CACE WP5 Meeting
Porto 22-23/07-2008

Security Policies in MPC

Janus Dam Nielsen
University of Aarhus
Overview

• A Grand Challenge
  • and the design-space it spans
• Examples of relevant security guaranties
• Examples of SMC applications
A Grand Challenge

Is P a correct realization of F?

Ideal functionality (F)

Program (P)

Yes

No
The Design-space

1. Security against all attacks in the UC sense
2. Formal verification of a program against a specific security model
3. Security against classes of attacks (insider/outsider attacks, intentional/unintentional, covert channels, etc.)
4. Security against individual attacks
SMCL

a place in space

- A domain-specific language for SMC
- Target audience is not crypto-people
- Provides some security guaranties
The Millionaire's Example

```smcl
declare client Millionaires:
    tunnel of sint netWorth;

    function void main(int[] args) {
        ask();
    }

    function void ask() {
        netWorth.put(readInt());
    }

    function void tell(bool b) {
        if (b) {
            display("You are the richest!");
        } else {
            display("Make more money!");
        }
    }

declare server Max:
    group of Millionaires mills;

    function void main(int[] args) {
        sint max = 0;
        sclient rich;

        foreach (client c in mills) {
            sint netWorth = c.netWorth.take();
            if (netWorth >= max) {
                max = netWorth;
                rich = c;
            }
        }

        foreach (client c in mills) {
            c.tell(open(c==rich|rich));
        }
    }
```

Security Model

- Adversary may:
  - Observe physical state of the server
  - Not observe private and secret values
Security Guaranties

- Security against attacks that are a function of the program trace
  - Timing, information flow, etc.

- Enforced by:
  - Carefully crafted semantics
  - Static analysis of well-typed SMCL programs

If an adversary corrupts more than the threshold of servers then all guaranties are off
Semantics

- Conditionals are a source of differences in the trace
- Execute both branches
- Termination
- Public side-effects

```
if (b) {
    x = y;
} else {
    x = z;
}

x = b*y + (1-b)*z
```
Hoistability

- Branches must agree on public side-effects
- Assignment to public variables
- Communication
- Function calls
- While loops and recursion with secret condition are not allowed
Semantic Information Leak

- Ideal computations are inefficient
- Prove that a pragmatic version reveals same information as the ideal version
- Assist the programmer

Ideal computation

open(e|x,y,z)
SMC Applications

- Auctions
- Negotiations
- Benchmarking
- Datamining
- Voting
- Survey
- Etc. (See WP4 d.1)
Sugarbeet auction

- Developed as a partnership with a private company
- Approx. 3 years of work
- Security model changed during development
- How security is achieved is not important for users - only that it is “secure”
Questions?