Computational Fuzzy Extractors

Benjamin Fuller, Xianrui Meng, and Leonid Reyzin

BOSTON UNIVERSITY

December 2, 2013

Key Derivation from Noisy Sources

High-entropy sources are often noisy

- Source value *changes* over time, $w_0 \neq w_1$
- Assume a bound on distance: $d(w_0, w_1) \leq d_{\max}$
- Consider Hamming distance today

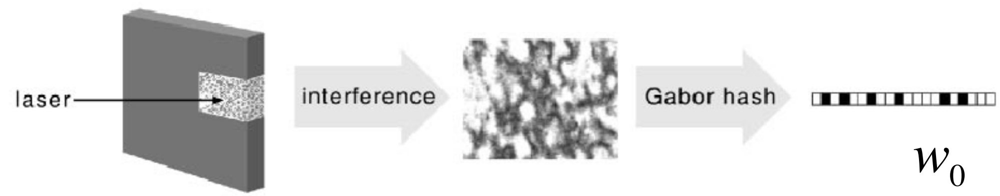
Want to derive a stable key from a noisy source

- Want w_0, w_1 to map to same key

Want the key to be *cryptographically* strong

- Appear uniform to the adversary

Physically Unclonable Functions (PUFs)
[PappuRechtTaylorGershenfeld02]



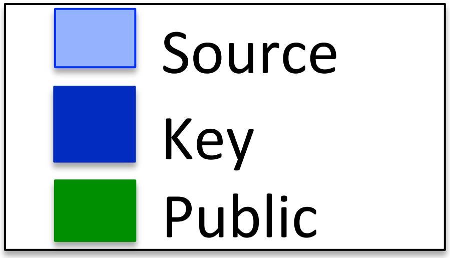
w_0

Biometric Data

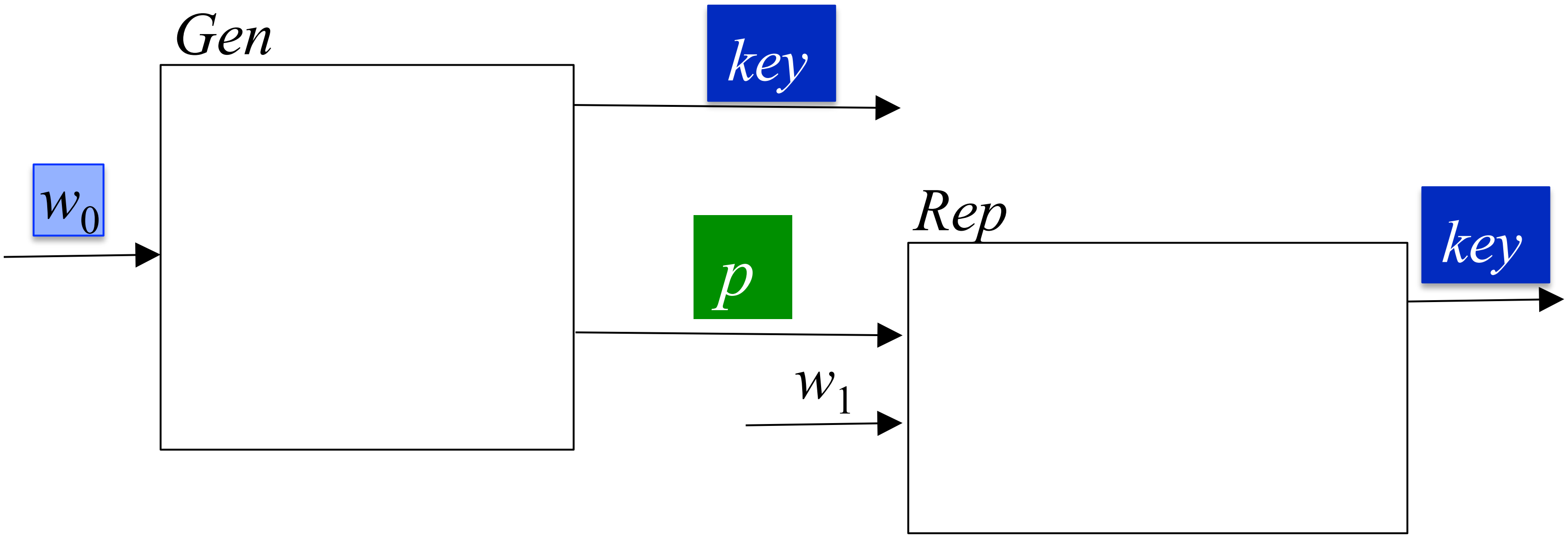
w_0

Goal of this talk: provide meaningful security for more sources

Fuzzy Extractors



- Assume source has min-entropy k (no w is likelier than 2^{-k})
- Lots of work on reliable keys from noisy data [BennettBrassardRobert85] … Our formalism: Fuzzy Extractors [DodisOstrovskyReyzinSmith04] …
- Correctness: Gen, Rep give same key if $d(w_0, w_1) < d_{\max}$
- Security: $(key\ , p) \approx (U\ , p)$

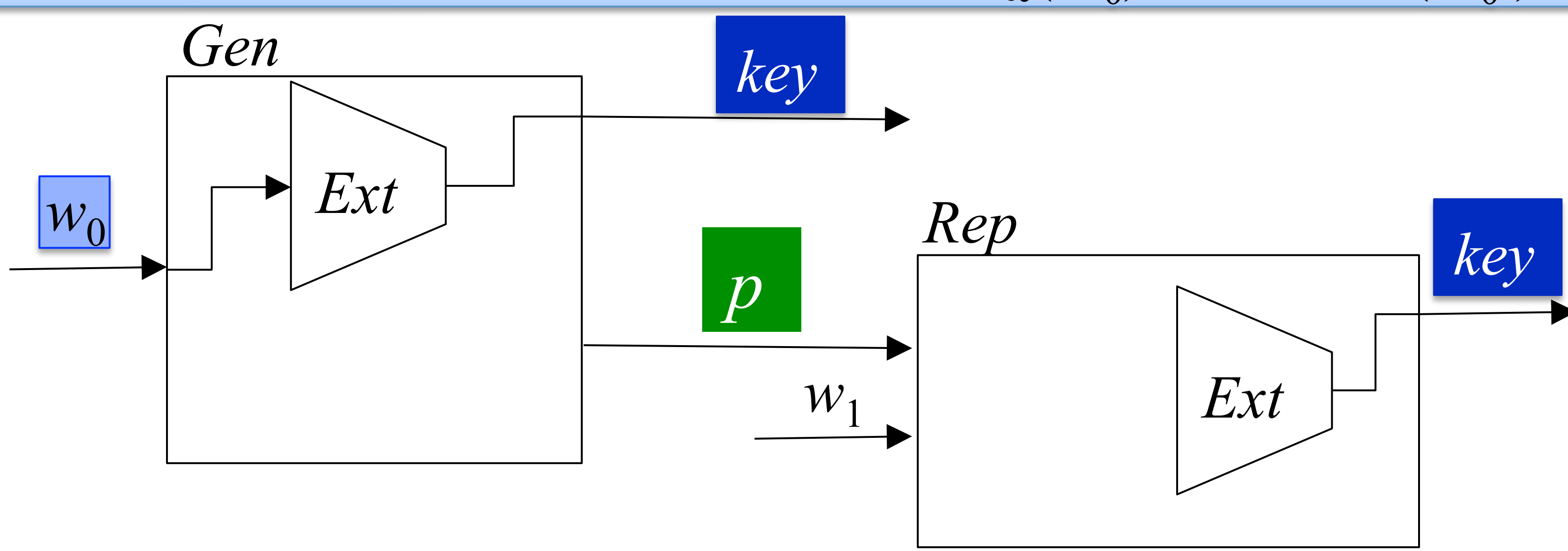


Fuzzy Extractors

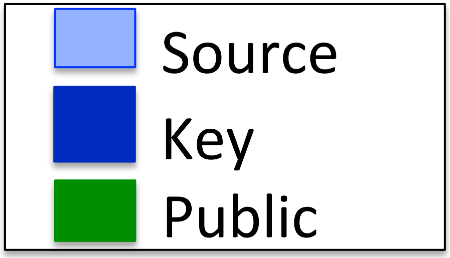
- Source
- Key
- Public

- Assume source has min-entropy k
  (no w is likelier than 2^{-k})
- Lots of work on reliable keys from noisy data [BennettBrassardRobert85] …
  Our formalism: Fuzzy Extractors [DodisOstrovskyReyzinSmith04] …
- Correctness: Gen, Rep give same key if $d(w_0, w_1) < d_{\max}$
- Security: $(key\ , p) \approx (U\ , p)$
- Typical Construction: - derive key using a randomness extractor

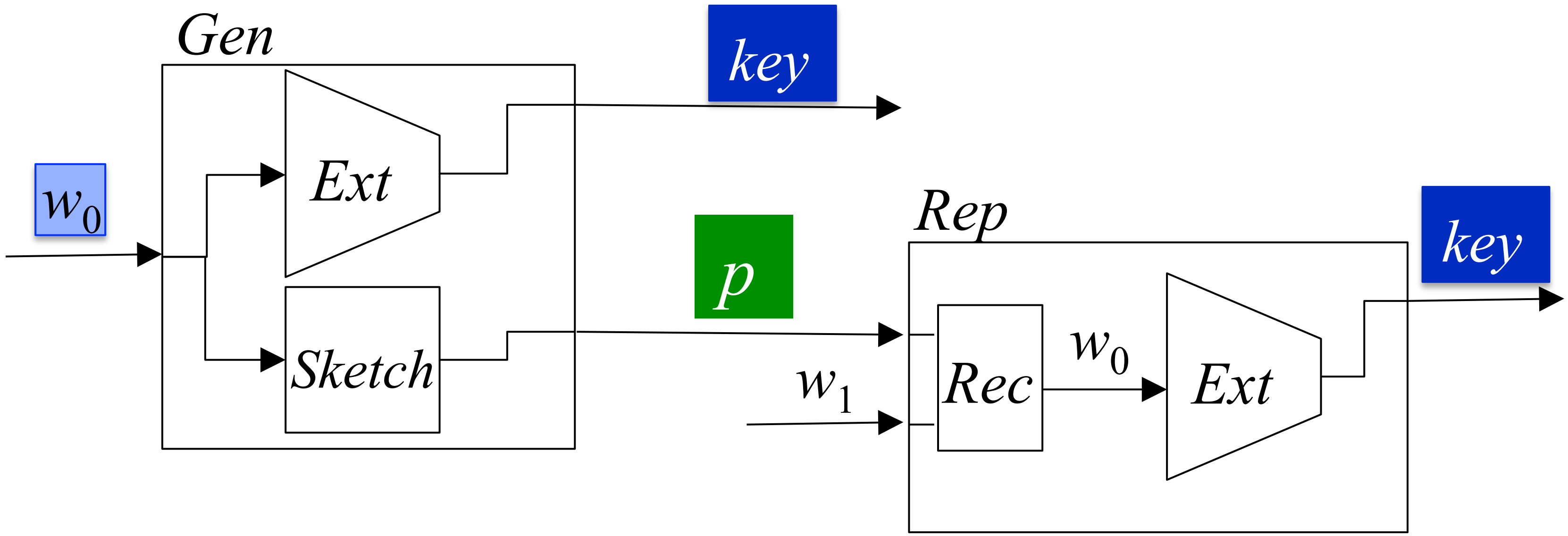
Converts high entropy sources to uniform: $H_\infty(W_0) \geq k \Rightarrow \mathrm{Ext}\,(W_0\,) \approx U$



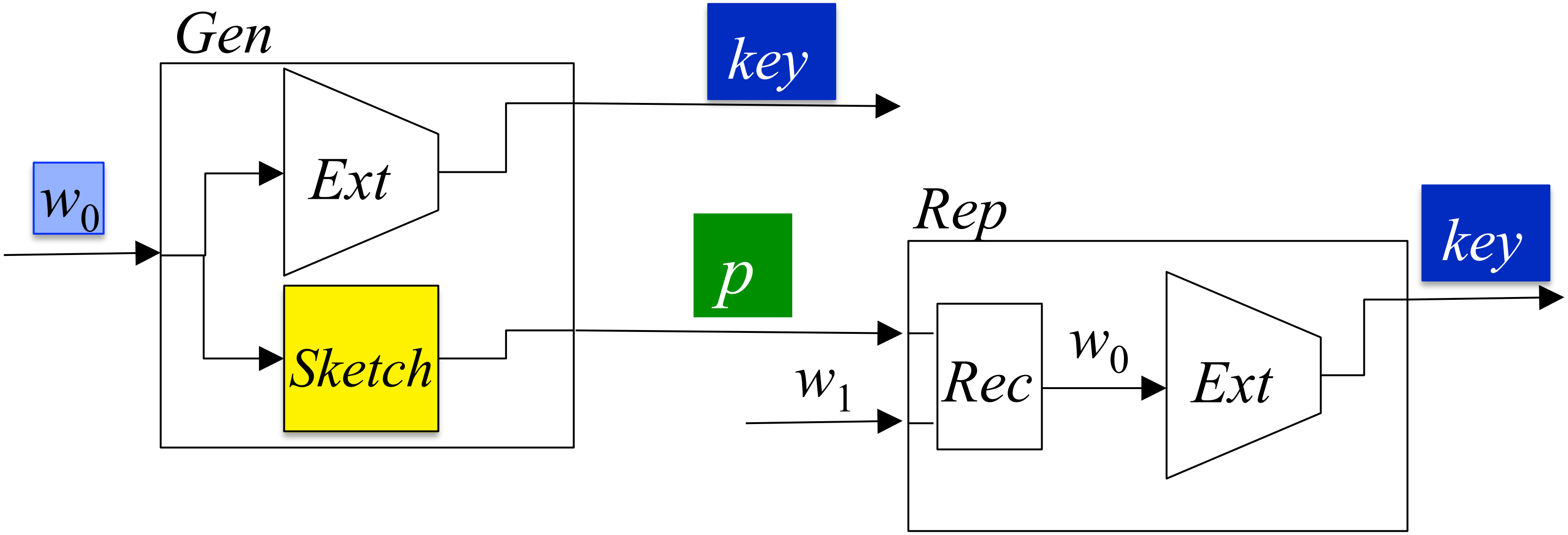
Fuzzy Extractors



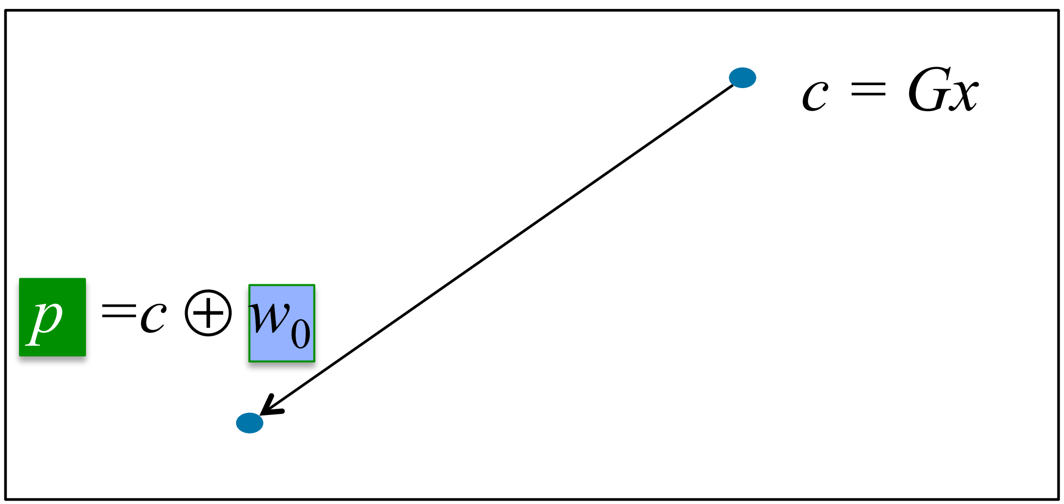
- Assume source has min-entropy k (no w is likelier than 2^{-k})
- Lots of work on reliable keys from noisy data [BennettBrassardRobert85] … Our formalism: Fuzzy Extractors [DodisOstrovskyReyzinSmith04] …
- Correctness: Gen, Rep give same key if $d(w_0, w_1) < d_{\max}$
- Security: $(key\ , p) \approx (U\ , p)$
- Typical Construction: - derive key using a randomness extractor
  - correct errors using a ***secure sketch***



Secure Sketches

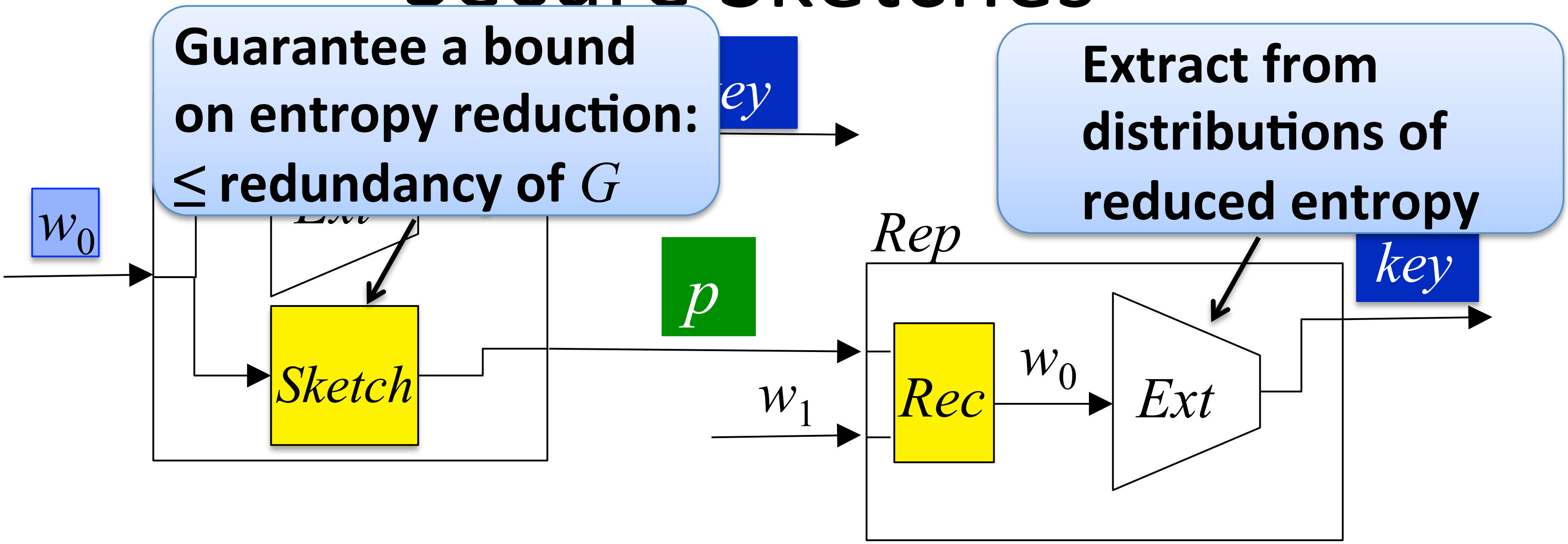


Code Offset Sketch



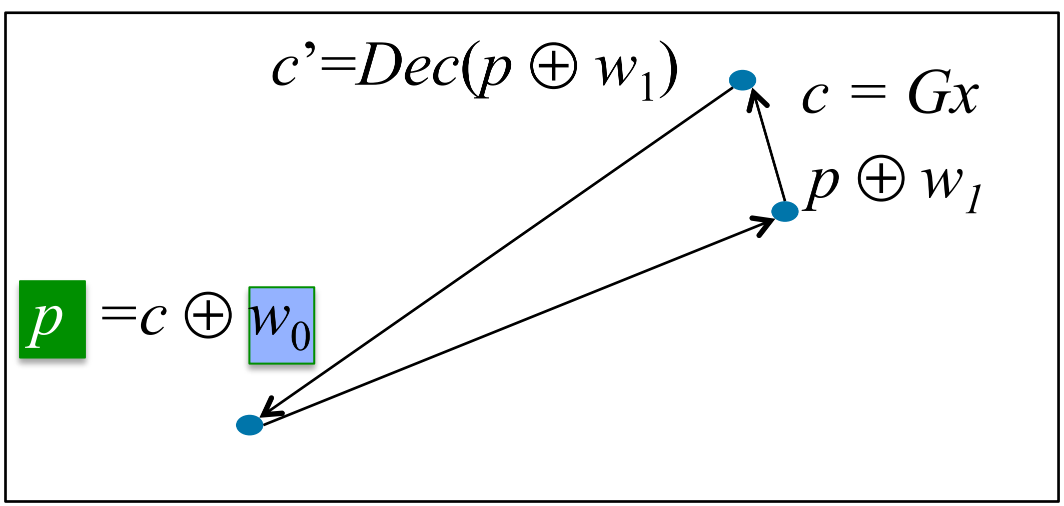
G generates a code that corrects $d_{\max}$ errors

Secure Sketches



Code Offset Sketch

G generates a code that corrects $d_{\max}$ errors



If w_0 and w_1 are close $c' = c$

$\Downarrow$

$w_0 = c' \oplus p$

p reveals information about w_0

Entropy Loss From Fuzzy Extractors

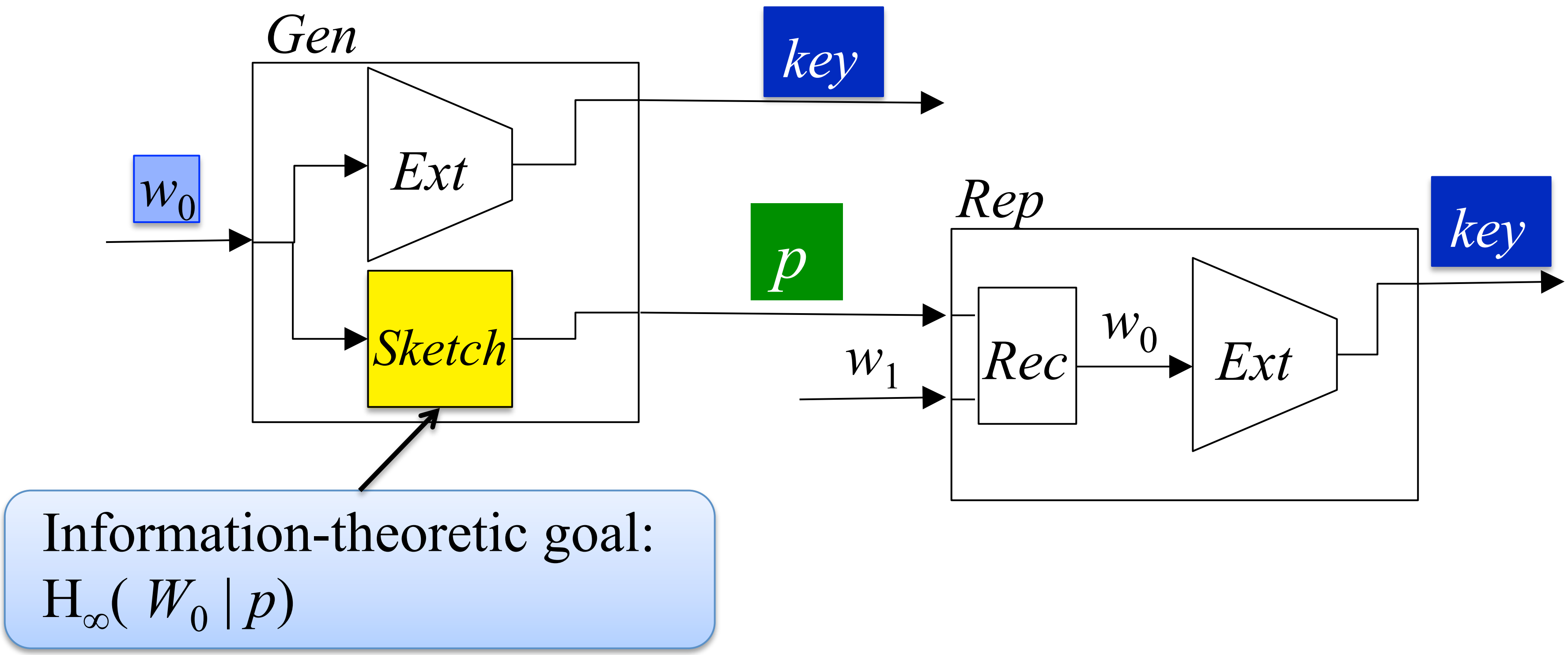
- Entropy is at a premium for physical sources
  - Iris ≈249 [Daugman1996]
  - Fingerprint ≈82 [RathaConnellBolle2001]
  - Passwords ≈31 [ShayKomanduri+2010]
- Above construction of fuzzy extractors, with standard analysis:
  - Secure sketch loss $=$ redundancy of code $\geq$ error correcting capability
    Loss necessary for information-theoretic sketch: [Smith07, DORS08]
  - Randomness extractor loss $\geq 2\log(1/\varepsilon)$
- Can we improve on this?
- One approach: define secure sketches/fuzzy extractors <u>computationally</u>
  - Give up on security against all-powerful adversaries, consider computational ones

Can we do better in computational setting?

Our Results:

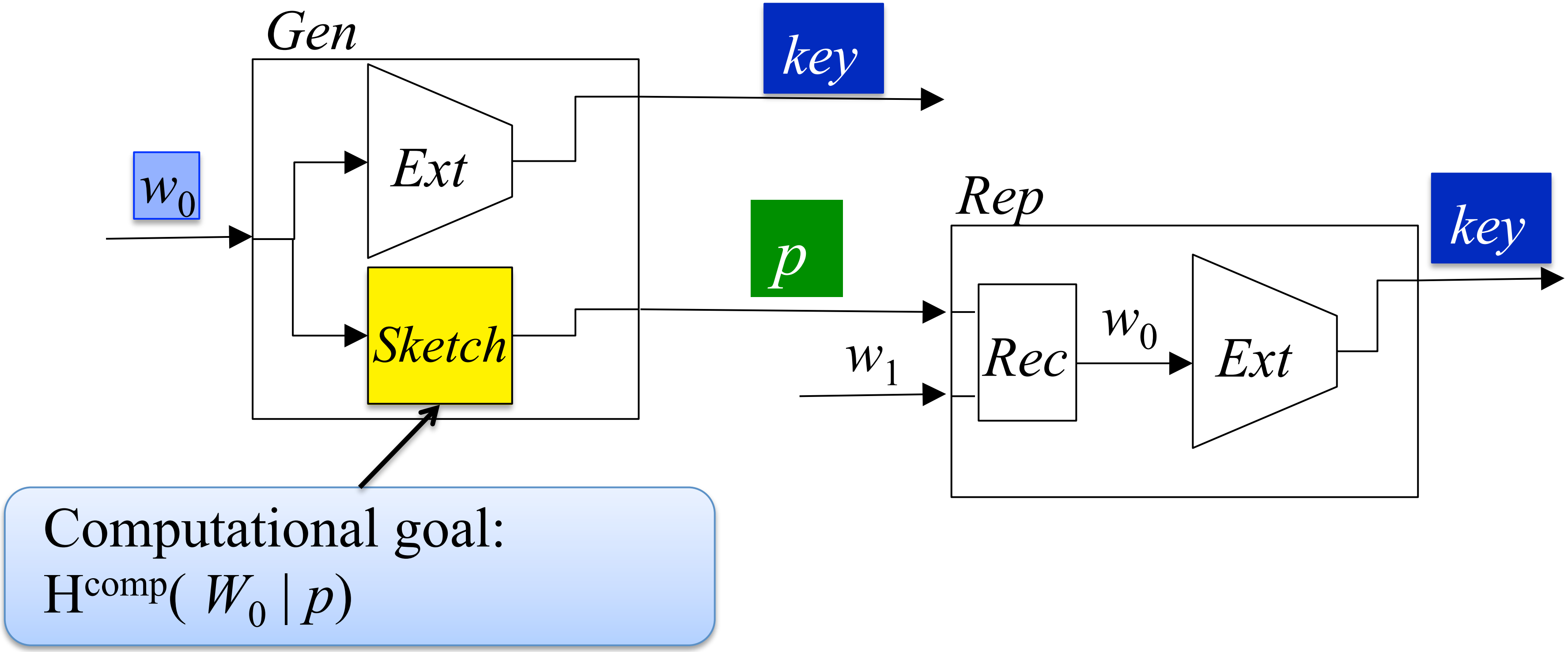
- For secure sketches: NO
  - We show that defining a secure sketch in computational setting does not improve entropy loss
- For fuzzy extractors: YES
  - We construct a *lossless* computational Fuzzy Extractor based on the Learning with Errors (LWE) problem
  - Caveat: this result shows only feasibility of a different construction and analysis; we do not claim to have a specific set of parameters for beating the traditional construction

Computational Secure Sketches



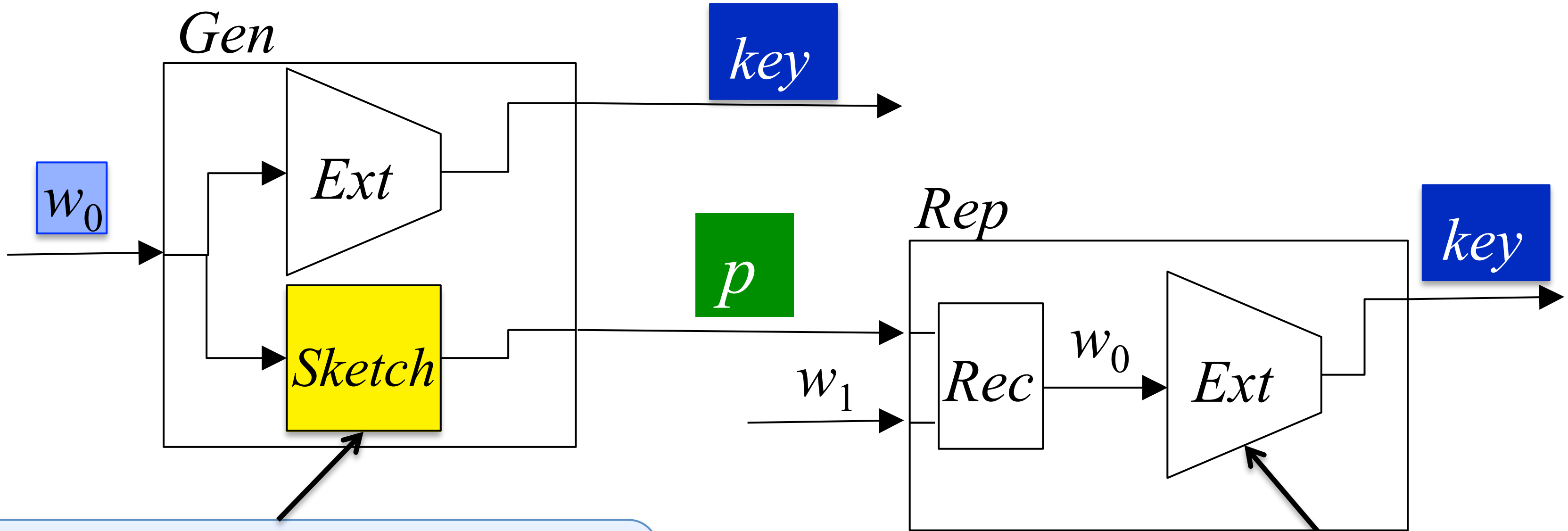
- Can we improve on this computationally?

Computational Secure Sketches



- Can we improve on this computationally?
- What does $\mathrm{H}^{\mathrm{comp}}(W_0 \mid p)$ mean?
- Most natural requirement:
  $(W_0 \mid p)$ is indistinguishable from $(Y \mid p)$ and $H_\infty(Y \mid p) \geq k$
- Known as HILL entropy [HåstadImpagliazzoLevinLuby99]

Computational Secure Sketches



Computational goal:
$H^{HILL}(W_0 \mid p)$

Good News:
Extractors yield pseudorandom keys from HILL entropy

- Can we improve on this computatio
- What does $H^{comp}(W_0 \mid p)$ mean?
- Most natural requirement:
  $(W_0 \mid p)$ is indistinguishable from $(Y \mid p)$ and $H_\infty(Y \mid p) \geq k$
- Known as HILL entropy [HåstadImpagliazzoLevinLuby99]

HILL Secure Sketches $\Rightarrow$ Secure Sketches

Our Theorem:

If $H^{\text{HILL}}(W_0 \mid p) \geq k$, then

there exists an error-correcting code C with 2^{k-2} points and

Rec corrects d_{max} random errors on C

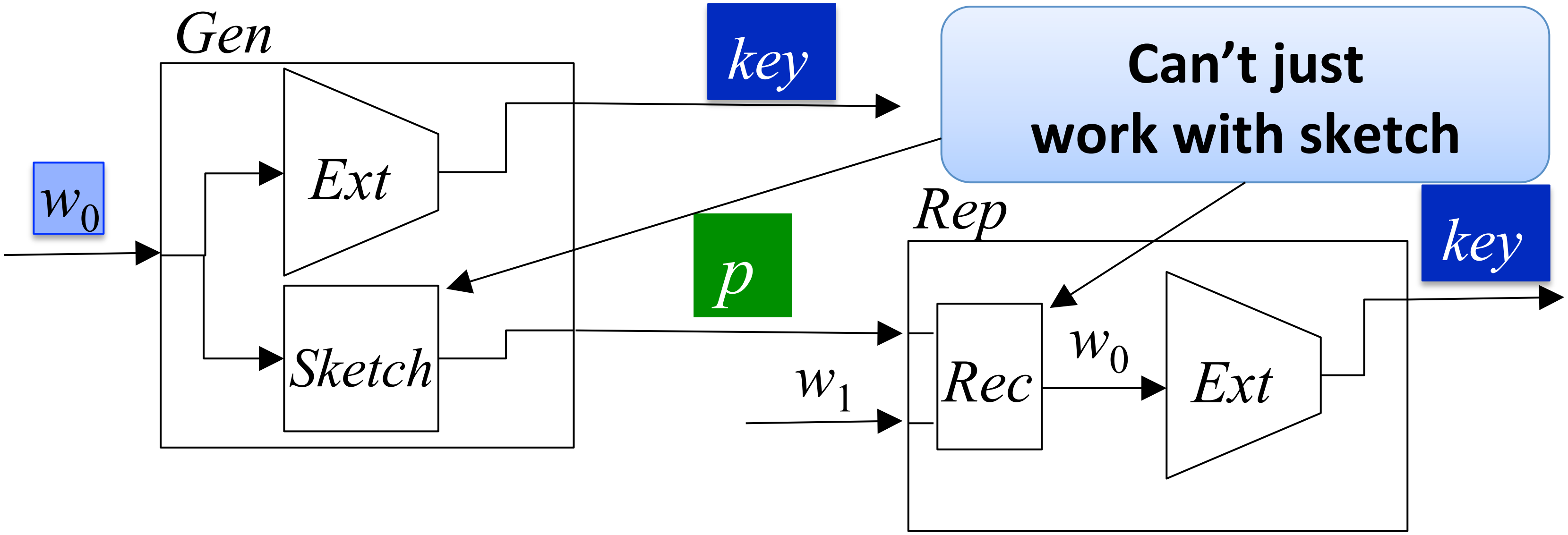
We can fix a p value where Rec functions as a good decoder for W_0.
Rec must also decode on indistinguishable distribution Y, and Y is large.

Corollary: (Using secure sketch of [Smith07])
If there exists a sketch with HILL entropy k,
then there exists a sketch with true entropy $k-2$.

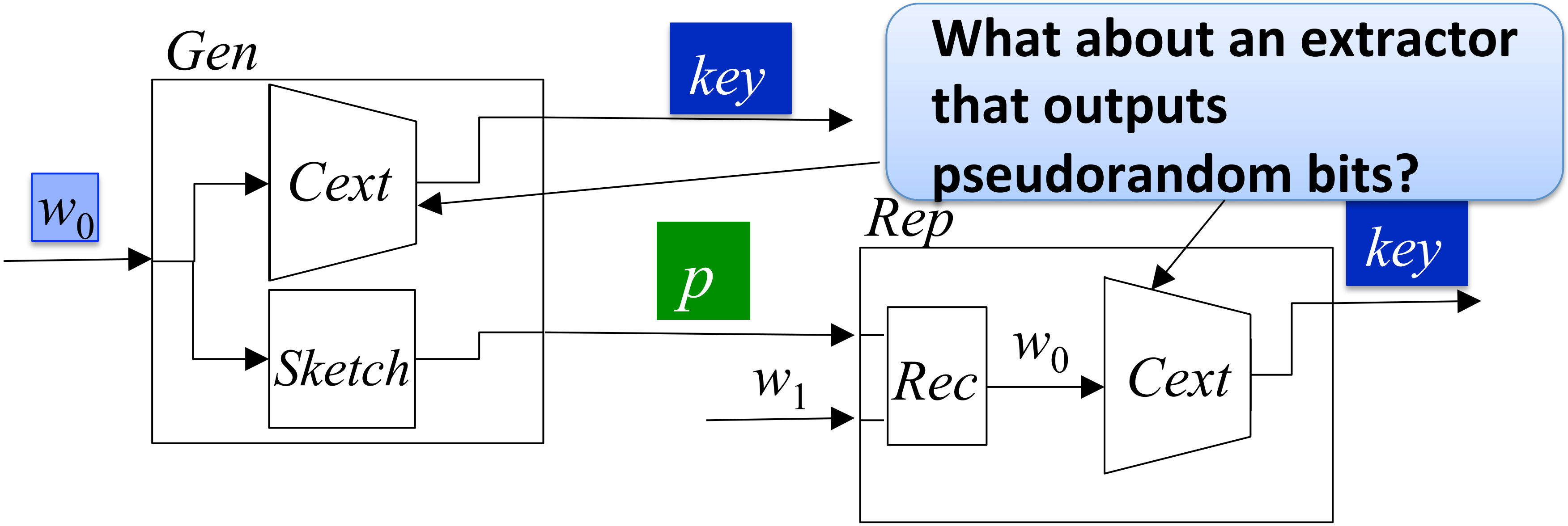
Can we do better in computational setting?

- For secure sketches: NO
  - A sketch that retains HILL entropy implies an information theoretic sketch
- For fuzzy extractors: YES
  - Can't just make the sketch "computational"
  - Other approaches?

Building a Computational Fuzzy Extractor

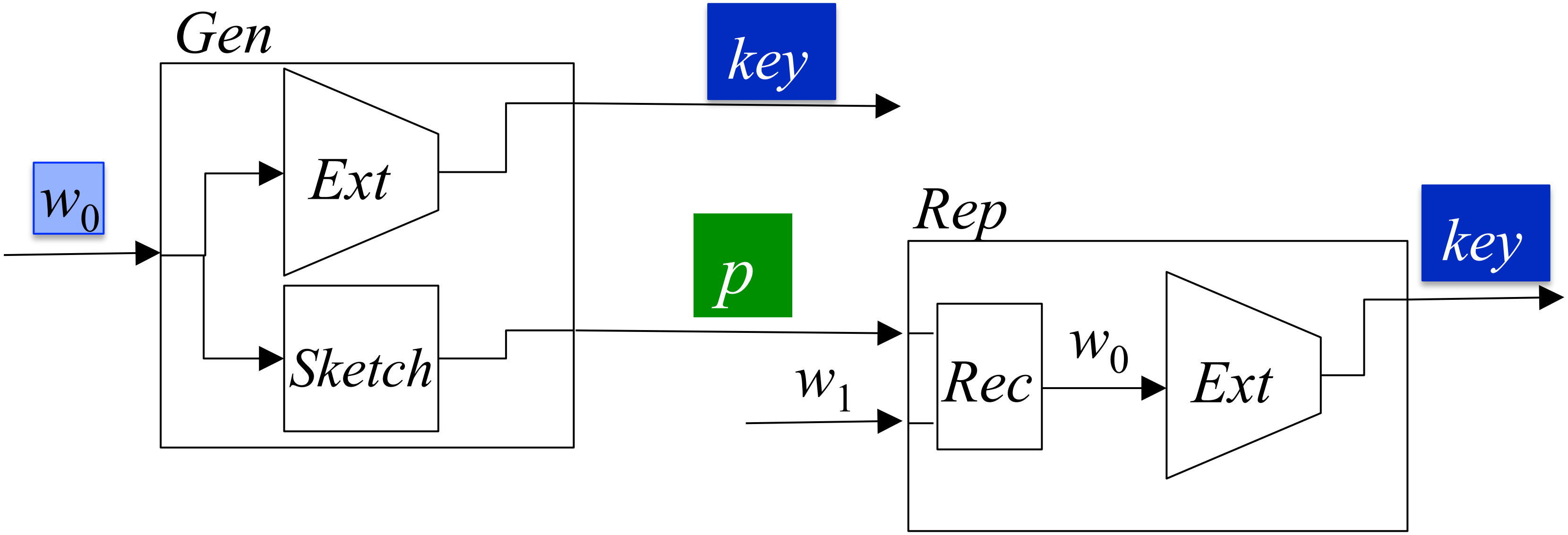


Building a Computational Fuzzy Extractor

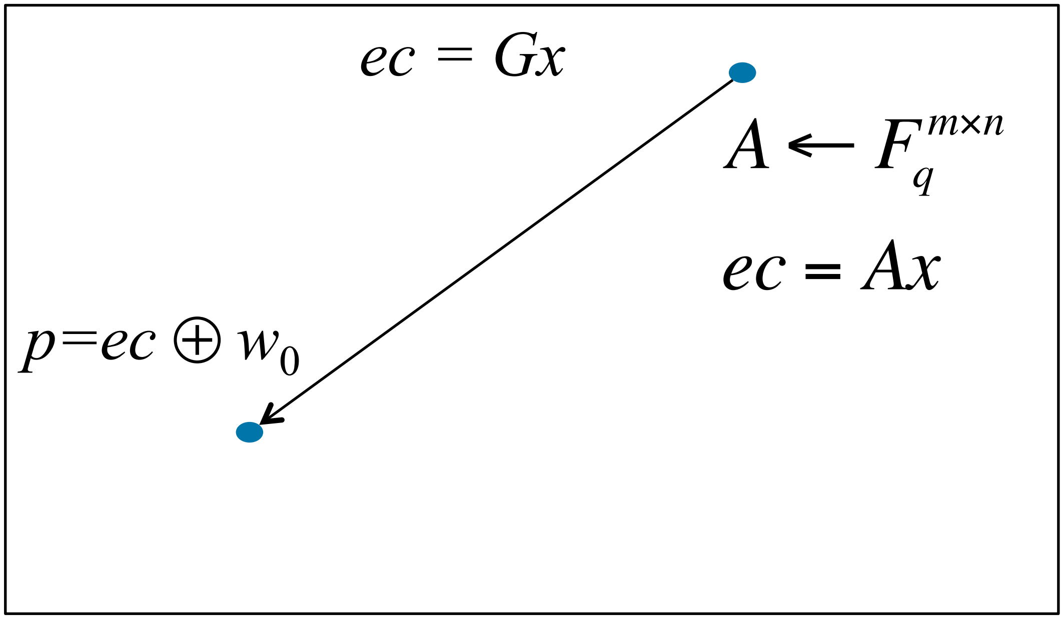


- Computational extractors convert high-entropy sources to pseudorandom bits [Krawczyk10]
- Natural construction: $Cext(w_0) = \mathrm{PRG}(Ext(w_0))$
- Extensions [DachmanSoledGennaroKrawczykMalkin12DodisYu13DodisPietrzakWichs13]
- All require enough residual entropy after Sketch to run crypto!
  - See [DachmanSoledGennaroKrawczykMalkin12] for conditions

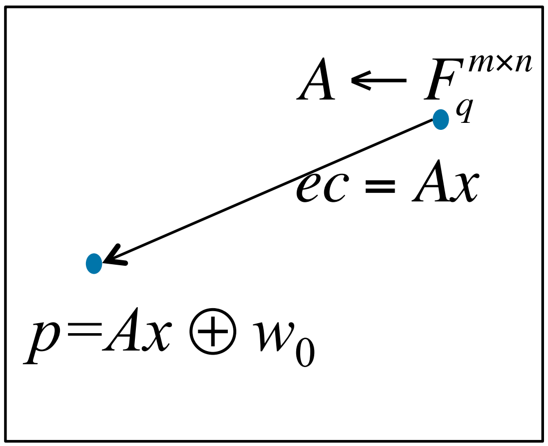
Building a Computational Fuzzy Extractor

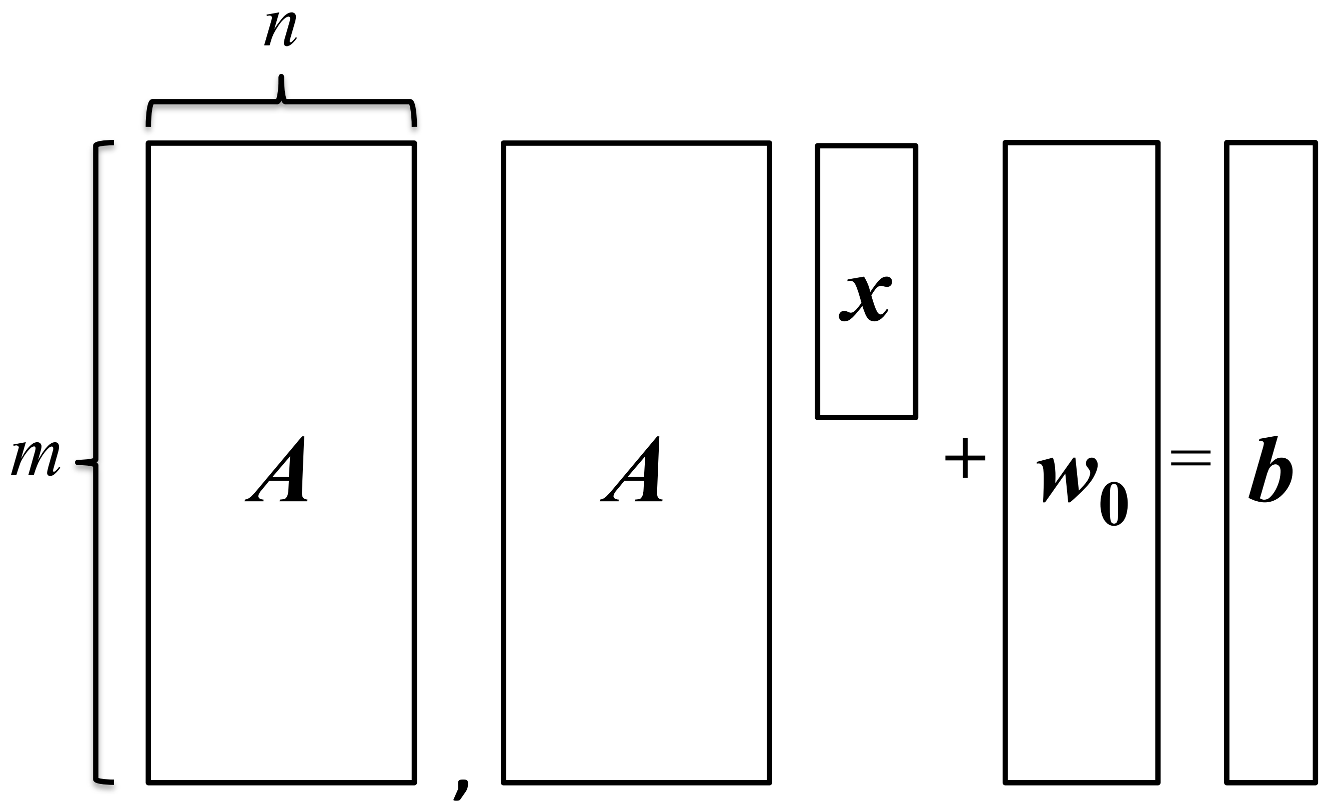


- We'll try to combine a sketch and an extractor
- We'll base our construction on the code offset sketch
- Instantiate with random linear code
- Base security on Learning with Errors (LWE)



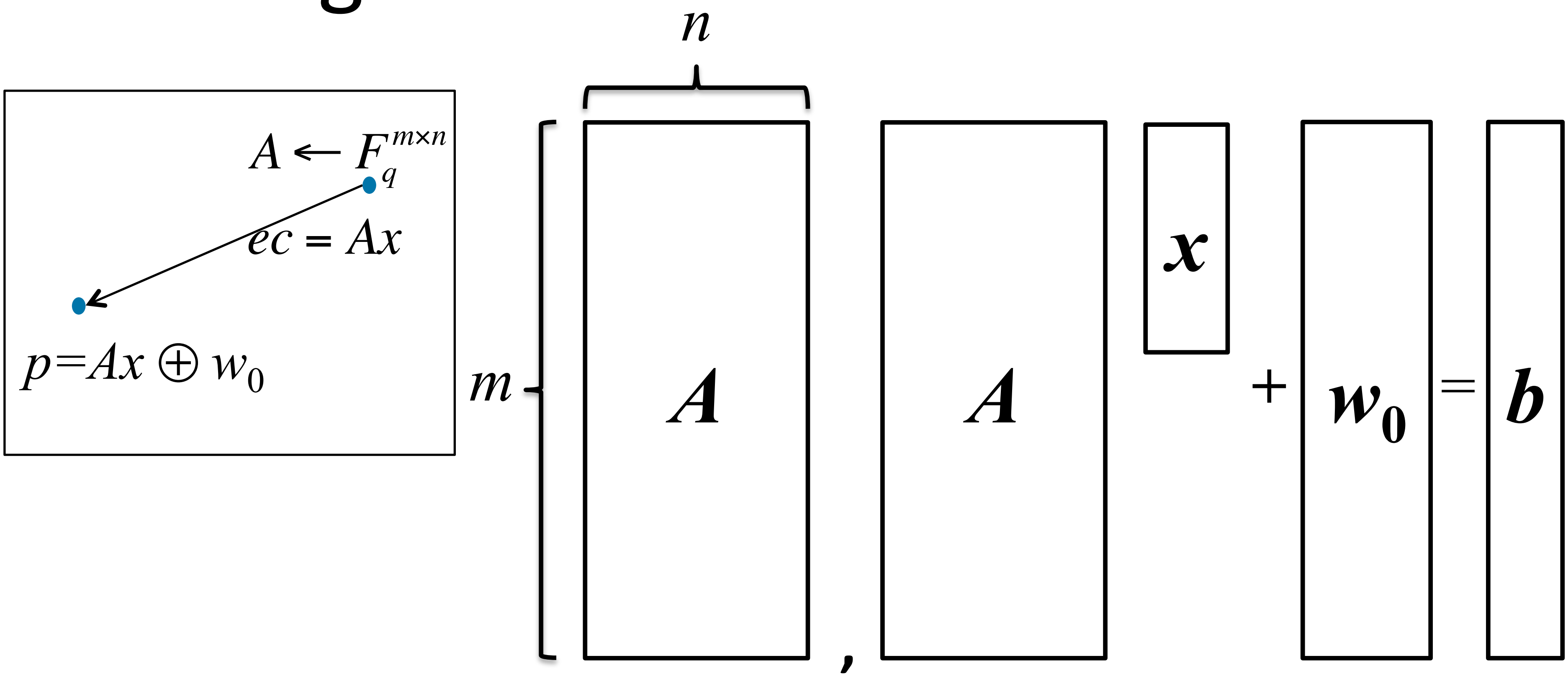
Learning with Errors





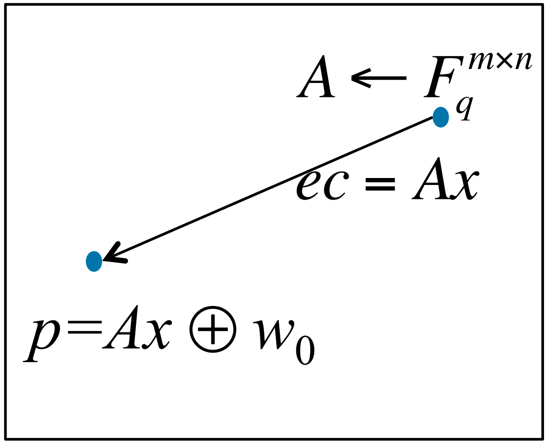
- Recovering x is known as learning with errors
- [Regev05] shows solving LWE implies approximating lattice problems
- LWE Error Distribution = Source Distribution W_0
  - Need error distribution where LWE is hard
  - Start from result of [Döttling&Müller-Quade13] and make some progress

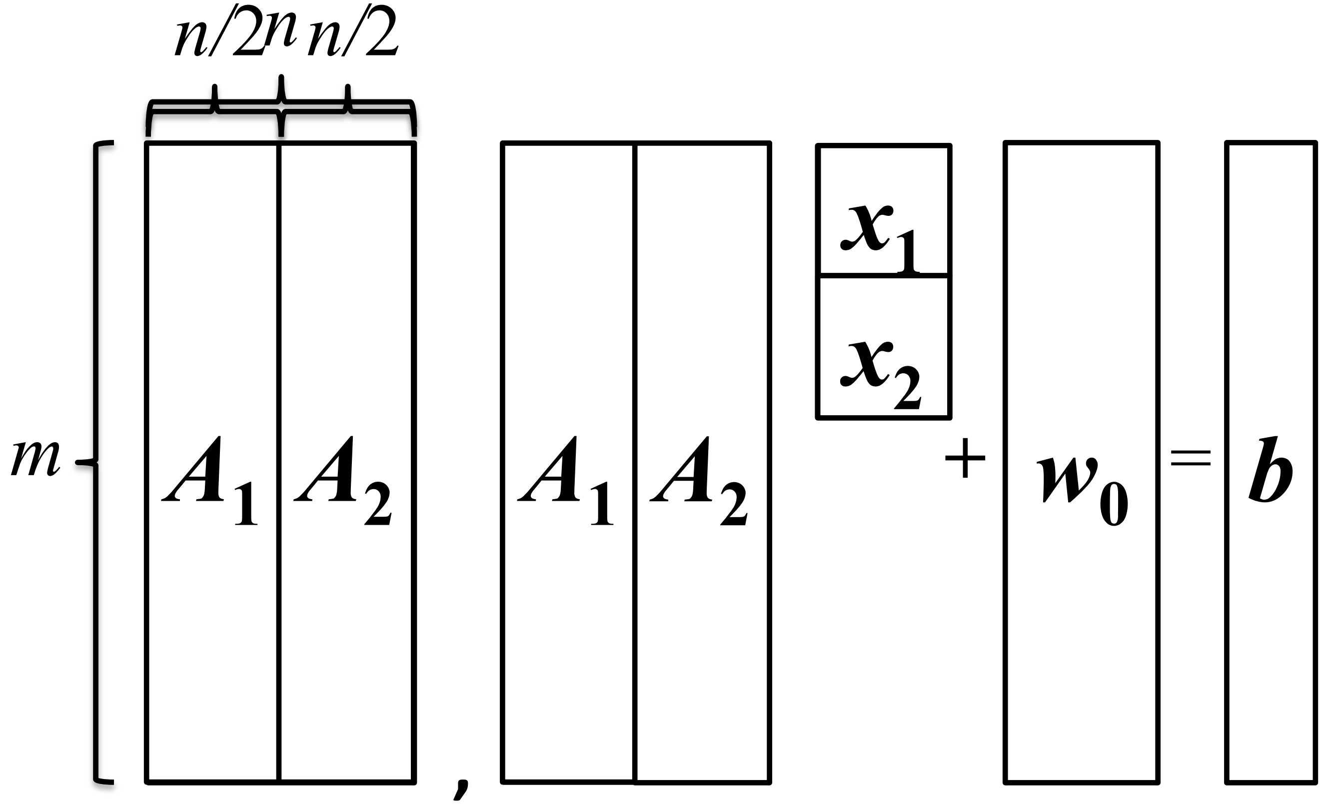
Learning with Errors



- Recovering x is known as learning with errors
- [Regev05] shows solving LWE implies approximating lattice problems
- LWE Error Distribution = Source Distribution W_0
- [AkaviaGoldwasserVaiku…09] show if LWE is secure on $n/2$ variables, any additional variables are hardcore

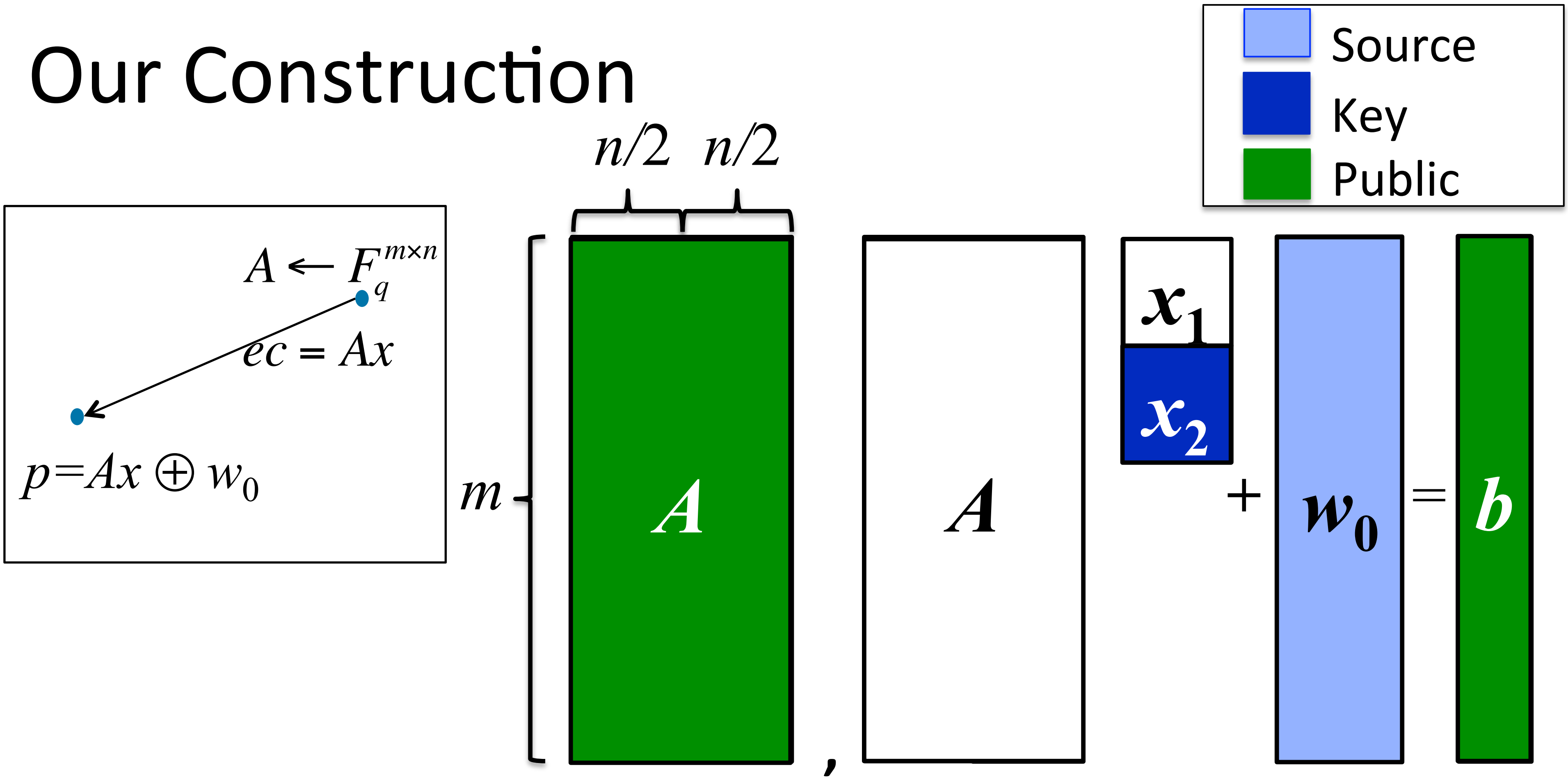
Learning with Errors





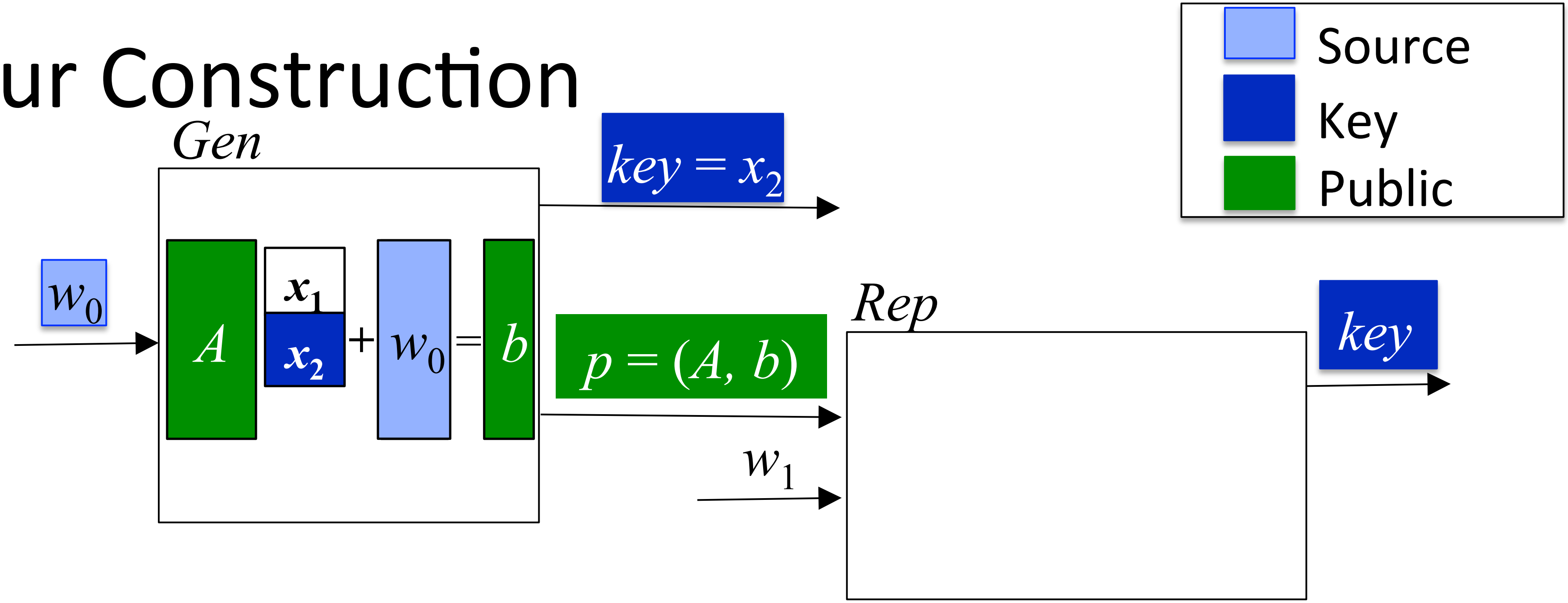
- Recovering x is known as learning with errors
- [Regev05] shows solving LWE implies approximating lattice problems
- LWE Error Distribution = Source Distribution W_0
- [AkaviaGoldwasserVaiku…09] show if LWE is secure on $n/2$ variables, any additional variables are hardcore $x_2 \mid A, b$ is pseudorandom

Our Construction



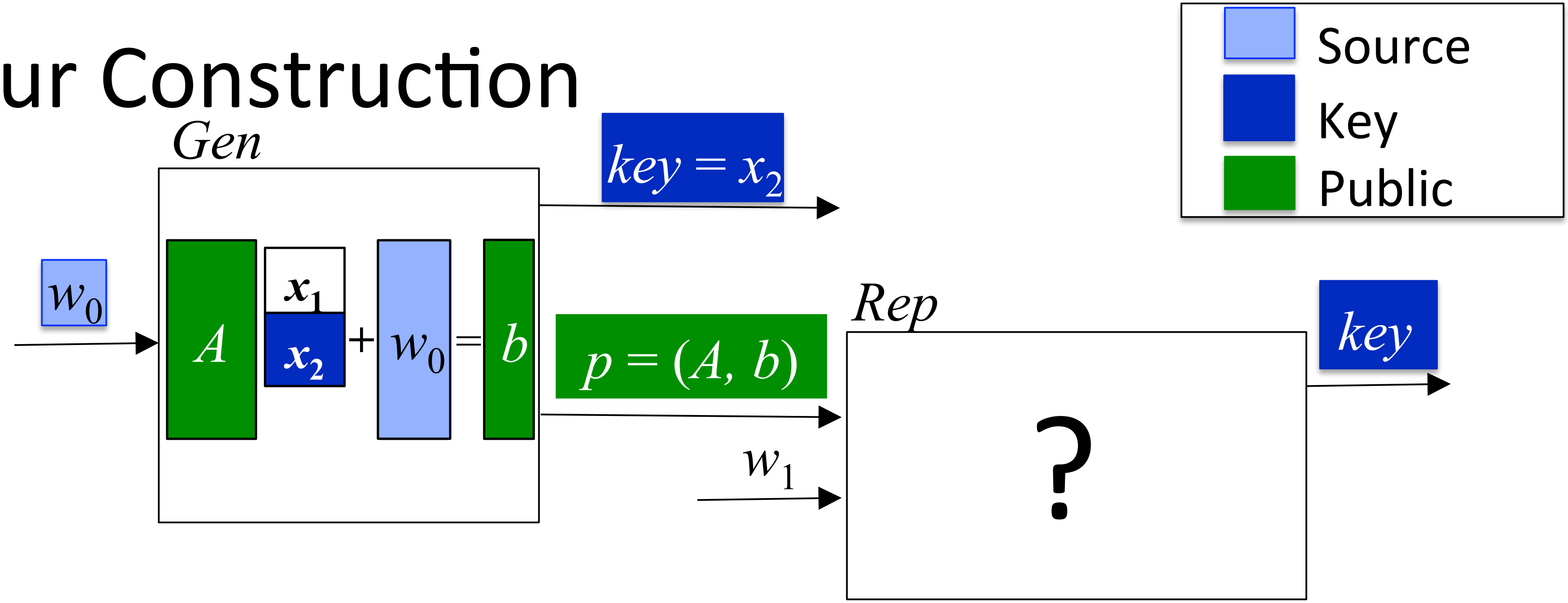
- Recovering x is known as learning with errors
- [Regev05] shows solving LWE implies approximating lattice problems
- LWE Error Distribution = Source Distribution W_0
- [AkaviaGoldwasserVaiku…09] show if LWE is secure on $n/2$ variables, any additional variables are hardcore $x_2 \mid A, b$ is pseudorandom

Our Construction

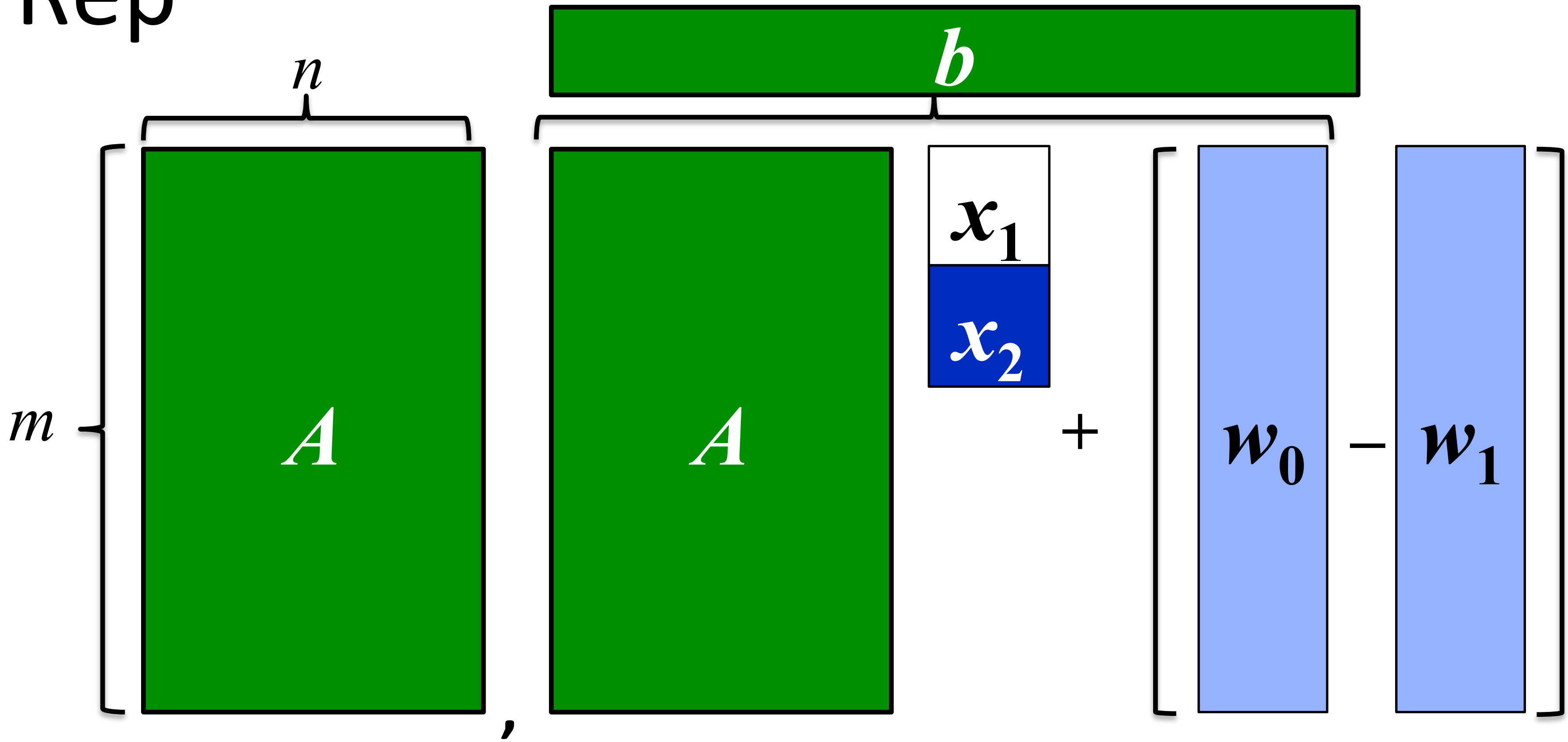


- Q: How are we avoiding our negative results?
- A We don't extract key from w_0 (we are not aware of any notion where $w_0 | (A, b)$ has high entropy)
- Instead, we use secret randomness, and hide it using w_0

Our Construction

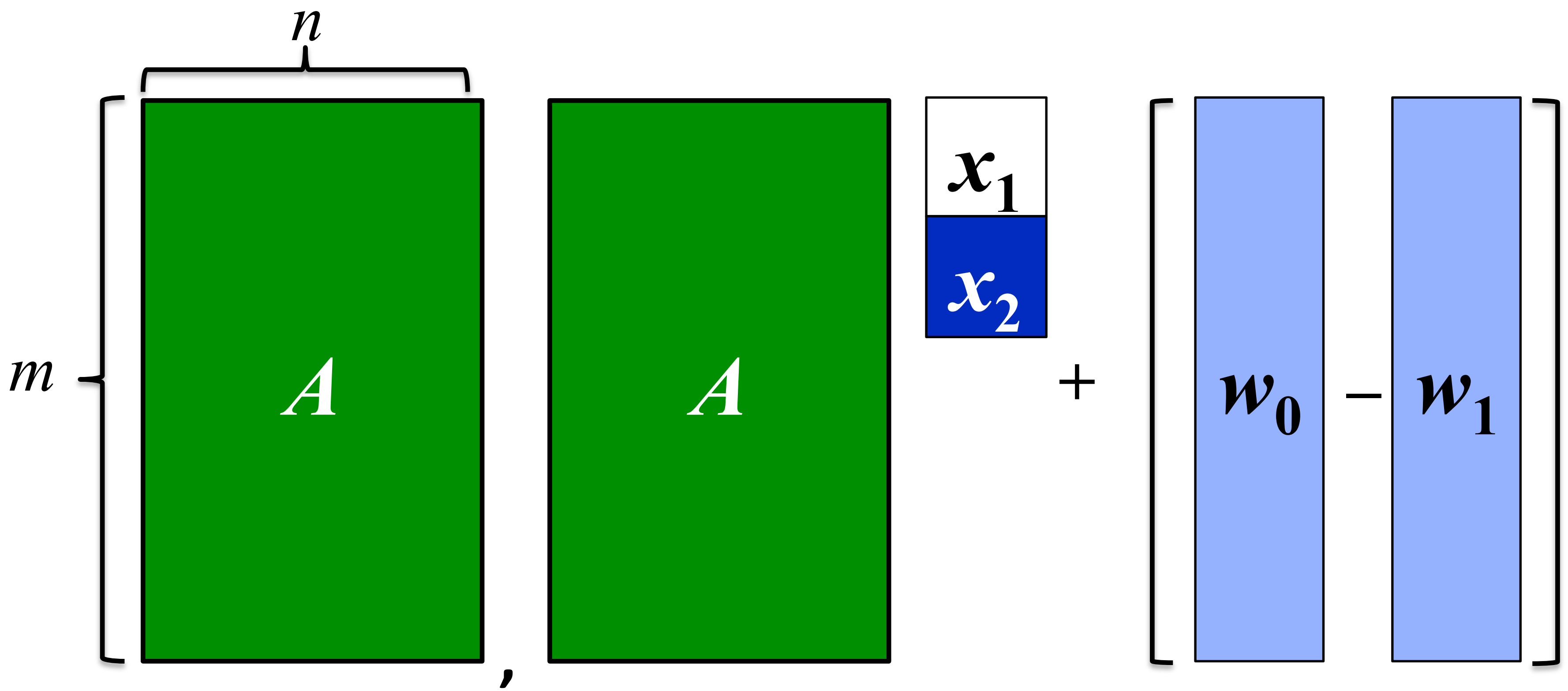


Rep



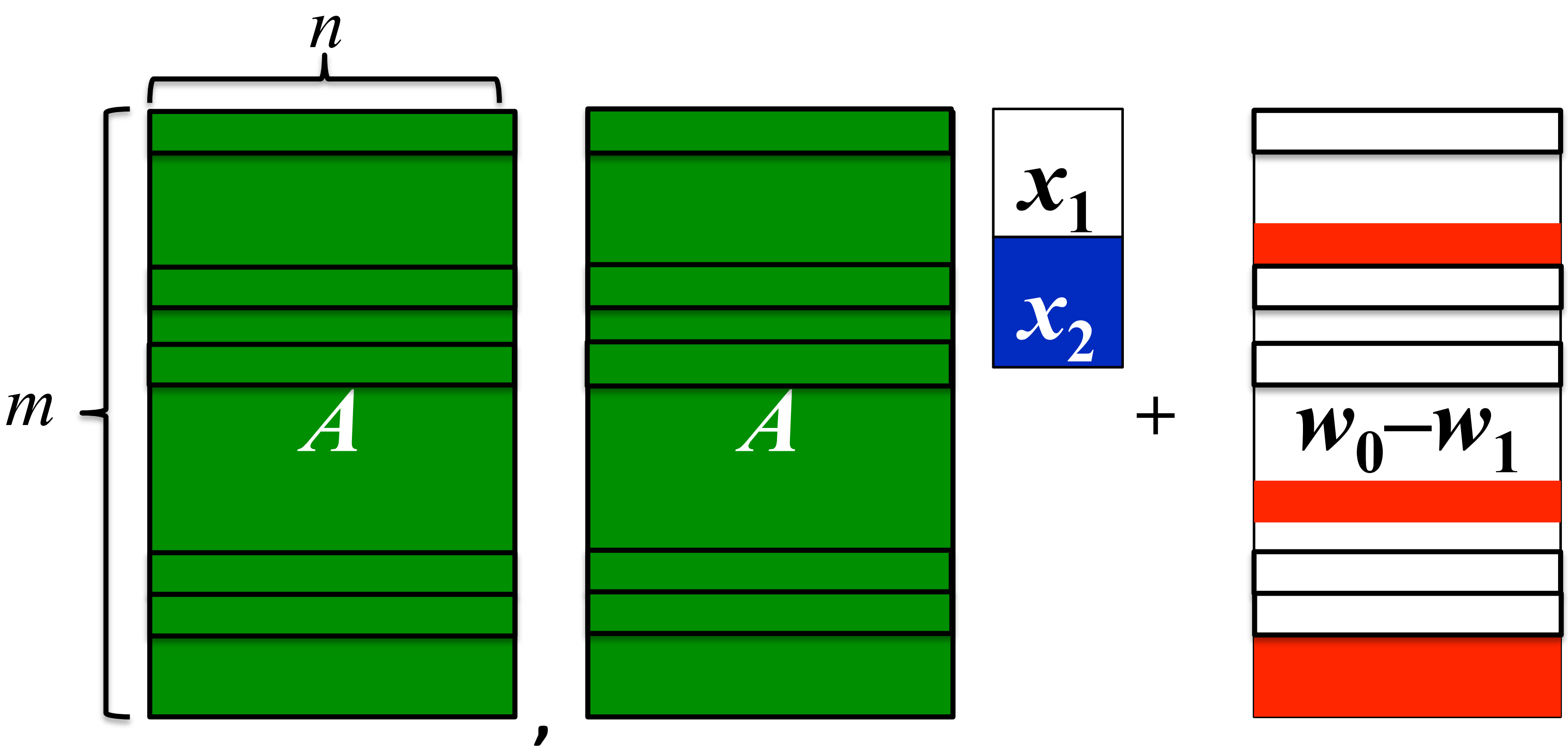
- *Rep* has A and something close to Ax
- This is a decoding problem (same as in the traditional construction)
- Decoding random codes is hard, but possible for small distances.
- (We can't use LWE trapdoor, because there is no secret storage)

Rep



Example algorithm for log many errors:

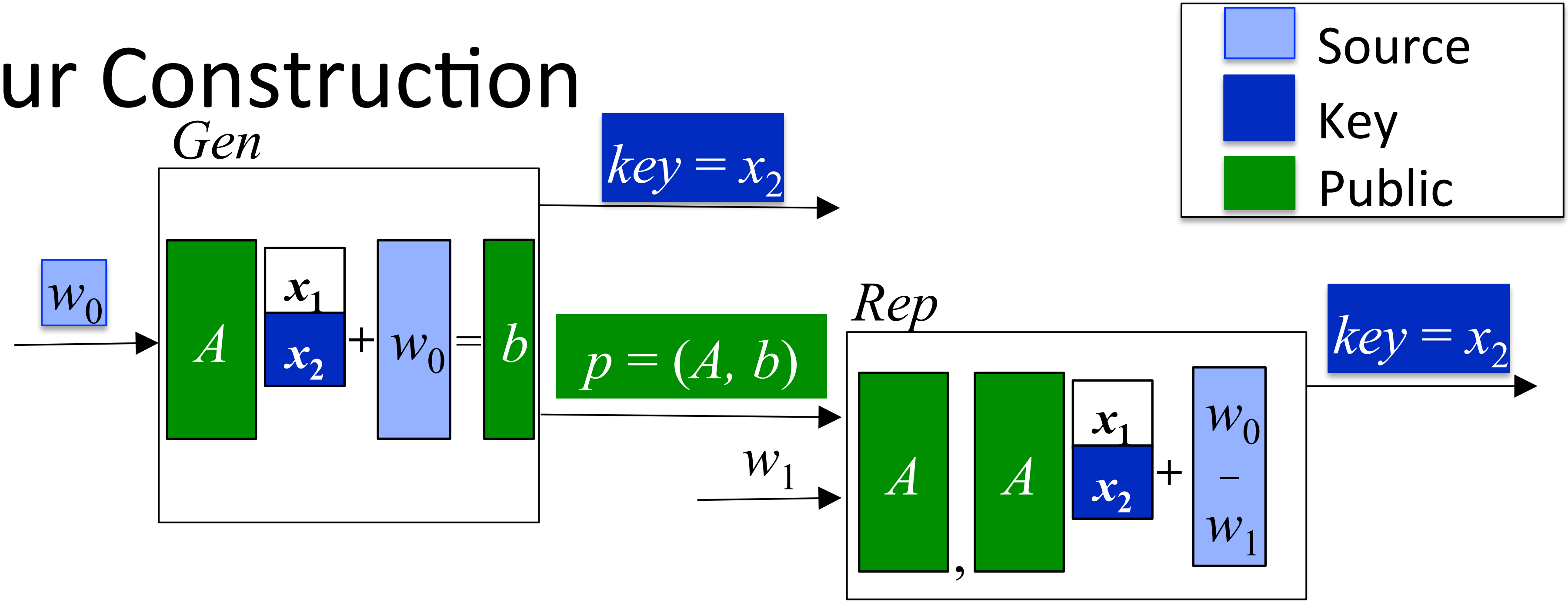
Rep



Example algorithm for log many errors:

- Select n random samples (hopefully, they have no errors)
- Solve linear system for x on these samples
- Verify correctness of x using other samples
- Repeat until successful

Our Construction



- Can correct as many errors as can be efficiently decoded for random linear code (our algorithm: logarithmically many)
- Each dimension of w_0 can be sampled with a fraction of the bits needed for each dimension of x (i.e., we can protect x using fewer than $|x|$ bits)
- So we can get as many bits in key as in w_0 -- lossless!
- Key length doesn't depend on how many errors are being corrected
- Intuition: key is encrypted by w_0 and decryption tolerates noise

Conclusion

- *Fuzzy Extractors* and *Secure Sketches* suffer from entropy losses in information theoretic setting
  - May keep the resulting key from being useful

- What about the Computational Setting?
- Negative Result: Entropy loss inherent for *Secure Sketches* (Additional results about unpredictability of $(W_0 \mid p)$)
- Positive Result:
  Construct lossless *Computational Fuzzy Extractor* using the *Learning with Errors* problem
  - For Hamming distance, with log errors and restricted class of sources (secure LWE error distributions)

Open Problems

- Improve error-tolerance
- Handle additional source distributions
- Beat information-theoretic constructions on practical parameter sizes
- Other computational assumptions?

Questions?