New Advances in Secure RAM Computation

Sanjam Garg
University of California, Berkeley

Based on joint works with

Steve Lu, Payman Mohassel, Charalampos Papamanthou, Rafail Ostrovsky and Alessandra Scafuro
Yao’s garbled circuits

User

\[ C \]

Server

\[ C(x) \]

\[ x \]

\[ C(x) \]
If the running time of the program $P$ is $T$ then the corresponding circuit is of size $T^3$.

Communication complexity and computational complexity of both parties grows with $T^3$. 

RAM analogue of Garbled circuits
More Ambitious: Garbled RAM
[LO13, GHLORW14]

- Size of garbled database is $\tilde{O}(|D|)$
- Communication and computation cost grows in $\tilde{O}(T_i)$
More Ambitious: Garbled RAM
[LO13,GHLORW14]

Garbled circuits lead to a solution where the communication and computational cost per program grows with database size.

- Full-security: Server learns nothing but the output
- Unprotected Memory Access (UMA): Server learns access pattern.

ORAM [Goldreich-Ostrovsky]
Putting in context – Secure Computation

• Traditional protocols – have large round complexity
  • Linear in running time [OS97, GKKKMR12 ...]

• Seeking an analogue of Yao’s garbled circuits
  • Non-interactive
Landscape: Garbled RAM

• Heuristic construction from OWFs [LO13]
  • Circularity Issue
  • Fixed using IBE [GHLORS14]
• Construction from OWFs [GLOS15]
• Using only black-box use of OWFs [GLO15]
  • OWF can’t be modeled as a random oracle

• Not talk about succinct constructions based on iO [CHJV14, BGT14, LP14, KLW15, CH15, CCCLLZ15...]
Outline of the rest of the talk

• RAM model
• LO13 approach
• Technical bottleneck in realizing black-box construction
• High level idea of black-box construction [GLO15]
• Extensions [GMP15, GM15, GGMP15, GP15]
RAM Model

Writes require additional work but let’s ignore that!
LO13 approach

Use garbled circuits!
LO13 approach

1) Somehow encrypt memory
2) translate table
LO13 approach

STEP 1: garbling/encrypting of the memory

\[ i \mapsto PRF_K(i, b_i) \]

PRF key \( K \) to garble
LO13 approach

STEP 2: translate table

\[ i \rightarrow PRF_K(i, b_i) \]

- PRF key $K$ to garble

- $Enc(PRF_K(j, 0), s_0)$

- $Enc(PRF_K(j, 1), s_1)$
Technical Bottleneck in Black-Box

• The data needs to be encrypted so that the server doesn’t learn it!

• CPU step garbled circuits need to decrypt the read values internally
  • Need of black-box use of cryptography seems inherent
GLO15 high level idea

• Garbled memory comprises of a collection of garbled circuits with data values hardwired in them

• Read implemented by a sub-routine call
  • Control flow is passed to memory circuits
GLO15 – for one read only

\[ j, s_0, s_1 \]

\[ b_1 \quad b_2 \quad \ldots \]
GLO15 – for one read only

Say $j = 2$

$\begin{align*}
   j, s_0, s_1 \quad \text{CPU step 1} \\
   j, s_0, s_1 \quad \text{CPU step 2} \\
   b_1 \quad \text{Outputs} \quad s_{b_2} \\
   \ldots \ldots \\
   \text{Memory no longer useful!}
\end{align*}$
**GLO15 – for $m$ reads only**

Say $j = 2$

Assume uniform memory accesses.

How many backups? How do we connect them?

Outputs $s_{b_2}$
Conclusion and Open Problems

• Secure Computation for RAM programs
  Round Efficient
  And Black Box
• Important for crypto for big data
• Theoretically practical secure computation.
Thanks!