TWC: Medium: Language-Hardware Co-Design for Practical and Verifiable Information Flow Control
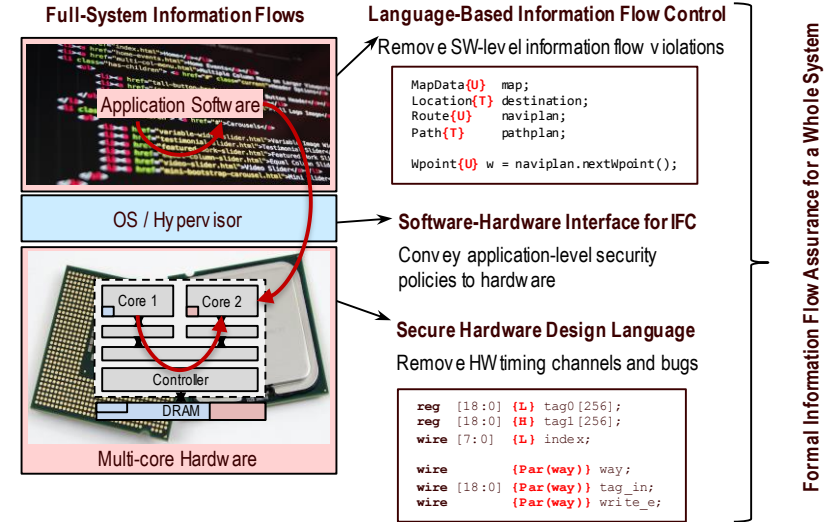
Principal Investigators: **G. Edward Suh and Andrew C. Myers**, Cornell University

Challenge

- **Problem**: Emerging applications and platforms require strong security assurance that is not possible today
  - No information leak among VMs in cloud computing
  - Integrity guarantee for safety-critical autonomous systems

- **Objective:** Co-design software and hardware with strong, comprehensive, verifiable information flow assurance
  - All software-visible information flows in a HW+SW system

Solution and Technical Highlights

- Integrated approach to offer **whole-system security** where all software-visible information flows are **verified statically**.

- Control information flow 1) within software and 2) hardware, and 3) between software and hardware
  - **Software**: Use language-level information flow control (IFC) to verify explicit and implicit flows, mitigate software timing channels, and identify needs for hardware-level control.
  - **Hardware**: Design a multi-core processor with strong information flow assurance, including timing channels
  - **Interface**: Develop a SW-HW interface with label virtualization to convey rich software-level security policies to hardware

- **Formal security assurance** with static information flow analysis

- Major technical developments in HW design and verification
  - **Efficient HW timing channel protection**: uni-directional protection for caches (DAC'16) and memory (HPCA'16), quantifiable information flow control for memory (HPCA'17)
  - **Security verification** of a simple ARM TrustZone prototype using static information flow analysis (ASPLOS'17)

Full-System Information Flows

Multi-core Hardware

Language-Based Information Flow Control
Remove SW-level information flow violations

```
MapData{U}   map;
Location{T}  destination;
Route{U}     naviplan;
Path{T}      pathplan;

Wpoint{U} w = naviplan.nextWpoint();
```

Software-Hardware Interface for IFC
Convey application-level security policies to hardware

Secure Hardware Design Language
Remove HW timing channels and bugs

```
reg  [18:0] {L} tag0[256];
reg  [18:0] {H} tag1[256];
wire [7:0]  {L} index;

wire        {Par(way)} way;
wire [18:0] {Par(way)} tag_in;
wire        {Par(way)} write_e;
```

Formal Information Flow Assurance for a Whole System

Scientific Impact

- Create a **new capability** to design computing systems with end-to-end security assurance across a whole system, including the software–hardware boundary.

- **New directions** in interdisciplinary research to co-design software, hardware, and formal verification methods for holistic system security

Broader Impact

- **Impact on society:** Significantly improve security and assurance of critical computing platforms

- **Education and outreach:** new course modules for programming languages and architecture, student training, high-school and K-12 outreach, and industry collaboration.