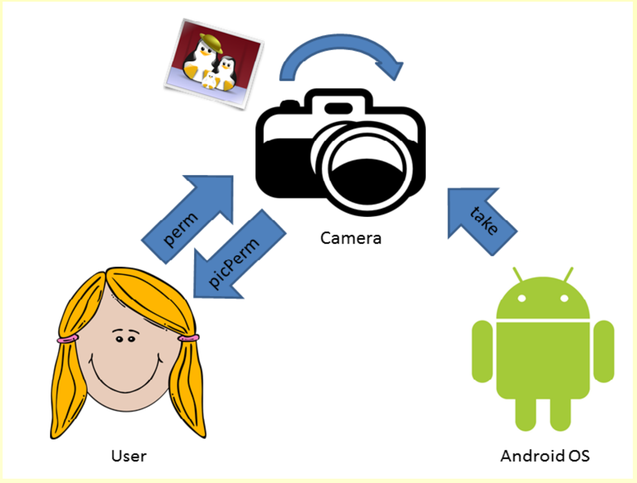
Contract Verification for Mobile Security

Hannah Gommerstadt, Frank Pfenning & Limin Jia
Carnegie Mellon University, Computer Science Department

Motivation



- Mobile applications may require very fine grained permissions for security reasons
- Consider a camera application that seeks permission from the user every time it tries to take a picture

Approach

- We use session types as contracts for processes
- The type system enforces a communication protocol between processes
- We prove that our type system prevents communication that violates the protocol

Contracts

- Contracts are specifications that are written in code and checked at runtime
- A contract can be expressed as the type of a function:

$$double\colon \{x\colon int\} \to int$$

$$sqrt\colon \{x\colon float \mid x \geq 0\} \to float$$

$$div\colon \{x\colon int\} \to \{y\colon int \mid y \neq 0\} \to float$$

Session Types

$c : A \otimes B$	send channel d : A along c, continue as B
$c : A \multimap B$	receive channel d : A along c, continue as B
$c : 1$	close channel c and terminate
$c : \oplus\{l_i : A_i\}$	send label l_i along c, continue as A_i
$c : \&\{l_i : A_i\}$	receive label l_i along c, continue as A_i

References

1. Higher-Order Processes, Functions, and Sessions: A Monadic Integration. Bernardo Toninho, Luís Caires, and Frank Pfenning. European Symposium on Programming (ESOP), pp. 350-369, March 2013.
2. F.Pfenning and D.Griffith. Polarized substructural session types. In A. Pitts, editor, Proceedings of the 18th International Conference on Foundations of Software Science and Computation Structures (FoSSaCS) 2015, London, England. Apr. 2015. Springer LNCS. Invited Talk.

Model

$$stype\ Cam = \&\{take : photoPerm \multimap picHandle \otimes Cam\}$$
$$stype\ User = \&\{picPerm : \oplus\{fail : User; succ : photoPerm \otimes User\}\}$$

- Contracts govern communication between camera and user processes
- Code snippet models interaction between camera and user processes

```
send cam take;
send user picPerm;
case user of
        fail =>
        succ => perm ← recv user;
                send cam perm;
                pic ← recv cam;
```

Future Directions

- When the type system detects that a process has violated its contract, that process is not always responsible
- Correctly blaming a process for a contract violation requires tracing the path of communication of the process

Carnegie Mellon
SCHOOL OF COMPUTER SCIENCE