Achievable CCA2 Relaxation for Homomorphic Encryption

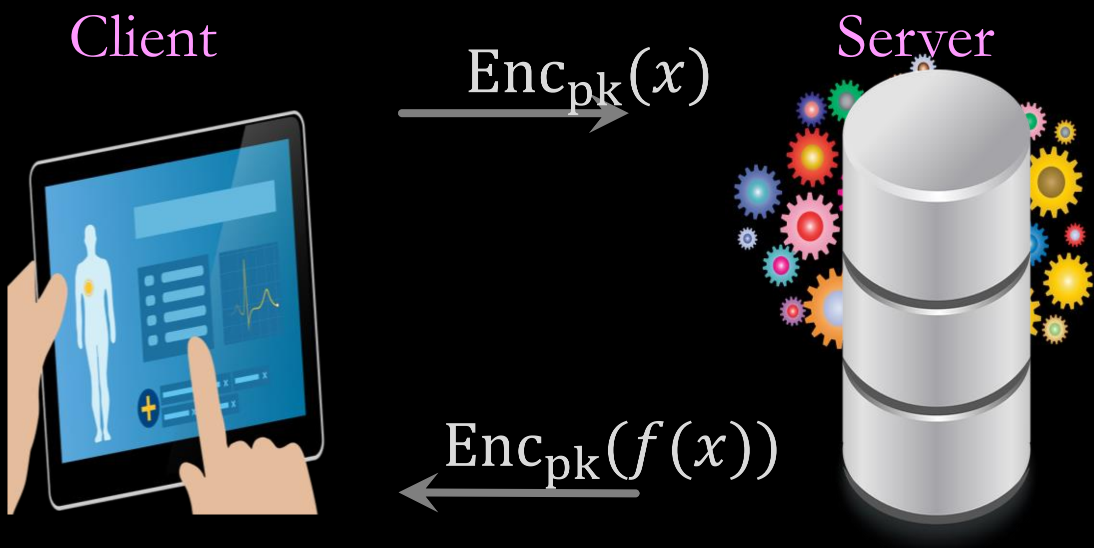
Adi Akavia	University of Haifa, Israel
Craig Gentry	TripleBlind, USA
Shai Halevi	Algorand Foundation, USA
Margarita Vald	Intuit Israel Inc., Israel

Secure outsourcing using Homomorphic Encryption (HE)

☑ Protects data in-use

☑ Low client complexity

☒ Deep computation is expensive
– e.g., refreshing

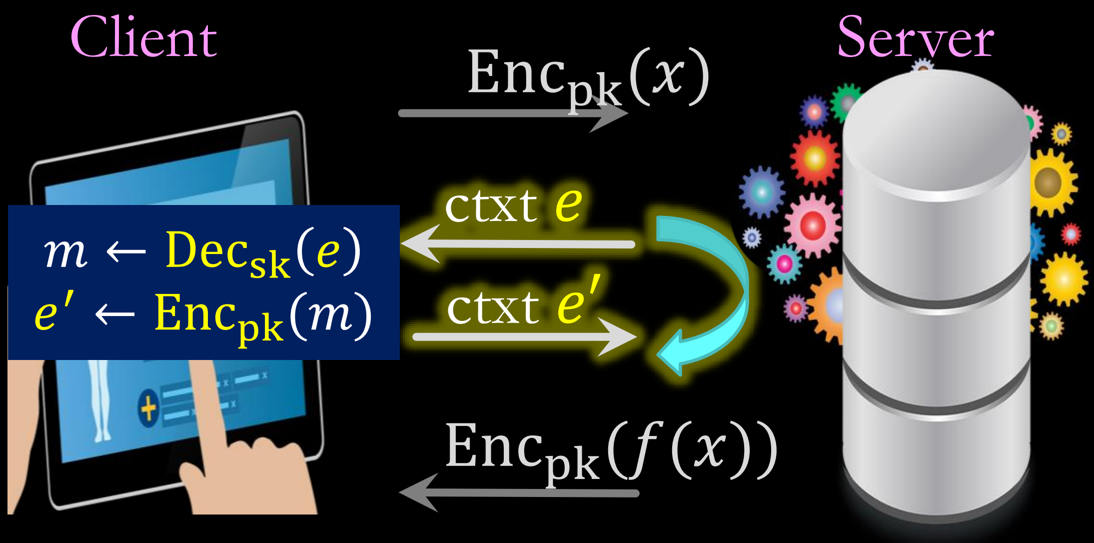


Client-aided secure outsourcing using HE

☑ Protects data in-use

☑ Low client complexity

☒ Deep computation is expensive
– **refreshing by client**, fast



*Q: privacy against **malicious** servers?*

Our Results I

on privacy against malicious server in client-aided protocols

Insufficiency: **CPA**-security does not guarantee privacy against **malicious** servers.

Define new notion – **funcCPA**, and **prove** it is:

strictly between CPA & CCA2

☑ **Sufficient** for privacy against **malicious** servers,

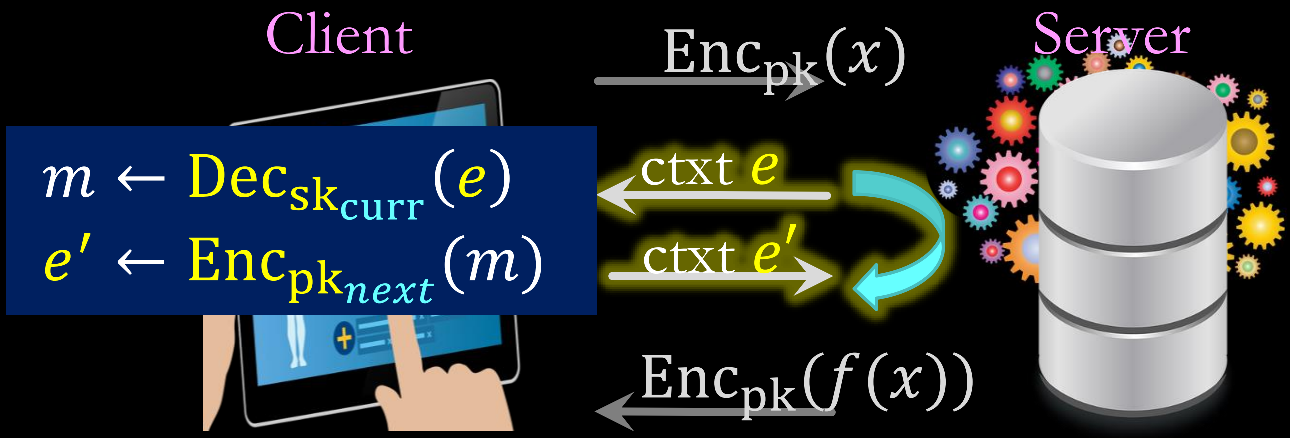
☑ **Achievable** from **circuit privacy**$^{+}$

Moreover, known schemes can be transformed to **circuit-private**$^{+}$

Our Results II

Can we prove existing HE scheme are funcCPA-secure?

Achievable: **leveled** BV, BGV, … are leveled **funcCPA**-secure.



Challenging: **funcCPA** implies **circular-security** for (non-leveled) BV and BGV

Insufficiency of CPA: Our Attack (simplified)

Theorem (Informal). Exist* **CPA**-secure PKE (*assuming ∃CPA-secure PKE)
so that client-aided outsourcing protocols instantiated with it
are vulnerable to **input-recovery attack** by malicious servers

Proof Idea: Starting from CPA-secure schemes, modify Enc, Dec as:

- $Enc'_{pk}(m)$: If m=sk, output 0|m
  --test by checking whether $Dec_m Enc_{pk}(r) = r$
  Otherwise, output $1|Enc_{pk}(m)$
- $Dec'_{sk}(c')$: Parse c' = b|c
  If b=0, output sk
  Otherwise, output $Dec_{sk}(c)$

Insufficiency of CPA: Our Attack (simplified)

Theorem (Informal). Exist* **CPA**-secure PKE (*assuming ∃CPA-secure PKE)
so that client-aided outsourcing protocols instantiated with it
are vulnerable to **input-recovery attack** by malicious servers

Proof Idea: Starting from CPA-secure schemes, modify Enc, Dec as:

- $\mathrm{Enc}'_{pk}(m)$: If $m=sk$, output $0|m$
  --test by checking whether $\mathrm{Dec}_m \mathrm{Enc}_{pk}(r) = r$
  Otherwise, output $1|\mathrm{Enc}_{pk}(m)$
- $\mathrm{Dec}'_{sk}(c')$: Parse $c' = b|c$
  If $b=0$, output sk
  Otherwise, output $\mathrm{Dec}_{sk}(c)$

Attacker sends
ct=(0|something)
to be re-encrypted

funcCPA-security: Definition & Sufficiency

Informal. funcCPA extends CPA by supporting **Refresh*** queries

*more generally Enc(g(Dec(c))

Theorem (informal). Client-aided protocols instantiated with a funcCPA-secure encryption guarantee **privacy** against **malicious** servers.

Pictorially: **CPA**-security Definition

Challenger		**Adversary**
$(pk,sk) \leftarrow Gen$	$pk \rightarrow$	
$b \leftarrow_R \{0,1\}$	$\leftarrow m_0, m_1$	
$c \leftarrow Enc_{pk}(m_b)$	$c \rightarrow$	
	$\leftarrow b'$	

CPA-security: $\forall$ppt adversary, $\Pr[b'=b] \leq \frac{1}{2} + negl$

Pictorially: **funcCPA**-security Definition

Challenger | **Adversary**

(pk,sk) $\leftarrow$ Gen —— pk ——>

$m \leftarrow \mathrm{Dec}_{sk}(e)$
$e' \leftarrow \mathrm{Enc}_{pk}(m)$

<—— ctxt e ——
—— ctxt e' ——>

b $\leftarrow_R$ {0,1} <—— m_0, m_1 ——

c $\leftarrow$ $\mathrm{Enc}_{pk}(m_b)$ —— c ——>

$m \leftarrow \mathrm{Dec}_{sk}(e)$
$e' \leftarrow \mathrm{Enc}_{pk}(m)$

<—— ctxt e ——
—— ctxt e' ——>

<—— b' ——

funcCPA-security: $\forall$ppt adversary, Pr[b'=b] $\leq$ ½ + negl

Pictorially: **Leveled funcCPA** Definition

Queries are answered by **next-level** ciphertexts

Challenger | **Adversary**

$(pk_t, sk_t)\}_t \leftarrow Gen$ — $\{pk\}_t$ →

$m \leftarrow Dec_{sk_{current}}(e)$
$e' \leftarrow Enc_{pk_{next}}(m)$

← ctxt e
ctxt e' →

$b \leftarrow_R \{0,1\}$ — ← $m_0, m_1, \mathbf{t}$

$c \leftarrow Enc_{pk_t}(m_b)$ — c →

$m \leftarrow Dec_{sk_{current}}(e)$
$e' \leftarrow Enc_{pk_{next}}(m)$

← ctxt e
ctxt e' →

← b’

leveled funcCPA-security: $\forall$ppt adversary, $Pr[b'=b] \leq$ ½ + negl

leveled **funcCPA**: Achievability by Existing Schemes

Theorem. Every **CPA**-secure **leveled HE** with **independent level keys** is **leveled funcCPA**-secure.

(pk_ℓ, sk_ℓ) at each level are **independent**

evk_ℓ computed from sk_ℓ & next-level $pk_{\ell-1}$

Observation: BV, BGV, B/FV (with a small modification) have independent level keys.

Proof Idea. Simulate answers to funcCPA queries by encryption of arbitrary message.

Indistinguishable views by (CPA-security and) level keys independence.

funcCPA: Achievability from **Circuit-Privacy$^+$**

Def (informal): A HE scheme E=(Gen, Enc, Dec, Eval) is **circuit-private$^+$** if

$$\textbf{Eval}_{pk}(C; c_1, \ldots, c_\ell) \approx \textbf{Enc}_{pk}(C(\mathrm{Dec}_{sk}(c_1), \ldots, \mathrm{Dec}_{sk}(c_\ell)))$$

where: **keys** – properly generated
ciphertexts – maliciously generated

Prior defs for circuit-privacy:

- **semi-honest:** both **keys & ciphertexts** – properly generated
- **malicious:** both **keys & ciphertexts** – maliciously generation

funcCPA: Achievability from **Circuit-Privacy$^+$**

Def (informal): A HE scheme E=(Gen, Enc, Dec, Eval) is **circuit-private$^+$** if

$$\mathbf{Eval}_{pk}(C; c_1, \ldots, c_\ell) \approx \mathbf{Enc}_{pk}(C(\mathrm{Dec}_{sk}(c_1), \ldots, \mathrm{Dec}_{sk}(c_\ell)))$$

where: **keys** – properly generated
ciphertexts – maliciously generated

Theorem: Suppose E is **CPA**-secure and **circuit-private$^+$** w.r.t $\mathcal{C}$,

Then E is **funcCPA** w.r.t $\mathcal{C}$.

Proof idea. Answer funcCPA queries using Eval. Indistinguishable by **circuit-privacy$^+$**

Construction: Circuit-Privacy^{+}

Theorem: Known HE schemes (e.g., BV and FHEW) can be transformed into **circuit-private^{+}**.

Proof: **Idea 1.** **Sanitize* Enc** and **Eval outputs** to make them stat. close.

Sanitization [DS16]: If $\mathrm{Dec}_{sk}(c_1) = \mathrm{Dec}_{sk}(c_2)$ (1)
Then $\mathrm{Sanitize}_{pk}(c_1) \approx_s \mathrm{Sanitize}_{pk}(c_2)$ (2)

Problem: **Eval** has **no correctness** guarantee on **malicious** inputs ciphertexts (i.e., no (1) and hence no (2))

Idea 2. **Sanitize** also **inputs** to **Eval** so, they are **stat. close** to **fresh** re-encryption (of some msg)

Conclusions

We propose new security notion – **funcCPA** – and show it is:

- Related to circular-security, though not known to be equivalent

- **Achievable:** 1) via generic transformation
  2) for existing (leveled) schemes

- **Sufficient** for **privacy** in client-aided protocols against **malicious** servers

Encryption	Type of client-aided protocol	Server
CPA	w. natural property	semi-honest
leveled **funcCPA**	next-level client's response	malicious
funcCPA	all	malicious

Open: Prove that fully hom. BGV, B/FV... are **funcCPA**, assuming circular-security.