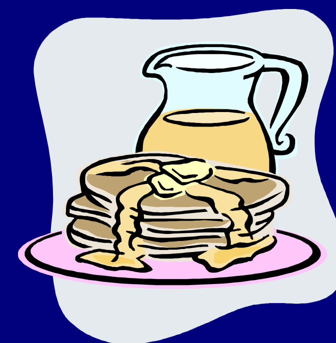
No relation to

On i-Hop Homomorphic Encryption



Craig Gentry, Shai Halevi, Vinod Vaikuntanathan

IBM Research

This Work is About…

Connections between:

- Homomorphic encryption (HE)
- Secure function evaluation (SFE)

Secure Function Evaluation (SFE)

Client Alice has data x

Server Bob has function f

Alice wants to learn $f(x)$

1. Without telling Bob what x is
2. Bob may not want Alice to know f
3. Client Alice may also want server Bob to do most of the work computing $f(x)$

Homomorphic Encryption (HE)

- Alice encrypts data x
  - sends to Bob $c \leftarrow \text{Enc}(x)$
- Bob computes on encrypted data
  - sets $c^* \leftarrow \text{Eval}(f, c)$
  - c^* is supposed to be an encryption of $f(x)$
  - Hopefully it hides f (function-private scheme)
- Alice decrypts, recovers $y \leftarrow \text{Dec}(c^*)$
  - Scheme is (fully) homomorphic if $y = f(x)$

Not necessarily $c^* \cong c$

A More Complex Setting

Alice(x)

Bob(f)

Charlie(g)

Dora(sk)

$c_0 \leftarrow \text{Enc}(x) \xrightarrow{c_0} c_1 \leftarrow \text{Eval}(f, c_0) \xrightarrow{c_1} c_2 \leftarrow \text{Eval}(g, c_1) \xrightarrow{c_2} y \leftarrow \text{Dec}(c_2)$, $y = g(f(x))$

Alice sends encrypted email to Dora:

1. Mail goes first to SMTP server at `BobsISP.com`
   - Bob's ISP looks for "Make money", if found then it tags email as suspicious
2. Mail goes next to `mailboxes.charlie.com`
   - More processing/tagging here
3. Dora's mail client fetches email and decrypts it

A More Complex Setting

Alice(x)

Bob(f)

Charlie(g)

Dora(sk)

$c_0 \leftarrow \text{Enc}(x) \xrightarrow{c_0} c_1 \leftarrow \text{Eval}(f, c_0) \xrightarrow{c_1} c_2 \leftarrow \text{Eval}(g, c_1) \xrightarrow{c_2} y \leftarrow \text{Dec}(c_2)$

$\underbrace{c_1 \leftarrow \text{Eval}(f, c_0) \longrightarrow c_2 \leftarrow \text{Eval}(g, c_1)}_{\text{2-Hop Homomorphic Encryption}}$

- c_1 is not a fresh ciphertext
  - May look completely different
- Can Charlie process it at all?
- What about security?

Background

- Yao's garbled circuits
  - Two-move 1-of-2 Oblivious Transfer
- "Folklore" connection to HE
  - Two-move SFE → function-private HE

1-of-2 Oblivious Transfer

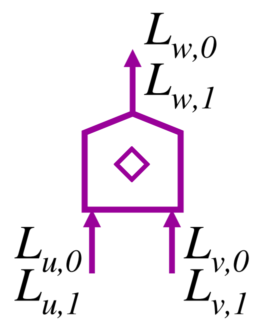
- Alice has bit b, Bob has two Strings L_0,L_1
- Alice learns L_b, Bob learns nothing
- Alice sets (c,s)←OT1(b) sends c to Bob
  - The c part in OT1(0), OT1(1) is indistinguishable
- Bob responds with r←OT2(c, L_0, L_1)
  - ∃ Sim such that for any $L_0, L_1, b,$ (c,s)←OT1(b)
    OT2$(c, L_0, L_1) \cong$ Sim$(c,$ s$, L_b)$
- Alice recovers L_b←OT-out(s,r)

honest-but-curious

Yao's Garbled Circuits

- Bob has f (fan-in-2 boolean circuit)
- Bob chooses two labels $L_{w,0}, L_{w,1}$ for every wire w in the f-circuit
- A gadget for gate $w = u \diamond v$:
  - Know $L_{u,a}$ and $L_{v,b}$ ➔ Learn $L_{w,a\diamond b}$

    { Enc$_{L_{u,a}}$**(Enc**$_{L_{v,b}}$**(**$L_{w,c}$**)) :** $c = a \diamond b$ **}**



- Collection of gadgets for all gates + mapping output labels to 0/1 is the garbled circuit $\Gamma(f)$

Yao's Protocol

- Run 1-of-2-OT for each input wire w with input x_j
  - Alice(x_j) $\leftrightarrow$ Bob($L_{w,0}$, $L_{w,1}$), Alice learns $L_{w,xj}$
- Bob also sends to Alice the garbled circuit $\Gamma(f)$
- Alice knows one label on each input wire
  - computes up the circuit
  - learns one output label, maps it to 0/1
- Bob learns nothing
- Alice's view simulatable knowing only $f(x)$ and $|f|$

Assuming circuit topology is "canonicalized"

Folklore: Yao's protocol → HE

- Roughly:
  - Alice's message $c \leftarrow$OT1(x) is Enc(x)
  - Bob's reply [OT2(c, labels), $\Gamma(f)$] is Eval(f,c)
- Not quite public-key encryption yet
  - Where are (pk, sk)?
  - Can be fixed with an auxiliary PKE
- Client does as much work as server
- Jumping ahead: how to extend it to multi-hop?

Plan for Today

- Definitions: i-hop homomorphic encryption
  - Function-privacy (hiding the function)
  - Compactness (server doing most of the work)
- "Folklore" connection to SFE
  - Yao's protocol → 1-hop non-compact HE
- Extensions to multi-Hop HE
  - DDH-based "re-randomizable Yao"
  - Generically 1-Hop ➔ i-Hop (not today)
    - With or without compactness

Homomorphic Encryption Schemes

- H = {KeyGen, Enc, Eval, Dec}

  (pk,sk) ← KeyGen(), c ← Enc(pk; x)
  c^* ← Eval(pk; f, c), y ← Dec(sk; c^*)

- Homomorphic: $\mathrm{Dec}_{sk}(\mathrm{Eval}_{pk}(f,\mathrm{Enc}_{pk}(x)))=f(x)$
- i-Hop Homomorphic (i = poly(sec-param)):

$$x \to \boxed{\mathrm{Enc}_{pk}(x)} \xrightarrow{c_0} \underbrace{\boxed{\mathrm{Eval}_{pk}(f_1,c_0)} \xrightarrow{c_1} \boxed{\mathrm{Eval}_{pk}(f_2,c_1)} \xrightarrow{c_2} \cdots \xrightarrow{c_j}}_{j\le i \text{ hops}} \boxed{\mathrm{Dec}_{sk}(x)} \to y$$

$$y = f_j(f_{j-1}(\ldots f_1(x) \ldots))$$

- Multi-hop Homomorphic: i-Hop for all i

Properties of Homomorphic Encryption

- Semantic Security [GoMi84]
  - $\forall x,x'$, $\text{Enc}_{\text{pk}}(x) \cong \text{Enc}_{\text{pk}}(x')$
- Compactness
  - The same circuit can decrypt $c_0, c_1, \ldots, c_i$
  - ➔The size of the c_j's cannot depend on the f_j's
    - Hence the name
  - Functionality, not security property

Function Privacy

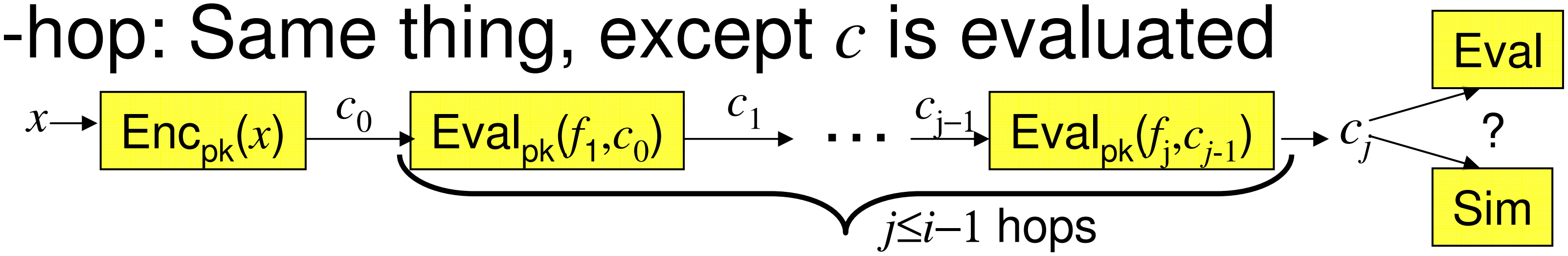
1-hop: Output of $\mathrm{Eval}_{pk}(f,c)$ can be simulated knowing only $pk, c, f(x)$

honest-but-curious

- $\exists$ Sim such that for any $f, x, pk, c \leftarrow \mathrm{Enc}_{pk}(x)$

  $$\mathbf{Eval}_{pk}(f,c) \cong \mathbf{Sim}(pk, c, f(x), |f|)$$

i-hop: Same thing, except c is evaluated



$$\mathbf{Eval}_{pk}(f,c_j) \cong \mathbf{Sim}(pk, c_j, f(f_j(\ldots f_1(x)\ldots)), |f|)$$

- Crucial aspect: indistinguishable given sk and c_j's
  - And randomness that was used to generate them

Aside: “fully” homomorphic

- If $c'\leftarrow$Eval(f,c) has the same distribution as “fresh” ciphertexts, then we get both compactness and function-privacy
- This is “fully” homomorphic
  - Very few candidates for “fully” homomorphic schemes [G09, vDGHV10]
    - Under “circular” assumptions
  - Not the topic of today’s talk

Yao's protocol ➔ 1-hop Function-Private HE

Alice(x) Bob(f) Dora(sk)

$(c,s) \leftarrow \text{SFE1}(x)$ $\xrightarrow{\quad c \quad}$

$r \leftarrow \text{SFE2}(f,c)$

$\xleftarrow{\quad r \quad}$

$y \leftarrow \text{SFE3}(s,r)$

Yao's protocol ➔ 1-hop Function-Private HE

Alice(x,pk) Bob(f) Dora(sk)

$$\underbrace{\boxed{\begin{array}{l}(c,s) \leftarrow \mathrm{SFE1}(x)\\ c' \leftarrow \mathrm{Enc}_{pk}(s)\end{array}}}_{\mathrm{Enc}'_{pk}(x)} \xrightarrow{c,\ c'} \underbrace{\boxed{r \leftarrow \mathrm{SFE2}(f,c)}}_{\mathrm{Eval}_{pk}(f,c,c')} \xrightarrow{r,\ c'} \underbrace{\boxed{\begin{array}{l}s \leftarrow \mathrm{Dec}_{sk}(c')\\ y \leftarrow \mathrm{SFE3}(s,r)\end{array}}}_{\mathrm{Dec}_{sk}(r,c')}$$

- Add an auxiliary encryption scheme
  - with (pk,sk)

Yao's protocol ➔ 1-hop Function-Private HE

Auxiliary scheme E = (Keygen, Enc, Dec)

- H.Keygen: Run $(\mathrm{pk},\mathrm{sk}) \leftarrow E$.Keygen()
- H.Enc$_{\mathrm{pk}}$(x): $(s,c) \leftarrow$ SFE1(x), $c' \leftarrow E$.Enc$_{\mathrm{pk}}(s)$
  Output $[c,c']$
- H.Eval$_{\mathrm{pk}}(f, [c,c'])$: Set $r \leftarrow$ SFE2(f,c)
  Output $[r,c']$
- H.Dec$_{\mathrm{sk}}([r,c'])$: Set $s \leftarrow E$.Dec$_{\mathrm{sk}}(c')$
  Output $y \leftarrow$ SFE3(s, r)

Works for every 2-move SFE protocol

Extending to multi-hop HE

- Can Charlie process evaluated ciphertext?

Alice(x,pk)

Bob(f)

Charlie(g)

$(c,s) \leftarrow$ SFE1(x)
$c' \leftarrow \mathrm{Enc}_{pk}(s)$

$\xrightarrow{c,\ c'}$

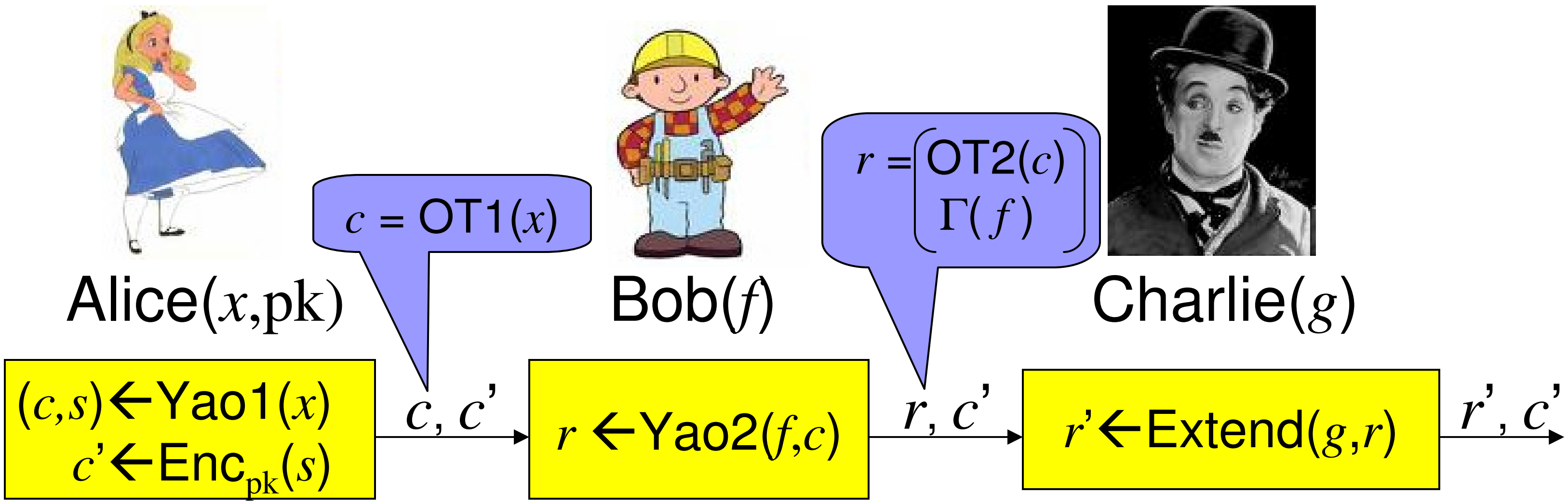
$r \leftarrow$ SFE2(f,c)

$\xrightarrow{r,\ c'}$

?

Extending to multi-hop HE

- Can Charlie process evaluated ciphertext?



- $\Gamma(f)$ include both labels for every f-output
  - Charlie can use them as g-input labels
  - Proceed to extend $\Gamma(f)$ into $\Gamma(g \circ f)$

Extendable 2-move SFE

- Given g and $r \leftarrow$ SFE2(f, SFE1(x)), compute r' = Extend(g,r) $\in$ SFE2($g \circ f$, SFE1(x))
  - I.e., r' in the support of SFE2($g \circ f$, SFE1(x))
- Maybe also require that the distributions

  SFE2($g \circ f$, SFE1(x))

  Extend(g, SFE2(f, SFE1(x))

  are identical/close/indistinguishable
  - This holds for Yao's protocol*

* Assuming appropriate canonicalization

Charlie's privacy

Alice(x) | Bob(f) | Charlie(g) | Dora(sk)

$(c,s) \leftarrow \text{Yao1}(x) \xrightarrow{c} r \leftarrow \text{Yao2}(f,c) \xrightarrow{r} r' \leftarrow \text{Extend}(g,r) \xrightarrow{r'} y \leftarrow \text{Yao3}(s,r')$

- Charlie's function g hidden from Alice, Dora
  - Since $r' \sim \text{Yao2}(g \circ f, c)$, then $g \circ f$ is hidden
- But not from Bob
  - r includes both labels for each input wire of g
    - Yao2 protects you when only one label is known
  - Given r, can fully recover g from r' 

Fixing Charlie's privacy

- Problem: Extend(g,r) is not random given r
- Solution: re-randomizable Yao
  - Given any $r \in \Gamma(f)$, produce another random garbling of the same circuit, $r' \leftarrow$reRand(r)
- $r' \leftarrow$reRand(r) $\cong \Gamma(f)$, even given r
- Charlie outputs $r' \leftarrow$reRand(Extend(g,r))

Re-Randomizable SFE

- Π=(SFE1, SFE2, SFE3) re-randomizable if $\forall\ x, f,\ (c,s)\leftarrow$SFE1$(x)$, $r\leftarrow$SFE2(f,c)

  Honest-but-curious

  reRand$(r) \cong$ SFE2(f,c)

  Identical / close / indistinguishable
  - Even given x, f, c, r, s

Thm: Extendable + re-Randomizable SFE
➔ multi-hop function-private HE

Proof: Evaluator j sets $r_j\leftarrow$reRand(Extend(f_j,r_{j-1})) ■

Re-randomizing Garbled Circuits

- DDH-based re-randomizable Yao Circuits
- Using Naor-Pinkas/Aiello-Ishai-Reingold for the OT protocol
  - Any "blindable OT" will do
- Using Boneh-Halevi-Hamburg-Ostrovsky for gate-gadget encryption
  - Need both key- and plaintext-homomorphism
  - And resistance to leakage…

DDH-based OT [NP01,AIR01]

- OT1(b) = <g, h, x=g^r, {y_b=h^r, $y_{1\text{-}b}$=$h^{r'}$}>
  - (g, h, x, y_b)-DDH, (g, h, x, $y_{1\text{-}b}$)-non-DDH
- OT2((g, h, x, y_0,y_1), γ_0, γ_1)

  = <($g^{s_0}h^{t_0}$, $x^{s_0}y_0^{t_0}$ g^{γ_0}),($g^{s_1}h^{t_1}$, $x^{s_1}y_1^{t_1}$ g^{γ_1})>

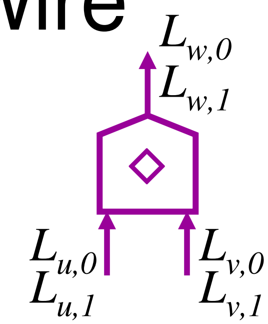
  γ_0, γ_1 are bits
- On strings $\vec{\gamma}_0$,$\vec{\gamma}_1$, use same (g,h,x,y_0,y_1) for all bits
- Scheme is additive homomorphic:
  - For every c←OT1(b), r←OT2(c,γ_0,γ_1), δ_0, δ_1
    reRand(c, r, δ_0, δ_1) ≡ OT2(c, $\gamma_0\oplus\delta_0$, $\gamma_1\oplus\delta_1$)

BHHO encryption [BHHO08]

- We view it as a secret-key encryption
- Secret key is a bit vector $s \in \{0,1\}^\ell$
- Encryption of bit b is a vector $<g_0, g_1, \ldots, g_\ell>$
  - Such that $g_0 \prod_j g_j^{s_j} = g^b$
  - BHHO public key is a random encryption of zero
- Key- and plaintext- additively-homomorphic
  - For every $s,t,\delta,\delta' \in \{0,1\}^\ell$, pk←Enc$_s(0)$, c←Enc$_s(t)$: c'←reRand$(\mathrm{pk},c,\delta,\delta') \cong$ Enc$_{s\oplus\delta}(t\oplus\delta')$
  - c' (pseudo)random, even given pk, c, s, t, δ, δ'

BHHO-based Yao Circuits

- Use NP/AIR protocol for the 1-of-2-OT
- Two ℓ-bit masks $L_{w,0}$, $L_{w,1}$ for every wire
  - Used as BHHO secret keys
- A gadget for gate $w = u\diamond v$:
  - Choose four random masks $\delta_{a,b}$ $(a,b\in\{0,1\})$
  - Gate gadget has four pairs (in random order)



$$\{ <\mathbf{Enc}_{L_{u,a}}(\delta_{a,b}), \mathbf{Enc}_{L_{v,b}}(\delta_{a,b}\oplus L_{w,c})> : c = a\diamond b \}$$

Is this re-Randomizable?

- Not quite…
- Want to XOR a random $\delta_{w,b}$ into each $L_{w,b}$
  - But don't know what ciphertexts use $L_{w,0}$ / $L_{w,1}$
  - Cannot use different masks for the two labels
- XOR the same mask to both $L_{w,0}$, $L_{w,1}$?
  - No. Bob knows old-$L_{w,0}$, old-$L_{w,1}$, Dora knows new-$L_{w,\mathrm{b}}$, together they can deduce new-$L_{w,1-\mathrm{b}}$

Better re-Randomization?

- We must apply the same transformation $T(*)$ to both labels of each wire
  - $T_\delta(x) = x \oplus \delta$ does not work
- We "really want" 2-universal hashing:
  - Given $L_0, L_1, T(L_b)$, want $T(L_{1-b})$ to be random
  - Must be able to apply T$(*)$ to both key, plaintext
- Even BHHO can't do this (as far as we know)
  - But it can get close…

Stronger homomorphism of BHHO

- Key- and plaintext-homomorphic for every transformation $T(*)$ that:
  - Is an affine function over Z_q^ℓ
  - Maps 0-1 vectors to 0-1 vectors
- In particular: bit permutations
  - multiplication by a permutation matrix
- For every pk←$\mathrm{Enc}_s(0)$, c←$\mathrm{Enc}_s(t)$, $\pi,\pi' \in S_\ell$
  c'←permute$(\mathrm{pk},c,\pi,\pi') \cong \mathrm{Enc}_{\pi(s)}(\pi'(t))$
  - c' (pseudo)random, even given pk, c, s, π, π'

Bit Permutation is "sort-of" Universal

- For random Hamming-weight-$\ell/2$ strings

Permutation Lemma:

For random $L, L' \in_R$ HW($\ell/2$), $\pi \in_R S_\ell$, the expected residual min-entropy of $\pi(L')$ given $\pi(L), L, L'$ is

$E_{L,L',\pi}\{\, H_\infty(\pi(L') \mid \pi(L), L, L') \,\} \geq \ell - {}^3\!/_2 \log \ell$

Proof: Fix $L, L', \pi(L)$, then $\pi(L')$ is uniform in the set $\{\, x \in \mathrm{HW}(\ell/2) : \mathrm{HD}(\pi(L), x) = \mathrm{HD}(L, L') \,\}$

- HD – Hamming Distance

■

BHHO is secure even with balanced keys

re-Randomizable BHHO-based Yao

- Labels have Hamming weight exactly $\ell/2$
- Use NP/AIR protocol for the 1-of-2-OT
- Two masks $L_{w,0},L_{w,1} \in \mathrm{HW}(\ell/2)$ for every wire
- A gadget for gate $w = u \diamond v$:
  - Gate gadget has four pairs (in random order)

    $\{ <\mathrm{Enc}_{L_{u,a}}(\delta_{a,b}), \mathrm{Enc}_{L_{v,b}}(\delta_{a,b} \oplus L_{w,c})> : c = a \diamond b \}$
- Instead of output labels (secret keys), provide corresponding public keys
  - Still extendable: can use pk for encryption

re-Randomization

Input: OT response r, garbled circuit Γ

- Choose a permutation π_w for every wire w
- For input wires, permute the OT response
  - We use bit-by-bit OT, and "blindable"
- Permute the gate gadgets accordingly
- Also re-randomize the gate masks $\delta_{a,b}$
  - Using the BHHO additive homomorphism

re-Randomizable yet?

L, L' random in the honest-but-curious model

- For each wire, adversary knows L, L', $\pi(L)$
  - Permutation lemma: min-entropy of $\pi(L')$ almost ℓ bits
- We use $\pi(L')$ as BHHO secret key
  - Use Naor-Segev'09 to argue security
- NS09: BHHO is secure, under leakage of O(ℓ) bits
- View L, L', $\pi(L)$ as *randomized leakage* on $\pi(L')$
  - Leaking only $^3/_2 \log \ell$ bits on the average
  - So we're safe
- Security proof is roughly the same as the Lindell-Pinkas proof of the basic Yao protocol

Summary

- Highlighted the multi-hop property for homomorphic encryption
  - In connection to function privacy, compactness
- Described connections to SFE
- A DDH-based multi-hop function private scheme
  - Not compact
  - Uses re-randomizable Yao circuits
- Other results (generic):
  - 1-hop FP ➔ i-hop FP for every constant i
  - 1-hop compact FP ➔ i-hop compact FP for every i
  - 1-hop compact + 1-hop FP ➔ 1-hop compact FP

Open Problems

- Malicious model
  - The generic constructions still apply
  - Not the randomized-Yao-circuit construction
    - Main sticky point is the permutation lemma
- Other extensions
  - General evaluation network (not just a chain)
  - Hiding the evaluation-network topology
  - Other adversary structures

Thank you

1-hop Function-Private → i-hop FP

- Given E = (KeyGen, Enc, Eval, Dec)
  - and a constant parameter d
- Build H_d = (KeyGen*, Enc*, Eval*, Dec*)
  - d-hop function-private, complexity $n^{O(d)}$
- Use d+1 E-public-keys
  - α_j encrypts j'th sk under j+1st pk
  - j^{th} node evaluates $f_j \circ \mathrm{Dec}_{c_{j-1}}(*)$ on ciphertext α_j
    - The input to $\mathrm{Dec}_{c_{j-1}}$ is sk
    - Ciphertext from node j-1 hard-wired in $\mathrm{Dec}_{c_{j-1}}$
    - α_j is a "fresh ciphertext", not an evaluated one

1-hop Function-Private → i-hop FP

KeyGen*: $(\mathrm{pk}_j,\mathrm{sk}_j) \leftarrow$ KeyGen(), $\alpha_j \leftarrow \mathrm{Enc}_{\mathrm{pk}_{j+1}}(\mathrm{sk}_j)$

- $\mathrm{sk}^*=\{\mathrm{sk}_j\}$, $\mathrm{pk}^*=\{(\alpha_j, \mathrm{pk}_j)\}$, $j=0,1, \ldots, d$

Enc* $_{\mathbf{pk}^*}(x)$: output [level-0, $\mathrm{Enc}_{\mathrm{pk}_0}(x)$]

Dec* $_{\mathbf{sk}^*}$([level-j, c]): output $\mathrm{Dec}_{\mathrm{sk}_j}(c)$

Eval* $_{\mathbf{pk}^*}$(f, [level-j, c]):

- Compute description of $F_{f,c}(s) \equiv f(\ \mathrm{Dec}_s(c)\ )$
  - Input is s, not c
- Set $c' \leftarrow \mathrm{Eval}_{\mathrm{pk}_{j+1}}(F_{f,c}, \alpha_j)$, output [level-($j$+1), c']

1-hop Function-Private ➔ i-hop FP

- The description size of $F_{f,c}(s) \equiv f(\,\mathrm{Dec}_s(c)\,)$ is at least $|f| + |c|$
- Size of c'=Eval$_{\mathrm{pk}_{j+1}}(F_{f,c}, \alpha_j)$ can be $n^{O(1)} \times |F_{f,c}|$
  - For a non-compact scheme (e.g., Yao-based)
- So after i hops, ciphertext size is

$$n^{O(1)} \times (|f_i| + n^{O(1)} \times (|f_{i-1}| + \ldots\ n^{O(1)} \times (|f_1| + c_0)\ \ldots))$$
$$\approx n^{O(i)} \times (c_0 + \Sigma_j |f_j|)$$

- Can only do constant many hops

1-hop Compact FP → i-hop Compact FP

- If underlying scheme is compact, then size of c'$=\text{Eval}_{\mathrm{pk}_{j+1}}(F_{f,c}, \alpha_j)$ does not grow
- Can do as many hops as α_j's in pk^*
- If pk^* includes $\alpha \leftarrow \text{Enc}_{\mathrm{pk}}(\mathrm{sk})$, then we can handle any number of hops
  - This assumes that scheme is circular secure

1-hop FP + 1-hop Compact ➔ 1-hop Compact FP

- Roughly, $\text{Eval}^*(f) = \text{cEval}(\text{pEval}(f))$
  - pEval makes it private, cEval compresses it
- pk^* includes ppk, cpk1,cpk2, and also $\alpha = \text{pEnc}_{ppk}(csk_0)$, $\beta = \text{cEnc}_{cpk_1}(psk)$
  - $sk^* = [csk_0, csk_1]$
- $\text{Eval}_{pk^*}(f, c)$: // c encrypted under cpk_0
  - Let $F_{f,c}(s) \equiv f(\text{cDec}_s(c))$, set $c' \leftarrow \text{pEval}_{ppk}(F_{f,c}, \alpha)$
  - Let $G_{c'}(s) \equiv \text{pDec}_s(c')$, set $c^* \leftarrow \text{cEval}_{cpk_2}(G_{c'}, \beta)$