Fully Homomorphic Encryption

Zvika Brakerski

Weizmann Institute of Science

Technion CRYPTODAY, December 2015

What Are You Searching For?

We know

Medical information, navigation, email, business information, other personal information…

Want privacy!

Outsourcing Computation

location, destination

medical, records

search query

x

x

$f(x)$

search results

diagnosis

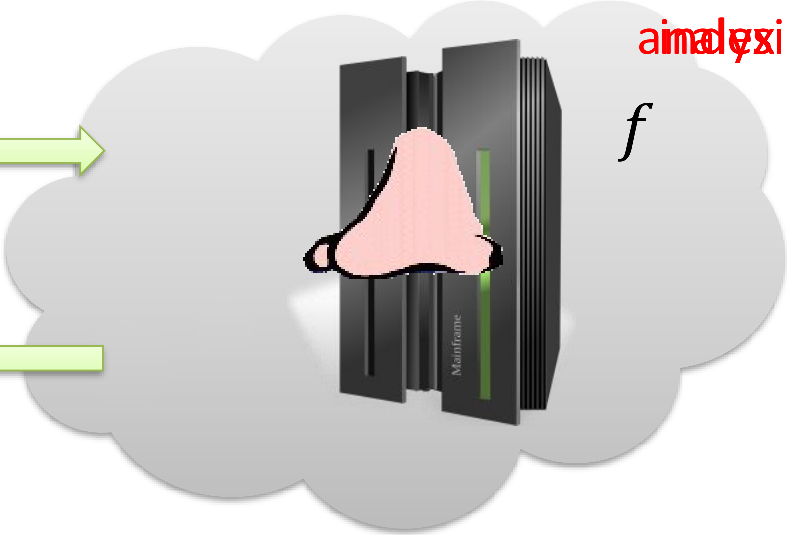
driving route

google web index

medical analysis

routing

f



What if x is private?

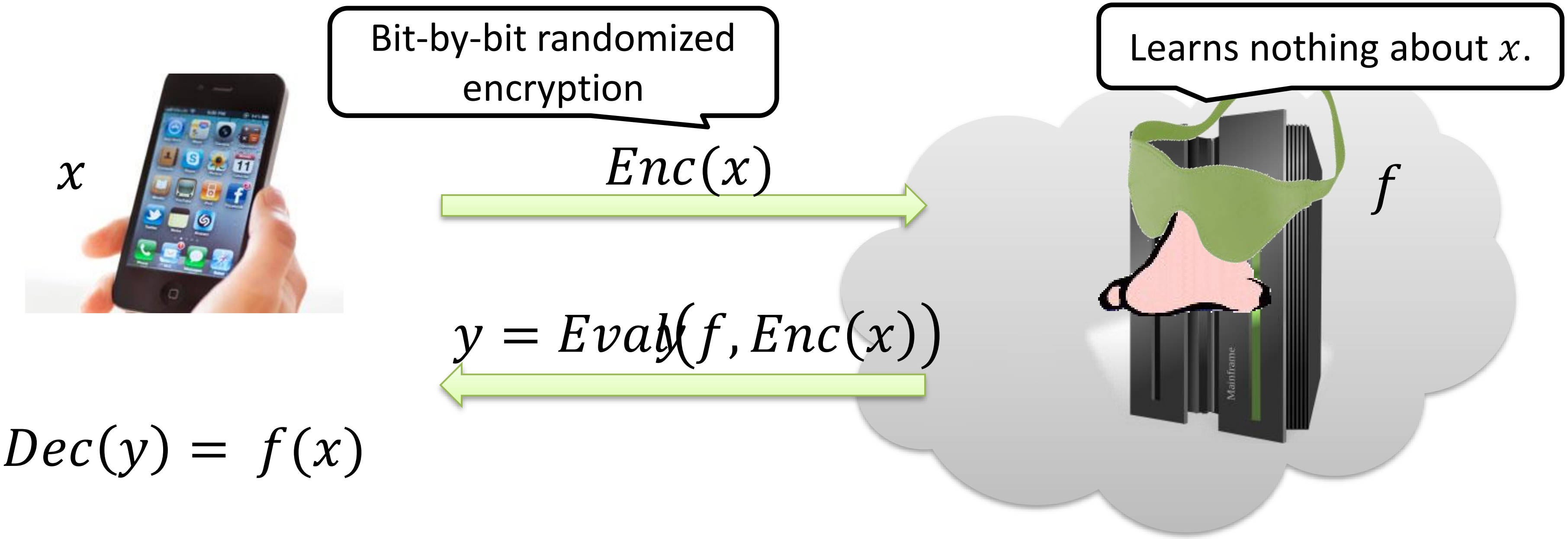
How to Keep Private From the Cloud

We promise we wont look at your data. Honest!

We want real protection.

Outsourcing Computation Privately

Fully Homomorphic Encryption (FHE)



x

Bit-by-bit randomized encryption

$Enc(x)$

Learns nothing about x.

f

$y = Eval(f, Enc(x))$

$Dec(y) = f(x)$

<span style="color:red">Fully</span> Homomorphic = Homomorphism for <span style="color:red">any</span> efficient f

Homomorphic

$f, Enc(x$

computational model: f given as circuit

Goal: $Eval$ for **universal** set of gates (NAND(x,y)=1-xy)

Some Applications

In the cloud:

- Private outsourcing of computation.
- Near-optimal private outsourcing of storage (single-server PIR). [G09,BV11b]
- Verifiable outsourcing (delegation). [GGP11,CKV11,KRR13,KRR15]
- Private machine learning in the cloud. [GLN12,HW13]

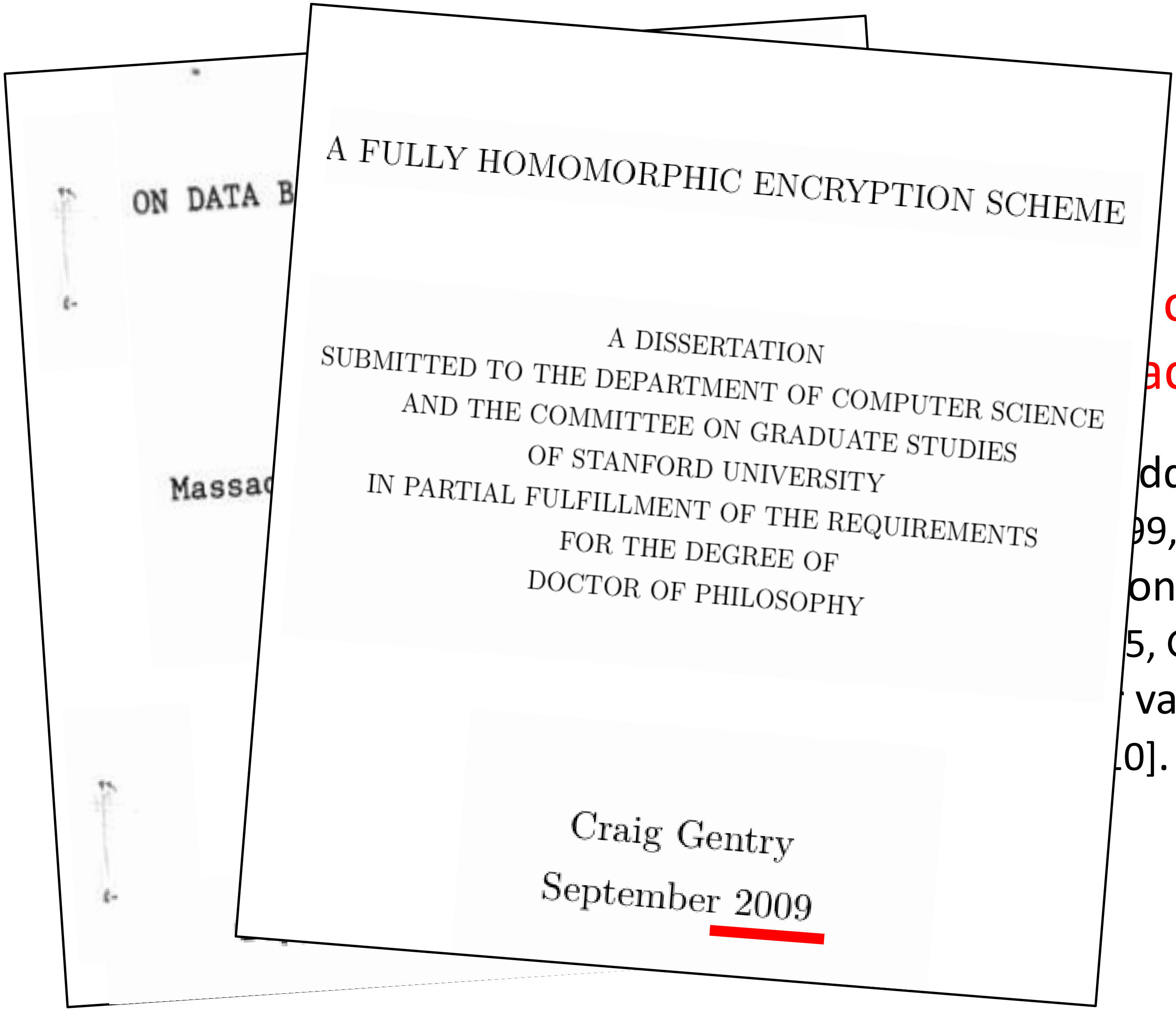
Secure multiparty computation:

- Low-communication multiparty computation. [AJLTVW12,LTV12]
- More efficient MPC. [BDOZ11,DPSZ12,DKLPSS12]

Primitives:

- Succinct argument systems. [GLR11,DFH11,BCCT11,BC12,BCCT12,BCGT13,…]
- General functional encryption. [GKPVZ12]
- Indistinguishability obfuscation for all circuits. [GGHRSW13]

Making Crypto History



ON DATA B

Massac

A FULLY HOMOMORPHIC ENCRYPTION SCHEME

A DISSERTATION
SUBMITTED TO THE DEPARTMENT OF COMPUTER SCIENCE
AND THE COMMITTEE ON GRADUATE STUDIES
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
DOCTOR OF PHILOSOPHY

Craig Gentry
September 2009

of hardly scratching
ace:

ddition [RSA78, R79, GM82,
99, R05].

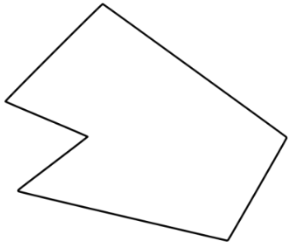
on + 1 multiplication
5, GHV10].

variants [SYY99, IP07,
0].

… is it even possible?

FHE Challenges

Understanding.



Security.

- Cryptographic assumptions.
- Security notions.

Efficiency.

- Size of keys/ciphertexts.
- Time overhead for Eval.
- Computational model.

Constructing (Somewhat) Homomorphic Encryption

secret algebraic equivalence
e.g. (mod p) for secret p

Basic Idea: Find scheme s.t. $c \approx m + 2e$

ciphertext

message

small (even) noise

Add/multiply ciphertexts ⇒ Add/multiply messages

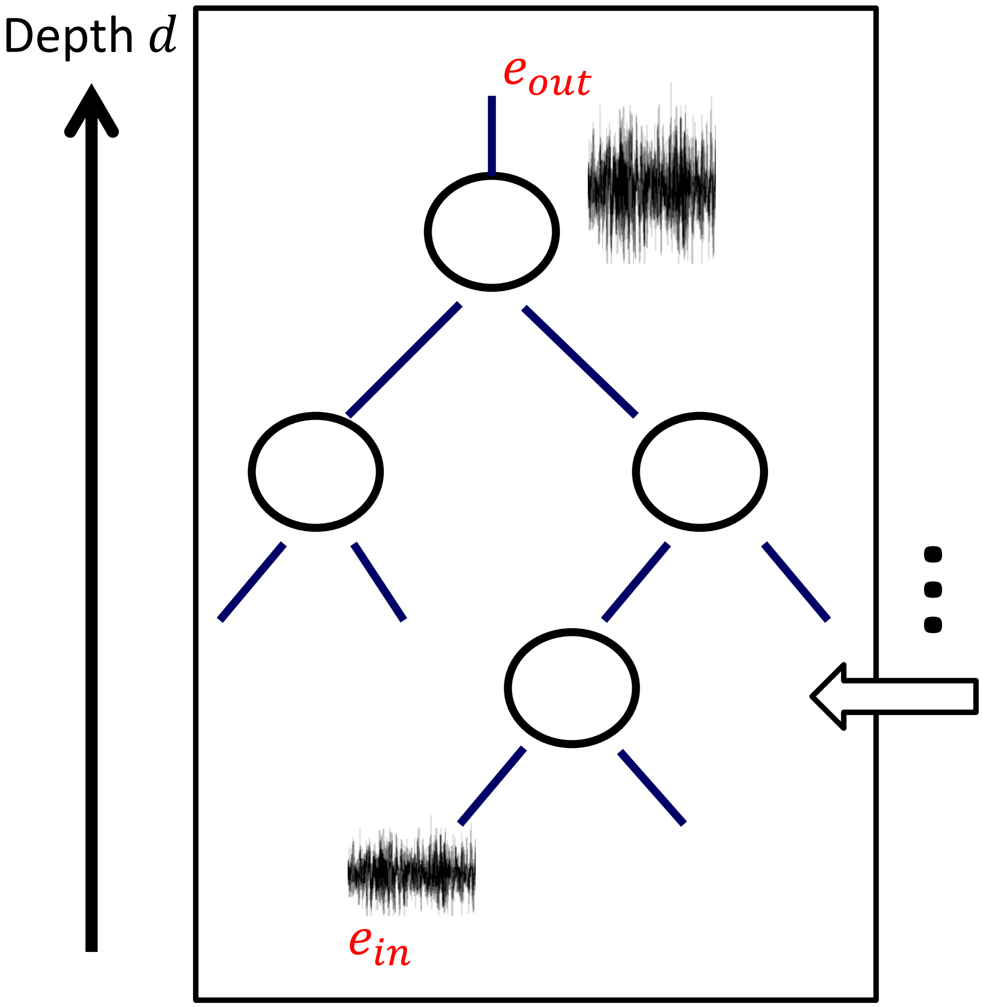
Security?

Noise grows with homomorphic evaluation –
must not grow “too much”!

In the example above: $|e_{mult}| \approx |e_{in}|^2$

Noise in Homomorphic Evaluation

Noise grows during homomorphic evaluation



$$|e_{out}| \leq E^{2^d}$$

$$|e_{i+1}| \leq |e_i|^2$$

$$|e_{in}| \leq E$$

Some of the Progress Since 2009

- From ad-hoc assumption to worst-case lattice assumption [BV11b,BGV12,BV14].
  - As secure as any other encryption scheme.

- Noise is down to $|e_{mult}| \approx k \cdot |e_{in}|$ [BGV12,B12,GSW13,BV14].
  - $|e_{out}| \leq k^d \cdot E$ (instead of E^{2^d}).
  - "Leveled" FHE.

- Using polynomial rings to improve efficiency [G09,SV10,BV11a,BGV12,GHS12a,GHS12b,GHS12c,GHPS13,AP13].

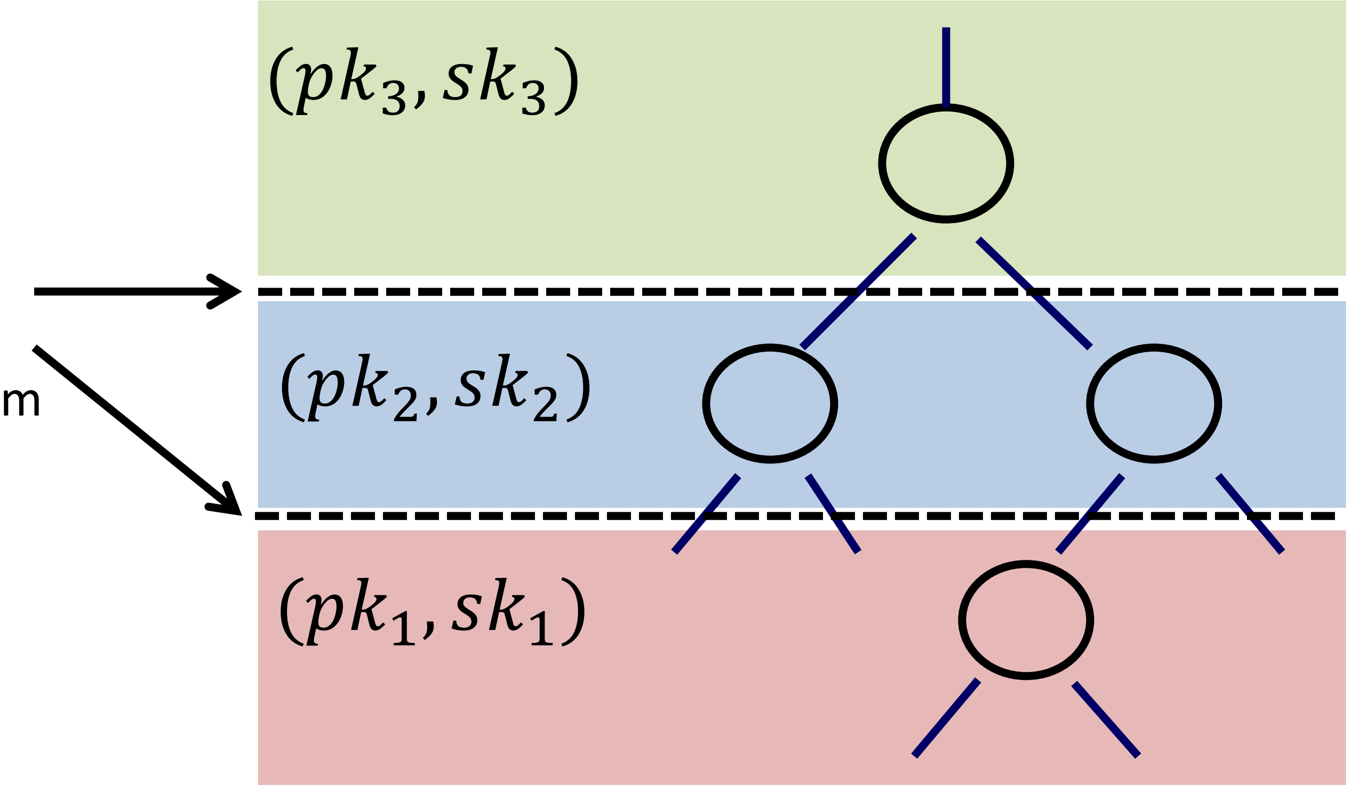
- "Batching" many messages in single ciphertext [SV10,BGV12,GHS12a,GHS12b,GHS12c,HS15].

- But still need "bootstrapping" to get full homomorphism…

Bootstrapping [G09]

Given scheme with bounded d_{hom}
How to extend its homomorphic capability?

Idea: Do a few operations, then "switch" to a new instance



(pk_3, sk_3)

(pk_2, sk_2)

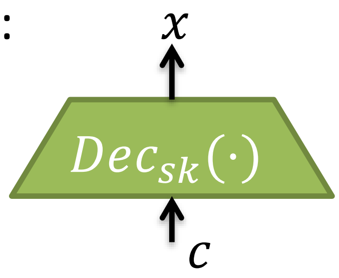
(pk_1, sk_1)

Switch keys

"cost" in homomorphism

How to Switch Keys

Decryption circuit:

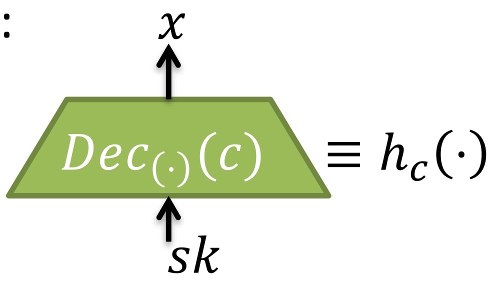


x

$Dec_{sk}(\cdot)$

c

Dual view:



x

$Dec_{(\cdot)}(c)$ $\equiv h_c(\cdot)$

sk

$$h_c(sk) = Dec_{sk}(c) = x$$

given c, server can compute circuit for $h_c(\cdot)$

Apply $h_c(\cdot)$ homomorphicly on sk ! $aux = Enc_{pk'}(sk)$

$$Eval_{pk'}(h_c, aux) = Eval_{pk'}\left(h_c, Enc_{pk'}(sk)\right)$$

$$= Enc_{pk'}\left(h_c(sk)\right) = Enc_{pk'}\left(Dec_{sk}(c)\right)$$

$$= Enc_{pk'}(x)$$

hom. capacity of output:

$$d_{hom} - d_{h_c} = d_{hom} - d_{dec}$$

Bootstrapping [G09]

Given scheme with bounded d_{hom}.

How to extend

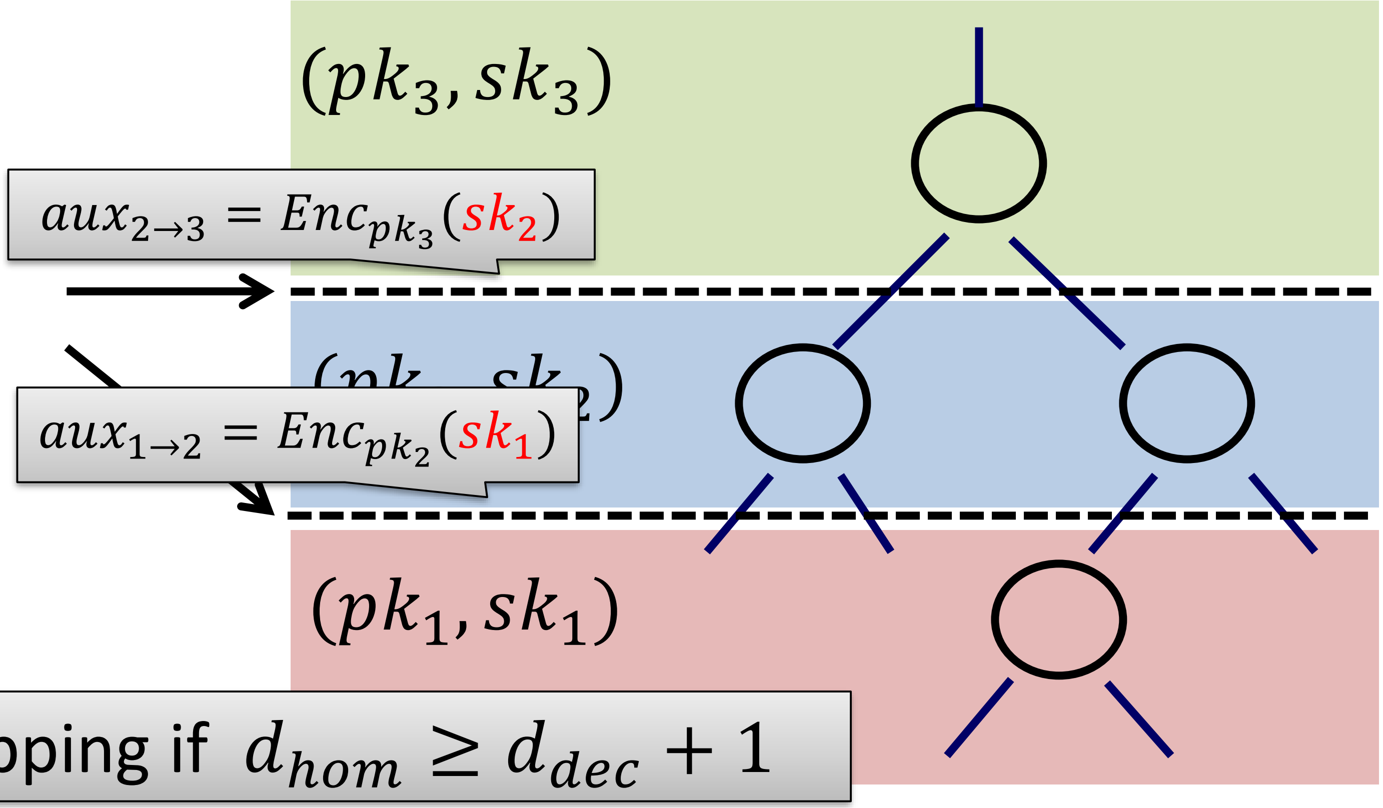
Downside: Need to generate many keys…

Idea: Do a few operations, then “switch” to a new instance

Switch keys

“cost” of d_{dec} hom. operations for switch

secure?



(pk_3, sk_3)
$aux_{2\to3} = Enc_{pk_3}(sk_2)$
(pk_2, sk_2)
$aux_{1\to2} = Enc_{pk_2}(sk_1)$
(pk_1, sk_1)

⇒ Bootstrapping if $d_{hom} \geq d_{dec} + 1$

Bootstrapping [G09]

Given scheme with bounded d_{hom}.
How to extend its homomorphic capability?

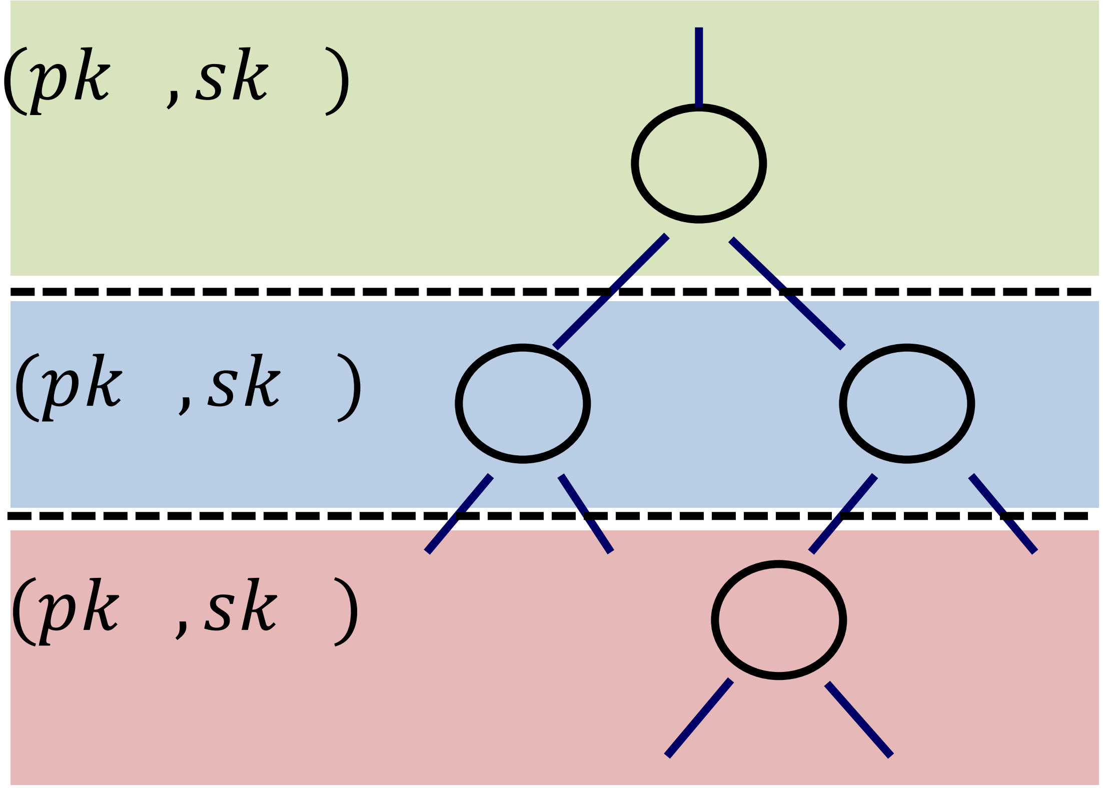
Idea: Do a few operations, then “switch” ~~to a new instance~~

$aux = Enc_{pk}(sk)$

switch from key to itself!

functionality of switching works

circular security required



(Some) Public Implementations of FHE

- HElib (IBM/NYU)
  - Ring-LWE (ideal-lattice) scheme of [BGV12], optimizations of [GHS12a]
  - https://github.com/shaih/HElib

- "Stanford FHE"
  - LWE scheme of [B12] with optimizations
  - http://cs.stanford.edu/~dwu4/fhe.html

- FHEW (UCSD)
  - Ring-LWE scheme of [DM14], built upon approximate eigenvector approach of [GSW13,BV14,AP14]
  - No batching but very fast bootstrapping
  - https://github.com/lducas/FHEW

So Where is That Homomorphic Google Search?

- Circuit model = huge overhead.
  - Inherent? Need to touch all elements to not leak.
- Bootstrapping is expensive.
  - No known alternative for deep computations.
- Memory requirements are huge (GBs).
  - Large ciphertexts, long keys.
  - Can “batch” to reduce overhead.

Thank You!