Flow Caml in a Nutshell

Vincent Simonet

INRIA Rocquencourt — Cristal project

Vincent.Simonet@inria.fr
http://cristal.inria.fr/~simonet/
The confinement problem

Information systems run simultaneously several processes on behalf of multiple agents which read and write data in a shared space.

It is often mandatory to control information flow in these systems, in order to preserve

- integrity (only authorized agents can modify data)

- confidentiality (only authorized agents can read data)

[Lampson (1973)]
**Information flow control**

**Access control**
By an authentication-based mechanism, the initial release of data is controlled. No further verification is performed.

⇒ Requires trust in programs which manipulate data

**Information flow control**
Every operation performed by a program in the system is verified w.r.t. the security policy.

⇒ Requires a prior analysis of programs and systems
Flow Caml

From the security analysis viewpoint
Flow Caml is an extension of the Objective Caml language which

- automatically checks information flow within programs thanks to its annotated type system [POPL’02]

- translates them to regular Caml code that can be compiled with ocamlc or ocamlopt to produce secure programs

From the typing viewpoint
One of the first real size implementation of a programming language with full type inference, subtyping and polymorphism, in the style of HM(X).
Security annotations

Types are annotated with security levels interpreted in a lattice of principals.
Constraints

Type schemes involve a set of subtyping constraints restricting the range of variables ’a, ’b, ... They give a precise and oriented description of information flow.
An example of data structure: lists

The type \((\text{'a, 'b})\text{ list}\) has two parameters: \text{'a} is the type of the elements of the list and \text{'b} is the security level of the list itself.
Interacting with external principals

Security levels represent external principals which the program may interact with. For instance, !stdout stands for the standard output. Initially, the security policy allows no information flow between different principals.
Writing programs with Flow Caml

The Flow Caml library does not provide low-level functions for interacting with other external entities (e.g. network, display, file system). Thus, Flow Caml programs must be divided in two parts:

- A **high level model of external principals**, implemented in Objective Caml. A Flow Caml interface must specify its behavior w.r.t. the security policy.

- The **body of the program** is written and automatically verified with the Flow Caml system.
Give it a try!

The prototype implementation is available at

http://cristal.inria.fr/~simonet/soft/flowcaml/

A short guided tour of Flow Caml with selected examples of interactive sessions is included in the workshop proceedings and available at

http://cristal.inria.fr/~simonet/publis/