Paragon

Practical programming with information flow control
Paragon

Practical, Flexible Programming with Information Flow Control

Niklas Broberg, PhD Thesis 2011
Chalmers University, Sweden
Java (eclipse)

```java
package se.chalmers.tmp;

public class Sample {
    public void foo() {
        Integer myInt;
        int i = myInt.intValue();
    }
}
```
package se.chalmers.tmp;

public class Sample {

    public void foo() {
        sendOverPublicChannel(missile_launch_codes);
    }

    Illegal information flow

    **Policy of argument:** Information can flow to no one
    **Expected policy:** Information can flow to anyone
    No quick fixes available
Information Flow Control

Enforcing policies on data inside a program
Information Flow Control

Not putting policies on files etc., but on:

- Variables
- Fields
- Method's arguments
- Method's return value
- Method's side effects
- ...


Information Flow Control

- Policy: “Missile Launch Codes are only to be read by the US Government”

send_To_Swedish_King ( Missile_Launch_Codes ) ;
Information Flow Control

• Policy: “The gender of the Queen's child is only to be announced on baptism day”

```java
if ( gender == "boy" )
   send_To_Newspaper ( "It's a boy!" ) ;
else
   send_To_Newspaper ( "It's a girl!" ) ;
```
Paragon

- Extends **Java** with information flow policies
- **Statically (type) checks** policy violations
- Fields etc. need to be **annotated** with policies by the programmer
- Policies are **defined** using **paralocks**
Paralocks

A simple and expressive policy specification language

(simpler than DLM)
Jif and DLM

- **Jif** (cornell) has the same goals as Paragon
- Jif was first, exists since **1997**
- **Jif is great! But...**
- Main obstruction: it's policy language **DLM**
Decentralized Label Model

- Not very intuitive, complicated
- Restricted to one model of information flow
- More complicated with introduction of a separate integrity lattice
Paralocks

- Goal: simple but expressive language
- Intuitive (Datalog) semantics
- Can be presented in 4 slides...
Paralocks Policies

- Policy: “Missile Launch Codes are only to be read by the US Government”

(Annotate the Missile Launch Codes with this policy)
Paralocks Policies

- Policy: “The newspaper can only know the gender on baptism day”

{ Newspaper : Today_Is_Baptism_Day }

Actors

Locks
Paralocks Policies

- Policy: “The newspaper can only know the gender on baptism day”

{ Newspaper : Today_Is_Baptism_Day }

- Locks can be opened and closed (open Today_Is_Baptism_Day;)

- Paragon keeps track of the lock state: the set of definitely opened locks

- Validity of an information flow depends on the current lock state
Paralocks Policies

- Policy: “Everyone can listen to this online music stream if they paid for it”

```paralock
{x : Has_Paid(x)}
```
{ 'x : Has_Paid('x) }

{ alice : }

Lock State
{ 'x : Has_Paid('x) }

Lock State

{ alice : }

20 / 25
{ 'x : Has_Paid('x) }

open Has_Paid(alice);

{ alice : }

Lock State
Has_Paid(alice)
{ 'x : Has_Paid('x) }

Lock State
Has_Paid(alice)

{ alice : }
Current Status

- First version of compiler (almost) done
- Can express DLM in Paragon (JifPoker)
- Extensions to Paralocks language
Future / Current Work

- **Case studies** using Paragon for
  - Android applications
  - Web services
  - Databases

- **Promoting** Paragon
  - Use it in a course?
  - Ask me!
Contact and questions

www.cse.chalmers.se/~d00nibro/paragon
vandeba@chalmers.se

Paralocks: role-based access control and beyond –
Niklas Broberg, David Sands, POPL 2010

A Datalog semantics for Paralocks –
Bart van Delft, David Sands, Niklas Broberg, STM 2012