An On-demand Secure Routing Protocol Resilient to Byzantine Failures

Baruch Awerbuch
Johns Hopkins University

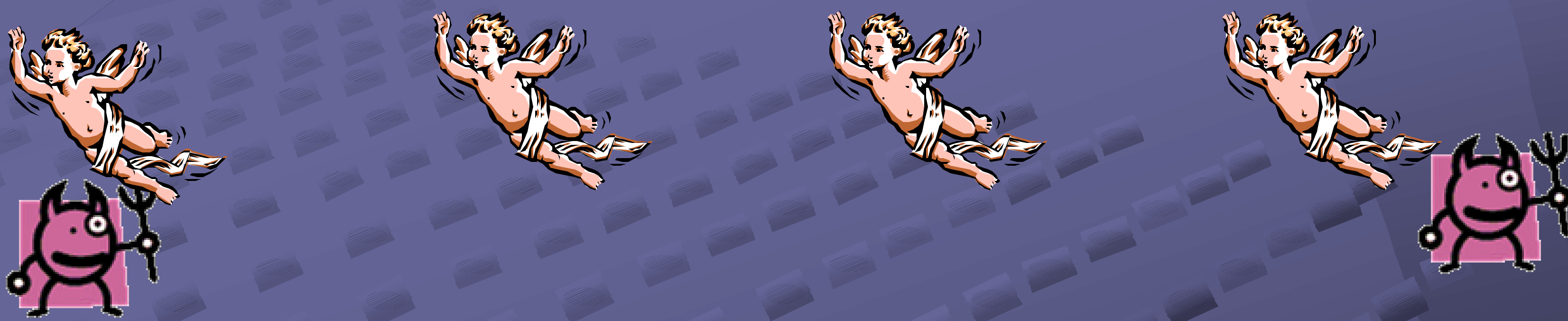
Joint work with David Holmer, Cristina Nita-Rotaru, and Herbert Rubens

Based on paper at WiSe2002

On-Demand vs. Proactive Routing Security Concerns

- On-Demand
  - Source Authentication
  - Caching presents adversarial opportunity
- Pro-active
  - Harder to secure since pieces of information can not be traced back to a single source.

Communication Vulnerabilities



Eavesdropping &
Impersonation

Encrypt Data
Authenticate Users

Denial of Service (DOS)

Monitor traffic
Localize damage

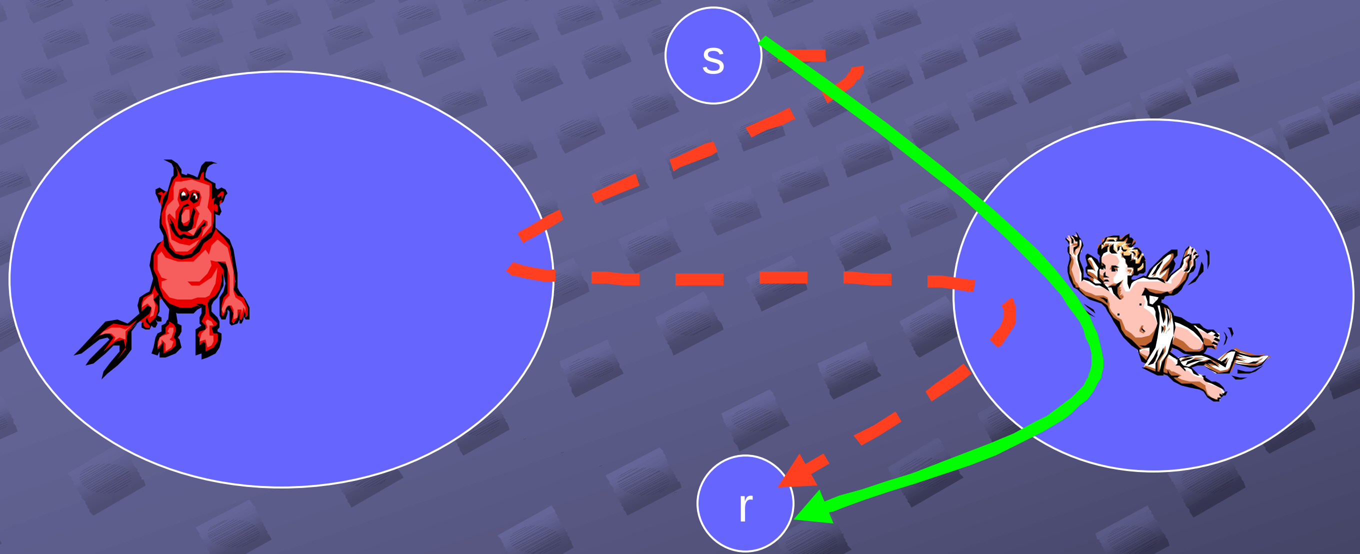
Routing:

(Hard Problem)

Sweep under rug

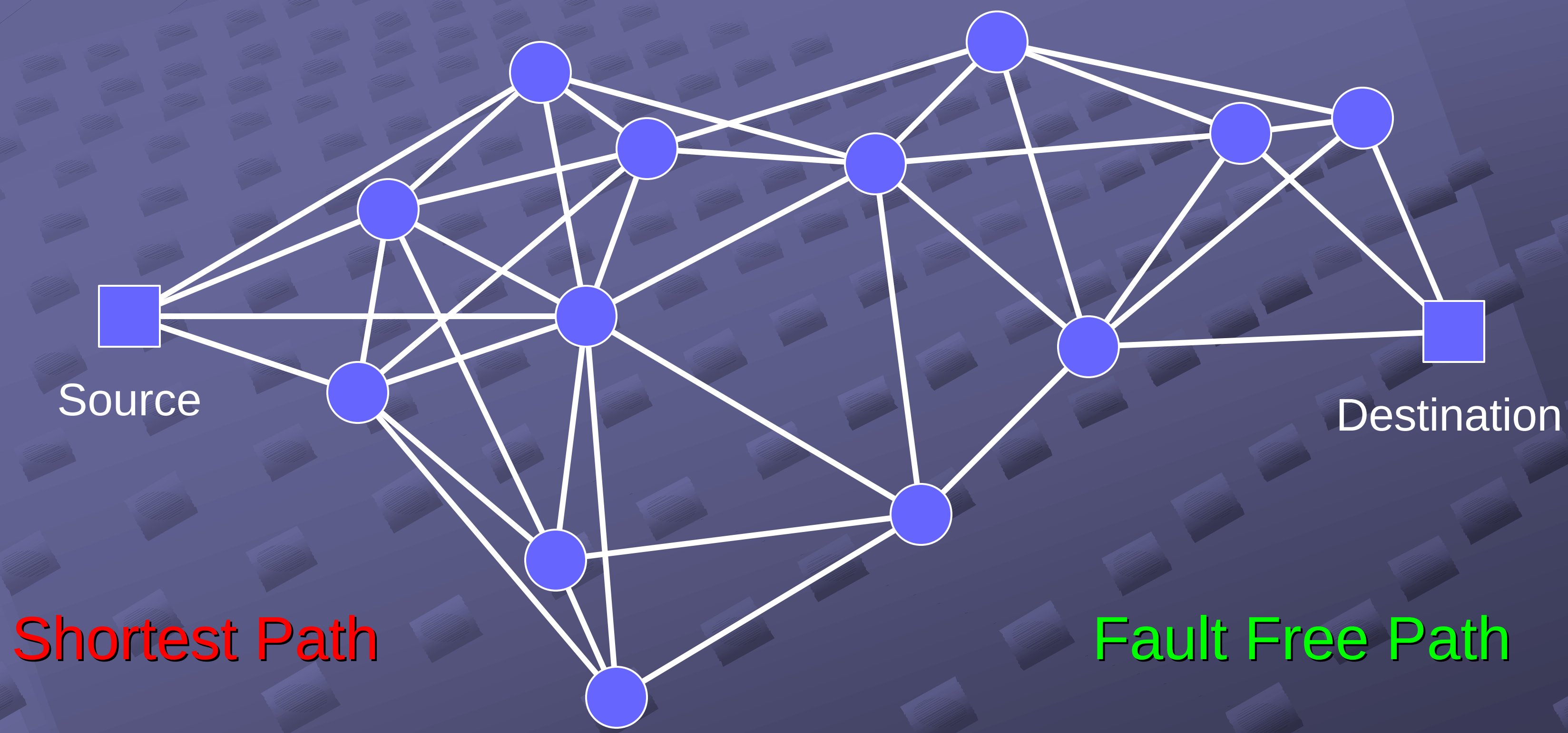
This talk's focus

Routing: objective



If there is a fault- free path from source to receiver:
- communication should proceed undisturbed
- consumes minimal resources in the reliable component

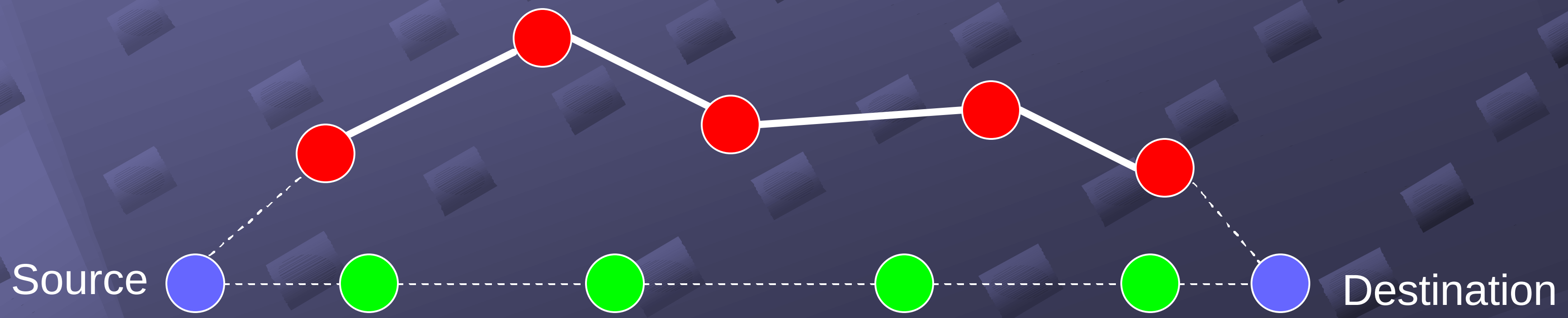
Problem Description



Trusted Node   Correct Node   Adversarial Node

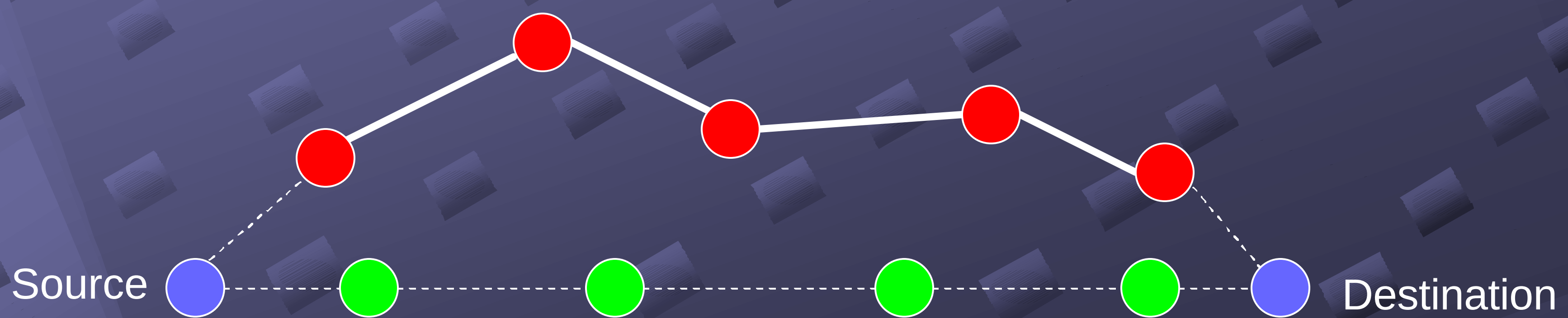
Worm Holes

- Two attackers establish a path and tunnel packets from one to the other
- The worm hole turns many adversarial hops into one virtual hop creating shortcuts in the network
- This allows a group of adversaries to easily draw packets into a black hole



Black hole attack

- Packets are simply dropped
- Adversaries can move thru the network
- Aggravated by wormhole attack



Related Work

- Terminodes
  - [Hubaux, Buttyan, Capkun 2001]
- Cornell
  - [Zhou, Haas 1999]
  - [Papadimitratos, Haas 2002]
- Watchdog
  - [Marti, Giuli, Lai, Baker 2000]
- Wormhole Detection, SEAD, Ariadne
  - [Hu, Perrig, Johnson 2002]
- University of Massachusetts
  - [Dahill, Levine, Shields, Royer 2002]

This talk: Unlimited # faults model

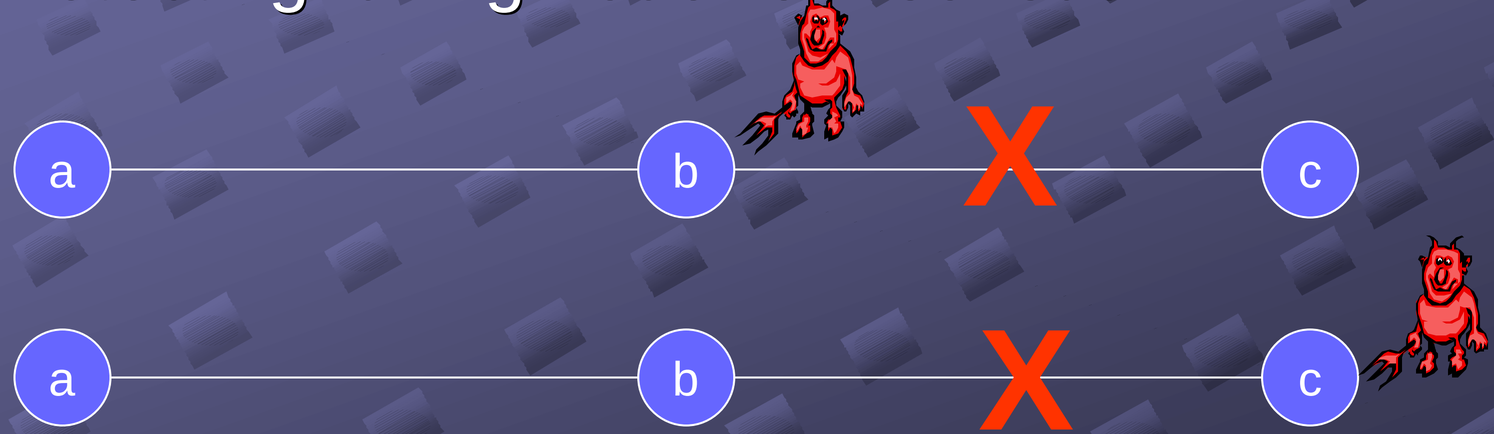
- Trust model
  - Source and Destination are trusted
  - Intermediate nodes are authenticated but not trusted
- Adversarial model
  - Majority of colluding byzantine adversaries
  - Focus on containment (not defeating) adversaries

Black Hole Attack

Problem: Adversary may delete a packet

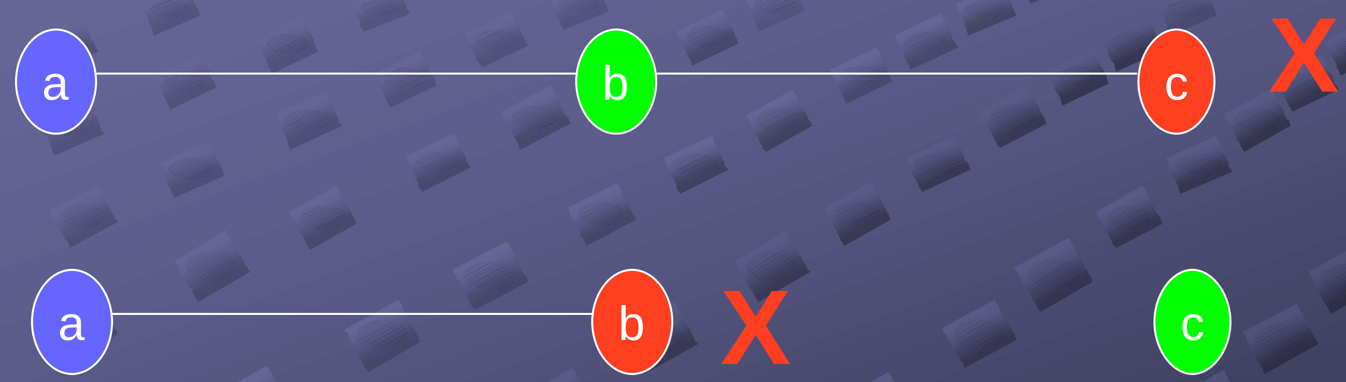
How do we detect and avoid black holes ?

- Reliable node may be blamed
- Detecting failing node: Consensus ?



Impossibility of detection

- Can't tell who is the adversary

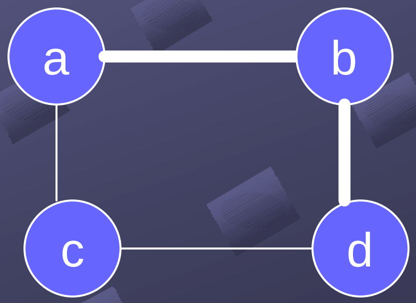


- This talk:

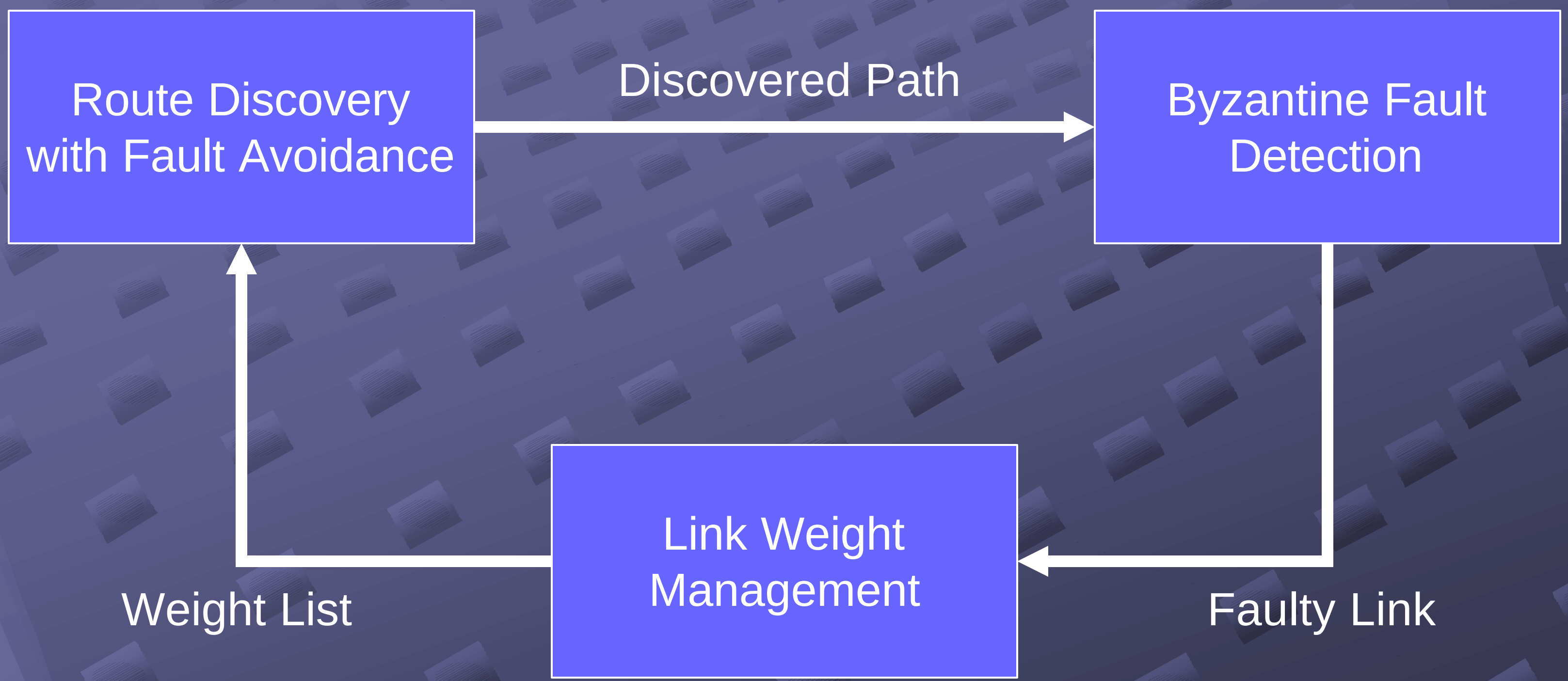
avoid both endpoints of contentious link

This Talk: link reputation system

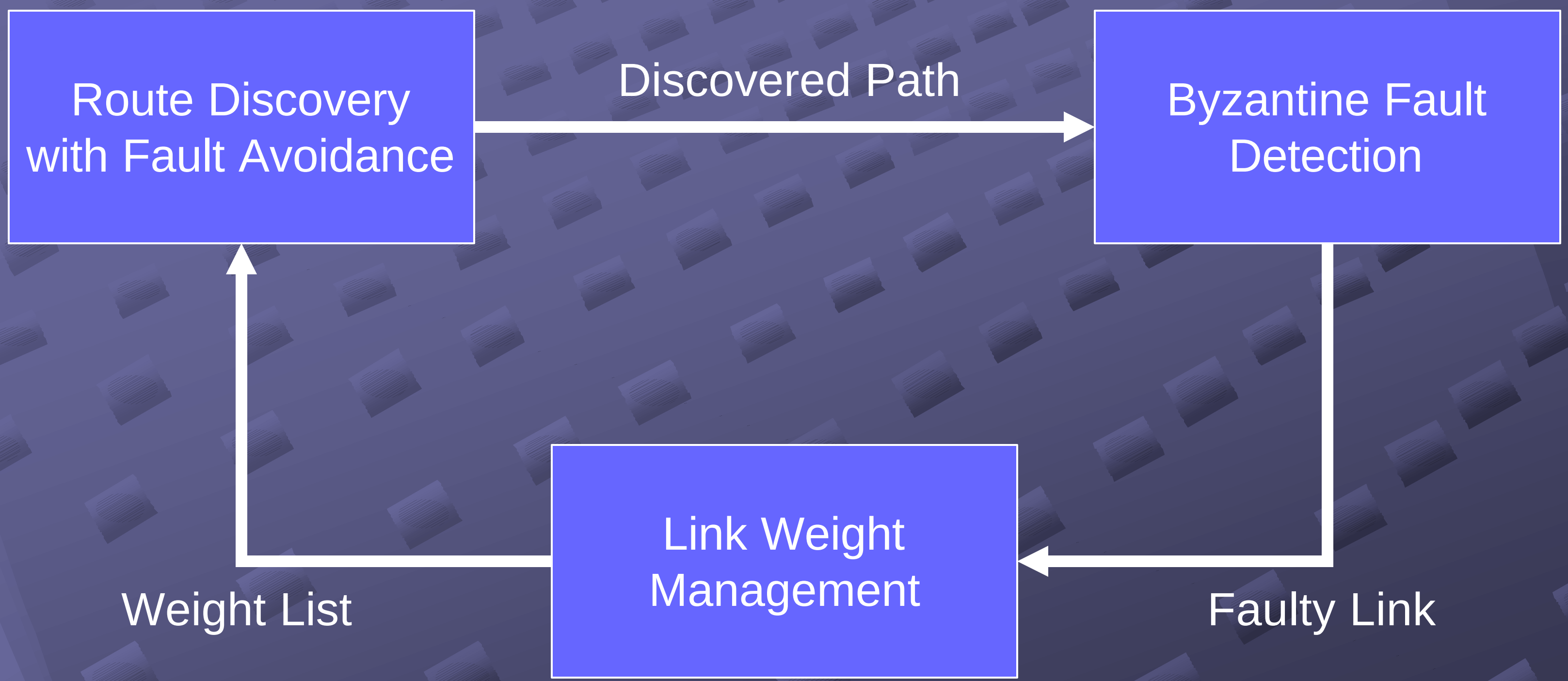
- Link Weight : reflection of performance statistics (doubled for each fault)
- Shortest paths w.r.t. link weights avoid faulty area



Protocol Overview

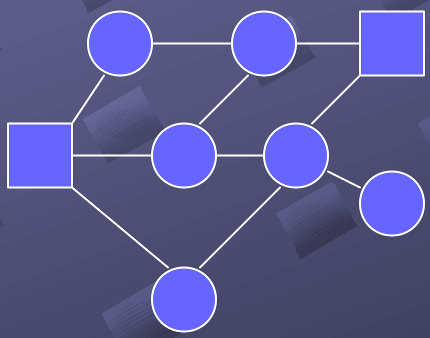


Route Discovery Phase

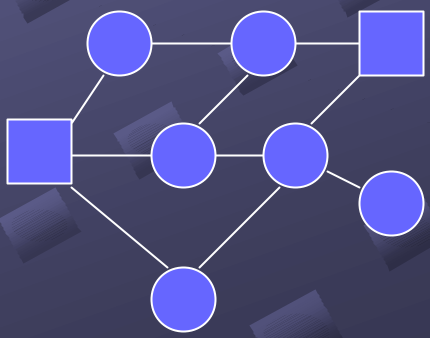


Route Discovery

- On-demand protocol
  - Finds a least weight path
- Request flood
  - Request includes weight list and signature
  - Signature verified at every hop
  - Prevents un-authorized route requests



Request



Response

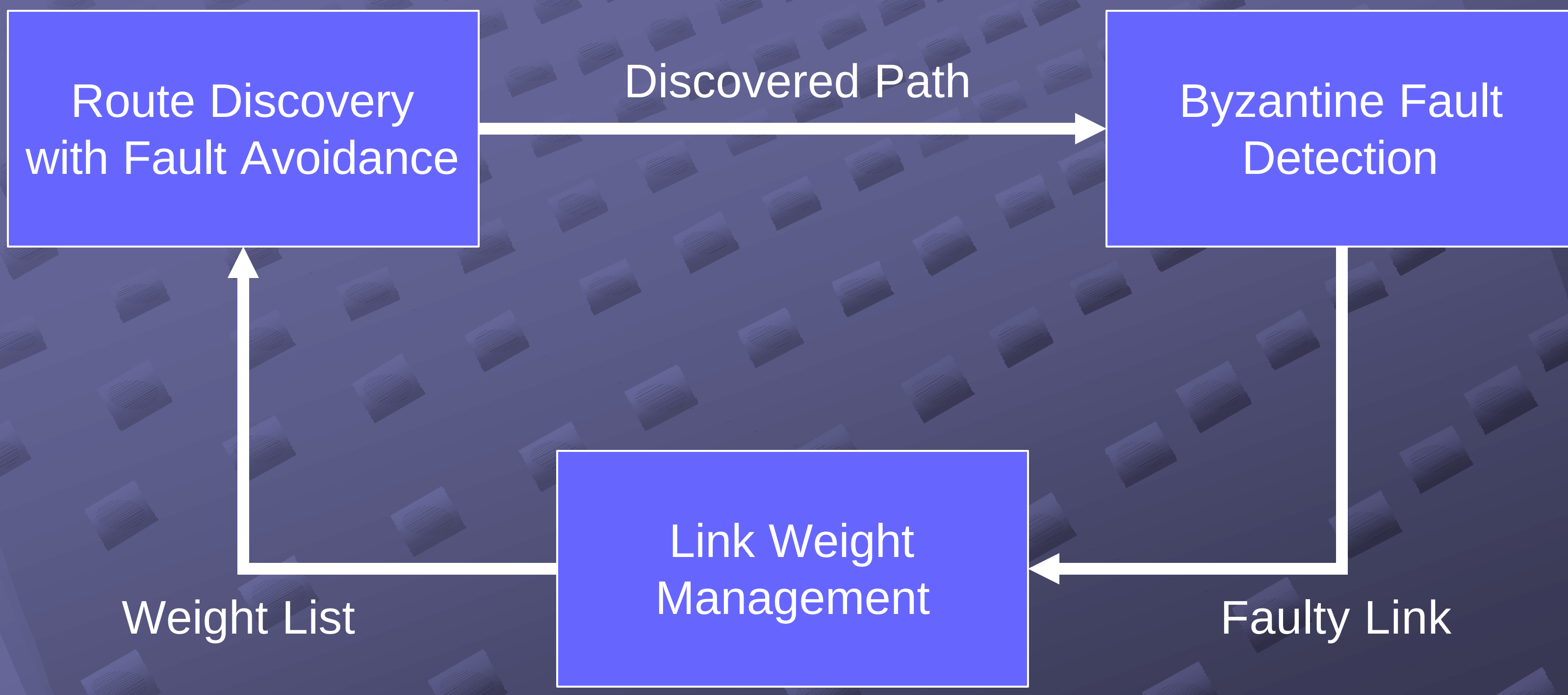
Flood Blocking

- Flood Blocking Attack
  - Adversary propagates a false short path
  - Intermediate nodes do not forward “inferior” valid path information
  - Source ignores the false path
  - No path is established
- Path must be verified at intermediate nodes

Route Discovery (cont.)

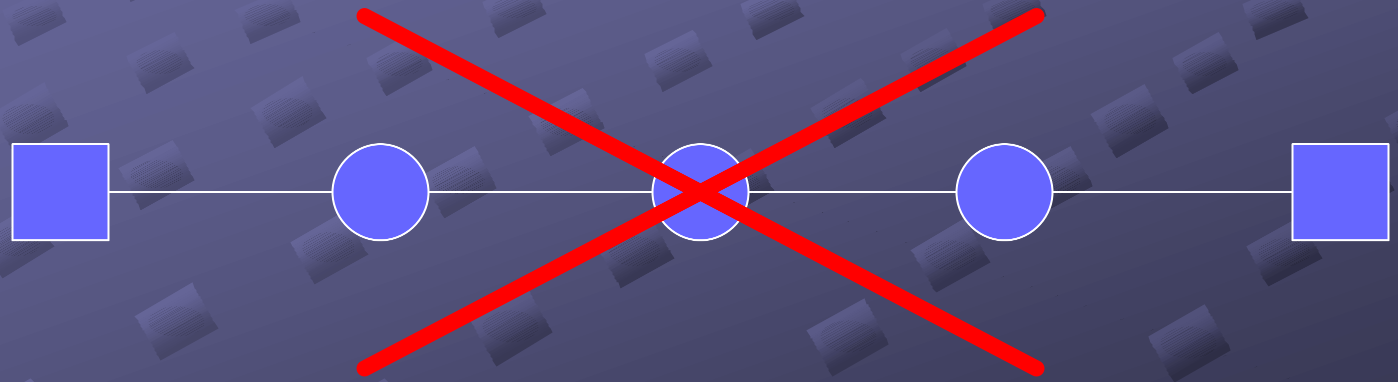
- Response flood
  - Prevents response block attack
  - Path and weight accumulated hop by hop
    - Appends signature to response
  - Only lower cost updates are re-broadcast
  - Every hops verifies the entire path
  - Prevents flood blocking attack
- Path is not guaranteed to be fault free
- Some path is always established

Fault Detection Phase



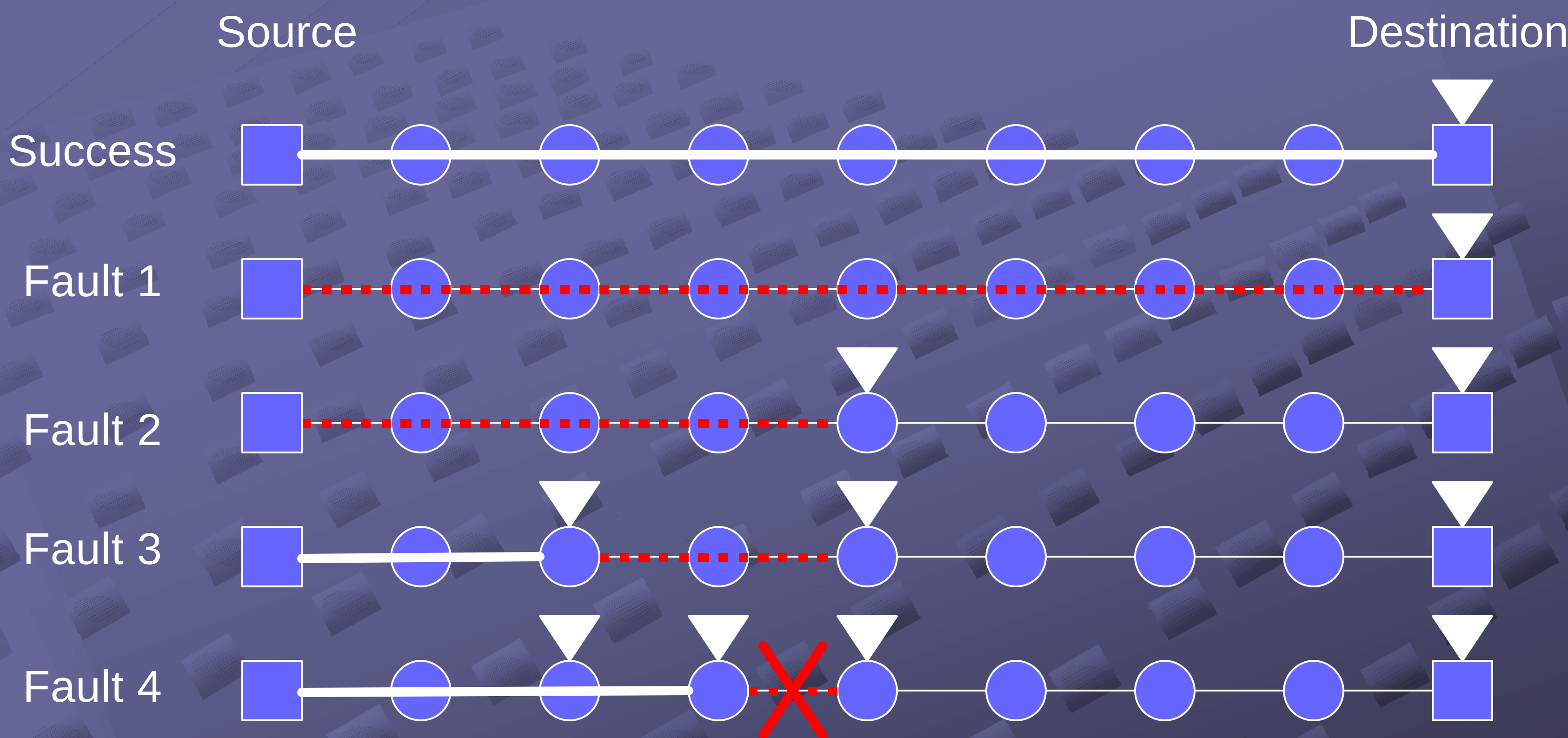
Fault Detection Strategy

- Probing technique using authenticated acknowledgements
- Naïve probing technique

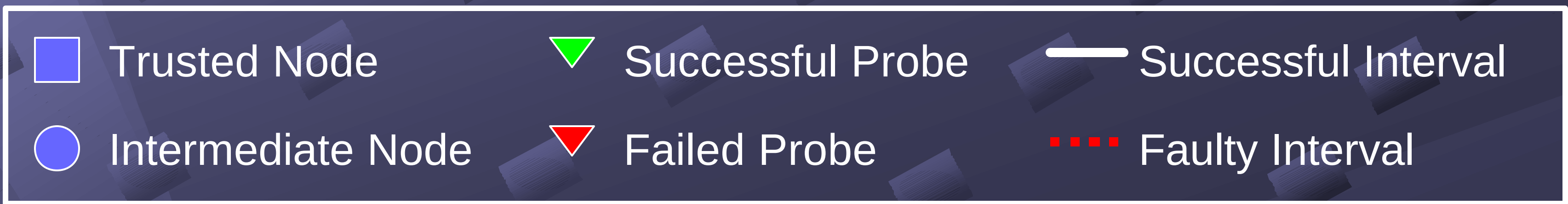


- Too much overhead per data packet!

Secure Adaptive Probing



Binary search = identified in log n faults



Probe & Ack Properties

- Probes
  - Inseparable from data - listed on all packets
  - Integrity checked at each probe - HMAC
  - Enforces path order - onion encrypted list
- Acks
  - Authenticated - HMAC
  - Single combined ack packet - individual acks added at each probe point & onion encrypted
    - Adversary can’t drop selective acks
  - Staggered timeouts - restarts ack packet
- A node can’t incriminate any link but its own

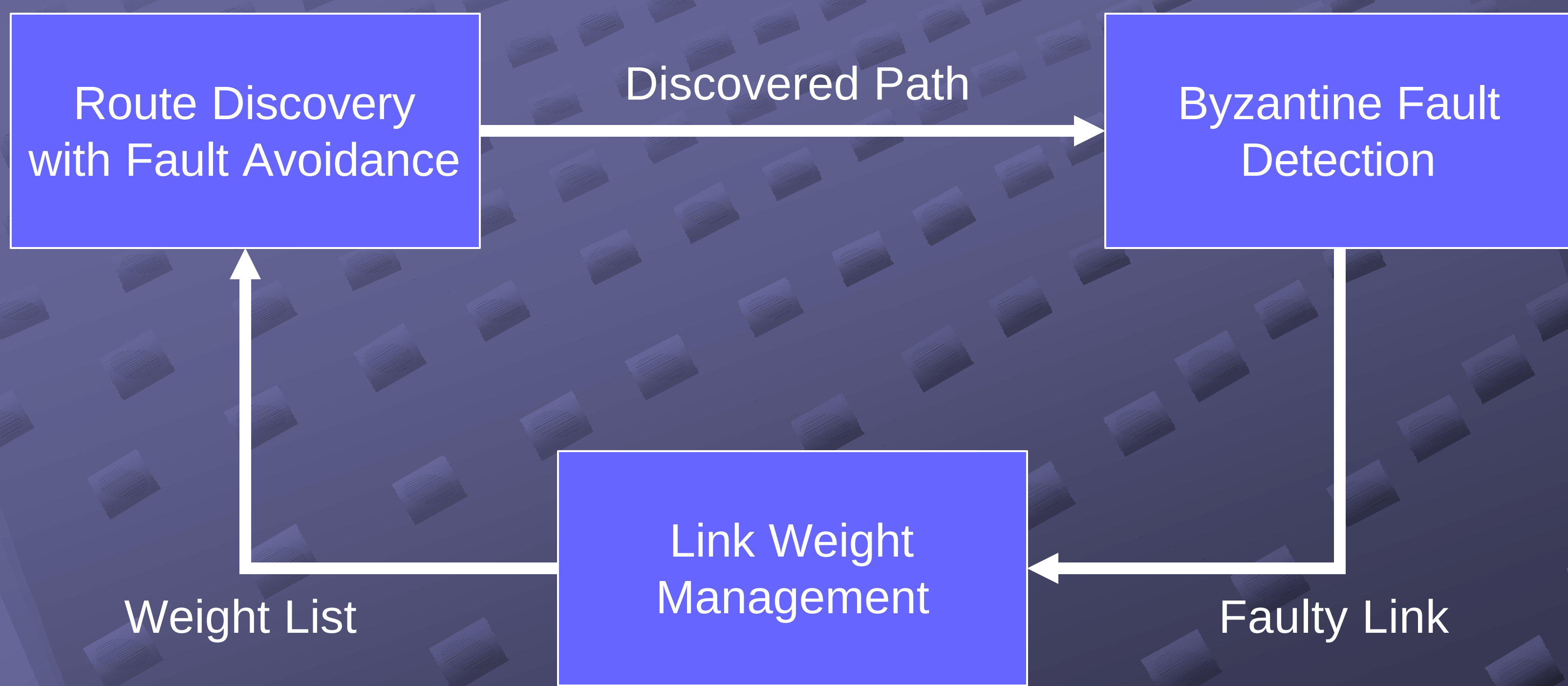
Probe & Ack Specification

- Probes
  - List of probes attached to every packet
  - Each probe is specified by an HMAC
  - Probes listed in path order
  - Remainder of probe list is onion encrypted
- Ack
  - Authentication via HMAC
  - Collected and onion encrypted at each probe point

Fault Identification

- Fault Definition
  - Packet loss rate violates a fixed threshold
  - Excessive delay also causes packet loss
- Identifies faulty links regardless of reason
  - Malicious behavior
  - Adverse network behavior
    - Congestion
    - Intermittent connectivity

Link Weight Management Phase



Link Weight Management

- Maintains a weight list of identified links
- Faulty links have their weight doubled
- Resets link weights
  - Timed by successful transmissions
  - Bounds average loss rate
- Network is never partitioned

Analysis

- Network of n nodes of which k are adversaries
- Assume a fault free path exists

$$\boxed{q^{-} - r \cdot q^{+} \le b \cdot kn \cdot \log^{2} n}$$

- Protocol bounds the number of packets lost communicating with the destination

Conclusion

- On-demand routing protocol resilient to colluding byzantine attackers
- Adaptive probing identifies a faulty link in log n faults
- Bounded long term loss rate
- Bounded total losses beyond long term rate

Future Work

- Investigate more sophisticated fault detection
  - Adaptive threshold
  - Probabilistic scheme
- Route caching
- Simulation and implementation

Questions?

- Funding by
  - Johns Hopkins Information Security Institute
  - DARPA

www.cnds.jhu.edu/archipelago/