Intrusion Tolerant Cloud Infrastructure

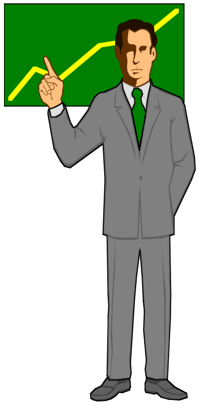
Daniel Obenshain, Marco Platania, Tom Tantillo, Yair Amir

Department of Computer Science
Johns Hopkins University

Andrew Newell, Cristina Nita-Rotaru

Department of Computer Science
Purdue University

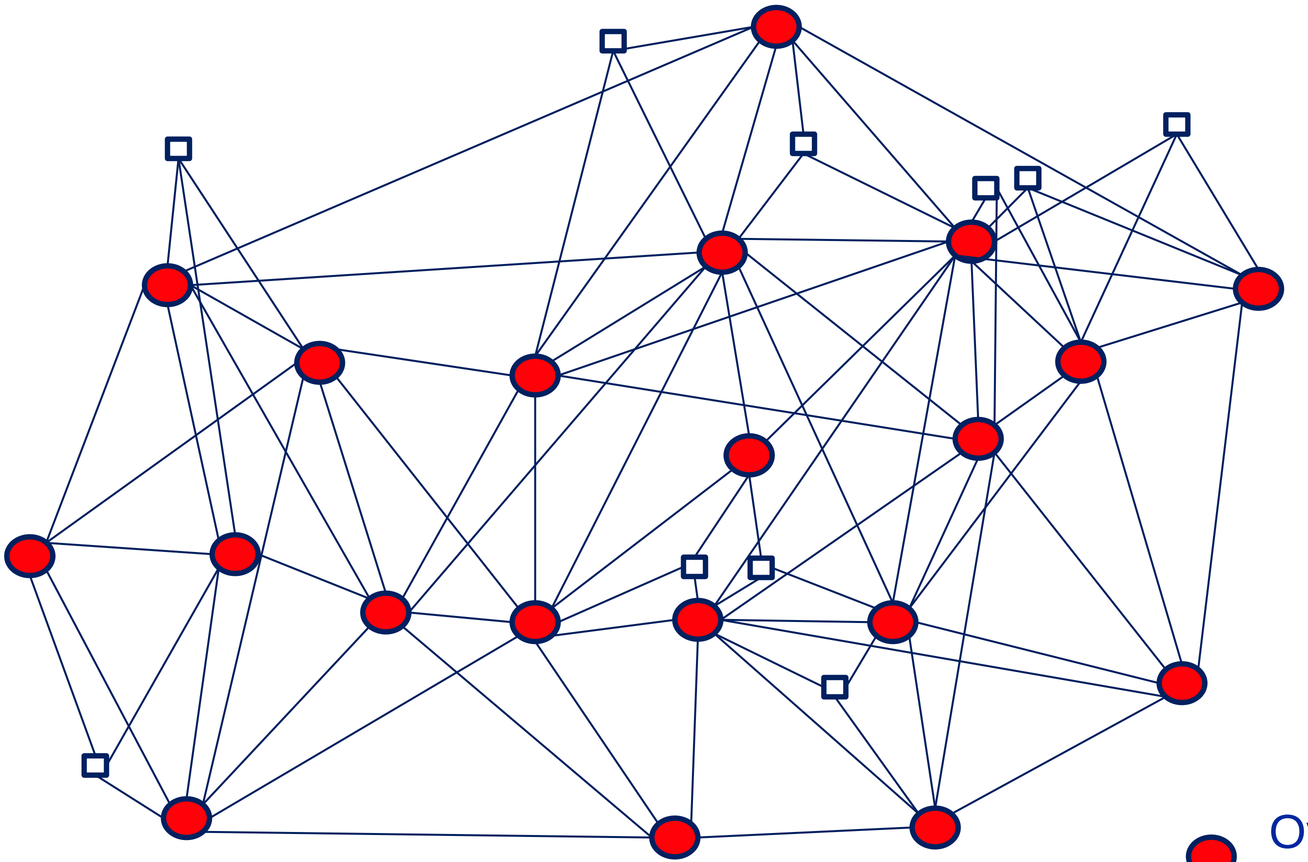
http://www.dsn.jhu.edu



Intrusion Tolerant Cloud Infrastructure

- Starting point: No practical intrusion-tolerant messaging and replication that can perform well on a global scale
- Intrusion-Tolerant Messaging for Cloud Monitoring & Control
  - Cloud infrastructure is remote to its administrators
  - Cloud management must be done through monitoring and control messaging
  - The chicken and the egg – at least part of the cloud software has to work to allow its administrators to make it work (e.g. react to attacks)
  - Result: Monitoring and control messaging must be intrusion-tolerant
- Intrusion-Tolerant Replication of Cloud Infrastructure State
  - Safety, Liveness, and Guaranteed Performance under attack
  - Requires that no more than *f* out of *3f+1* total replicas be compromised simultaneously
  - Result: Proactive recovery combined with diversity limits the adversary's window of opportunity

A Network Model of the Cloud

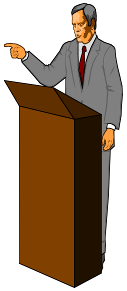


Overlay Node
(in datacenters)

Source / Destination
(cloud clients)

Outline

- Project goal
- Intrusion-tolerant messaging (**Spines**)
  - Flooding and K-Path Routing disseminations
  - Monitoring: Priority-based dissemination
  - Control: Reliable dissemination
- Intrusion-tolerant Replication (**Prime**)
  - Proactive recovery: defense across space and time
  - Theoretical model: resiliency through proactive recovery
  - Physical and virtual approaches
  - Support for 1 Terabyte application state size



Intrusion Tolerant Messaging

- Monitoring and control messaging must be intrusion-tolerant to protect the cloud infrastructure
- Normal routing algorithms are insufficient
  - Compromised trusted nodes can disrupt the routing protocol
- Any node (even all nodes) can be a source
- Any node can be compromised
  - Compromised nodes may be undetectable
  - Cannot prefer one node's traffic over another's
- Protected by cryptographic mechanisms

Intrusion Tolerant Messaging

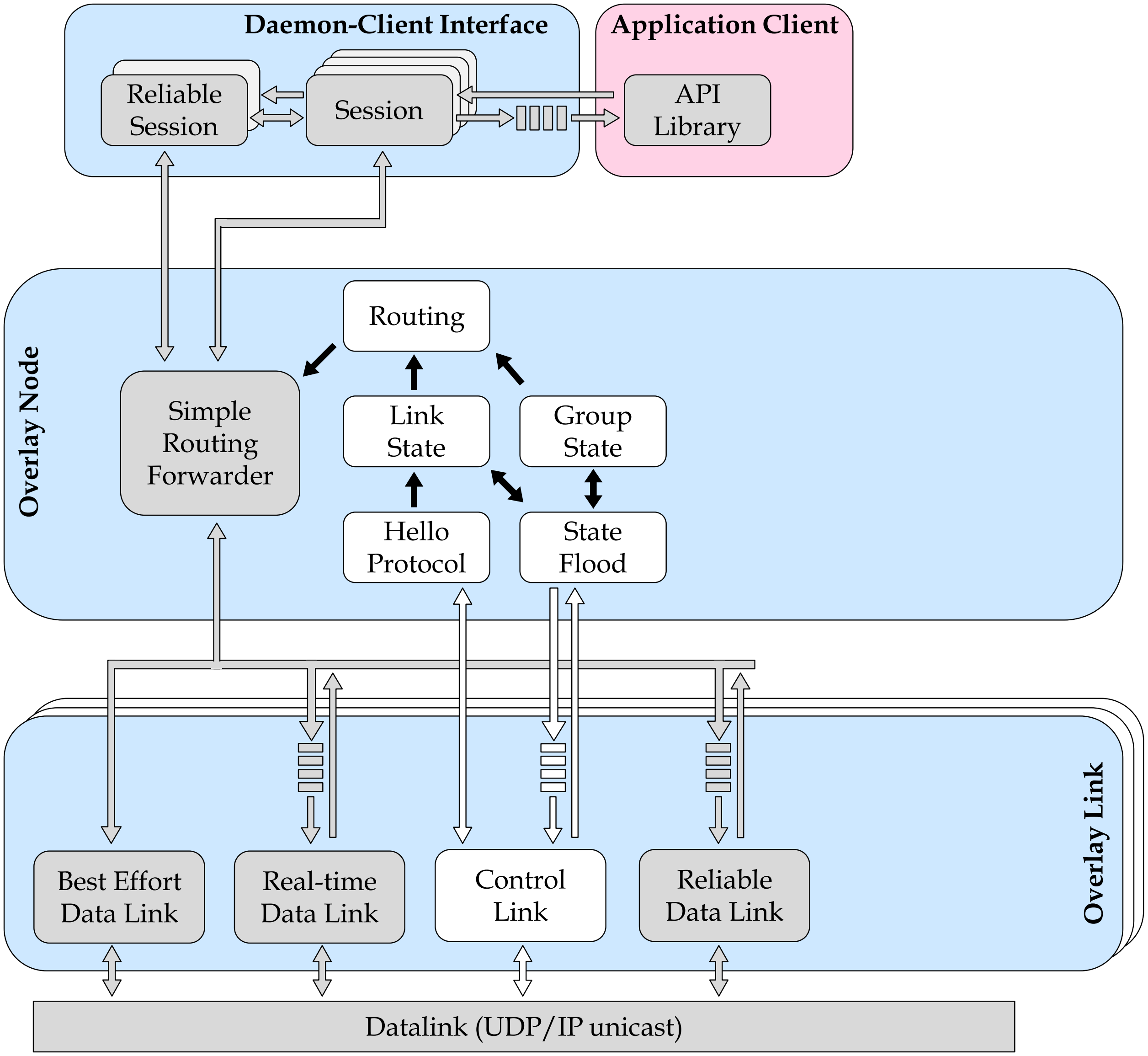
	Priority-Based	Reliable
Controlled Flooding	Source-fair scheme	Source-destination pair fair scheme
	Source-defined priority for each message	Back pressure employed all the way back to the source
K-Paths Routing	Select a source in round-robin order and send its highest priority message	Keep message until all neighbors have it (option) or end-to-end ACK is received
	Motivated by the real-time demands of cloud monitoring messages	Motivated by the reliability demands of cloud control messages

Intrusion Tolerant Messaging

	Priority-Based	Reliable
Controlled Flooding	Controlled authenticated flooding on a specified subset of the network topology	
K-Paths Routing	Source-based routing on *K* node-disjoint paths	

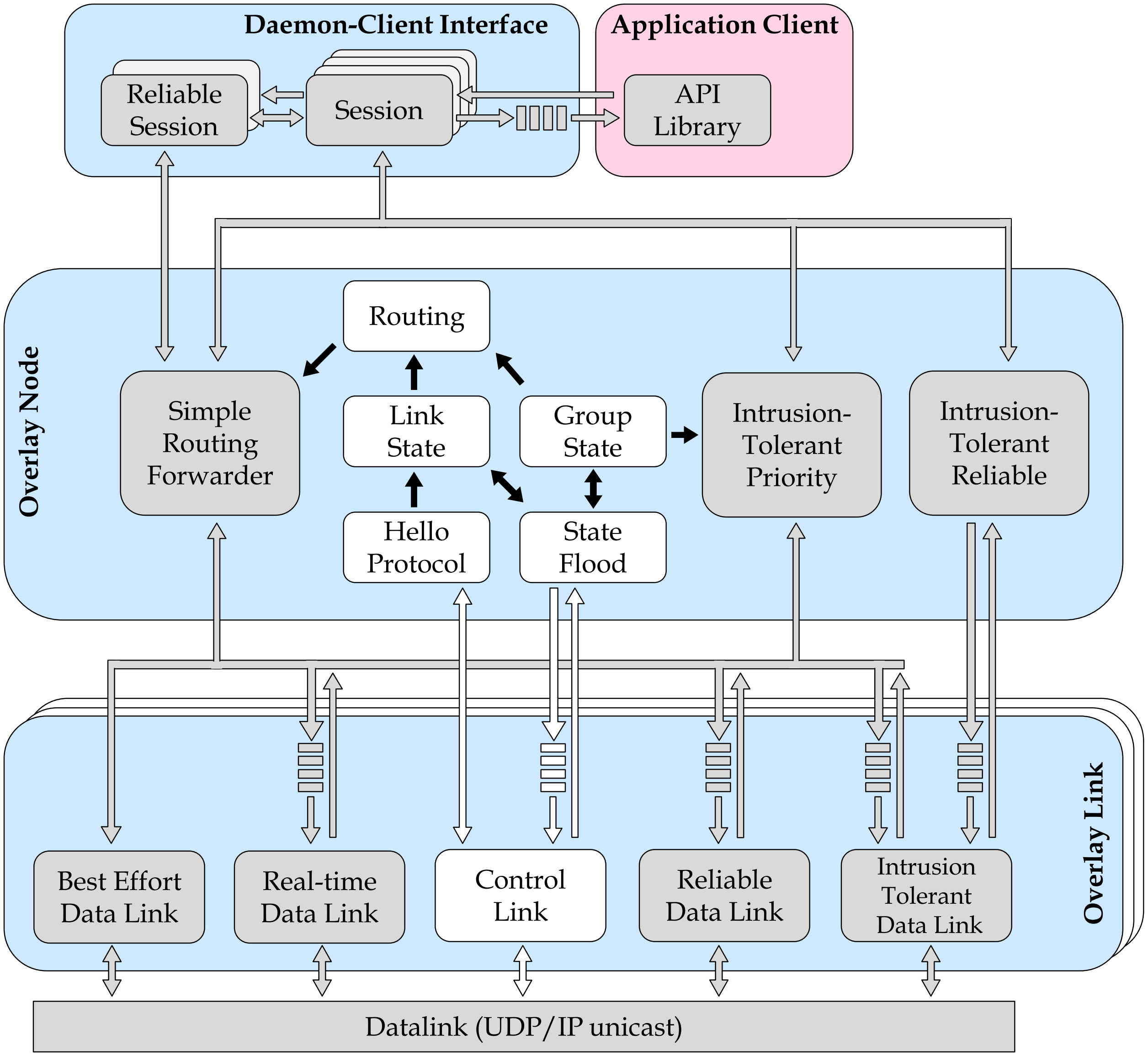
The Spines Architecture

www.spines.org



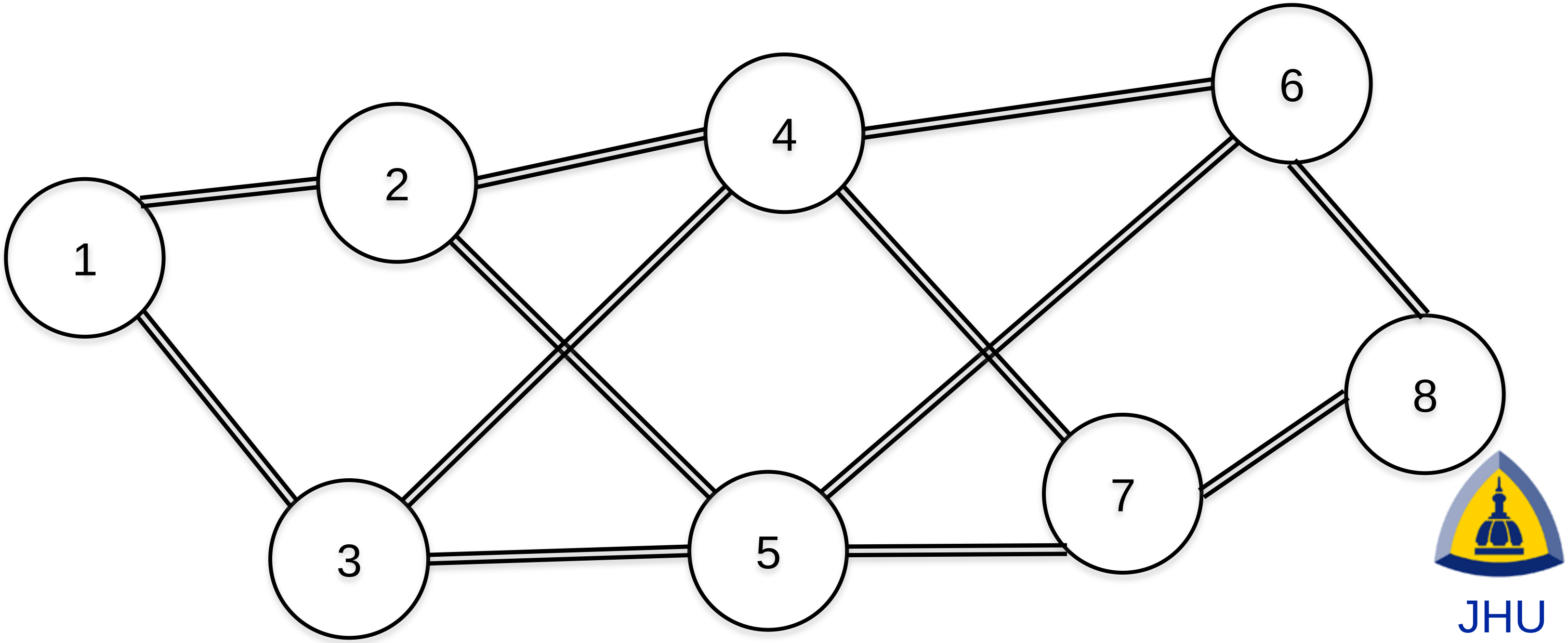
The New Spines Architecture

www.spines.org



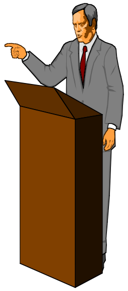
Cloud Validation

U.S.-Wide Topology

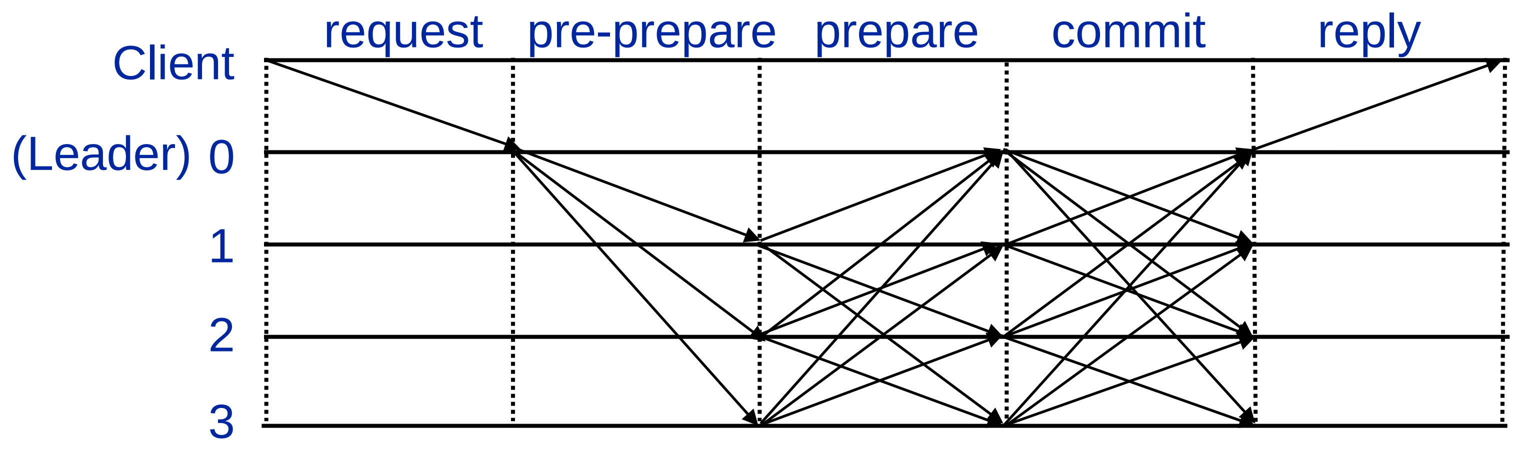


Outline

- Project goal
- Intrusion-tolerant messaging (**Spines**)
  - Flooding and K-Path Routing disseminations
  - Monitoring: Priority-based dissemination
  - Control: Reliable dissemination
- Intrusion-tolerant Replication (**Prime**)
  - Proactive recovery: defense across space and time
  - Theoretical model: resiliency through proactive recovery
  - Physical and virtual approaches
  - Support for 1 Terabyte application state size

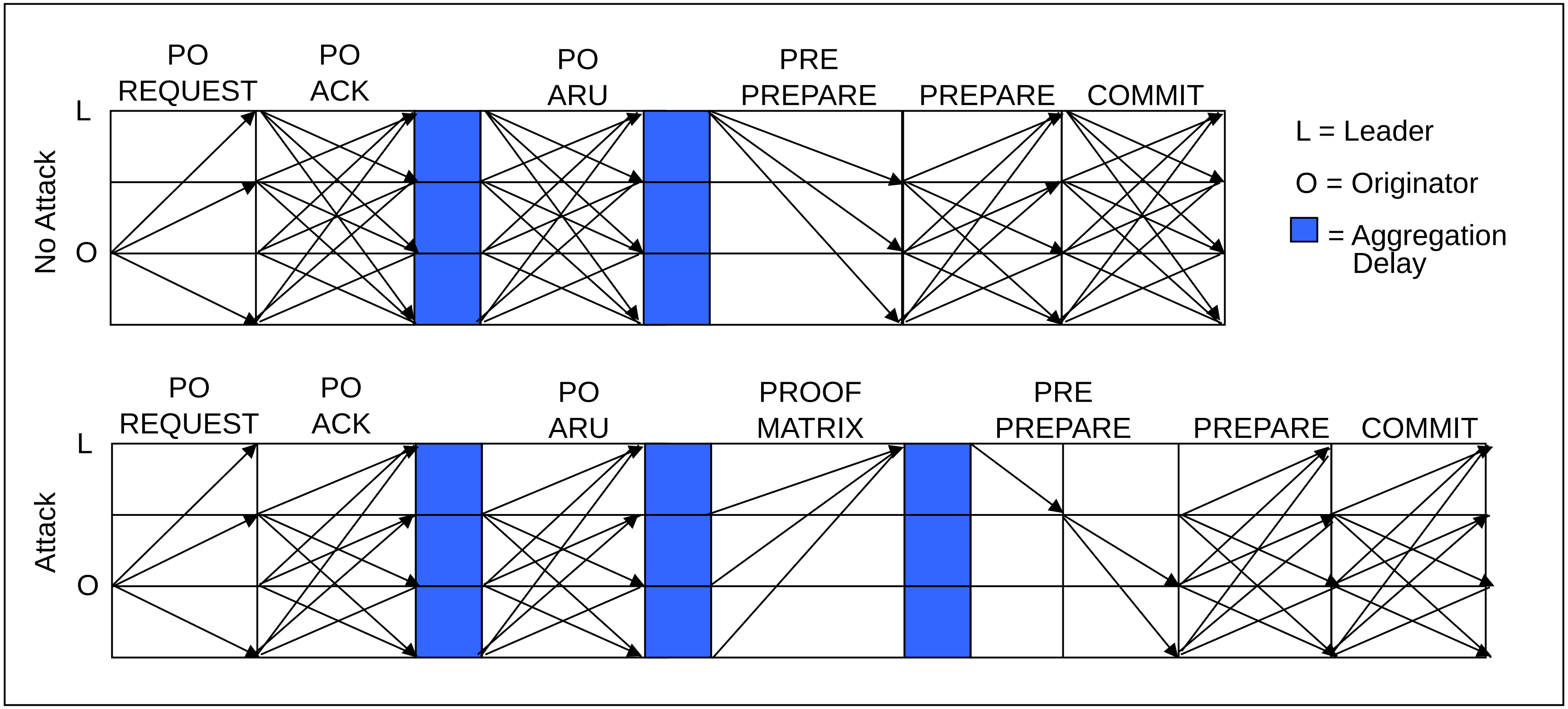


Byzantine Replication (BFT)



- State machine replication sustaining *f* out of 3*f*+1 compromised replicas with the following guarantees:
- Safety: all correct replicas maintain consistent state
- Liveness: eventual progress
- Outcome: good performance under "normal" conditions
- Problem: no performance guarantees while under attack

Prime: Byzantine Replication with Performance Guarantees Under Attack



- Limiting the power of a malicious leader
  - Bounded-delay performance guarantee
- Integrated by Siemens to their SCADA product for the power grid

Defense across space and time

- Problem: Prime (and BFT in general) is fragile over a long system lifetime
- Solution:
  - Space: diversify the execution environment as much as possible to generate different versions of the same application
  - Time: periodic and proactive replica rejuvenation to clean potentially undetected intrusions
  - Diversity + Proactive Recovery = building blocks for the construction of long-lived intrusion-tolerant systems

Novelty Claims

- Theoretical model that computes how resilient the system is over its lifetime
- Support for applications with large state (e.g. 1 terabyte)
- First construction of subsystems that support the assumption of a practical survivable data replication system:
  - Prime – providing guaranteed performance while under attack
  - MultiCompiler – compiler-based fine-grained diversity providing protection across space
  - Proactive Recovery – providing protection across time

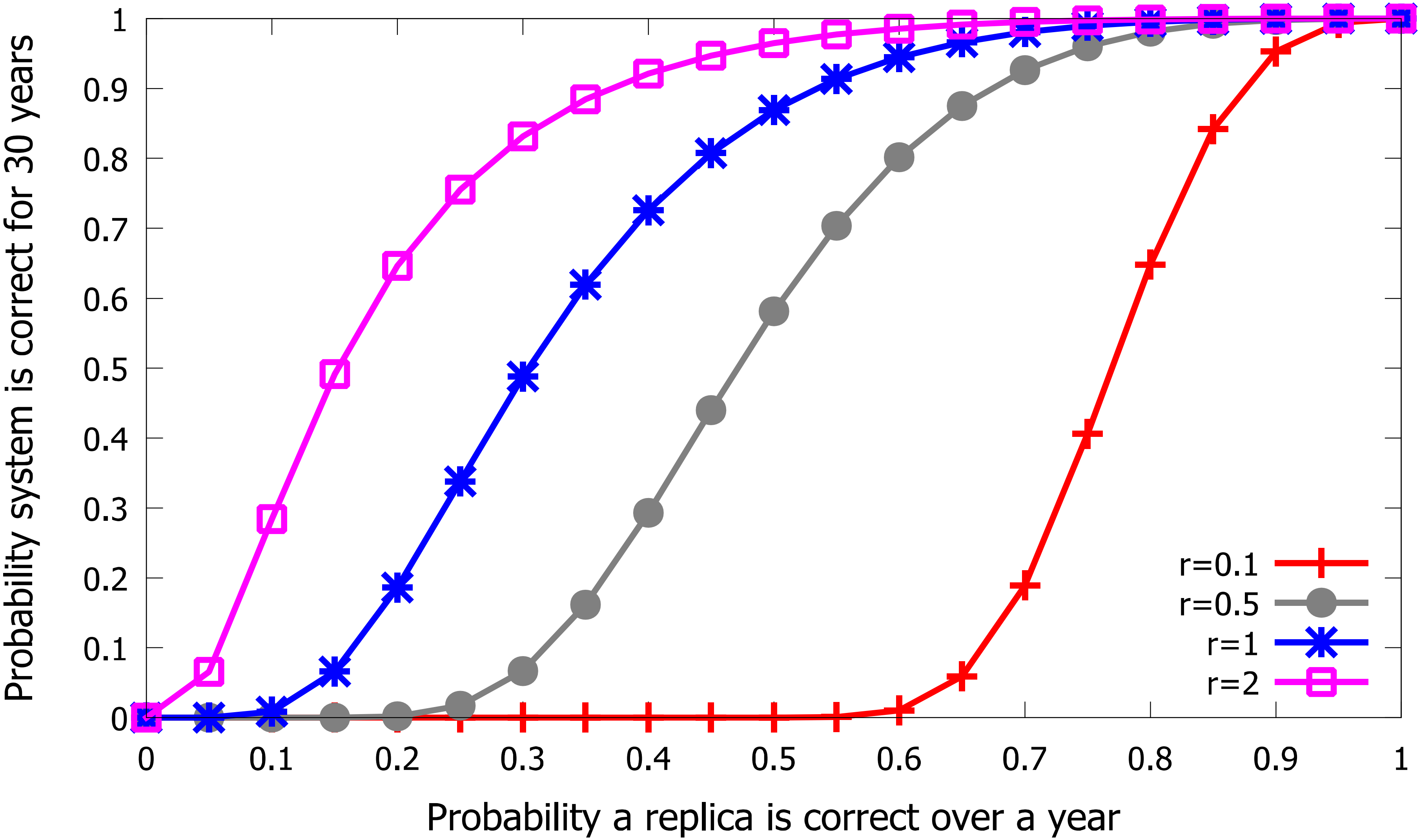
Proactive Recovery Operation Sequence

- Replica rejuvenation
  - The replica restarts periodically from a fresh copy of OS and application code from read-only memory
  - Use of fine-grained diversity
- Session key replacement
  - If the replica was malicious, its private key can be used to forge messages
  - Session key is based on unforgeable cryptographic material, e.g., Trusted Platform Module (TPM)
- State validation
- State transfer if needed

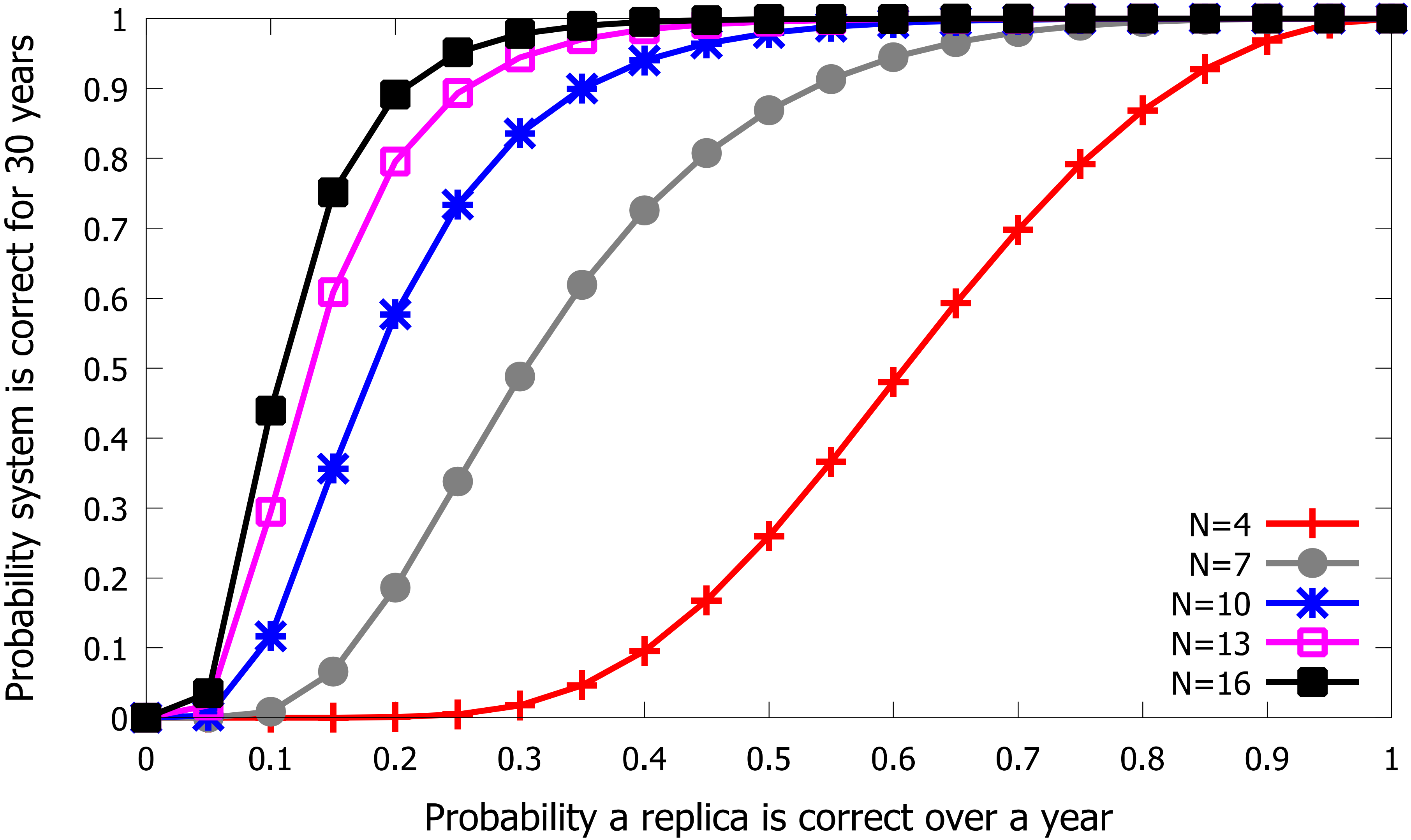
Theoretical model: resiliency through proactive recovery

- A model to compute how resilient the system is over its lifetime (e.g. for 30 years)
- Assumptions
  - No more than f simultaneous failures
  - Replicas get compromised independently
- Input parameters
  - Number of replicas: 3f+1
  - Strength of a replica (i.e. probability that a replica remains correct over a year)
  - Rejuvenation rate (i.e. number of rejuvenations per day)
  - System lifetime
- Output
  - Probability the system remains correct over its lifetime

Varying the Rejuvenation Rate
(7 replicas system – can sustain 2 bad guys)

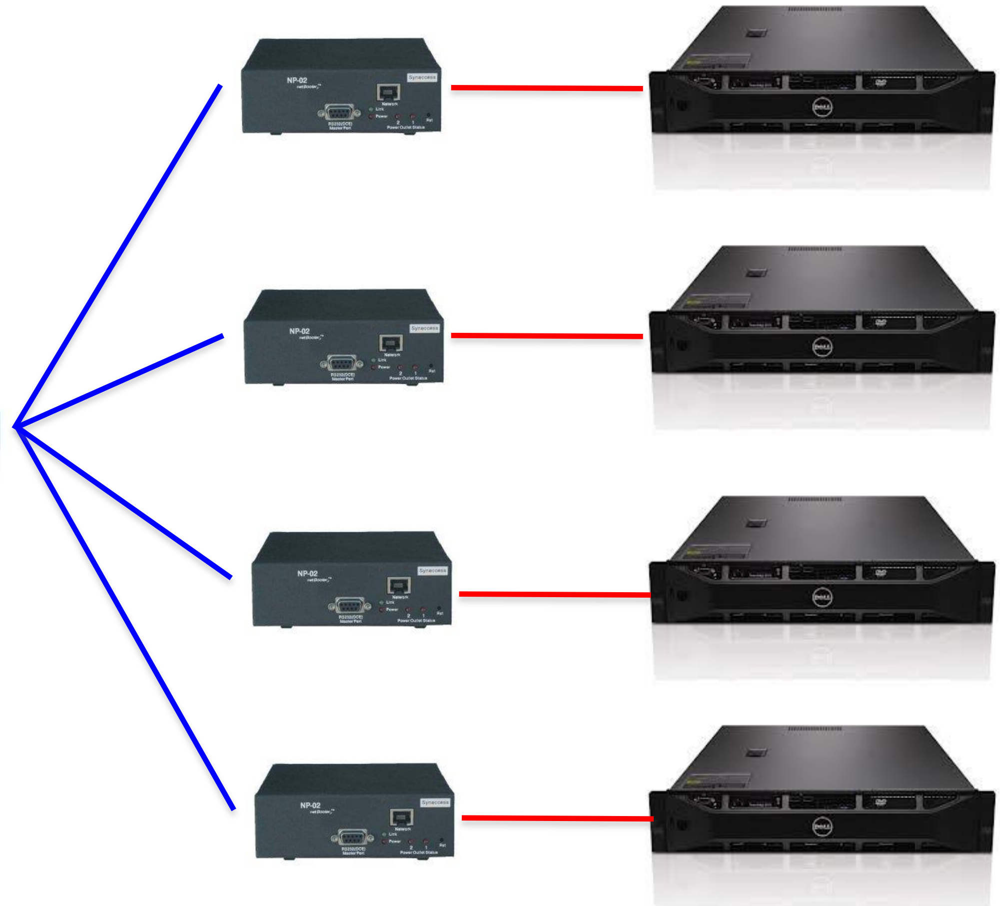


Varying the Number of Replicas
(1 rejuvenation of a single replica per day)



A Physical System Approach

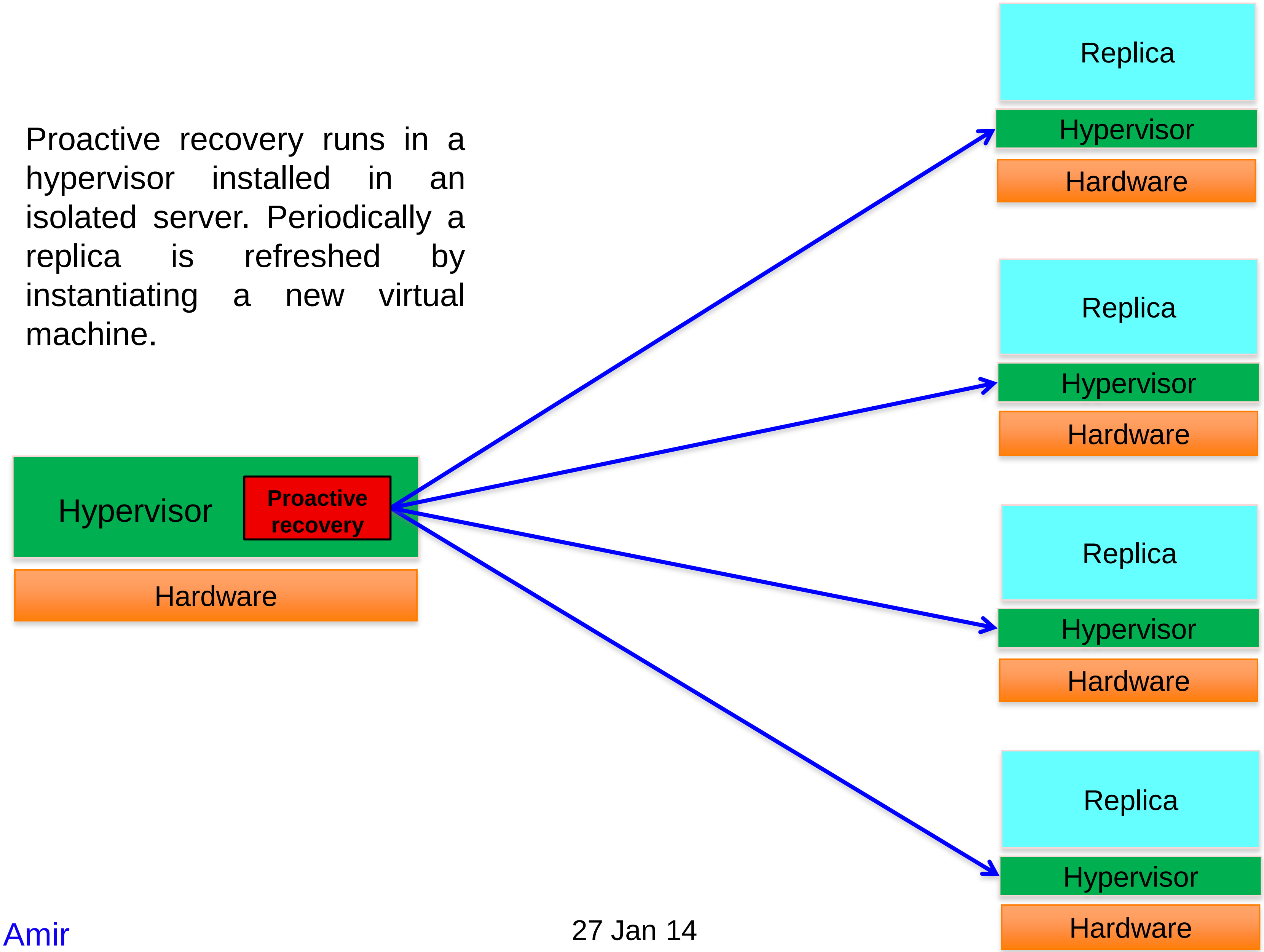
Proactive recovery logic runs in an isolated Next Unit of Computing (NUC). Servers that host Prime replicas are plugged into remote power switches. A network switch connects the NUC to remote power switches.



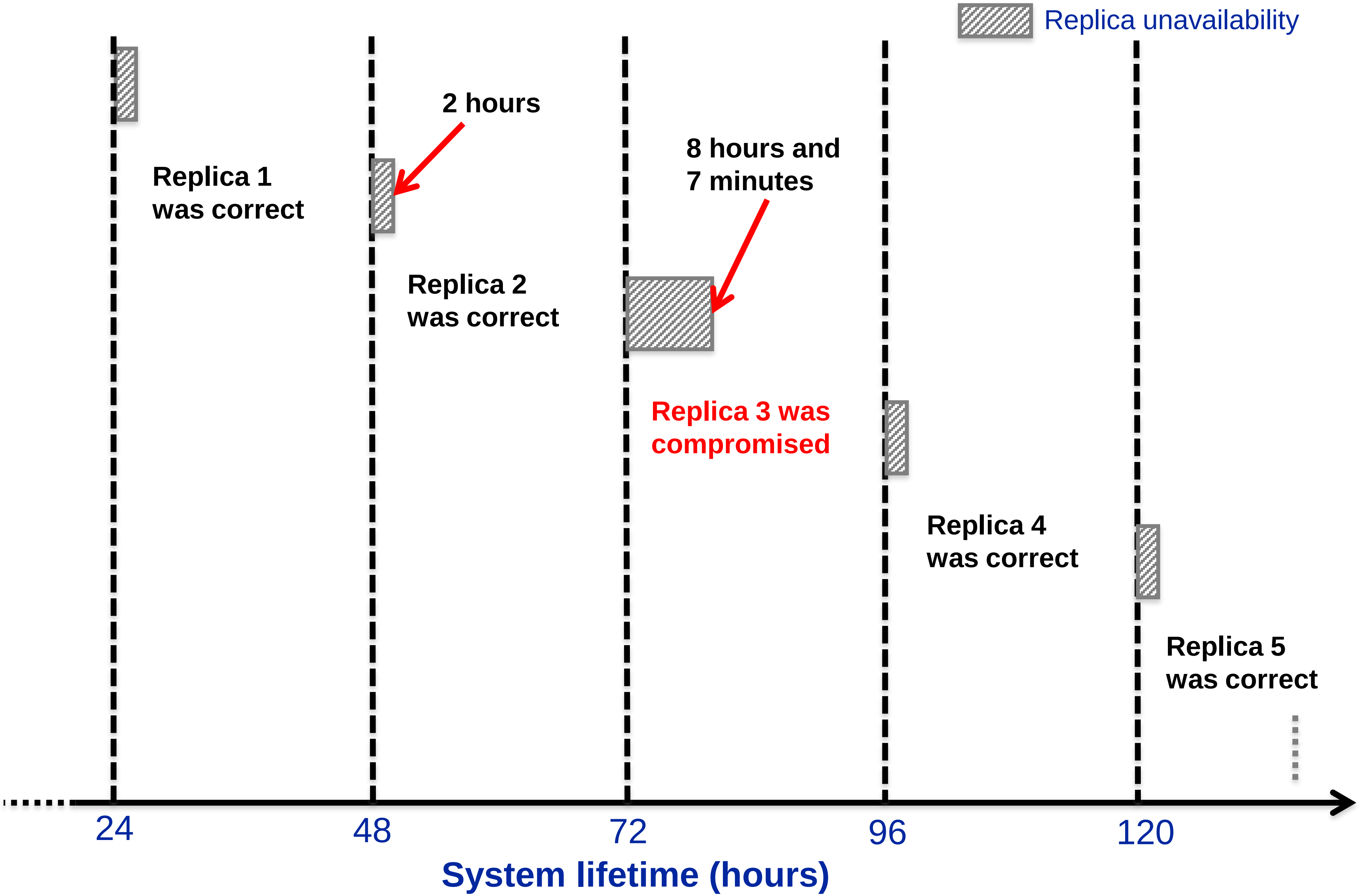
Periodically, the NUC activates a remote power switch, which cycles the power to restart the server that hosts a Prime replica, rebooting a fresh copy from a read-only device

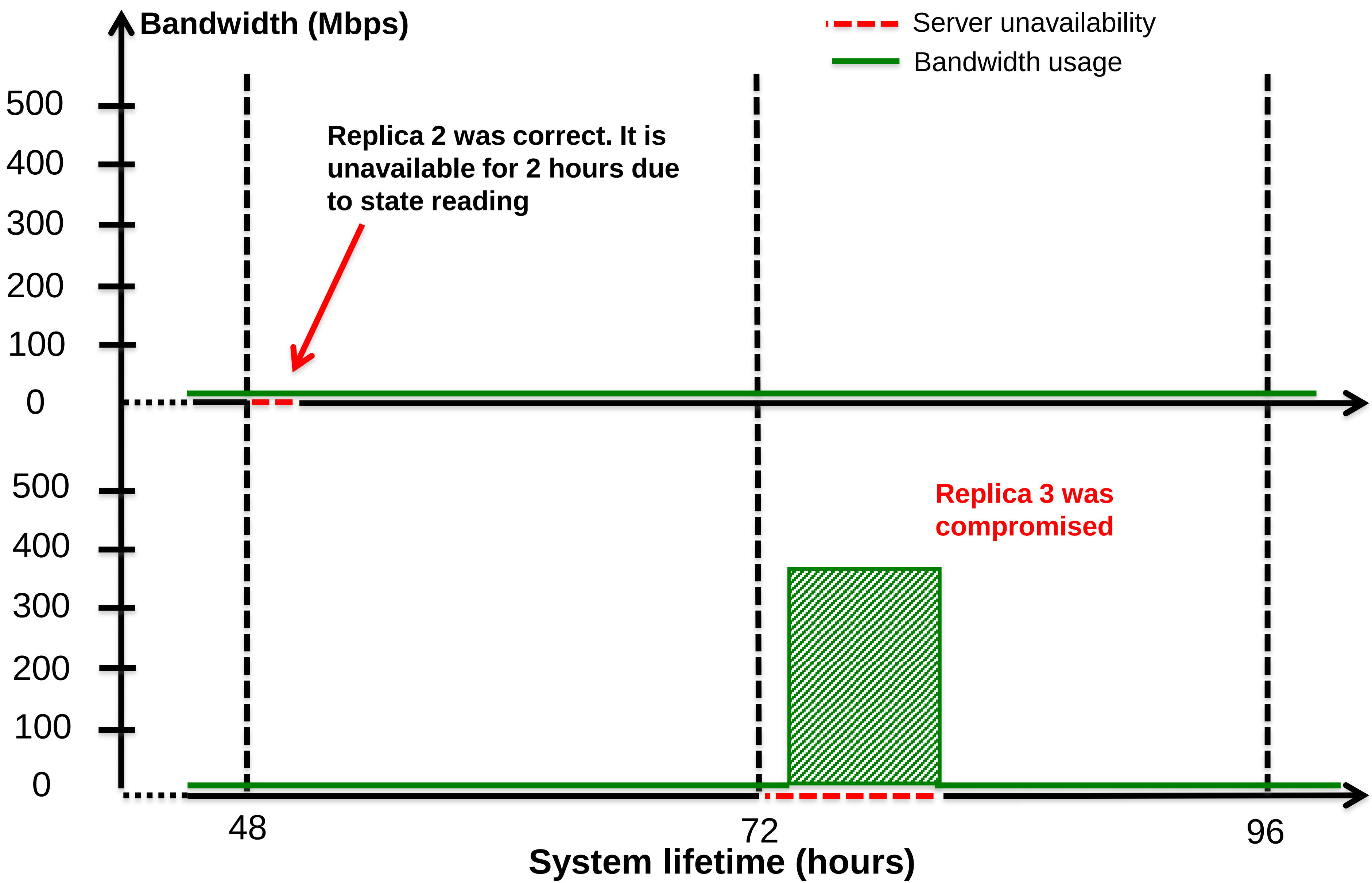
A Virtualized System Approach

Proactive recovery runs in a hypervisor installed in an isolated server. Periodically a replica is refreshed by instantiating a new virtual machine.



Measurements for 1 Terabyte State Transfer





Bandwidth (Mbps)
Server unavailability
Bandwidth usage
500
400
300
200
100
0
Replica 2 was correct. It is unavailable for 2 hours due to state reading
500
400
300
200
100
0
Replica 3 was compromised
48
72
96
System lifetime (hours)

Outline

- Project goal
- Intrusion-tolerant messaging (**Spines**)
  - Flooding and K-Path Routing disseminations
  - Monitoring: Priority-based dissemination
  - Control: Reliable dissemination
- Intrusion-tolerant Replication (**Prime**)
  - Proactive recovery: defense across space and time
  - Theoretical model: resiliency through proactive recovery
  - Physical and virtual approaches
  - Support for 1 Terabyte application state size