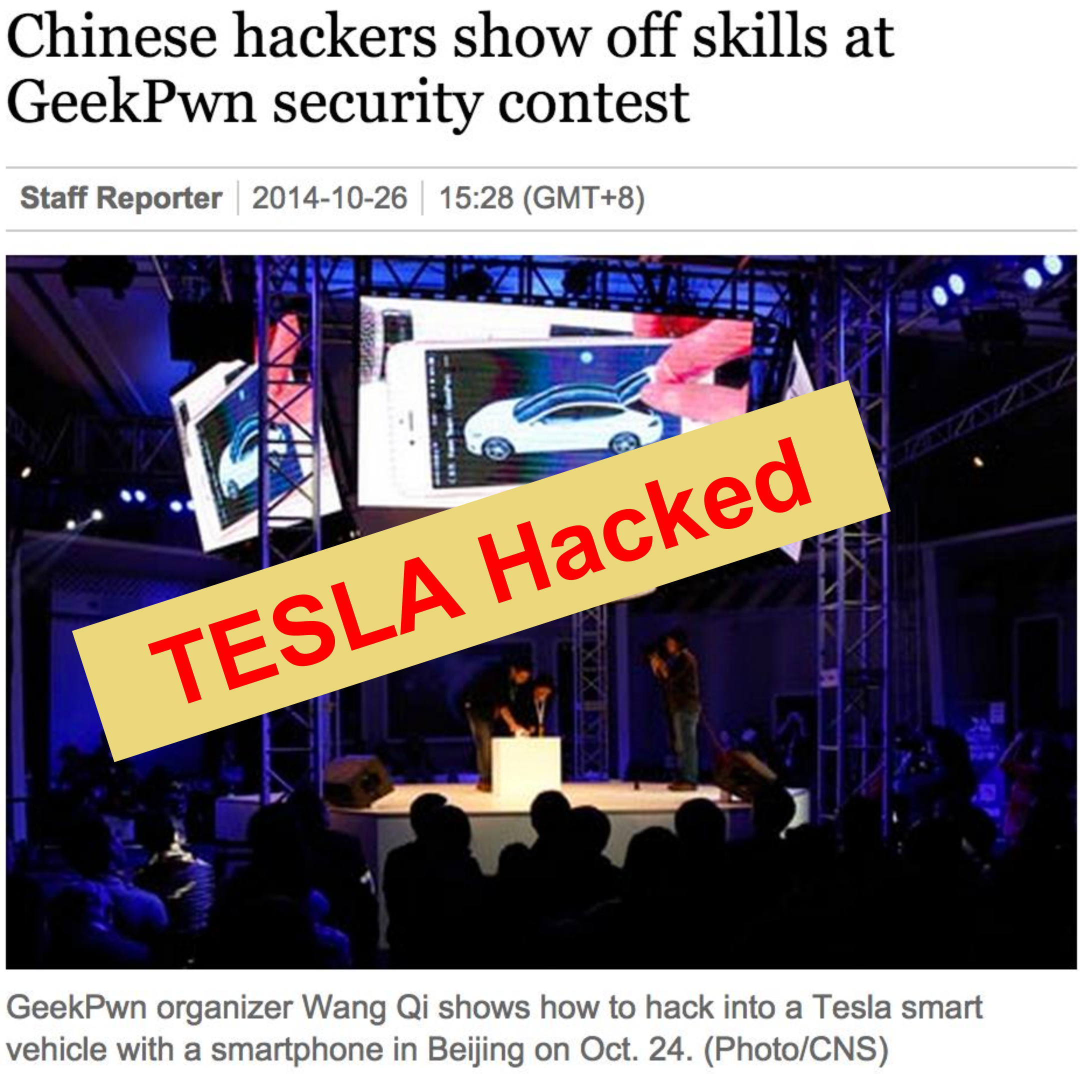
Attack Surface Analysis and Program Hardening of CPS Systems

Chao Zhang, Mathias Payer, Dawn Song

UC Berkeley

Vulnerable CPS Systems

TESLA

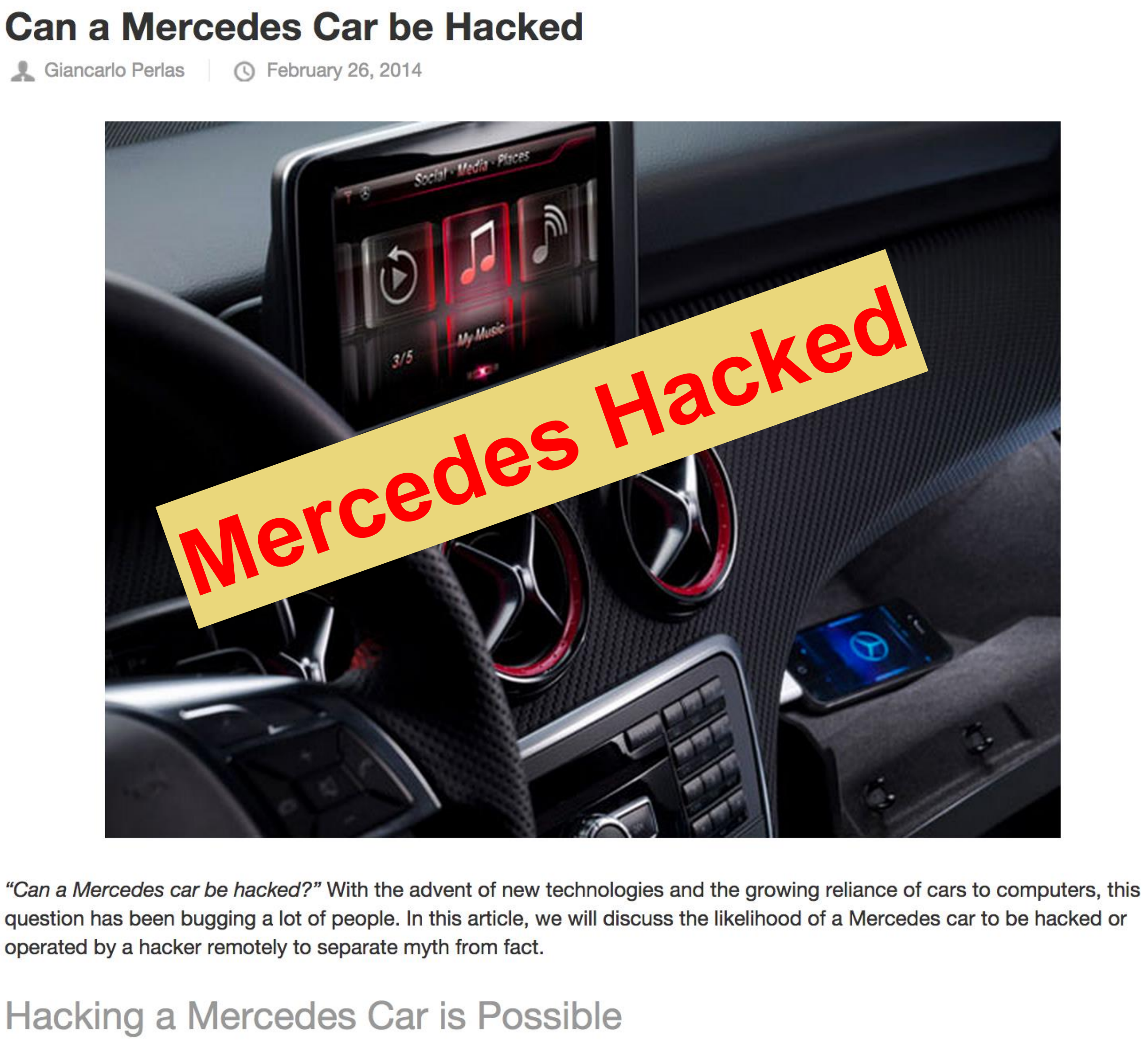


Chinese hackers show off skills at GeekPwn security contest

Staff Reporter | 2014-10-26 | 15:28 (GMT+8)

GeekPwn organizer Wang Qi shows how to hack into a Tesla smart vehicle with a smartphone in Beijing on Oct. 24. (Photo/CNS)

Mercedes



Can a Mercedes Car be Hacked

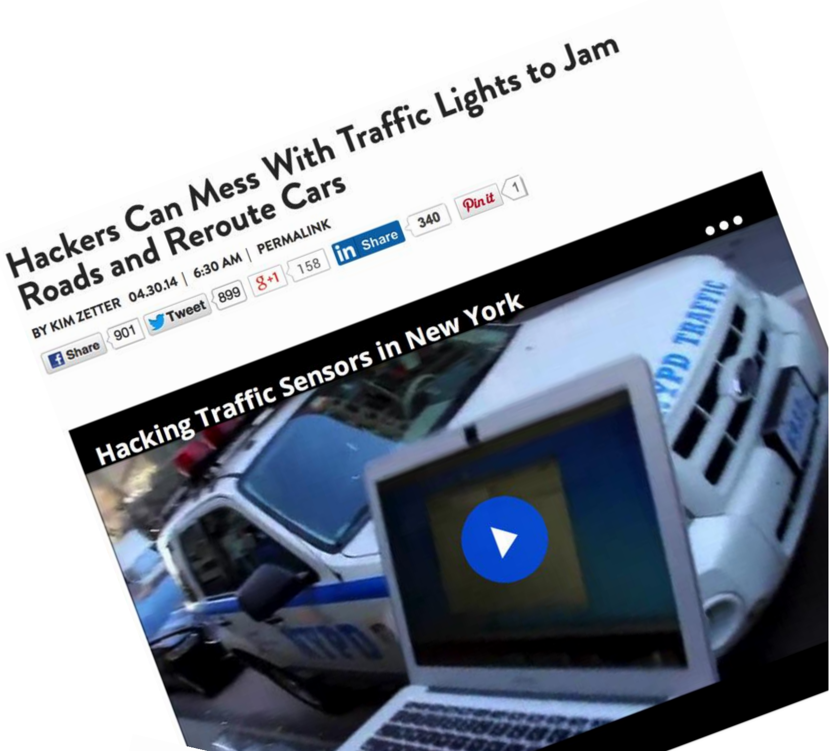
Giancarlo Perlas | February 26, 2014

"Can a Mercedes car be hacked?" With the advent of new technologies and the growing reliance of cars to computers, this question has been bugging a lot of people. In this article, we will discuss the likelihood of a Mercedes car to be hacked or operated by a hacker remotely to separate myth from fact.

Hacking a Mercedes Car is Possible

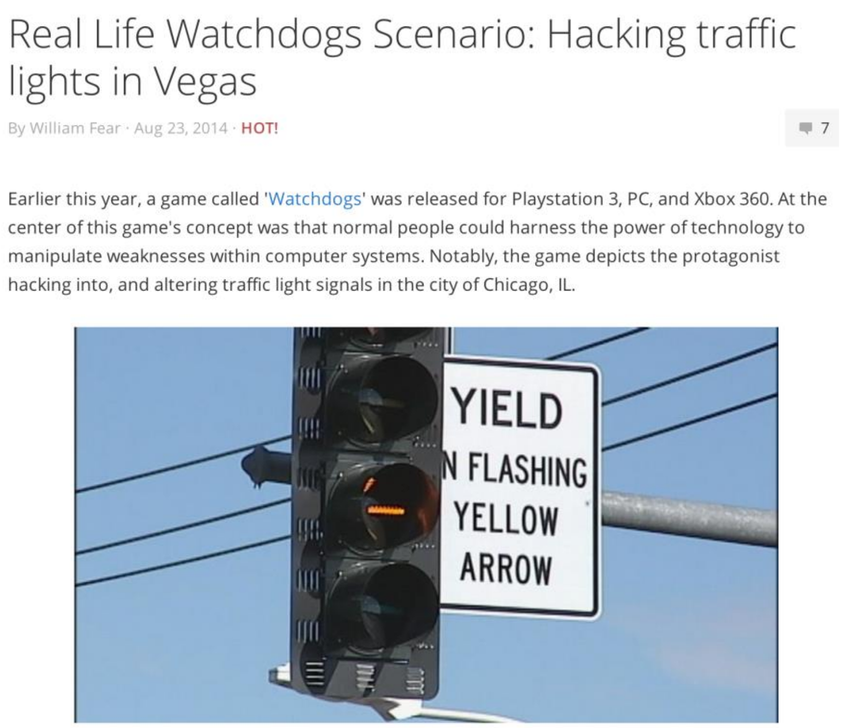
More Attacks

(traffic lights, navigation routes, signs, …)



Hackers Can Mess With Traffic Lights to Jam Roads and Reroute Cars

BY KIM ZETTER 04.30.14 | 6:30 AM | PERMALINK

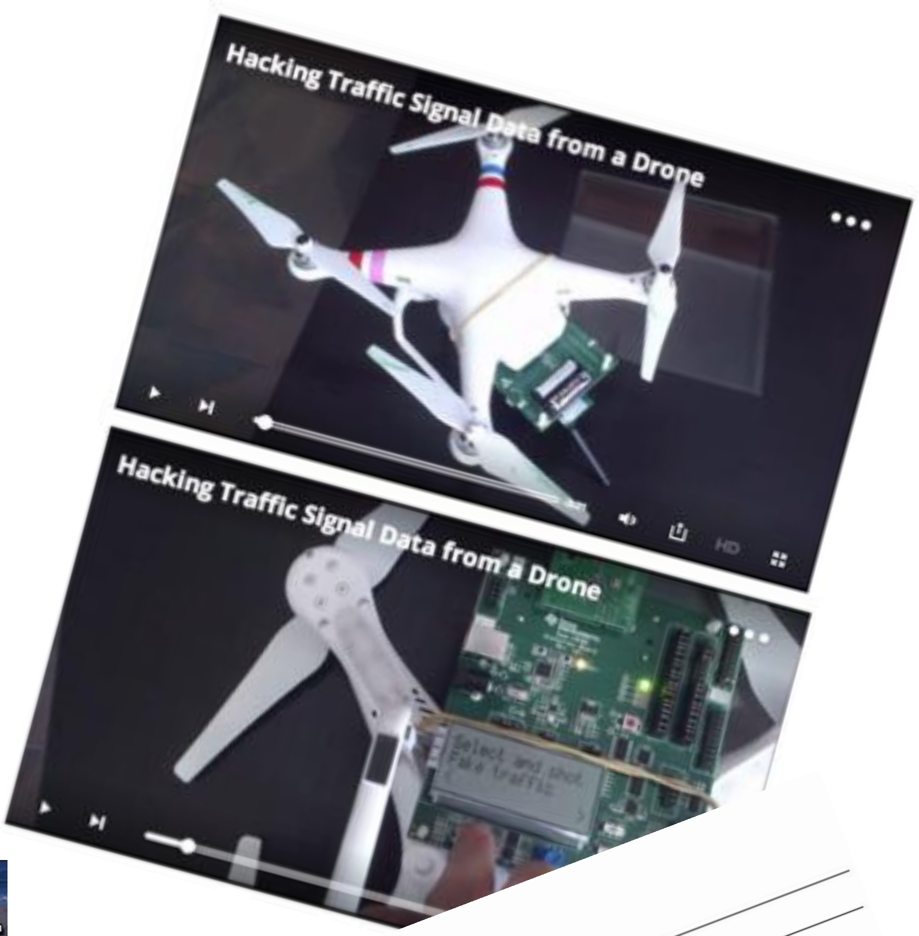


Real Life Watchdogs Scenario: Hacking traffic lights in Vegas

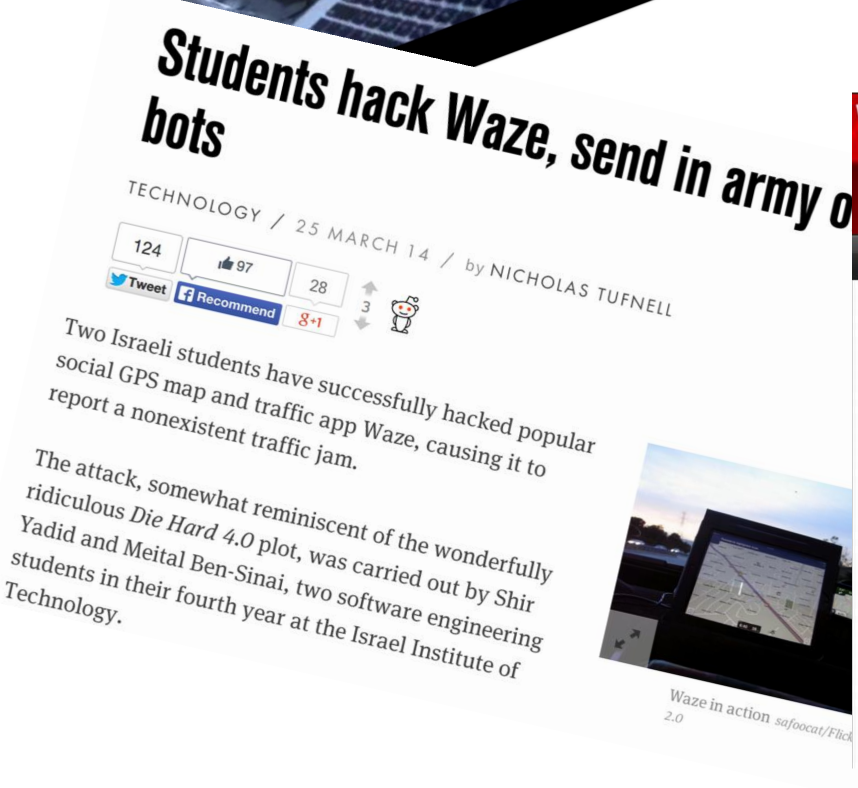
By William Fear · Aug 23, 2014 · HOT!

Earlier this year, a game called 'Watchdogs' was released for Playstation 3, PC, and Xbox 360. At the center of this game's concept was that normal people could harness the power of technology to manipulate weaknesses within computer systems. Notably, the game depicts the protagonist hacking into, and altering traffic light signals in the city of Chicago, IL.

7
YIELD
N FLASHING
YELLOW
ARROW



Hacking Traffic Signal Data from a Drone
Hacking Traffic Signal Data from a Drone
HD



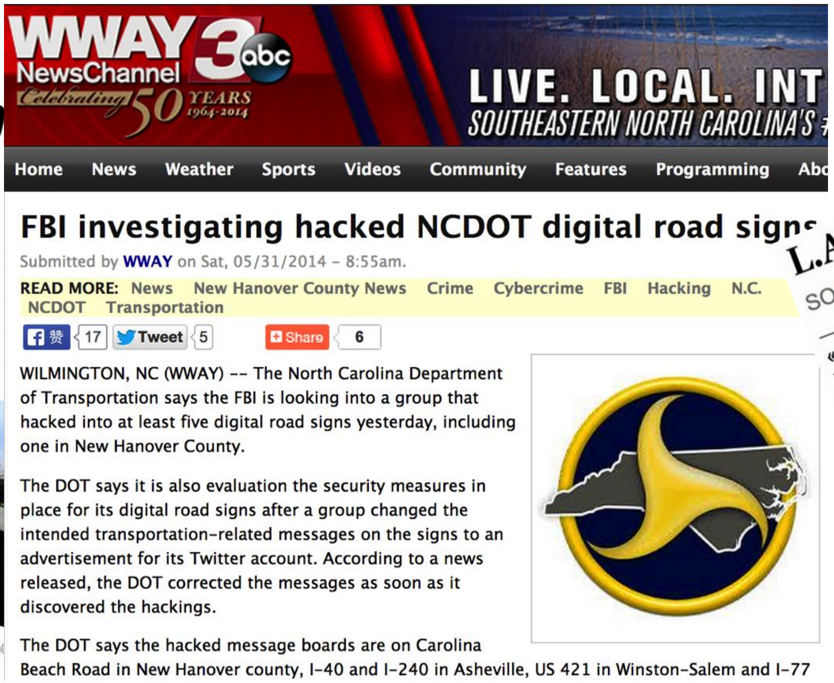
Students hack Waze, send in army o bots

TECHNOLOGY / 25 MARCH 14 / by NICHOLAS TUFNELL

Two Israeli students have successfully hacked popular social GPS map and traffic app Waze, causing it to report a nonexistent traffic jam.

The attack, somewhat reminiscent of the wonderfully ridiculous *Die Hard 4.0* plot, was carried out by Shir Yadid and Meital Ben-Sinai, two software engineering students in their fourth year at the Israel Institute of Technology.

Waze in action *safoocat/Flick* 2.0



FBI investigating hacked NCDOT digital road signs

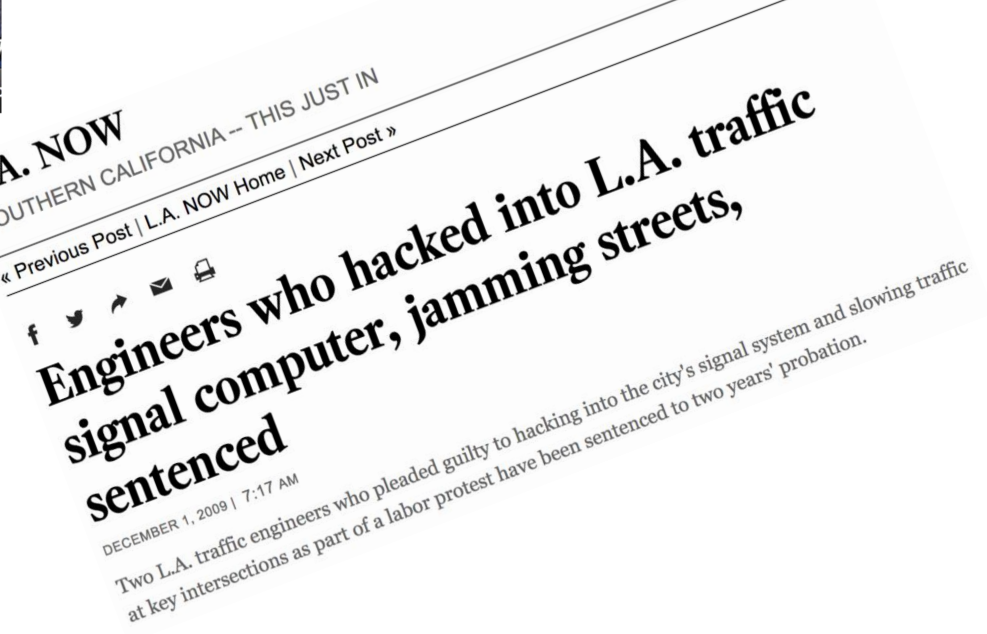
Submitted by WWAY on Sat, 05/31/2014 - 8:55am.

READ MORE: News New Hanover County News Crime Cybercrime FBI Hacking N.C. NCDOT Transportation

WILMINGTON, NC (WWAY) -- The North Carolina Department of Transportation says the FBI is looking into a group that hacked into at least five digital road signs yesterday, including one in New Hanover County.

The DOT says it is also evaluation the security measures in place for its digital road signs after a group changed the intended transportation-related messages on the signs to an advertisement for its Twitter account. According to a news released, the DOT corrected the messages as soon as it discovered the hackings.

The DOT says the hacked message boards are on Carolina Beach Road in New Hanover county, I-40 and I-240 in Asheville, US 421 in Winston-Salem and I-77



L.A. NOW

SOUTHERN CALIFORNIA -- THIS JUST IN

« Previous Post | L.A. NOW Home | Next Post »

Engineers who hacked into L.A. traffic signal computer, jamming streets, sentenced

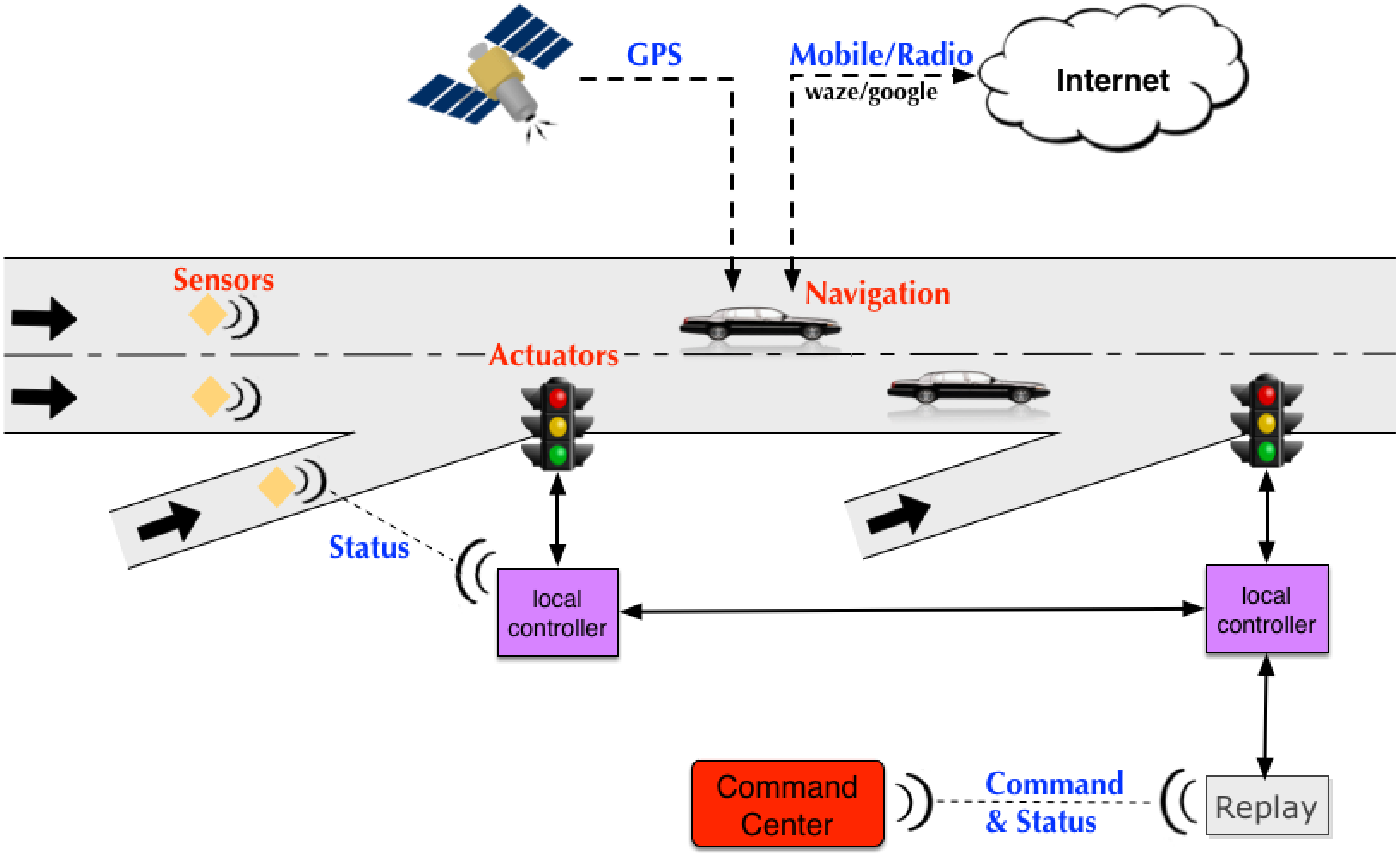
DECEMBER 1, 2009 | 7:17 AM

Two L.A. traffic engineers who pleaded guilty to hacking into the city's signal system and slowing traffic at key intersections as part of a labor protest have been sentenced to two years' probation.

Overview

- Motivation
- ➡ Attack Surface Analysis of the Transportation CPS
- Program Hardening of CPS
  - Without source code: CCFIR
  - With source code: CPI

Case Study: Intelligent Transportation System



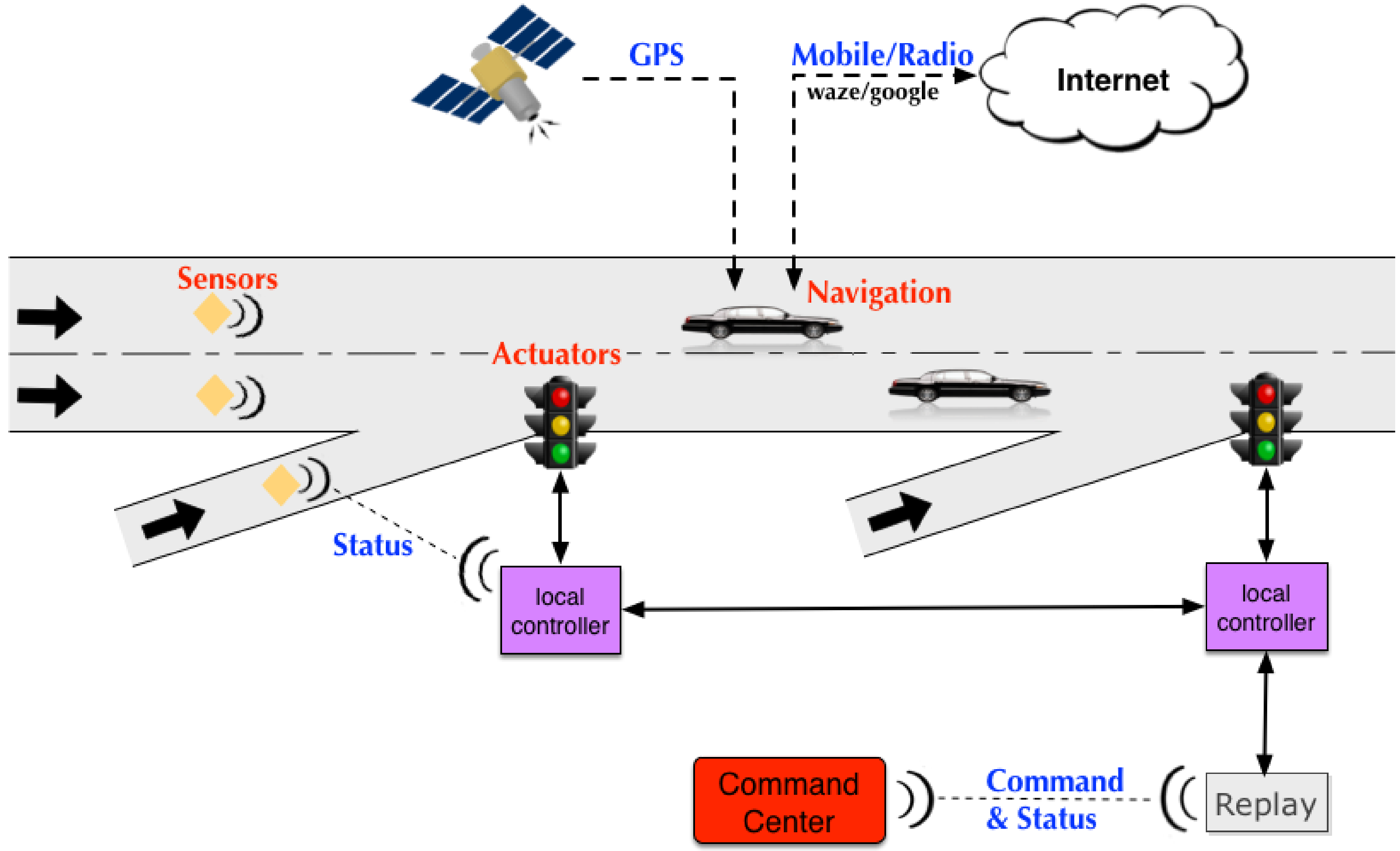
Threat Model

- Access Level
  - **Physical access attacks**
    - low cost, low control
    - easy to launch
  - **Locality access attacks**
    - medium cost, medium control
    - previous case study
  - **Remote access attacks**
    - medium cost, high control

- Vulnerable Components
  - Sensors (loop detectors)
  - Actuators (ramp metering)
  - Local controllers (2070 boxes)
  - command center
    - operators
    - IT infrastructure
  - navigation device
  - vehicle

Attack Surface Analysis: Physical Access

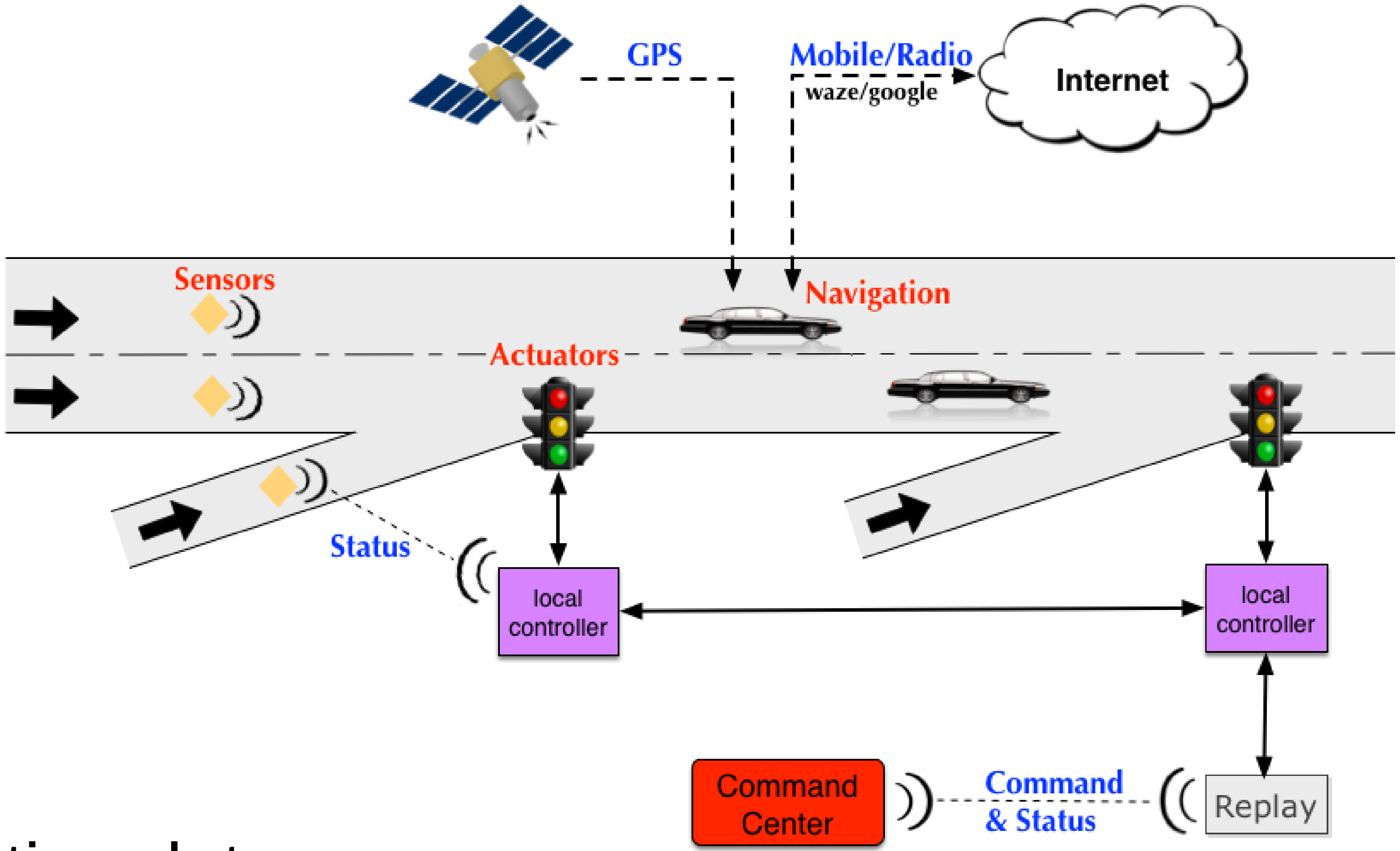
- Vulnerable Components
  - Sensors (loop detectors)
  - Actuators (ramp metering)
  - Local controllers (2070 boxes)
  - command center
  - navigation device
  - vehicle



- Possible attacks
  - copper theft (wires)
  - replace a single sensor/actuator/control box
  - replace a set of sensors/actuators/control boxes
  - implant malicious device into vehicles
    - http://www.benzinsider.com/2014/02/can-a-mercedes-car-be-hacked/
  - malicious operators
    - http://latimesblogs.latimes.com/lanow/2009/12/engineers-who-hacked-in-la-traffic-signal-computers-jamming-traffic-sentenced.html

Attack Surface Analysis: Locality Access

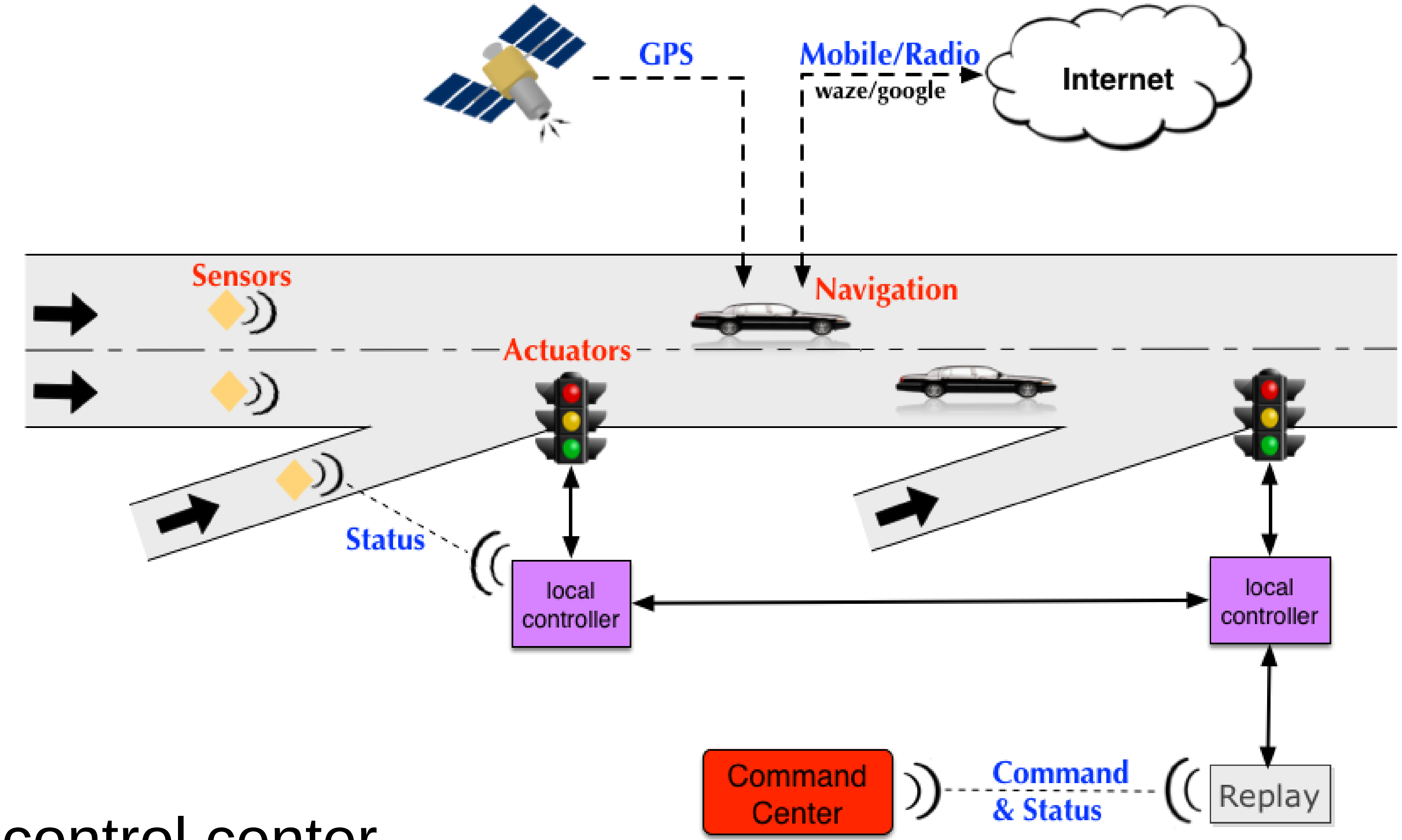
- Vulnerable Components
  - Sensors (loop detectors)
  - Actuators (ramp metering)
  - Local controllers (2070 boxes)
  - command center
  - navigation device
  - vehicle



- Possible attacks:
  - monitor communication data
  - spoof communication data
    - http://www.wired.com/2014/04/traffic-lights-hacking/
  - attack software running on sensors/actuators/controllers

Attack Surface Analysis: Remote Access

- Vulnerable Components
  - Sensors (loop detectors)
  - Actuators (ramp metering)
  - Local controllers (2070 boxes)
  - command center
  - navigation device
  - vehicle



- Possible attacks
  - attack software in the control center
    - http://www.wwaytv3.com/2014/05/31/fbi-investigating-hacked-ncdot-digital-road-signs
  - attack navigation software
    - http://www.wired.co.uk/news/archive/2014-03/25/waze-hacked-fake-traffic-jam
  - attack intelligent vehicles' software
    - http://www.wantchinatimes.com/news-subclass-cnt.aspx?id=20141026000071&cid=1103

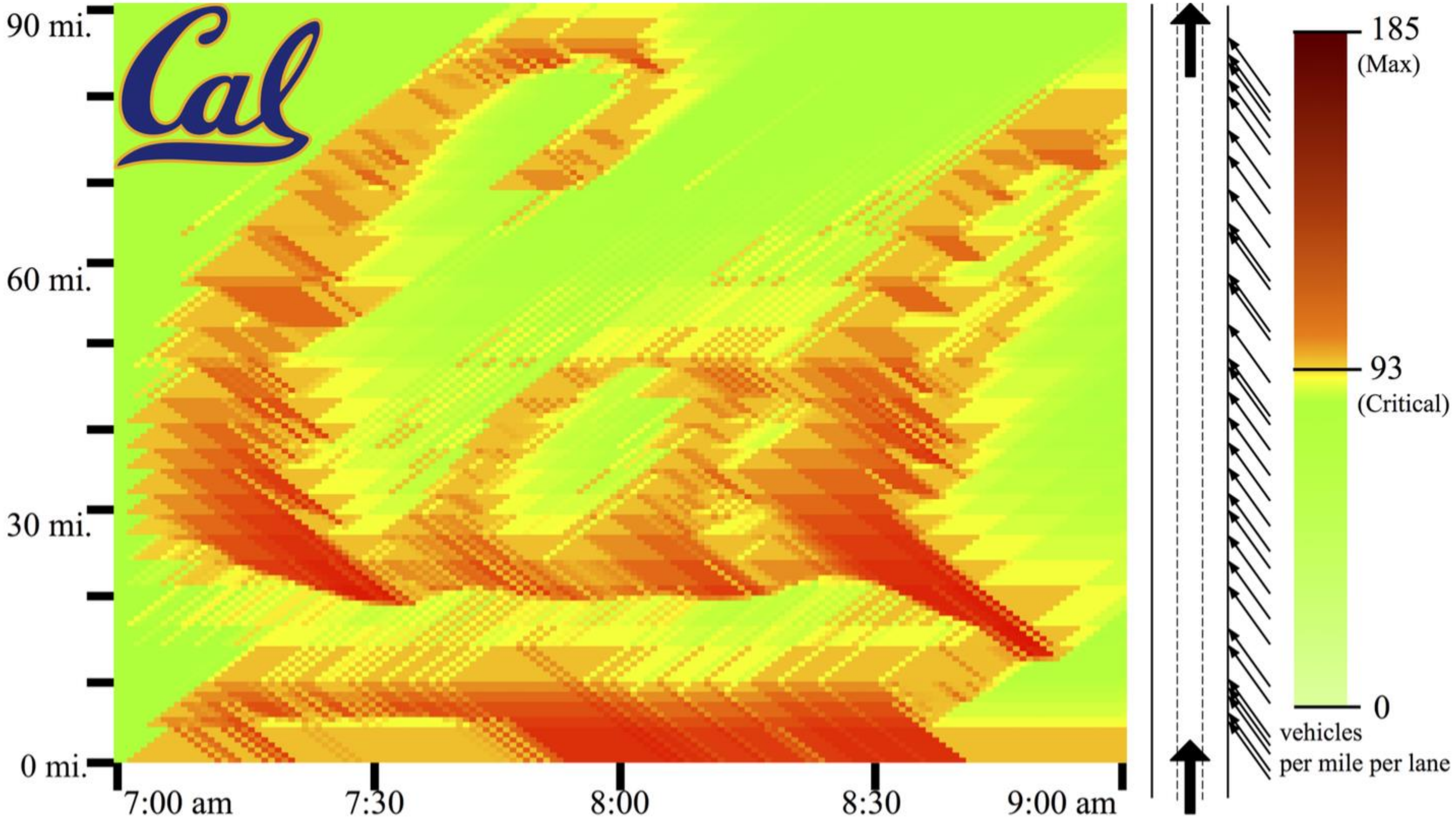
Proof-of-Concept Attacks

- Congestion-on-demand attack
  - ➢ create congestion patterns of a specific nature

- Catch-me-if-you-can attack
  - ➢ create a VIP-lane to get through

- ***work in cooperation with Prof. Alexander Bayen***

Congestion-on-demand



Object of the attack: a Cal logo (space-time diagram)

Security Challenge

- Software inevitably have vulnerabilities.
- Limited resources in CPS.

How to protect them from being exploited?

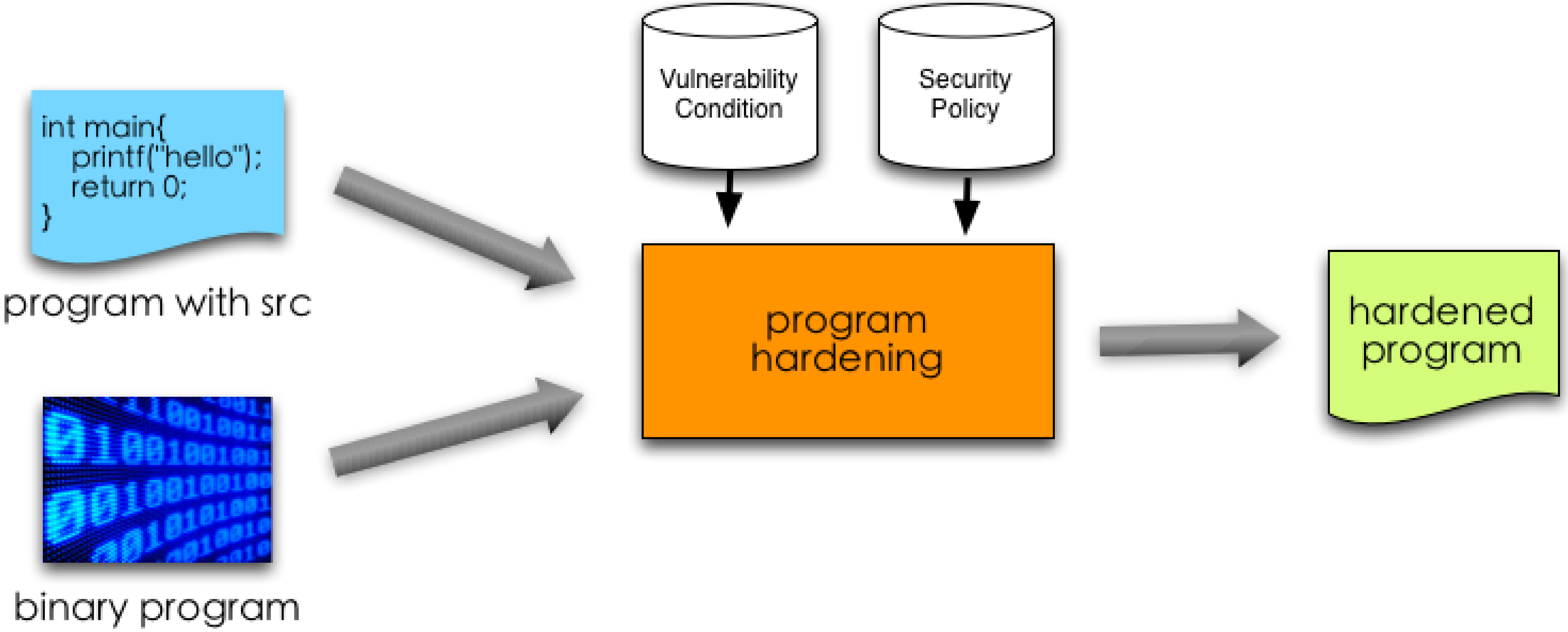
Overview

- Motivation
- Attack Surface Analysis of the Transportation CPS

➡ Program Hardening of CPS

  - ➢ Without source code: CCFIR
  - ➢ With source code: CPI

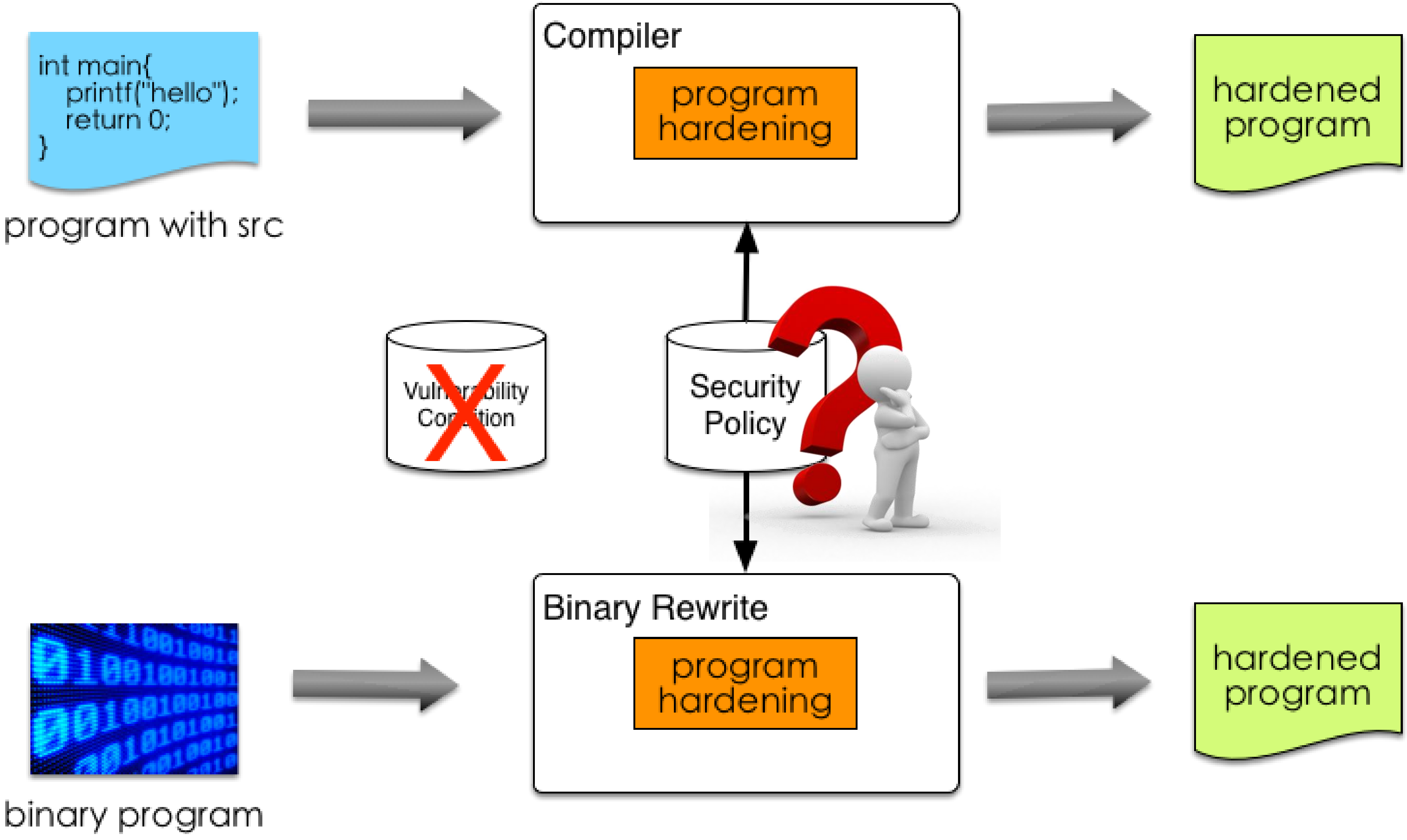
Program Hardening



```
int main{
   printf("hello");
   return 0;
}
```

- Fix vulnerabilities
- Deploy security checks

Our solutions



```
int main{
   printf("hello");
   return 0;
}
```

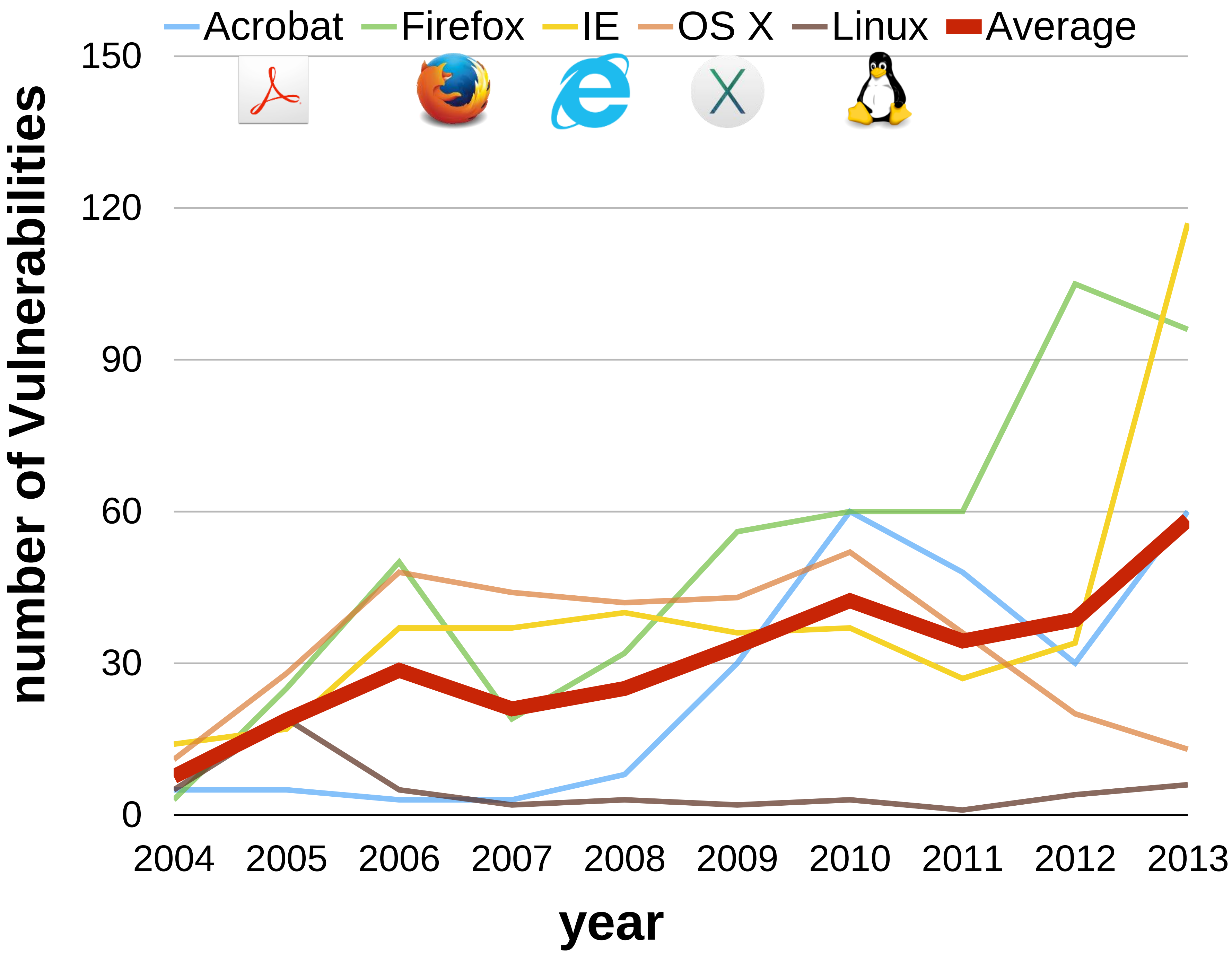
To select a security policy and enforce it,

Know your enemy first.

Sun Tzu

Top Vulnerabilities in CVE

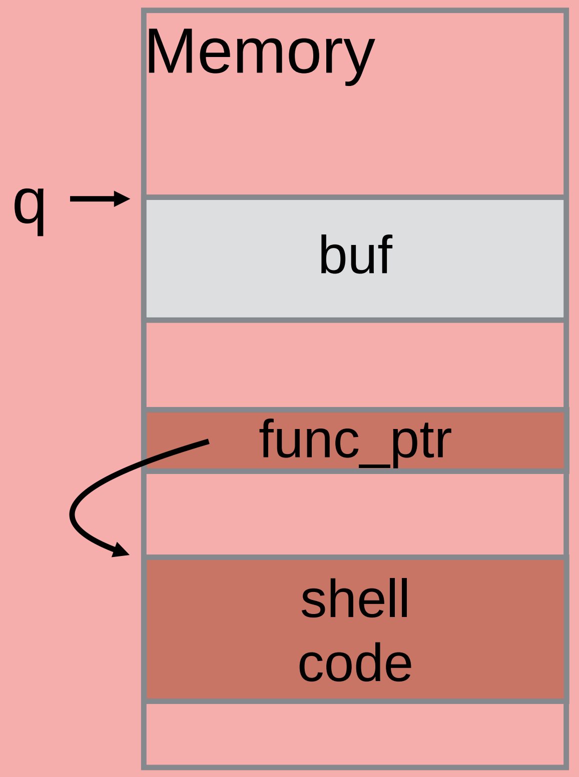
(Control-Flow Hijack)



Control-Flow Hijack Attack

```
int buf[100];
int *q = buf + input;
*q = input2;
…
(*func_ptr)();
```

☠️ ***execute arbitrary code!***



It started 50 years ago…

Security Policy

Control-flow hijack

```
int *q = buf + input;
*q = input2;
…
(*func_ptr)();
```

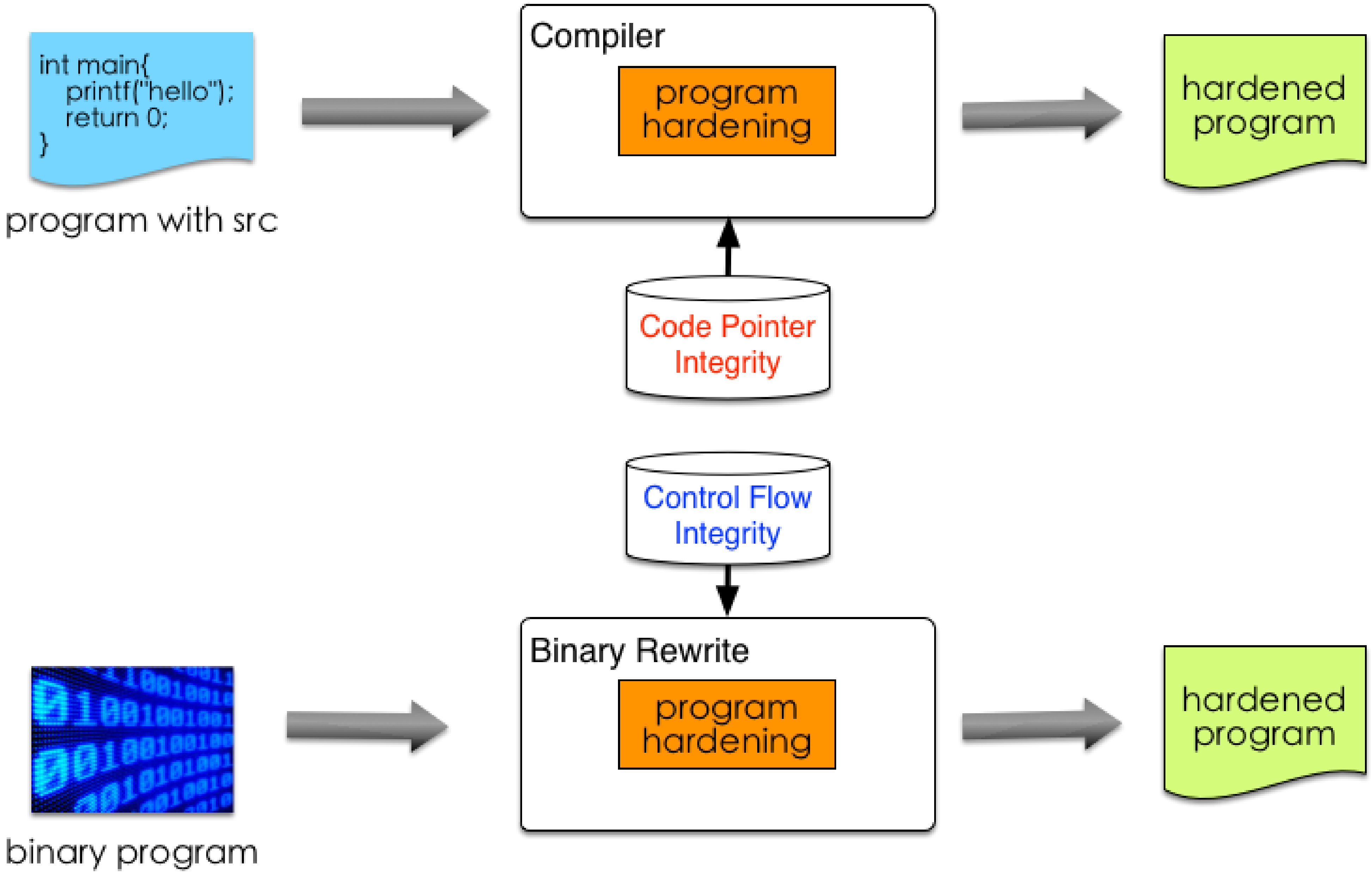
Control-flow integrity

The control-flow target should be legitimate.

Code Pointer integrity

The control-flow target cannot be tampered.

Our solutions



```
int main{
    printf("hello");
    return 0;
}
```

Overview

- Motivation
- Attack Surface Analysis of the Transportation CPS
- Program Hardening of CPS
    - ➡ Without source code: CCFIR
    - With source code: CPI

CCFIR’s policy

original

```
int *q = buf + input;
*q = input2;
…
(*func_ptr)();
```

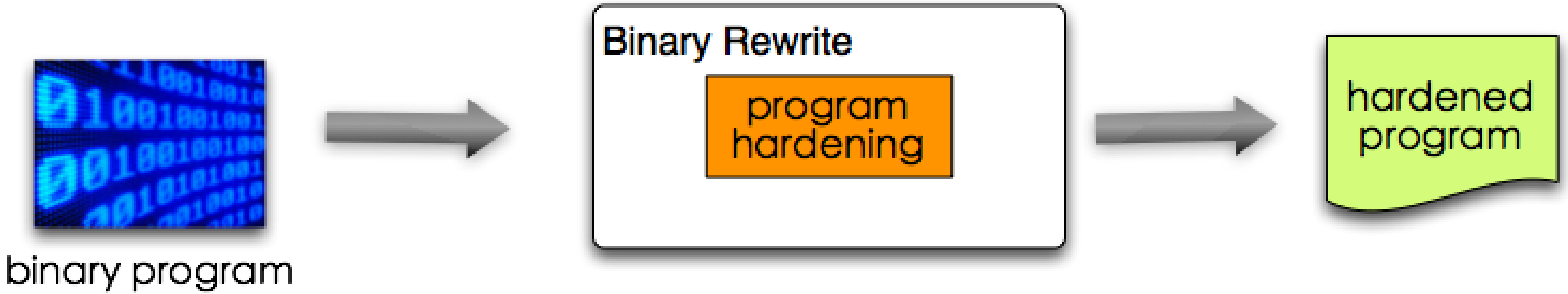
hardened

```
int *q = buf + input;
*q = input2;
…
if func_ptr ∈ Springboard:
        (*func_ptr)();
```

Springboard:
a special memory region instrumented by CCFIR,
cannot be modified by attckers

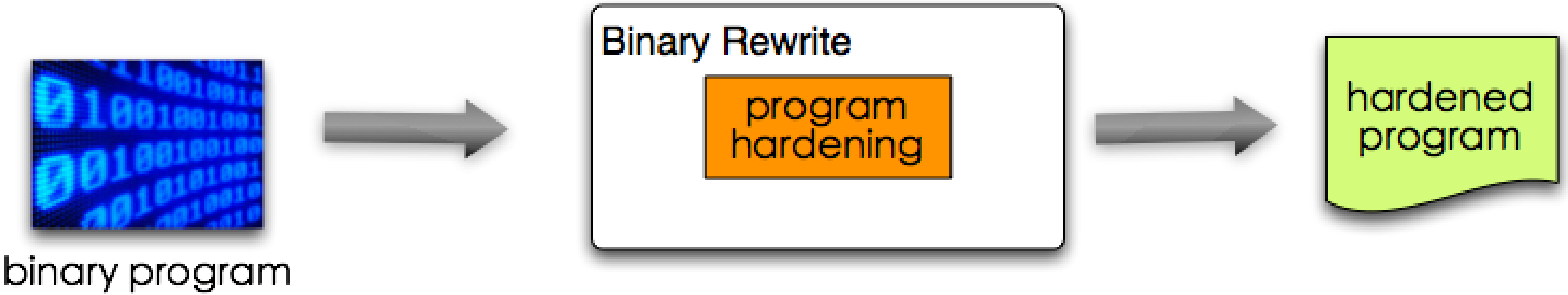
CCFIR: Practical Control Flow Integrity & Randomization for Binary Executables
Chao Zhang, Tao Wei, Zhaofeng Chen, Lei Duan, Laszlo Szekeres, Stephen McCamant, Dawn Song, Wei Zou.
IEEE Security & Privacy 2013

Architecture of CCFIR



- Understand the binary program
  - disassembly
  - a novel algorithm

Architecture of CCFIR



- Understand the binary program
- Rewrite the binary program
  - move all legitimate control-flow targets to Springboard
  - check all control-flow instructions’ target at runtime

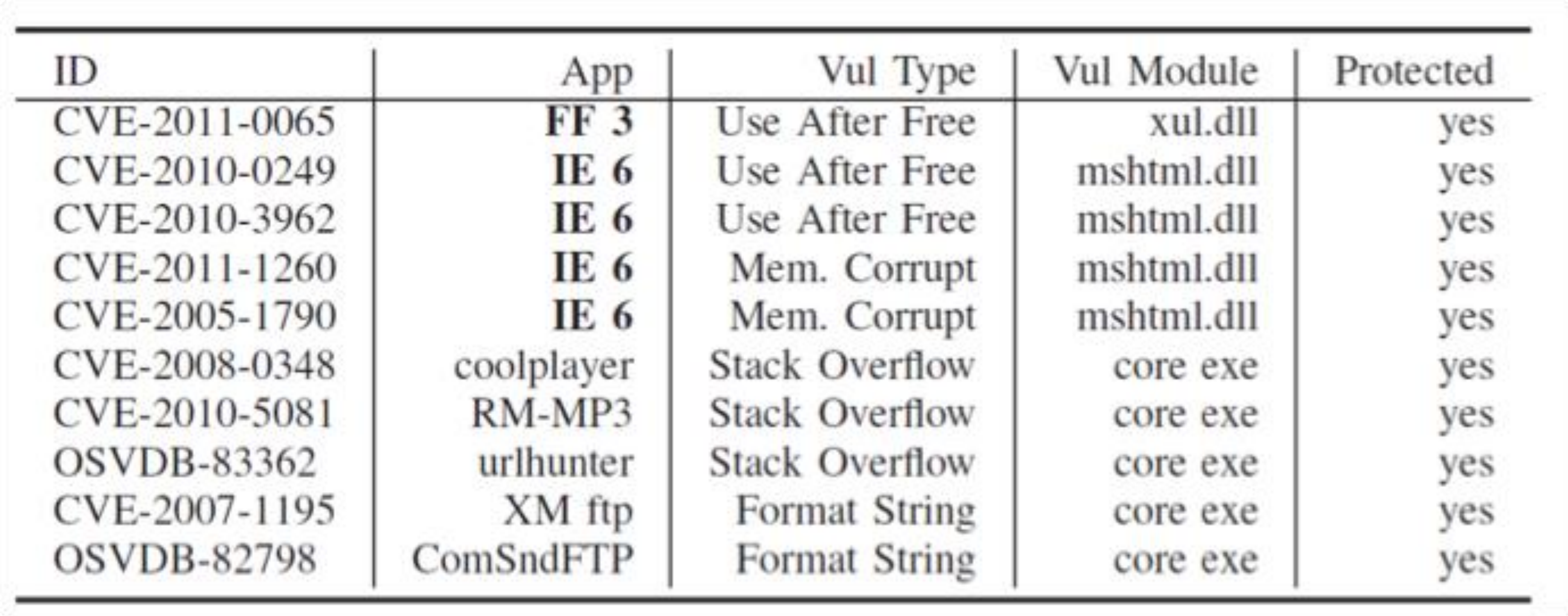
```
int *q = buf + input;
*q = input2;
…
if func_ptr ∈ Springboard:
    (*func_ptr)();
```

How good is CCFIR?

- Time to harden binary programs
  - ➢ SPECint2000 & SPECfp2000, 10s seconds
- Runtime Overhead
  - ➢ SPECint2000, average 3.6%, max 8.6%
  - ➢ SPECfp2000, average 0.59%, max 3.98%

How good is CCFIR?

- Defeat real world exploits



ID	App	Vul Type	Vul Module	Protected
CVE-2011-0065	**FF 3**	Use After Free	xul.dll	yes
CVE-2010-0249	**IE 6**	Use After Free	mshtml.dll	yes
CVE-2010-3962	**IE 6**	Use After Free	mshtml.dll	yes
CVE-2011-1260	**IE 6**	Mem. Corrupt	mshtml.dll	yes
CVE-2005-1790	**IE 6**	Mem. Corrupt	mshtml.dll	yes
CVE-2008-0348	coolplayer	Stack Overflow	core exe	yes
CVE-2010-5081	RM-MP3	Stack Overflow	core exe	yes
OSVDB-83362	urlhunter	Stack Overflow	core exe	yes
CVE-2007-1195	XM ftp	Format String	core exe	yes
OSVDB-82798	ComSndFTP	Format String	core exe	yes

Overview

- Motivation
- Attack Surface Analysis of the Transportation CPS
- Program Hardening of CPS
    - ➢ Without source code: CCFIR
    - ➡ With source code: CPI

python Java C# Swift ...

Sample Python program
(Dropbox SDK example):

Python program	3 KLOC of Python
Python runtime	500 KLOC of C
libc	2500 KLOC of C

python™ Java™ C# Swift …

C/C++	Overhead
SoftBound+CETS	116%
CCured (language modifications)	56%
Watchdog (hardware modifications)	29%
AddressSanitizer (approximate)	73%

Code Pointer Integrity

- Separate sensitive pointers and regular data

  Sensitive pointers =
  code pointers + ***indirect pointers to sensitive pointers***

- Enforce sensitive pointers accesses to be safe

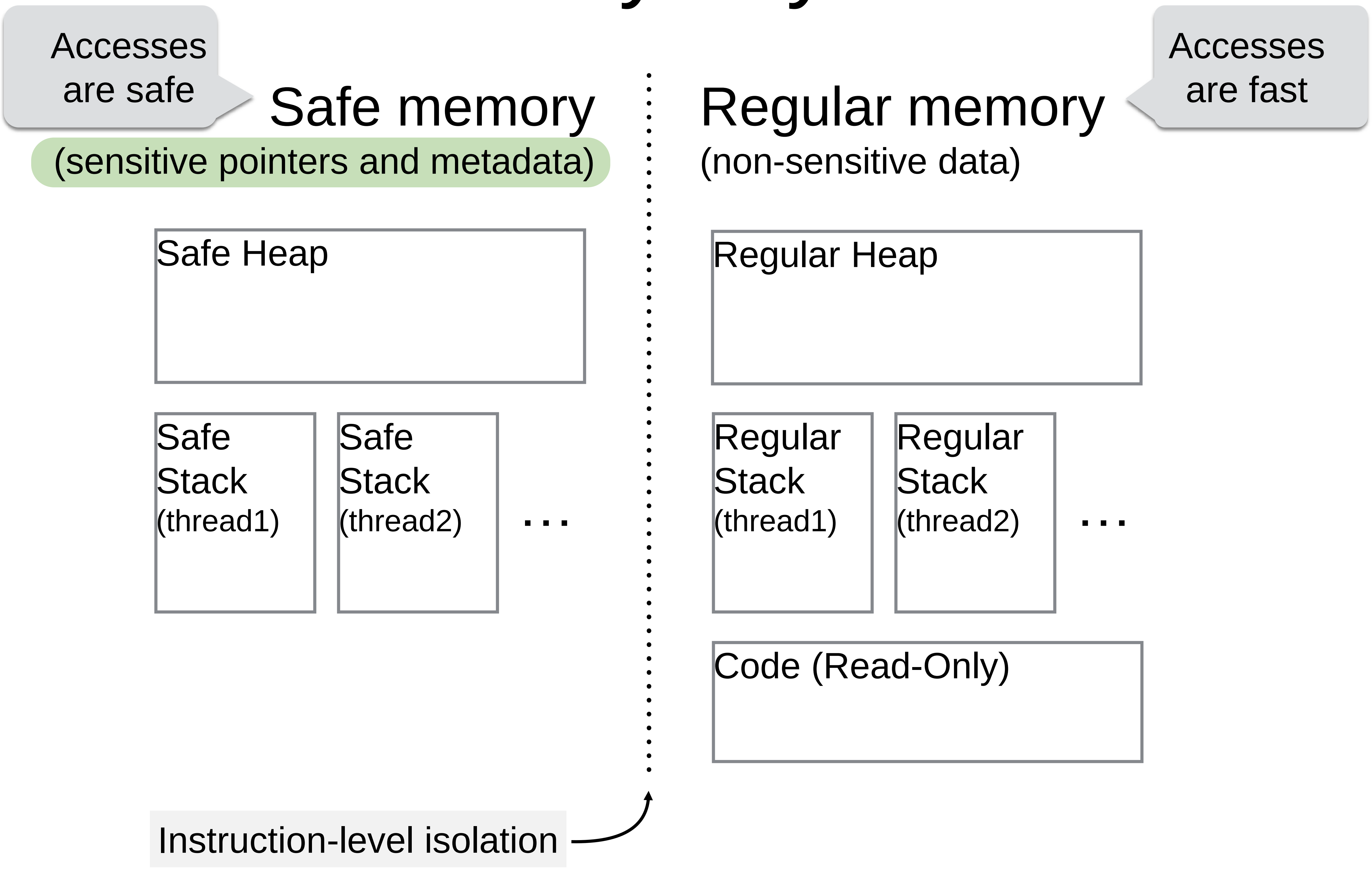
  Separation + ***runtime checks***

- Keep regular data accesses intact (fast)

  Instruction-level safe region isolation

Code Pointer Integrity
Volodymyr Kuznetsov, László Szekeres, Mathias Payer,
George Candea, R. Sekar, Dawn Song
OSDI'2014

Guaranteed Protection (CPI): Memory Layout



Accesses are safe

Safe memory

(sensitive pointers and metadata)

Safe Heap

Safe Stack (thread1)

Safe Stack (thread2)

. . .

Accesses are fast

Regular memory

(non-sensitive data)

Regular Heap

Regular Stack (thread1)

Regular Stack (thread2)

. . .

Code (Read-Only)

Instruction-level isolation

Guaranteed Protection (CPI)

Guaranteed memory safety for all sensitive[1] pointers

⇓

Guaranteed protection against control-flow hijack attacks enabled by memory bugs

[1] Sensitive pointers = code pointers and **pointers used to access sensitive pointers**

How secure is it?

- RIPE[1] defense evaluation benchmark:
  - ➢ **CPI prevents all attacks from RIPE**
- Future attacks:
  - ➢ **Formal proof of CPI correctness in the paper**

[1]Wilander at al., ACSAC 2011

How practical is it?

```
cc -fcpi foo.c
```

- LLVM-based prototype at http://levee.epfl.ch
- Plan to integrate upstream into LLVM

Full OS Distribution

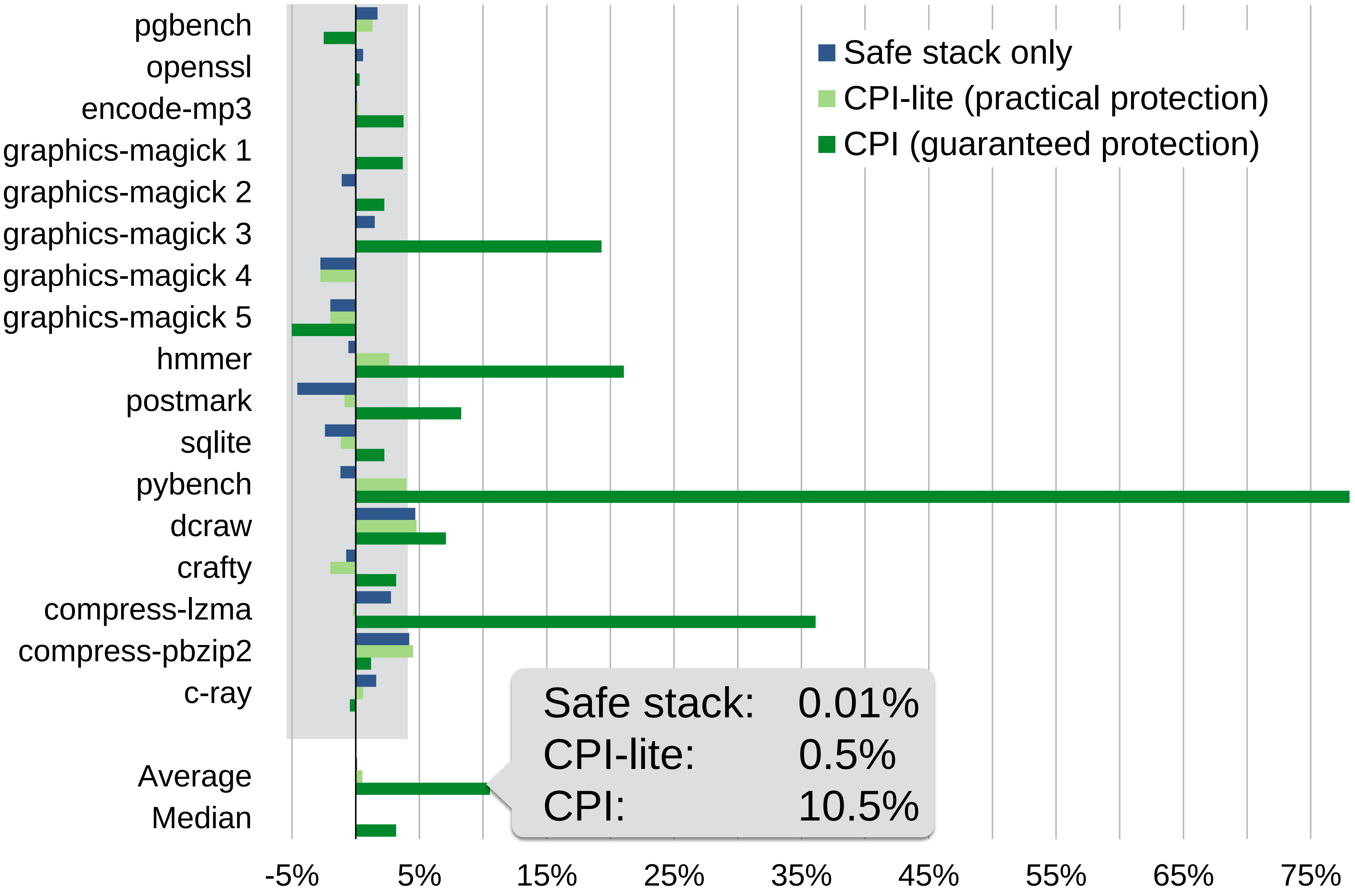
FreeBSD hardened

- Recompiled the entire FreeBSD userspace…
- … and more than 100 packages

python™ SQLite lame OpenSSL

PostgreSQL GraphicsMagick Apache

Performance overhead on Phoronix



Safe stack: 0.01%
CPI-lite: 0.5%
CPI: 10.5%

Code-Pointer Integrity

Control-flow hijack protection

Practical protection

Guaranteed protection

and

Unmodified C/C++

0.5 - 1.9% overhead

8.4 - 10.5% overhead

Key insight: memory safety for code pointers only

FreeBSD hardened

python

SQLite

lame

OpenSSL

PostgreSQL

GraphicsMagick

Apache

http://levee.epfl.ch

Ongoing Work

- Deploy program hardening to real-world CPS system
  - ➢ CCFIR
  - ➢ CPI
- Find other potential attacks against CPS system

Conclusion

- CPS systems are vulnerable
    - Case study: transportation system attacks [in collaboration with Prof. Bayen]
- Program hardening is necessary and effective to protect CPS systems.
- Two new solutions to automatically harden programs.
    - with or without program source code
    - low overhead, full system protection

Thanks!