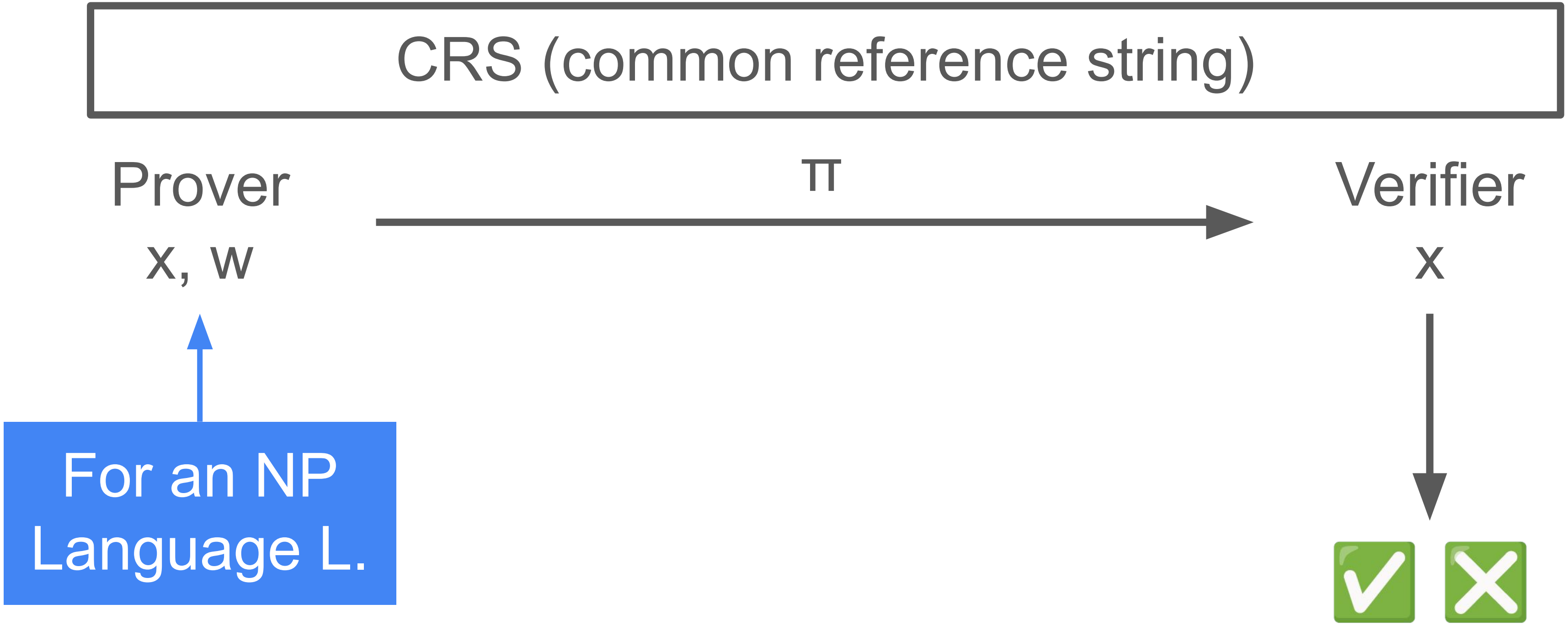
Batch Arguments to NIZKs from One-Way Functions

Eli Bradley
UT Austin

Brent Waters
UT Austin
NTT Research

David J. Wu
UT Austin

Non-Interactive Zero Knowledge Argument (NIZK) [GMR85, BFM88]



Non-Interactive Zero Knowledge Argument (NIZK) [GMR85, BFM88]

CRS (common reference string)

Prover $\xrightarrow{\pi}$ Verifier

x, w — x

✅ ❎

- Completeness: Honest proofs verify.
- Soundness: False statements don’t verify.
- Zero-Knowledge: Proofs can be simulated without witnesses.

Batch Argument (BARG) [CJJ21a]

CRS (common reference string)

Prover

$x_1, \ldots, x_m$

$w_1, \ldots, w_m$

π

Verifier

$x_1, \ldots, x_m$

✅ ❎

- Completeness
- Soundness
- Succinctness: $|\pi| \sim o(m) \cdot \text{poly}(\lambda, |x|)$.

Batch Argument (BARG) [CJJ21a]

CRS (common reference string)

Prover

$x_1, \ldots, x_m$

$w_1, \ldots, w_m$

π

Verifier

$x_1, \ldots, x_m$

Compressing proof loses information about witnesses.

- Completeness
- Soundness
- Succinctness: $|\pi| \sim o(m) \cdot \mathrm{poly}(\lambda, |x|)$.

Batch Argument (BARG) [CJJ21a]

CRS (common reference string)

Prover

$x_1, \ldots, x_m$

$w_1, \ldots, w_m$

π

Verifier

$x_1, \ldots, x_m$

✅ ❎

Existing constructions use NIZK techniques.
[CJJ21b, WW22, DGKV22, CGJ+23]

Can we get NIZKs from BARGs?

YES!

Previous and Concurrent Works

- [CW23]: NIZK from BARG, Local PRG, dual-mode commitment.
- [BKP$^+$23]: NIZK from BARG, dual-mode commitment.
    - Concurrent update to [BKP$^+$23]: NIZK from BARG, OWF.

Can we do it without additional assumptions?

Main Technical Result

(Adaptively-sound) BARG + OWF $\Rightarrow$ NIZK

Construct hidden-bits generator and apply [KMY20] to get NIZK.

Hidden-Bits Generator [FLS90, QRW19, KMY20]

Prover

$(r, st) \leftarrow \text{GenBits}$

$A = \{1, 4, 5\}$

$\pi \leftarrow \text{Prove}(st, A)$

r_1	r_2	r_3	r_4	r_5	r_6

$A, (r_1, r_4, r_5), \pi$

Verifier

✅ ❎

- Binding: r_A corresponds to $r \in \text{supp(GenBits)}$.
- Hiding: Hidden bits remain pseudorandom.
- Sparsity: Density of supp(GenBits) in $\{0, 1\}^m$ is low.

Our Construction

Our Construction: Overview

PRG + BARG $\Longrightarrow$ HBG

Why should this work?

Our Construction: Overview

Idea: Use **PRG** and **BARG** to build HBG.

- Binding ⇒ BARG
- Hiding ⇒ PRG

Our Construction: Overview

Idea: Use **PRG** and **BARG** to build HBG.

- Seed: PRG seed
- Output: PRG output
- Proof: BARG proof + openings

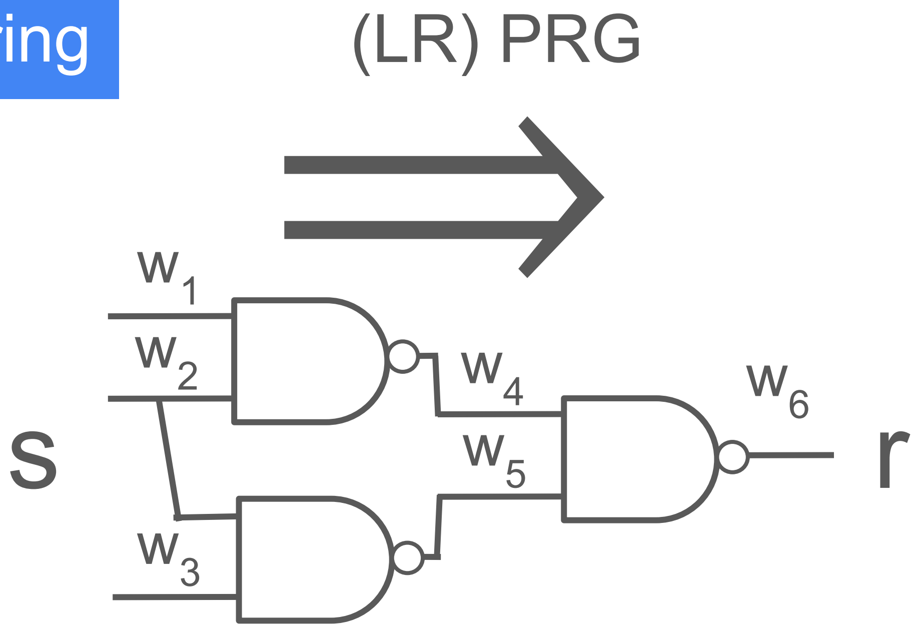
Template of [KMY20] for SNARGs, [CW23] for BARGs.

Our Construction: GenBits

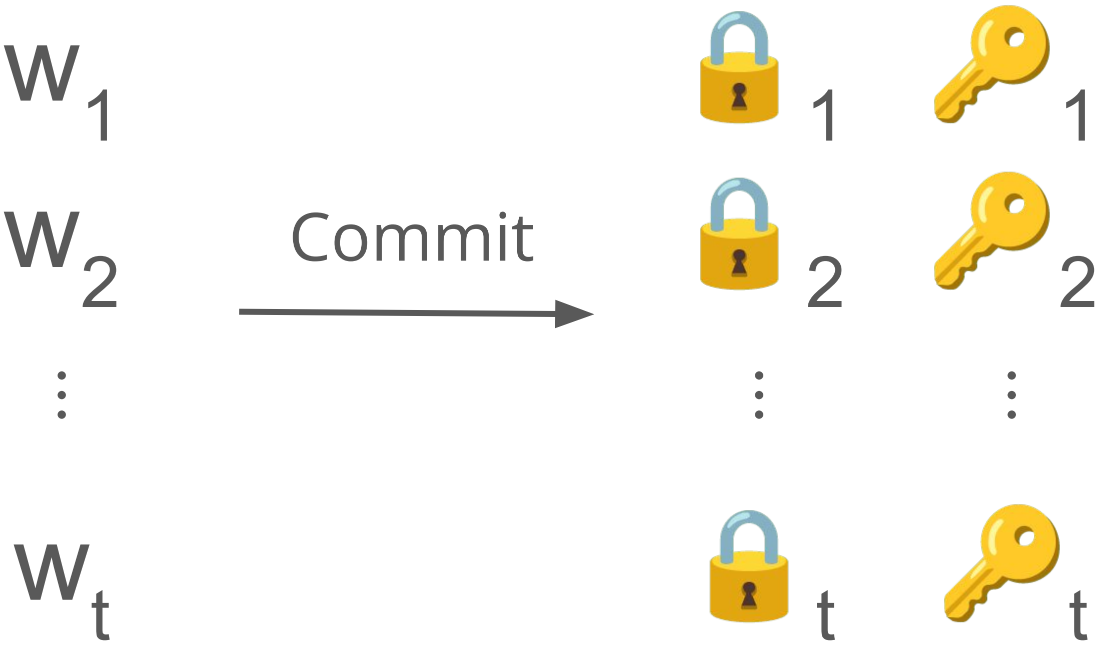
Hidden-bits string

- $s \leftarrow \{0, 1\}^n$
- $r = \mathrm{PRG}(s)$
- $st = s$

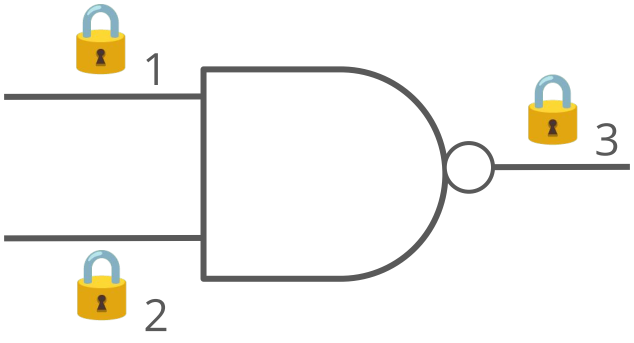
Prover's internal state



Our Construction: Prove



Our Construction: Prove



BARG over all internal gates.

Statement:

- crs_1, crs_2, crs_3
- 🔒$_1$ 🔒$_2$ 🔒$_3$

Witness:

- Wire values w_1, w_2, w_3
- 🔑$_1$ 🔑$_2$ 🔑$_3$

Check commitments and

$w_3 = \text{NAND}(w_1, w_2)$.

Our Construction: Prove

Output:

- Commitments 🔒$_1$ … 🔒$_t$.
- Openings for revealed output wires.
- BARG proof π_{BARG}.

Security: Binding

Binding: r_A corresponds to $r \in \text{supp}(\text{GenBits})$.

Security: Binding

Binding: r_A corresponds to r = PRG(s).

Check the PRG circuit is consistently evaluated.

Recall prover gives:

- Bit commitments to wire values.
- BARG proof each gate is correctly evaluated.
- Openings for output bits.

Security: Hiding

Proof π:

- Commitments 🔒$_1$ ... 🔒$_t$. ✅
- Openings for revealed output wires. ✅
- BARG proof π_{BARG}. ❓

Security: Hiding

Proof π:

- Commitments 🔒$_1$ … 🔒$_t$. ✅
- Openings for revealed output wires. ✅
- BARG proof π_{BARG}. ✅

PRG leakage-resilience

Summary

- We constructed a hidden-bits generator from:
  - Adaptively sound BARG
  - Leakage-resilient wPRF ($\Leftarrow$ OWF)
  - One-time dual-mode bit commitment ($\Leftarrow$ OWF)
- We get NIZK from the same assumptions [KMY20].

Additional Results

Stronger than OWF.

Somewhere-sound BARG + PKE ⇒ NIZK

Weaker than adaptive soundness.

Open Problems

Can we get NIZK with weaker assumptions?

- NIZK from index BARG [CJJ21b]?
- NIZK from non-adaptively sound BARG?

Thank you!

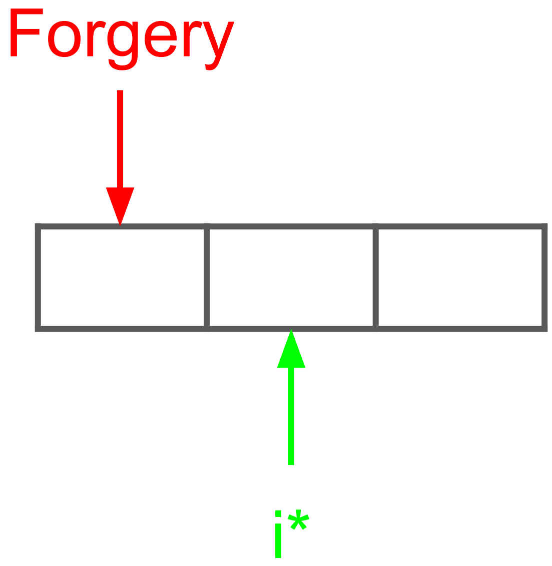
https://eprint.iacr.org/2023/1938.pdf

Getting Adaptively-Sound BARG

Somewhere-soundness

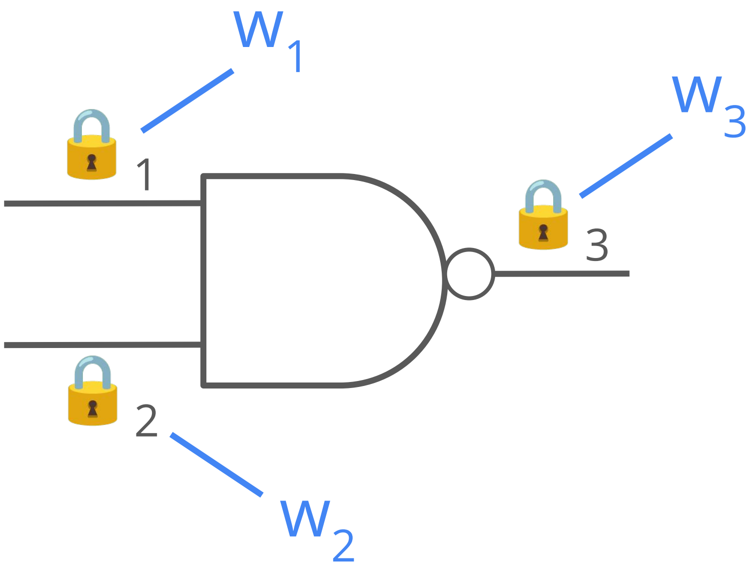
+ sub-exponential index hiding

⟹ Adaptive soundness.



Security: Binding

Binding: r_A corresponds to r = PRG(s).

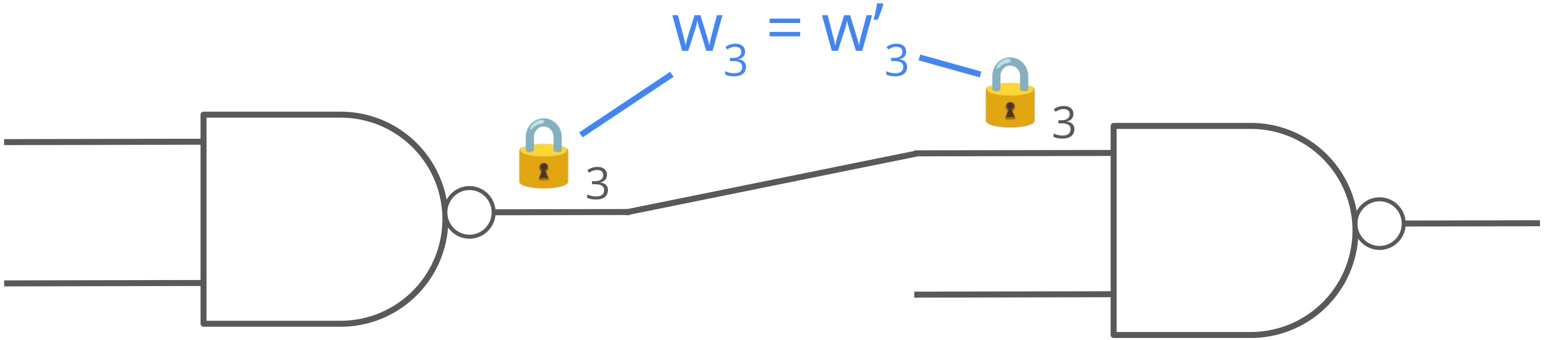


w_3 = NAND(w_1, w_2)

BARG soundness ⟹ gate consistency

Security: Binding

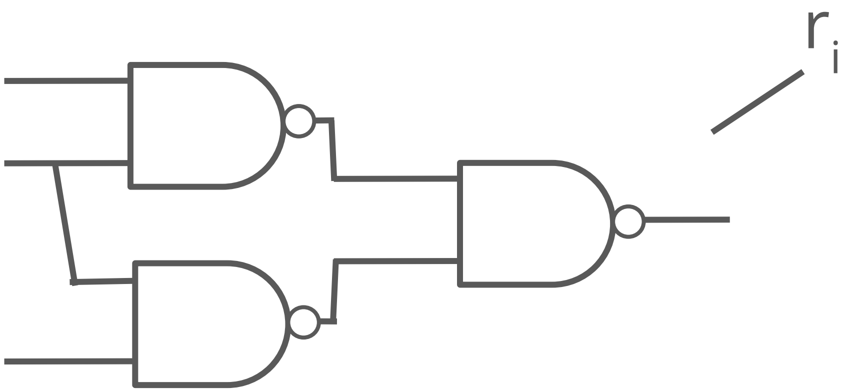
Binding: r_A corresponds to r = PRG(s).



BC statistical binding ⟹ wire consistency

Security: Binding

Binding: r_A corresponds to r = PRG(s).



Openings ⟹ output consistency

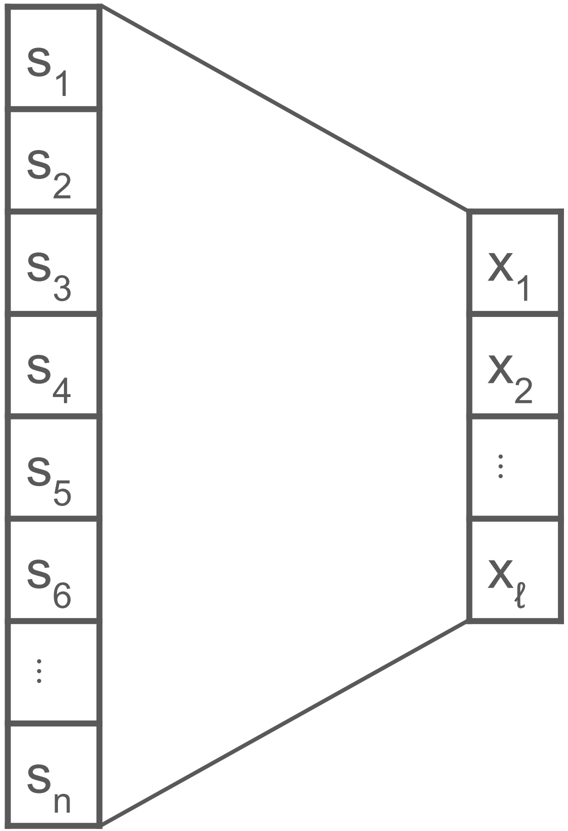
Leakage-Resilient PRG

$\forall$ leak: $\{0, 1\}^n \rightarrow \{0, 1\}^\ell$

(PRG(s), leak(s))

$\approx_c$

(Uniform, leak(s))



Leakage-Resilient PRG

$\forall$ leak: $\{0, 1\}^n \rightarrow \{0, 1\}^\ell$

(PRG(s), leak(s))

Leakage Resilient wPRF

(Uniform, leak(s))