Trusted Router and Collaborative Attacks

Bharat Bhargava

Trusted Router and Protection Against Collaborative Attacks

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

Collaborative Attacks

Informal definition:

"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
  - 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

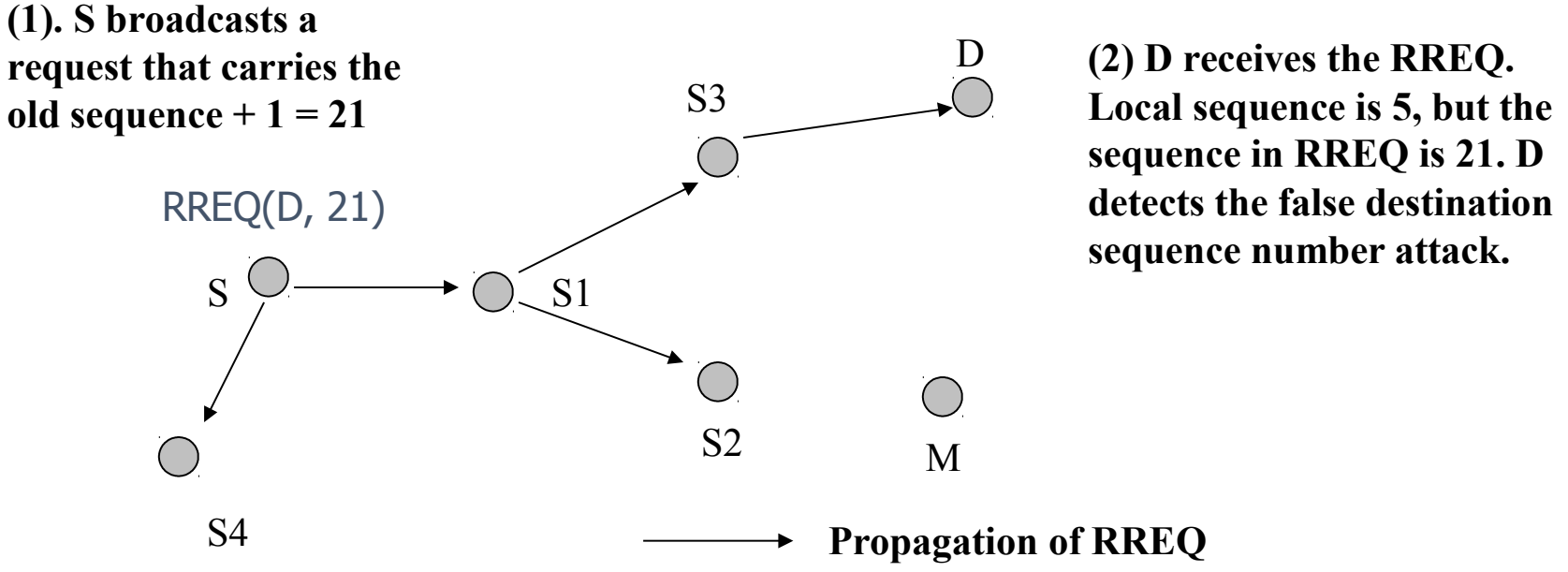
Blackhole attack detection: Reverse Labeling Restriction (RLR)

- Every host maintains a blacklist to record suspicious hosts who gave wrong route related information
- Blacklists are updated after an attack is detected
- The destination host will broadcast an INVALID packet with its signature when it finds that the system is under attack on sequence. The packet carries the host's identification, current sequence, new sequence, and its own blacklist
- Every host receiving this packet will examine its route entry to the destination host. The previous host that provides the false route will be added into this host's blacklist

RLR (cont'd)

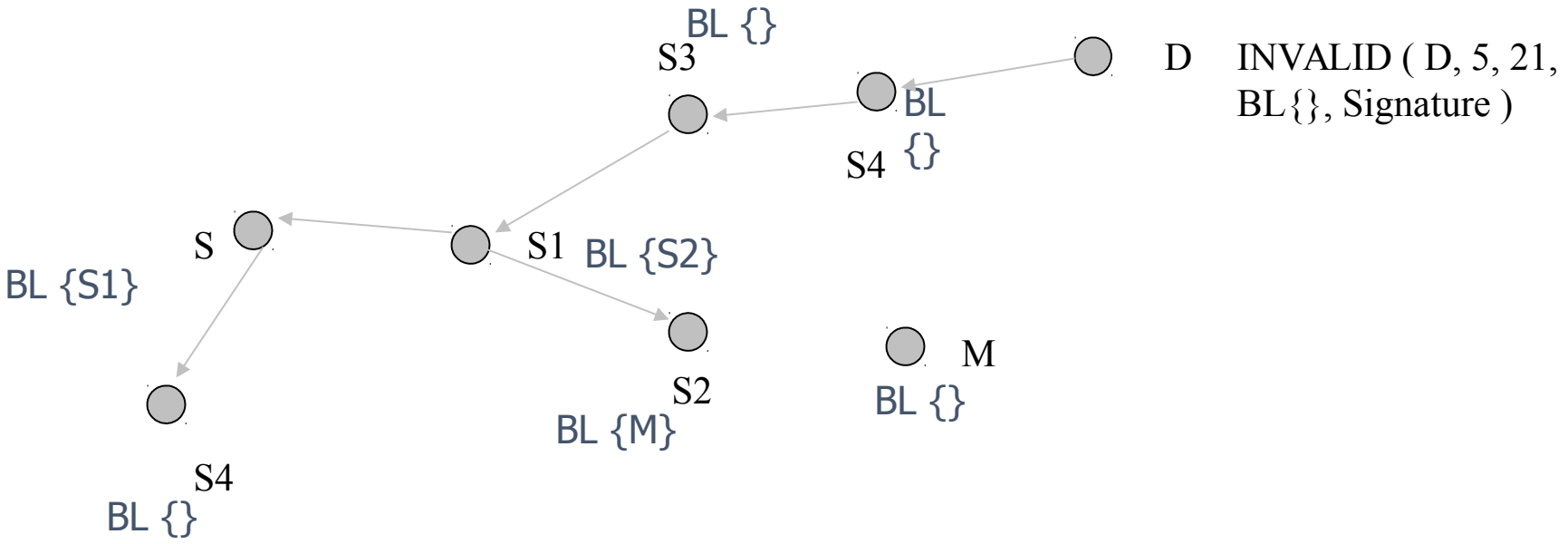
Detecting false destination sequence attack by destination host during route rediscovery

- During Route Rediscovery, False Destination Sequence Number Attack is Detected, S needs to find D again
- Node movement breaks the path from S to M (trigger route rediscovery)



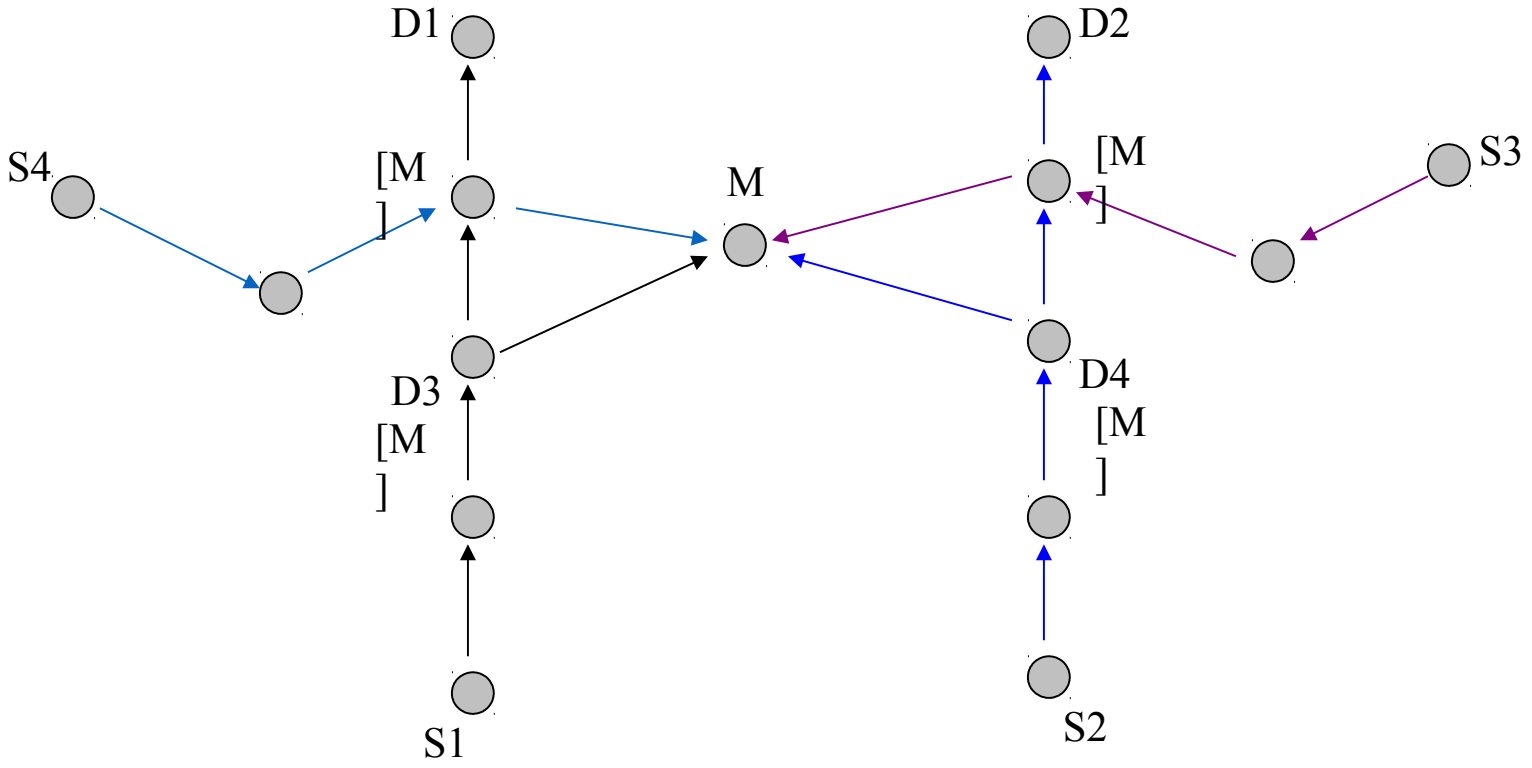
RLR (cont'd)

- Correct destination sequence number is broadcasted. Blacklist at each host in the path is determined



RLR (cont'd)

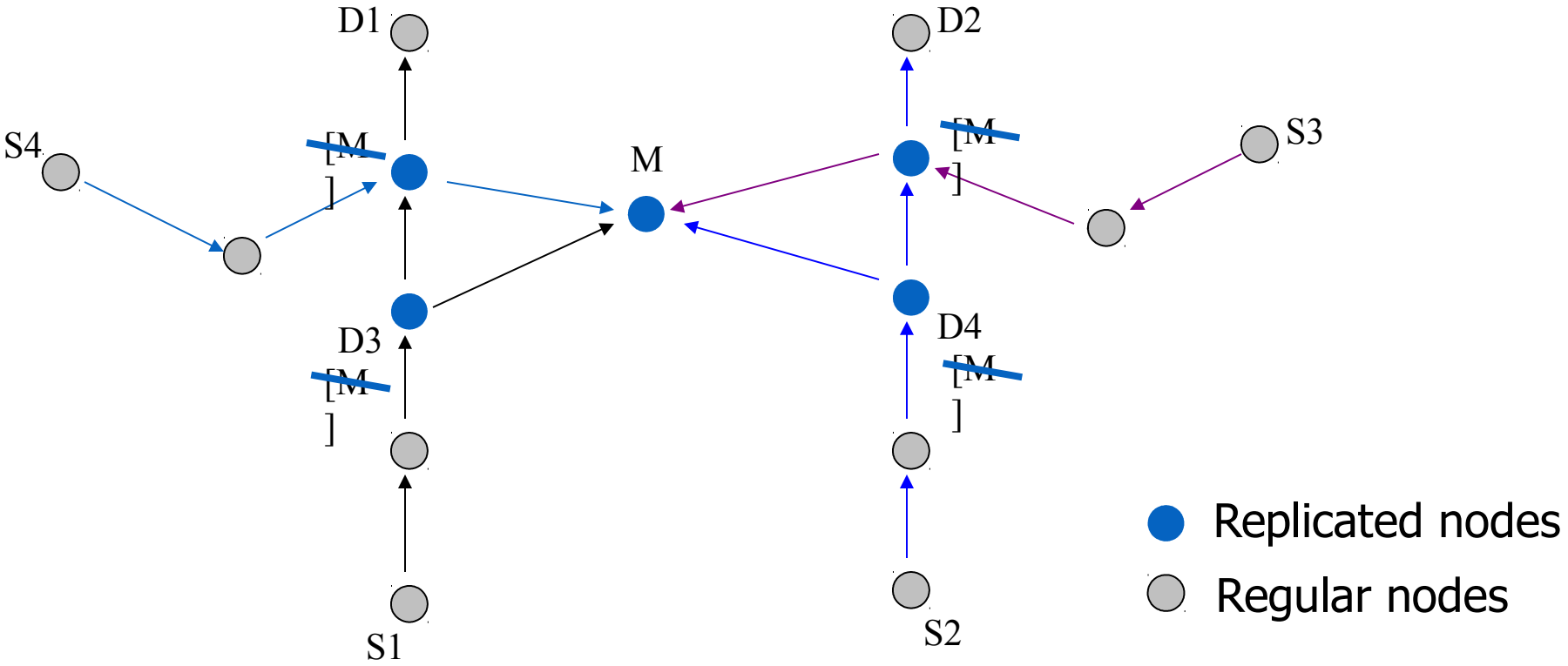
- Malicious site is in blacklists of multiple destination hosts



M attacks 4 routes (S1-D1, S2-D2, S3-D3, and S4-D4). When the first two false routes are detected, D3 and D4 add M into their blacklists. When later D3 and D4 become victim destinations, they will broadcast their blacklists, and every host will get two votes that M is malicious host

Two Attacks in Collaboration: blackhole & replication

- The RLR scheme cannot detect the two attacks working simultaneously
- The malicious node M relies on the replicated neighboring nodes to avoid the blacklist



Defending against Collaborative Packet Drop Attacks on Router

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:

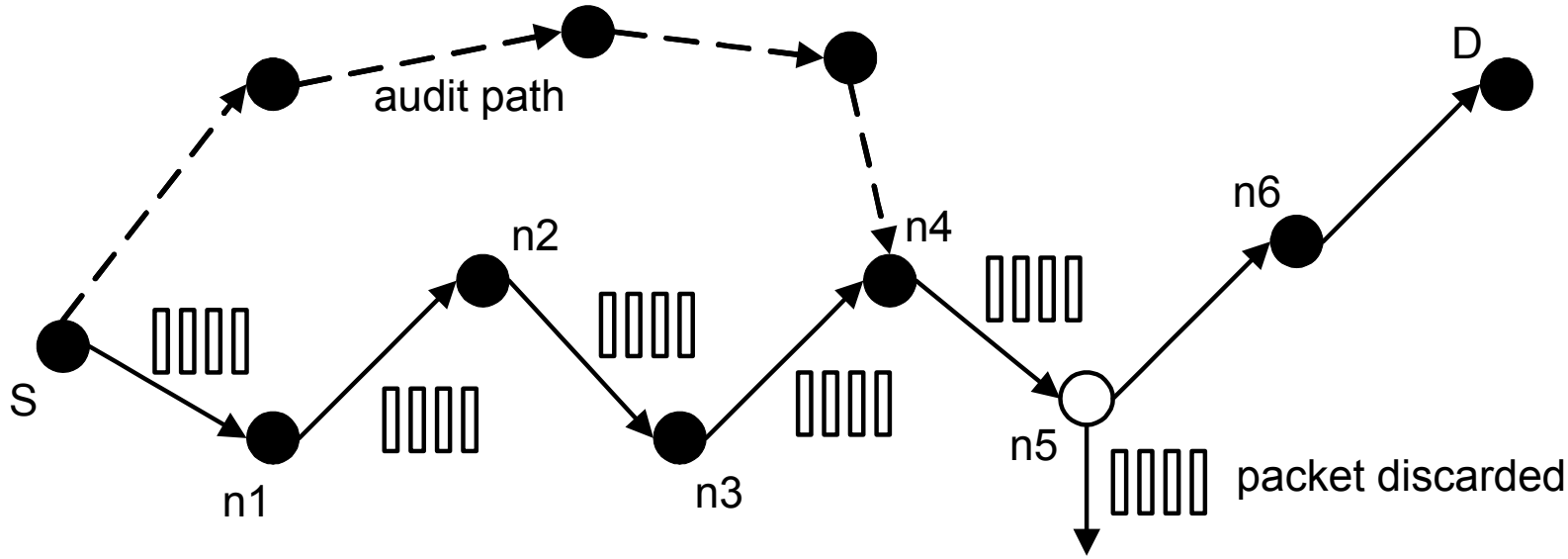
- Proposed by researchers in Arizona, ACM WiSec 2009
- Random audit based detector of packet drop
- A reactive approach: will be activated only when something bad happens
- Assumptions:
  - At least two node disjoint paths b/w any pair of nodes
  - Know the identity of the intermediate nodes
  - Pair-wise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

REAct system and Vulnerability

Working procedure of REAct

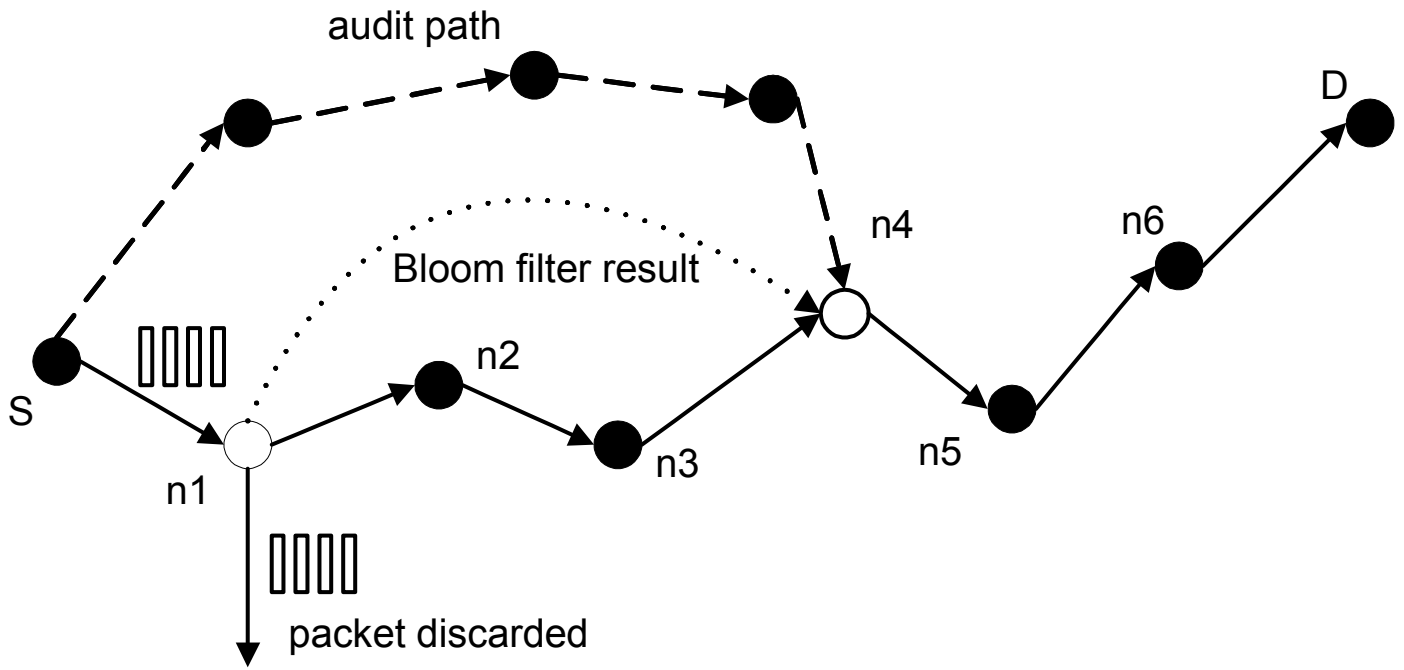
- Destination detects the drop in packet arriving rate and notifies the source
- Source randomly selects an intermediate node and asks it to generate a behavioral proof of the received packets
- Intermediate node constructs a bloom filter using these packets
- Source compares the bloom filter to its 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