Practical Dynamic Composition in Secure Computation: VoIP and SQL
Mind the Gap!

Dave Archer
Galois, Inc.
Face Matching (LSS, FHE)

LSS
Peter, Tews, Katzenbeisser
5 seconds per Face @ 200 Face DB

EigenFaces

Troncoso-Pastoriza and Perez-Gonzalez
12 seconds to match
Scanning Encrypted e-Mail

Trainable Bayesian SPAM filter @ ~256B / < 1 minute
Bogdanov et al.

FHE string match detection @ 256 bytes / 10 sec
Rohloff, Cousins, et al.

LSS regexp match detection @ kByte / 30 sec
Galois

GC regexp match detection
Shelat et al.

Mail server

1 or 3 servers

© Galois, Inc. 2015
**Linear Regression**

**3D Linear Regression**
10k points in ~5 seconds
Galois

**SHE linear regression**
10k points in 280 seconds
Boneh at al.

HE to aggregate inputs, GC to solve CHOLESKY DECOMPOSITION
100M 20D ridge regression in 10 hours
Nikolaenko, Weinsberg, et al.

\[
m = \frac{(n \sum xy) - (\Sigma x \Sigma y)}{(n \sum x^2) - (\Sigma x)^2}
\]

\[
b = \frac{(\Sigma y \sum x^2) - (\Sigma x \sum xy)}{(n \sum x^2) - (\Sigma x)^2}
\]

\[y = mx + b\]
Garbled circuits with outsourcing to Cloud servers
Butler, Traynor et al.
### Satellite Conjunction Analysis

**2-party**

**Ostrovsky and Lu**

**LSS - bulk analysis & Heat map**

**Bogdanov et al.**

\[
P = \frac{1}{\sqrt{8\pi \sigma_x}} \int_{-\infty}^{\infty} \left[ \frac{1}{\sqrt{2\pi \sigma_y}} \right] \exp \left( \frac{-(x - x_0)^2}{2\sigma_x^2} \right) dx
\]

<table>
<thead>
<tr>
<th>Individual</th>
<th>Sharemind (Bulk)</th>
<th>SPACE (Bulk)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Object A</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Position</td>
<td>587</td>
<td>249</td>
</tr>
<tr>
<td>Velocity</td>
<td>-2</td>
<td>0</td>
</tr>
<tr>
<td>Error</td>
<td>67382</td>
<td>33695</td>
</tr>
<tr>
<td>Radius</td>
<td>39</td>
<td></td>
</tr>
<tr>
<td><strong>Object B</strong></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Position</td>
<td>260</td>
<td>307</td>
</tr>
<tr>
<td>Velocity</td>
<td>10</td>
<td>5</td>
</tr>
<tr>
<td>Error</td>
<td>29607</td>
<td>26419</td>
</tr>
<tr>
<td>Radius</td>
<td>19</td>
<td></td>
</tr>
</tbody>
</table>

Plaintext result: 0.9% (computed in 0.00 sec)

![Conjunction Analysis (on deathstar.galois.com)](image_url)

DARPA DISTRIBUTION STATEMENT A: Approved for public release: Distribution unlimited
And so on...

- Distributed Shared HMAC verification
- Distributed Shared Digital Signatures
- Secure statistical analytics for Differential Privacy Computation
- ...
End-to-End Encrypted VoIP Conferencing

FHE
Rohloff, Cousins, et al.

LSS - 4 voices @ streaming 12kb/s audio
Galois

VoIP coordinator (modified uMurmur)

Encrypted

16kHz audio: 1440 compressed 8-bit samples every 90 ms

Encrypted

9 MPC servers or 1 FHE server

Amazon ECS

© Galois, Inc. 2015

DARPA DISTRIBUTION STATEMENT A: Approved for public release: Distribution unlimited
Problem: variable sample count in each fixed input window

Inbound voice packets
asynchronous arrival
varying # of parties
90ms accumulation window

Secret-shared lookup table
accumulator

Outbound processed packets

Solution: “Help out” the cryptographic solution:
Impose external control loop and persistent state
Error-prone, awkward and complicated
PDaaS

- only by people I authorize
- only under conditions I allow
- never for evil, and
- only when documented transparently
SELECT 
  ship_name,
  stype AS ship_type,
  count(*) AS Number_Of_Berths,
  distance/speed AS eta
FROM ship
  JOIN track ON ship.id = track.ship_id
  JOIN shiptype ON ship.shiptype_id = shiptype.id
  JOIN hasberth ON ship.id = hasberth.ship_id
  JOIN berth ON berth.id = hasberth.berth_id,
  community
WHERE (community.community_name = 'Bajor') AND (occupied = FALSE) AND (ship_time = 25)
GROUP BY ship_name, stype, ship_lat, ship_lon, community_lat, community_lon, cruisingspeed
ORDER BY eta;
Faster would be better...

SPJU SQL
no recursion
no nested...yet

Q1, Auth

Maybe Answer
Partial Query Processing in SPDZ

Decrypt to Shares

Sort

Join

Select

Project

Aggregate and apply differential privacy

Encrypt for user

Large record counts from multiple relations

SPDZ “plaintext RAM” computation model
- Arbitrary RAM computation (registers and heap)
- Does not hide access pattern (does it matter?)
- Memory statically allocated at compile-time
- Unbounded updates to registers and heap
- Direct/indirect heap access, direct register access
- Unbounded re-use and update during program

- Reactive: inputs and program based on prior outputs
- Sub-linear access - not all data need be touched
- Loops need not be unrolled — reduced program size

A bit racy (in leakage), but simpler than unrolling circuits
Simple-minded Conclusions

- RAM model often appears useful in “real-world” computations
  - Dynamic iteration count in loops
  - Sub-linear access cost for data
  - Reactive computation during run-time
- ORAM model may be too rich in security, too slow in practice
- Circuit model impractical for realistic data sizes, dynamic computation