Engineering Privacy-Friendly Computations

George Danezis[1,2]

[1] University College London
[2] Microsoft Research, Cambridge

Abstract. In the past few years tremendous cryptographic progress has been made in relation to primitives for privacy friendly-computations. These include celebrated results around fully homomorphic encryption, faster somehow homomorphic encryption, and ways to leverage them to support more efficient secret-sharing based secure multi-party computations. Similar break-through in verifiable computation, and succinct arguments of knowledge, make it practical to verify complex computations, as part of privacy-preserving client side program execution. Besides computations themselves, notions like differential privacy attempt to capture the essence of what it means for computations to leak little personal information, and have been mapped to existing data query languages.

So, is the problem of computation on private data solved, or just about to be solved? In this talk, I argue that the models of generic computation supported by cryptographic primitives are complete, but rather removed from what a typical engineer or data analyst expects. Furthermore, the use of these cryptographic technologies impose constrains that require fundamental changes in the engineering of computing systems. While those challenges are not obviously cryptographic in nature, they are nevertheless hard to overcome, have serious performance implications, and errors open avenues for attack.

Throughout the talk I use examples from our own work relating to privacy-friendly computations within smart grid and smart metering deployments for private billing, privacy-friendly aggregation, statistics and fraud detection. These experiences have guided the design of ZQL, a cryptographic language and compiler for zero-knowledge proofs, as well as more recent tools that compile using secret-sharing based primitives.