Security in Process Calculi

Service Oriented Architectures

Module 1 – Basic technologies

Ernesto Damiani

Università degli Studi di Milano

Overview

Pi calculus

- Core language for parallel programming
- Modeling security via name scoping

Applied pi calculus

- Modeling cryptographic primitives with functions and equational theories
- Equivalence-based notions of security
- A little bit of operational semantics
- Security as testing equivalence

Pi Calculus

[Milner et al.]

Fundamental language for concurrent systems

- High-level mathematical model of parallel processes
- The “core” of concurrent programming languages
- By comparison, lambda-calculus is the “core” of functional programming languages

Mobility is a basic primitive

- Basic computational step is the transfer of a communication link between two processes
- Interconnections between processes change as they communicate

Can be used as a general programming language

A Little Bit of History

1980: Calculus of Communicating Systems (CCS) [Milner]

1992: Pi Calculus [Milner, Parrow, Walker]

- Ability to pass channel names between processes

1998: Spi Calculus [Abadi, Gordon]

- Adds cryptographic primitives to pi calculus
- Security modeled as scoping
- Equivalence-based specification of security properties
- Connection with computational models of cryptography

2001: Applied Pi Calculus [Abadi, Fournet]

- Generic functions, including crypto primitives

Pi Calculus Syntax

Terms

- $M, N ::= x$ *variables*
  $\mid n$ *names*

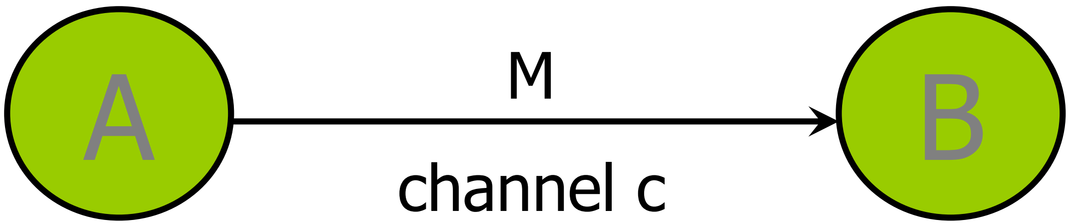
Let u range over names and variables

Processes

- $P,Q ::= nil$ *empty process*
  $\mid \bar{u}\langle N \rangle.P$ *send term N on channel u*
  $\mid u(x).P$ *receive term from channel P and assign to x*
  $\mid !P$ *replicate process P*
  $\mid P|Q$ *run processes P and Q in parallel*
  $\mid (\nu n)P$ *restrict name n to process P*

Modeling Secrecy with Scoping

A sends M to B over secure channel c



$$A(M) = \bar{c}\langle M \rangle$$

$$B = c(x).nil$$

$$P(M) = (\nu c)(A(M) | B)$$

This restriction ensures that channel c is "invisible" to any process except A and B (other processes don't know name c)

Secrecy as Equivalence

$$A(M) = \bar{c}\langle M\rangle$$
$$B = c(x).nil$$
$$P(M) = (\nu c)(A(M)|B)$$

Without (νc), attacker could run process c(x) and tell the difference between P(M) and P(M')

P(M) and P(M') are "equivalent" for any values of M and M'

- No attacker can distinguish P(M) and P(M')

Different notions of "equivalence"

- Testing equivalence or observational congruence
- Indistinguishability by any probabilistic polynomial-time Turing machine

Another Formulation of Secrecy

$$A(M) = \bar{c}\langle M \rangle$$
$$B = c(x).nil$$
$$P(M) = (\nu c)(A(M) \mid B)$$

No attacker can learn name n from P(n)

- Let Q be an arbitrary attacker process, and suppose it runs in parallel with P(n)
- **For any process Q in which n does not occur free, P(n) | Q will never output n**