Computing on Encrypted Data

Secure Internet of Things Seminar

David Wu
January, 2015
New Applications in the Internet of Things

Smart Homes

aggregation + analytics

usage statistics and reports

report energy consumption

Smart Homes
The Power of the Cloud

Question: provide service, preserve privacy

analytics recommendations personalization

lots of user information = big incentives
Secure Multiparty Computation (MPC)

Multiple parties want to compute a joint function on private inputs

at end of computation, each party learns the average power consumption

privacy guarantee: no party learns anything extra about other parties’ inputs

private input: individual power consumption
Two Party Computation (2PC)

- Simpler scenario: two-party computation (2PC)
- 2PC: Mostly “solved” problem: Yao’s circuits [Yao82]
  - Express function as a Boolean circuit
Two-Party Computation (2PC)

- Yao’s circuits very efficient and heavily optimized [KSS09]
  - Evaluating circuits with 1.29 billion gates in 18 minutes (1.2 gates / µs) [ALSZ13]

- Yao’s circuit provides semi-honest security: malicious security via cut-and-choose, but not as efficient
Going Beyond 2PC

- General MPC also “solved” [GMW87]

secret share inputs with all parties

jointly evaluate circuit, gate-by-gate
Secure Multiparty Computation

• General MPC suffices to evaluate arbitrary functions amongst many parties: should be viewed as a feasibility result

• Limitations of general MPC
  • many rounds of communication / interaction
  • possibly large bandwidth
  • hard to coordinate interactions with large number of parties

• Other considerations (not discussed): fairness, guaranteeing output delivery
This Talk: Homomorphic Encryption

GMW Protocol and General MPC

Many rounds of interaction
Boolean circuits (typically)

Interaction

Few rounds of interaction
Arithmetic circuits

Custom Protocols

General methods for secure computation

Homomorphic Encryption
Homomorphic Encryption

Homomorphic encryption scheme: encryption scheme that allows computation on ciphertexts

Comprises of three functions:

\[ \text{Enc}(m, pk) \rightarrow c \]
\[ \text{Dec}(c, sk) \rightarrow m \]

Must satisfy usual notion of semantic security
Homomorphic Encryption

Homomorphic encryption scheme: encryption scheme that allows computation on ciphertexts

Comprises of three functions:

\[
\begin{align*}
    c_1 &= \text{Enc}_{pk}(m_1) \\
    c_2 &= \text{Enc}_{pk}(m_2) \\
    c_3 &= \text{Eval}_f(c_1, c_2) \\
    \text{Dec}_{sk}\left(\text{Eval}_f(ek, c_1, c_2)\right) &= f(m_1, m_2)
\end{align*}
\]
Fully Homomorphic Encryption (FHE)

Many homomorphic encryption schemes:

- ElGamal: $f(m_0, m_1) = m_0 m_1$
- Paillier: $f(m_0, m_1) = m_0 + m_1$

Fully homomorphic encryption: homomorphic with respect to two operations: addition and multiplication

- [BGN05]: one multiplication, many additions
- [Gen09]: first FHE construction from lattices
Privately Outsourcing Computation

encrypted results of computation

encrypted data

Leveraging computational power of the cloud
Machine Learning in the Cloud

1. Publish public key
2. Upload encrypted values
3. Compute model homomorphically
4. Decrypt to obtain model

aggregation + analytics

report energy consumption
Machine Learning in the Cloud

- Passive adversary sitting in the cloud does *not* see client data
- Power company only obtains resulting model, not individual data points (assuming no collusion)
- Parties only need to communicate with cloud (the power of public-key encryption)
Big Data, Limited Computation

• Homomorphic encryption is expensive, especially compared to symmetric primitives such as AES

• Can be unsuitable for encrypting large volumes of data
"Hybrid" Homomorphic Encryption

Encrypt AES key using homomorphic encryption (expensive), encrypt data using AES (cheap)

Current performance: $\approx 400$ seconds to decrypt 120 AES-128 blocks (4 s/block) [GHS15]
Constructing FHE

• FHE: can homomorphically compute arbitrary number of operations

• Difficult to construct – start with something simpler: *somewhat homomorphic encryption scheme (SWHE)*

• SWHE: can homomorphically evaluate a few operations (circuits of low depth)
Gentry’s Blueprint: SWHE to FHE

• Gentry described general *bootstrapping* method of achieving FHE from SWHE [Gen’09]

• Starting point: SWHE scheme that can evaluate its own decryption circuit
Gentry’s Blueprint: From SWHE to FHE

Homomorphism Remaining: From SWHE to FHE

- recrypt functionality
- ciphertext
- encrypt the ciphertext
- encryption of secret key
- homomorphically evaluate the decryption function

Many operations remaining

No operations remaining

Homomorphism Remaining
Bootstrappable SWHE

- First bootstrappable construction by Gentry based on ideal lattices [Gen09]

- Tons of progress in constructions of FHE in the ensuing years [vDGHV10, SV10, BV11a, BV11b, Bra12, BGV12, GHS12, GSW13], and more!

- Have been simplified enough that the description can fit in a blog post [BB12]
Conceptually Simple FHE [GSW13]

• Ciphertexts are $n \times n$ matrices over $\mathbb{Z}_q$
• Secret key is a vector $\nu \in \mathbb{Z}_{q}^{n}$

Encryption of $m$ satisfies above relation

$v$ is a “noisy” eigenvector of $C$
Conceptually Simple FHE [GSW13]

• Suppose that $v$ has a “large” component $v_i$

\[
C_i \times v = m \times v + e
\]

- ciphertext
- secret key
- message
- noise

• Can decrypt as follows:

\[
\left\langle C_i, v \right\rangle \frac{v_i}{v_i} = \left\langle m v_i + e_i, v \right\rangle = m
\]

Relation holds if $\left| \frac{e_i}{v_i} \right| < \frac{1}{2}$

$C_i$ is $i^{th}$ row of $C$
Conceptually Simple FHE [GSW13]

Homomorphic addition

\[ C_1 \times v = m_1 \times v + e_1 \]

\[ C_2 \times v = m_2 \times v + e_2 \]

\[ C_1 + C_2 \times v = (m_1 + m_2) \times v + e_1 + e_2 \]

Homomorphic addition is matrix addition

Noise terms also add
Conceptually Simple FHE [GSW13]

Homomorphic multiplication

\[
\begin{align*}
\mathbf{C}_1 \times \mathbf{v} &= m_1 \times \mathbf{v} + e_1 \\
\mathbf{C}_2 \times \mathbf{v} &= m_2 \times \mathbf{v} + e_2
\end{align*}
\]

\[
(C_1 C_2)v = (m_1 m_2)v + C_1 e_2 + m_2 e_1
\]

Homomorphic multiplication is matrix multiplication.

\[
\text{noise could blow up if } C_1 \text{ or } m_2 \text{ are not small}
\]
Conceptually Simple FHE [GSW13]

• Basic principles: ciphertexts are matrices, messages are approximate eigenvalues

• Homomorphic operations correspond to matrix addition and multiplication (and some tricks to constrain noise)

• Hardness based on learning with errors (LWE) [Reg05]
The Story so Far...

• Simple FHE schemes exist

• But... bootstrapping is expensive!
  • At 76 bits of security: each bootstrapping operation requires 320 seconds and 3.4 GB of memory [HS14]
  • Other implementations exist, but generally less flexible / efficient

• SWHE (without bootstrapping) closer to practical: can evaluate shallow circuits
Application: Statistical Analysis

- Consider simple statistical models: computing the mean or covariance (for example, average power consumption)

- Problem: given $n$ vectors $x_1, \ldots, x_n$, compute
  - Mean: $\mu = \frac{1}{n} \sum_{i=1}^{n} x_i$
  - Covariance: $\Sigma_X = \frac{1}{n^2} (nX^T X - \mu \mu^T)$
Application: Statistical Analysis

- Can also perform linear regression: given design matrix $X$ and response vector $y$, evaluate normal equations
  $$\theta = (X^TX)^{-1}X^Ty$$
- Matrix inversion (over $\mathbb{Q}$) using Cramer’s rule
- Depth $n$ for $n$-dimensional data
Batch Computation [SV11]

Algebraic structure of some schemes enable encryption + operations on vectors at no extra cost

Plaintext Space: ring $R$

Chinese Remainder Theorem: $R \cong \bigotimes_{i=1}^{k} R_{p_i}$
Batch Computation [SV11]

Encrypt + process array of values at no extra cost:

\[
\begin{array}{cccc}
1 & 2 & 3 & 4 \\
\hline
7 & 5 & 3 & 1 \\
\hline
8 & 7 & 6 & 5
\end{array}
\]

One homomorphic operation

In practice: \( \geq 5000 \) slots
Based on implementation of Brakerski’s scheme [Bra12]
Time to Perform Linear Regression on Encrypted Data (2 Dimensions)

- Few ciphertexts due to batching
- Multiplications dominate
Application: Private Information Retrieval

I want to see record $i$...

client learns record $i$, server learns nothing
PIR from Homomorphic Encryption [KO97]

represent database as matrix

query is an encrypted basis vector

server evaluates inner product

database components in the clear: additive homomorphism suffices
PIR from Homomorphic Encryption

- $O(\sqrt{n})$ communication with additive homomorphism alone
- Naturally generalizes:
  - $O(3\sqrt{n})$ with one multiplication
  - $O(k\sqrt{n})$ with degree $(k - 1)$-homomorphism
- Benefits tremendously from batching
PIR from Homomorphic Encryption

• Outperforms trivial PIR for very large databases

• However, recursive KO-PIR with additive homomorphism is still state-of-the-art
Concluding Remarks

- Internet of Things brings many security challenges
- Many generic cryptographic tools: 2PC, MPC, FHE
  - 2PC/MPC work well for small number of parties
  - SWHE/FHE preferable with many parties (IoT scale)
- FHE still nascent technology – should be viewed as a “proof-of-concept” rather than practical solution
- SWHE closer to practical, suitable for evaluating simple (low-depth) functionalities
- Big open problem to develop more practical constructions!
Questions?