Some Reflections on Working with Ed Clarke

Somesh Jha

University of Wisconsin


VSL 2014

Entered CMU in 1992

- After working in IBM in the compiler group
- Immigration Course
  - Various faculty speak about their research
- Thoroughly confused!

Nico Haberman

- Chair of CS
- Used to come to several talks
- I requested a meeting
- ~30 minutes chat

Working with Ed
First Few Papers

- Verification of Futurebus+ Cache-Coherence Protocol
- Symmetry in Model Checking
- Improved fix-point algorithms

Some General Thoughts

- Insisted that all his students take grad logic I and 2 with Peter Andrews
  - Used ETPS (Thanks Frank!)

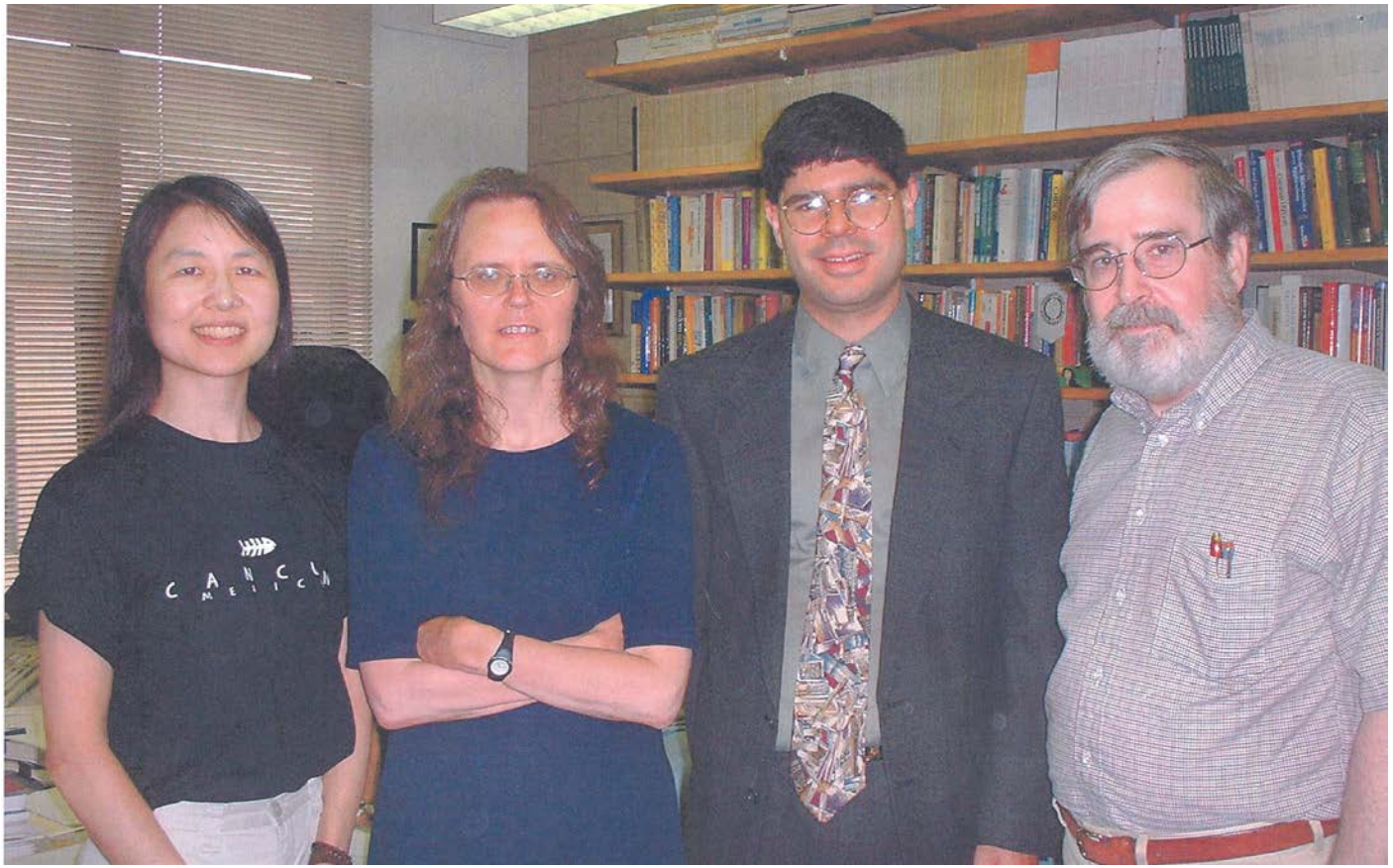
- Never stopped me from taking classes
  - Took a ton of classes!

Some General Thoughts

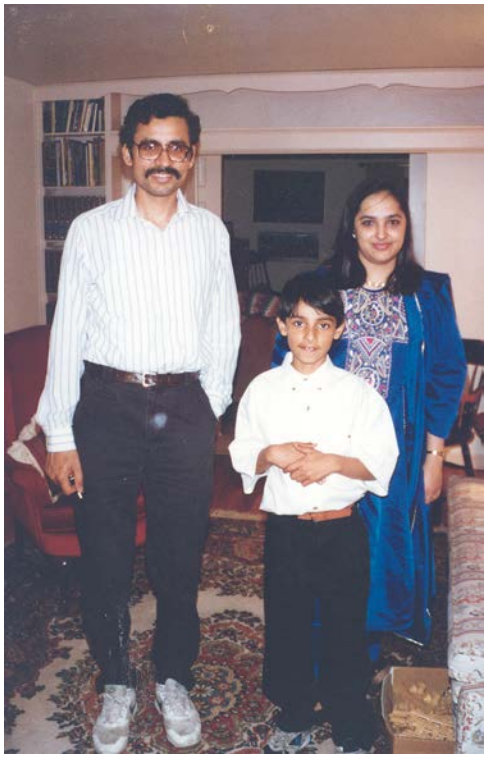
- Really good about fostering collaborations
- Really good at making “abstract connections”

- Responsible for getting me into security
  - Brutus (Marrero, Clarke, Jha)
  - Combined model checking with natural deduction

Marrero's Defense




The
Godfather




LOUISIANA

Synthesis of Secure Programs

News is Grim

- See talks at
  - DARPA Cyber Colloqium
  - http://www.darpa.mil/Cyber_Colloqium_Presentations.aspx
- What do we do?

Clean-slate Design

- Rethink the entire system stack

- Networks
  - NSF program
    - See http://cleanslate.stanford.edu
  - See DARPA Mission Resilient Clouds (MRC) program
- Hosts
  - DARPA CRASH program

Some Interesting Systems

- Operating systems with powerful capabilities
  - Asbestos, HiStar, Flume
  - Capsicum
  - ….
- Virtual-machine based
  - Proxos
  - Overshadow
- Possible to build applications with strong guarantees
  - *Web server*: No information flow between threads handling different requests

What happens to all the code?

- Should we implement all the code from scratch?
- Can we help programmers adapt their code for these new platforms?

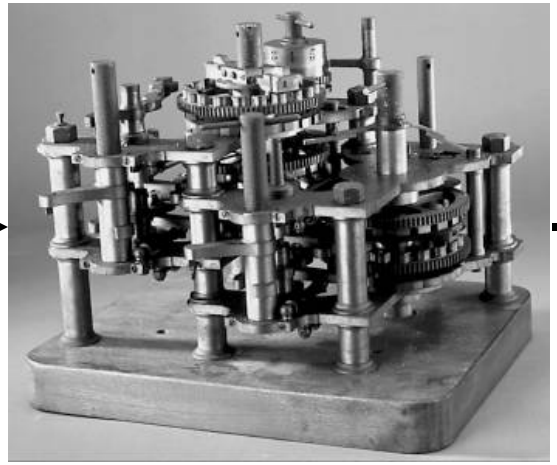
- Analogy
  - We have strong foundation
  - Can we build a strong house on top of it?

Retrofitting legacy code

Need systematic techniques to retrofit legacy code for security

Legacy code

INSECURE



Retrofitted code

SECURE

Premise

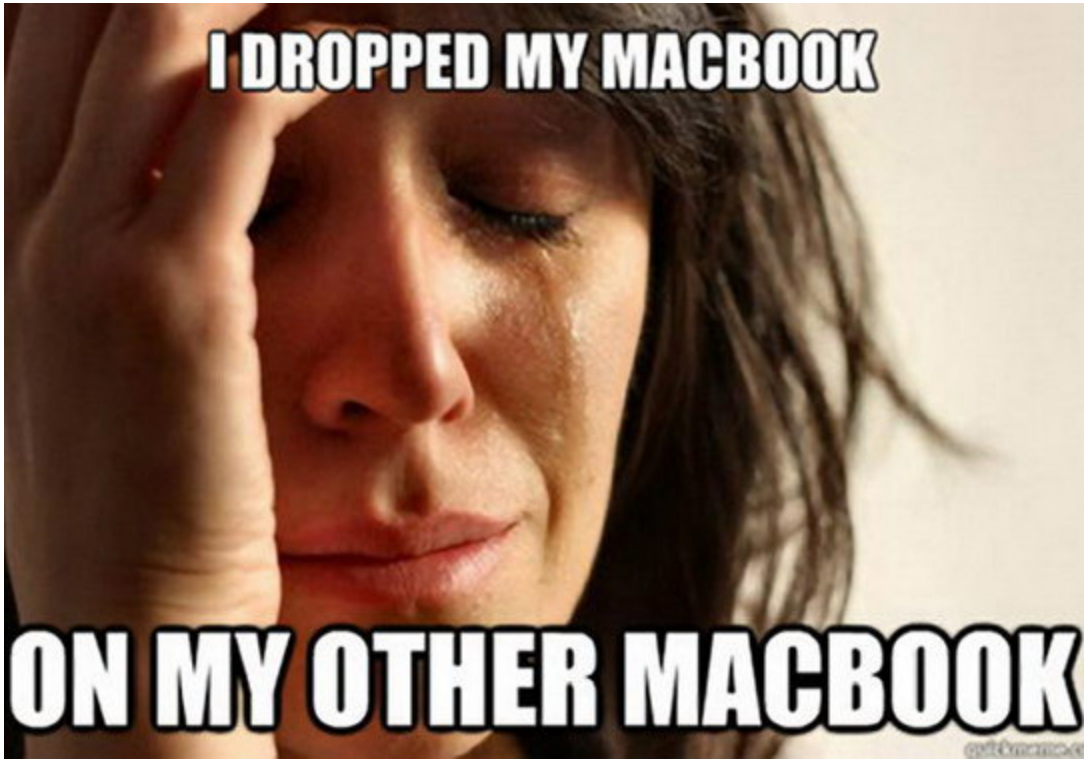
- Techniques and ideas from
  - Verification
  - Static Analysis
  - …
- Can help with this problem

Collaborators and Funding

National Science Foundation
WHERE DISCOVERIES BEGIN

DARPA

The Problem

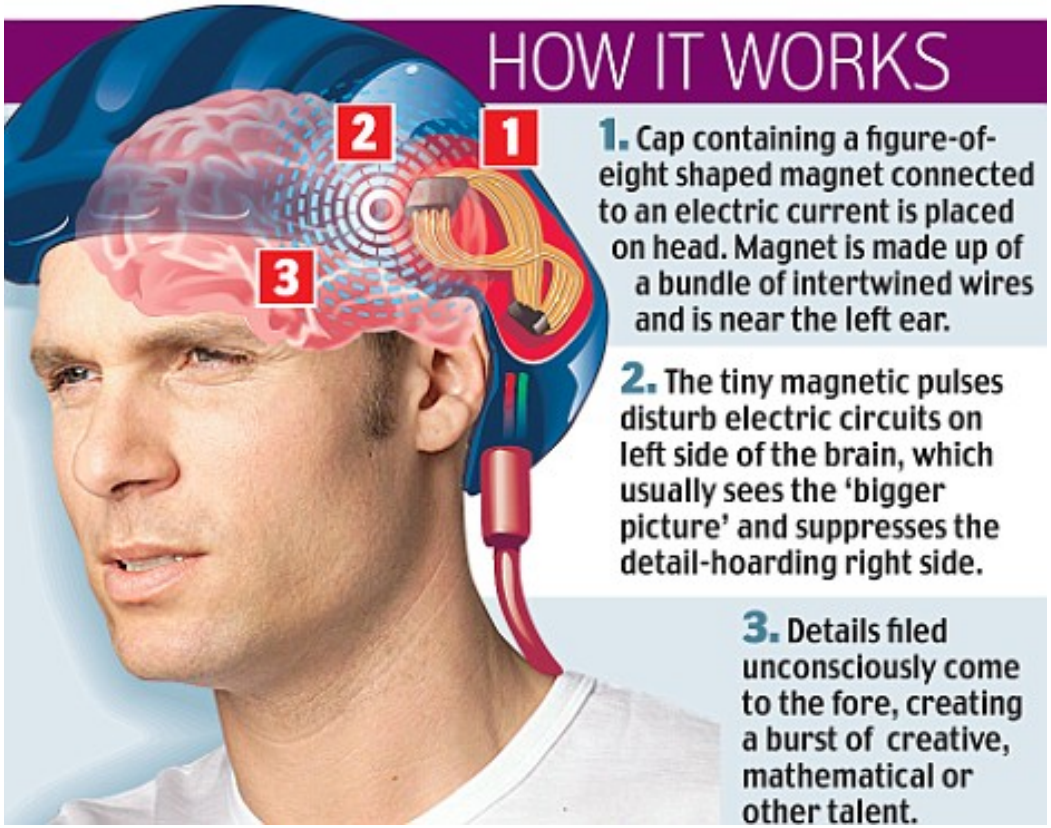


Rewriting Programs for a Capability System
[Harris et. al., Oakland 2013]

- Basic problem: take an insecure program and a policy, instrument program to invoke OS primitives to satisfy the policy

- Key technique: reduce to safety game between program and instrumentation

The Technique

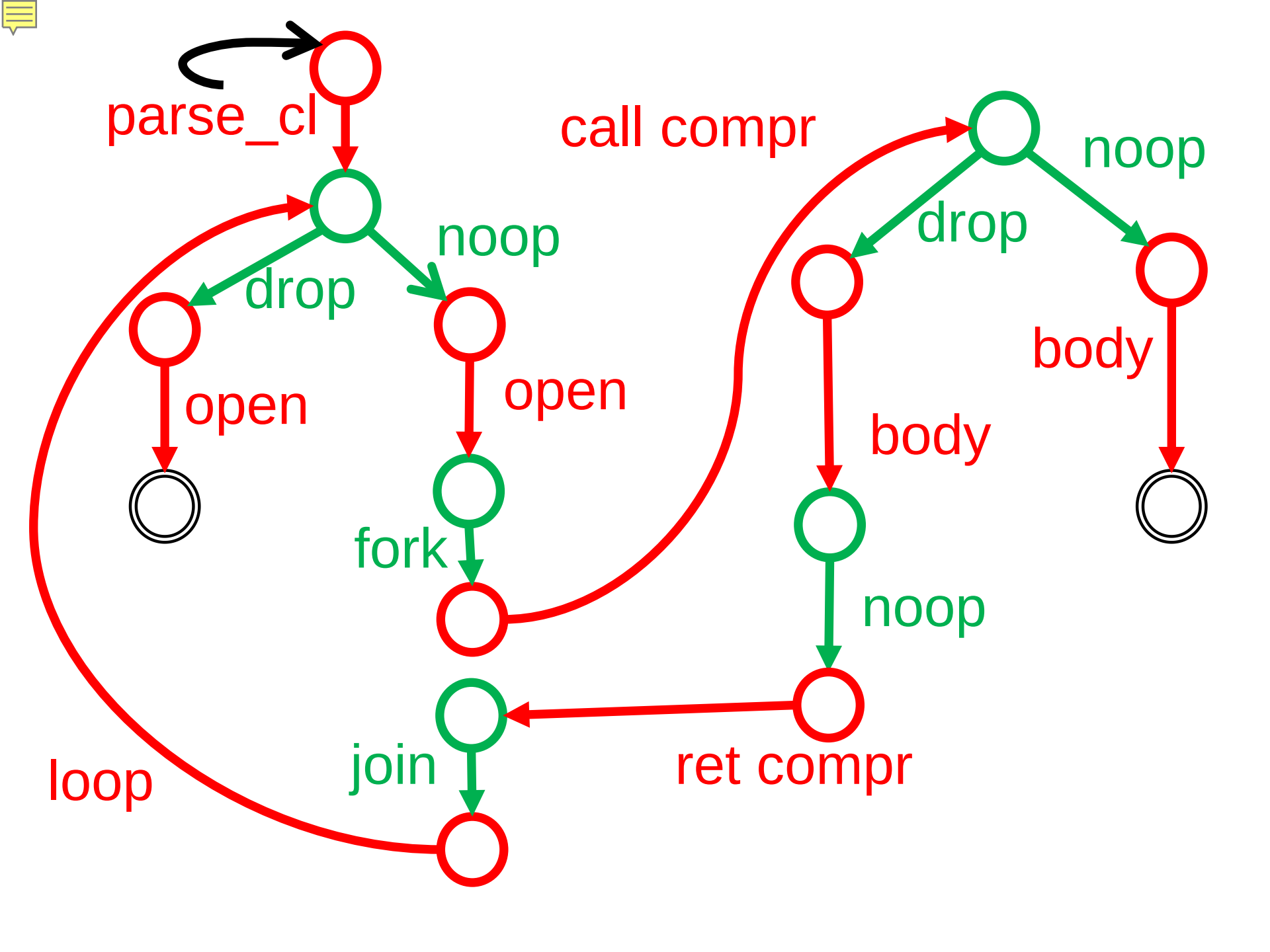


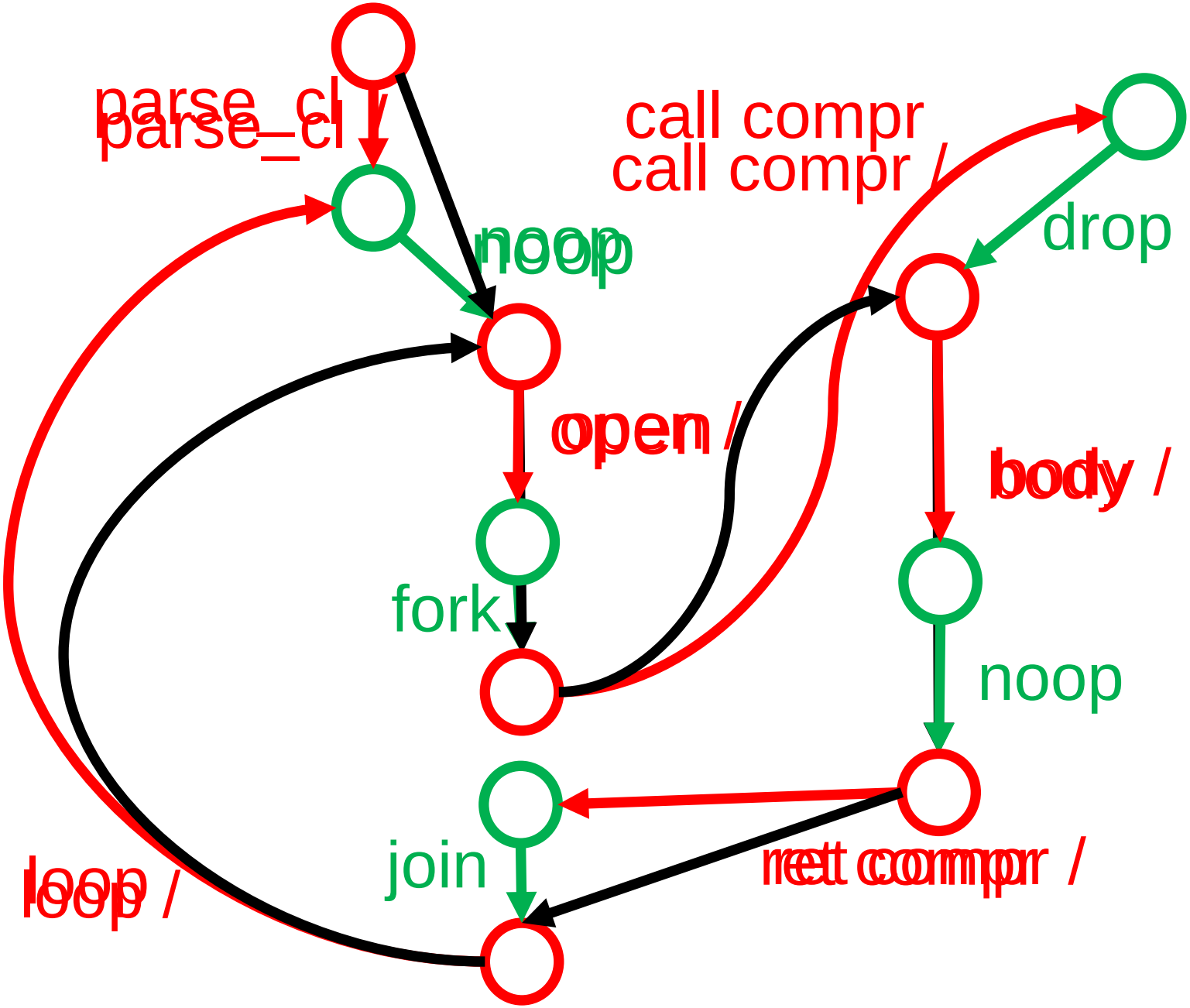
Weaving as a Game

Two steps:

1. Model uninstrumented program, policy, and Capsicum as languages/automata

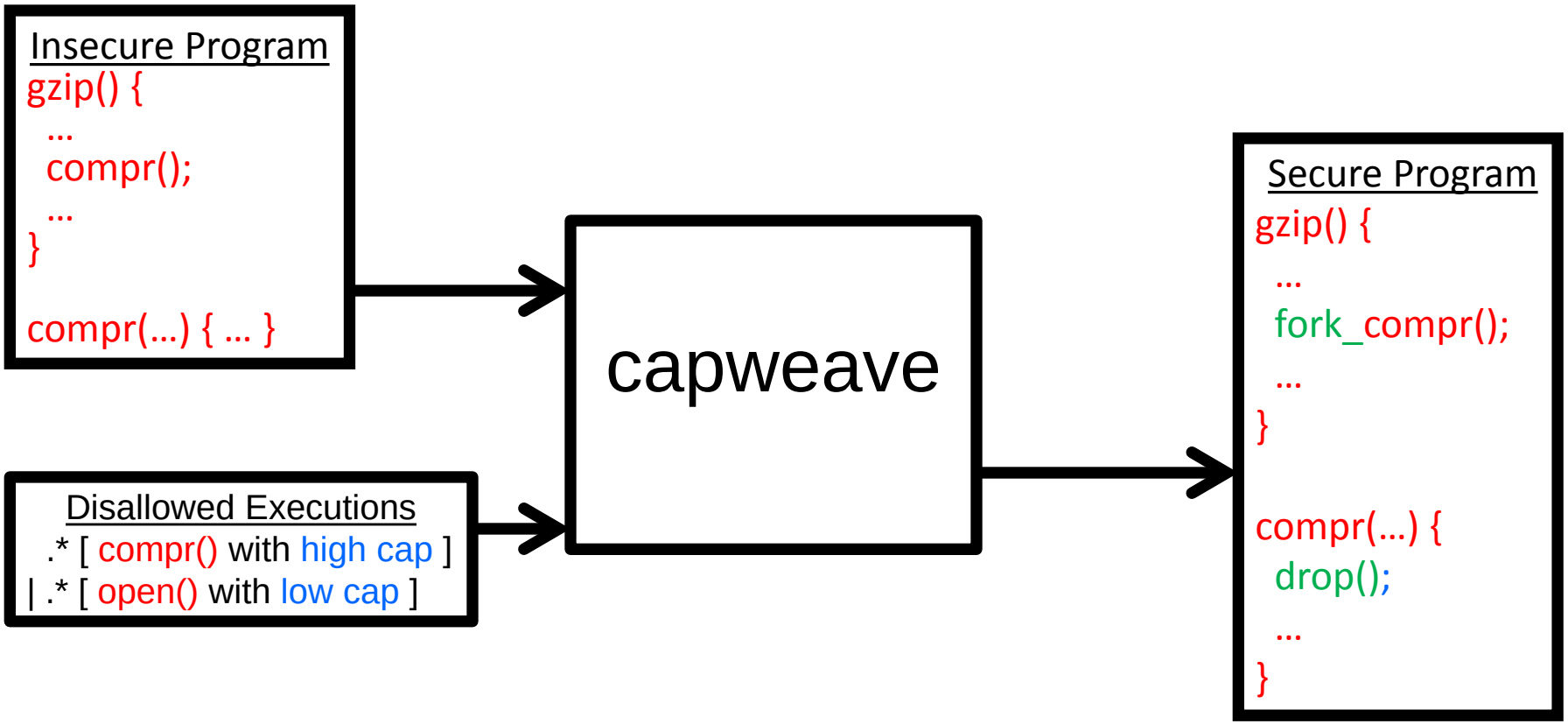
2. From automata, translate weaving problem to a two-player safety game


parse_cl
drop
noop
open
open
fork
loop
join
call compr
noop
drop
body
body
noop
ret compr


parse_cl /
noop
call compr /
drop
open /
body /
fork
noop
join
loop /
rett compr /

Questions

Summary


Insecure Program
gzip() {
…
compr();
…
}
compr(…) { … }
Disallowed Executions
.* [compr() with high cap]
| .* [open() with low cap]
capweave
Secure Program
gzip() {
…
fork_compr();
…
}
compr(…) {
drop();
…
}