Functional Encryption with Bounded Collusions via Multi-Party Computation

Sergey Gorbunov -- {U of Toronto}
Vinod Vaikuntanathan -- {U of Toronto}
Hoeteck Wee -- {George Washington U}
Public Key Encryption

Bob

Only Bob can decrypt and compute on m!

Alice

CT = Enc(PK, m)

Charlie

PK

SK
Public Key Encryption

How can we:
- Allow Charlie to learn a function $C$ of $m$?
- Ensure Charlie doesn’t learn more than $C(m)$?
- Without asking Bob to do the work (outsourcing)
- And without asking Bob to be online (availability)

$$CT = Enc(PK, m)$$
Public Key Encryption

How can we:

- Allow Charlie to learn a function $C$ of $m$?
- Ensure Charlie doesn’t learn more than $C(m)$?
- Without asking Bob to do the work (outsourcing)
- And without asking Bob to be online (availability)

$CT = Enc(PK, m)$
Public Key Encryption

How can we:
- Allow Charlie to learn a function $C$ of $m$?
- Ensure Charlie doesn’t learn more than $C(m)$?
- Without asking Bob to do the work (outsourcing)
- And without asking Bob to be online (availability)

$CT = Enc(PK, m)$

Alice

Bob

Charlie

$SK$

$PK$
Fully Homomorphic Encryption [Gentry 09]

How can we:
- Allow Charlie to learn a function $C$ of $m$?
- ensure Charlie doesn’t learn more than $C(m)$?
- without asking Bob to do the work (outsourcing)
- and without asking Bob to be online (availability)

$CT = Enc(PK, m)$

$H. Eval(C, CT) = Enc(C(m))$
Functional Encryption \cite{Boneh11, O'Neill10}

Allow Charlie to learn a function of $M$!
Functional Encryption [Boneh, Sahai, Waters 11] [O'Neill 10]

Allow Charlie to learn a function of $M$!

...Let $C$ be a family of circuits and $M$ be a message space

Bob

MSK

Alice

MPK

Charlie

C

GVW12

FE with Bounded Collusions via MPC
**Functional Encryption**  \([\text{BSW'11, O'N10}]\)

*Allow Charlie to learn a function of \(M\!*

...Let \(C\) be a family of circuits and \(M\) be a message space

\[
SK = \text{Keygen}(\text{MSK}, C)
\]

(Charlie no longer needs to communicate to Bob)
Functional Encryption

[BSW’11, O’N10]

Allow Charlie to learn a function of M!

...Let C be a family of circuits and M be a message space

Bob

Alice

Charlie

MSK

MPK

CT = Enc(MPK, m)

SK = Keygen(MSK, C)

Dec(SK, CT) = C(m)

GVW12
**Functional Encryption**  
[BSW’11, O’N10]

...Let $C$ be a family of circuits and $M$ be a message space.

Security:

Adv should not learn anything about $m$, except $C(m)$.

$SK = Keygen(MSK, C)$

$CT = Enc(MPK, m)$

$Dec(SK, CT) = C(m)$
$SK = \text{Keygen}(MSK, C)$

$C = \text{circuit opening urgent emails}$

$CT = \text{Enc}(MPK, email)$

$Dec(SK, CT) = \text{email if urgent} \downarrow \text{otherwise}$
Special Cases of FE

- Identity-Based Encryption [Sha84, BF01, Coc01, BW06]

\[ C_{id}(id', \mu) = \mu \text{ if } id = id' \]
\[ \perp \text{ otherwise} \]

- Fuzzy IBE [SW05]
- Attribute-Based Encryption [GPSW06, LOSTW10]
- Inner Product Predicate Encryption [KSW08, LOSTW10]
Can we construct functional encryption for all circuits?
Can we construct functional encryption for all circuits?

Yes we can!

with a small catch ...
Functional Encryption

Allow Charlie to learn $q$ functions of $M$ ($q$ is fixed before setup)

$CT = Enc(MPK, m)$

$SK_1, ..., SK_q$

$SK_i = Keygen(MSK, C_i)$

Security against $q$ – Bounded Collusions:
Adv should not learn anything about $m$, except $C_1(m), ..., C_q(m)$

$Dec(SK_1, CT) = C_1(m), ..., C_q(m)$
Functional Encryption

q-collusion security

\[ SK_1 = \text{Keygen}(MSK, C_1) \]

\[ SK_q = \text{Keygen}(MSK, C_q) \]

\[ SK_n = \text{Keygen}(MSK, C_n) \]

Colluding Advs shouldn’t learn anything about \( m \), except: \( C_1(m), \ldots, C_q(m) \)
Previous Work

$q$-collusion security

- Key-insulated public key cryptosystems
  [Dodis, Katz, Xu, Yung 02]
- Bounded CCA2
  [Cramer, Hanaoka, Hofheinz, Imai, Kiltz, Pass, Shelat, Vaikuntanathan 07]
- Bounded-collusion IBE
  [Goldwasser, Lewko, Wilson 12]
Our Result

**Theorem**: There exists a q-bounded non-adaptive simulation-secure FE scheme for all poly-size circuits, assuming:

- CPA-secure Public-key Encryption and
- PRGs computable in low-depth
Our Result

**Theorem**: There exists a q-bounded non-adaptive simulation-secure FE scheme for all poly-size circuits, assuming:
- CPA-secure Public-key Encryption and
- PRGs computable in low-depth

- Extends to adaptive for bounded # of messages
Our Result

**Theorem**: There exists a $q$-bounded non-adaptive simulation-secure FE scheme for all poly-size circuits, assuming:

- CPA-secure Public-key Encryption and
- PRGs computable in low-depth
- factoring
- discrete logarithm
- lattice problems
Our Result

**Theorem**: There exists a q-bounded non-adaptive simulation-secure **public index predicate encryption** scheme for all poly-size circuits, assuming:

- CPA-secure Public-key Encryption and
- PRGs computable in low depth
Our Result

**Theorem**: There exists a \textbf{q-bounded} non-adaptive simulation-secure FE scheme for all poly-size circuits, assuming:

- CPA-secure Public-key Encryption and
- PRGs computable in low-depth

Remark 1:

[Thm: Agrawal, G, Vaikuntanathan, Wee 12]

*For unbounded collusions, it is impossible to achieve non-adaptive simulation secure FE for all circuits.*
Our Result

**Theorem**: There exists a q-bounded non-adaptive simulation-secure FE scheme for all poly-size circuits, assuming:
- CPA-secure Public-key Encryption and
- PRGs computable in low-depth

Remark 2:

[Thm: Boneh, Sahai, Waters 11]  
*It is impossible to achieve adaptive simulation secure FE for all circuits.*  
*(many messages, 1 SK)*
**Theorem**: There exists a q-bounded non-adaptive *simulation-secure* FE scheme for all poly-size circuits, assuming:

- CPA-secure Public-key Encryption and
- PRGs computable in low-depth

**Remark 3:**

*Simulation* Security $\rightarrow$ IND security
Roadmap

1-FE for arbitrary circuits [Sahai, Seyalioglu 10]

Using MPC
[Ben-Or, Goldwasser, Wigderson 88]

q-FE for degree-d circuits

FE Bootstrapping Theorem: Using Randomized Encodings
[Applebaum, Ishai, Kushilevitz 05]
[Yao 86]

q-FE for arbitrary circuits
Roadmap

1-FE for arbitrary circuits [Sahai, Seyalioglu 10]

Using MPC
[Ben-Or, Goldwasser, Wigderson 88]

q-FE for degree-d circuits

Class of functions:

- Computes bounded degree polynomial

- For all $C \in C$, $C(\cdot)$ is $l$-variate polynomial over $\mathbb{F}$ of degree $d$
1-FE for all circuits

[Ciphertext CT]: A universal garbled circuit encoding $m$ [Yao 82]

[Secret key $SK^C$]: Set of input labels

It is correct but **NOT** secure for two sets of input labels! (i.e. **insecure for $q=2$**)

[Sahai, Seyalioglu 10]
q-bounded Collusions FE

C(•) is a degree d polynomial

Shamir's SS [Shamir 79]

**Important property:** Given two shares \( s_1(i) \) and \( s_2(i) \), we can perform computation over the shares! [Ben-Or, Goldwasser, Wigderson 88]

\[
s_1(i) + s_2(i) = (s_1 + s_2)(i) \quad \text{(additive homomorphism)}
\]

\[
s_1(i) \cdot s_2(i) = (s_1 \cdot s_2)(i) \quad \text{(multiplicative homomorphism)}
\]
q-bounded Collusions FE

C(•) is a degree d polynomial

Shamir’s SS [Shamir 79]

Important property: Given two shares $s_1(i)$ and $s_2(i)$, we can perform computation over the shares! [Ben-Or, Goldwasser, Wigderson 88]

\[ s_1(i) + s_2(i) = (s_1 + s_2)(i) \]  
(additive homomorphism)

\[ s_1(i) \times s_2(i) = (s_1 \times s_2)(i) \]  
(multiplicative homomorphism)

Catch:
Degree of the underlying polynomial increases with each multiplication!
q-bounded Collusions FE

\[ C(\bullet) \text{ is a degree } d \text{ polynomial} \]

**Parameters:** \( N = N(d, q), t = t(q) \), \( 1\text{-FE: (Setup}^1, \text{Keygen}^1, \text{Enc}^1, \text{Dec}^1) \)

**Setup:** Run Setup\(^1\) \( N \) times:

\[
\begin{array}{cccccc}
\text{MPK}_1 & \text{MPK}_2 & \text{MPK}_3 & \ldots & \text{MPK}_{N-1} & \text{MPK}_N \\
\text{MSK}_1 & \text{MSK}_2 & \text{MSK}_3 & \ldots & \text{MSK}_{N-1} & \text{MSK}_N \\
\end{array}
\]
q-bounded Collusions FE

\( C(\bullet) \) is a degree d polynomial

**Parameters:** \( N = N(d, q), t = t(q) \), 1-FE: (Setup\(^1\), Keygen\(^1\), Enc\(^1\), Dec\(^1\))

Setup: Run Setup\(^1\) \( N \) times:

Random subset \( S \) of secret keys \( \{MSK_i\} \) is chosen

Run Keygen\(^1\) on \( C \) for all \( MSK_i \) in \( S \)
q-bounded Collusions FE

\( C(\bullet) \) is a degree d polynomial

**Parameters:** \( N = N(d, q), t = t(q) \), 1-FE: (Setup\(^1\), Keygen\(^1\), Enc\(^1\), Dec\(^1\))

**Setup:** Run Setup\(^1\) N times:

\[
\text{Setup: Run Setup}\(^1\) N times:
\]

\[
\text{Keygen}_{\text{MSK}}(C): \quad sK_1^C \quad SK_2^C \quad \ldots \quad SK_{N-1}^C
\]

\[
\text{Enc}_{\text{MPK}}(m): \quad m_1 \quad m_2 \quad m_3 \quad \ldots \quad m_{N-1} \quad m_N
\]

\[
\downarrow \text{MPK}_1 \quad \downarrow \text{MPK}_2 \quad \downarrow \text{MPK}_3 \quad \ldots \quad \downarrow \text{MPK}_{N-1} \quad \downarrow \text{MPK}_N
\]

\[
\downarrow \text{CT}_1 \quad \downarrow \text{CT}_2 \quad \downarrow \text{CT}_3 \quad \ldots \quad \downarrow \text{CT}_{N-1} \quad \downarrow \text{CT}_N
\]

Share \( m \rightarrow (m_1, \ldots, m_N) \) using degree \( t \) polynomial

GVW12

FE with Bounded Collusions via MPC
q-bounded Collusions FE

$C(\bullet)$ is a degree d polynomial

**Parameters:** $N = N(d, q), t = t(q)$, 1-FE: (Setup¹, Keygen¹, Enc¹, Dec¹)

Setup: Run Setup¹ $N$ times:

Keygenₘₛₘₖ(C):

Encₘₚₖ(m):

Dec(CT,SKₖ):

$C(m_i) = \text{Dec}^1(SK_i, CT_i)$

GVW12
q-bounded Collusions FE

C(•) is a degree d polynomial

**Parameters**: $N = N(d, q)$

Setup: Run Setup$^1$ $N$ times:

- **Keygen**$^1$ MSK($C$):
  - SK$^1_k$

- **Enc**$^1$ MPK($m$):
  - CT$_1$
  - CT$_2$
  - CT$_{N-1}$
  - CT$_N$

- **Dec**(CT, SK$^C$):
  - C($m_1$)
  - C($m_2$)
  - C($m_{N-1}$)

C($m_2$) is a share of C($m$) -- by Homomorphism of Shamir’s Secret Sharing

Keygen$^1$, Enc$^1$, Dec$^1$
**q-bounded Collusions FE**

$C(\bullet)$ is a degree $d$ polynomial

**Parameters:** $N = N(d, q), t = t(q)$, 1-FE: $(\text{Setup}^1, \text{Keygen}^1, \text{Enc}^1, \text{Dec}^1)$

**Setup:** Run $\text{Setup}^1$ $N$ times:

- **Keygen**$_{\text{MSK}}(C)$:
  - $\text{SK}_1^C$
  - $\text{SK}_2^C$
  - $\text{SK}_{N-1}^C$
  - $\text{SK}_N^C$

- **Enc**$_{\text{MPK}}(m)$:
  - $m_1$
  - $m_2$
  - $m_{N-1}$
  - $m_N$

- **Dec**(CT, SK$^C$):
  - $C(m_1)$
  - $C(m_2)$
  - $C(m_{N-1})$
  - Secret Sharing of $C(m)$!
q-bounded Collusions FE

\( C(\bullet) \) is a degree d polynomial

**Parameters**: \( N = N(d, q), t = t(q) \), 1-FE: (Setup\(^1\), Keygen\(^1\), Enc\(^1\), Dec\(^1\))

Setup: Run Setup\(^1\) N times:

\[ \text{Setup}: \text{Run Setup}\(^1\) N \text{ times:} \]

\[ \text{Keygen}_{\text{MSK}}(C): \]

\[ \text{Enc}_{\text{MPK}}(m): \]

\[ \text{Dec}(\text{CT,SK}^C): \]

Reconstruct \( C(m) \) from the shares

**Diagram**:

- MPK\(_1\), MSK\(_1\) → SK\(_1^C\)
- MPK\(_2\), MSK\(_2\) → SK\(_2^C\)
- MPK\(_3\), MSK\(_3\) → SK\(_3^C\)
- \( \ldots \)
- MPK\(_{N-1}\), MSK\(_{N-1}\) → SK\(_{N-1}^C\)
- MPK\(_N\), MSK\(_N\) → SK\(_N^C\)

- \( m_1 \) → CT\(_1\)
- \( m_2 \) → CT\(_2\)
- \( m_3 \) → CT\(_3\)
- \( \ldots \)
- \( m_{N-1} \) → CT\(_{N-1}\)
- \( m_N \) → CT\(_N\)

\[ \text{C}(m_1) \rightarrow \text{C}(m_2) \rightarrow \text{C}(m_{N-1}) \]

GVW12

FE with Bounded Collusions via MPC
q-bounded Collusions FE

$C(\bullet)$ is a degree $d$ polynomial

**Parameters:** $N = N(d, q)$,

Setup: Run Setup$^1$ $N$ times:

- $\text{Setup}$: Run Setup$^1$ $N$ times:
  - $\text{Enc}_{MPK}(m)$:
    - $CT_{1} \rightarrow C(m_{1})$
    - $CT_{2} \rightarrow C(m_{2})$
    - $CT_{3} \rightarrow C(m_{3})$
    - $\ldots$
    - $CT_{N-1} \rightarrow C(m_{N-1})$
    - $CT_{N} \rightarrow C(m_{N})$

- $\text{Dec}(CT, SK^{C})$:
  - $C(m_{1}) \rightarrow C(m_{1})$
  - $C(m_{2}) \rightarrow C(m_{2})$
  - $\ldots$
  - $C(m_{N-1}) \rightarrow C(m_{N-1})$

- $\text{Keygen}_{MSK}(C)$:
  - $SK_{1}^{C} \rightarrow SK_{1}^{C}$
  - $SK_{2}^{C} \rightarrow SK_{2}^{C}$
  - $\ldots$
  - $SK_{N-1}^{C} \rightarrow SK_{N-1}^{C}$
  - $SK_{N}^{C} \rightarrow SK_{N}^{C}$

Correctness:

- $m_{i} = s(i)$, where $s()$ is degree $t$,
- $C(\bullet)$ is degree $d$,
- $C(s(i))$ is degree $dt$ polynomial,
- Give $dt+1$ SK’s

Reconstruct $C(m)$ from the shares

GVW12
q-bounded Collusions FE

\( C(\bullet) \) is a degree d polynomial

**Parameters**: \( N = N(d, q), t = t(q) \), 1-FE: (Setup\(^1\), Keygen\(^1\), Enc\(^1\), Dec\(^1\))

**Setup**: Run Setup\(^1\) \( N \) times:

**Keygen\(_{MSK}(C_1)\)**: \( SK_1^{C_1} \)

**Keygen\(_{MSK}(C_2)\)**:

**Enc\(_{MPK}(m)\)**: \( m_1 \rightarrow CT_1 \), \( m_2 \rightarrow CT_2 \), \( m_3 \rightarrow CT_3 \), ... \( m_{N-1} \rightarrow CT_{N-1} \), \( m_N \rightarrow CT_N \)

**Dec(CT,SK\(_C\))**: ...
q-bounded Collusions FE

C(•) is a degree d polynomial

Parameters: \( N = N(d, q) \), \( t = t(q) \), 1-FE: (Setup\(^1\), Keygen\(^1\), Enc\(^1\), Dec\(^1\))

Setup: Run Setup\(^1\) \( N \) times:

- Keygen\(_{\text{MSK}}(C_1)\): \( \text{sk}^c_1 \) to \( \text{SK}^{c_1}_2 \) to \( \text{SK}^{c_1}_{N-1} \) to \( \text{SK}^{c_1}_N \)
- Keygen\(_{\text{MSK}}(C_2)\):
- Enc\(_{\text{MPK}}(m)\): \( m_1 \) to \( \text{CT}_1 \) to \( \text{CT}_2 \) to \( \cdots \) to \( \text{CT}_{N-1} \) to \( \text{CT}_N \)
- Dec(CT,SK\(_C^C\)): \( \text{c}_1(m_1) \) to \( \text{c}_1(m_2) \) to \( \text{c}_2(m_2) \) to \( \cdots \) to \( \text{c}_1(m_{N-1}) \) to \( \text{c}_2(m_{N-1}) \)
q-bounded Collusions FE

$C(\bullet)$ is a degree d polynomial

**Parameters:** $N = N(d, q), t = t(q)$, 1-FE:

Setup: Run Setup$^1$ $N$ times:

**Keygen$_{MSK}(C_1)$:** $SK^1_{C_1}$

**Keygen$_{MSK}(C_2)$:** $SK^2_{C_2}$

**Enc$_{MPK}(m)$:** $m_1 \rightarrow CT_1$, $m_2 \rightarrow CT_2$, $m_3 \rightarrow CT_3$, ..., $m_{N-1} \rightarrow CT_{N-1}$, $m_N \rightarrow CT_N$

**Dec($CT, SK^C$):** $C_1(m_1)$, $C_2(m_2)$, $C_2(m_3)$, ..., $C_1(m_{N-1})$, $C_2(m_{N-1})$

Security (intuition): We are OK, given that the Decryptor learns $\leq t$ shares.
q-bounded Collusions FE

$C(\bullet)$ is a degree d polynomial

**Technical Problem 1:**

- Adversary learns shares $C(m_i)$, so the simulator must be able to simulate them. However, these are not random shares, so unclear how to simulate. (known problem in BGW)
q-bounded Collusions FE
\[ C(\bullet) \text{ is a degree } d \text{ polynomial} \]

**Technical Problem 1:**
- Adversary learns shares \( C(m_i) \), so the simulator must be able to simulate them. However, these are not random shares, so unclear how to simulate. (known problem in BGW)

**Solution**
- Randomize each share \( C(m_i) \) by adding random share \( r_i \) of 0
\[
C'(m_i||r_i) = C(m_i) + r_i
\]
q-bounded Collusions FE

$C(\bullet)$ is a degree d polynomial

**Technical Problem 1:**
- Adversary learns shares $C(m_i)$, so the simulator must be able to simulate them. However, these are not random shares, so unclear how to simulate. (known problem in BGW)

**Solution**
- Randomize each share $C(m_i)$ by adding random share $r_i$ of 0
  
  $C'(m_i||r_i) = C(m_i) + r_i$

**Technical Problem 2:**
- Adding random shares of 0 of the same polynomial creates correlation between shares of $C_1(m) \ldots C_q(m)$
q-bounded Collusions FE
C(•) is a degree d polynomial

**Technical Problem 1:**
- Adversary learns shares $C(m_i)$, so the simulator must be able to simulate them. However, these are not random shares, so unclear how to simulate. (known problem in BGW)

**Solution**
- Randomize each share $C(m_i)$ by adding random share $r_i$ of 0
  $$C'(m_i||r_i) = C(m_i) + r_i$$

**Technical Problem 2:**
- Adding random shares of 0 of the same polynomial creates correlation between shares of $C_1(m) ... C_q(m)$

**Solution**
- Add a $q$-wise independent random shares of 0
  $$C'_w(m||\vec{r}_i) = C(m_i) + \sum_{j \in w} r_i [j]$$
q-bounded Collusions FE

1-FE for arbitrary circuits [SS’10, Yao’86]

\[ \text{Using MPC [BGW’88]} \]

\[ \text{DONE!} \]

q-FE for degree-d circuits

\[ \text{FE Bootstrapping Theorem: Using Randomized Encodings} \]

\[ \text{[AIK’05,Yao’86]} \]

q-FE for arbitrary circuits
q-bounded Collusions FE

q-FE for degree-d circuits

FE Bootstrapping Theorem:
Using Randomized Encodings

[Applebaum, Ishai, Kushilevitz 05]
[Yao 86]

q-FE for arbitrary circuits

Idea: A function computing a randomized encoding for C is of low degree. (assuming low degree PRG)
[AIK05]
q-bounded Collusions FE

1-FE for arbitrary circuits \([SS’10, Yao’86]\)

q-FE for degree-\(d\) circuits

Using MPC \([BGW’88]\)

FE Bootstrapping Theorem: Using Randomized Encodings

\([AIK’05, Yao’86]\)

q-FE for arbitrary circuits

\(\text{DONE!}\)
q-bounded Collusions FE

1-FE for arbitrary circuits [SS’10, Yao’86]

\[ \begin{aligned}
&\text{DONE!} \\
&\text{q-FE for degree-d circuits}
\end{aligned}\]

\[ \begin{aligned}
&\text{DONE!} \\
&\text{q-FE for arbitrary circuits}
\end{aligned}\]

Using MPC [BGW’88]

FE Bootstrapping Theorem: Using Randomized Encodings [AIK’05, Yao’86]

Open Problems:
- IND-secure FE for all circuits (unbounded collusions)?
- New connections amongst MPC, ZK and FE?
Thank you!
Small Pairwise Intersection:
Let $S_1, S_2, \ldots, S_n \in [N]$. Want to make sure:

$$| \bigcup_{i \neq j} (S_i \cap S_j) | \leq t$$

Cover-Freeness:
Let $w_1, w_2, \ldots, w_n \in [N]$. Want to make sure:

For all $i \in [q]$, $w_i \setminus \bigcup_{i \neq j} w_j \neq \emptyset$
Class of functions:

- Deterministic

- Computes bounded degree polynomial

- \( M = \mathbb{F}^l \), for all \( C \),
  \( C(\cdot) \) is \( l \) – variate polynomial over \( \mathbb{F} \) of degree \( d \)

- Handles arithmetic and boolean circuits (Set \( \mathbb{F} \) to be a large extension of \( \mathbb{F}_2 \)) (constant fan-in)