Private Information Retrieval (PIR) against FoundationDB

Mentor: Dan Lambright <dlambrig@gmail.com>
Open Source: yes

Skills

- Required
  - Java
  - Interest in cryptographic protocols and/or databases
  - Ability to do iterative debugging in distributed environments
- Nice to have: C++

Description

Path ORAM (Oblivious RAM) is a data structure and technique that can be used to implement Private Information Retrieval (PIR) protocols. PIR is a cryptographic technique that allows a user to retrieve data from a database without revealing which specific data item they are requesting.

Encryption will hide the contents of data on a network, making any captured traffic inscrutable. But without PIR, *access patterns* to a database on a server can still reveal data. This attack may be done using "traffic analysis" or "access pattern analysis," whereby an adversary analyzes the *patterns* of data accesses, such as the timing, frequency, and volume of requests, to infer sensitive information.

Path ORAM is one way to address this. The algorithm continuously reshuffles data blocks randomly throughout a tree structure; each data block is mapped to a random leaf node. The mapping is stored in a "position map" on the client. When a block is accessed, it is fetched into the client's local cache, then the cached block plus dummy blocks are shuffled and written back to random locations in the tree (the position map is also updated). Over time, the constant random shuffling hides any observable correlation between access patterns and physical data locations, ensuring privacy. This comes with the cost of high network and storage overhead due to the tree size.

Objectives

- Implement the path ORAM algorithm as an access method to FoundationDB. This will include client and server code that will be a front end to FoundationDB.
- Implement the FoundationDB API so that it works with the pathORAM front end. "Put", "get", "range read", and "clear range" should be implemented.

- Replicate an attack that leverages access patterns.
- Understand the overhead pathORAM incurs, and explore how it could be mitigated.

Mentorship

Your mentor will meet with you weekly and help setup FoundationDB, discuss PathORAM, etc. Our first meeting can be in person.

Team breakdown

PathORAM algorithm has open source implementations in Java. They can be enhanced to write to FoundationDB. The major FoundationDB APIs should be supported, including range reads, put, and get, etc, so applications can just link to the front end libraries and run.

Work could be divided between the client and server. Someone should also carry out sophisticated attacks that discover data using access patterns.

It will be necessary to measure the overhead in using pathORAM for various operations.

References

The original paper
https://people.csail.mit.edu/devadas/pubs/PathORam.pdf

A simplified explanation
https://research.kudelskisecurity.com/2020/04/22/an-introduction-to-oblivious-ram-oram/