Wireless Security

• **DoS attacks and defenses**
  - Physical layer: jamming
  - **MAC** layer: greedy **MAC**
  - Network layer: routing attacks
  - Transport layer: cross layer attacks
The Feasibility of Launching and Detecting Jamming Attacks in Wireless Networks
What is radio interference attack?

- Intentionally interfering with the physical transmission and reception of wireless communications.
  - Emitting radio frequency signals that do not follow underlying MAC protocol. (Jamming)
If you’re a jammer, how would you jam the channel?
Jamming attack models

- **Constant jammer**
  - Always emit random bits of radio signal

- **Deceptive jammer**
  - Always emit preamble bits

- **Random jammer**
  - Alternate between sleeping and jamming states -> Conserve Energy

- **Reactive jammer**
  - Transmit signal when jammer senses channel activity -> Harder to detect
Effectiveness

- **PSR**: Packet Send Ratio
- **PDR**: Packet Delivery Ratio

<table>
<thead>
<tr>
<th>$d_{XA}$ (inch)</th>
<th>BMAC</th>
<th>1.1.1 MAC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSR (%)</td>
<td>PDR (%)</td>
</tr>
<tr>
<td>38.6</td>
<td>74.37</td>
<td>0.43</td>
</tr>
<tr>
<td>54.0</td>
<td>77.17</td>
<td>0.53</td>
</tr>
<tr>
<td>72.0</td>
<td>99.57</td>
<td>93.57</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$d_{XA}$ (inch)</th>
<th>BMAC</th>
<th>1.1.1 MAC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSR (%)</td>
<td>PDR (%)</td>
</tr>
<tr>
<td>38.6</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>54.0</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>72.0</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$d_{XA}$ (inch)</th>
<th>BMAC</th>
<th>1.1.1 MAC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSR (%)</td>
<td>PDR (%)</td>
</tr>
<tr>
<td>$t_j = U[0,31]$</td>
<td>38.6</td>
<td>79.45</td>
</tr>
<tr>
<td>$t_2 = U[0,31]$</td>
<td>44.0</td>
<td>80.15</td>
</tr>
<tr>
<td>54.0</td>
<td>80.43</td>
<td>99.00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>$d_{XA}$ (inch)</th>
<th>BMAC</th>
<th>1.1.1 MAC</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>PSR (%)</td>
<td>PDR (%)</td>
</tr>
<tr>
<td>38.6</td>
<td>99.00</td>
<td>0.00</td>
</tr>
<tr>
<td>54.0</td>
<td>100.0</td>
<td>99.24</td>
</tr>
<tr>
<td>72.0</td>
<td>100.0</td>
<td>99.35</td>
</tr>
<tr>
<td>$m = 7$ bytes</td>
<td>38.6</td>
<td>99.00</td>
</tr>
<tr>
<td></td>
<td>54.0</td>
<td>100.0</td>
</tr>
<tr>
<td></td>
<td>72.0</td>
<td>100.0</td>
</tr>
<tr>
<td>$m = 33$ bytes</td>
<td>38.6</td>
<td>99.00</td>
</tr>
<tr>
<td></td>
<td>44.0</td>
<td>99.00</td>
</tr>
<tr>
<td></td>
<td>54.0</td>
<td>99.25</td>
</tr>
</tbody>
</table>

Table 1: The resulting PSR and PDR for different jammer models under various scenarios.
How would you detect jamming?
Overview

• What is radio interference attack?

• **Measurements to detect jamming attacks**
  - Signal strength
    • Match jam signals with legitimate signal pattern
  - Carrier sensing time
    • Jamming incurs long carrier sensing time (skip)
  - Packet delivery ratio (PDR)
    • Jamming incurs lower PDR

• Detection schemes of jamming attacks
Signal strength spectral discrimination

- Employed Higher Order Crossings (HOC) to show difference between samples.
Packet Delivery Ratio (PDR)

- **PDR degradation**
  - Small PDR degradation
    - normal congestion (e.g., PDR ~ 78% under 3 flows with MaxTraffic)
  - Large PDR degradation
    - Sender battery failure, sender moving out of communication range, or being jammed.

- Better measure than signal strength or carrier sensing time if we can differentiate a jamming attack from other network dynamics.
Overview

- What is radio interference attack?
- Measurements to detect jamming attacks
- Detection schemes of jamming attacks
  - PDR + signal strength
  - PDR + location information
PDR + Signal strength

- Classify poor link by PDR,
  Consistency check by signal strength

<table>
<thead>
<tr>
<th>PDR</th>
<th>Observed signal strength</th>
<th>Typical scenarios</th>
</tr>
</thead>
<tbody>
<tr>
<td>PDR = 0</td>
<td>low signal strength</td>
<td>non-jammed: neighbor failure, neighbor absence,</td>
</tr>
<tr>
<td></td>
<td></td>
<td>neighbors being blocked, etc.</td>
</tr>
<tr>
<td>PDR = 0</td>
<td>high signal strength</td>
<td>node jammed</td>
</tr>
<tr>
<td>PDR low</td>
<td>low signal strength</td>
<td>non-jammed: neighbor being faraway</td>
</tr>
<tr>
<td>PDR low</td>
<td>high signal strength</td>
<td>node jammed</td>
</tr>
</tbody>
</table>
PDR + Signal strength

- **Jammed-region:**
  - SS > -73 dBm
  - PDR < 65%

- **Disadv:** For PDR window,
  - It must be jammed for a while
  - Hard to choose SS granularity
PDR + Location Information

- **Classify poor link by PDR**, Consistency check by GPS or virtual coordinates

- **Problem**: Node without neighbor

<table>
<thead>
<tr>
<th>Constant Jammer</th>
<th>1.1.1 MAC</th>
<th>PSR (%)</th>
<th>PDR (%)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$d_{XA}$ (inch)</td>
<td>$d_{XA}$ (inch)</td>
<td></td>
<td></td>
</tr>
<tr>
<td>38.6</td>
<td>38.6</td>
<td>1.00</td>
<td>1.94</td>
</tr>
<tr>
<td>54.0</td>
<td>54.0</td>
<td>1.02</td>
<td>2.91</td>
</tr>
<tr>
<td>72.0</td>
<td>72.0</td>
<td>0.92</td>
<td>3.26</td>
</tr>
</tbody>
</table>
Summary

• Present four different jammer attack models
• Develop detection schemes
  - SS, carrier sensing time, and PDR alone is not enough
  - PDR + signal strength
    • Reactive consistency check
    • Due to node mobility, PDR window length and SS granularity should be selected carefully.
  - PDR + location information
    • Proactive consistency check
    • Given location information a priori, only need normal PDR.
      - Density of network matters
      - May not work well under obstacles
      - Cannot determine jam for isolated nodes.
      - Freq. of location advertisement matters
Questions

• What are limitations in the detection schemes?

• How to mitigate jamming?
DoS Attacks and Defenses

• **Attacks at various layers**
  - Physical layer: jamming
  - **MAC** layer: greedy sender and receiver
  - Network layer: routing attacks
  - Transport layer: cross layer attacks
GREEDY Sender IN IEEE 802.11 HOTSPOTS
Motivation

• Hotspot industry is a tremendous financial success

  - Revenue:
    • 969 million in 2005
    • 3.46 billion in 2009

  - # hotspots
    • 100,000 in 2005
    • 200,000 by 2009

• Increasing motives for users to misbehave
Benefits of MAC Misbehavior

- More effective than routing and transport misbehavior
  - MAC misbehavior is applicable to both WLAN and multihop wireless networks
  - Can affect all traffic using the same MAC
  - Can be further combined to other misbehavior to increase its impact
System Model

- AP is trusted and implements detection system
  - No modification to clients
- Only clients misbehave
- Clients are greedy and do not maliciously disrupt network

Figure 1: The distributed coordination function (DCF) of IEEE 802.11 operating in RTS/CTS mode.
How to misbehave?
Misbehavior Techniques

- **MAC greedy misbehavior on data path**
  - Scramble CTS frames
    - Action: Cheater hears RTS frame destined to another node, intentionally causes collision of CTS
    - Effect?
  - Scramble DATA/ACK frame
    - Effect?
  - Transmit RTS or DATA after SIFS as opposed to DIFS
  - Increasing NAV to prevent other nodes within range from transmitting
  - Reduce the backoff time
  - A cheater can combine several of the above techniques or dynamically change its misbehavior
How to detect misbehavior?
Detecting Misbehavior

• Why not using throughput?
Detecting Misbehavior

• Why not using throughput?
  - Throughput is affected by many factors, such as traffic demands, SNR, transport protocol, device drivers, protocol implementation, etc.

• Backoff
  - Most direct way to detect cheaters
  - Challenges
    • How to determine backoff time?
    • Hidden terminals: not everyone sees idle channels and busy channels at the same time
Domino

- Collect traffic traces of sending stations every monitoring periods
- Pass the traces for several tests

---

```plaintext
loop
  if monitoring period elapsed since last check then
    for each active station $S_i$ do
      for $j = 1$ to $6$ do
        execute Test $j$
```
Components of Domino

- Periodic monitoring
  - Scrambled frames
    - True
    - False
  - Shorter than DIFS
    - True
    - False
  - Oversized NAV
    - True
    - False
  - Maximum backoff
    - True
    - False
  - Actual backoff
    - True
    - False
  - Consecutive backoff
    - True

Node S is misbehaving
Call the punishing function
Detailed Tests

• Condition of a cheater

\[ \text{check}_x(S_i, \text{condition}_x): \]

\[
\begin{align*}
\text{if } \text{condition}_x \text{ is true} & \text{ then} \\
\text{cheat}\_\text{count}_x(S_i) & := \text{cheat}\_\text{count}_x(S_i) + 1 \\
\text{if } \text{cheat}\_\text{count}_x(S_i) > K_x & \text{ then} \\
S_i & \text{ is misbehaving} \\
& \text{call the punishing function} \\
\text{else if } \text{cheat}\_\text{count}_x(S_i) > 0 & \text{ then} \\
\text{cheat}\_\text{count}_x(S_i) & := \text{cheat}\_\text{count}_x(S_i) - 1
\end{align*}
\]

• Any comments?
Scrambled Frames

- Its retransmission is significantly fewer than that of other stations

<table>
<thead>
<tr>
<th>Test 1 Scrambled frames</th>
</tr>
</thead>
<tbody>
<tr>
<td>$condition_1 := num_{rtx}(S_i) &lt; \phi \times E_{j\neq i}[num_{rtx}(S_j)]$</td>
</tr>
<tr>
<td>call $check_1(S_i, condition_1)$</td>
</tr>
</tbody>
</table>
Detection of Manipulated protocol

• **Shorter than DIFS**

  \[
  \text{Test 2 Shorter than DIFS} \\
  \text{condition}_2 := \text{idle\_time\_after\_ACK}(S_i) < \text{DIFS} \\
  \text{call check}_2(S_i, \text{condition}_2)
  \]

• **Oversized NAV**

  \[
  \text{Test 3 Oversized NAV} \\
  \text{condition}_3 := \text{NAV}(S_i) > A \times \text{tx\_duration}(S_i) \\
  \text{call check}_3(S_i, \text{condition}_3)
  \]
Detecting too small CW?
Detection of Manipulated protocol (Cont.)

• Backoff manipulation

<table>
<thead>
<tr>
<th>Test 4 Maximum backoff</th>
</tr>
</thead>
<tbody>
<tr>
<td>$condition_4 := max_{bkf}(S_i) &lt; threshold_{max_{bkf}}$</td>
</tr>
<tr>
<td>call $check_4(S_i, condition_4)$</td>
</tr>
</tbody>
</table>

• Any problems?
Detection of Manipulated protocol (Cont.)

- **Backoff manipulation**

---

**Test 5 Actual backoff**

\[ condition_5 := B_{ac}[S_i] < \alpha_{ac} \times B_{acnom} \]

call check_5(S_i, condition_5)
Detection of Manipulated protocol (Cont.)

- **Consecutive backoff**

**Test 6 Consecutive backoff**

\[
\text{conditions}_6 := B_{co}[S_i] < \alpha_{co} \times B_{conom}
\]

**Figure 4:** Measurement of the *consecutive backoff*. Backoff values are taken only between consecutive non-interleaved transmissions from \( S \).
SIMULATION RESULTS

- Simulation topology

Two Cases

1) UDP traffic: one cheater and 7 regular stations sending CBR traffic, rate being 500 bytes/packet, 200 packets

2) TCP traffic: Each of the 8 stations runs an FTP application; with one station cheating.

RESULTS: Are averaged over 10 simulations, 110s each. Monitor period is 10s

Figure 5: Simulation scenarios: 8 stations send UDP or TCP data to the AP, which also generates traffic similarly to one station. The distance between the stations and the AP is 50m. All stations are within range of each other.

Ns-2 was used for the simulation
Impact of Misbehavior

The cheater gains significantly more throughput than well-behaved stations.
Domino accurately detects backoff interval under UDP traffic.
Testbed Evaluation

- Cheater modified the CW used in the backoff procedure.

Figure 9: Experimental setup.
Performance Results

- *CW min and CW max is set to 0, 1, 3, 7, 15 which correspond to misbehavior coefficients of 1, .93, .8, .53 and 0.*
Summary

• **MAC misbehavior is a serious attack**
  - Easy to launch
    • Compliant to IEEE 802.11 standard
    • No hardware change is required
  - Have large impact

• How to become greedy receivers?
DoS Attacks against control plane
1: Association request
2: Association response
3: EAPOL-Start
4: Request/Identity
5: EAP-Response/Identity
6: EAP-Request
7: EAP-Response
8: EAP-Success
9: EAPOL-Key (WEP)

5: Radius-Access-Request
6: Radius-Access-Challenge
7: Radius-Access-Challenge
8: Radius-Access-Accept

Distribute dynamic key for WEP
Misbehavior Techniques (Cont.)

- Misbehavior against management frames
  - Spoof disassociation frames
  - Spoof de-authentication frames
    - Neither authentication or association frames are authenticated
Power saving with wake-up patterns (infrastructure)

Traffic Indication Map (TIM): list of unicast receivers transmitted by AP
Delivery Traffic Indication Map (DTIM): list of broadcast/multicast receivers transmitted by AP
• **Misbehavior against management frames**
  - Spoof disassociation frames
  - Spoof de-authentication frames
    - Neither authentication or association frames are authenticated
  - Power control
    - Spoof polling message to cause AP to discard client’s packet while it is sleeping
    - Spoof TIM message to convince the client that there is no pending data for it
    - Spoof beacon frames with incorrect timestamp → cause clients to be out of time sync from APs
  - The above are DoS attacks (which may or may not benefit the cheater).
  - They all leverage the fact that management frames are not authenticated.
Wireless Security

• **DoS attacks and defenses**
  - Physical layer: jamming
  - MAC layer: greedy MAC
  - Network layer: routing attacks
  - Transport layer: cross layer attacks
Ariadne: A Secure On-Demand Routing Protocol for Ad Hoc Networks

Yih-Chun Hu
Adrian Perrig
David B. Johnson
Introduction

• Ad hoc network routing protocol
  - Assume every node cooperates and follows the protocol
  - How to make nodes cooperate?
  - How to make routing protocol tolerate adversary behaviors?

• Contributions of this paper
  - Focus on DSR
  - Give a model for the types of attacks for ad hoc networks
  - Present design and evaluation of new on-demand secure ad hoc network routing protocols
    • Ariadne with TESLA
Basic Operation of DSR

- **Route Discovery**

- **Route Maintenance**

- Omitting various optimization technique
Ad Hoc Network Routing Security (1/2)

- **Attacker Model**
  - Omit passive attack
    - Mainly threat confidentiality or anonymity
  - Active-y-x model
    - Attacker has \( x \) nodes, and among these \( y \) nodes are compromised nodes
    - Distribute the cryptographic information of \( y \) nodes to \( x-y \) nodes
  - Active VC model
    - Attacker has all nodes in a vertex cut
Ad Hoc Network Routing Security (2/2)

• General attacks on ad hoc network routing protocols
  - Routing disruption attacks
    • Routing legitimate data packets in dysfunctional way
      - Routing loop, black hole, gray hole, detours, gratuitous detour, black mail, worm hole
    • Rushing attack
      - Disseminates route request packet quickly
      - Suppressing any later legitimate route request packet (nodes think it’s duplication)
  - Resource consumption attacks
    • Consuming bandwidth and computational resource
    • Inject extra packets
    • DoS attack: effective for control packets (Why?)
Assumptions (1/2)

• **Network assumptions**
  - Disregard non-network-layer attacks
  - Bidirectional link
  - May drop, corrupt, reorder, duplicate packets in transmission

• **Node assumptions**
  - Little computational resources
  - Loosely synchronized (when used TELSA)
    • GPS can be used
  - Do not assume trusted hardware such as tamper proof
Assumptions (2/2)

• Ariadne relies on secrecy and authenticity of keys

• Security assumptions and key setup
  - Three key setup mechanism can be used
    • Pair-wise shared secret keys
    • Digital signature
      - One authentic public key for each node
    • TESLA
      - Assume setting up key sharing mechanism between communicating nodes
      - One authentic public TESLA key for each node
  - Key setup mechanism in paper
    • Key Distribution Center with shared secret keys or TESLA
Goals of Secure Routing

Three conditions of secure routing

- Target Authentication
  - To authenticate destination of route request
- Data authentication
  - To authenticate nodes in route request and route reply
  - TESLA
  - Shared symmetric key
    - Route reply packet has MAC list of all nodes in route
  - Digital signature
    - Route reply packet has signature list instead
- Per-hop hashing
  - To verify that no hop is omitted or added
Overview of TESLA (1/2)

• **TESLA**
  - Broadcast authentication protocol
  - Only a single message authentication code (MAC) is added
    - Asymmetric
    - Different from RSA: using clock sync. and delayed key disclosure instead of one-way expensive trapdoor functions
  - Assuming loose time synchronization and known pessimistic end-to-end delay
    - Maximum synchronization error ($\Delta$)
    - Pessimistic end-to-end delay ($\varepsilon$)
    - Key publishing delay $d$
Overview of TESLA (2/2)

- **Protocol**

  - **Sender**
    - Generate one-way hash chain
      \[ K_n, \ldots, K_0 \text{ s.t. } H(K_i) = K_{i-1} \]
    - Publish key
    - Publish schedule for \( K_i \) to \( T_0 + i*t \)
    - Packet \( P_i \)
      \[ = (M_i \mid \text{MAC}(K_i, M_i) \mid K_i-d) \]
    - Send \( P_i \) at \( T_s \)
      (Sender’s clock)

  - **Receiver**
    - Receive \( P_i \) by \( T_r \)
      (Receiver’s clock)
    - If \( T_r(\text{at most } T_s + \epsilon + 2\Delta) > T_0 + i*t \), Drop \( P_i \)
    - Else store it until \( P_i+d \) received
    - At Received \( P_i+d \),
      - verify \( K_n = H^{n-i}(K_i) \)
      - compute \( \text{MAC}(K_i, M_i) \)
      - with \( K_i \) in packet \( P_i+d \)
Ariadne (Design Goals)

- Low computational and communicational overhead
  - To prevent DoS Attack
  - Using TESLA for authentication on nodes in routing path
- DoS protection
Ariadne
(Basic Ariadne Route Discovery)

• Ariadne route discovery with TESLA
  - Assuming shared key exist between source and destination \((K_{SD}, K_{DS})\)
  - All nodes know authentic TESLA key of one-way hash key chain of other nodes

• Notation
  - \(S, D\) : source , destination
  - \(A, B, C, D\) : nodes
  - \(K_{AB}\) : secret MAC keys shared between A and B , only used for each direction of communication
  - \(MAC_{K_{AB}}(M)\) : computation of message authentication code (MAC) of message \(M\) with MAC Key \(K_{AB}\)
Ariadne
(Basic Ariadne Route Discovery)

- Protocols

Source

\[ h_0 = MAC_{K_{SD}}(REQ|S|D|id|ti) \]

\[ \langle REP|S|D|id|ti|(A,B),(M_A,MB)|MD,(K_{Bti}, K_{Ati}) \rangle \]

\[ h_1 = H[A,h_0] \]

\[ M_A = MAC_{K_{Ati}}(REQ,S,D,id,ti,h_1,(A),()) \]

\[ \langle REP|S|D|id|ti|(A,B),(M_A,MB)|MD,(K_{Bti}) \rangle \]

\[ h_2 = H[B,h_1] \]

\[ M_B = MAC_{K_{Bti}}(REQ,S,D,id,ti,h_2,(A,B),(M_A)) \]

\[ \langle REP|S|D|id|ti|(A,B),(M_A,MB)|MD,(K_{Bti}) \rangle \]

\[ M_D = MAC_{K_{DS}}(REP,D,S, ti,(A,B),(M_A,M_B)) \]
Ariadne
(Basic Ariadne Route Maintenance)

- Protocols: securing route error msg.

\[(RE|B,D|t_i+d|MAC K_{B_{t_i+d}}(RE|B,D|t_i+d)|K_{B_{t_i}})\]

\[(RE|B,D|t_i|MAC K_{B_{t_i}}(RE|B,D|t_i)|K_{B_{t_i-d}})\]

Store it until $K_{B_{t_i}}$ receives

Verify MAC and remove the path from routing cache
Ariadne Evaluation

- **Performance Evaluation**
  - **Parameters**
    - **Scenario**
      - Number of nodes: 50, Maximum velocity: 20 m/s
      - Space: 1500 m * 300 m, Nominal radio range: 250 m
      - Source-destination pairs: 20, Source data pattern: 4 packets/sec
      - Application data payload size: 512 bytes/packet
      - Total application Data Load: 327 kbps
      - Raw physical bandwidth: 2 Mbps
    - **DSR**
      - Initial route request timeout: 2 sec, Maximum route request timeout: 40 sec
      - Cache size: 32 routes Cache, replacement policy: FIFO
    - **TESLA**
      - TESLA time interval: 1 sec
      - Pessimistic end-to-end propagation time: 0.2 sec
      - Maximum time sync. error: 0.1 sec
      - Hash length: 80 bits
Ariadne Evaluation (Cont.)

- **Moves according to random way point model**
- **Compares DSR, Ariadne, DSR with no optimization**

![Graph showing packet delivery ratio over pause time for DSR, Ariadne, and DSR-NoOpt models.]
Ariadne Evaluation (Cont.)

(b) Packet Overhead

(c) Byte Overhead
Ariadne Evaluation (Cont.)

(d) Path Optimality

(c) Average Latency
Ariadne Evaluation (Cont.)

• Security Analysis
  - Ariadne guarantees
    • If destination has uncompromised neighbor, it will return route reply
    • If at least one route reply returned to source, Ariadne can route packets along uncompromised route
  - Preventing attacks
    • Message Authentication Code with hop-by-hop hashing
    • TESLA maximum end-to-end delay feature
    • TESLA hash-chaining feature
Denial of Service Resilience in Ad Hoc Networks

I. Aad, J. Hubaux and E. Knightly
EPFL, Switzerland and Rice University
DoS Attacks

- Attackers prevent legitimate users from getting served

- Common DoS schemes
  - Manipulating lots of traffic
  - Pro: effective regardless of upper layer protocols
  - Con: easy to detect
Black Holes

• BH participate in all routing control operations
• Establish routes through themselves
• Once path established, BH drop all data packets
• How to detect black holes?
Black Hole Detection

- Watch if the next hop forwards the packets
- Challenges?
Black Hole Detection

- **Watch if the next hop forwards the packets**
- **Challenges**
  - **Dynamic power control**
    - False positive: heard by next hop but not previous hop
    - False negative: heard by previous hop but not next hop
  - **Directional antennas**
    - False positive: heard by next hop but not previous hop
    - False negative: heard by previous hop but not next hop
  - **Detection timescales**
    - Single packet loss implies problematic route
    - Large number of packet losses implies problematic route but more traffic is affected
How to increase its damage?
Increase the Damage of Attacks

- **Attract more traffic that is affected**
  - **Rushing attacks**
    - If attackers attract twice as many flows compared with uniform graph (2a/N instead of a/N), flow goodput drops from 52% to 34% with 10% attackers
  - **Mobile JF and BH attackers**
    - Mobile attackers moves around to attain an optimal position that affects a large amount of flows passing through it
Stealthy DoS schemes

- Can attackers launch DoS by manipulating a small amount of traffic?
  - Harder to detect
Stealthy DoS schemes

- Can attackers launch DoS by manipulating a small amount of traffic?
  - Harder to detect

- Jellyfish
  - Cross-layer attack
  - Exploit the feedback-based protocol (TCP)
  - Types of attacks
    - JF reorder attack
    - JF periodic dropping attack
    - JF delay variance attack
JellyFish Reorder Attack

• JF nodes
  - deliver all packets
  - after placing them randomly in a FIFO buffer

• Results in near-zero goodput despite delivering all packets
  - Hard to detect because no packet dropping
JellyFish Periodic Dropping Attack

- Attackers drop all packets for a short period of time once per retransmission time-out (RTO)
  - Effect: consecutive packet losses → TCP timeout at the victim flow
- When the flow attempts to exit timeout RTO seconds later, JF will soon/immediately drop again
- Hard to detect
  - Effective even when dropping only a small fraction of packets
Primary effect on throughput:

- To obtain the null at 1 second, the JF drops packets for 90ms every 1 second
  - Dropping 9% of the time and forwarding 91% of the time
  - Hard to detect because these are values that can be incurred by a congested node

- Multiple packet losses within a RT-time are an indication of severe congestion
  - Flow must back off aggressively and wait RTO seconds before entering slow start

![Graph showing goodput over time for different hop counts]
JellyFish Delay Variance Attack

- JFs manipulate packet delays to reduce TCP throughput
- This results in
  - TCP sending traffic in bursts due to “self-clocking”, leads to increased collisions and loss
  - Incorrect estimations of available bandwidth
  - Excessively high RTO value
JF-jitter effect on throughput

- JF alternates between periods of serving packets at its maximum capacity and serving no packets
- Idle and active periods are of equal length
  - TCP goodput decreases with increasing jitter
JellyFish vs. Black Holes

- JF has nearly same impact as BH
- JF only works for TCP flows, while BH works for both TCP and UDP
- JF is much harder to detect than BH
How to respond?
Victim’s response

• Once malicious nodes are detected there are three solutions:
  - Establish new path excluding any node from prior malfunctioning path
    • difficult to achieve in small/sparse networks!
  - Employ multipath routing and adapt path weights according to path goodput
    • severely decreases throughput under TCP
  - Establish backup routes by keeping all route reply messages
Summary

- **DoS attacks and defenses**
  - Open loop protocol: affect a large amount of traffic (e.g., blackhole)
  - Closed-loop protocol: don’t need to affect a large amount of traffic (e.g., jellyfish)