Chair of Network Architectures and Services
Department of Informatics
Technical University of Munich

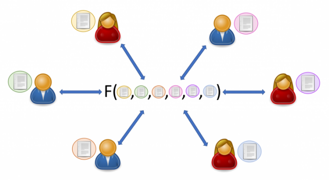
Thesis M.Sc.

IDP

Implementing and Benchmarking Secure Multiparty Computation Protocols with DPDK

Motivation

Secure Multiparty Computation enables parties to execute functions on obliviously shared inputs without revealing them. Consider multiple hospitals that want to study the adverse effects of a certain medication based on their patient's data. While joining these datasets could enable more statistically significant results, hospitals might be prohibited to share their private patient data with each other. Secure Multiparty Computation enables these hospitals to perform the computation on obliviously shared data and only reveal the final result of the computation performed. A popular approach to enable Secure Multiparty Computation is designing an addition and multiplication protocol based on a secret sharing (SS) scheme. With those two protocols in place, any function can be represented as a circuit consisting of addition and multiplication gates. While most protocols allow the parties to evaluate additions gates non-interactively, multiplication gates requires the parties to exchange a certain number of elements. Before continuing to evaluate a circuit each party must typically wait until it receives those required elements before proceeding with the next circuit's layer. Thus, a Secure Multiparty Computation protocol is bottlenecked by both network latency and network bandwidth between peers.



The Data Plane Development Kit (DPDK) is is an open-source collection of libraries and drivers for high performance packet processing. DPDK enables direct access to the network interface and bypasses the regular Linux kernel networking stack. This enables a high-performant and low-latency communication and is therefore well-suited for communication heavy protocols, like secret sharing SMC schemes. DPDK applications process whole IP-packet and therefore the programming scheme differs from usual socket programming, where application layer messages are exchanged.

Goal of this thesis is to identify a suitable SMC protocol candidate that fits the DPDK workflow. This candidate is to be implemented as an DPDK application and its performance to be compared to a traditional socket implementation.

Tasks

- Get familiar with Secure Multiparty Computation Protocols and DPDK
- Implement a Secure Multiparty Computation Protocol as a DPDK application
- Benchmark the runtime of your implementation in different network settings
- Experience with Linux and C/C++ programming is required

Sources

- https://eprint.iacr.org/2020/300
- https://www.dpdk.org/

Contact

Christopher Harth-Kitzerow	christopher.harth-kitzerow@tum.de
Manuel Simon	simonm@net.in.tum.de
Eric Hauser	hauser@net.in.tum.de