LEFTOVER

REVISITED

Joint work with Boaz Barak, Hugo Krawczyk, Olivier Pereira, Krzysztof Pietrzak, Francois-Xavier Standaert and Yu Yu

Yevgeniy Dodis (New York University)
Imperfect Random Sources

- Ideal randomness is crucial in many areas
  - Especially cryptography (i.e., secret keys) [MP91, DOPS04, BD07]

- However, often deal with imperfect randomness
  - Physical sources, biometric data, partial knowledge about secrets, extracting from group elements (DH key exchange), …

- Necessary assumption: must have (min-)entropy
  - (Min-entropy) \( m \)-source: \( \Pr[X=x] \leq 2^{-m} \), for all \( x \)

- Can we extract (nearly) perfect randomness from such realistic, imperfect sources?
Tool: Randomness Extractor [NZ96].

- Input: a weak secret $X$ and a uniformly random seed $S$.
- Output: extracted key $R = \text{Ext}(X; S)$.
- $R$ is uniformly random, even conditioned on the seed $S$.

$\text{Ext}(X; S), S \approx \text{(Uniform, S)}$

Many uses in complexity theory and cryptography.
- Well beyond key derivation (de-randomization, etc.)
Parameters

- **Min-entropy** $m$.
- **Output length** $\nu$.
  - Equivalent measure: Entropy Loss $L = m - \nu$.
- **Error** $\varepsilon$ (measures statistical distance from uniform).
  - Defines security parameter $k = \log(1/\varepsilon)$
- **Seed Length** $n$.
- **Optimal Parameters** [Sip, RT, DO]:
  - Seed length $n = \mathcal{O}(\text{security parameter } \log(1/\varepsilon))$
  - Entropy loss $L = 2\log(1/\varepsilon)$
- Can we match them efficiently?
Leftover Hash Lemma (LHL)
Leftover Hash Lemma (LHL)

- **Universal Hash Family** $\mathcal{H} = \{ h : \mathcal{X} \rightarrow \{0,1\}^v \}$:

  $\forall x \neq y, \ Pr_h[ h(x) = h(y) ] = \frac{1}{2^v}$

- **Leftover Hash Lemma** [HILL].
  
  Universal hash functions $\{h\}$ yield good extractors:
  
  $(h(X), h) \approx_\varepsilon (U_v, h)$

  - **optimal entropy loss**: $L = 2 \log(1/\varepsilon)$
  - **sub-optimal seed length**: $n \geq |X|

- **Pros**: simple, very fast, nice algebraic properties

- **Cons**: large seed and entropy loss
Part I: Improving the Entropy Loss
Is it Important?

- Yes! Many sources do not have “extra” $2 \log(1/\varepsilon)$ bits
  - Biometrics, physical sources, DH keys of elliptic curves (EC)
    - DH: lower “start-up” min-entropy also improves efficiency
- Heuristic extractors, analyzed in the random oracle model, have “no entropy loss”
- End Result: practitioners prefer heuristic key derivation to provable key derivation (see [DGH^+,Kra])
- Goal: provably reduce $2 \log(1/\varepsilon)$ entropy loss of LHL closer to “no entropy loss” of heuristic extractors
Is not $2\log(1/\varepsilon)$ entropy loss optimal?

- Yes, if must protect against all distinguishers $D$

- **Cryptographic Setting:** restricted distinguishers $D$
  - $D = \text{combination of attacker } A \text{ and challenger } C$
  - $D$ outputs $1 \iff A$ won the game against $C$

- **Case Study:** key derivation for signature/MAC
  - **Assume:** $\Pr[A \text{ forges sig with random key}] \leq \varepsilon \ (= \text{negl})$
  - **Hope:** $\Pr[A \text{ forges sig with extracted key}] \leq \varepsilon' \ (\approx \varepsilon)$
  - **Key Insight:** only care about distinguishers which almost never succeed (on uniform keys) in the first place!
  - Better entropy loss might be possible!
Improved Entropy Loss for Key Derivation

- **Setting**: application $P$ needs a $\nu$–bit secret key $R$
  - **Ideal Model**: $R \leftarrow U_\nu$ is uniform
  - **Real Model**: $R \leftarrow \text{Ext}(X; S)$, where $H_\infty(X) = \nu + L$

- **Assumption**: $P$ is $\varepsilon$–secure in the ideal model

- **Conclusion**: $P$ is $\varepsilon'$–secure in the real model

- **Standard LHL**: if $\text{Ext}$ is universal hash function, then
  \[ \varepsilon' \leq \varepsilon + \sqrt{2^{-L}} \]

- **Our Result**: For a “wide range” of applications $P$
  \[ \varepsilon' \leq \varepsilon + \sqrt{\varepsilon \cdot 2^{-L}} \]
Improved Entropy Loss for Key Derivation

**Moral:**
Might extract more if know why you are extracting

**Standard LHL:** if Ext is universal hash function, then
\[ \varepsilon' \leq \varepsilon + \sqrt{2^{-L}} \]

**Our Result:** For a “wide range” of applications \( P \)
\[ \varepsilon' \leq \varepsilon + \sqrt{\varepsilon \cdot 2^{-L}} \]
Comparison

- **Standard LHL**: \( \varepsilon' \leq \varepsilon + \sqrt{2^{-L}} \)
  - Must have \( L \geq 2\log(1/\varepsilon) \) for \( \varepsilon' = 2\varepsilon \)
  - Not meaningful for \( L \leq 0 \), irrespective of \( \varepsilon \)

- **RO Heuristic**: \( \varepsilon' \leq \varepsilon + \varepsilon \cdot 2^{-L} \)
  - Suffices to have \( L \geq 0 \) (no entropy loss) for \( \varepsilon' = 2\varepsilon \)
  - Meaningful for \( L \leq 0 \), “borrow” security from application

- **Our Result**: \( \varepsilon' \leq \varepsilon + \sqrt{\varepsilon} \cdot 2^{-L} \)
  - “Halfway in between” standard LHL and RO
    - Suffices to have \( L \geq \log(1/\varepsilon) \) for \( \varepsilon' = 2\varepsilon \)
    - Like RO, meaningful for \( L \leq 0 \) (e.g. get \( \varepsilon' = \sqrt{\varepsilon} \) when \( L=0 \))
Which Applications?

- All “unpredictability” applications
  - MAC, signature, one-way-function, ID scheme, …

- Prominent “indistinguishability” applications
  - (stateless) CPA/CCA secure encryption, weak PRFs
  - But not PRFs, PRPs, stream ciphers, one-time pad
    - Note: OK to derive AES key for CPA encryption/MAC!

- Observation: composing with a weak PRF, can include any (computationally-secure) application!
  - E.g., PRFs/PRPs/stream ciphers, but not one-time pad
  - Cost: one wPRF call + wPRF input now part of the seed
Part II: Improving the Seed Length
Expand-then-Extract

- Recall, best $n = O(\text{sec. param. } k)$
  - But LHL needs $n \geq |X|$

- **Idea**: use pseudorandom generator (PRG) $G$ to expand the seed from $k$ bits to $n = |X|$ bits:
  $$\text{Ext}'(X; s) = \text{Ext}(X; G(s))$$
  - Friendly to “streaming” sources
  - Can result in very fast implementations

- **Hope**: extracted bits are pseudorandom

- Is this idea sound?
Soundness of Expand-then-Extract

- **Trivial:** \((\text{Ext}(X; G(S)), G(S)) \approx_c (U_n, G(S))\)
  - Otherwise distinguish \(G(U_k)\) from \(U_n\)

- **Problem:** need \((\text{Ext}(X; G(S)), S) \approx_c (U_n, S)\) \((*)\)

- **Theorem 1:** Under DDH assumption, there exists a PRG \(G\) and a universal hash function \(\text{Ext}\) (thus, extractor, by LHL) s.t. can break \((*)\) efficiently with advantage \(\approx 1\) on any source \(X\)
  - Thus, expand-then-extract might be insecure 😞
OK to Extract Small Number of Bits!

- **Theorem 2**: Extract-then-expand is **secure** when number of extracted bits \( v < \text{“log(PRG security)”} \)
  - **Note 1**: PRG should be secure against \( O(\frac{\exp(v)}{\varepsilon}) \) size circuits
  - **Note 2**: extracted bits are still **statistically** random!
  - **Note 3**: same min-entropy \( m \), error drops to \( \sqrt{\varepsilon} \)

- **Corollary**: always safe to extract \( v = O(\log k) \) bits, sometimes might be safe to extract \( v = \Omega(k) \) bits 😊

- Seed Length \( n \)? At best, \( n = O(v + \log(1/\varepsilon)) \), same as “almost universal” hash functions 😞
Expand-then-Extract Secure in **Minicrypt**

- Counter-example used DDH – “public-key gadget”

- **Minicrypt**: one of Impagliazzo’s worlds, where PRGs exist but no public-key encryption (PKE)

- **Theorem 3**: Extract-then-expand is secure in **Minicrypt**
  - True for any number of extracted bits, but “settle” for efficiently samplable sources and pseudorandom bits
  - Similar in spirit to [HN, Pie, Dzi, DI, PS], **but simpler!**
Expand-then-Extract Secure in Minicrypt

- **Theorem 3**: if $X$ is efficiently samplable, $G$ is a PRG and $D$ efficiently distinguishes $(\text{Ext}(X; G(S)), S)$ from $(U, S)$, then PKE exist
- **Secret Key** = $S$, **Public Key** = $G(S)$
- **Encryption** $\text{Enc}_{PK}(b)$: send ciphertext $R$, where
  - if $b = 0$, sample $X$ and set $R \leftarrow \text{Ext}(X; G(S))$
  - if $b = 1$, set $R \leftarrow U$
- **Decryption** $\text{Dec}_{SK}(R)$: use $D(R, S)$ to recover $b$
- **Semantic security** follows from PRG security:
  $$ ( \text{Ext}(X; G(S)), G(S) ) \approx_c ( U, G(S) ) $$
Interpretation

“It’s not what it looks like”
Corollary: Let $G$ be a PRG. Assume there exists no PKE with $sk = S$, $pk = G(S)$, pseudorandom ciphertexts and $\approx$ same security as $G$. Then expand-then-extract is secure with $G$.

“Practical” PRGs (e.g. AES) unlikely to yield such a PKE

- No black-box construction known (even with powerful “cryptomania” assumptions, like NIZK, IBE, FHE, etc.)
- Possible that no PKE is as secure as AES!
- Would be a major breakthrough with, say, AES

Moral: formal evidence that expand-then-extract might be “secure in practice” (with “actually used” ciphers)
Summary

- Can improve large entropy loss and seed length of LHL
- Entropy loss: for a wide range of applications reduce entropy loss from $2\log(1/\varepsilon)$ to $\log(1/\varepsilon)$
  - Directly includes all authentication and some privacy applications (including CPA encryption, weak PRFs)
  - Using wPRFs, computational extractor for all applications!
- Seed length: expand-then-extract approach
  - Not sound in general…
  - Sound for extracting small # of bits
  - Sound for “practical” PRGs (which do not “imply” PKE)
When life gives you lemmas...

1.1
1.2

It's time to write a paper!

Available at http://eprint.iacr.org/2011/088