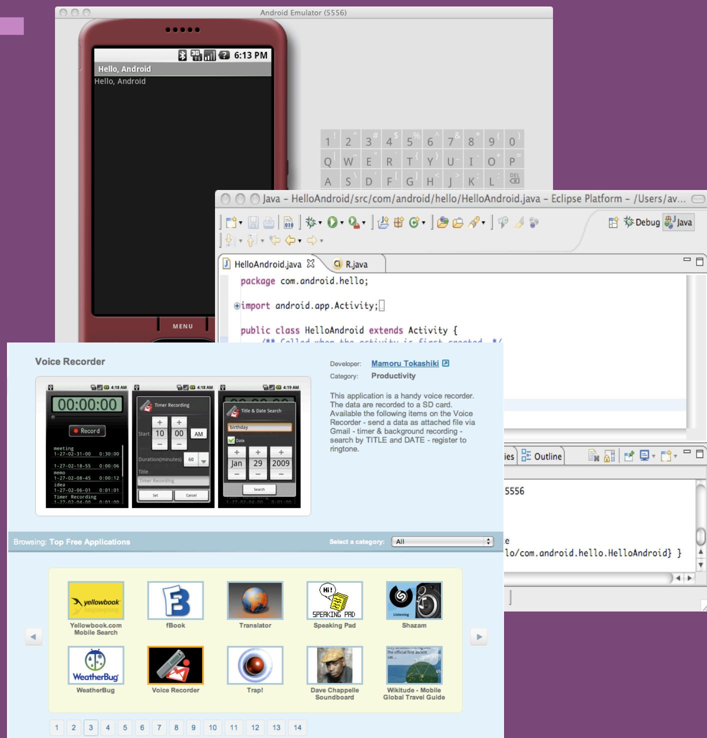
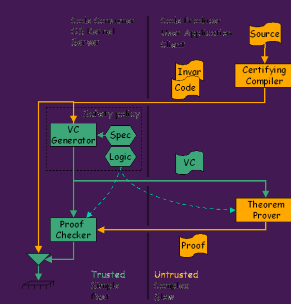




Language-Based Security on Android

(call for participation)

Avik Chaudhuri

+ What is Android?

- Open-source platform for mobile devices
- Designed to be a complete software stack
  - Operating system
  - Middleware
  - Core applications
- Accompanied with an SDK
  - Tools and APIs to develop new applications
  - Based on the Java programming language
  - Compiled to run on a VM designed to optimize memory and hardware resources in a mobile environment

Why should I care?

- Open Handset Alliance
  - major mobile operators (T-Mobile, Vodafone,…)
  - semiconductor companies (Intel, NVIDIA,…)
  - handset manufacturers (LG, Motorola, Samsung, Sony Ericsson,…)
  - software companies (**Google**,…)
- **Free** and **open-source** software
- **Core applications** and **new applications** on par
  - All applications can access the underlying mobile device
  - All applications have similar structure, and can share functionality

How can I join in?

- Maturing into a major platform in the mobile phone industry
  - Some Android-based phones available
    - More planned for release
  - Many Android applications available
    - Many more developed and installed every day
- Actually a source of concern!

What do we understand about security on Android?

- Costs of ignorance? (Same as in other settings)
- Benefits of knowledge? (Perhaps more than in other settings)

This reminds me of PCC!

- Alice downloads on her phone a new app developed by Bob
- How does she know whether it is safe to run the app?
  - Can she trust the app to safely access her data?
  - If she cannot, is there still a way to safely run the app?

These concerns are important for Alice (user).

These concerns are also important for Bob (developer).

- How does Bob convince Alice that his app is safe to run?

How about certified installation?

- Bob constructs a safety proof for his app
  - Sound static analysis of the Java code
- Alice verifies the proof before installing the application

Ingredients?

- **Operational semantics** for applications on Android
  - Formal specifications of the APIs provided by the SDK
- **Static checking** for safety of Android applications
  - Formalized as a security type system for application code
  - Soundness of type system provides proofs

What are Android applications?

- Package of **components**
  - Can be run as necessary (possibly even by other apps)

- Components follow structure

Activity controls some **window** of the app

Service runs in **background** (even if window is switched),
exposes interface for communication with other apps

Receiver reacts asynchronously to **messages** from other apps

Provider stores **content** relevant to the app (usually in database),
allows sharing of data with other apps

Show me an example.

Music player app

Activities

View all songs

Edit details of a song

Services

Play song in background

Receivers

Pause song on incoming call

Resume song when call ends

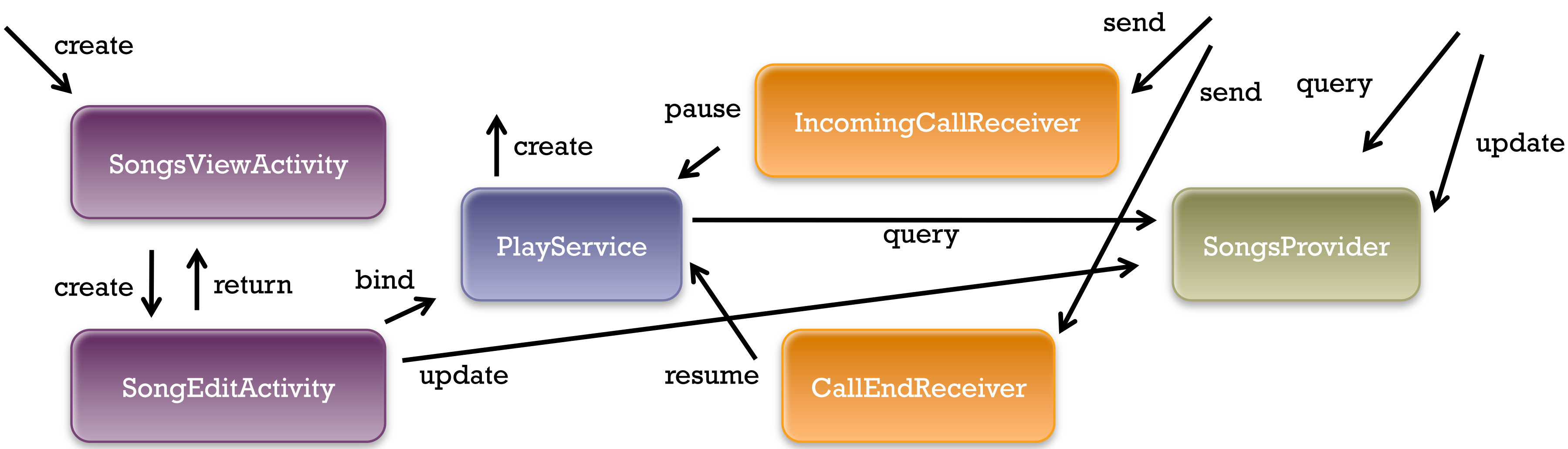
Providers

Share songs

How do I develop this in the SDK?

Inherit built-in classes, override their methods!

```
Activity { …, onCreate(x), onActivityResult(x), … }
Service { …, onBind(x), … }
Receiver { …, send(x), onReceive(x), … }
Provider { …, query(x), update(x), … }
```



What about security?

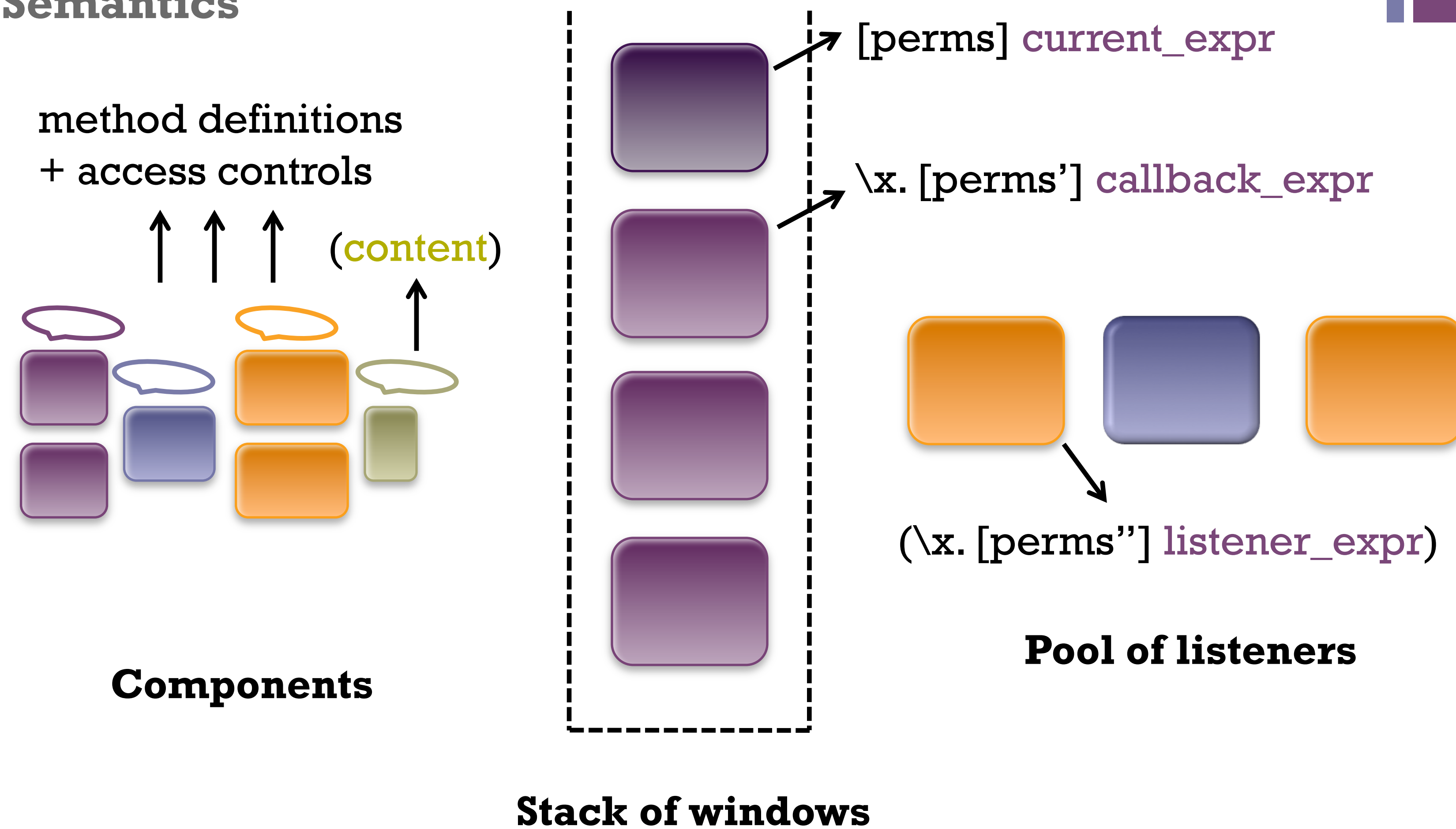
- Apps require permissions to access components of other apps
  - To call activities
  - To bind to services
  - To send messages to receivers, and to receive messages from apps
  - To query and to update content stored by providers

Permissions are set at install time, not at run time

- Required permissions are **declared** in a manifest
- Permissions are set **statically** by the package installer
- The app **blocks** if it requires other permissions at run time

I don’t know how my app is run!

Semantics



A formal abstract syntax

Program syntax

$v ::=$	value
$\quad n$	$\quad$ name
$\quad x$	$\quad$ variable
$\quad \mathsf{void}$	$\quad$ void
$i ::= (n, v)$	intent
$t ::=$	code
$\quad \mathsf{call}(i)$	$\quad$ call activity
$\quad \mathsf{return}(v)$	$\quad$ return from activity
$\quad \mathsf{bind}(i, \lambda x.t)$	$\quad$ bind to service
$\quad \mathsf{register}(\mathtt{SEND}, \lambda x.t)$	$\quad$ register new receiver
$\quad \mathsf{send}(\mathtt{RECEIVE}, i)$	$\quad$ send to receiver
$\quad !n$	$\quad$ read from provider
$\quad n := v$	$\quad$ write to provider
$\quad \mathsf{let}\ x = t\ \mathsf{in}\ t'$	$\quad$ evaluate
$\quad$ ↰ t	$\quad$ fork
$\quad t + t'$	$\quad$ choice
$\quad v$	$\quad$ result

model UI

enforce permissions

Application syntax

$d ::=$	definition
$\quad \mathsf{activity}(\mathtt{CALL}, \mathtt{PERMS}, \lambda x.t, \lambda x.t')$	$\quad$ activity
$\quad \mathsf{service}(\mathtt{BIND}, \mathtt{PERMS}, \lambda x.t)$	$\quad$ service
$\quad \mathsf{receiver}(\mathtt{SEND}, \mathtt{PERMS}, \lambda x.t)$	$\quad$ receiver
$\quad \mathsf{provider}(\mathtt{READ}, \mathtt{WRITE}, v)$	$\quad$ provider
$D ::= \varnothing \mid D, n \mapsto d$	hash of definitions

locate definitions

Encode away services

Encodings

$$\mathsf{service}(\mathtt{BIND}, \mathtt{PERMS}, \lambda x.t) \triangleq \mathsf{receiver}(\mathtt{BIND}, \mathtt{PERMS}, \lambda x.t)$$

$$\mathsf{bind}((n, v), \lambda x.t) \triangleq \mathsf{let}\ x = \mathsf{send}(\bot, (n, v))\ \mathsf{in}\ t$$

A formal operational semantics

Local reduction $D; e; E \rightarrow D'; e'; E'$

(Red let-distr) $D; [\mathsf{PERMS}]\ \mathsf{let}\ x = t\ \mathsf{in}\ t'; E \rightarrow D; \mathsf{let}\ x = [\mathsf{PERMS}]\ t\ \mathsf{in}\ [\mathsf{PERMS}]\ t'; E$

(Red let-eval)
$$\frac{D; e; E \rightarrow D'; e'; E'}{D; \mathsf{let}\ x = e\ \mathsf{in}\ [\mathsf{PERMS}]\ t; E \rightarrow D'; \mathsf{let}\ x = e'\ \mathsf{in}\ [\mathsf{PERMS}]\ t; E'}$$

(Red let-return) $D; \mathsf{let}\ x = [\mathsf{PERMS}']\ v\ \mathsf{in}\ [\mathsf{PERMS}]\ t; E \rightarrow D; [\mathsf{PERMS}]\ t\{v/x\}; E$

(Red fork)
$$\frac{E' = E, [\mathsf{PERMS}]\ t}{D; [\mathsf{PERMS}] \upharpoonleft t; E \rightarrow D; [\mathsf{PERMS}]\ \mathsf{void}; E'}$$

(Red read)
$$\frac{D(n) = \mathsf{provider}(\mathsf{READ}, \_, v) \qquad \mathsf{PERMS} \sqsupseteq \mathsf{READ}}{D; [\mathsf{PERMS}]\ !n; E \rightarrow D; [\mathsf{PERMS}]\ v; E}$$

(Red write)
$$\frac{D(n) = \mathsf{provider}(\mathsf{READ}, \mathsf{WRITE}, \_) \qquad \mathsf{PERMS} \sqsupseteq \mathsf{WRITE} \qquad D' = D[n \mapsto \mathsf{provider}(\mathsf{READ}, \mathsf{WRITE}, v)]}{D; [\mathsf{PERMS}]\ n := v; E \rightarrow D'; [\mathsf{PERMS}]\ \mathsf{void}; E}$$

(Red register)
$$\frac{n\ \mathsf{fresh} \qquad D' = D, n \mapsto \mathsf{receiver}(\mathsf{SEND}, \mathsf{PERMS}, \lambda x.t)}{D; [\mathsf{PERMS}]\ \mathsf{register}(\mathsf{SEND}, \lambda x.t); E \rightarrow D'; [\mathsf{PERMS}]\ n; E}$$

(Red send)
$$\frac{D(n) = \mathsf{receiver}(\mathsf{SEND}, \mathsf{PERMS}', \lambda x.t) \qquad \mathsf{PERMS} \sqsupseteq \mathsf{SEND} \qquad \mathsf{PERMS}' \sqsupseteq \mathsf{RECEIVE}}{D; [\mathsf{PERMS}]\ \mathsf{send}(\mathsf{RECEIVE}, (n, v)); E \rightarrow D; [\mathsf{PERMS}']\ t\{v/x\}; E}$$

(Red choice-l) $D; [\mathsf{PERMS}]\ t + t'; E \rightarrow D; [\mathsf{PERMS}]\ t; E$

(Red choice-r) $D; [\mathsf{PERMS}]\ t + t'; E \rightarrow D; [\mathsf{PERMS}]\ t'; E$

Global reduction $D; S; E \longrightarrow D'; S; E'$

(Red local)
$$\frac{D; e; E \rightarrow D'; e'; E'}{D; \langle e, \lambda x.e'' \rangle :: S; E \longrightarrow D'; \langle e', \lambda x.e'' \rangle :: S; E'}$$

(Red call)
$$\frac{D(n) = \mathsf{activity}(\mathsf{CALL}, \mathsf{PERMS}', \lambda x.t, \lambda x.t') \qquad \mathsf{PERMS} \sqsupseteq \mathsf{CALL} \qquad S' = \langle [\mathsf{PERMS}]\ \mathsf{void}, \lambda x.e \rangle :: S}{D; \langle [\mathsf{PERMS}]\ \mathsf{call}((n, v)), \lambda x.e \rangle :: S; E \longrightarrow D; \langle [\mathsf{PERMS}']\ t\{v/x\}, \lambda x.[\mathsf{PERMS}']\ t' \rangle :: S'; E}$$

(Red return)
$$\frac{S = \langle [\mathsf{PERMS}']\ \mathsf{void}, \lambda x.e' \rangle :: S'}{D; \langle [\mathsf{PERMS}]\ \mathsf{return}(v), \lambda x.e \rangle :: S; E \longrightarrow D; \langle e'\{v/x\}, \lambda x.e' \rangle :: S'; E}$$

(Red thread)
$$\frac{D; e; E \rightarrow D'; e'; E'}{D; S; E, e \longrightarrow D'; S; E', e'}$$

data flows +
control flows +
permission checks

A security type system

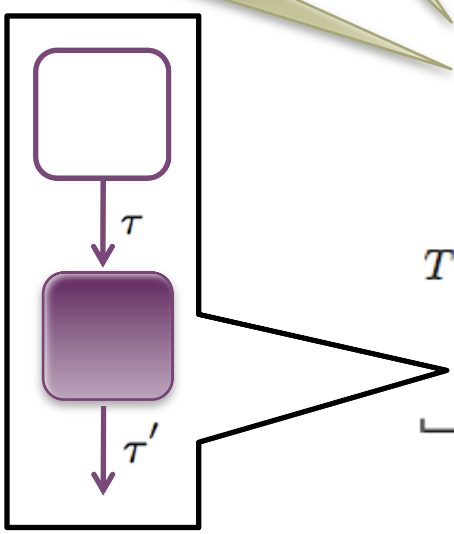
call activities from the **stack**

send to receivers from the **stack** or the **pool**

data that READ can read and WRITE can write

Types

$\tau ::=$	data type
$\mathsf{Any}(\mathtt{READ}, \mathtt{WRITE})$	any
$\mathsf{Activity}(\mathtt{CALL}, \tau \rightarrow \tau' \Rightarrow \tau'')$	activity
$\mathsf{Receiver}(\mathtt{SEND}, \tau \rightarrow T)$	receiver
$\mathsf{Provider}(\mathtt{READ}, \mathtt{WRITE})$	provider
Stuck	stuck
$T ::=$	type
τ	data type
$\tau \Rightarrow \tau'$	stack type



Stack types

Blocking due to access control

What guarantees do I get?

- **Sub-typing** captures **data flow**
- **Type preservation** implies **data-flow security**

Well-typed programs preserve their types [PLAS 09]

- Focus on data flow rather than information flow
- Guarantees should extend under refinement

Future work

Type-check concrete application code

- Build on existing static analysis tools for Java (e.g., WALA)

How does this help?

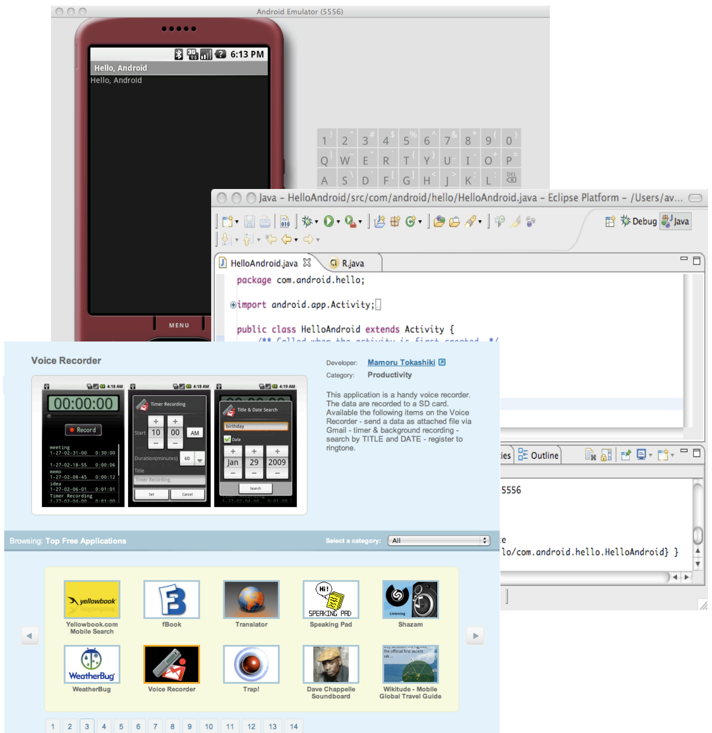
- Alice can safely run any well-typed app on her phone
  - Guaranteed to preserve the secrecy and integrity of her data
- Any app type-checks if it has no permission
  - Alice can safely run such an app on her phone (w/o permissions)
- Bob can convince Alice that his app is safe if it type-checks

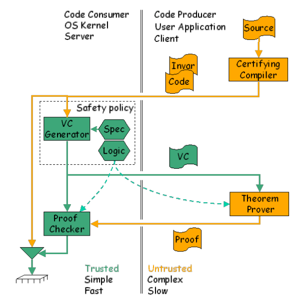
Certified installer for Android applications

Install apps that type-check, or are certified to type-check

Opportunity to bring language-based security to the mainstream!

Questions?


Android Emulator (5556)
Hello, Android
Hello, Android
Java - HelloAndroid/src/com/android/hello/HelloAndroid.java - Eclipse Platform - /Users/av...
HelloAndroid.java
R.java
package com.android.hello;
import android.app.Activity;
public class HelloAndroid extends Activity {
Voice Recorder
Developer: Mamoru Tokashiki
Category: Productivity
This application is a handy voice recorder. The data are recorded to a SD card. Available the following items on the Voice Recorder - send a data as attached file via Gmail - timer & background recording - search by TITLE and DATE - register to ringtone.
Browsing: Top Free Applications
Yellowbook.com Mobile Search
fBook
Translator
Speaking Pad
Shazam
WeatherBug
Voice Recorder
Trap!
Dave Chappelle Soundboard
Wikitude - Mobile Global Travel Guide


Code Consumer
OS Kernel
Server
Code Producer
User Application
Client
Source
Invar
Code
Certifying Compiler
Safety policy
VC Generator
Spec
Logic
VC
Theorem Prover
Proof Checker
Proof
Trusted
Simple
Fast
Untrusted
Complex
Slow