Fragmentation Considered Vulnerable

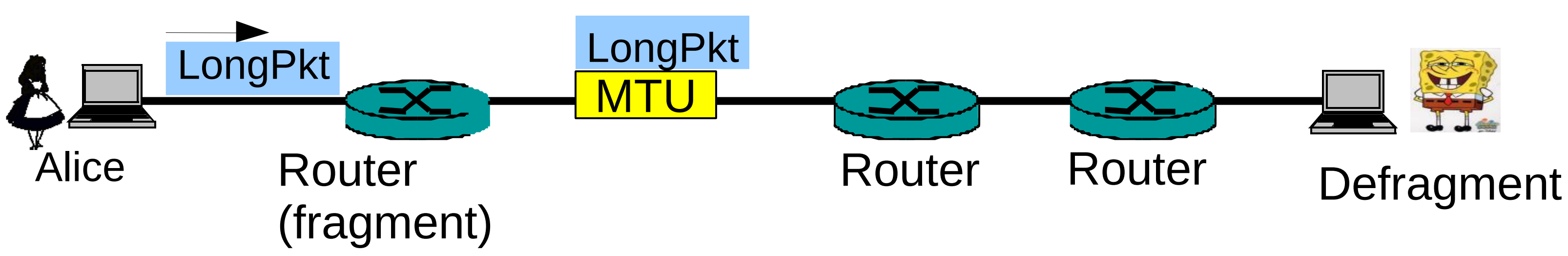
Yossi Gilad & Amir Herzberg
Computer Science Department, Bar Ilan University

Overview

- IP fragmentation recap
- `Easy case' fragment miss-association attacks
- Fragmentation attacks in tunnels
- Conclusions

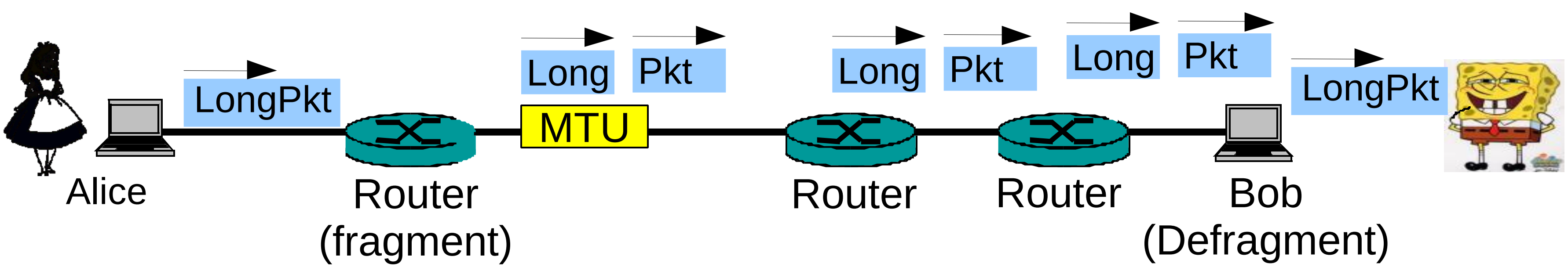
IP Fragmentation - Recap

- Today: attacks on IP fragmentation
  - Blind (spoofing only) attacker
  - Interception and DoS attacks.
- The Internet is a diverse network
  - Different Maximal Transmission Units (MTUs) on different links/nets
- What if |long-pkt|>MTU?



IP Fragmentation - Recap

- Solution 1: Path MTU discovery (PMTUd)
  - Discard oversized pkt, inform sender (via ICMP)
  - Requires connection
- Solution 2: IP fragmentation
  - 'Break' long pkt into fragments (|frag|<MTU)
  - Fragment at: any node (IPv4) / only src (IPv6)
  - Defragment: only at destination
    - According to: source, destination, protocol & frag ID



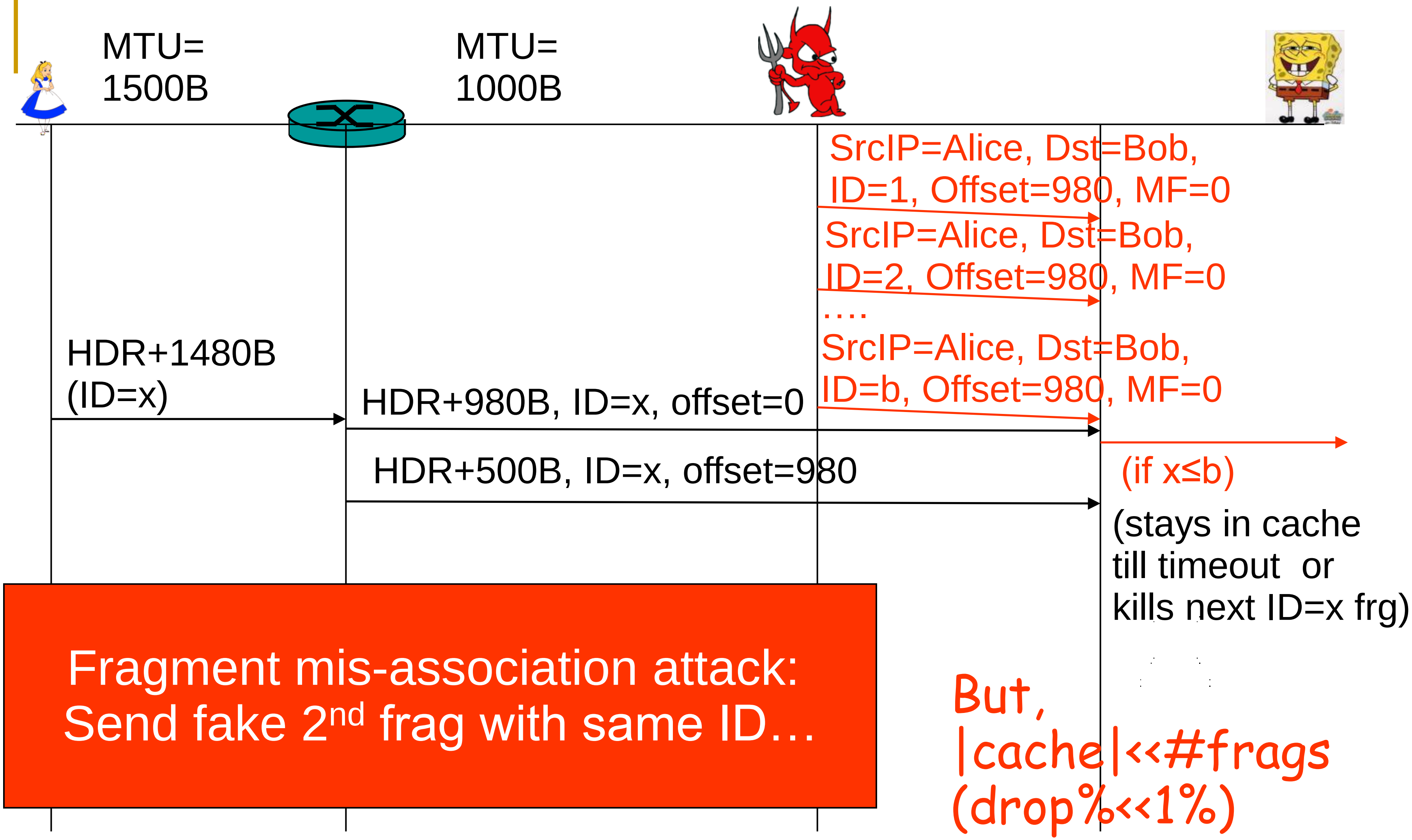
‘Fragmentation considered Harmful’

- IP fragmentation is conceptually easy, but…
  - Wasteful/harmful [KentMogul87]
  - Complexities: may arrive late or out of order, overlap
  - How much storage? How long keep fragments in cache?
- But: still often used
  - PMTUd often fails (for UDP, no ICMP feedback,…)
  - Fragmentation is common in UDP and tunneled traffic [Shannon02]

‘Fragmentation considered Harmful’

- Implementation vulnerabilities:
  - Memory allocation DoS attacks: TearDrop, Rose…
  - Tiny fragment evasion of firewalls
- Specification vulnerabilities:
  - Fragment cache overflow attack [KPS03]
  - Zalewski (2003) notes that fragmented TCP traffic can be vulnerable to (blind) TCP injections
  - Fragment mis-association attack [M04,rfc4963]

Fragment Misassociation Attack

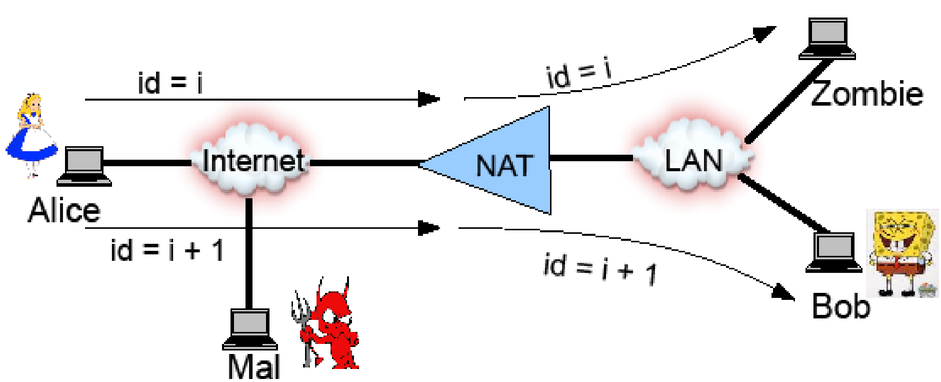


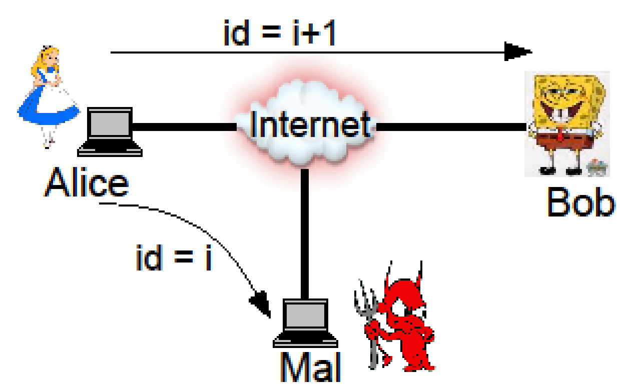
What if…

- Frag mis-association has low drop rate
- What if attacker can *find* the `next' ID?
  - Trivial to `kill' packet (DoS)
  - Can also `inject' a fragment
    - Need to fix checksum
      - Checksum can be disabled for UDP
- How is the IP ID chosen (by the sender)?
  - Usually a counter – this is specifically recommended by IPv6 specification
  - Two main approaches:
    - Global counter (Windows)
    - Per-destination counter (Linux)

Sometimes, ID Exposing Is Easy

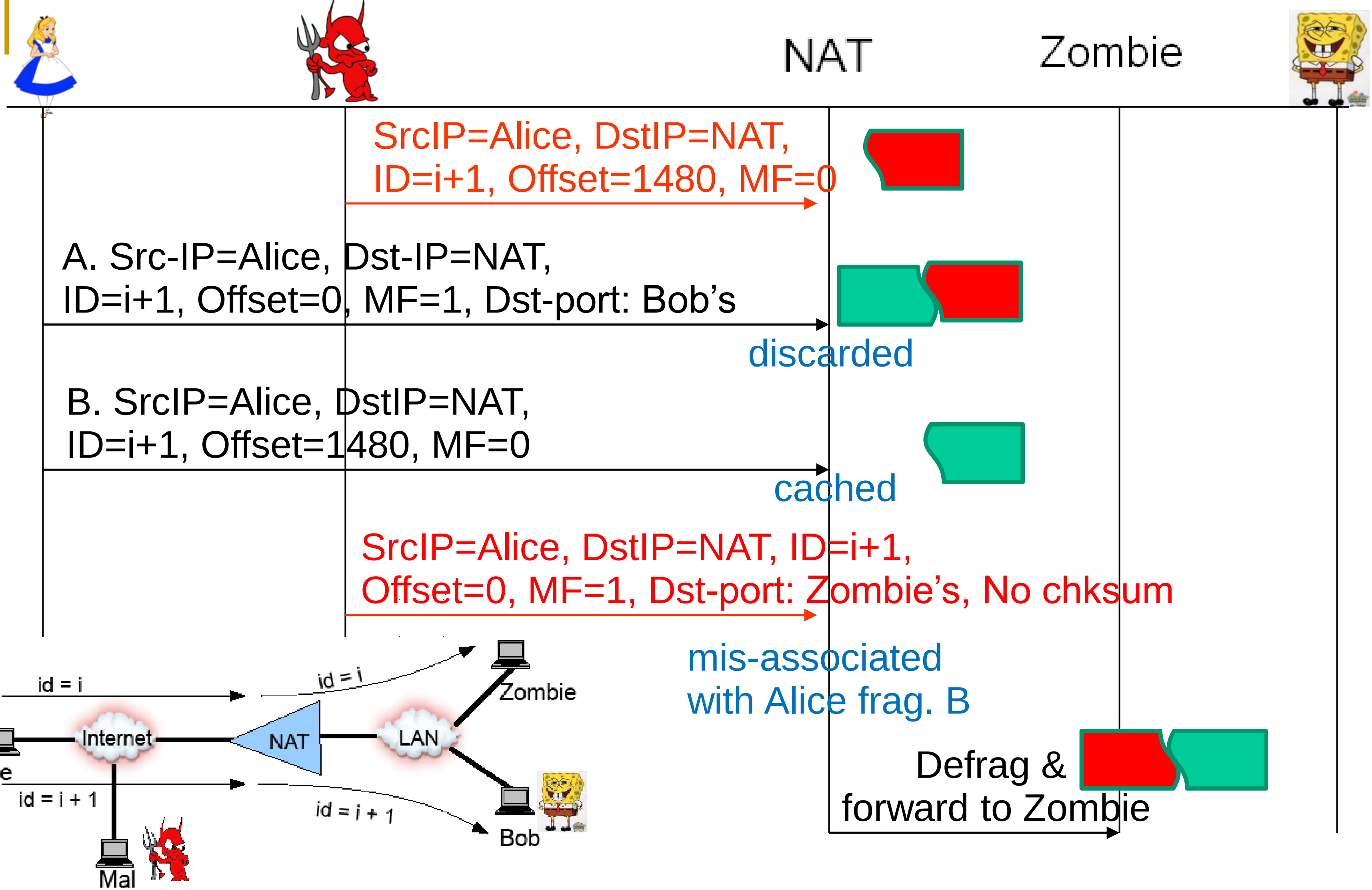
- When the sender uses a global identifier
  - Just by observing any packet from the sender
- When the attacker has a zombie behind the NAT with the destination
  - Can also intercept fragments!
    - Rewrite transport layer header





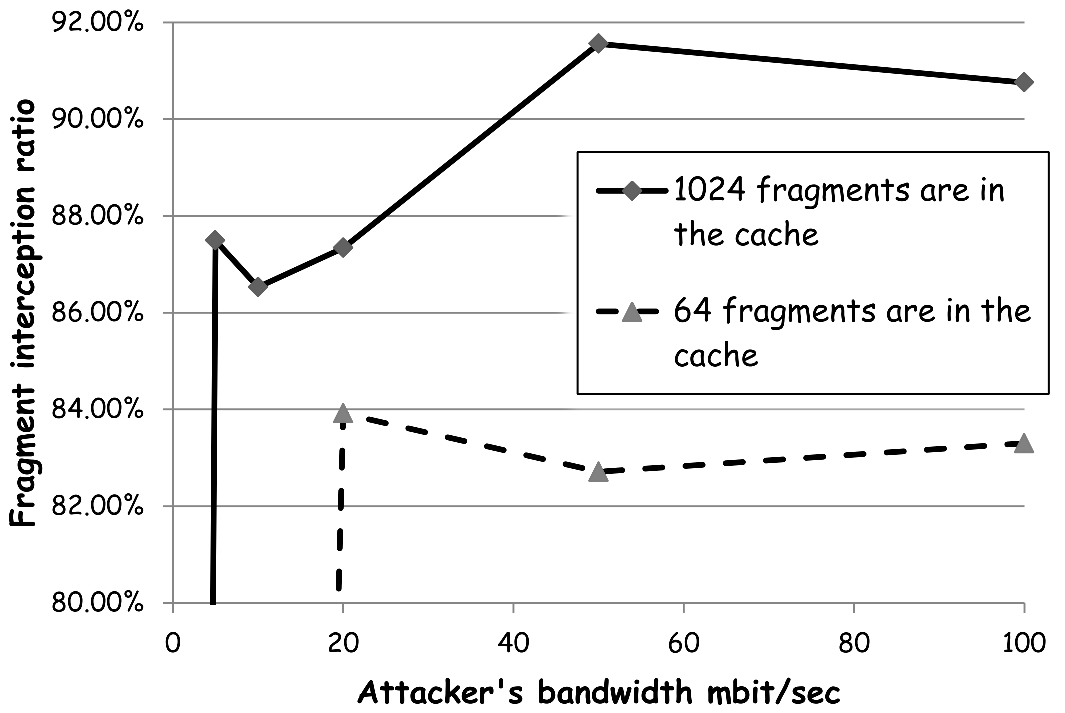
Sometimes, ID Exposing Is Easy

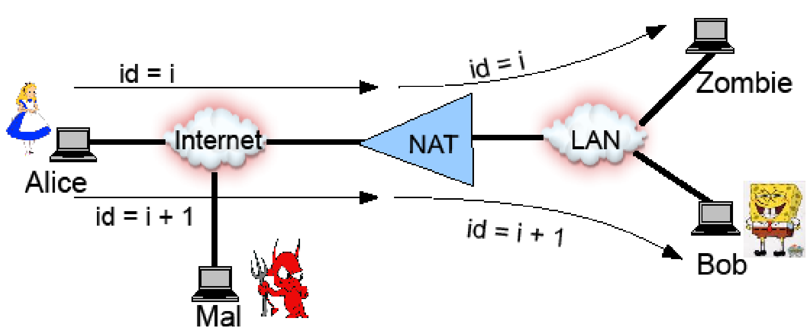
Intercepting fragments



Fragment Interception: Results

- Results for IP tables based NAT



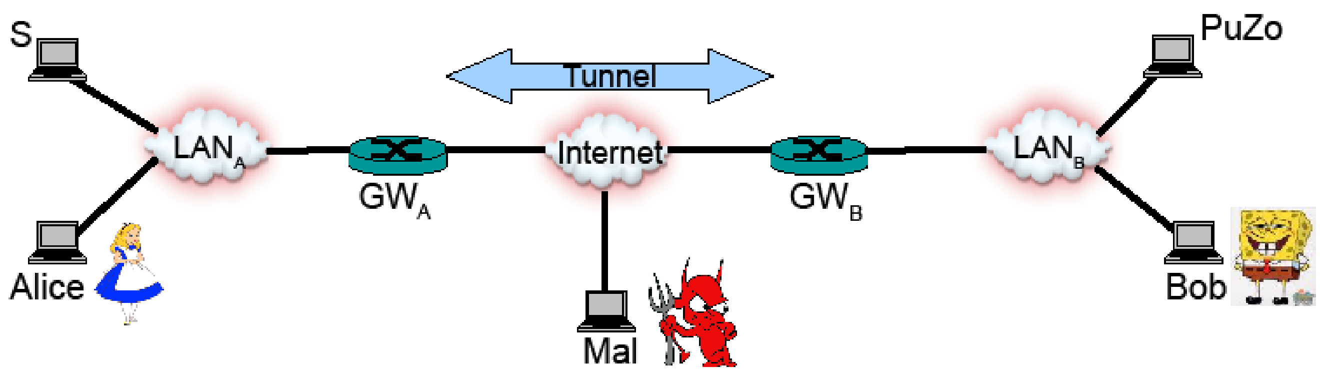


Other Cases?

- Can similar attacks apply when sender uses per-destination IP-IDs?
  - Easy: if there is NAT (shown before)
- What if there is no NAT?
- Yes!
  - For a tunnel scenario

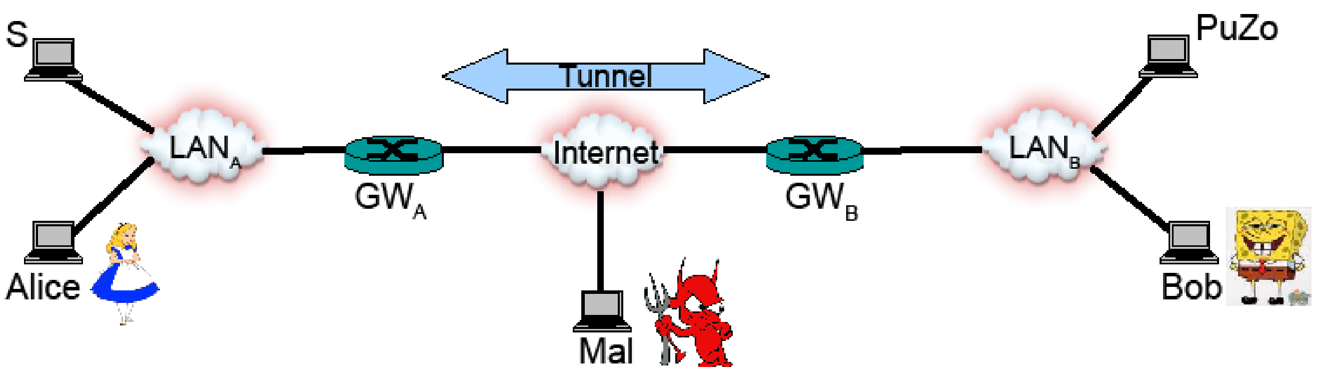
ID Exposing Attack

- Alice and Bob are connected via a tunnel
- Main difference from NAT scenario:
  - Packets `on the Internet' have a different IP header
    - Adversarial agent, PuZo, can not `see' the `Internet' ID
- Improved motivation: fragmentation is common in tunnels
- In talk: Zombie (to receive raw IP packet)
  - In paper: Puppet (running in sandbox)

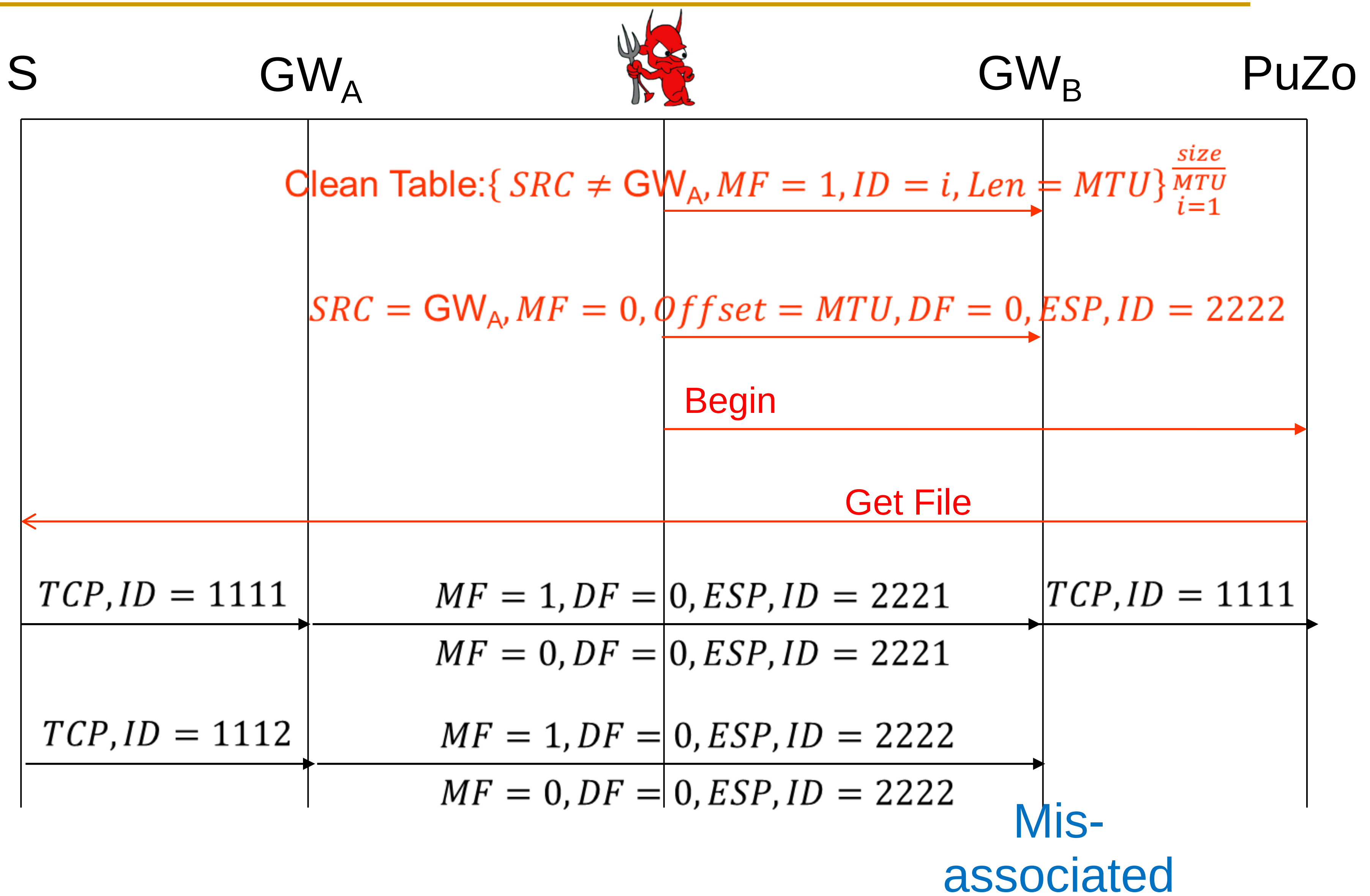


ID Exposing Attack

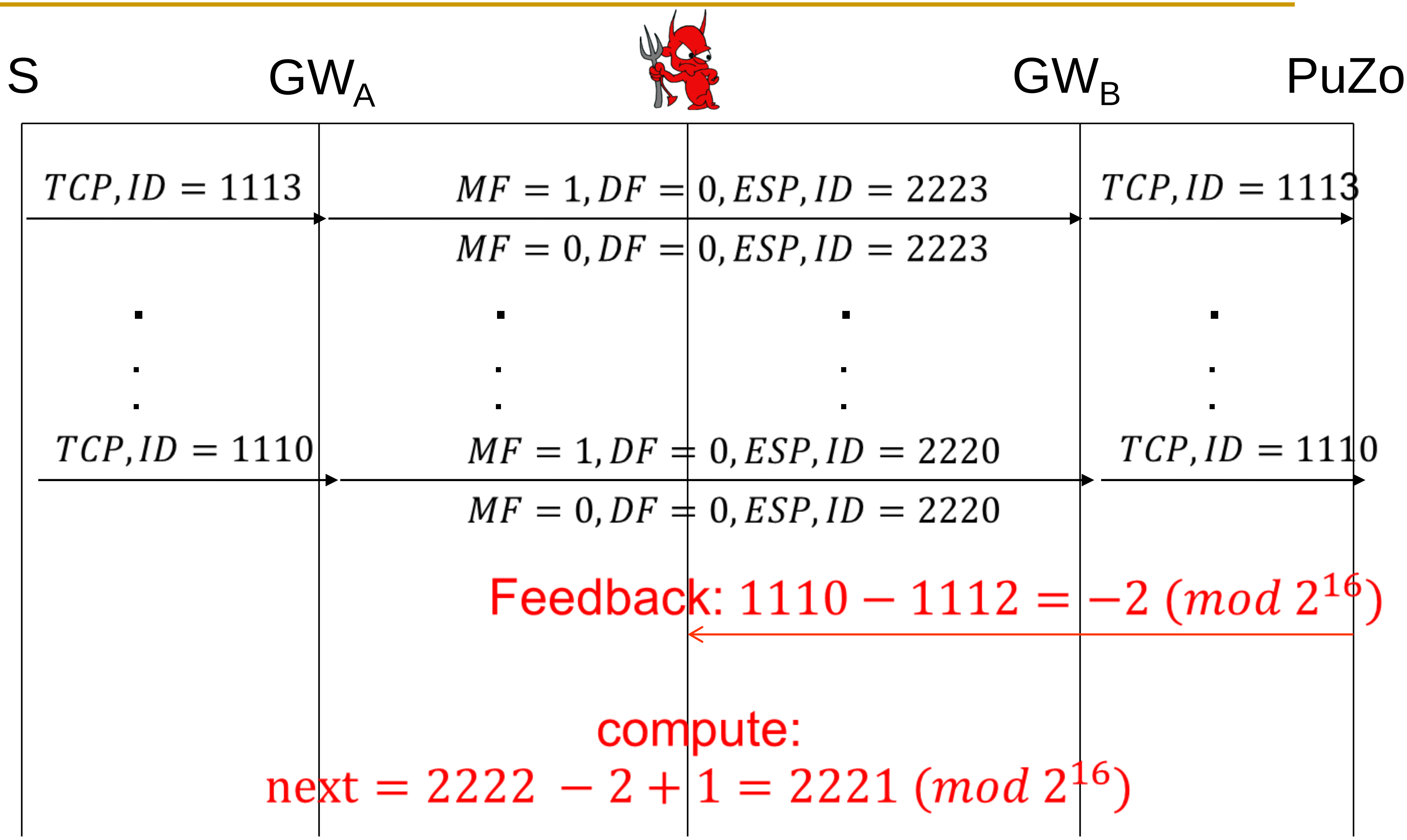
- Use packet loss as a side channel to identify the current ID within the tunnel
- We assume no benign traffic or packet loss
  - Full version shows how to deal with those



ID Exposing Attack – Basic Version

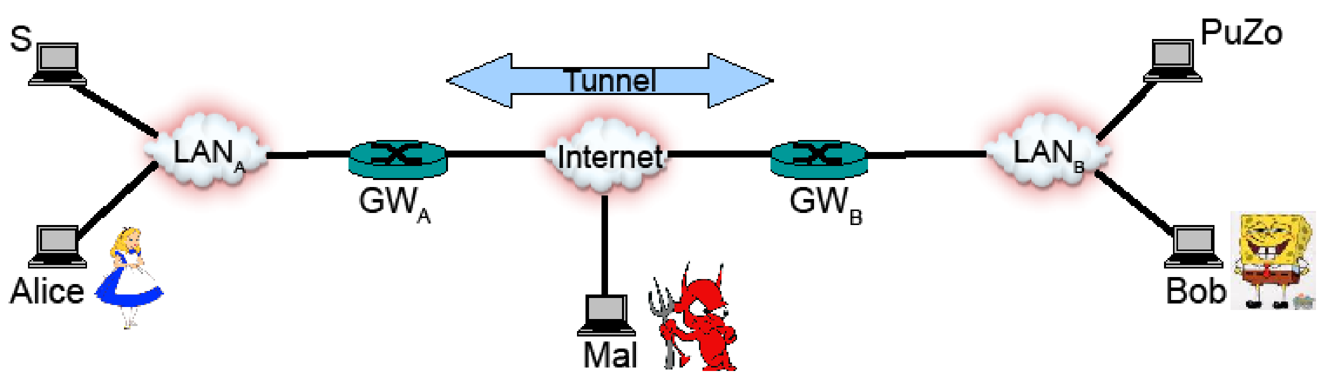


ID Exposing Attack – Basic Version



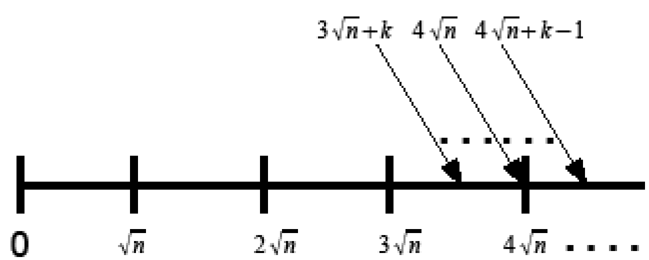
ID Exposing Attack - Meet in the Middle

- But… if n is the number of possible identifiers, this attack requires to send O(n) packets.
  - 2^{16} for IPv4, for 2^{32} IPv6
- Revise with meet in the middle technique



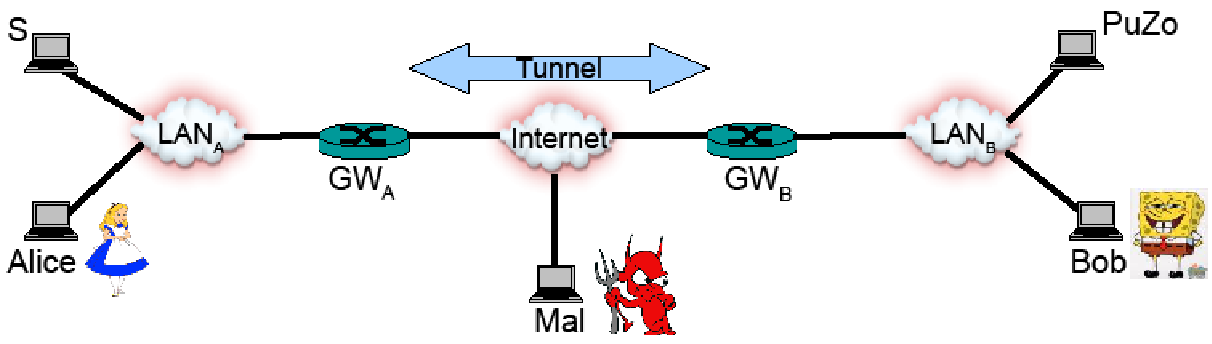
ID Exposing Attack - Meet in the Middle

- Send $\sqrt{n}$ fragments → lay $\sqrt{n}$ traps
- Narrow the search space to $\sqrt{n}$
  - Detect loss → assume `ID hit' (frag. mis-association)
- Exhaustive search over all remaining IDs
- Reduced number of packets to O($\sqrt{n}$)
  - Also feasible for IPv6 (n = 2^{32})



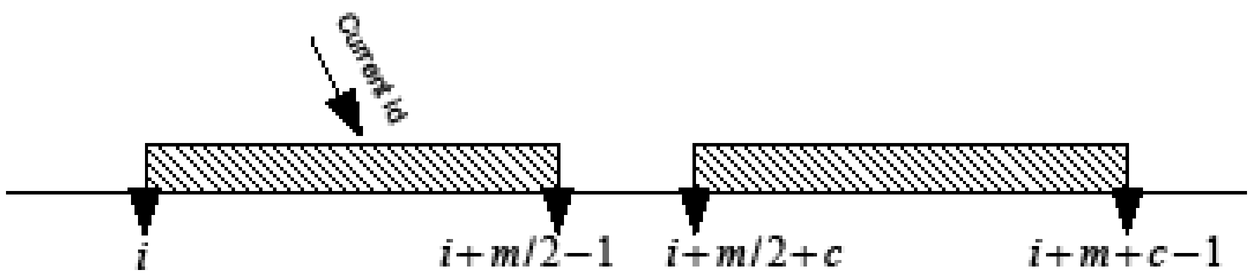
Continual Deny & Expose

- Mal has the current ID
  - Goal: deny fragmented traffic
- Main Difficulty: maintain synchronization with current IP ID
  - Incremented for every packet (regardless of arrival/loss)



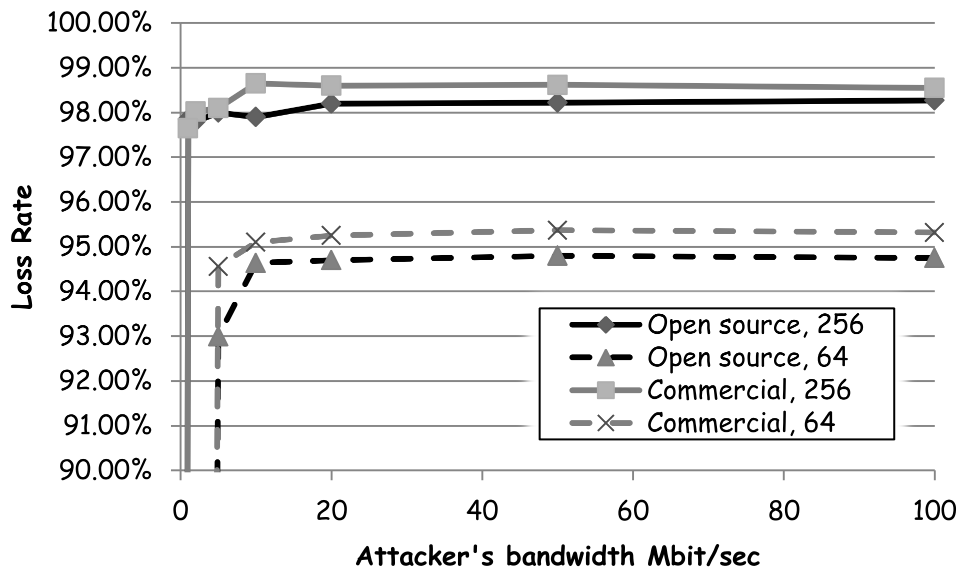
Continual Deny & Expose

- Basic idea: use PuZo to `monitor' IP ID progress
  - Send two sequences of spoofed fragments with consecutive IDs
  - Small `gap' of unset IDs between them
  - PuZo makes a periodic request for data
  - Response arrives → ID within the gap
  - Send the next sequence



Continual Deny & Expose - Results

- Success depends on the number of forged fragment attacker can `cache in'
  - Usually 64 or no limitation (except cache size, 6500+)



Conclusions

- Fix IP ID
  - Add appropriate defenses to network firewalls, IDS/IPS
- Need to improve specification of networking protocols
  - Need to develop validation techniques
- Further motivation for [Gont11]

Questions?