COLLABORATIVE ATTACKS AND DEFENSE

Bharat Bhargava
CERIAS and CS department
Purdue University
www.cs.purdue.edu/homes/bb

Trusted Router and Protection Against Collaborative Attacks

- Characterizing collaborative/coordinated attacks
- Types of collaborative attacks
- Identifying Malicious activity
- Identifying Collaborative Attack

Collaborative Attacks

<u>Informal definition:</u>

"Collaborative attacks (CA) occur when more than one attacker or running process synchronize their actions to disturb a target network"

Collaborative Attacks (cont’d)

- Forms of collaborative attacks
    - Multiple attacks occur when a system is disturbed by more than one attacker
    - Attacks in quick sequences is another way to perpetrate CA by launching sequential disruptions in short intervals
    - Attacks may concentrate on a group of nodes or spread to different group of nodes just for confusing the detection/prevention system in place
    - Attacks may be long-lived or short-lived
    - Collaborative attacks can be launched intentionally or accidentally
    - Attacks on routing

Collaborative Attacks (cont’d)

- Open issues
  - Comprehensive understanding of the coordination among attacks and/or the collaboration among various attackers
  - Characterization and Modeling of CAs
  - Intrusion Detection Systems (IDS) capable of correlating CAs
  - Coordinated prevention/defense mechanisms

Collaborative Attacks (cont’d)

- From a low-level technical point of view, attacks can be categorized into:
    - Attacks that may overshadow (cover) each other
    - Attacks that may diminish the effects of others
    - Attacks that interfere with each other
    - Attacks that may expose other attacks
    - Attacks that may be launched in sequence
    - Attacks that may target different areas of the network
    - Attacks that are just below the threshold of detection but persist in large numbers

Examples of Attacks that can Collaborate

- Denial-of-Messages (DoM) attacks
- Blackhole attacks
- Wormhole attacks
- Replication attacks
- Sybil attacks
- Rushing attacks
- Malicious flooding

We are investigating the interactions among these forms of attacks

Example of probably **incompatible** attacks:

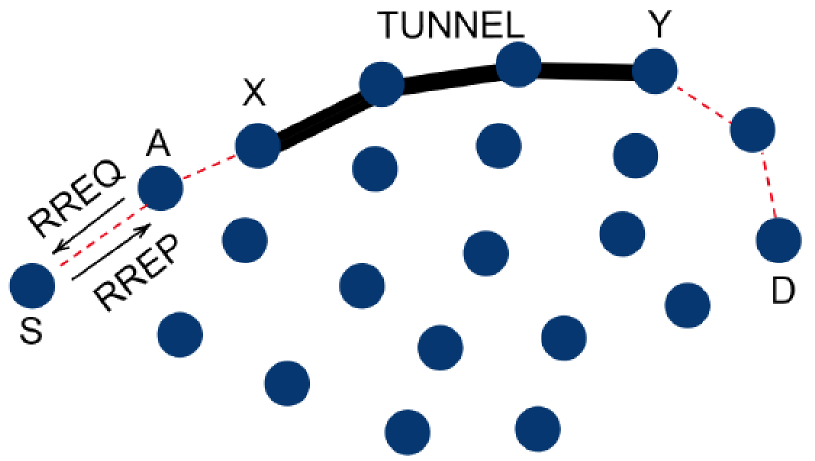
Wormhole attacks need fast connections, but **DoM** attacks reduce bandwidth!

Current Proposed Solutions

- Blackhole attack detection
    - Reverse Labeling Restriction (RLR)
- Wormhole Attacks: defense mechanism
    - E2E detector and Cell-based Open Tunnel Avoidance (COTA)
- Sybil Attack detection
    - Light-weight method based on hierarchical architecture
- Modeling Collaborative Attacks using Causal Model
- Detecting Collaboration using Machine Learning

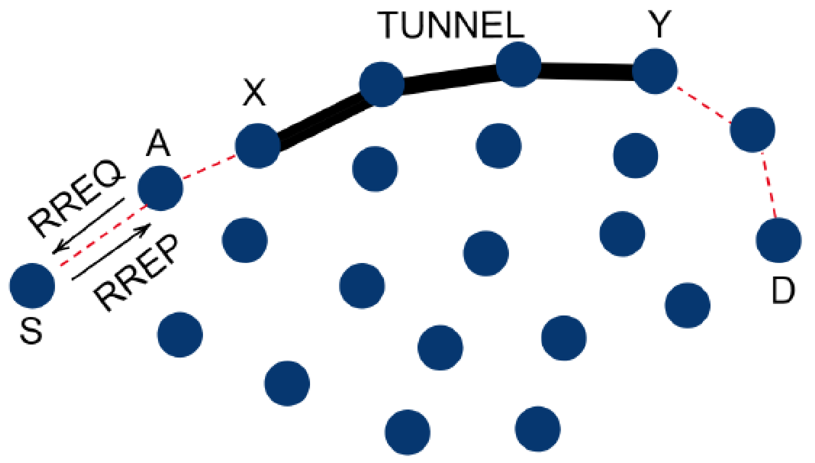
An example: blackhole attack and wormhole attack collaboration

- The attacker aims to attract as many packets as possible
  - to extract information about the system by packet inspection
  - or, to selectively drop the packets
- The goal is achieved by blackhole attack - wormhole attack collaboration

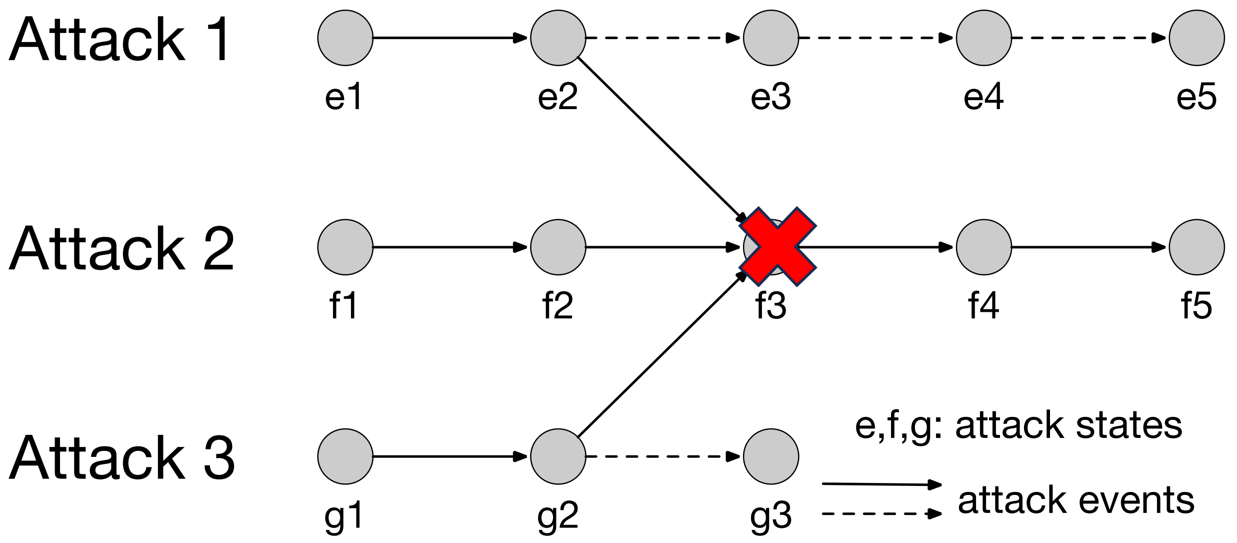


An example: blackhole attack and wormhole attack collaboration (cont'd)

- A is a blackhole attacker that attracts packets by sending fake RREP.
- X and Y are two ends of a wormhole that attract packets by advertising the one-hop route between them, namely, the wormhole.
- If A forward packets it attracted to X instead of dropping them like a normal blackhole, X will have more packets collected compared with not launching an attack or launching a wormhole attack only. The attack goal is achieved.



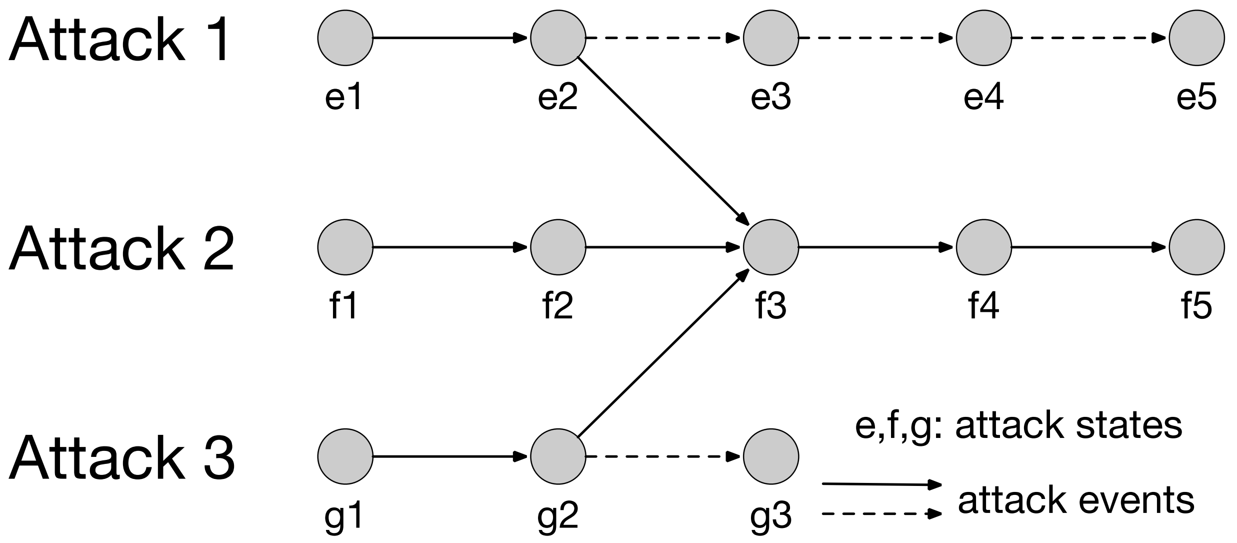
Modeling collaborative attacks with causal graph model



A collaborative attack can be modeled as a causal graph model <*S*, *E*, *M*, *L*>, where

- *S* is the set of attack states
- *E* is the set of events triggering state transition, which can be further defined by
  - Message exchange between attackers where messages are from set *E*, and
  - Local attack operations from operation set *L*
- The goal is to identify the malicious event sequences of collaborative attacks and stop it from reaching the key state (f3 in the above example)

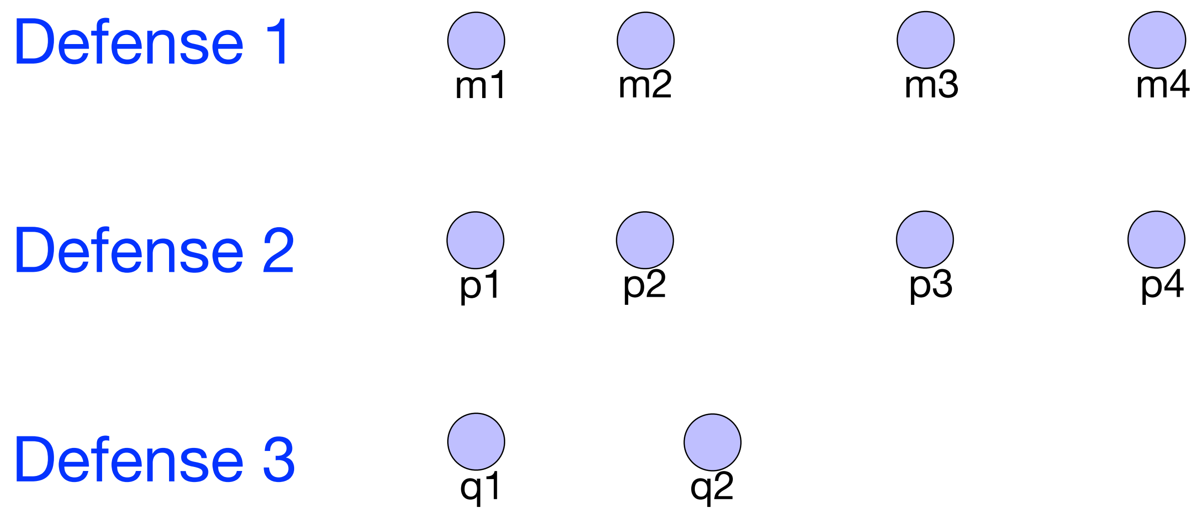
Modeling collaborative attacks with causal graph model



To prevent the attack model from reaching f3, the defender should collaborate to

1. stop Attacks 1, 2 and 3 from reaching e2, f2 and g2, respectively, or
2. stop the communication between attackers.

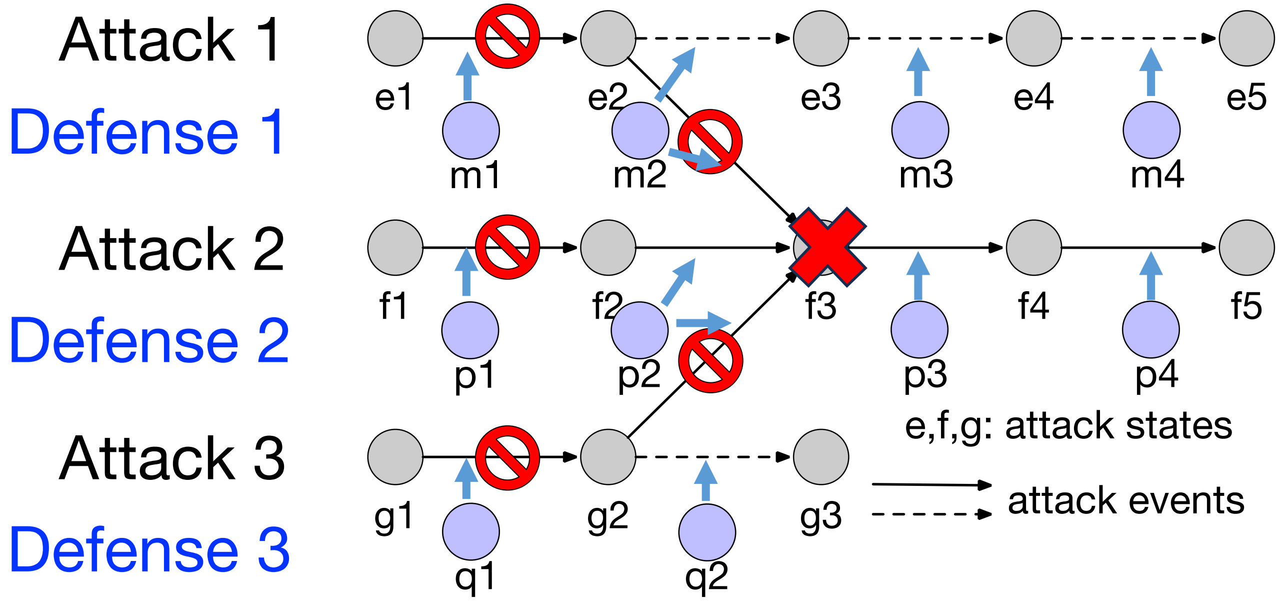
A defense strategy consists of multiple defensive events



A defense strategy consists of a set of defensive events, in the above example

- Defense 1 = {m_1, m_2, m_3, m_4}
- Defense 2 = {p_1, p_2, p_3, p_4}
- Defense 3 = {q_1, q_2}

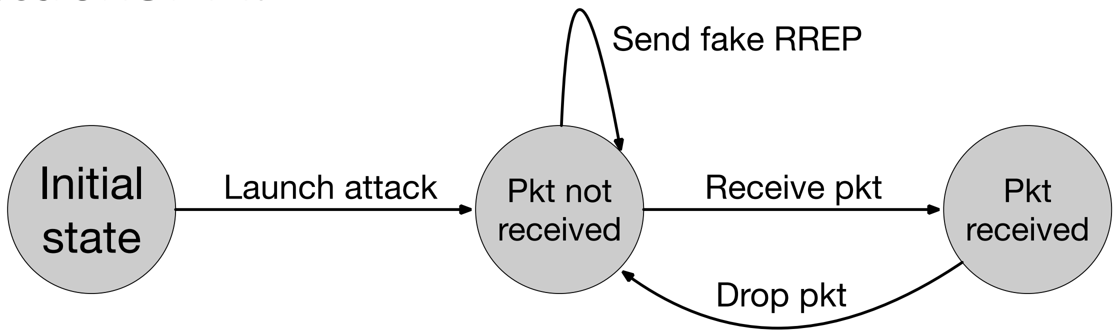
Defensive events collaborating to interfere with Attack events



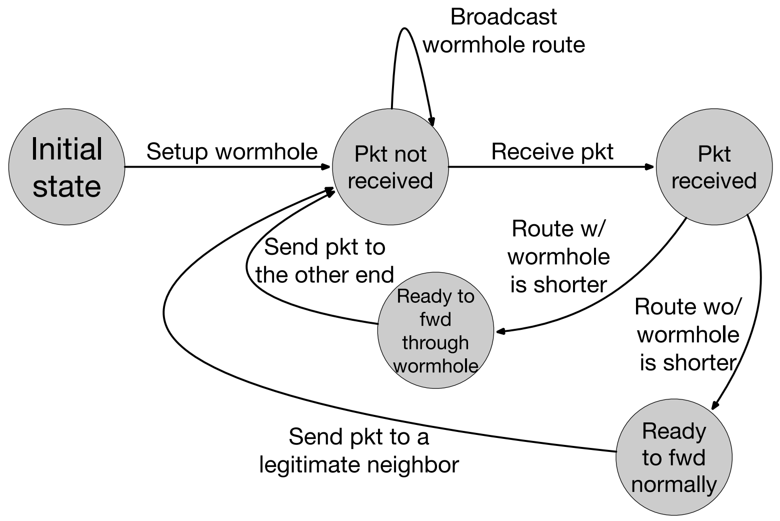
The defensive events collaborate to interfere with attack events or inter-attack communications to stop collaborative attack state transitions.

Graphs for blockhole attacks and wormhole attacks

- For blackhole attacker A:



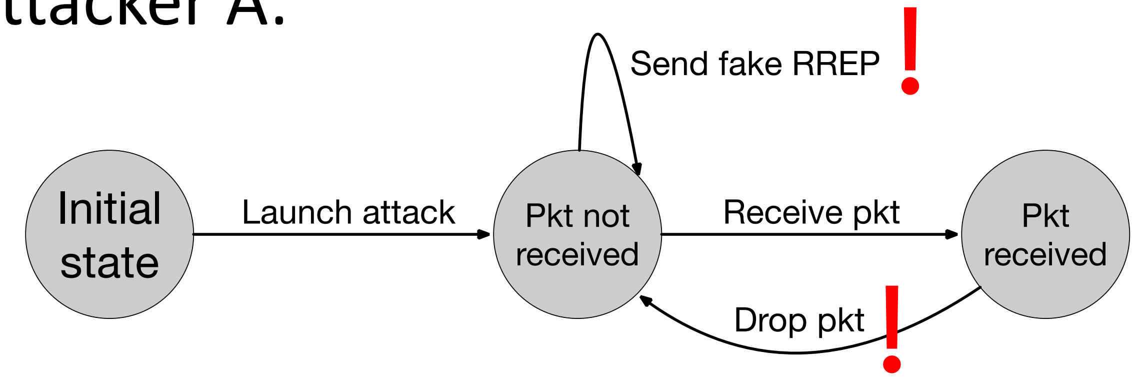
- For each of two ends of the wormhole, X and Y:



Broadcast wormhole route
Initial state
Setup wormhole
Pkt not received
Receive pkt
Pkt received
Send pkt to the other end
Route w/ wormhole is shorter
Ready to fwd through wormhole
Route wo/ wormhole is shorter
Ready to fwd normally
Send pkt to a legitimate neighbor

Defend against single attacks by detecting abnormal events

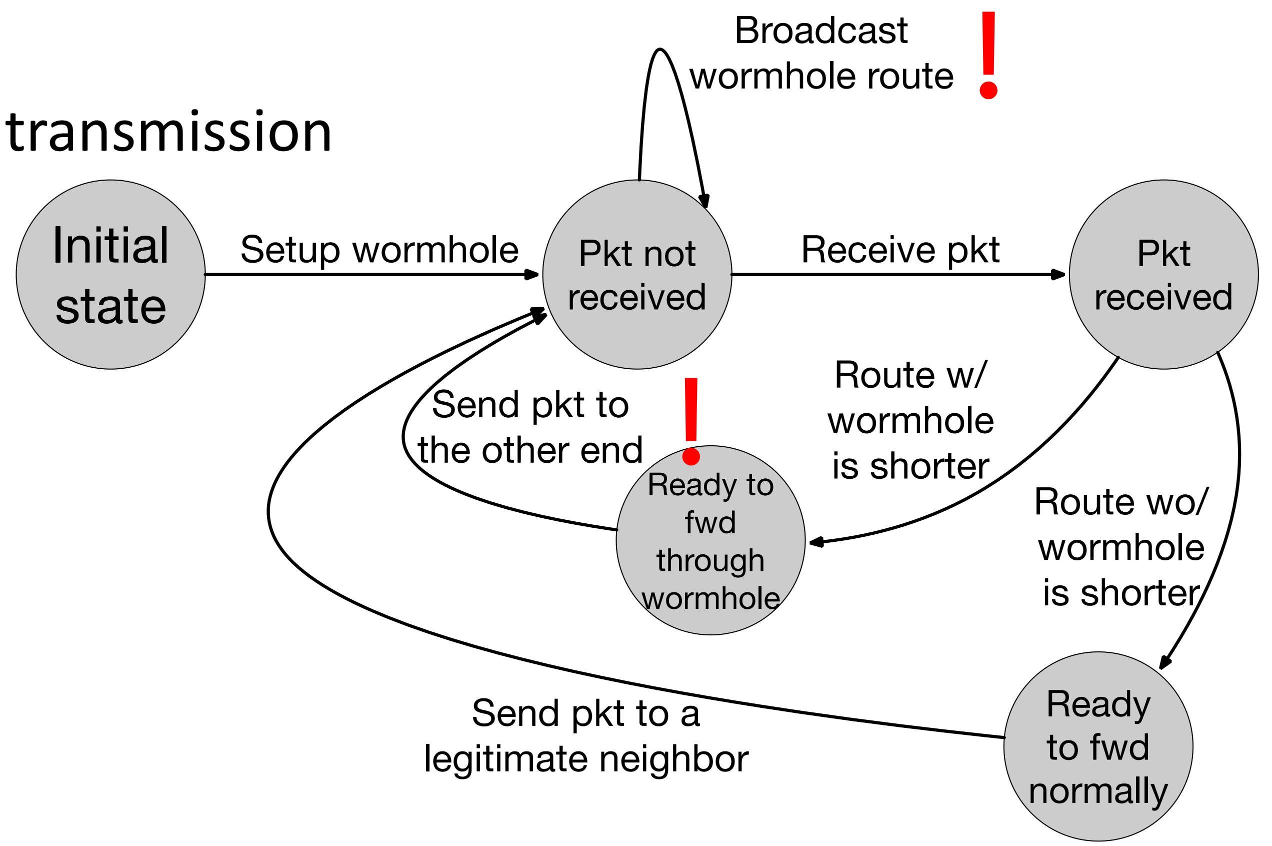
- For blackhole attacker A:



- Detector $D_A = \{d_{A1,}\ d_{A2}\}$
    - d_{A1}: monitors fake RREP
    - d_{A2}: monitors packet drop

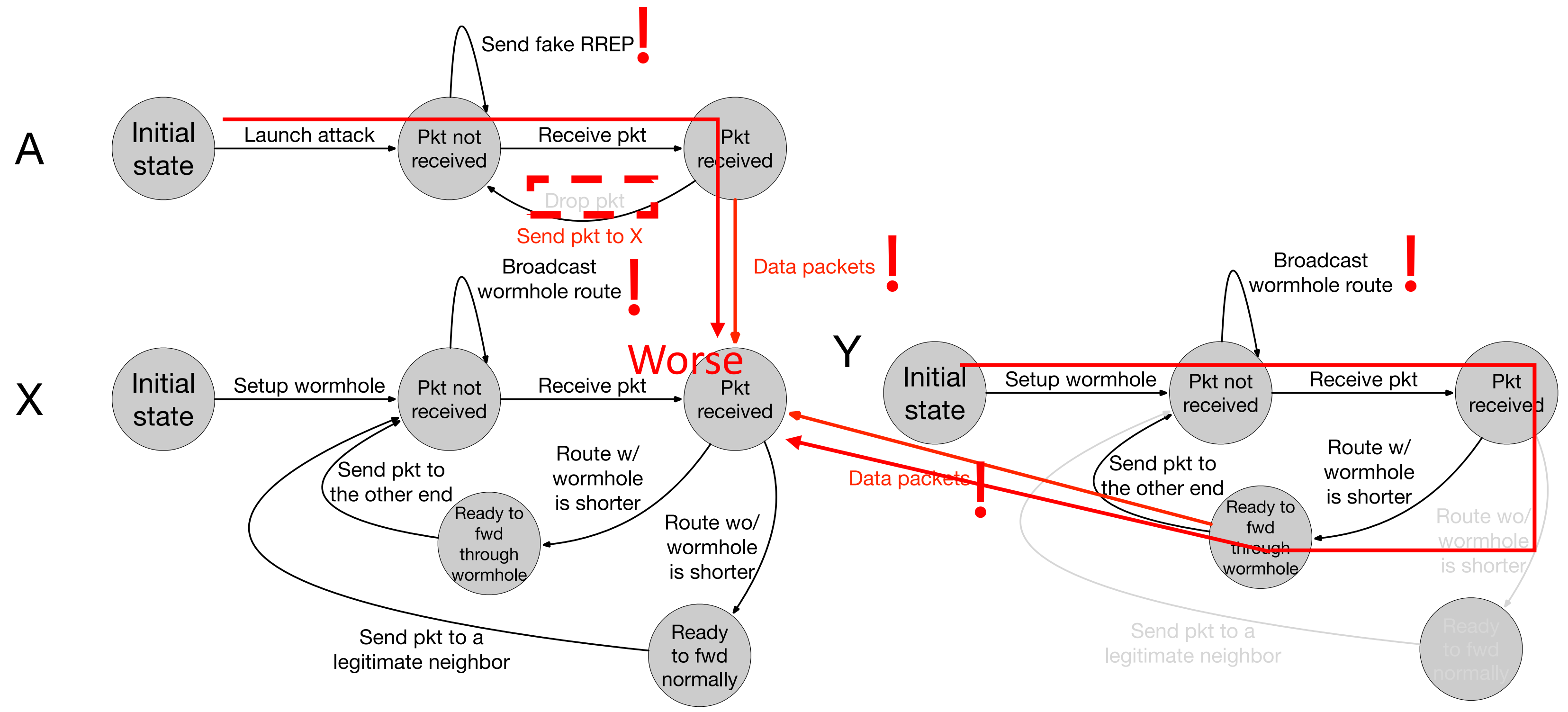
Defend against single attacks by detecting abnormal events (cont'd)

- For two ends of the wormhole X and Y:
- Detector $D_B = \{d_{B1}, d_{B2}\}$
  - d_{B1}: monitors abnormal RREP
  - d_{B2}: monitors abnormal packet transmission


Initial state
Setup wormhole
Pkt not received
Broadcast wormhole route
!
Receive pkt
Pkt received
Route w/ wormhole is shorter
Route wo/ wormhole is shorter
Ready to fwd through wormhole
!
Send pkt to the other end
Ready to fwd normally
Send pkt to a legitimate neighbor

Modeling and detecting collaborative attacks

- The 'bad' state (`X receives pkt`) becomes 'worse'.
- Detection d_{A2} becomes ineffective since A no longer drops packets.
- The bad state can be reached through more paths due to collaboration.
- Thus, the detectors DA and DB have to collaborate to defend. $D_{collab} = \{d_{A1}, d_{B1}, d_{B2}\}$

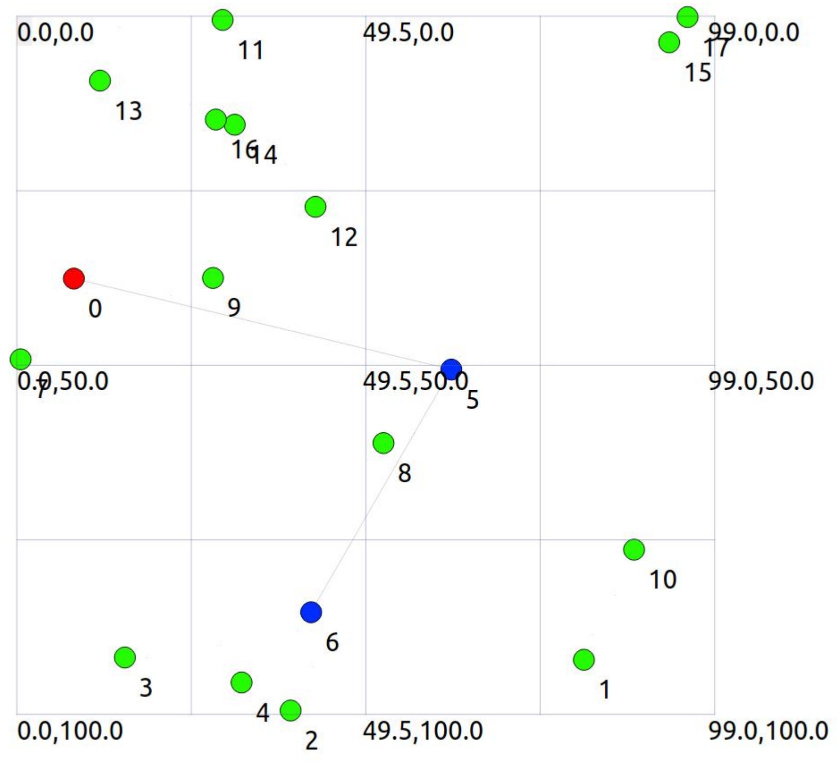


Evaluate the detection and defend mechanisms with ns3

We have implemented the above example with ns3

- Red node is the blackhole attacker A
- Blue nodes are wormhole attackers X and Y
- Gray lines are tunnel and wormhole

We will evaluate our solution mechanism with ns3



Follow-up research questions

- Given more single attack patterns (e.g., replication attacks, rushing attacks), we need to define causal rules to automatically judge whether and how those attacks can collaborate.
- As the number of attackers increases, the collaborative attack graph becomes more complex, we need more advanced techniques to exhaust the paths from initial states to bad states.
- In a system, how do we know that abnormal events are happening which may lead the system to a bad state?
  - We plan to use machine learning approaches (next slides)

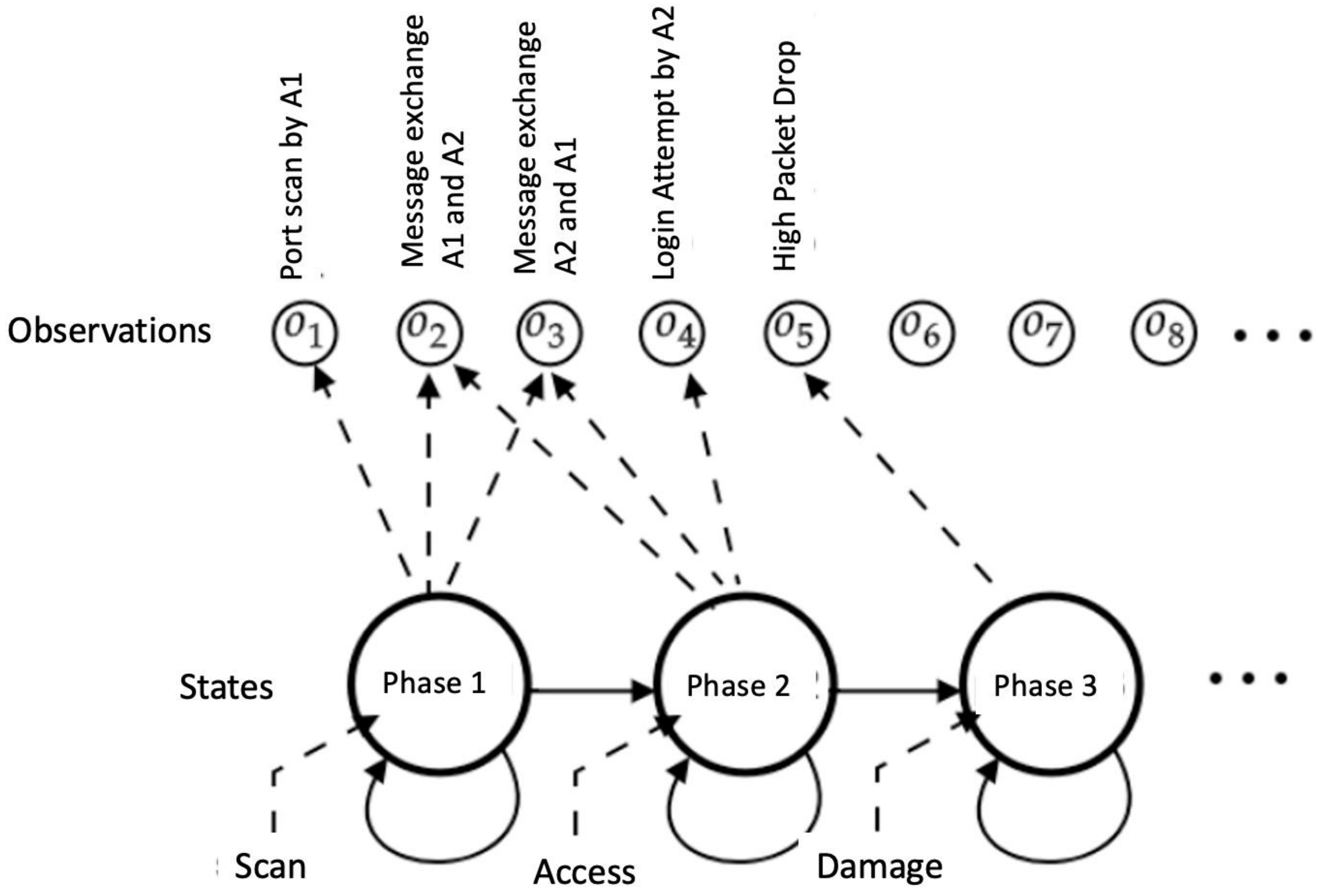
Why to use Machine Learning

- The ML algorithm can continuously learn and adapt to new attack patterns.
- It can analyze large amounts of data to detect advanced threats and reduce false positives/negatives.
- It is initially resource-intensive but more economical in the long term.
- It reduces manual updates.

Hidden Markov Model

- A sudden spike in port scans on critical servers, coupled with a surge in message traffic and repeated login failures, potentially indicates a collaborative attack.
- HMM leverages temporal relationships and probabilistic state transitions to dynamically monitor and identify potential network attacks.

Hidden Markov Model



How Hidden Markov Model works?

- System States can be Normal, Port Scan, Access, Damage
- The system observes events called observations.
- The algorithm can be trained on data including port activities, message logs, etc.
- Baum-Welch (BW) and Viterbi algorithms can be used for training.
- HMM can be applied to real-time data for state sequence analysis.
- If there are suspicious activities, the IP address of the attacker can be blocked, and notifications can be provided

Utilizing LSTM (Long Short-Term Memory) Network

- LSTM's strength lies in its ability to process sequential data and capture long-term patterns, making it highly effective for real-time network security monitoring.

For instance, LSTMs can detect irregular patterns, such as identifying a mild port scan followed by spikes in messages and unusual login attempts (both could happen days later “mild port scan”). This capability allows LSTM to flag such sequences as potential collaborative attacks.

How can LSTM work

- Dataset:
  - Timestep 1: event 1 [Normal Traffic]
  - Timestep 2: event 2 [Port Scan]
  - Timestep 3: event 3 [Increased Messaging]
  - Timestep 4: event 4 [Failed Logins]

- LSTM Processing: LSTM can process all previous states to predict collaboration
  - input: event 1 => output: No Attack
  - input: event 1, event 2 => output: Port Scan
  - input: event 1, event 2, event 3 => output: Access
  - input: event 1, event 2, event, event 4 => output: Potential collaborative attack

Contrastive Learning

- Contrastive Learning's strength lies in its ability to learn nuanced differences between normal and malicious behaviors.
- A model trained with Contrastive Learning could recognize irregular message exchanges and login attempts, distinguishing them from normal behavior and flagging them as potential attacks.
- Not every login attempt failure is malicious; contrastive learning can help distinguish between normal login attempt failure and malicious login attempt failure.

How does Contrastive Learning work?

- The algorithm is provided an event that is an Anchor (without anomaly) during training.

- All other events are positive (P) or negative (N). The positive (P) event is an anomaly. The negative (N) is a legitimate data point.

- The algorithm maximizes the distance between A and P while minimizing the distance between A and N.

- In real-time, the algorithms compute the distance between the event and anchor to predict whether the event is an anomaly or legitimate.

Problem Statement

Packet drop attacks put severe threats to Ad Hoc network performance and safety

- Directly impact the parameters such as packet delivery ratio
- Will impact security mechanisms such as distributed node behavior monitoring
- Different approaches have been proposed
    - Vulnerable to collaborative attacks
    - Have strong assumptions of the nodes

Problem Statement

Many research efforts focus on individual attackers

- The effectiveness of detection methods will be weakened under collaborative attacks
  - E.g., in “watchdog”, multiple malicious nodes can provide fake evidences to support each other’s innocence
  - In wormhole and Sybil attacks, malicious nodes may share keys to hide their real identities

Problem Statement

We focus on collaborative packet drop attacks. Why?

- Secure and robust data delivery is a top priority for many applications
- The proposed approach can be achieved as a reactive method: reduce overhead during normal operations
- Can be applied in parallel to secure routing

Related Work

Detecting packet drop attacks

- Audit based approaches
  - Whether or not the next hop forward the packets
  - Use both first hand and second hand evidences
  - Problems:
    - Energy consumption of eavesdropping
    - Can be cheated by directional antenna
    - Authenticity of the evidence
- Incentive based approaches
  - Nuggets and credits
- Multi-hop acknowledgement

Related Work

Collaborative attacks and detection

- Classification of the collaborative attacks
- Collusion attack model on secure routing protocols
- Collaborative attacks on key management in MANET
- Detection mechanisms:
    - Collaborative IDS systems
    - Ideas from immune systems
    - Byzantine behavior based detection

REAct system and Vulnerability

REAct system:

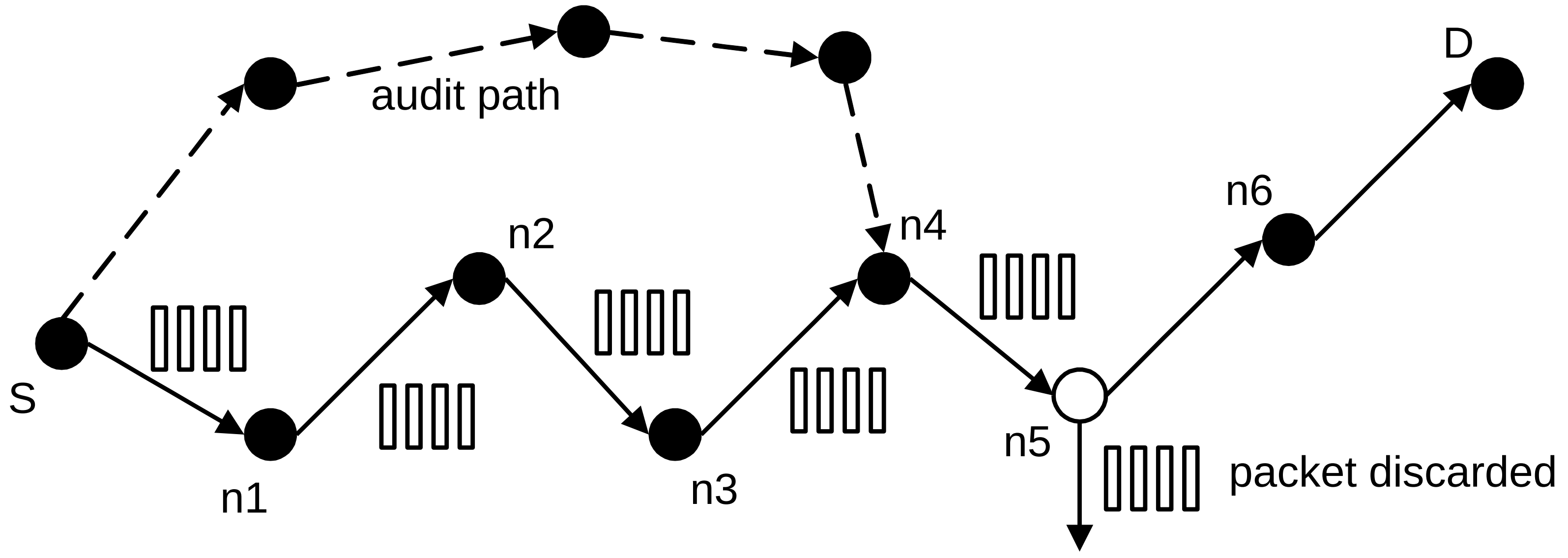
- Proposed by researchers in Arizona, ACM WiSec 2009
- Random audit based detector of packet drop
- A reactive approach: will be activated only when something bad happens
- Assumptions:
    - At least two node disjoint paths b/w any pair of nodes
    - Know the identity of the intermediate nodes
    - Pair-wise keys b/w the source and the intermediate nodes

REAct system and Vulnerability

Working procedure of REAct

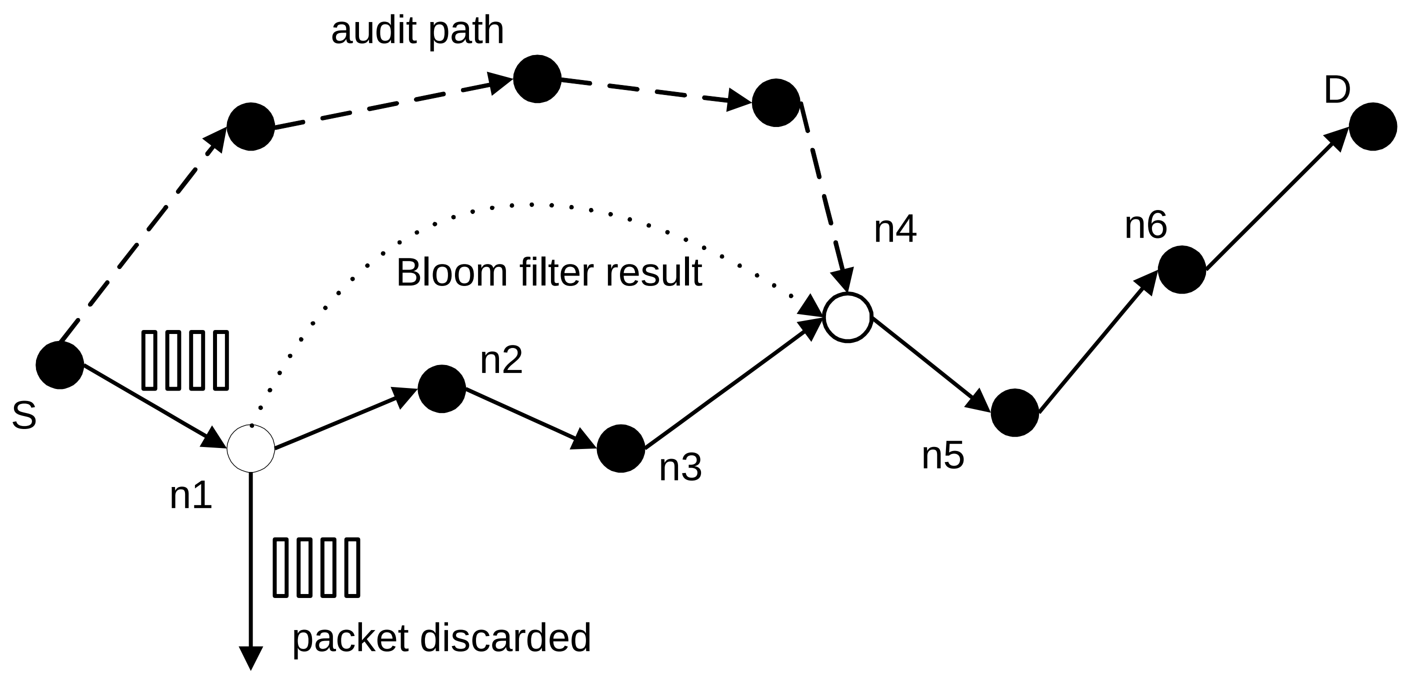
- Destination detects the drop in packet arriving rate and notifies the source
- Source randomly selects an intermediate node and asks it to generate a behavioral proof of the received packets
- Intermediate node constructs a bloom filter using these packets
- Source compares the bloom filter to its own value
    - If match: the attacker is after the intermediate node
    - Otherwise, it is before the intermediate node
- Repeat the procedure until the bad link is located

REAct system and vulnerability



Example of REAct: the source selects n4 to be the first audited node. n4 generates the correct bloom filter, so the attacker is between n4 and D.

Collaborative attacks on REAct



n1 and n4 are collusive attackers. n1 discards the packets but delivers the bloom filter to n4. Now the source will think that the attacker is between n4 and D.

Why REAct is vulnerable to this attack: the source can verify the bloom filter, but not the generator of the filter.

Proposed approach

Assumptions:

- Source shares a different secret key and a different random number with every intermediate node
- All nodes in the network agree on a hash function *h()*
- There are multiple attackers in the network
    - They share their secret keys and random numbers
    - Attackers have their own communication channel
    - An attacker can impersonate other attackers

Proposed approach

Hash based approach:

- Every node will add a fingerprint into the packet

  S1 sends out the packet to n1:

  S → *n1*: (*S*, *D*, data packet, random number *t0*)

  Node *n1* will combine the received packet and its random number *r1* to calculate the new fingerprint:

  t1 = h(*r1* || *S* || *D* || data packet || *t0* || *r1*)

  n1 → *n2*: (*S*, *D*, data packet, *t1*)

  The audited node will generate the bloom filter based on the data packets and the fingerprints

  The source will generate its own bloom filter and compare it to the value of the audited node

Proposed approach

Why our approach is safe

- The node behavioral proofs in our proposed approach contain information from both the data packets and the intermediate nodes.
- Theorem 1. If node *ni* correctly generates the value *ti*, then all innocent nodes in the path before *ni* (including *ni*) must have correctly received the data packet selected by *S*.

Proposed approach

Why this approach is safe

- The ordered hash calculations guarantee that any update, insertion, and deletion operations to the sequence of forwarding nodes will be detected.
- Therefore, we have:
  - if the behavioral proof passes the test of S, the suspicious set will be reduced to {*ni, ni+1, ---, D*}
  - if the behavioral proof fails the test of S, the suspicious set will be reduced to {S, n1, ---, ni}

Discussion

- Indistinguishable audit packets
  - The malicious node should not tell the difference between the data packets and audited packets
  - The source will attach a random number to every data packet
- Reducing computation overhead
  - A hash function needs 20 machine cycles to process one byte
  - We can choose a part of the bytes in the packet to generate the fingerprint. In this way, we can balance the overhead and the detection capability.

Discussion

- Security of the proposed approach
  - The hash function is easy to compute: very hard to conduct DoS attacks on our approach
  - It is hard for attackers to generate fake fingerprint: they have to have a non-negligible advantage in breaking the hash function
- The attackers will adjust their behavior to avoid detection
  - The source may choose multiple nodes to be audited at the same time
  - The source should adopt a random pattern to determine the audited nodes

Dealing with Collaborative Attacks

- Earlier approach is vulnerable to collaborative attacks
- Propose a new mechanism for nodes to generate behavioral proofs
  - Hash based packet commitment
  - Contain both contents of the packets and information of the forwarding paths
  - Introduce limited computation and communication overhead
- Extensions:
  - Investigate other collaborative attacks
  - Integrate our detection method with secure routing protocols