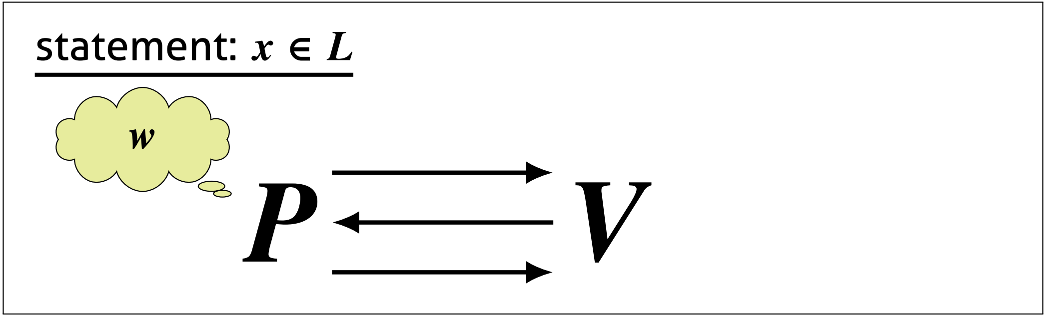
Public-Coin 3-Round Zero-Knowledge from Learning with Errors and Keyless Multi-Collision-Resistant Hash

Susumu Kiyoshima

NTT Research

Zero-knowledge (ZK) arguments

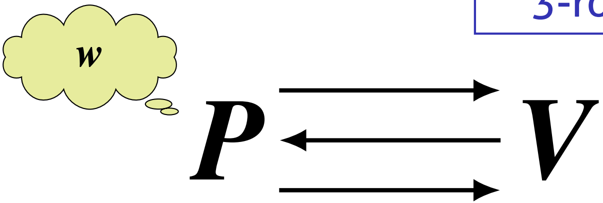


- **Completeness:** $x \in L \Rightarrow V$ accepts a proof created by honest P
- **Soundness:** $x \notin L \Rightarrow V$ rejects a proof created by PPT malicious P
- **ZK:** $x \in L \Rightarrow$ PPT malicious V cannot learn anything beyond $x \in$ L

Zero-knowledge (ZK) arguments

statement: $x \in L$

Our focus:
3-round constructions



- **Completeness:** $x \in L \Rightarrow V$ accepts a proof created by honest P
- **Soundness:** $x \notin L \Rightarrow V$ rejects a proof created by PPT malicious P
- **ZK:** $x \in L \Rightarrow$ PPT malicious V cannot learn anything beyond $x \in$ L

3-round ZK arguments

☺ **Optimal in terms of round complexity**

- **2-round is impossible** (even w/ non-black-box simulation) [Goldreich–Oren94]

☹ **Difficult to obtain**

- **3-round ZK with black-box simulation is impossible** [Goldreich–Krawczyk96]
- **Until recently, 3-round ZK had been obtained only under:**
  - **unfalsifiable assumptions** (e.g., knowledge-of-exponent assumptions)
    [Hada–Tanaka98, Bellare–Palacio04, Canetti–Dakdouk08, …]
  - **weak definitions** (e.g., super-poly simulation, bounded non-uniformity, weak ZK, …)
    [Pass03, Bitansky–Canetti–Paneth–Rosen14, Bitansky–Brakerski–Kalai–Paneth–Vaikuntanathan16, Bitansky–Khurana–Paneth19,…]

3-round ZK by [Bitansky–Kalai–Paneth18 (BKP18)]

Theorem [BKP18]

3-round ZK argument can be obtained by relying on:

(1) quasi-poly hardness of LWE (or FHE + standard crypto), and

(2) slightly super-poly hardness of keyless multi-collision-resistant hash function

3-round ZK by [Bitansky–Kalai–Paneth18 (BKP18)]

Theorem [BKP18]

3-round ZK argument can be obtained by relying on:
(1) quasi-poly hardness of LWE (or FHE + standard crypto), and
(2) slightly super-poly hardness of keyless multi-collision-resistant hash function

3-round ZK by [Bitansky–Kalai–Paneth18 (BKP18)]

Theorem [BKP18]

3-round ZK argument can be obtained by relying on:

(1) quasi-poly hardness of LWE (or FHE + standard crypto), and

(2) slightly super-poly hardness of keyless multi-collision-resistant hash function

- ***n*-collision:** distinct $x_1, \ldots, x_n$ s.t. $H(x_1) = \cdots = H(x_n)$

3-round ZK by [Bitansky–Kalai–Paneth18 (BKP18)]

Theorem [BKP18]

3-round ZK argument can be obtained by relying on:
(1) quasi-poly hardness of LWE (or FHE + standard crypto), and
(2) slightly super-poly hardness of keyless multi-collision-resistant hash function

- **n-collision:** distinct $x_1, \ldots, x_n$ s.t. $H(x_1) = \cdots = H(x_n)$
- **N-collision resistance:** any adversary with non-uniform advice of size s cannot find $N(s)$-collision for $N(s) \gg s$ (e.g., $N(s) = \mathrm{poly}(s)$)

3-round ZK by [Bitansky–Kalai–Paneth18 (BKP18)]

Theorem [BKP18]

3-round ZK argument can be obtained by relying on:
(1) quasi-poly hardness of LWE (or FHE + standard crypto), and
(2) slightly super-poly hardness of keyless multi-collision-resistant hash function

3-round ZK from simple & falsifiable assumptions!

Our result

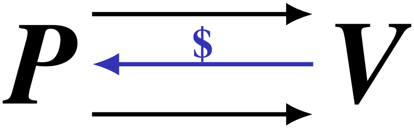
Theorem

Public-coin 3-round ZK argument can be obtained by relying on:
(1) sub-exponential hardness of LWE, and
(2) slightly super-poly hardness of keyless multi-collision-resistant hash function

Our result

Theorem

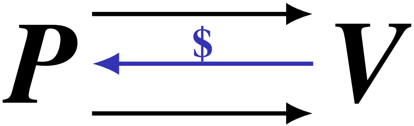
<u>Public-coin</u> 3-round ZK argument can be obtained by relying on:
(1) sub-exponential hardness of LWE, and
(2) slightly super-poly hardness of keyless multi-collision-resistant hash function



Our result

Theorem

Public-coin 3-round ZK argument can be obtained by relying on:
(1) sub-exponential hardness of LWE, and
(2) slightly super-poly hardness of keyless multi-collision-resistant hash function

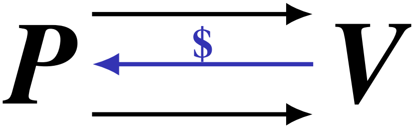


☺ theoretically natural target

Our result

Theorem

Public-coin 3-round ZK argument can be obtained by relying on:
(1) sub-exponential hardness of LWE, and
(2) slightly super-poly hardness of keyless multi-collision-resistant hash function



☺ theoretically natural target

☺ useful properties
– public verifiabilty
– leakage resislience about V's state

Our result

Theorem

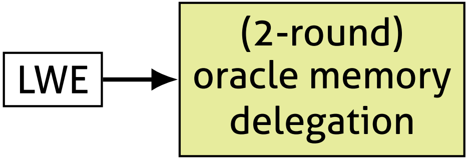
Public-coin 3-round ZK argument can be obtained by relying on:
(1) sub-exponential hardness of LWE, and
(2) slightly super-poly hardness of keyless multi-collision-resistant hash function

- **Comparison with 3-round ZK of [BKP18]**
  - ☺ public-coin construction
  - ☹ slightly stronger assumptions
    (sub-exponentially hard LWE rather than quasi-poly hard LWE)

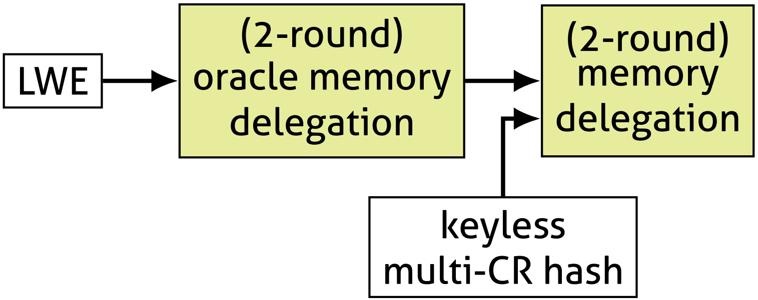
Overview of techniques

Prior approach [BKP18]

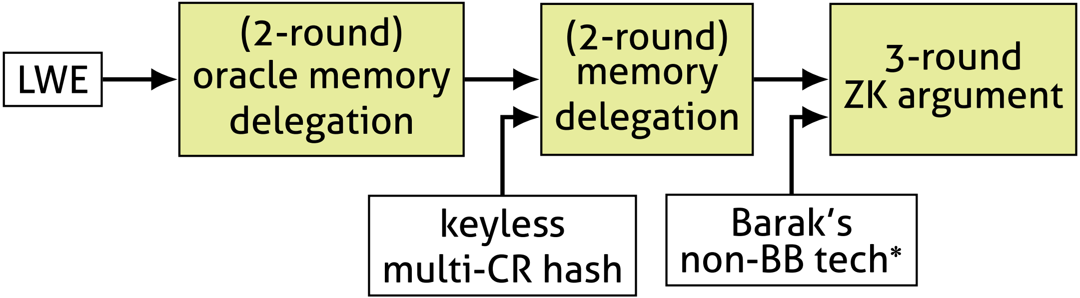
Prior approach [BKP18]



Prior approach [BKP18]

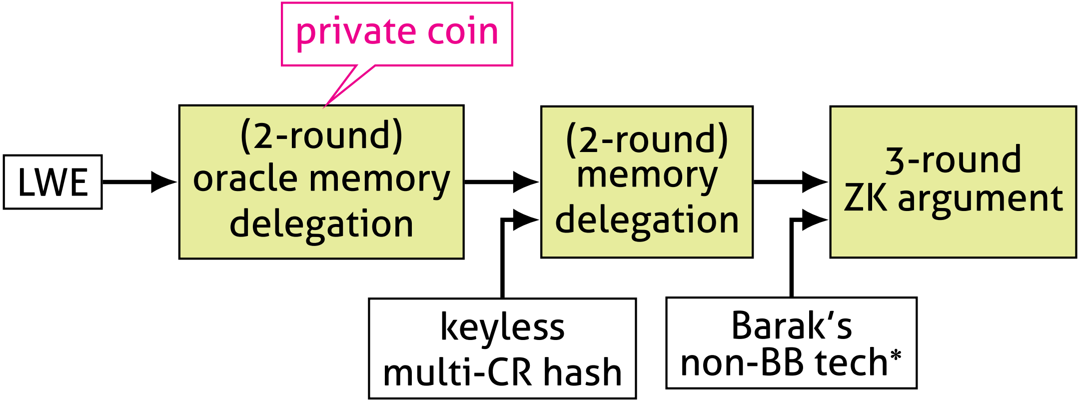


Prior approach [BKP18]



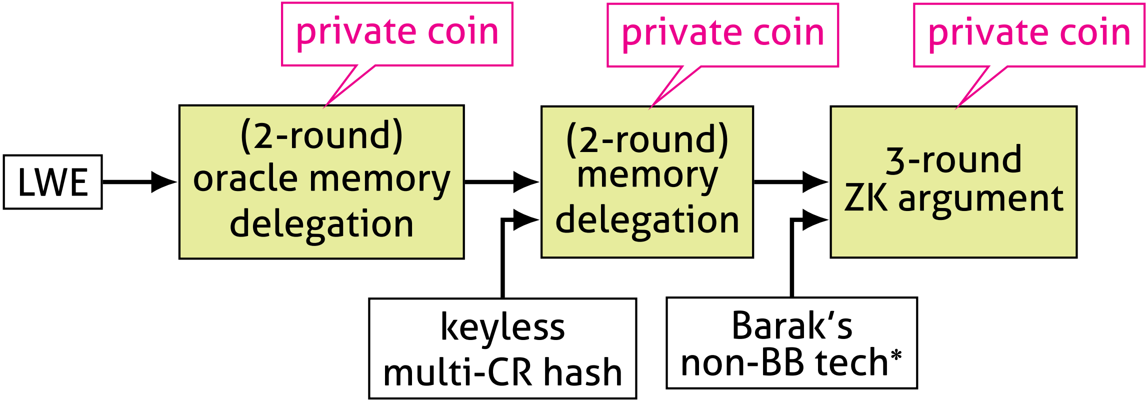
* memory delegation is used as universal argument

Prior approach [BKP18]



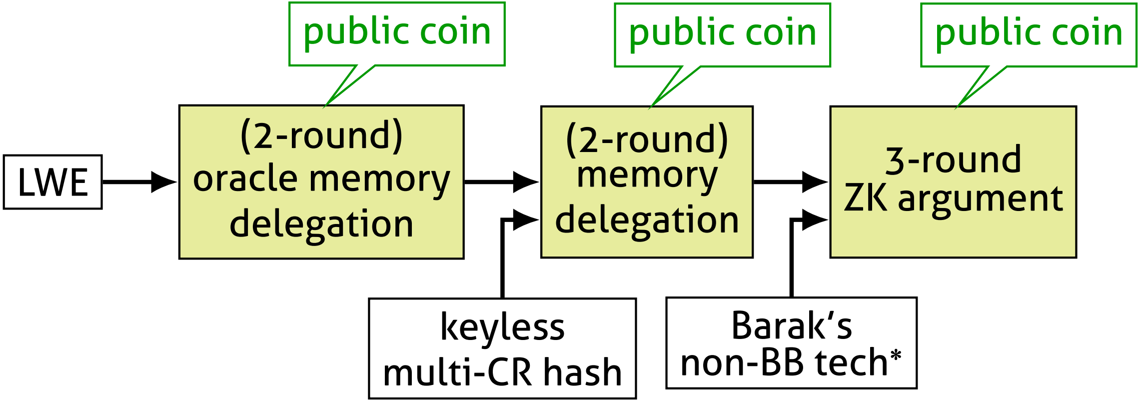
* memory delegation is used as universal argument

Prior approach [BKP18]



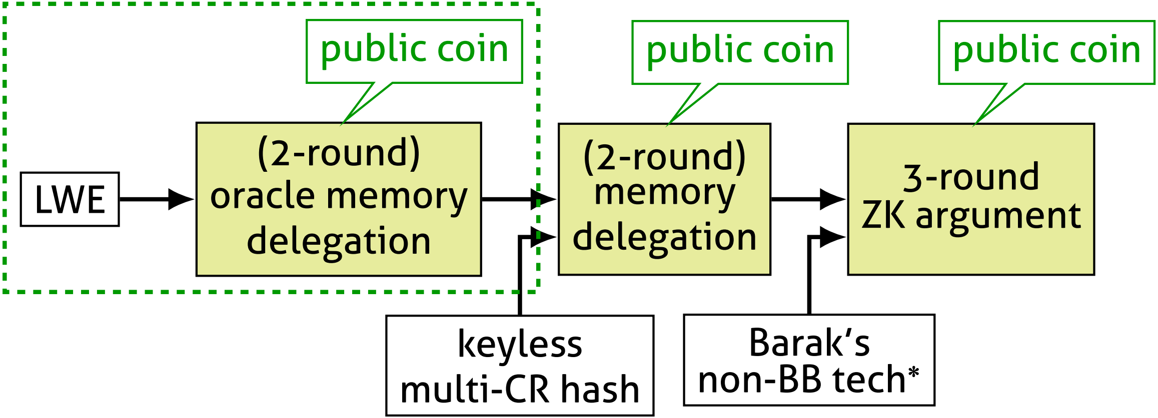
* memory delegation is used as universal argument

Our approach



* memory delegation is used as universal argument

Our approach



* memory delegation is used as universal argument

Oracle memory delegation [BKP18]

memory: x

P

V

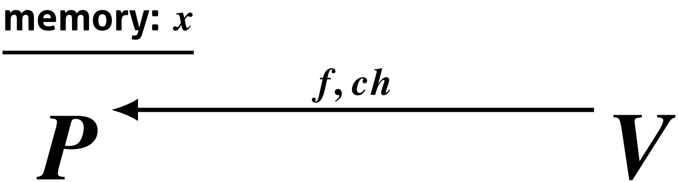
Oracle memory delegation [BKP18]

memory: x

P V

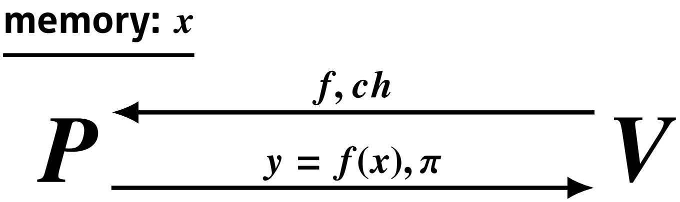
- **Goal:** V delegates (heavy) computation on the memory x to P

Oracle memory delegation [BKP18]



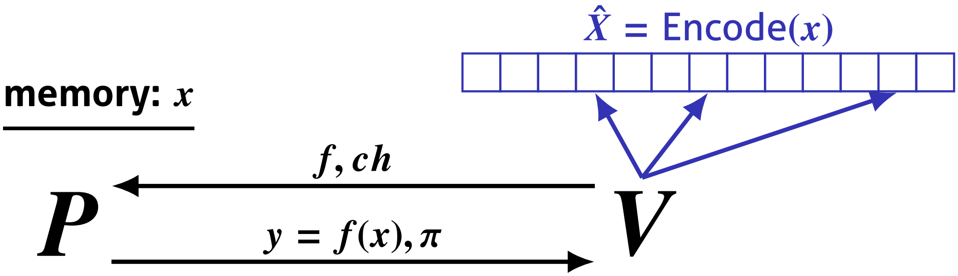
- **Goal:** V delegates (heavy) computation on the memory x to P

Oracle memory delegation [BKP18]



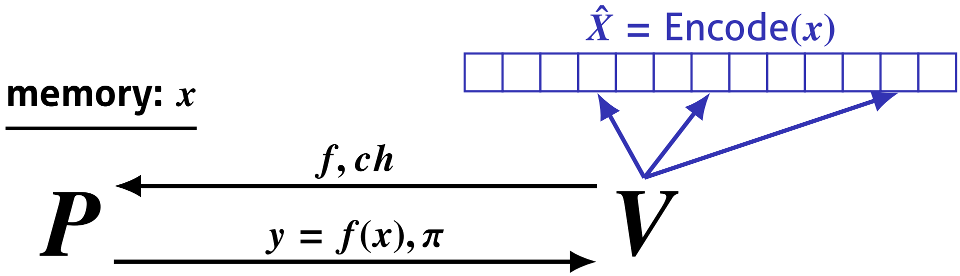
- **Goal:** V delegates (heavy) computation on the memory x to P

Oracle memory delegation [BKP18]



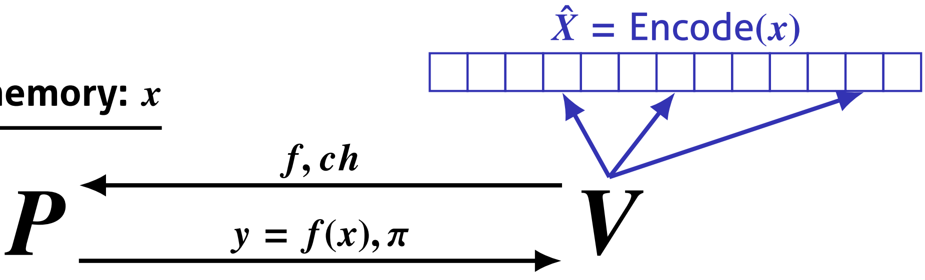
- **Goal:** V delegates (heavy) computation on the memory x to P

Oracle memory delegation [BKP18]



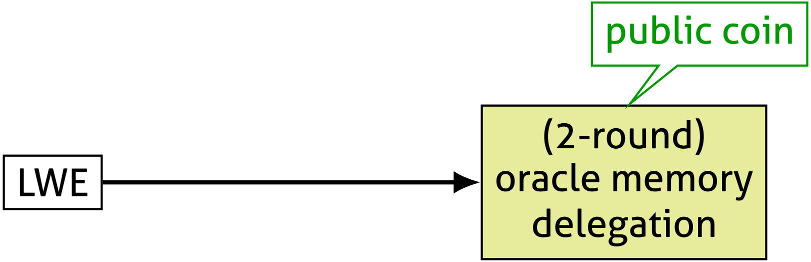
- **Efficiency:** V runs in polynomial time in the security parameter λ even for memory x of slightly super-poly length (e.g., $\lambda^{\log \log \lambda}$)

Oracle memory delegation [BKP18]

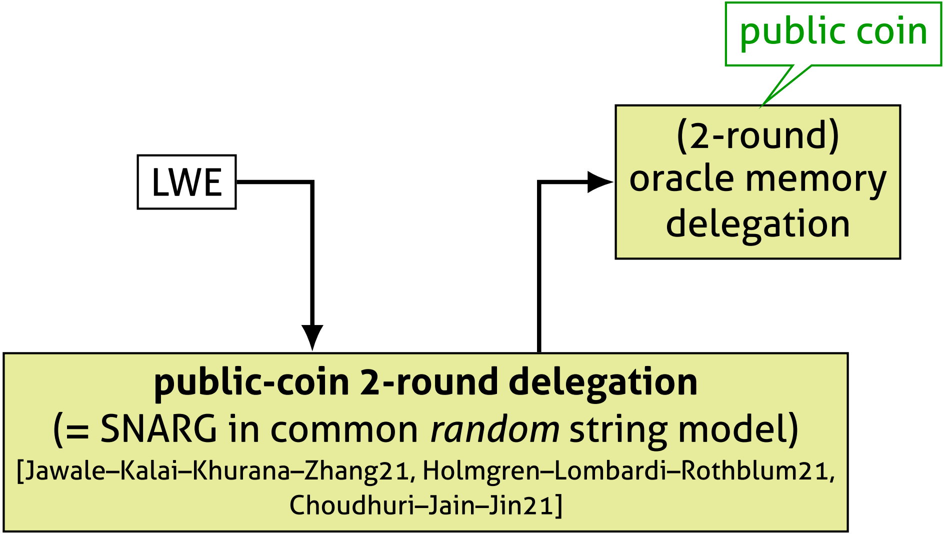


- **Efficiency:** V runs in polynomial time in the security parameter λ even for memory x of slightly super-poly length (e.g., $\lambda^{\log \log \lambda}$)
- **Soundness (intuition):** Once $\hat{X}$ is fixed, PPT malicious P can give an accepting proof π for at most a single y

Our goal



Our goal



NTTResearch

Building block #1

NTT Research

Building block #1

public-coin 2-round delegation of

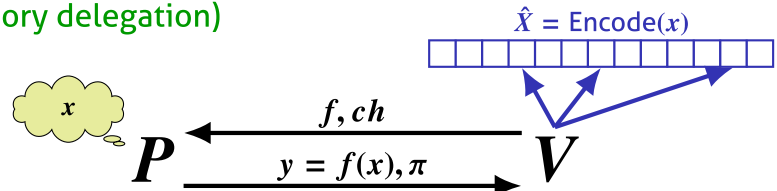
[Jawale–Kalai–Khurana–Zhang21 (JKKZ21), Holmgren–Lombardi–Rothblum21 (HLR21)]

Building block #1

public-coin 2-round delegation of

[Jawale–Kalai–Khurana–Zhang21 (JKKZ21), Holmgren–Lombardi–Rothblum21 (HLR21)]

- **Construction:** Fiat-Shamir + succinct proof of [Goldwasser–Kalai–Rothblum08]
- **Assumption:** Sub-exponential hardness of LWE
- **Key property:** V only needs to read a small part of an encoding of x (as in oracle memory delegation)



Building block #1

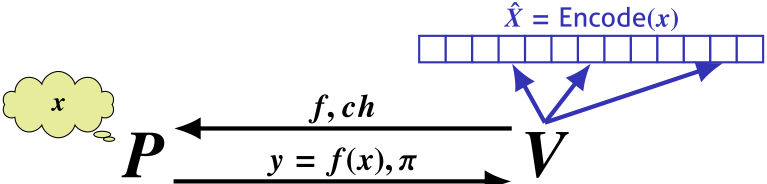
public-coin 2-round delegation of

[Jawale–Kalai–Khurana–Zhang21 (JKKZ21), Holmgren–Lombardi–Rothblum21 (HLR21)]

☺ can be converted to public-coin oracle memory delegation easily

☹ only works for a limited class of computations

- bounded-depth computations with a certain form of succinct descriptions



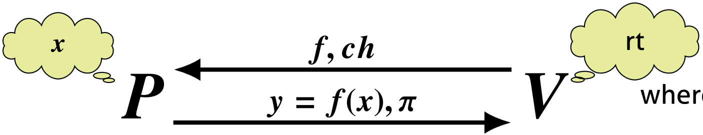
Building block #2

public-coin 2-round RAM delegation of [Choudhuri–Jain–Jin21 (CJJ21)]

Building block #2

public-coin 2-round RAM delegation of [Choudhuri–Jain–Jin21 (CJJ21)]

- **Assumption:** $\lambda^{\omega(1)}$-hardness of LWE for proofs about $\lambda^{\omega(1)}$-time computations
- **Key property:** V does not need to be have x in the clear (as in oracle memory delegation)



where rt = $\mathsf{MerkleHash}_h(x)$

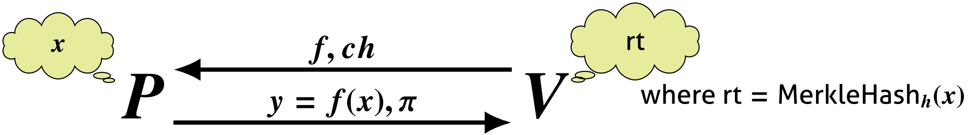
Building block #2

public-coin 2-round RAM delegation of [Choudhuri–Jain–Jin21 (CJJ21)]

☺ works for all $\lambda^{\omega(1)}$-time computations

☹ cannot be converted to oracle memory delegation easily

- How should V obtain Merkle hash of x in oracle memory delegation?



What should we do?

delegation of [JKKZ21,HLR21]

- ☺ can be converted to oracle memory delegation
- ☹ works for a limited class of computations (bounded-depth circuit w/ succinct descriptions)

RAM delegation of [CJJ21]

- ☺ works for all $\lambda^{\omega(1)}$-time computations
- ☹ cannot be converted to oracle memory delegation (V needs to have Merkle hash of memory x)

What should we do?

delegation of [JKKZ21,HLR21]

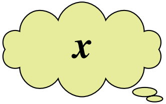
- ☺ can be converted to oracle memory delegation
- ☹ works for a limited class of computations (bounded-depth circuit w/ succinct descriptions)

RAM delegation of [CJJ21]

- ☺ works for all $\lambda^{\omega(1)}$-time computations
- ☹ cannot be converted to oracle memory delegation (V needs to have Merkle hash of memory x)

Let's combine these two!

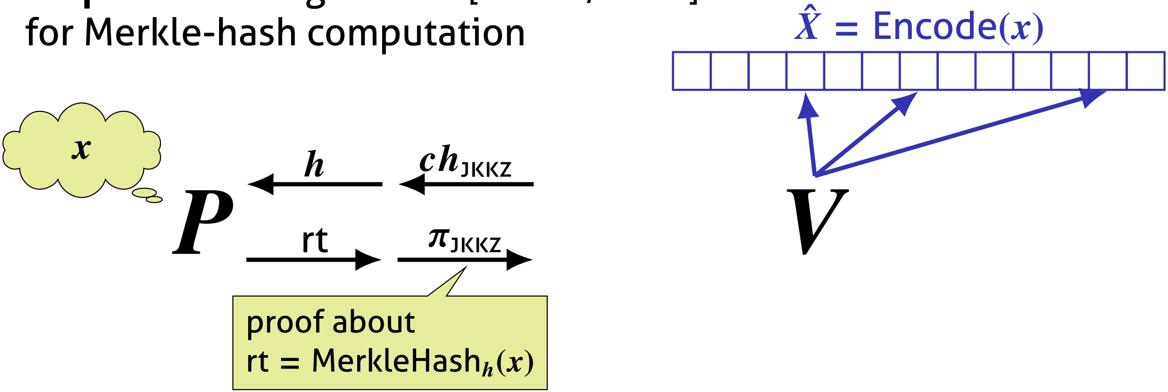
Our public-coin oracle memory delegation



P V

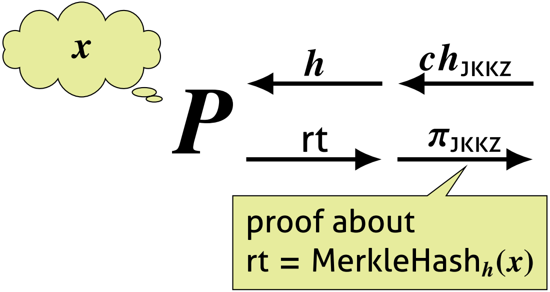
Our public-coin oracle memory delegation

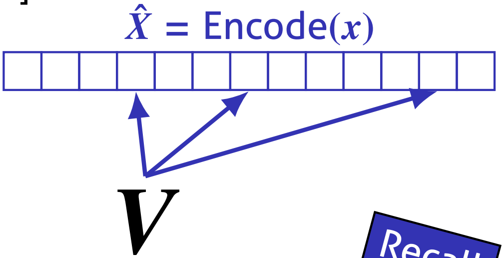
Step 1: use delegation of [JKKZ21,HLR21]
for Merkle-hash computation



Our public-coin oracle memory delegation

Step 1: use delegation of [JKKZ21,HLR21]
for Merkle-hash computation





Recall

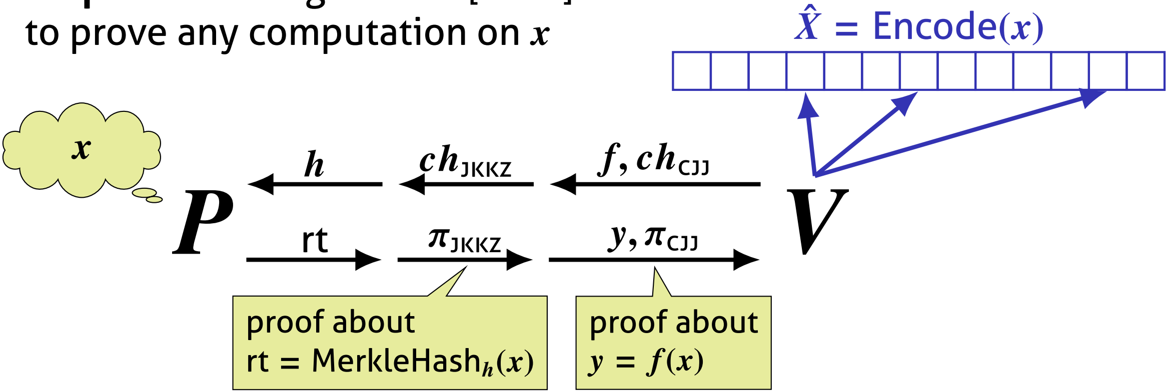
delegation of [JKKZ21,HLR21]

☺ can be converted to oracle memory delegation

☹ works for bounded-depth circuit w/ succinct descriptions

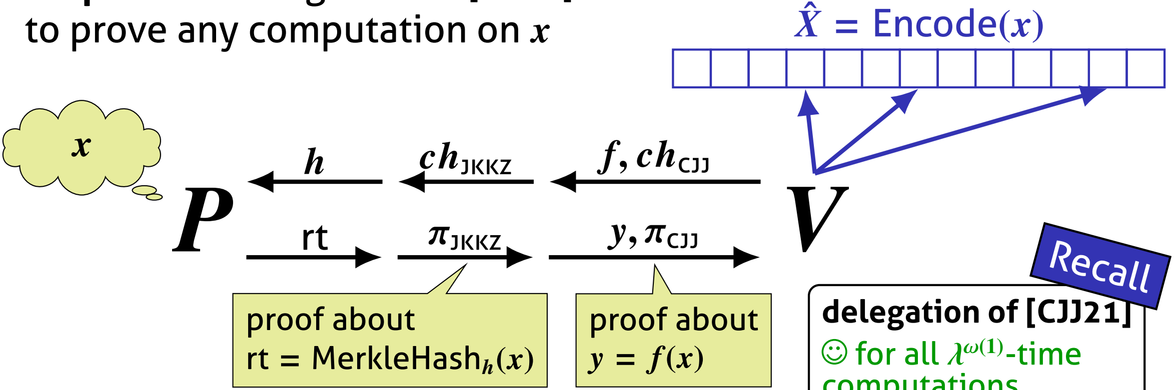
Our public-coin oracle memory delegation

Step 2: use delegation of [CJJ21] to prove any computation on x



Our public-coin oracle memory delegation

Step 2: use delegation of [CJJ21]
to prove any computation on x



delegation of [CJJ21]

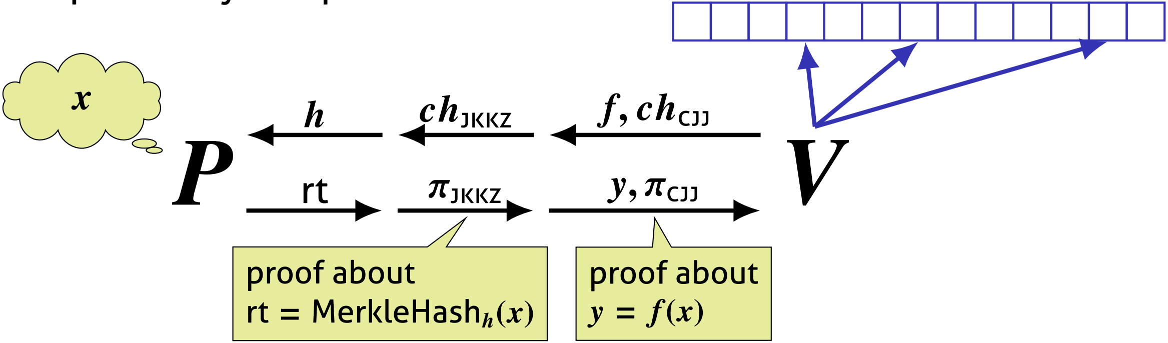
☺ for all $\lambda^{\omega(1)}$-time computations

☹ V needs to have Merkle hash of x

Our public-coin oracle memory delegation

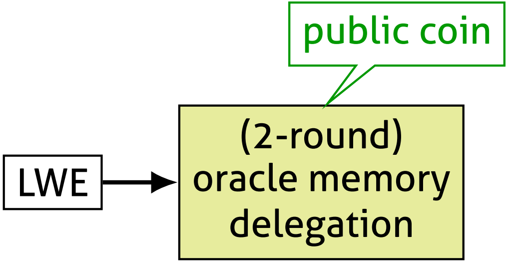
Step 2: use delegation of [CJJ21]
to prove any computation on x

$\hat{X}$ = Encode(x)

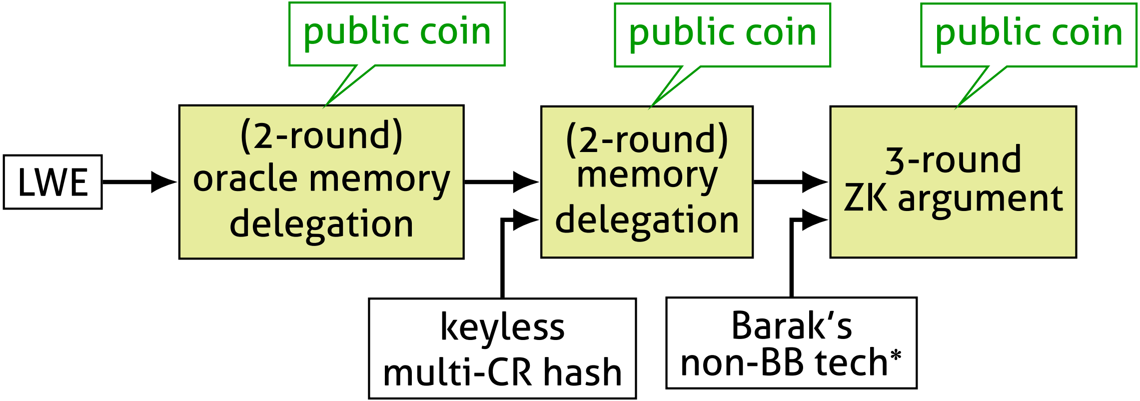


Soundness of π_{CJJ} holds since rt is proved to be correct!

Roadmap to public-coin 3-round ZK



Roadmap to public-coin 3-round ZK



* memory delegation is used as universal argument

Conclusion

Conclusion

Theorem

Public-coin 3-round ZK argument can be obtained by relying on:
(1) sub-exponential hardness of LWE, and
(2) slightly super-poly hardness of keyless multi-collision-resistant hash function

Public-coin 3-round ZK from simple & falsifiable assumptions!

Conclusion

Theorem

Public-coin 3-round ZK argument can be obtained by relying on:
(1) sub-exponential hardness of LWE, and
(2) slightly super-poly hardness of keyless multi-collision-resistant hash function

- **Open questions:**
  - from quasi-polynomial hardness of LWE?
  - from more standard assumptions (compared with keyless multi-CR hash)?

Conclusion

Theorem

Public-coin 3-round ZK argument can be obtained by relying on:
(1) sub-exponential hardness of LWE, and
(2) slightly super-poly hardness of keyless multi-collision-resistant hash function

- **Open questions:**
  - from quasi-polynomial hardness of LWE?
  - from more standard assumptions (compared with keyless multi-CR hash)?

Thank You!