U.S. NAVAL RESEARCH LABORATORY

Shadow: Scalable and Deterministic Network Experimentation

Dr. Rob Jansen
U.S. Naval Research Laboratory
Center for High Assurance Computer Systems

Cybersecurity Experimentation of the Future
Community Engagement Event
Marina Del Rey, CA
May 15th, 2018

U.S. NAVAL RESEARCH LABORATORY

The Science of Cybersecurity

- The most important property of experiments:
  - Experimental control – isolate important factors
  - Easily achievable with deterministic experimentation
  - Determinism yields repeatable / reproducible experiments

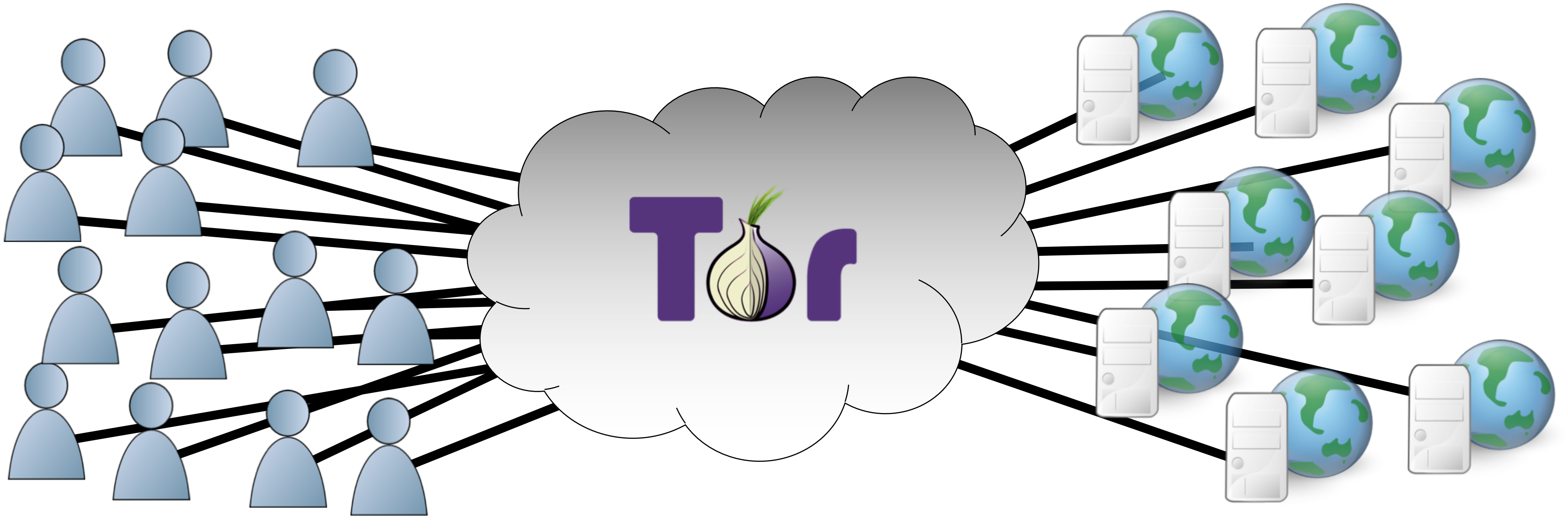
- Requirements for large distributed systems (e.g., Tor)
  - Realistic – execute system software (not an abstraction)
  - Scalable – can run studied system at scale

- Shadow
  - Network simulator with above design goals

Tor Experimentation

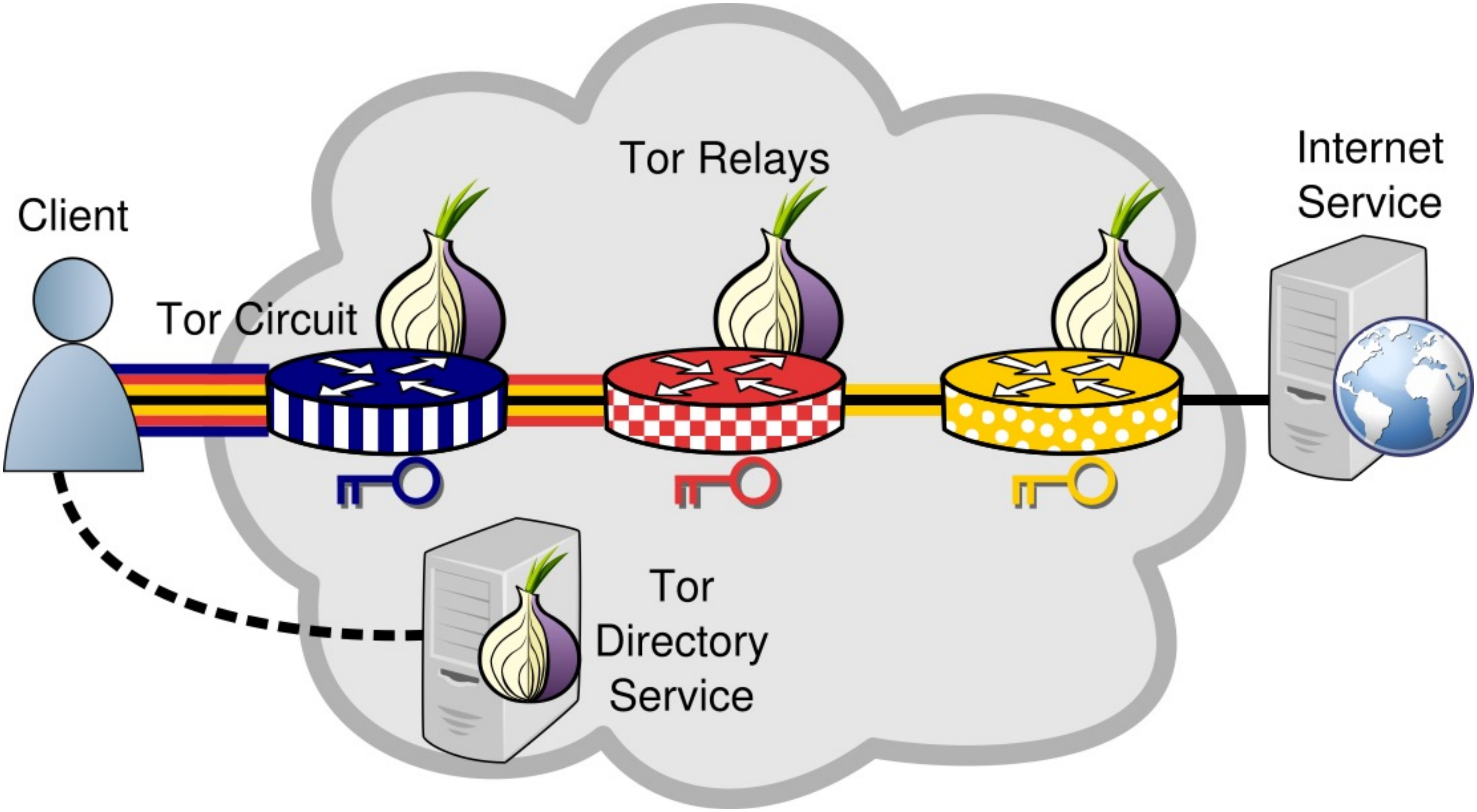
Tor Overview

Tor: a censorship resistant, privacy-enhancing anonymous communication system



~6500 Relays, 100 Gbit/s
Estimated ~2 M. Users/Day
(metrics.torproject.org)

Onion Routing



Tor Experimentation Options

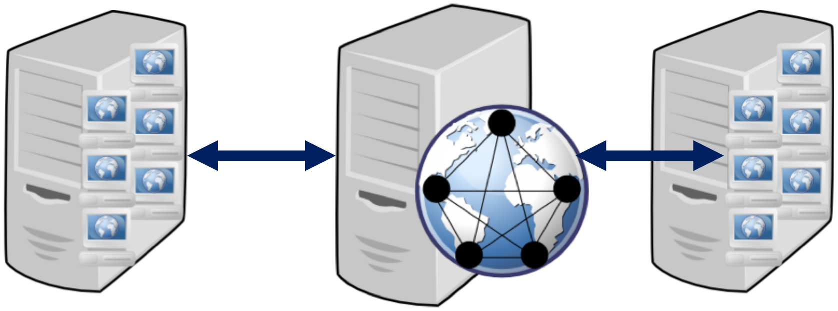
More Realistic, Costly ↑

Approach	Notes
Live Network	• Target environment, most “realistic” • Lengthy deployment, security risks
Testbed	• Target OS, uses Internet protocols • Requires significant hardware investment
Emulation	• Target OS, uses Internet protocols • Large VM overhead
Simulation	• Deterministic, scalable, decoupled from real time • Abstractions reduce realism

More Control, Scalable ↓

Simulation vs. Emulation: Realism

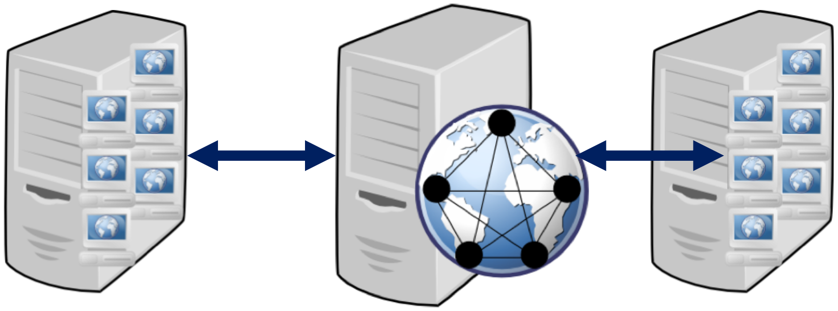
Simulation	Emulation
Abstracts away most system components	Runs the real OS, kernel, protocols, applications
Simulator is generally only internally consistent	Software is interoperable with external components
Less resource intensive	More resource intensive



U.S. NAVAL RESEARCH LABORATORY

Simulation vs. Emulation: Time

Simulation	Emulation
“As-fast-as-possible”	Real time
Control over clock, can pause time without issue	Time must advance in synchrony with wall-clock
Weak hardware extends total experiment runtime	Weak hardware causes glitches that are difficult to detect and diagnose

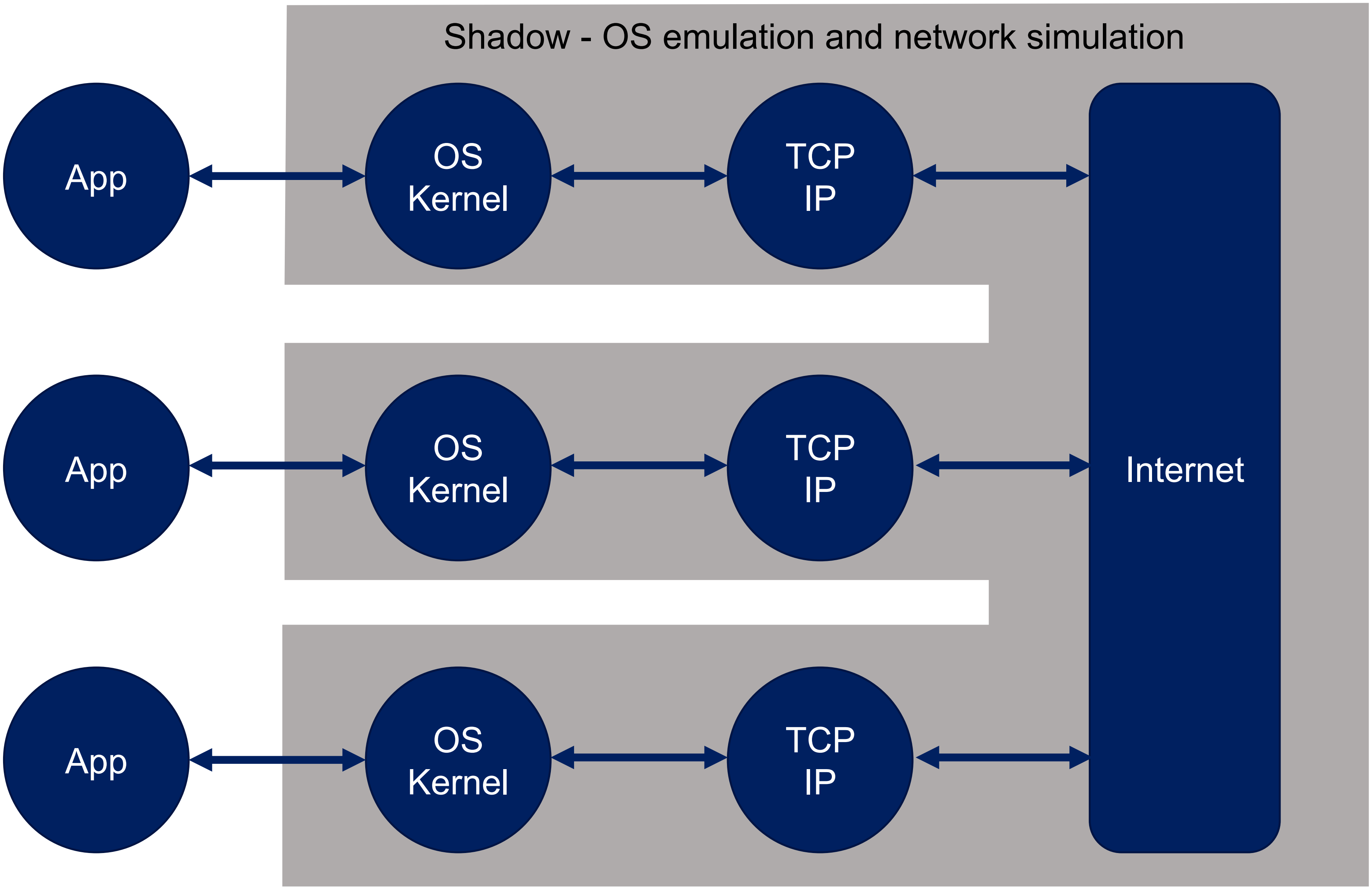


Shadow Design

What is Shadow?

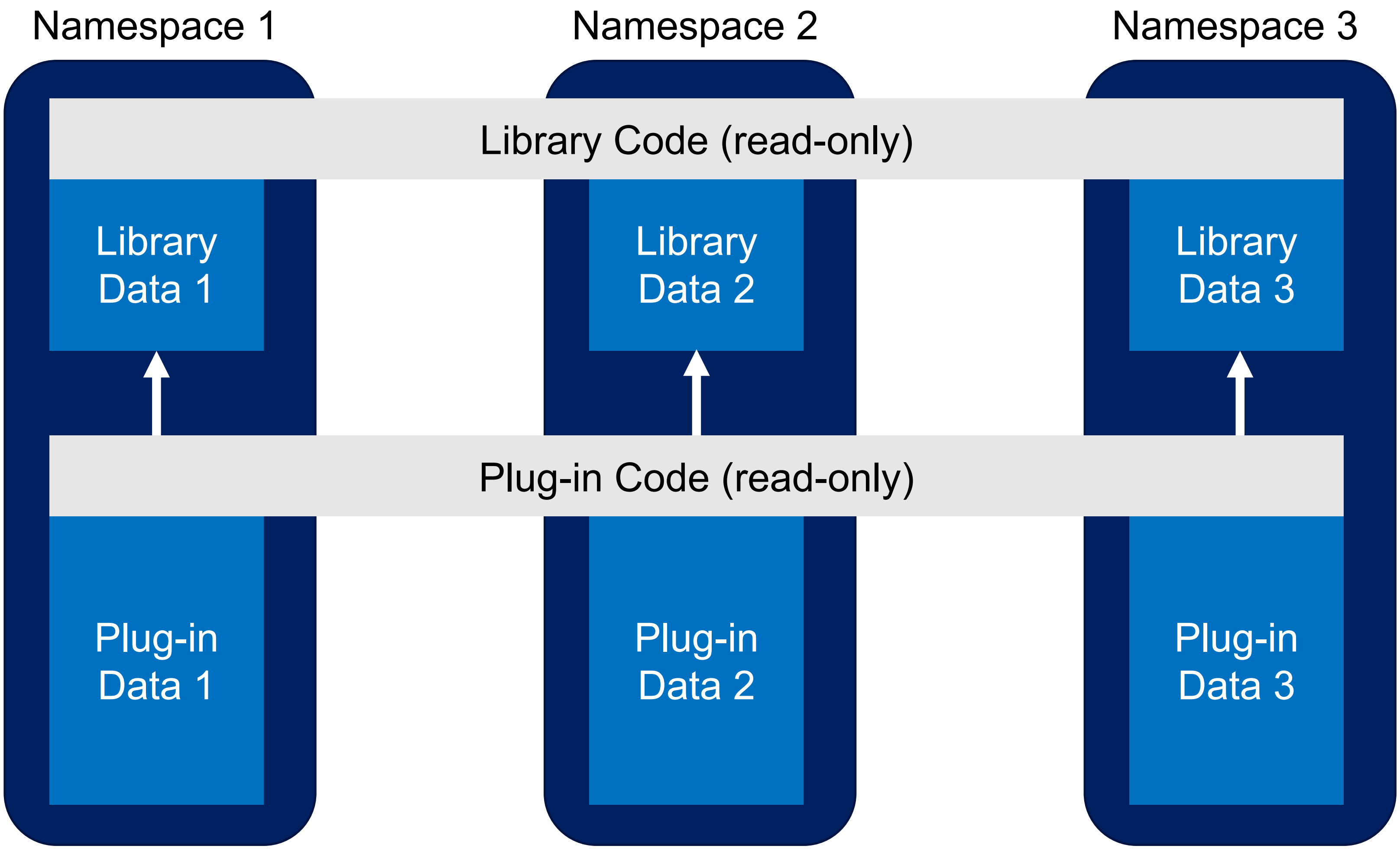
- Deterministic, parallel discrete-event network simulator
- Directly executes apps as plug-ins (e.g., Tor, Bitcoin)
- Models routing, latency, bandwidth
- Simulates time, CPU, OS
  - TCP/UDP, sockets, queuing, threading
- Emulates POSIX C API on Linux

How does Shadow Work?

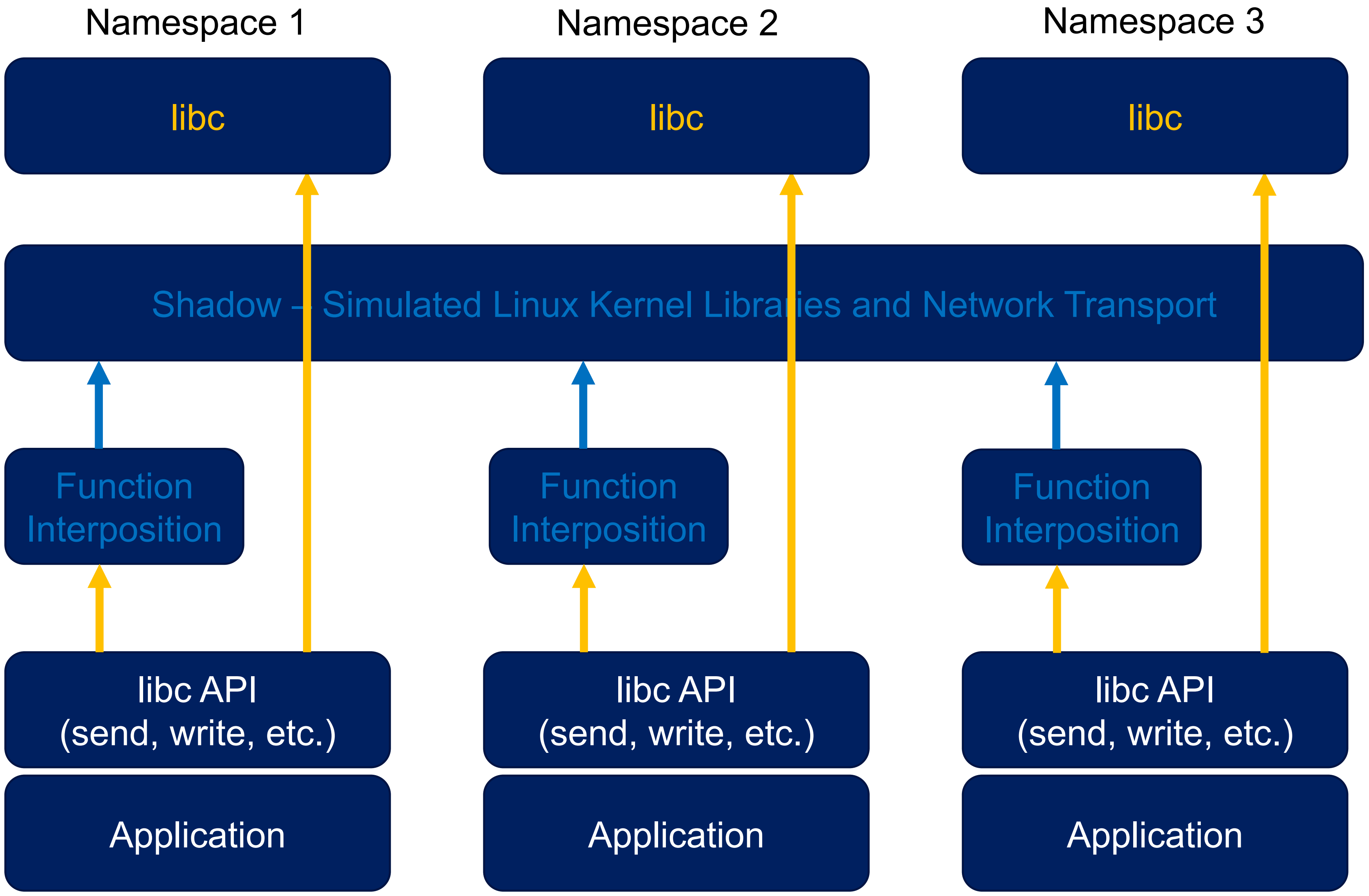


App Memory Management

Apps loaded in independent namespaces, “copy-on-write”



Direct Execution in a Simulator


Namespace 1
Namespace 2
Namespace 3
libc
libc
libc
Shadow – Simulated Linux Kernel Libraries and Network Transport
Function Interposition
Function Interposition
Function Interposition
libc API (send, write, etc.)
libc API (send, write, etc.)
libc API (send, write, etc.)
Application
Application
Application

Shadow Uses Cases

- Tor
  - Latency and throughput correlation attacks
  - Denial of Service attacks (sockets, RAM, bandwidth)
  - Changes to path selection algorithms
  - Traffic admission control algorithms
  - Traffic scheduling and prioritization algorithms
  - Network load balancing algorithms
  - Process RAM consumption and optimization
- Network and memory attacks in Bitcoin
- Distributed secure multiparty computation algorithms
- Software debugging

Questions

Dr. Rob Jansen
Center for High Assurance Computer Systems
U.S. Naval Research Laboratory
rob.g.jansen@nrl.navy.mil
robgjansen.com, @robgjansen

The Shadow Simulator
shadow.github.io
github.com/shadow