Secure Multiparty Computation: Background

Yehuda Lindell
Bar-Ilan University and Dyadic Security
Secure Multiparty Computation

- A set of two or more parties with private inputs
- Parties wish to jointly compute a function of their inputs so that certain security properties are preserved
  - Privacy
  - Correctness
- Properties must be ensured even if some of the parties attack the protocol
- Great generality
Security Requirements

- Consider collaboration between competing companies
  - Two companies wish to see how many clients they have in common, or how much revenue is generated by clients in common (or not in common)
  - Maybe for due diligence, or combined advertising

- Adversarial threats
  - An adversary may try to learn the other company’s client information (breach of privacy)
  - An adversary may wish to have the result show many clients in common (breach of correctness)
General Security Properties

- **Privacy**: only the output is revealed
- **Correctness**: the function is computed correctly
- The actual definition generalizes this and compares a real execution to an ideal setting
Modeling Adversaries

- **Adversarial behavior**
  - **Semi-honest:** follows the protocol specification
    - Tries to learn more than allowed by inspecting transcript
  - **Malicious:** follows any arbitrary strategy
  - **Covert:** follows any arbitrary strategy, but is averse to being caught...
Defining Security: the Ideal/Real Paradigm

- What is the best we could hope for?
  - An incorruptible trusted party
  - All parties send inputs to trusted party (over perfectly secure communication lines)
  - Trusted party computes output
  - Trusted party sends each party its output (over perfectly secure communication lines)
  - This is an ideal world

- What can an adversary do?
  - Just choose its input...
Defining Security: the Ideal/Real Paradigm

- Definition of Security (informal):
  A real-world protocol is secure if it behaves (input/output) like an ideal-world execution

- Stated differently: secure computation protocols emulate an incorruptible trusted party in a world with no trust

- Since the ideal-world adversary cannot do anything (essentially), the same is true of the real-world adversary
  - Privacy, correctness, independence of inputs (and more) all hold
The Ideal/Real Paradigm

\[ f(x', y) \]

\[ \approx \]

Real World

Protocol

arbitrary output

protocol output

Ideal World

Trusted Party

\[ f(x', y) \]

arbitrary output

f(x', y)
Using Secure Computation

- Define an ideal-model
  - Specify the input/output behavior of the function only
  - Imagine an incorruptible trusted party computing it for you

- Argue that the ideal model suffices for your application

- Replace the trusted party with a real protocol

- There is no need to understand the underlying mechanism to incorporate secure computation
Feasibility of Secure Computation

- A fundamental theorem: any function can be securely computed!
  - Privacy-preserving data mining
  - Private genetic testing
  - Private social networking
  - Private database search

- But, this is a theoretical feasibility result
  - Can it be realized in practice?