# Secure multiparty computation-Part 1

**Lucie Deptová**

November 16, 2014

# Motivation

- Survey of criminal justice
- Millionaire's problem
- Privite bidding in auctions

# What is secure multiparty computation?

MPC:

- Set of parties, each has some secrete input
- Function of all their inputs
- How to solve this? Thrusted party X MPC protocol

Formally introduced as 2PC in 1982 by Andrew Yao
The word 'secure':

- Computation without revealing intputs
- Providing security properties
- Some parties may be corrupted, malicious

# What do we expect?

We want these security properties:

- Privacy
- Correctness
- Independence of inputs
- Fairness
- Guaranteed output delivery

For example secure auction

# How do we define security?

This point of view is problematic:

- Auction→a list of concerns, Elections→a list of concerns...
- Every application→a new definition
- Are all concerns covered?

An alternative point of view:

- Defining generally, capturing all security demands:
  1. Well-defined adversary model
  2. Well-defined execution setting

# Model of adversary

- Who is adversary?
- How does he behave?
  - Semi-honest
  - Malicious
  - Covert
- What is his power?
  - Polynomial-time
  - Unbounded

# Types of models

Corruption strategy

- Static: easier one
- Adaptive : much harder

Execution setting

- Stand-alone: helpful techniques and tools
- Concurrent general composition: realistic, very important, goal for constructions

# Basic definitions and notation

- Security parameter n: if large enough, security is ensured for all inputs of arbitrarily length
- $X(a, n)$:=random variable, which represents an output of protocol execution with input $a$ and security parameter $n$
- $\{X(a, n)\}$:=infinite serie, we will usually index by $n$ implicitly
- We call function $\mu$ negligible if: $\forall p$ polynomial $\exists N : \forall n > N : \mu(n) < \frac{1}{p(n)}$

# Basic definitions and notation

- $X = \{X(a,n)\}$ and $Y = \{Y(a,n)\}$ are computationally indistinguishable (denote $X \approx Y$) if for all polynomial-time distinguisher $D$ there exists a negligible function $\mu$ such that for all $a$ and for all large enough $n$ is :
  $|Pr[D(X(a,n)) = 1] - Pr[D(Y(a,n)) = 1]| < \mu(n)$
- If $D$ has unbounded power, X and Y are statistically close
- Let's have $m$ parties, $x = (x_1, ..., x_m)$ is input vector, $f_1(x), ..., f_m(x)$ are random variables then we call $f = (f_1, ..., f_m)$ a (probabilistic) functionality, we denote
  $(x_1, ..., x_m) \to (f_1(x_1, ..., x_m), ..., f_m(x_1, ..., x_m))$.
  We don't have to compute the same function for all parties.

# Feasability results

- If we assume an honest majority any functionality can be securely computed even if an adversary has unbounded power.
- Without an honest majority we can't achieve fairness generally but any functionality can be computed securely without it.

# Zero knowledge proof

- Prover $P$, verifier $V$, language $L$
- $P$ wants to prove $x \in L$ to $V$ without revealing anything else about $x$ e.g. 26781 (=113x237) is not a prime
- The proof has to satisfy 3 properties:
  - Completeness: $V$ accepts if honest $P$ and $V$ interact and $x \in L$
  - Soundness: For any $P^*$ $V$ accepts if $x \notin L$ only with negligible probability
  - Zero-knowledge: For every $V^*$ there exists a simulator $S$ that given $x$ can produce a transcript which 'looks the same' as the interaction between $P$ and $V^*$.

# Security for semi-honest adversary

- Our protocol runs as it should $\rightarrow$ output, transcript of protocol execution
- What does an adversary have?
- Can he learn something more?
- We simulate the adversary's view of a protocol execution from given input and output

An exact definition:

- A protocol is secure for semi-honest adversary if for every semi-honest adversary $A$ there exists a simulator $S$ such that for every set of corrupted parties $I$ and for every input vector $x$ the two following sequences are close:
  - The output of $A$ and the outputs of honest parties
  - The output of $S$ given $x_i$ an $f_i(x)$ for all $i \in I$ and the outputs of honest parties

# Security for semi-honest adversary

- What does 'close' mean?
  - Computational security
  - Statistical security
  - Perfect security
- To simulate an adversary it's sufficient to simulate the transcript from communication with adversary
- The joint distribution-example
- Deterministic functionalities
- Didn't we break our security properties?
  - Privacy, corectness, independence of inputs ect.