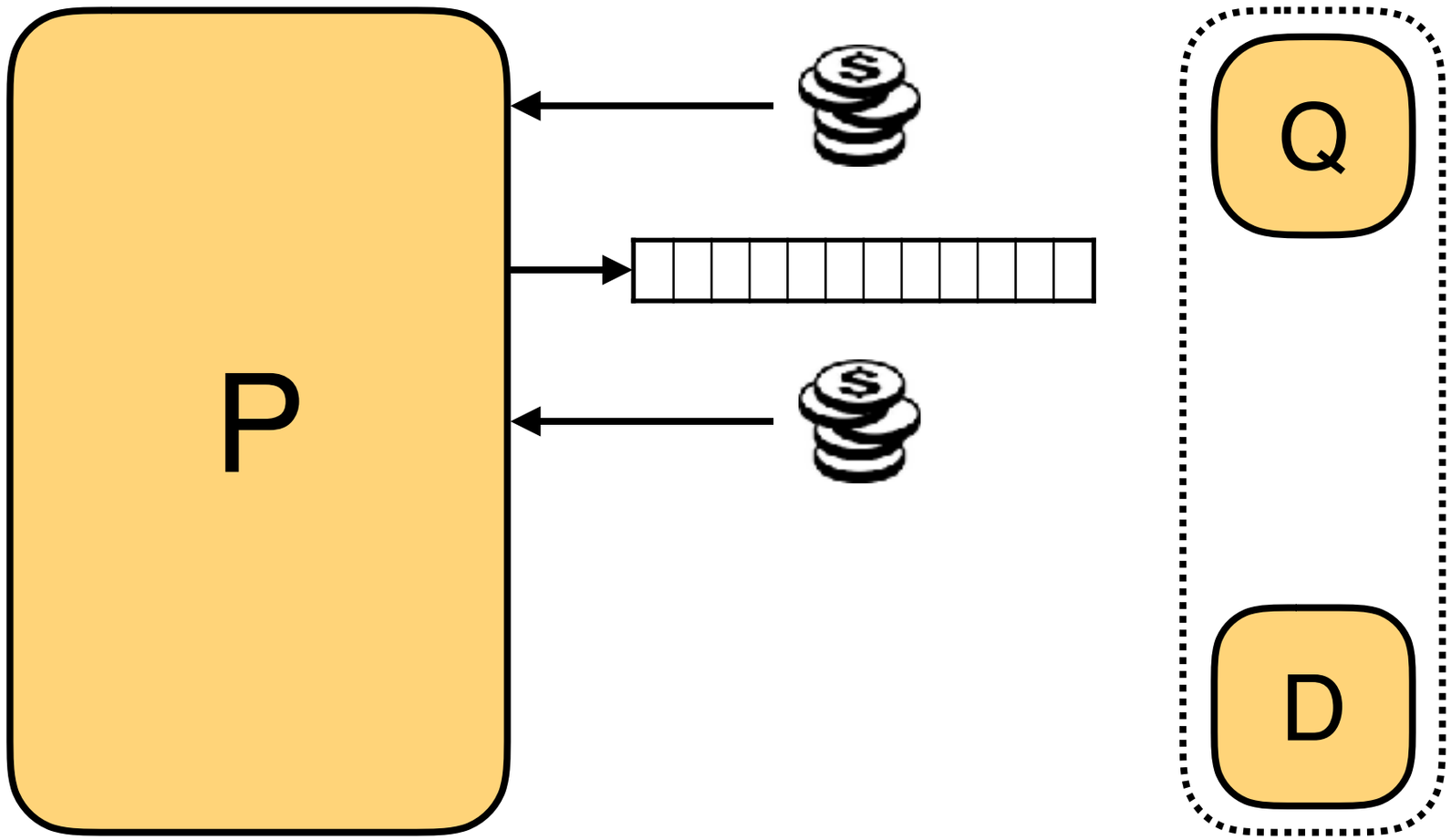
Tackling Problem #2:
Interactive Oracle Proofs

[B**C**S16][RRR16]



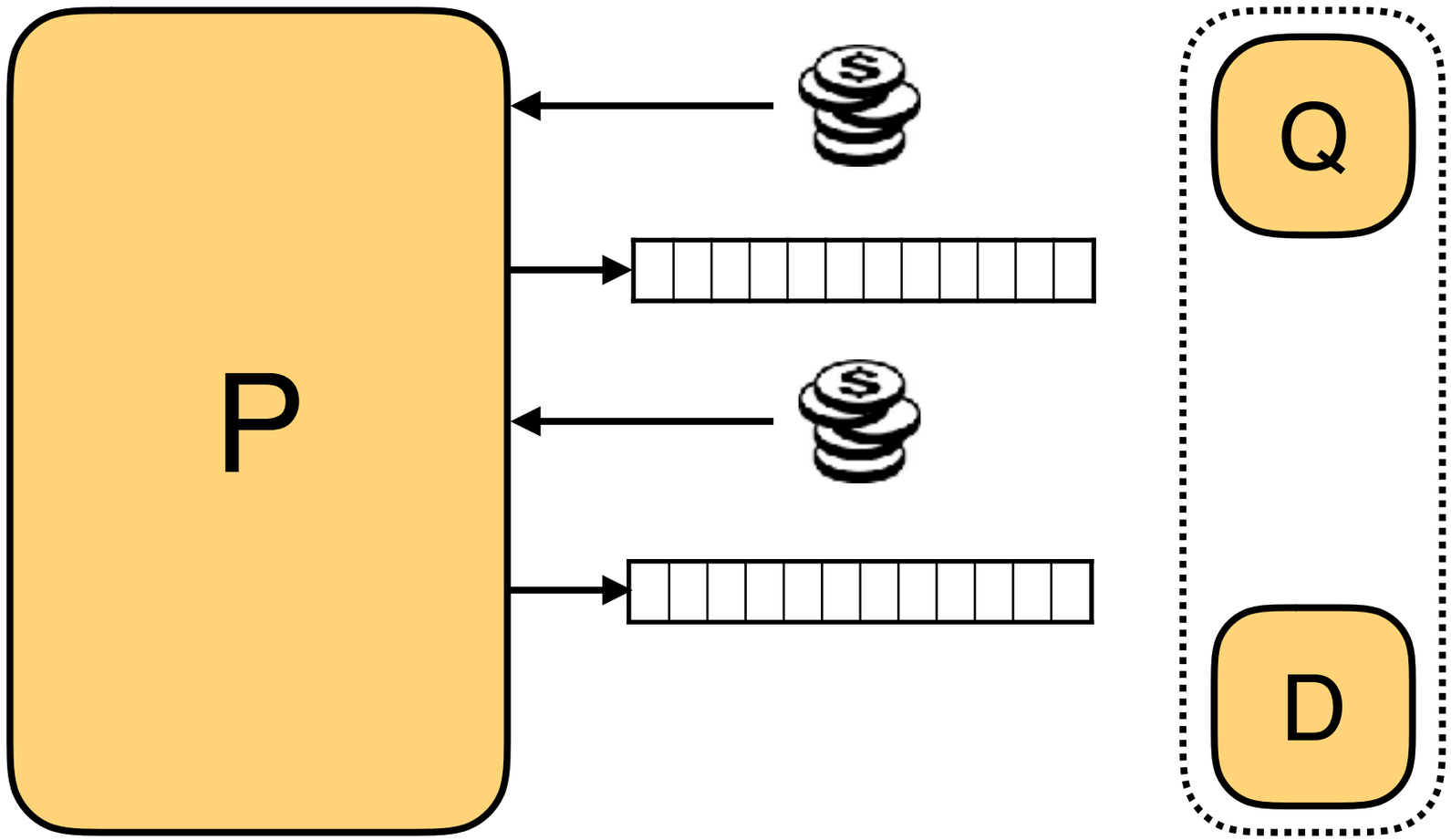
Tackling Problem #2:
Interactive Oracle Proofs

[B**C**S16][RRR16]



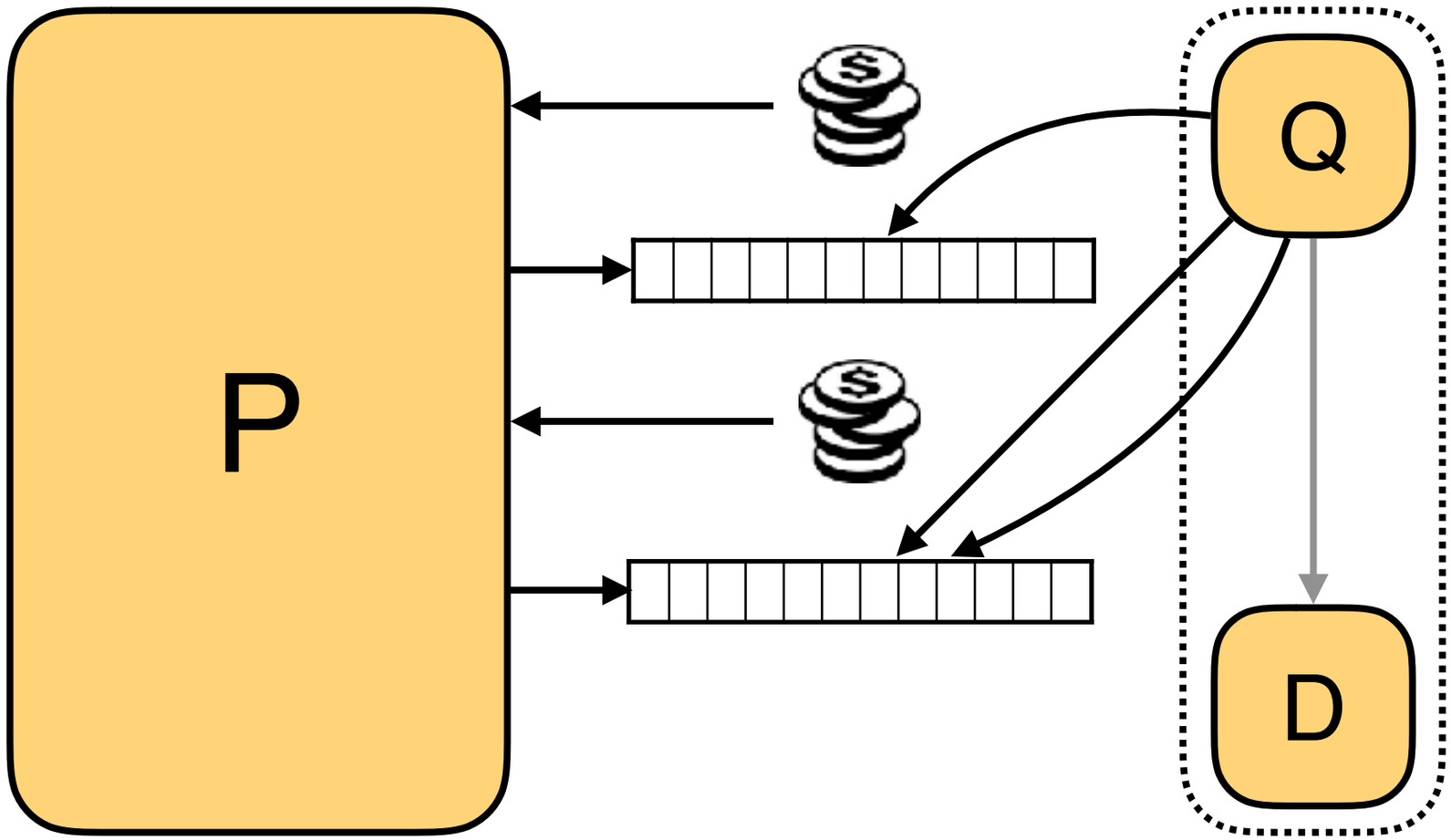
Tackling Problem #2:
Interactive Oracle Proofs

[B**C**S16][RRR16]



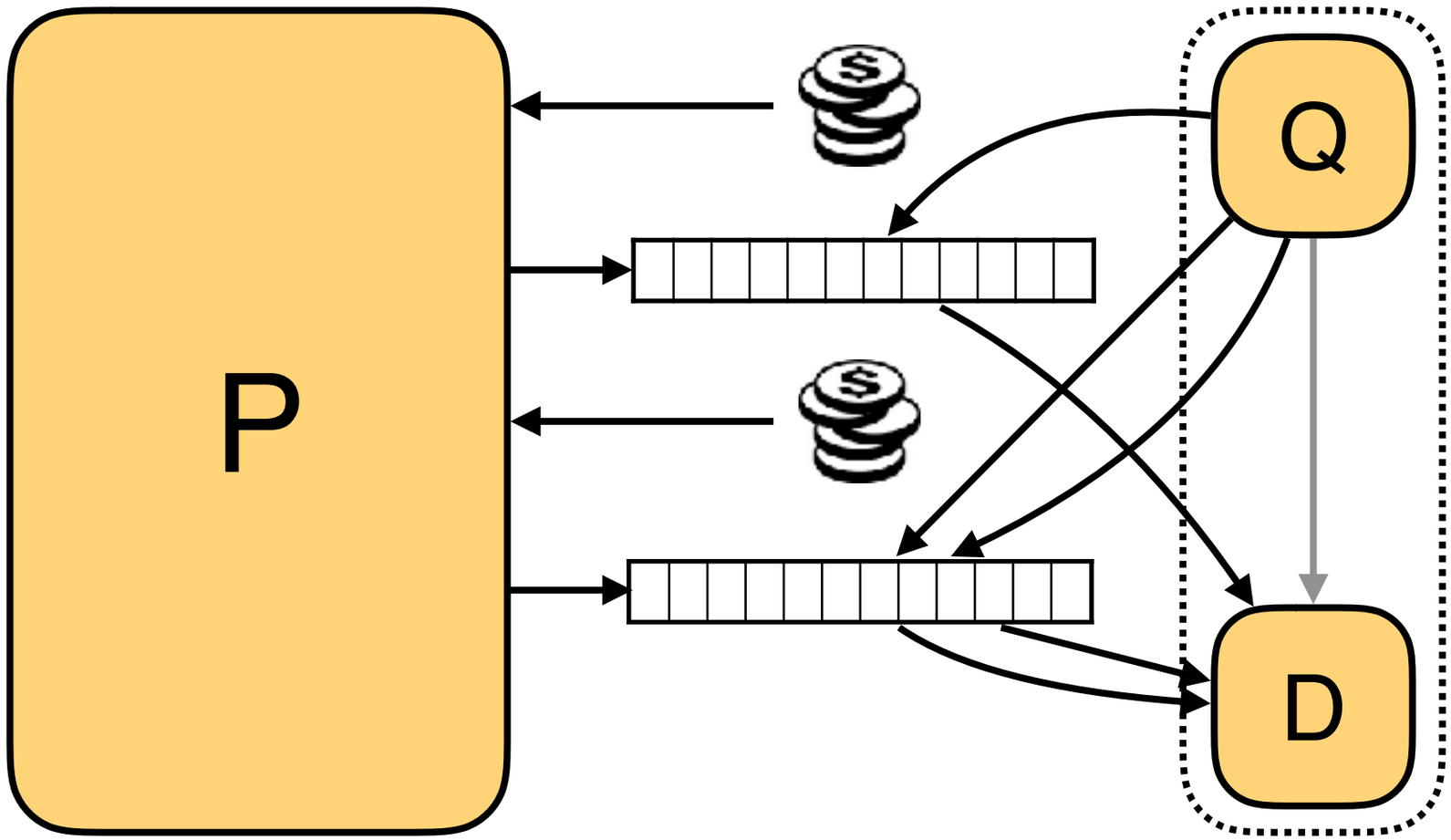
Tackling Problem #2:
Interactive Oracle Proofs

[B**C**S16][RRR16]



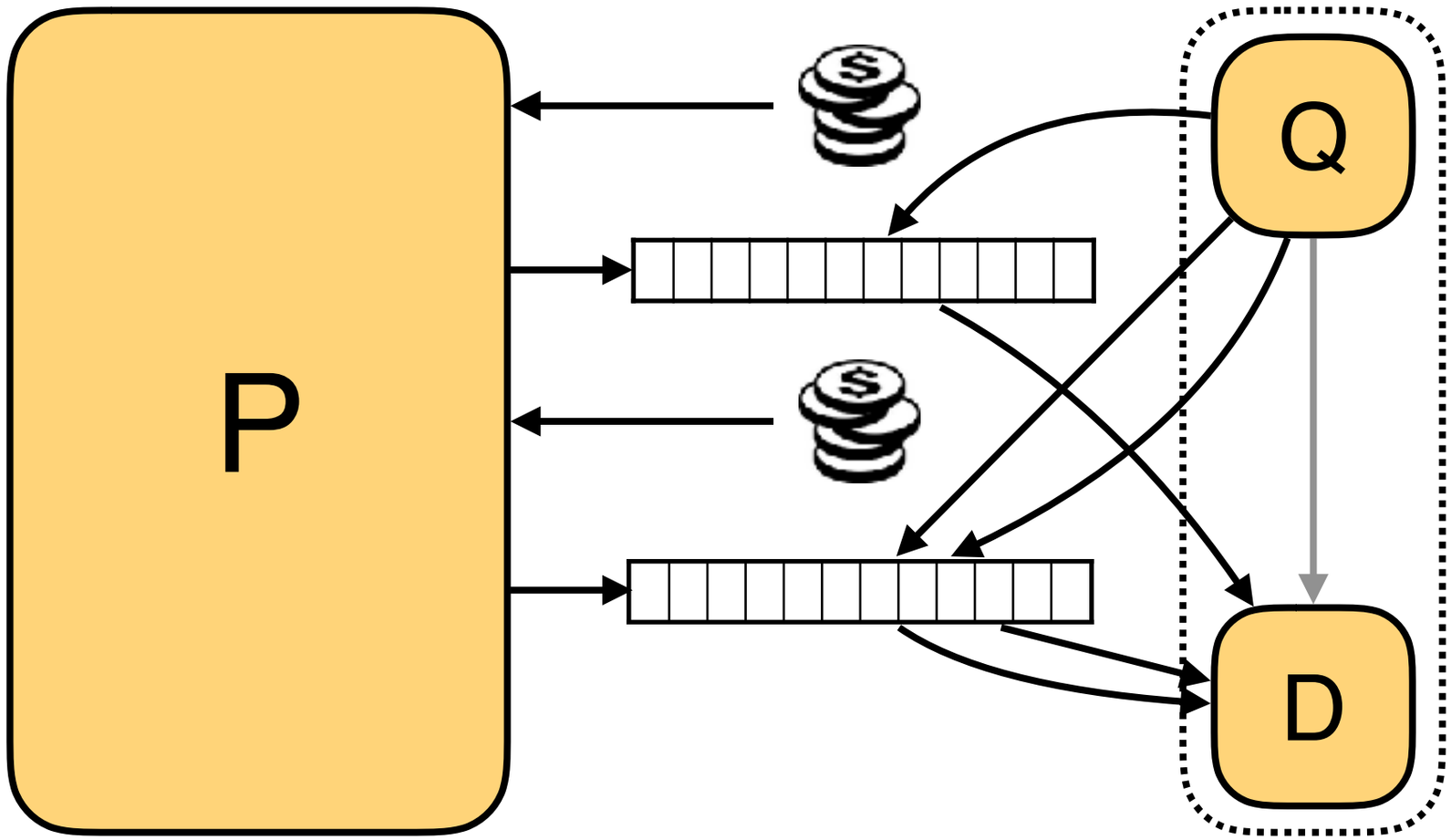
Tackling Problem #2:
Interactive Oracle Proofs

[B**C**S16][RRR16]



Tackling Problem #2:
Interactive Oracle Proofs

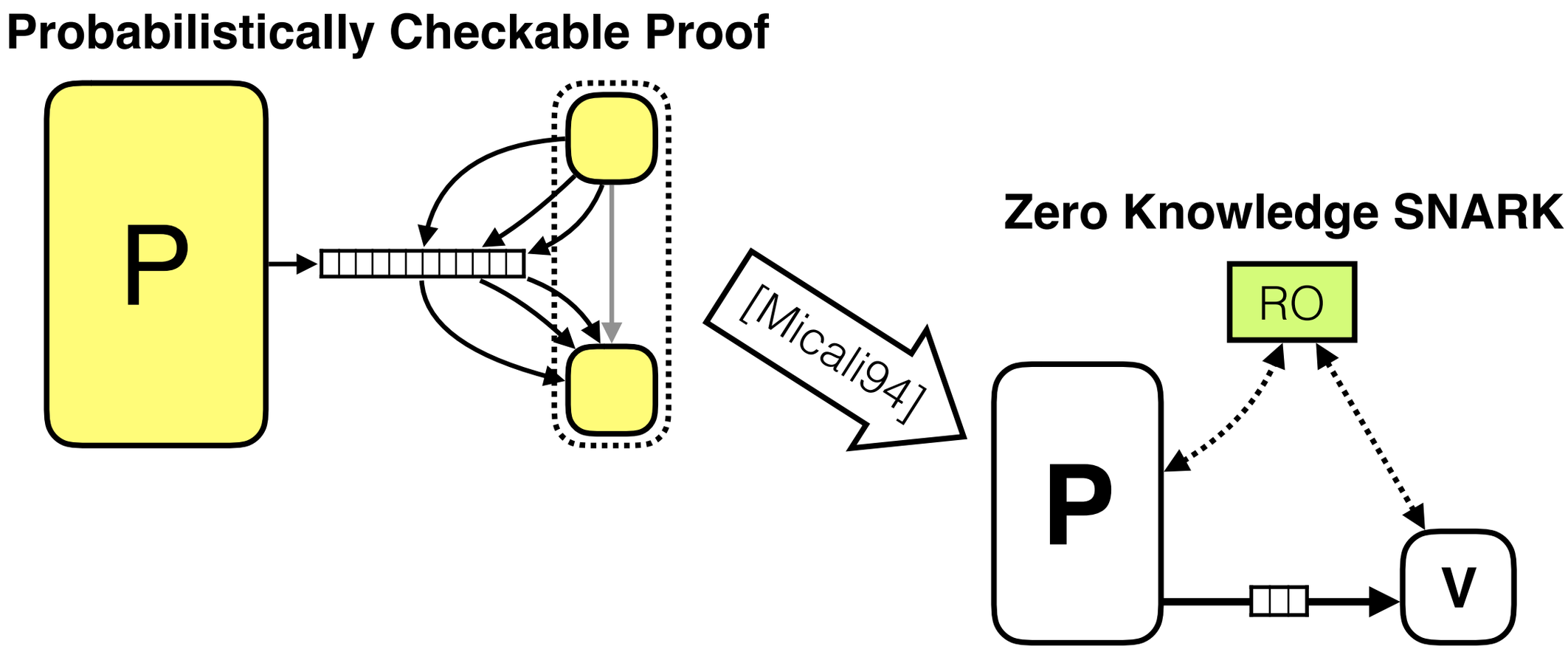
[B**C**S16][RRR16]



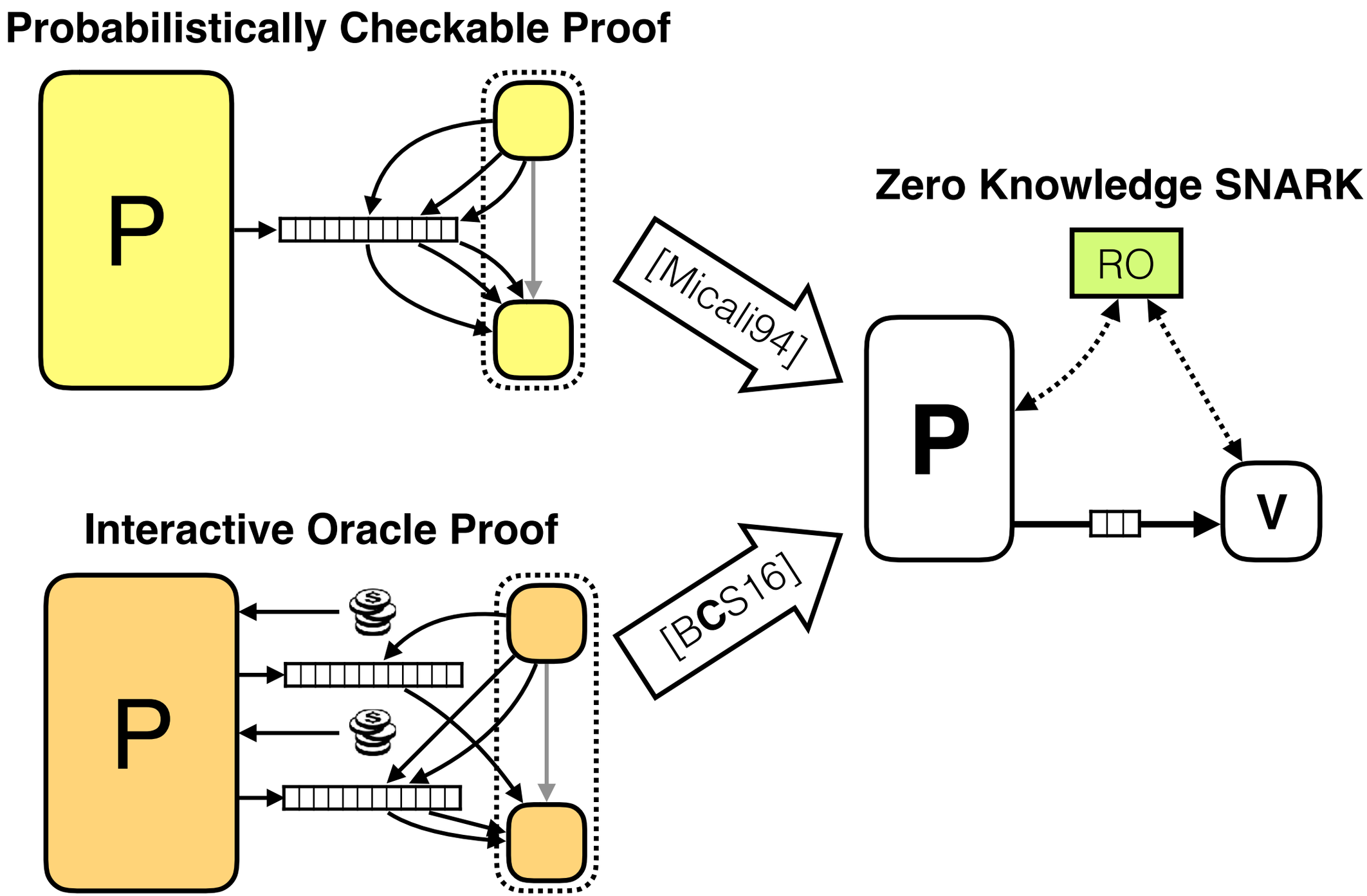
	IP	PCP	IOP
randomness	✓	✓	✓
interaction	✓		✓
probabilistic checking		✓	✓

Tackling Problem #2:
New ZK-SNARKs With Random Oracles

Tackling Problem #2:
New ZK-SNARKs With Random Oracles

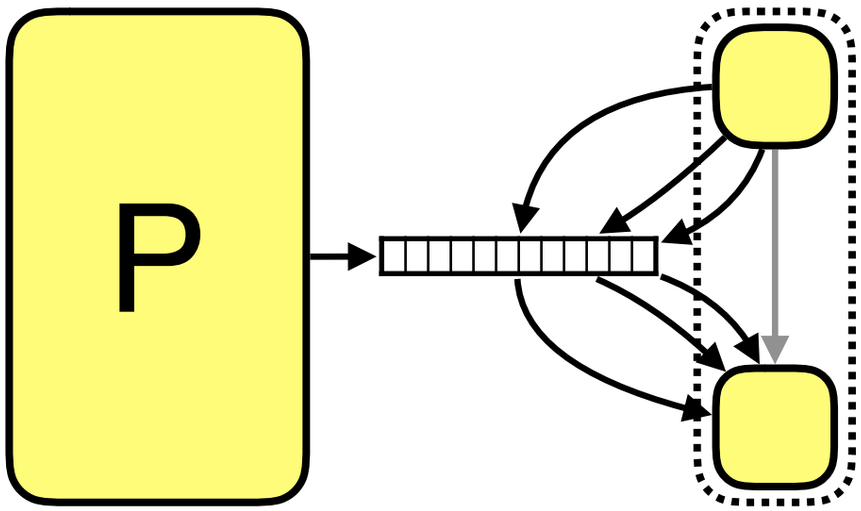


Tackling Problem #2:
New ZK-SNARKs With Random Oracles



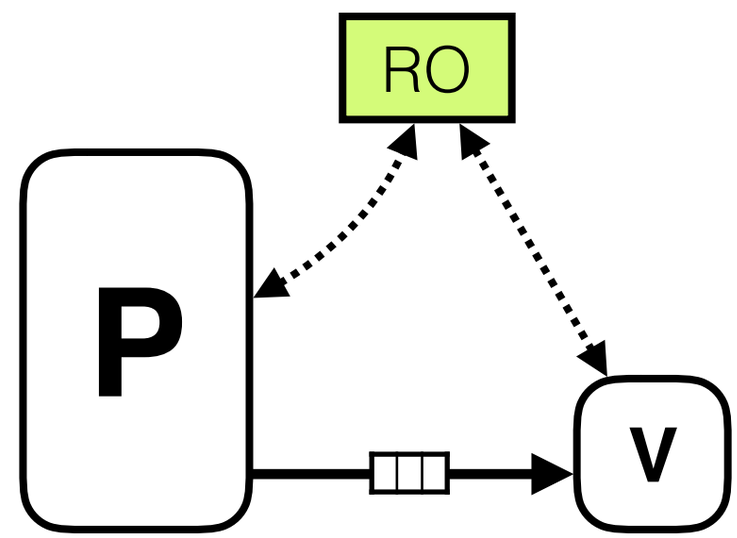
Tackling Problem #2:
New ZK-SNARKs With Random Oracles

Probabilistically Checkable Proof

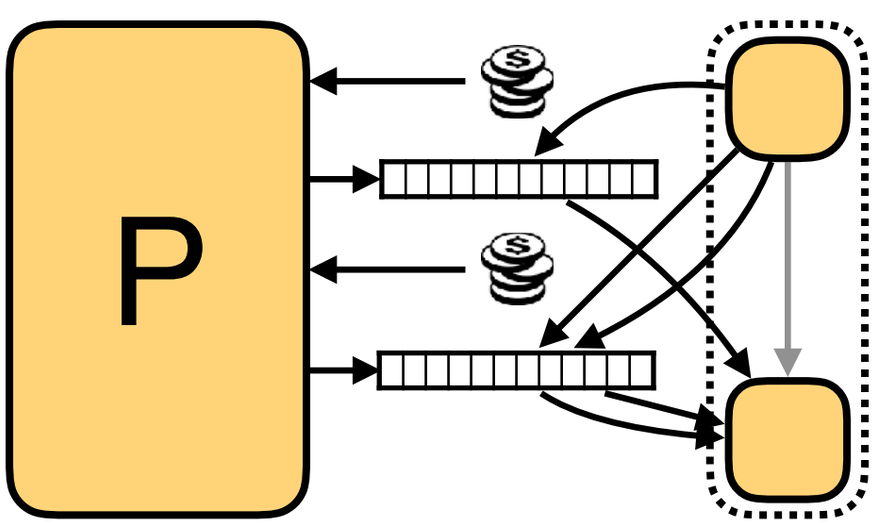


[Micali94]

Zero Knowledge SNARK



Interactive Oracle Proof



[BCS16]

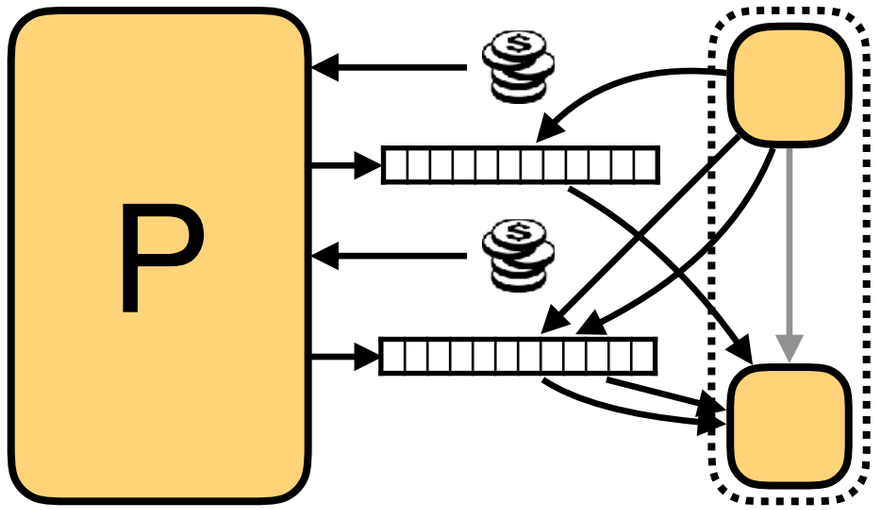
Q: any efficiency gains?

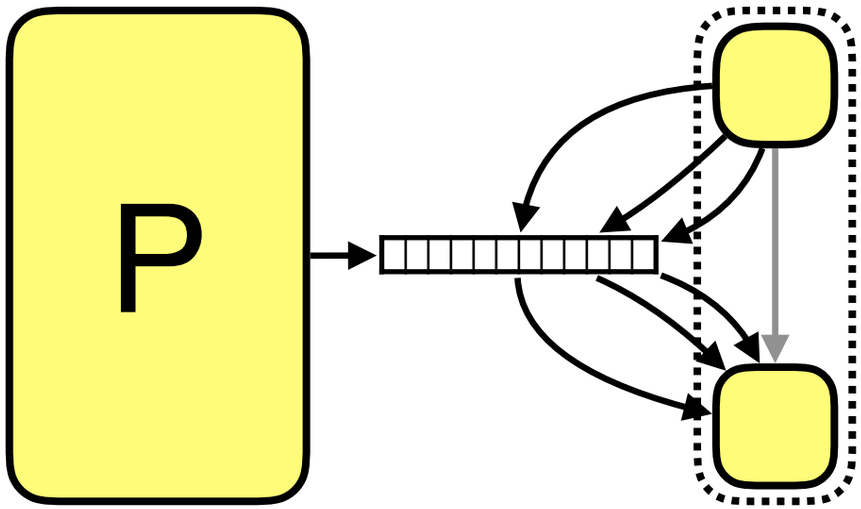
Tackling Problem #2:

IOPs are more efficient than PCPs

Tackling Problem #2:

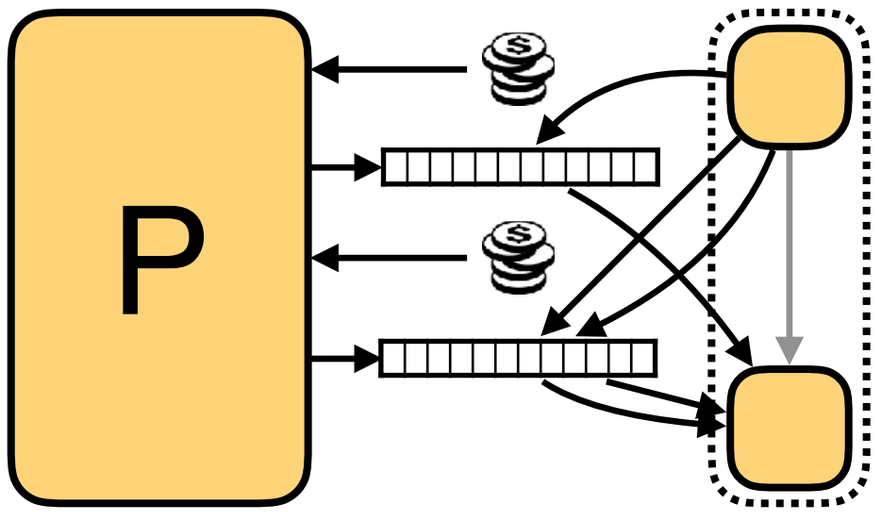
IOPs are more efficient than PCPs

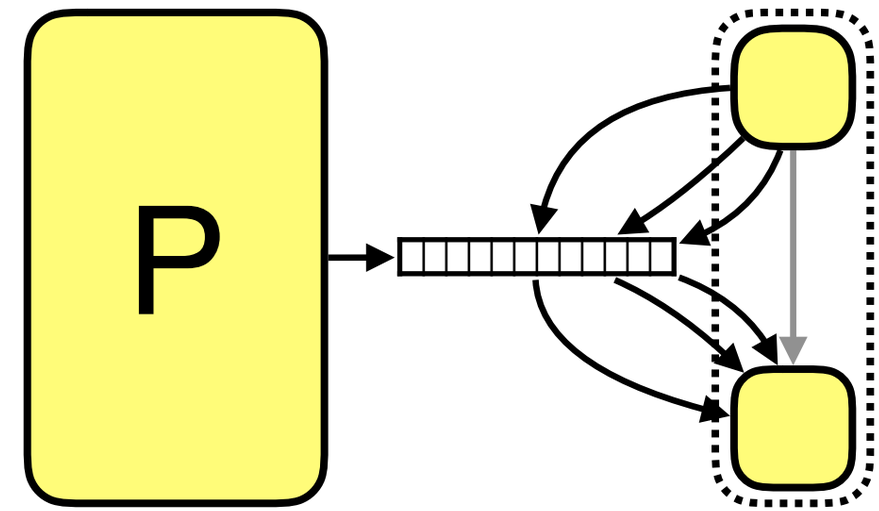




Tackling Problem #2:

IOPs are more efficient than PCPs

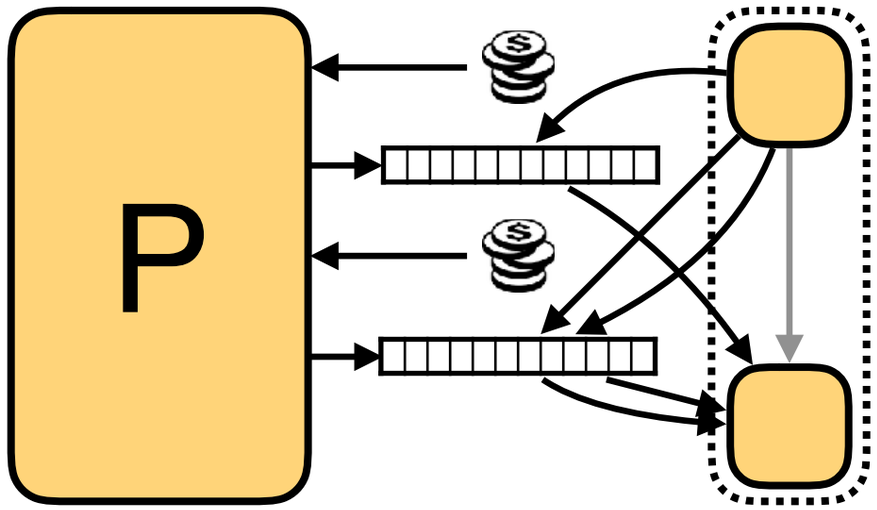




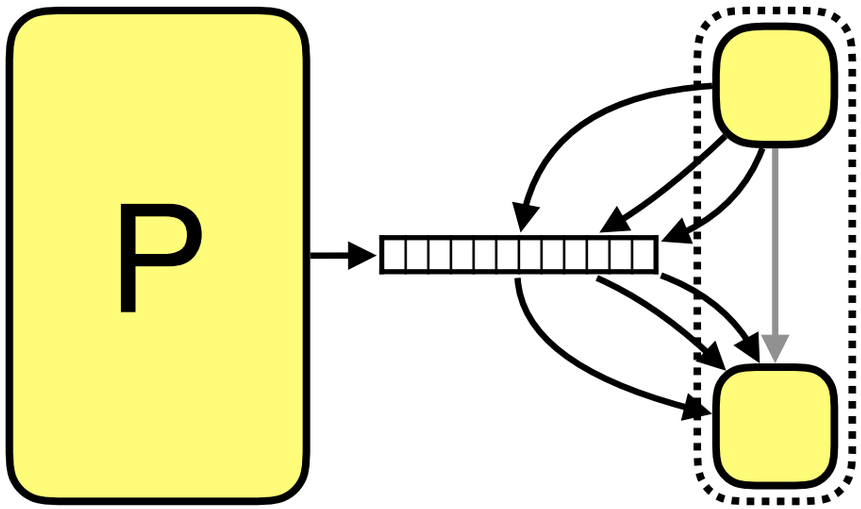
best proof length
without ZK

Tackling Problem #2:

IOPs are more efficient than PCPs

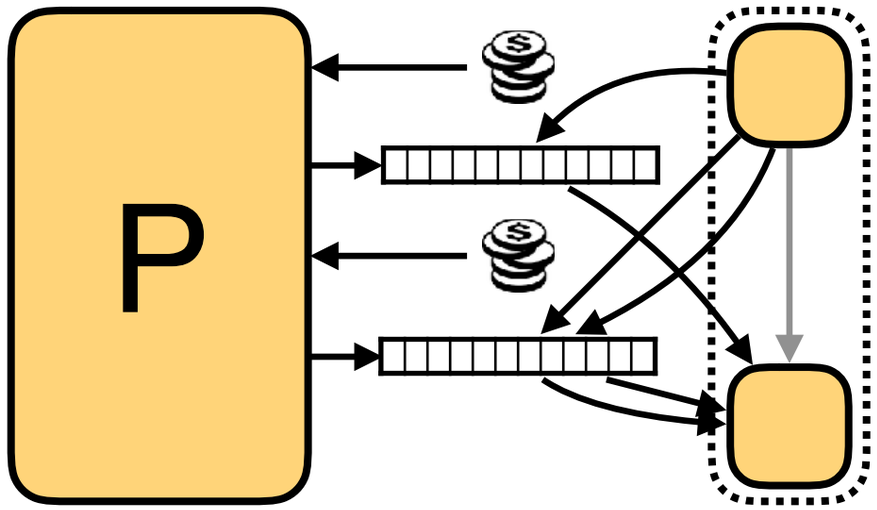


best proof length
without ZK



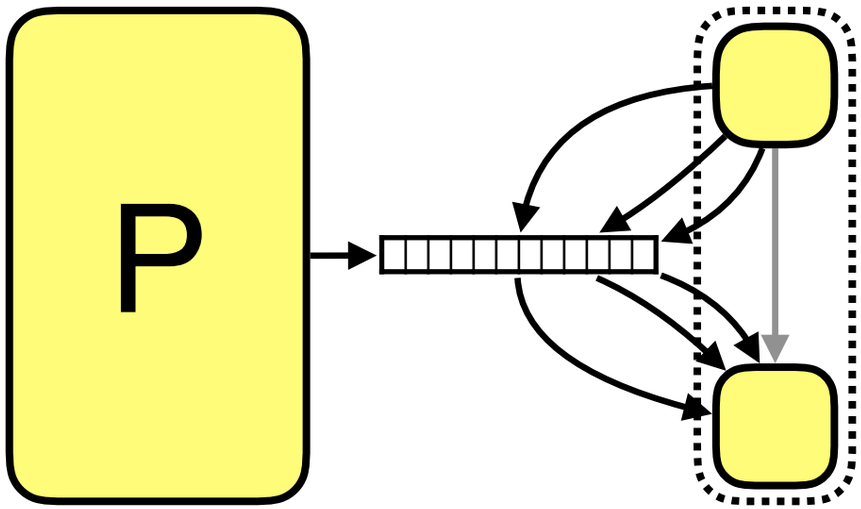
quasilinear
[BS08][Din07]

Tackling Problem #2:
IOPs are more efficient than PCPs



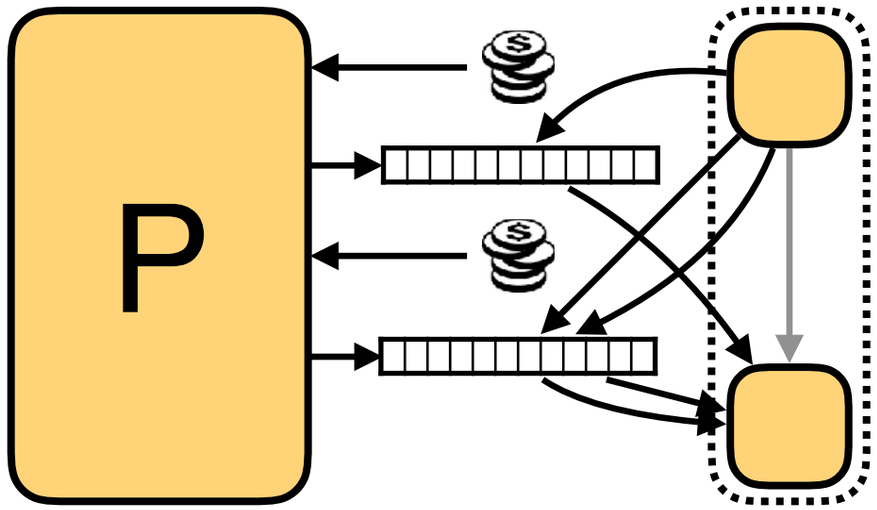
linear
[B**C**GRS16]

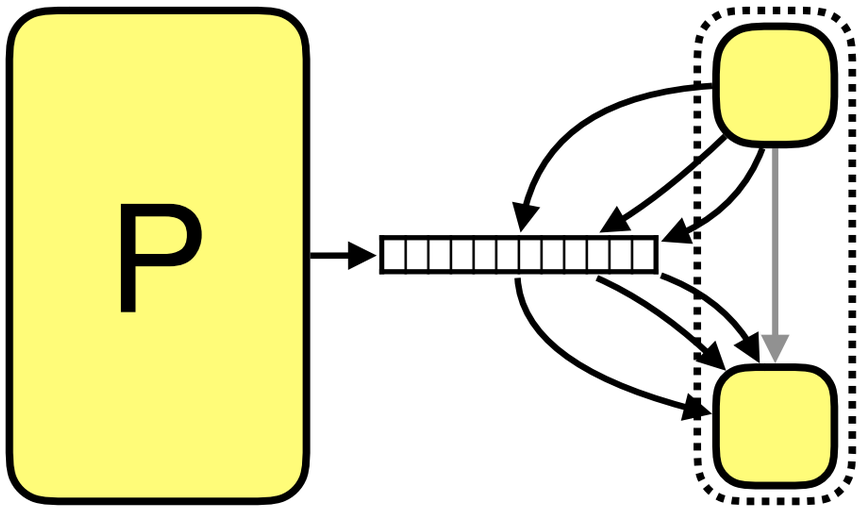
best proof length
without ZK



quasilinear
[BS08][Din07]

Tackling Problem #2:
IOPs are more efficient than PCPs





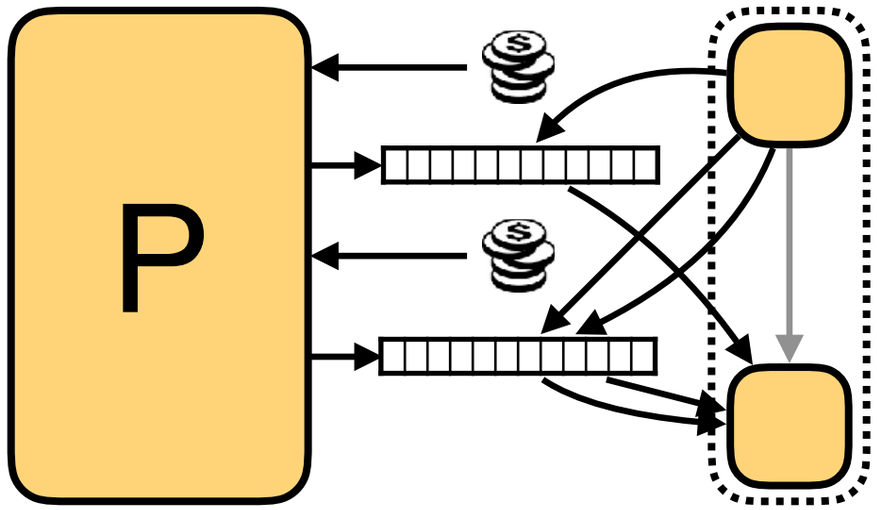
linear
[B**C**GRS16]

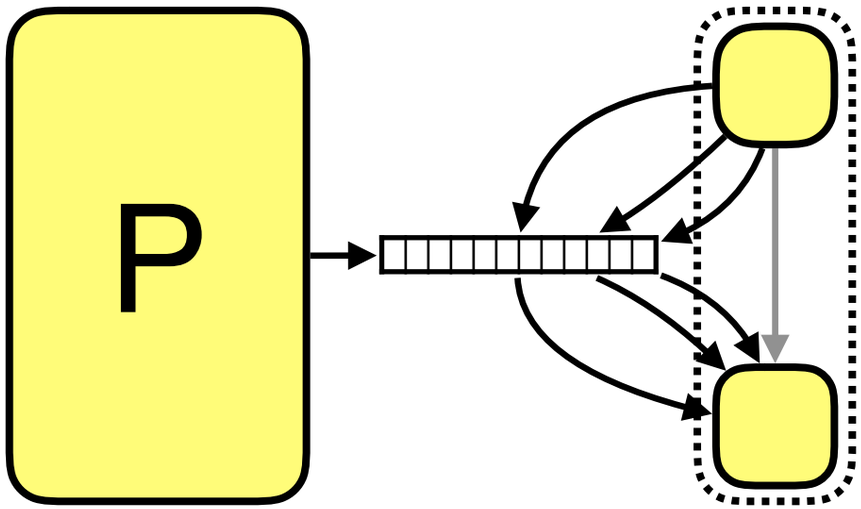
best proof length
without ZK

quasilinear
[BS08][Din07]

best proof length
with ZK

Tackling Problem #2:
IOPs are more efficient than PCPs





linear
[B**C**GRS16]

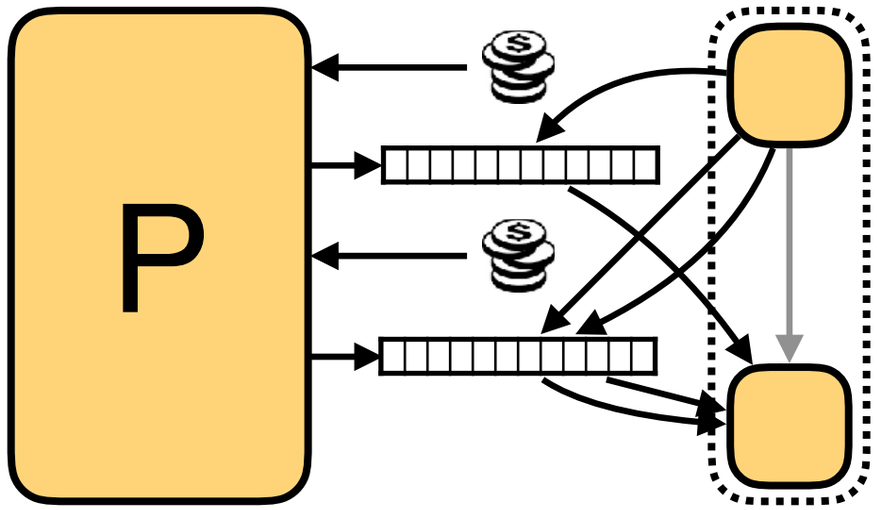
best proof length
without ZK

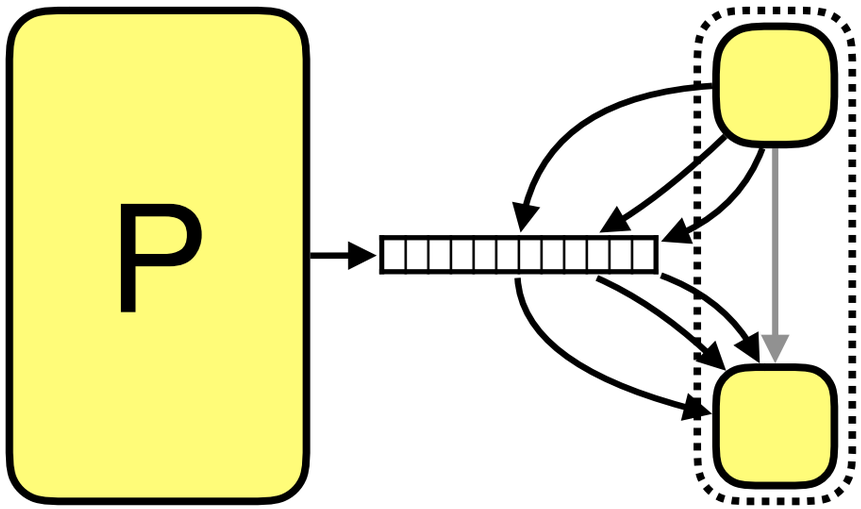
best proof length
with ZK

quasilinear
[BS08][Din07]

polynomial
[KPT97]

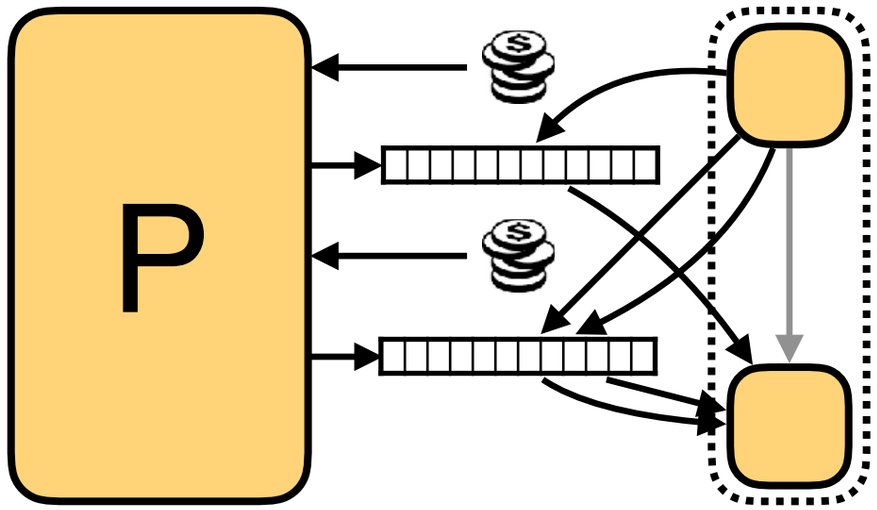
Tackling Problem #2:
IOPs are more efficient than PCPs

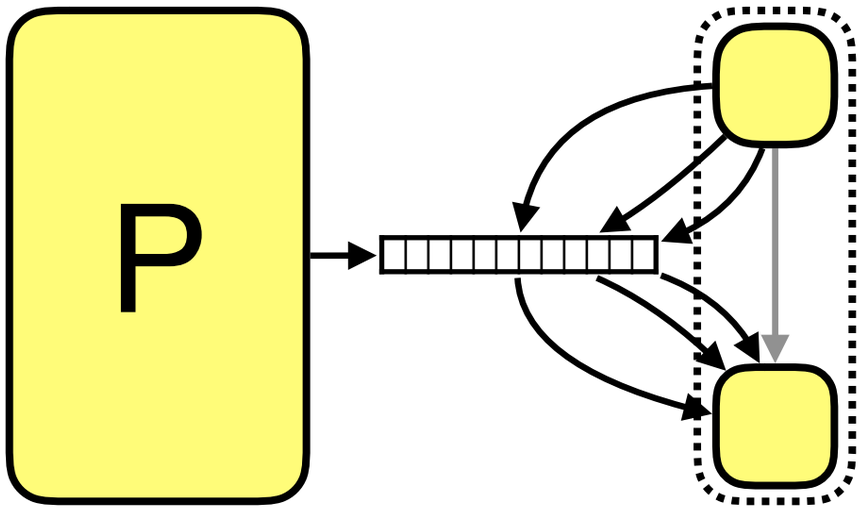




linear [B**C**GRS16]	best proof length without ZK	quasilinear [BS08][Din07]
quasilinear [B**C**GV16]	best proof length with ZK	polynomial [KPT97]

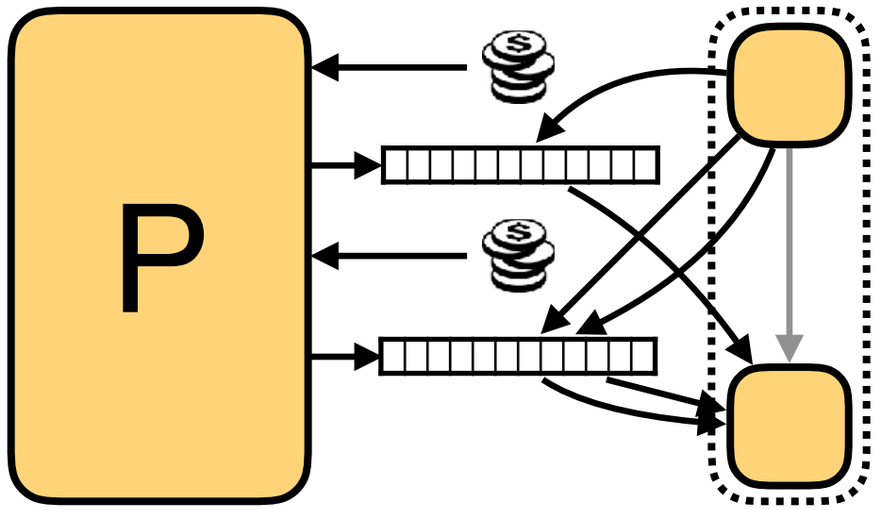
Tackling Problem #2:
IOPs are more efficient than PCPs

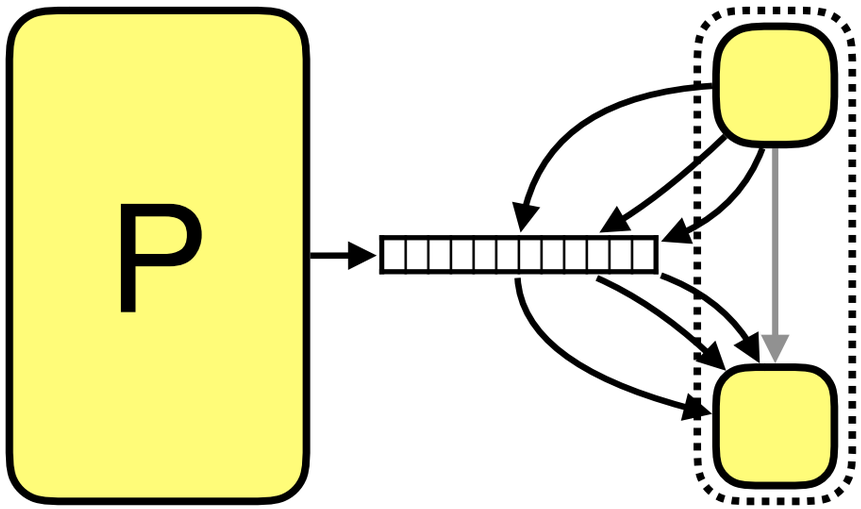




linear [B**C**GRS16]	best proof length without ZK	quasilinear [BS08][Din07]
quasilinear [B**C**GV16]	best proof length with ZK	polynomial [KPT97]
cheaper ZK… [B**C**FGRS16][B**C**FS17]		

Tackling Problem #2:
IOPs are more efficient than PCPs

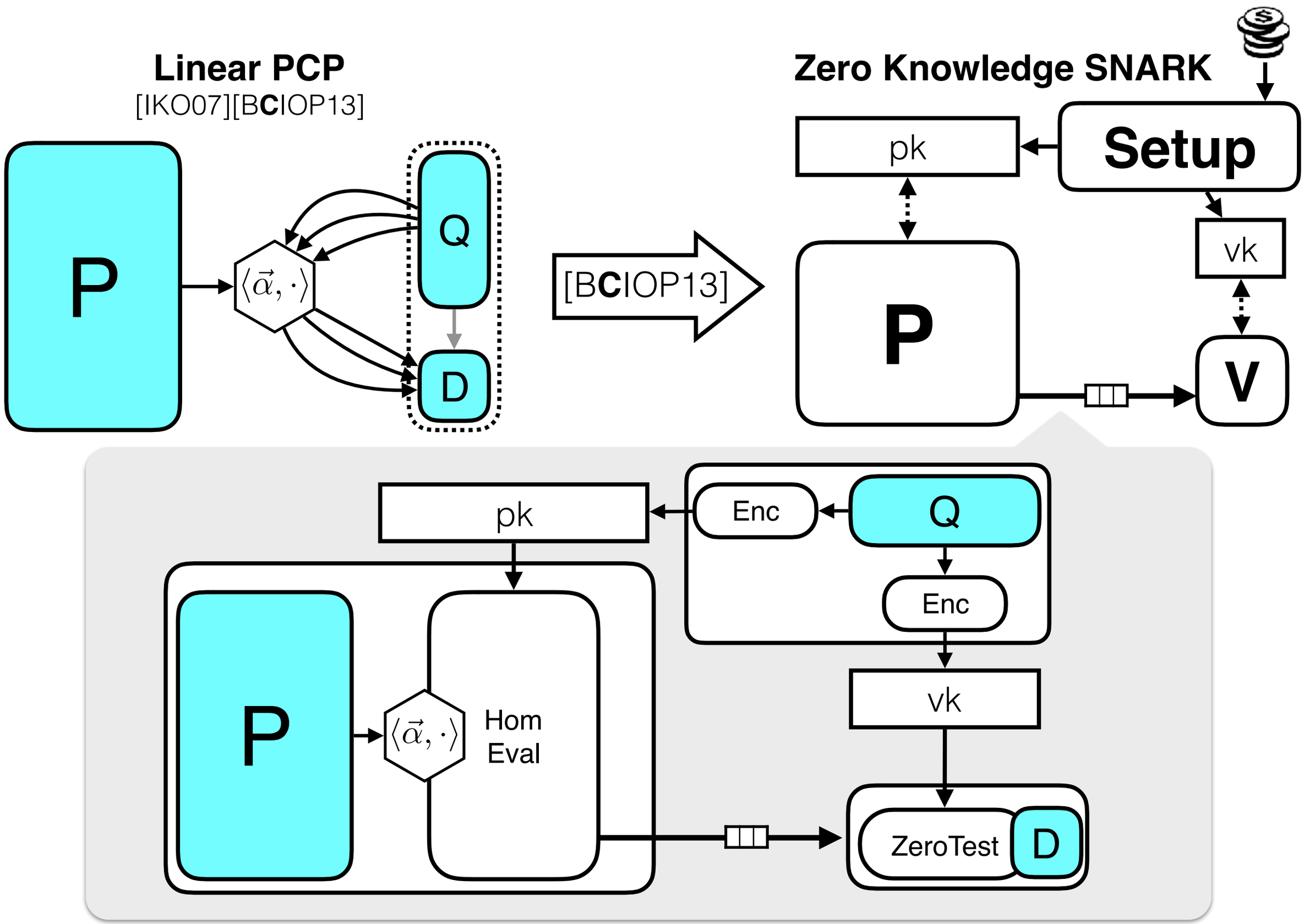




linear [B**C**GRS16]	best proof length without ZK	quasilinear [BS08][Din07]
quasilinear [B**C**GV16]	best proof length with ZK	polynomial [KPT97]
cheaper ZK… [B**C**FGRS16][B**C**FS17]		

Great progress, but still more research is needed for practical deployment.

Deployed Today



libsnark: C++ library for ZK-SNARKs
(libsnark.org)

libsnark: C++ library for ZK-SNARKs
(libsnark.org)

Applications

Frontends

Backends

Shared Core

libsnark: C++ library for ZK-SNARKs
(libsnark.org)

Applications

Frontends

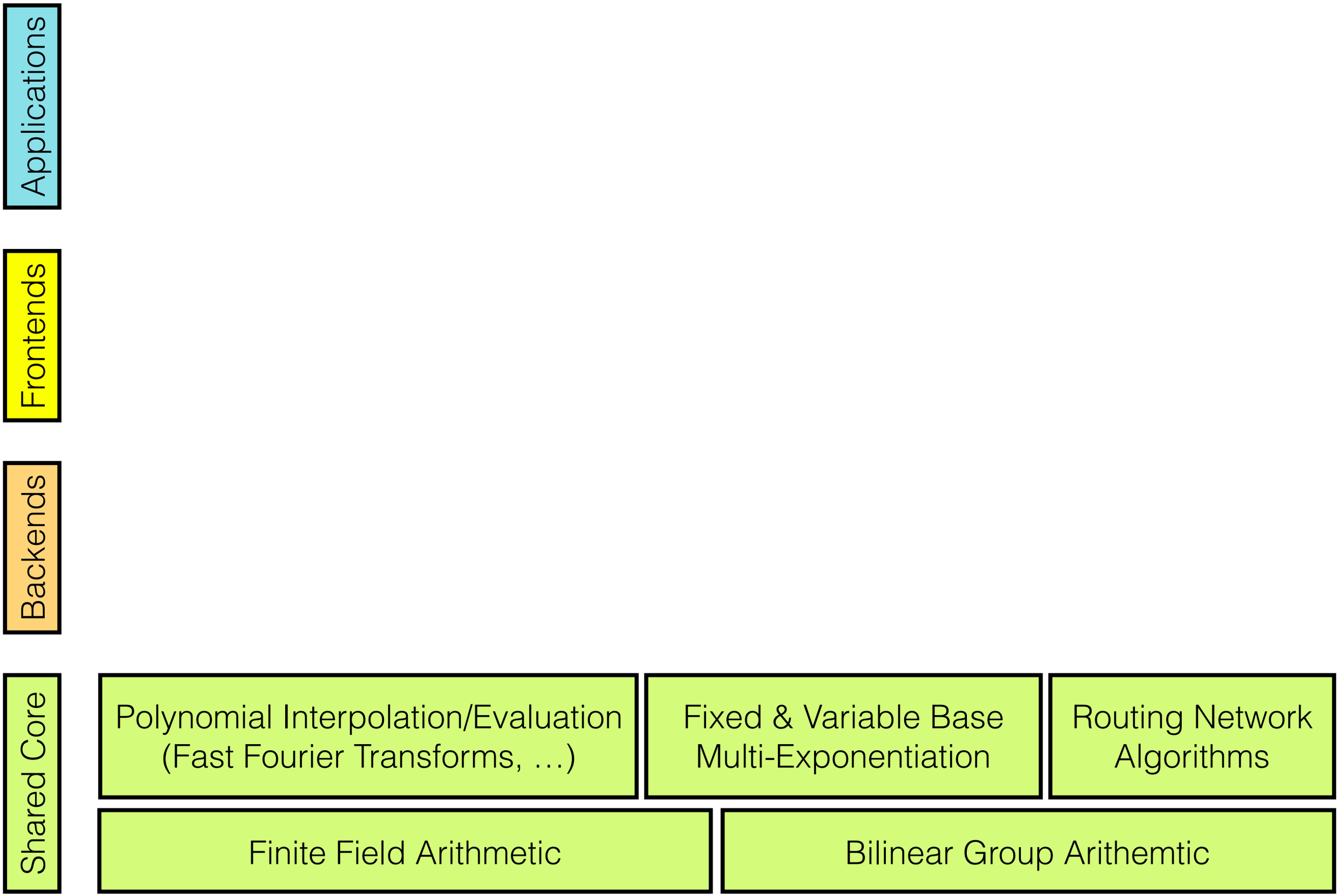
Backends

Shared Core

Finite Field Arithmetic

Bilinear Group Arithemtic

libsnark: C++ library for ZK-SNARKs
(libsnark.org)



libsnark: C++ library for ZK-SNARKs
(libsnark.org)

Applications

Frontends

Backends

Arthmetic Circuits

Shared Core

Polynomial Interpolation/Evaluation (Fast Fourier Transforms, …)

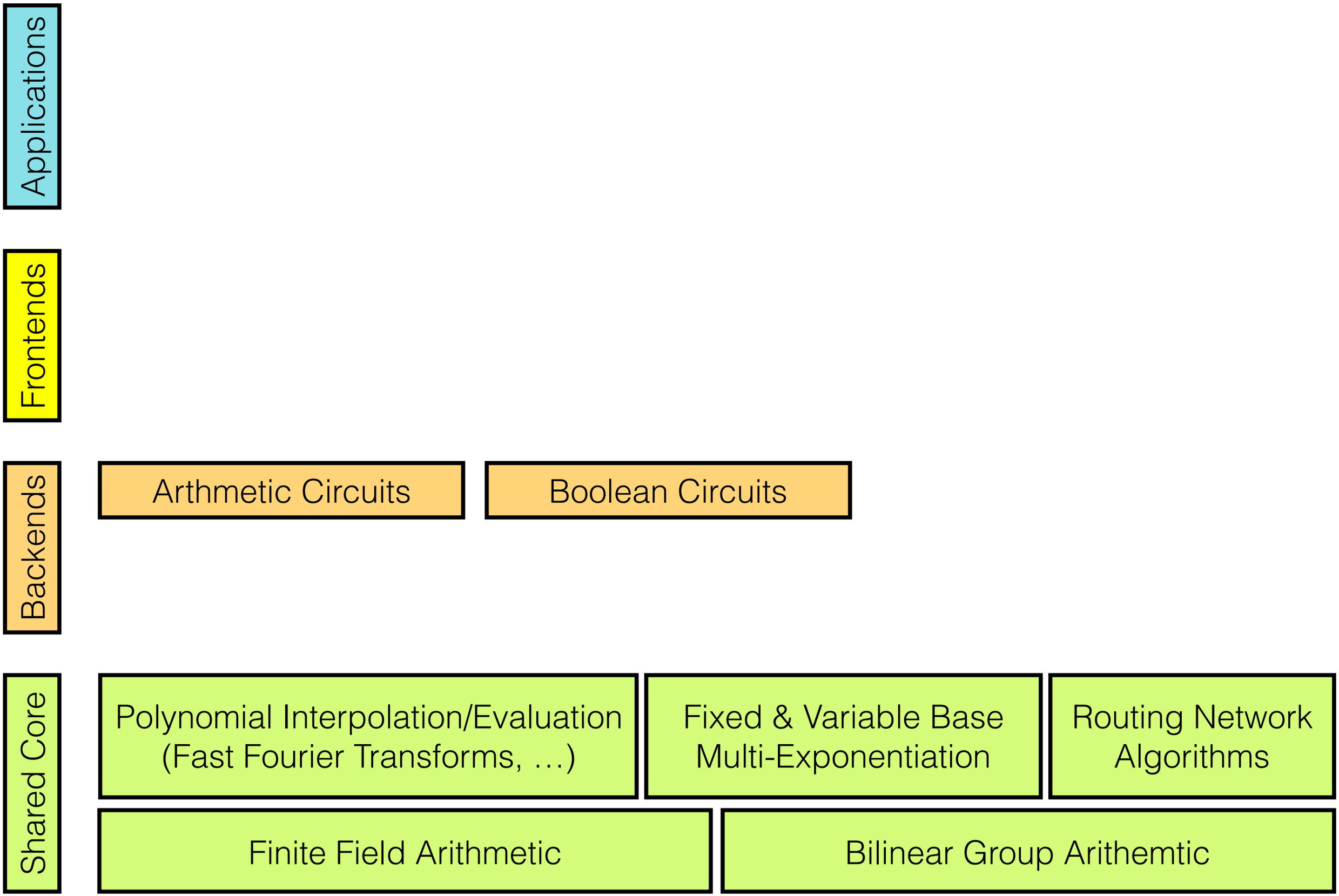
Fixed & Variable Base Multi-Exponentiation

Routing Network Algorithms

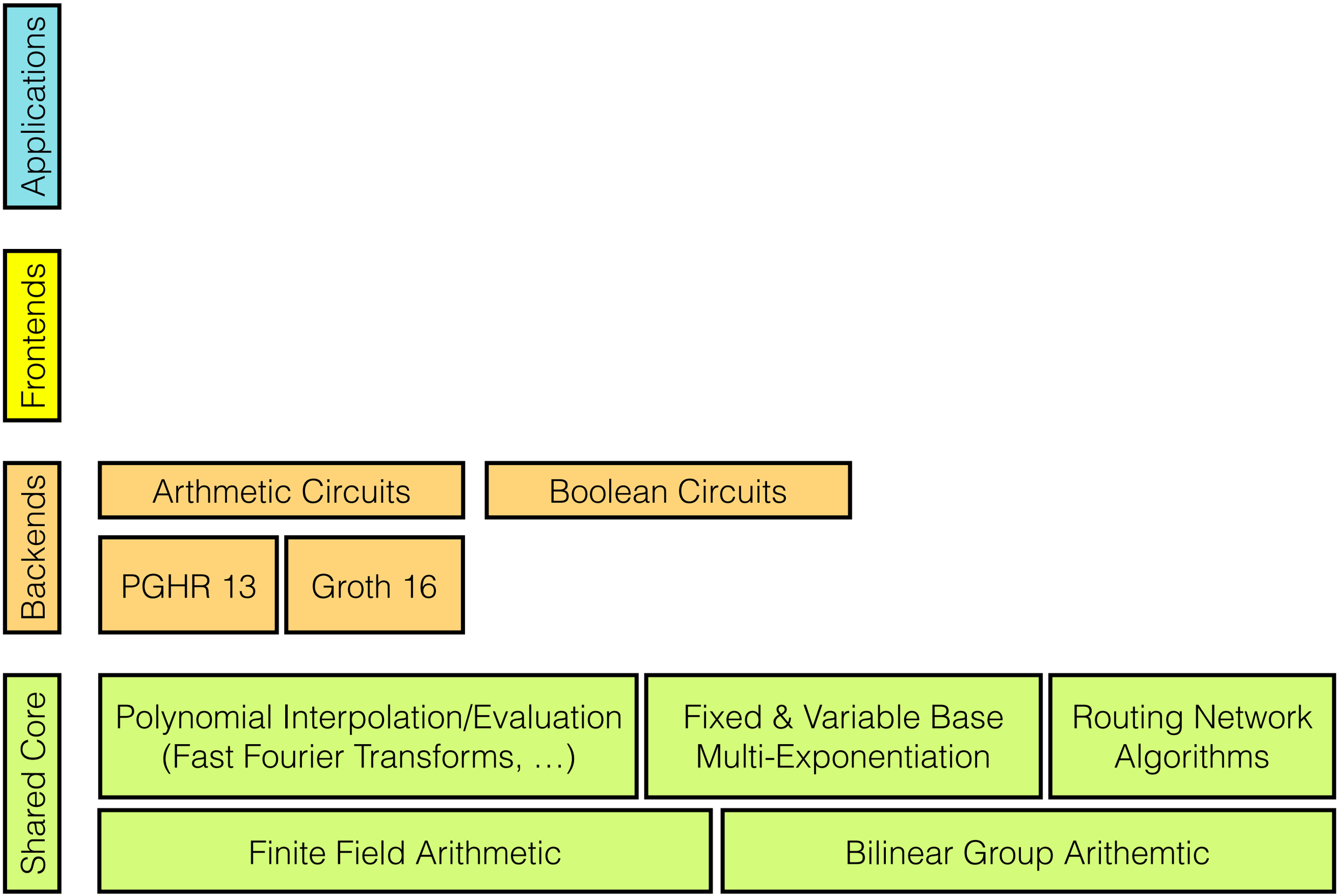
Finite Field Arithmetic

Bilinear Group Arithemtic

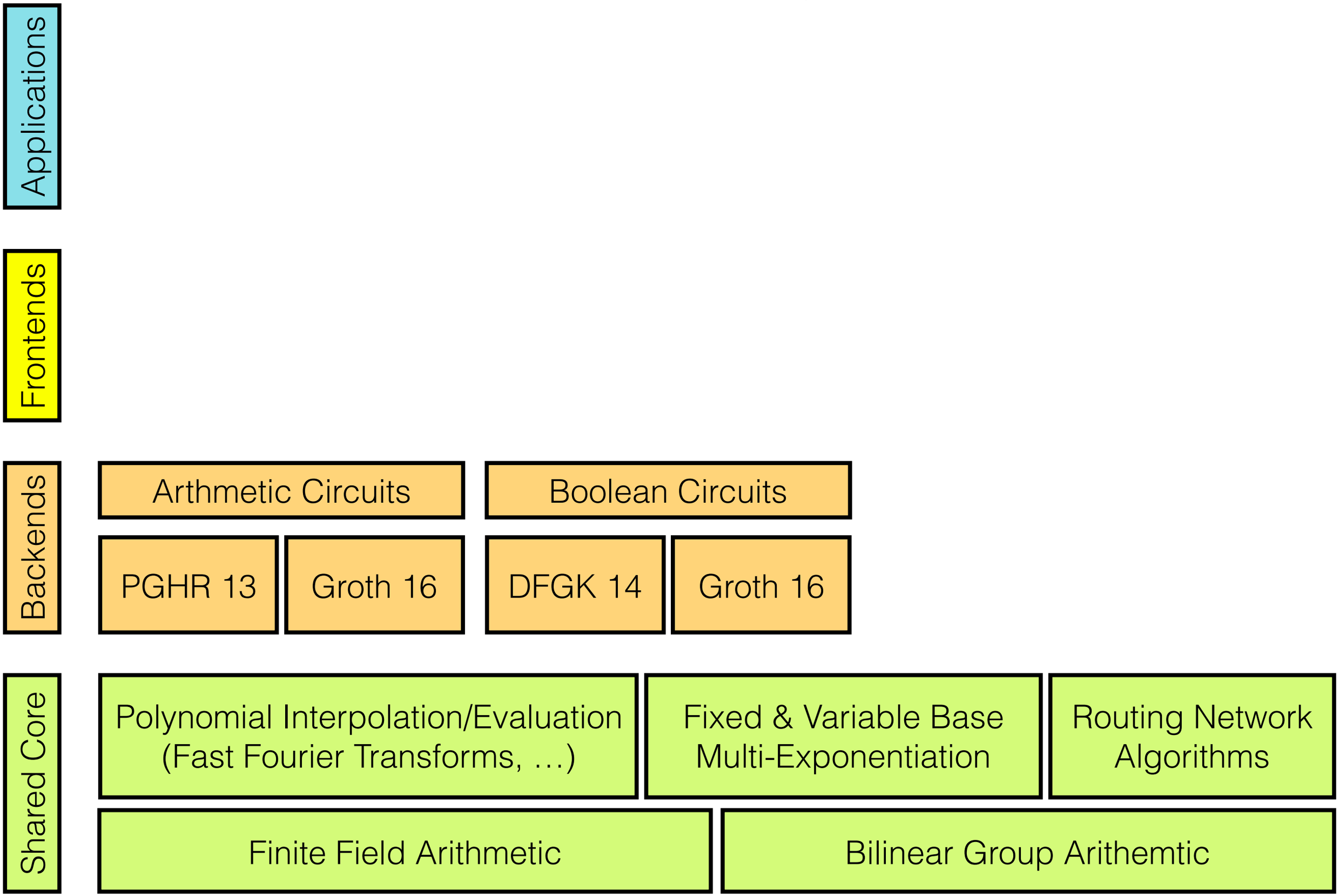
libsnark: C++ library for ZK-SNARKs
(libsnark.org)



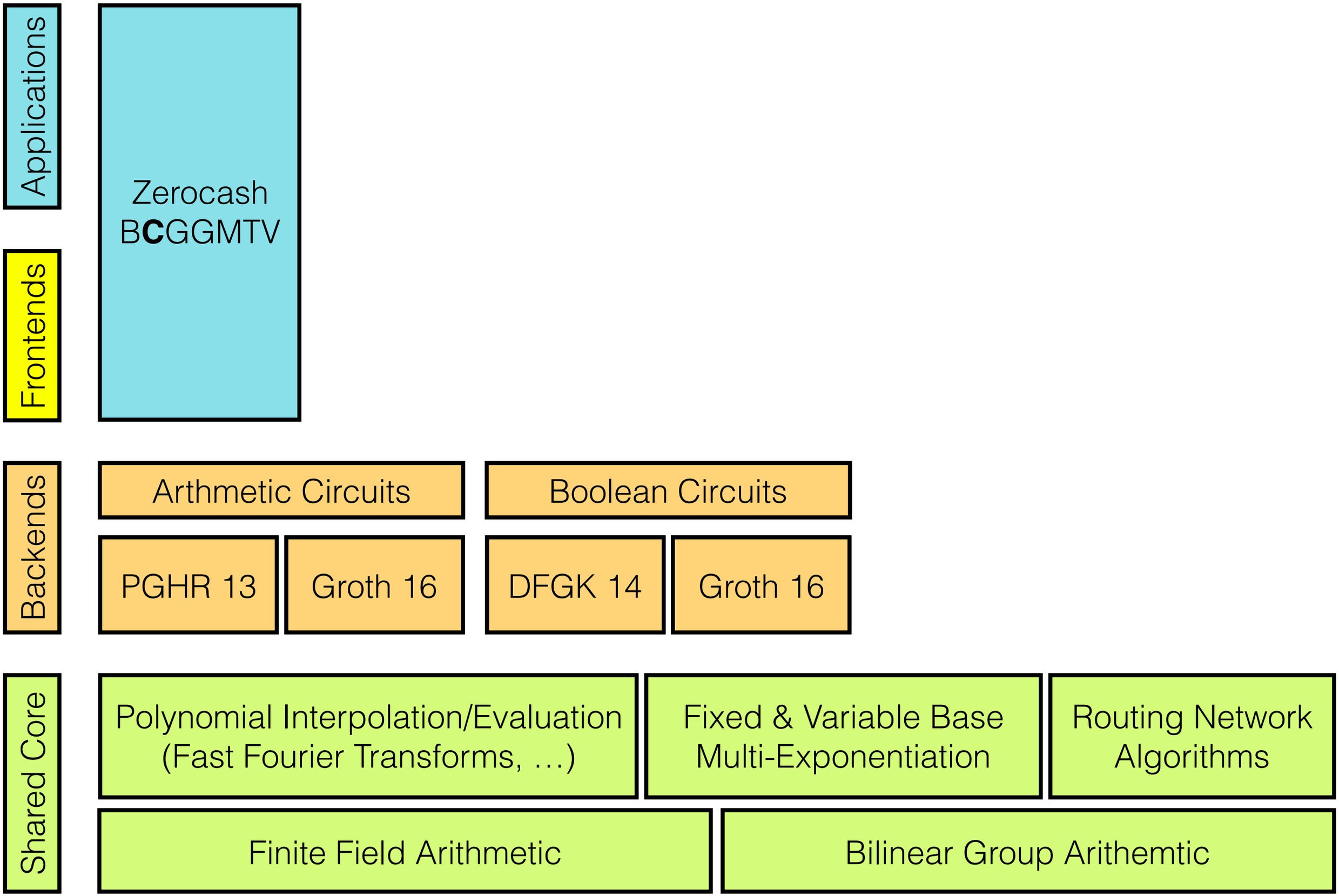
libsnark: C++ library for ZK-SNARKs
(libsnark.org)



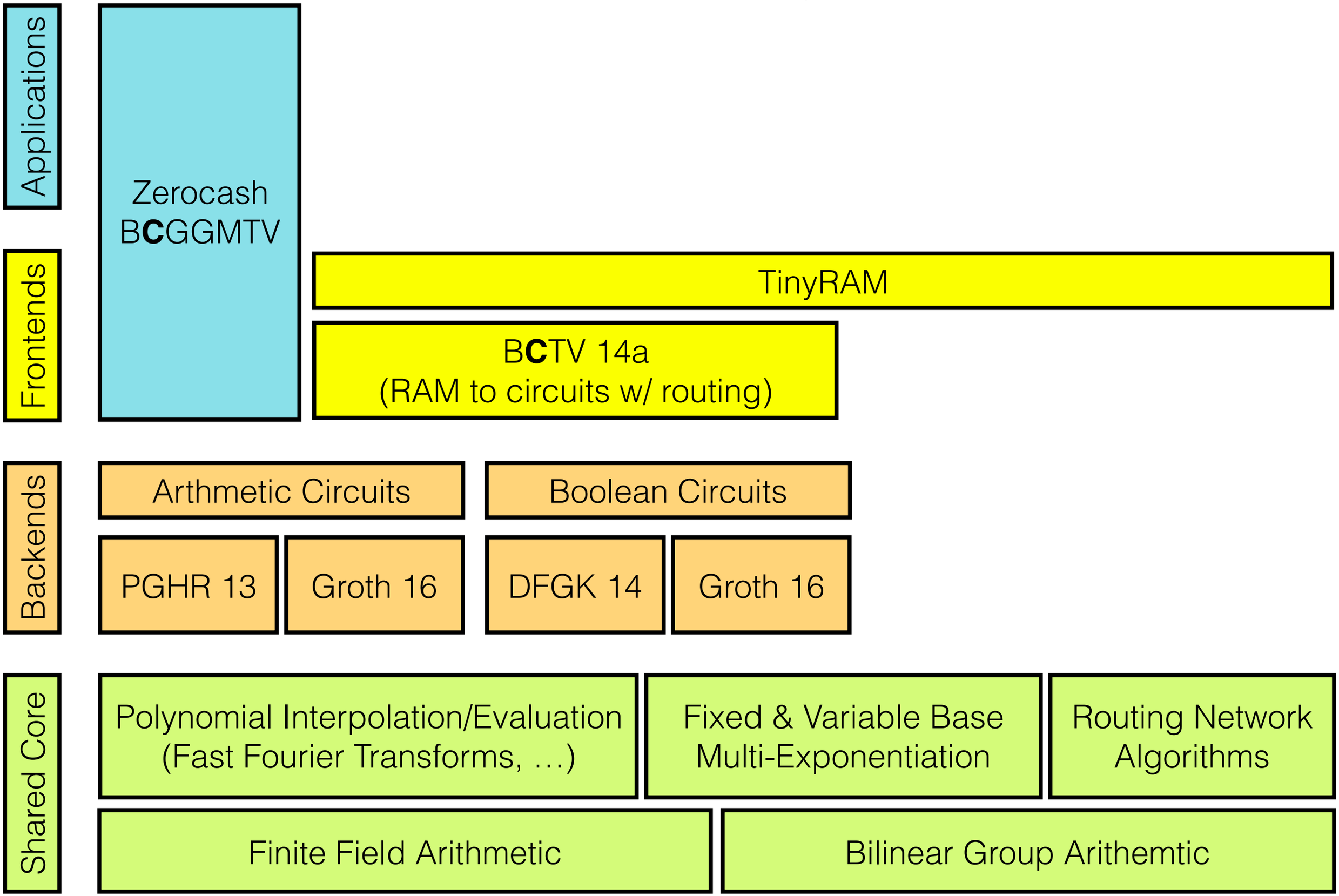
libsnark: C++ library for ZK-SNARKs
(libsnark.org)



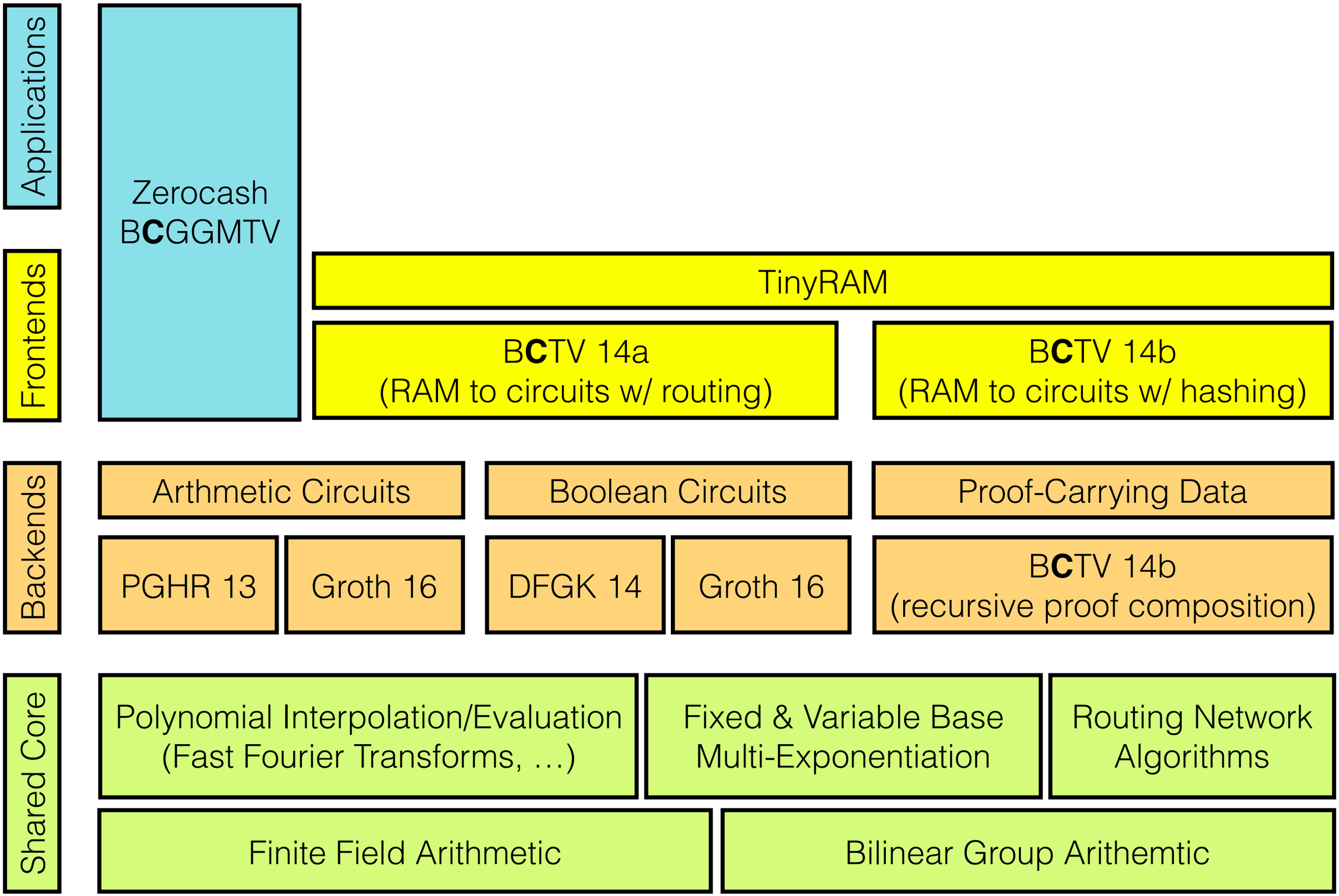
libsnark: C++ library for ZK-SNARKs
(libsnark.org)



libsnark: C++ library for ZK-SNARKs
(libsnark.org)



libsnark: C++ library for ZK-SNARKs
(libsnark.org)



Thanks!