Outsourcing Two-Party Privacy-Preserving Computation

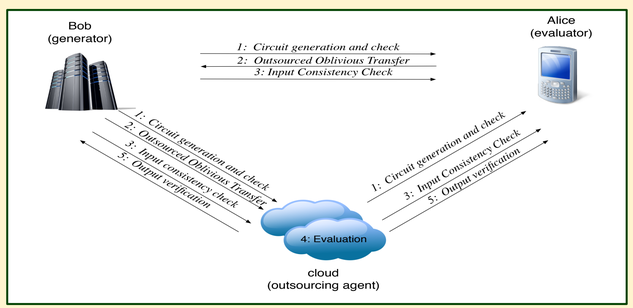
Benjamin Mood and Kevin Butler – Computer and Information Science Department, University of Oregon

with Hank Carter and Patrick Traynor (Georgia Tech).

Let's imagine the Secret Service wants a way to compute the shortest and safest path for the inauguration taking into account various hazards on the road but they don't want to reveal their location to the CIA. The CIA knows the various hazards, however their knowledge is classified. We can use privacy-preserving two party computation to perform a shortest path computation and keep the hazards and location a secret but still get the shortest and safest path.

We created a mapping application where one party can find a safe route from a phone navigating around hazards entered by another party and no information is leaked about the route or hazards. To use a mobile phone, we securely outsource the majority of the computation away from the mobile device in a privacy preserving manner.

OUTSOURCING COMPUTATION



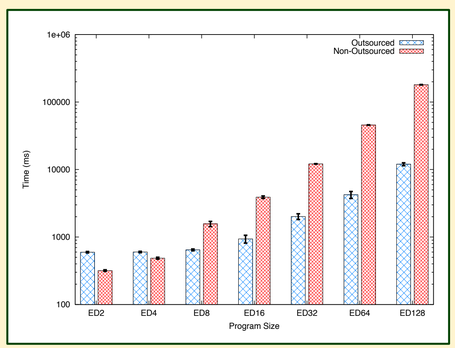
We outsource the computation by modifying a typical privacy preserving protocol to include a new way to distribute input to the function and a new way to check the correctness of the program where the phone verifies the correctness of the computation by the other two parties. The complete new protocol is showed in the above diagram. The main difficulty in outsourcing the computation is protecting the privacy of the two primary users while using a powerful cloud. Our protocol protects the privacy of the computation between all three parties and is secure in the malicious model assuming no collusion between the cloud and Bob, the non-mobile phone party. The malicious model takes into account adversaries who will do anything to subvert the protocol.

IMPLEMENTATION DETAILS

We use garbled circuits as our preserving privacy construct. Our implementation runs the evaluation N times, for malicious security, splitting into check and actual evaluations of the function. A larger N gives more security. These N evaluations are parallelized using OpenMPI. We used an implementation where all aspects of the protocol are secure in the malicious model: the way input is gathered, the way input consistency is verified across multiple evaluations, and how we verify the function is correct. Using these methods we know both parties input is consistent, the correct function is evaluated, and the output is correct.

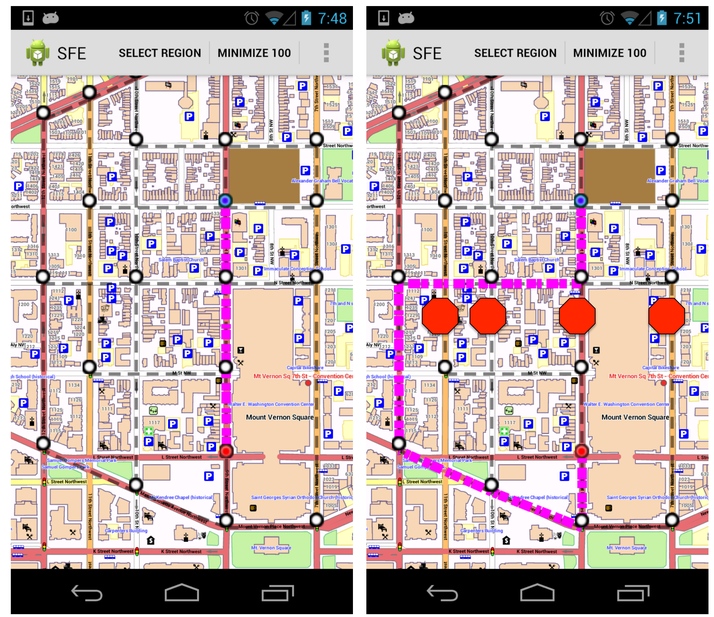
We modified the original implementation to work in serialization for the mobile phone. Difficulties of our implementation include porting the cryptography library PBC to our Android phone and correctness of the new outsourced version of the implementation.

EVALUATION OF OUTSOURCING



In this figure, in log scale, we compare the runtime of the outsourced protocol to the run time of the non-outsourced protocol on a phone. We use a test program which compares strings of length N where N is a power of 2 from 2 to 128. As can be seen in the figure above, our new outsourced protocol is many times faster than the previous non-protocol.

MAPPING APPLICATION



We were able to perform a shortest path calculation with 20 intersections in about 25 seconds on our test phone, an Android Galaxy Nexus. For our tests we minimized the number of intersections from 53, in this region, down to 20. To this end only intersections with dots on top of them are included into the computation. Streets which may be used in the computation are represented by a grey dashed line. We also can evaluate a 100 intersection computation.

The left figure indicates the route without hazards which is the fastest and most direct route. The right figure indicates the route when hazards are included in the computation. In the right figure there were hazards added between M and N street on all streets other than on 12th street for a total of four hazards. These hazards are denoted with red octagons.

Research supported by DARPA

For further information, Benjamin Mood (*bmood@cs.uoregon.edu*).