FUNCTIONAL SIGNATURES AND PSEUDORANDOM FUNCTIONS

Elette Boyle
Shafi Goldwasser
Ioana Ivan

Traditional Paradigm: **All or Nothing**

- Encryption [DH76]
  - Given SK, can decrypt.
  - Otherwise, can't distinguish encryptions of different messages.
- Digital Signatures [DH76]
  - Given SK, can sign.
  - Otherwise can't forge single new signature.
- Pseudo Random Functions [GGM84]
  - Given SK, can compute.
  - Otherwise can't distinguish from random function.

The Secret Key

2000's: Auxiliary Keys ≈ Partial Abilities

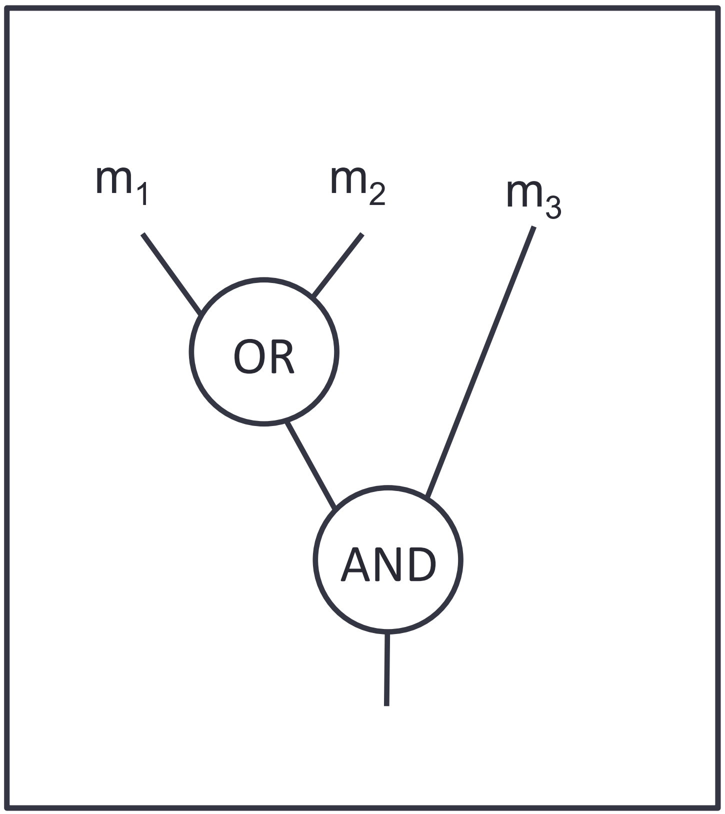
Paradigm Shift for Encryption: Enable **Partial** “Abilities” [2005-on]

- IBE, HIBE, Attribute Based, Policy based, **Functional Encryption** [GPSW06, SW05, BSW11]
  - Master secret key MSK $\Rightarrow$ can compute m from Enc(m).
  - Auxiliary key SK_g $\Rightarrow$ can compute g(m) from Enc(m) and nothing else.

$\Rightarrow$

Functional Encryption ⇒ Cloud Computing Application

Today: Shift Paradigms in the domain of **Digital Signatures** and **Pseudo-random Functions**



g

Corresponds to a function

Functional Digital Signatures

- **Functional** Digital Signatures:
  - master secret key MSK⇒ can sign any message
  - Auxiliary secret key SK_g ⇒ can sign only messages in Range(g)
    - **Interpretation 1:**
      - Can sign only messages that have gone through approved processing
    - **Interpretation 2:**
      - Can sign any message that satisfies a certain predicate

- Related Work:
  - “Signatures of correct computation” - [PST13]
  - “Policy-based signatures” - [BF13]
  - “Delegatable Functional Signatures” - [BMS13]

Example : certified modifications

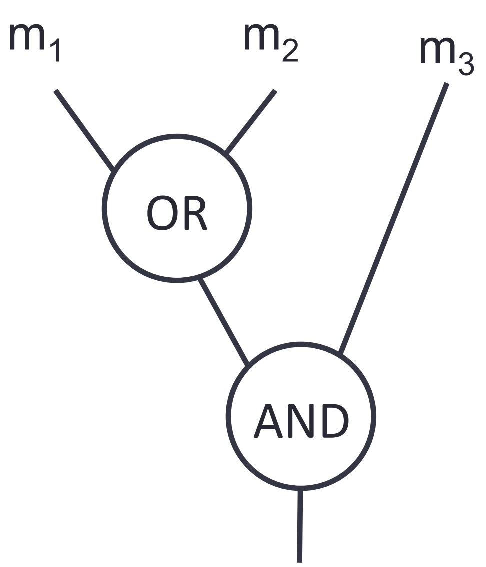
- Certifying that only allowable computations were performed on data.

- Restrict the photo-shop to touch ups of authentic images, e.g. don’t allow cropping or merging of photos.

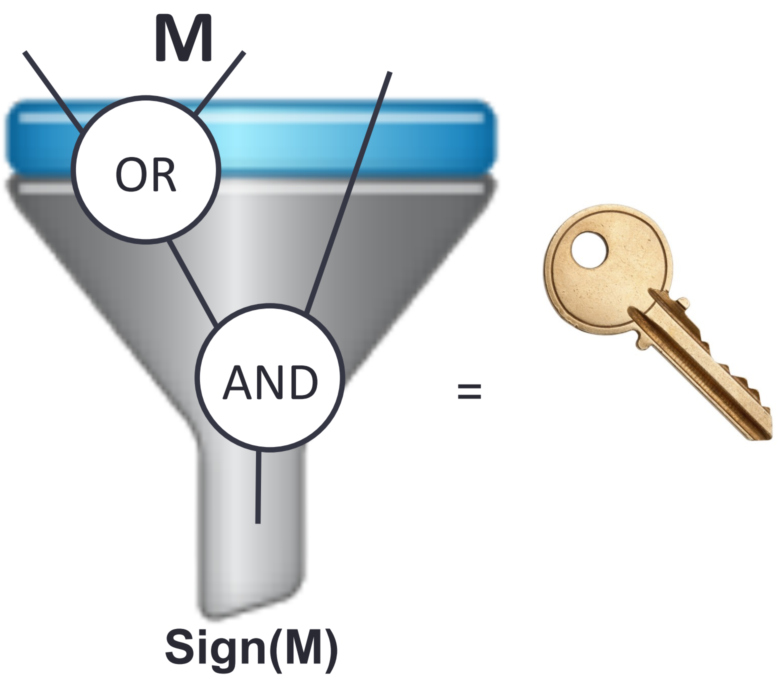
Signature Filters

Sign(M) only if g(M) = 1

Circuit for function g

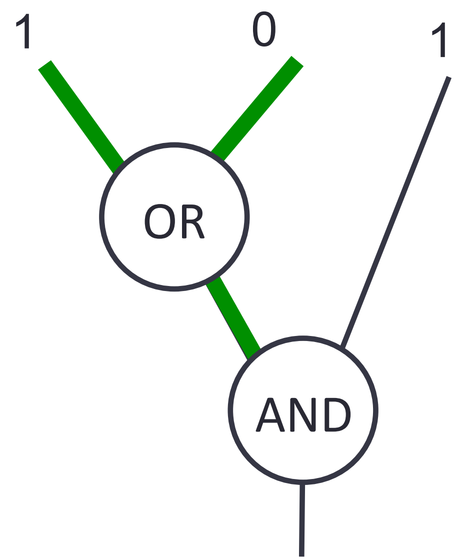


Signing Filter for g

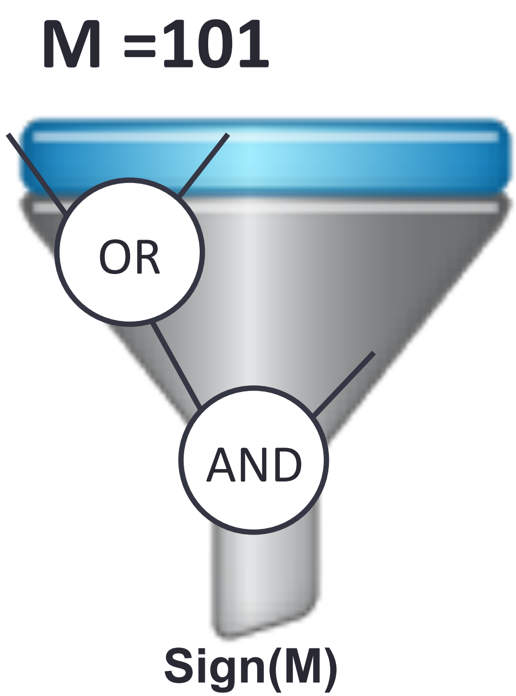


Signature Filters

Circuit for function g

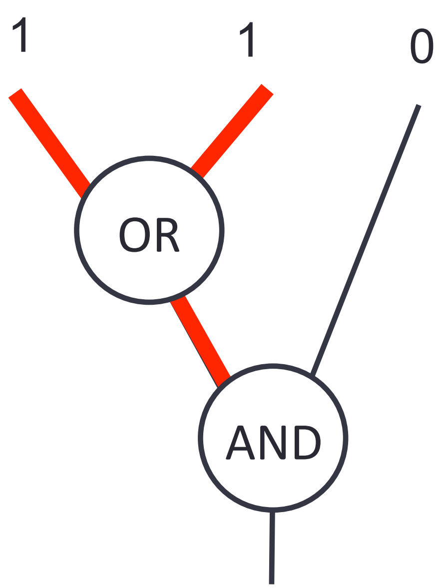


Signing Filter for g



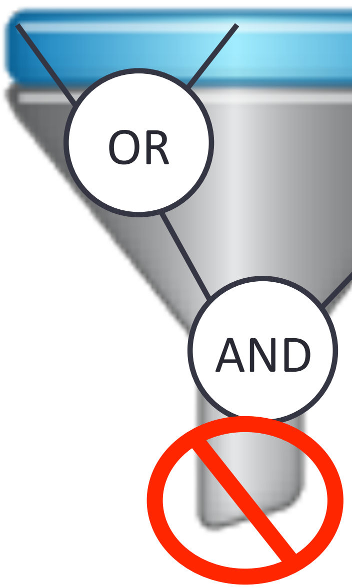
Signature Filters

Circuit for function g



Signing Filter for g

M =110



Functional Signatures - Definition

- A *functional signature* scheme is a tuple of algorithms:
  - Setup(1^k) : (MSK, VK)
  - KeyGen(MSK, g) : sk_g
  - Sign(g, sk_g, m) : (g(m), σ)
  - Verify(VK, σ, m*): 0 or 1
- Correctness:
  - Given $sk_{g,}$ and m, one can sign g(m).
- Security Game:

Special case: g(m) = m iff P(m)=1

More generally: NP relation
g(m,w) = m iff R(m,w)=1

Functional Signatures – Security Game

Setup : (VK, MSK)

VK

g

KeyGen(MSK, g)

} Type I queries

g’,m

Sign(g’, $sk_{g'}$, m)

} Type II queries

.

.

(m*, σ*)

I can forge!

The adversary WINS if Verify(VK, σ* m*) = 1 for NEW m* s.t
m* NOT in Range(g) for any queried g

Prob [WIN] should be negligible

Additional Desirable Properties

- Function Privacy
  - $g_1(m_1) = g_2(m_2) \Rightarrow \text{Sign}(sk_{g1}, m_1) \approx_c \text{Sign}(sk_{g2}, m_2)$

- Succinctness
  - $|\text{Sign}(g, sk_g, m)| = \text{poly}(\lambda, |g(m)|)$, independent of $|g|$ and $|m|$

Our Results

- Theorem 1: OWF ⇒ functional signatures for P
  (NOT succinct or function private)

- Theorem 2: Enhanced trapdoor permutations
  ⇒ function-private functional sigantures for P
  (NOT succinct)

- Theorem 3: SNARKs for NP ⇒ succinct, function-private
  functional signatures for P

- SNARKs: Succinct non-interactive arguments of knowledge s.t. |proof| << |witness|.

The Necessity of Non Falsifiable Assumptions for Succinctness

- Theorem 3 relied on SNARKs

- SNARKs have been shown to exist based on various knowledge assumptions. [BCCT12, GLR12, GGPR12 …]

- SNARKs and SNARGs can't be proved secure using black-box reductions to falsifiable assumption. [GW11]

- Theorem : Succinct functional signatures for P
  $\Rightarrow$ SNARGs for NP.

Functional Pseudorandom Functions (PRF)

Functional PRF $F = \{f_K\}$ w.r.t. $G = \{g_i\}$:

- *KeyGen(k, g) = SK_g*
- *Given SK_g, x in range(g) ⇒ can compute $f_k(x)$*
  x NOT in range(g) ⇒ $f_k(x)$ pseudorandom

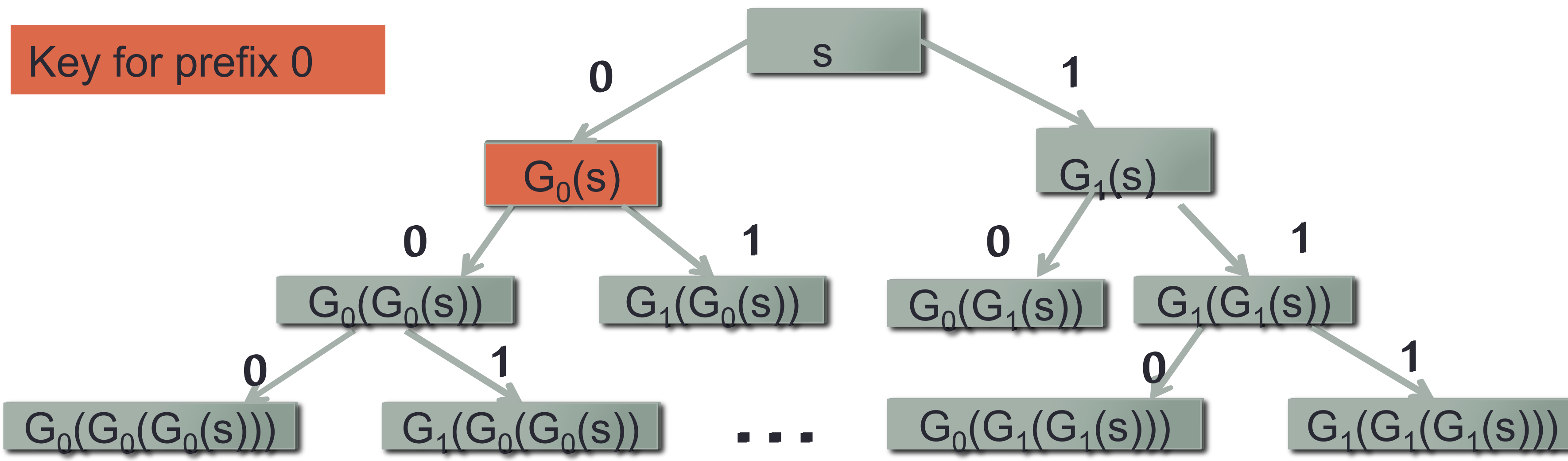
Special case: g(x) = x iff P(x)=1

- Security notion:
  - The adversary requests keys SK_g for functions g in G, $f_k(x) \approx$ random for any x NOT in Range(g) for any g
  - Adaptive versus selective.
- Independent Work: [BW13, KPTZ13].

Construction: Functional PRFs for G_{pre} = prefix-fixing functions

SK_{prefix} allows evaluation on any string x = "prefix||y"

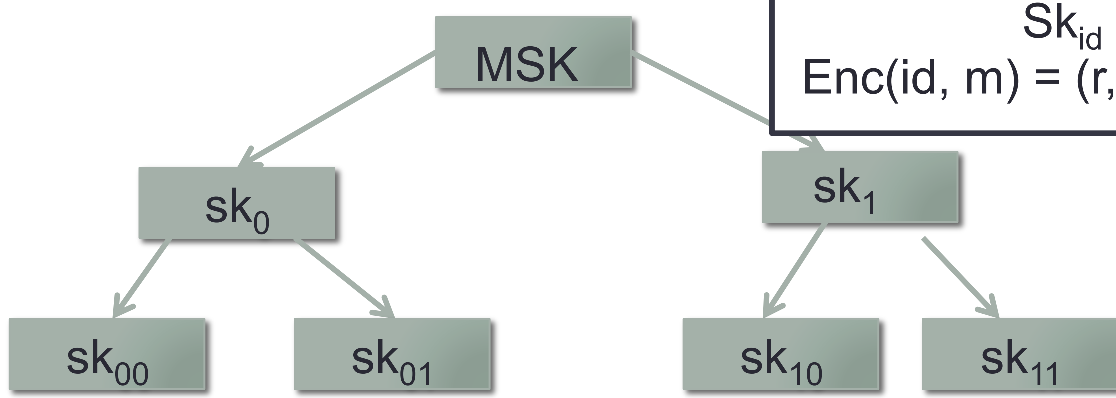
Theorem: OWF ⇒ selectively secure functional PRF for G_{pre}



GGM-based construction

Applications of functional PRFs for prefix-fixing

- Secret-key HIBE



$$Sk_{id} = K_{id}$$
$$Enc(id, m) = (r, m + PRF(id|r))$$

- Functional prefix-fixing PRFs ⇒ punctured PRFs [SW13]
  - Many Applications!

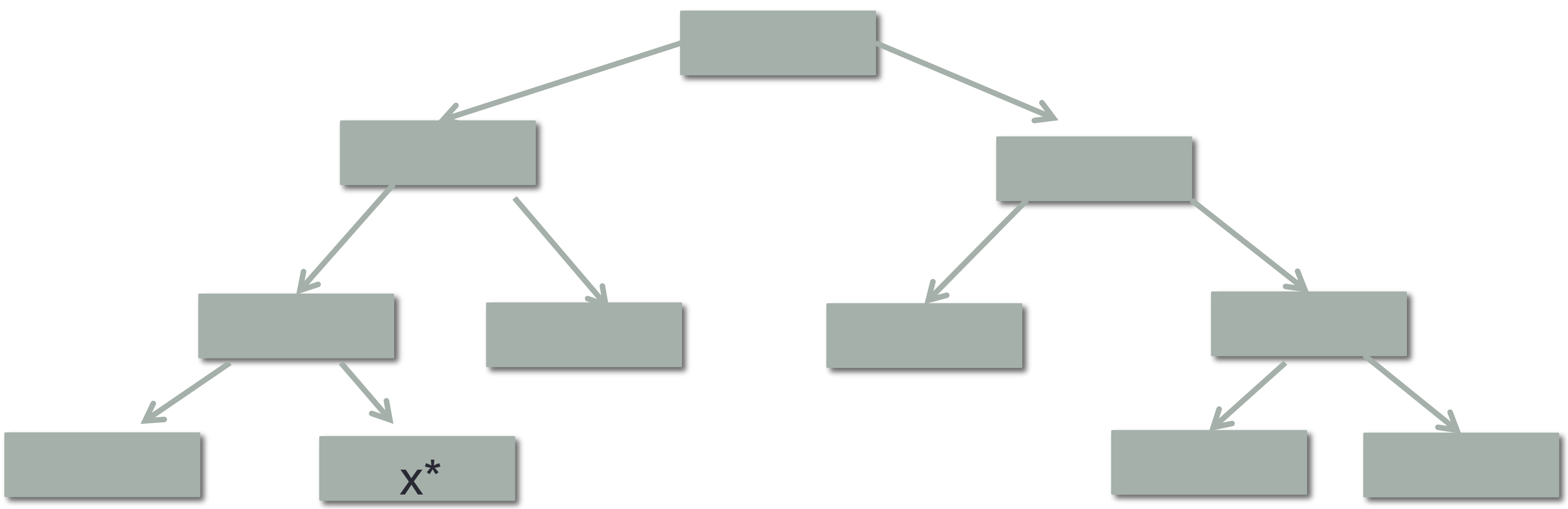
Functional PRF for G_{pre} ⇒ Punctured PRFs

[SW13]

Punctured PRF: "puncture" input x*

Key K_{x^*} lets you compute PRF_K on all inputs except x*.

[SW13]: Functional PRFs for prefix-fixing ⇒ punctured PRFs



Many Applications of **Punctured PRFs**

- Punctured PRFs + indistinguishability Obfuscation (iO)
  ⇒ many applications:

  - PKE, selectively secure signatures, NIZK [SW13]
  - Deniable encryption [SW13]
  - Instantiation of random oracle for full-domain hash applications [HSW13].
  - Efficient traitor-tracing PKE [BZ13]
  - …

Conclusion

- Introduce new primitives:
  Functional Digital Signatures & Functional PRFs

- Functional Signatures:
  - Several constructions supporting keys for P
  - Tradeoff between assumptions & features

- Functional PRFs:
  - Construction for the prefix-fixing family based on OWF

Open Problems

- Functional PRFs for general function families G
  - Construction for general predicates using multilinear maps [BW13]

- Verifiable Functional PRFs?

- Further Applications of Functional Signatures and PRFs

Thank you