TDC690
Single-Packet IP Traceback

Authors: Alex Snoren, Craig Partridge, Luis Sanchez, Christine Jones, Fabrice Tchakountio, Beverly Schwartz, Stephen Kent, W. Timothy Strayer

IEEE/ACM Transactions on Networking Vol 10, No 6, December 2002

Graphic References: Jessica Kornblum DSL Seminar 2001

Reviewer: J. Elarde

Agenda

- Introduction
- IP Traceback
- Related Work
- Packet Digesting
- Source Path Isolation
- Practical Implementation
- Analysis and Discussion
- Summary/Critique

Introduction

Problem

- Today's Internet is extremely vulnerable to hackers.
  - DDOS and Single Packet(Teardrop) attacks.
- The IP protocol design does not support reliable identification of the originator.
  - Beyond deliberate attempts, widespread packet forwarding techniques such as NAT and encapsulation also can obscure origin.
- No system to-date can track single packet in efficient and scalable fashion.

Authors Contributions

- The authors present a hash based Source Path Isolation Engine (SPIE) to enable IP traceback:
  - Generates audit trails for traffic within the network.
  - Can trace the origin of a single packet in the delivered by the network in recent past.
  - Analytical and simulation results presented.

IP Traceback

IP Traceback - Assumptions

- Packet may be addressed to more than one host
- Duplicate packets can exist
- Router may be subverted but not often
- Attackers are aware of the monitoring
- Routing behavior may be unstable
- Packet size should increase as the result of tracing
- End hosts may be resource constrained
- Traceback is infrequent

IP Traceback - Goals

- Identify source of any piece of data delivered by the network.
  - Construct an “Attack Path”.
- Possible origins:
  - The Ingress point to the traceback enabled network
  - Actual host
  - One or more compromised routers
- Privacy must not be compromised.
- Robustness: Limit the false positives, and no false negatives.

IP Traceback and Transformations

- Packets may be modified (transformed) as part of the normal forwarding process.
- Examples:
  - TTL decrementing
  - Encapsulation
  - Router processing ICMP Echo, IP Multicast, Fragmentation, IP option processing
  - Network Address translation, IPsec tunneling.
- CAIDA study < 3% of wide-are traffic undergoes transformation.

Related Work

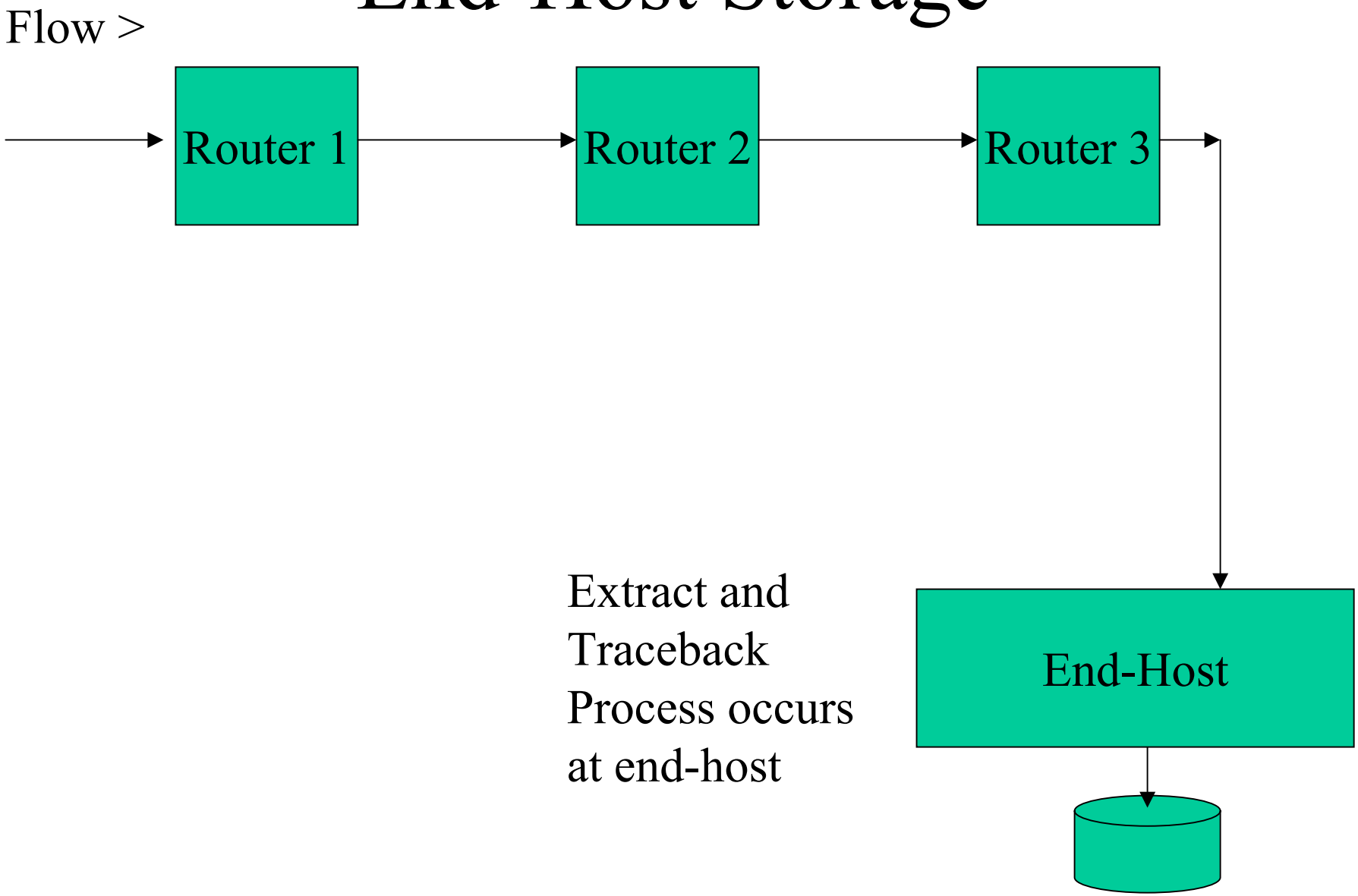
Approaches to Traceback

- Audit the flow as it traverses the network.
  - End-Host Storage auditing techniques
  - Infrastructure Approaches
  - Specialized routing
- Infer route based upon its impact on the state of the network.
- As size of flow decreases difficulty increases.

Auditing Techniques

- End-Host Storage:
  - Distribute burden of storing state information and performing computations at end-hosts.
  - Savage et al. and Bellovin explore in-band and out-of-band signaling respectively to accomplish this.
    - Not every packet traced only subset of flow.
    - Auditing routers provide information to end-host to reconstruct route.
    - Savages et al. uses a packet marking scheme to encode and communicate information to the end-host.
    - Bellovin sends audit information via ICMP to the end-host.

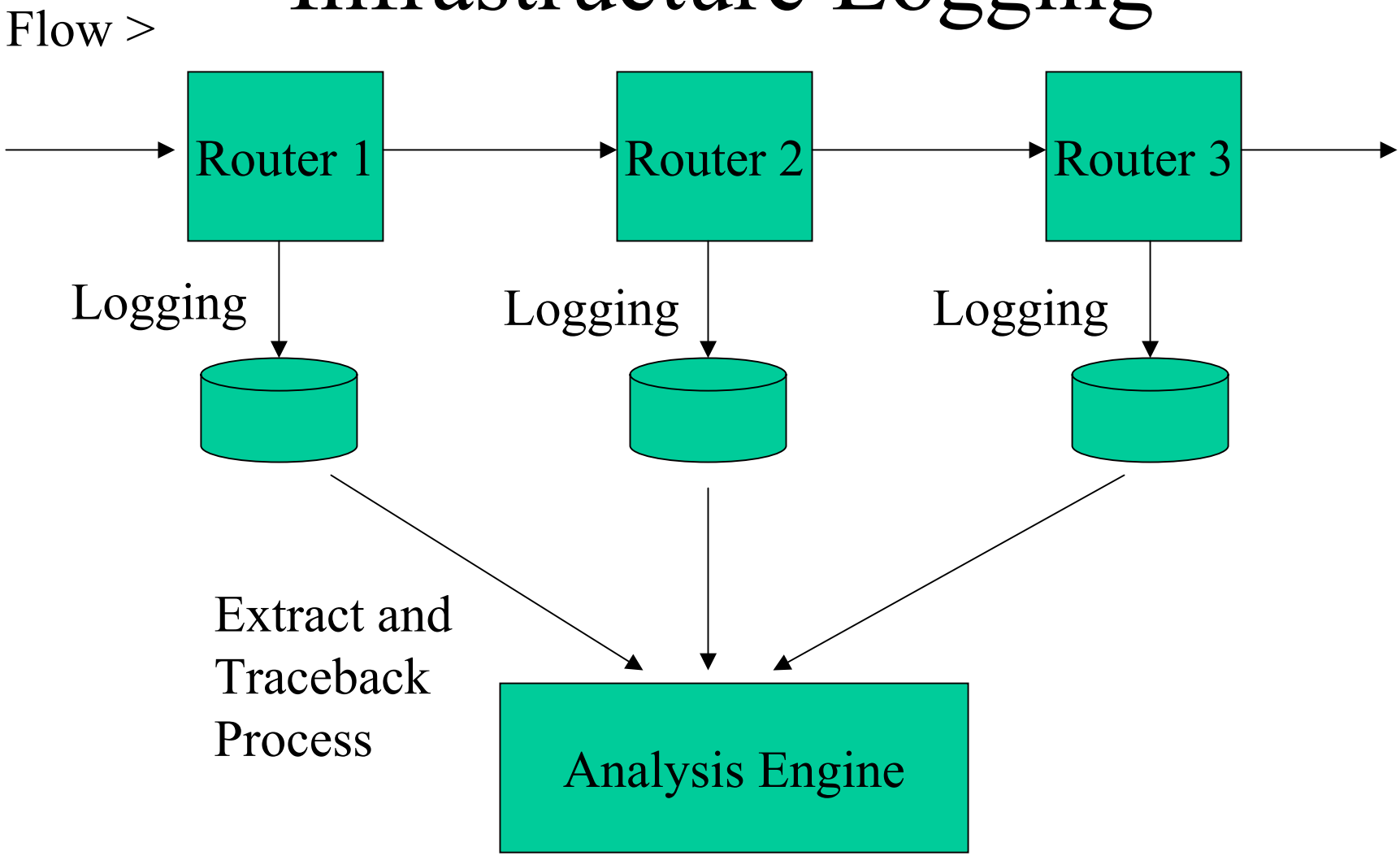
End-Host Storage



Infrastructure Approaches

- Logging Method: Log packets at points throughout the network and use extraction techniques to reconstruct route. (Sager)
  - Problem: log size and storage – OC-192, 60 seconds, 16 links = 1.2TB
- Sampling reduces effectiveness and Privacy problem exists.

Infrastructure Logging



Specialized Routing

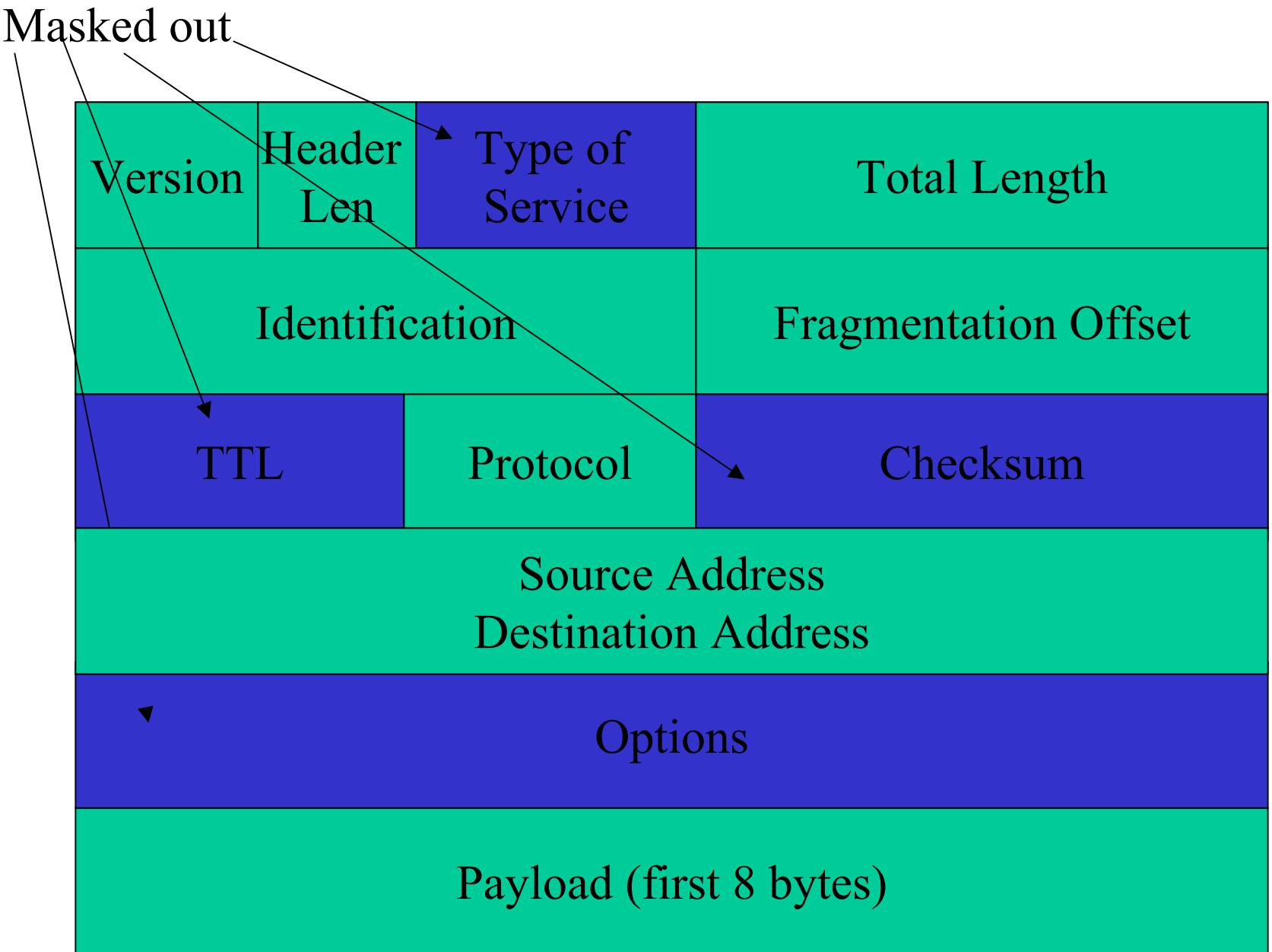
- The logging method traceback extraction is expensive and repetitious across each hop.
- Techniques have been developed to streamline and automate the process.
  - ISPs have develop ad hoc methods of conducting input debugging across their networks.
  - (Schnackenberg et al.) propose a generalized Intruder Detection and Isolation Protocol (IDIP) to facilitate interactions among routers during traceback.
  - (Stone) suggests constructing an overlay network so all routers do not need to support logging.

Packet Digesting

Packet Digesting

- SPIE implements an auditing technique while reducing storage requirements significantly.
- Auditing is accomplished by computing and storing a packet digest.
- Privacy is maintained.

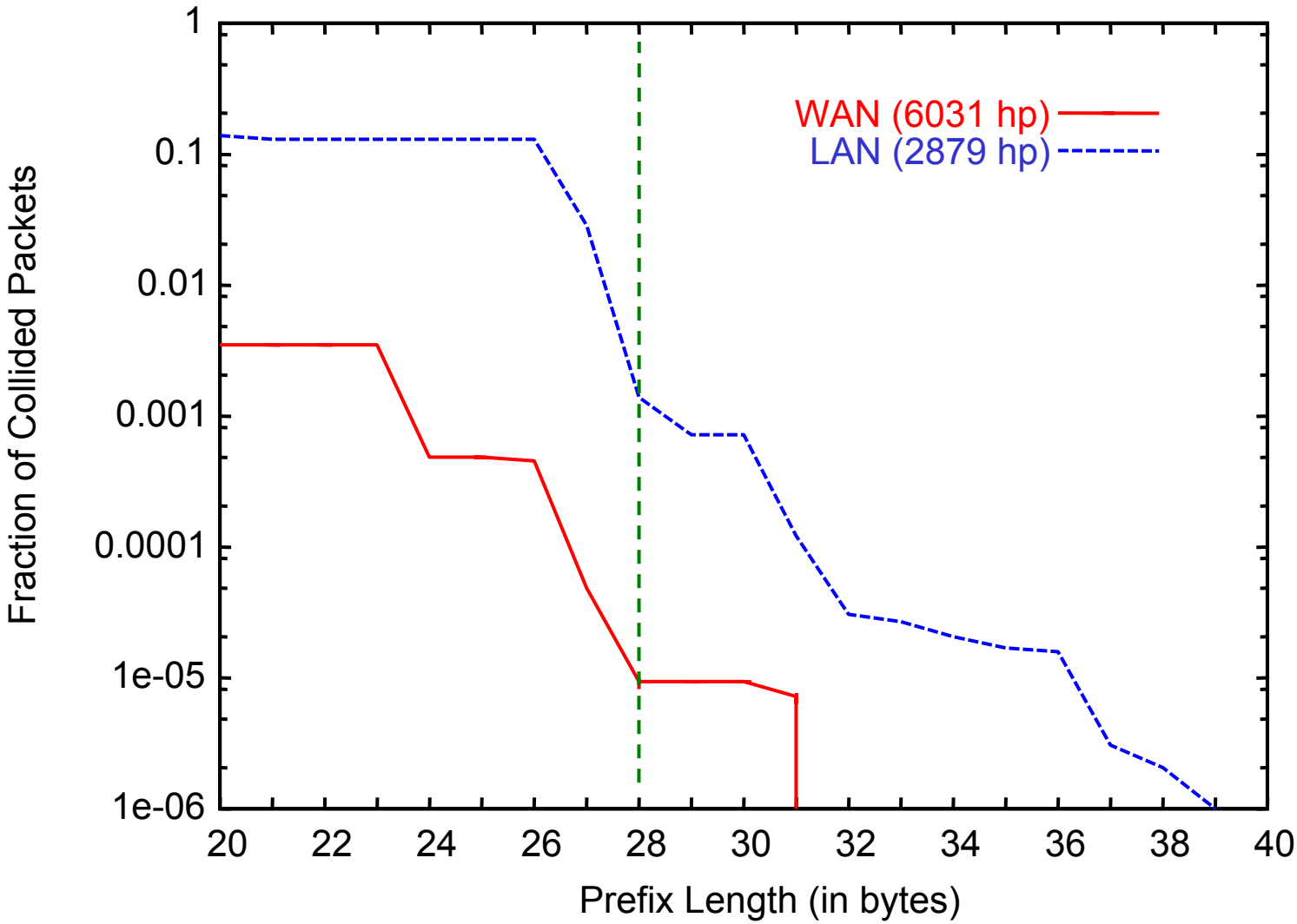
Digest Input



Digest Input

- 20 Bytes header with 4 bytes masked and 8 bytes of payload are sufficient to identify duplicates.
- Collision Rates
  - LAN .139%
  - WAN .00092%
- Most collisions are ICMP or packets with IP Identification field set to zero.
- Higher collision rate on LAN is due to the lack of address diversity.

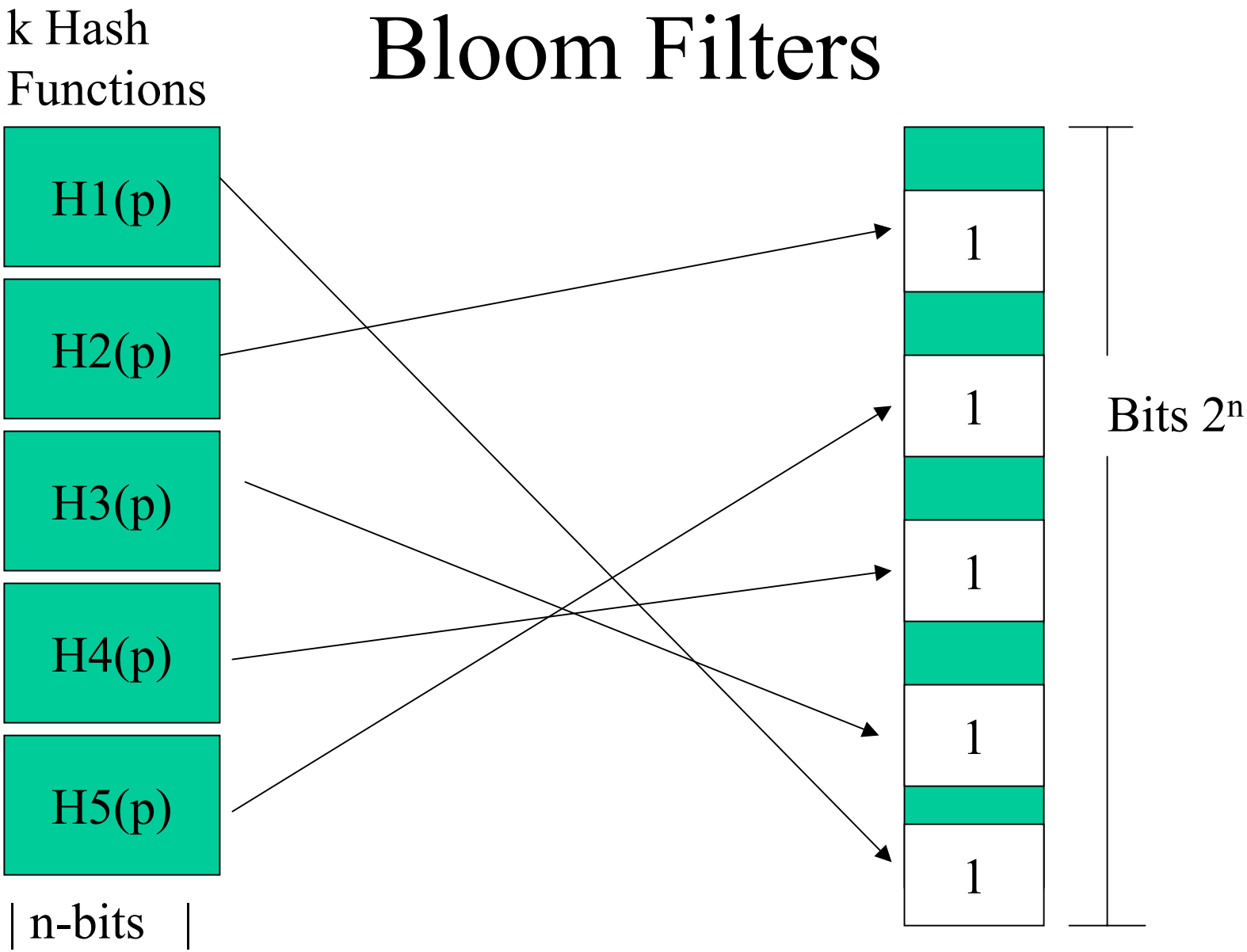
Collisions Vs. Digest Input



Bloom Filters

- To reduce storage requirements further, Bloom filters are used to store the data.
- A Bloom filter computes k distinct packet digests for each packet using a hash function.
- And then uses the n-bit results to index into an array 2^{n} –bit array.
- If any bit is zero then the packet is not stored in the table.
- If all ones, it is likely the packet was stored.

Bloom Filters



Hashing Requirements

- Each hash function
  - Uniform distribution of input -> output

    H1(x) = H1(y) for some x,y -> unlikely
- Use k independent hash functions
  - Collisions among k functions independent
  - H1(x) = H2(y) for some x,y -> unlikely
- Compute at high speed. Digests must be archived and cleared at interval t.

Source Path Isolation Engine

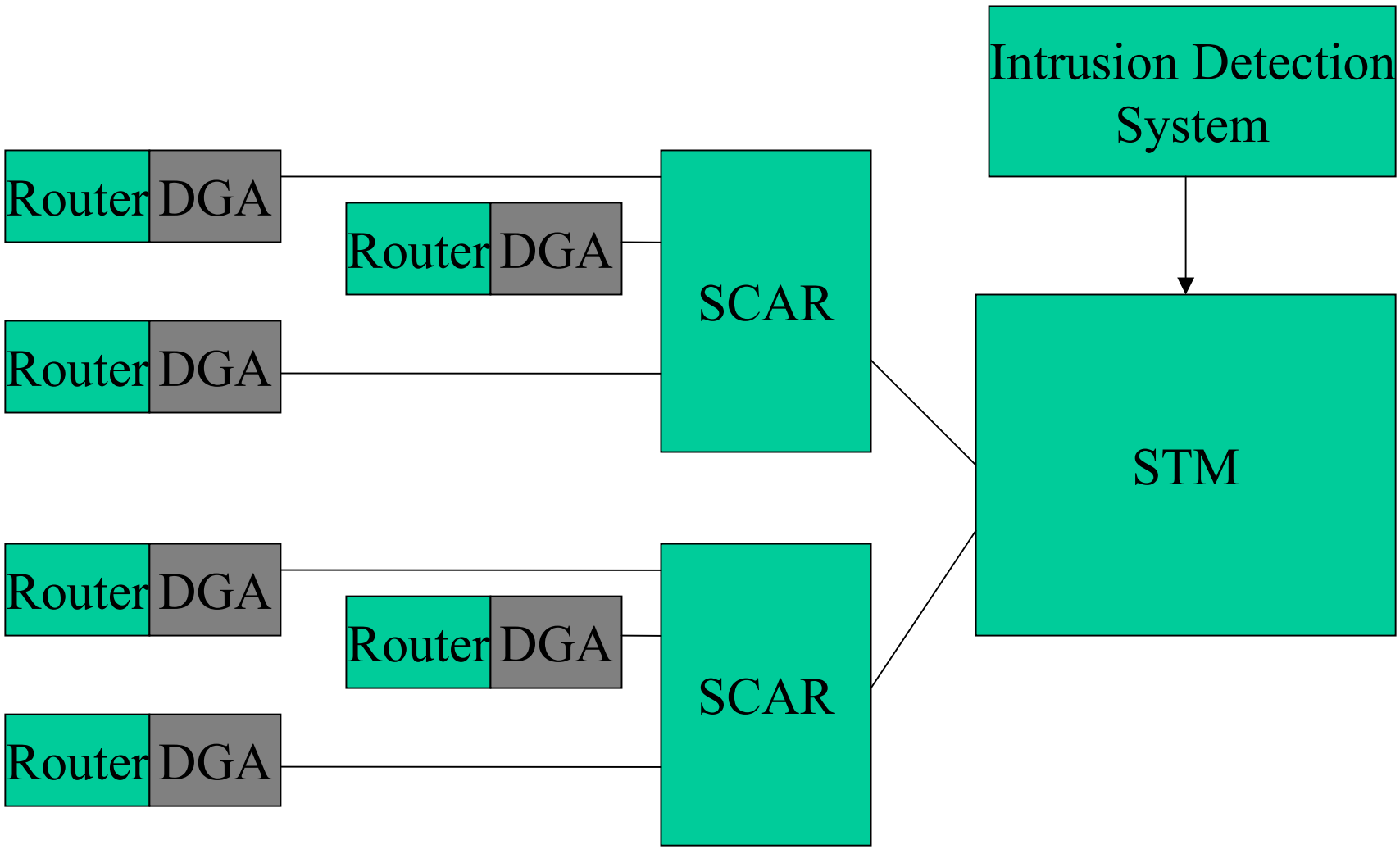
SPIE Process

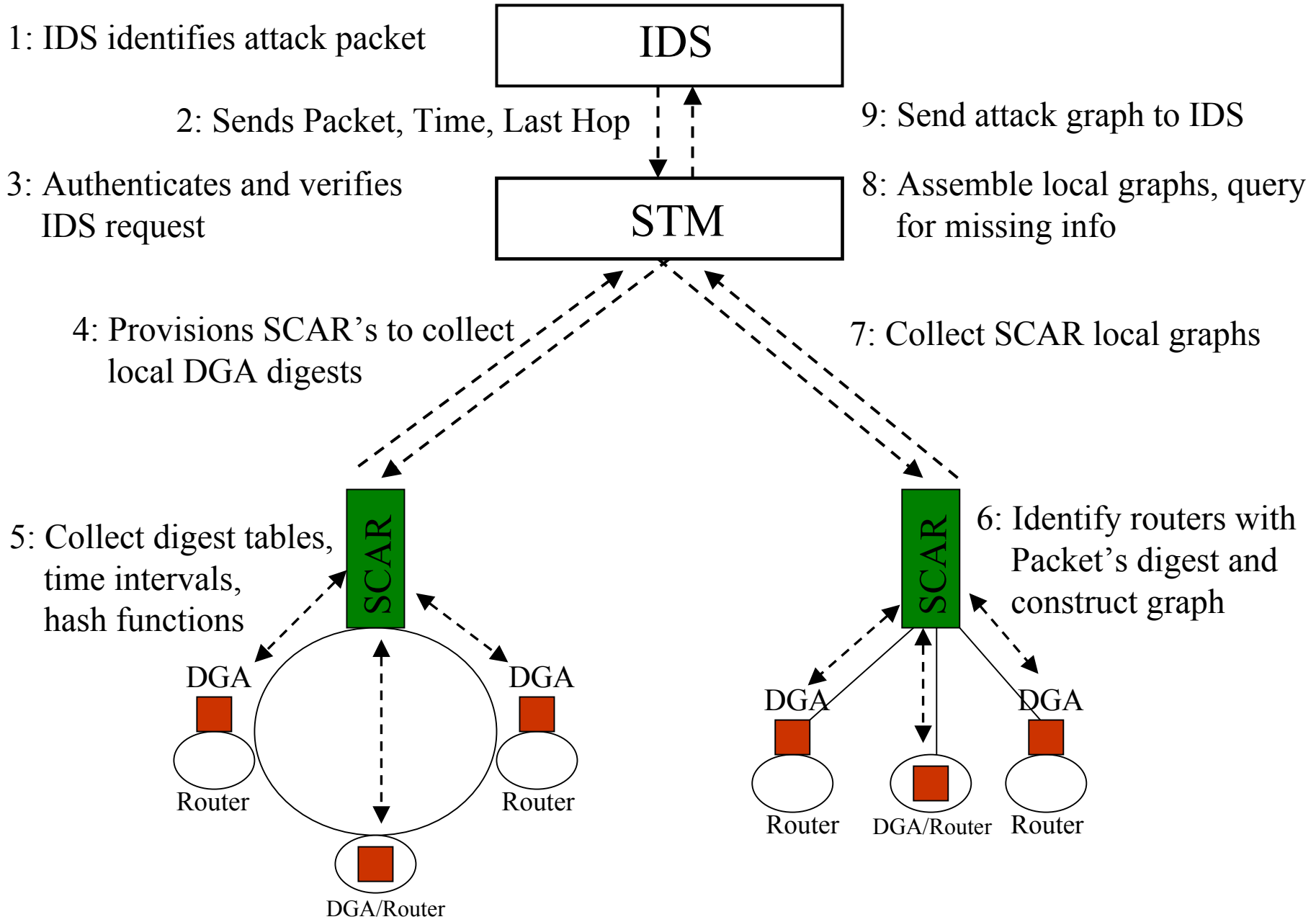
- SPIE routers maintain a cache of packet digests for recently forwarded packets.
- If a packet is determined to be offensive, a query is dispatched to the SPIE.
- The SPIE queries routers for packet digests of relevant time periods.
- The results are used to build an attack graph.

SPIE Architecture

- Data Generator Agent (DGA)
  - Stores digests in a time stamped table
  - Periodically pages out portions of the table.
  - Integrated into router or outboard box monitoring router output.
- SPIE Collection and Reduction Agents (SCARs)
  - SCARs monitor a region of the network
  - Produce an attack graph periodically.
- SPIE Traceback Manager
  - Controls process and is linked to Intrusion detection system
  - Dispatches request to SCARS
  - Collects results and assembles complete attack graph.

SPIE Architecture





Source: Jessica Kornblum

Handling Transformations

- The SPIE must handle fragmentation, Network address translation, ICMP, IP Tunneling.
- A Transformation Lookup Table (TLT) is maintained to reconstruct the original packet.
- The TLT consists of the transformed digest, type flags, and the changed packet data.
- NAT and tunneling handled by a standard rule set due to volume.

Practical Implementation

SPIE Prototype

- The authors constructed a PC based SPIE prototype and used MD5 for the hash functions.
- The MD5 algorithm takes as input a message of arbitrary length and produces as output a 128-bit "fingerprint" or "message digest" of the input.
  - It is conjectured that it is computationally infeasible to produce two messages having the same message digest, or to produce any message having a given pre-specified target message digest.
- The 128 bit result is separated into 4 independent digests for the Bloom filters.

Analysis and Discussion

Analysis

- Effectiveness is dependent upon:
  - Length of time the digest is retained.
  - The accuracy of the attack graph - fewer false positives.
- Both can be controlled by adjusting the amount of memory.
- Authors use an analytical model to estimate the false positive upper bound to be 5 nodes in 35 – expected to be substantially less in practice.
- Simulation study performed to research probability of false-positive reporting nodes.

Simulation Configuration

- Ran a simulation using an ISPs (70 node T1-OC3) actual network topology and sampled link utilization.
- Simulated attack by randomly selecting a source and victim and generating 1000 packets.
- Each simulation result represents the average of 5000 runs.
- Three simulations conducted to validate computed analytical upper bound.

False Positive Rate Attack Graph

- False Positive rate can be reflected by the number of false nodes in the attack graph generated.

  $P1 = n*p / (1-p)$

- Parameters:

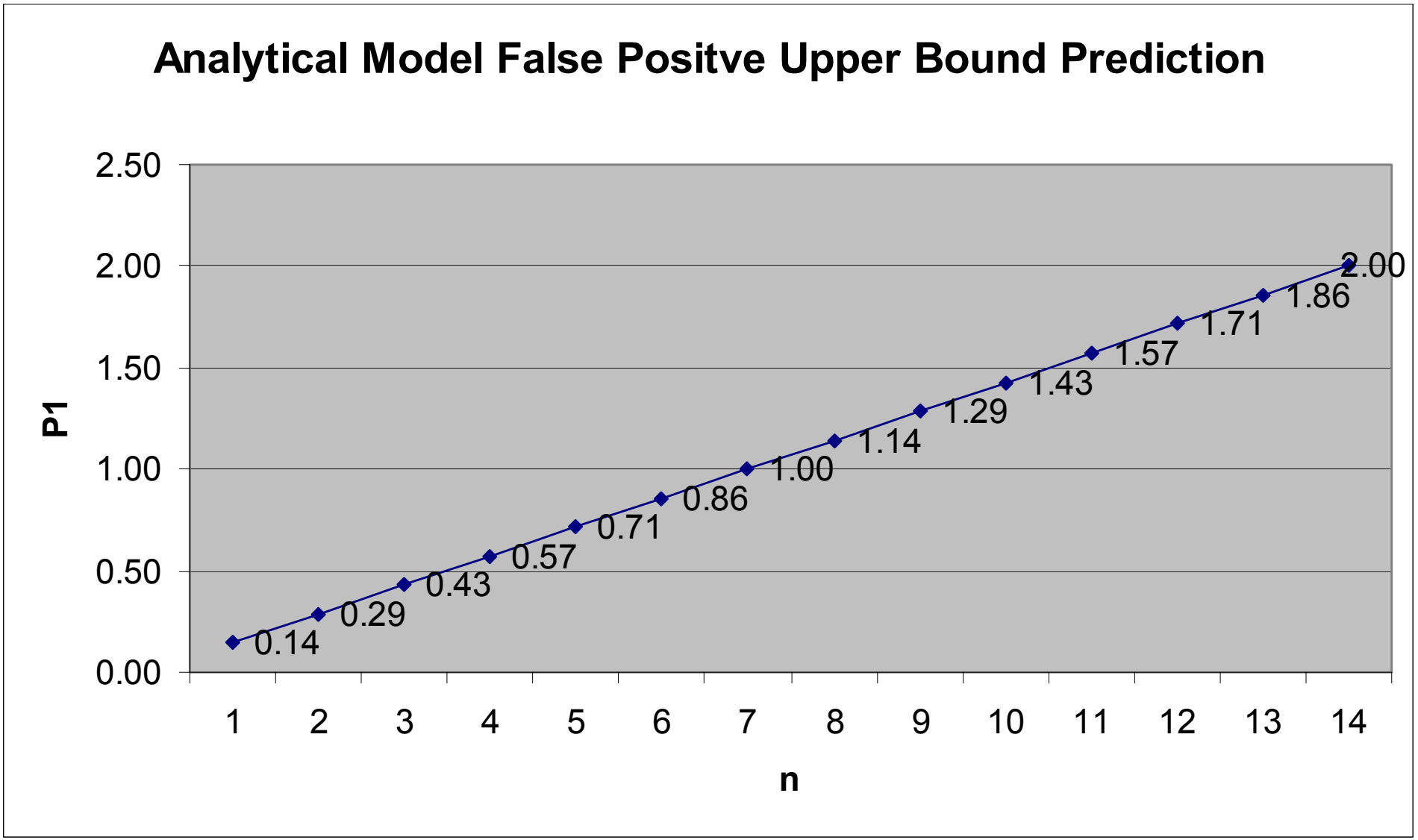
  n: total number of nodes in the true attack graph

  p: 1/8 an arbitrary tuning parameter.

  d: average number of router's neighbors.

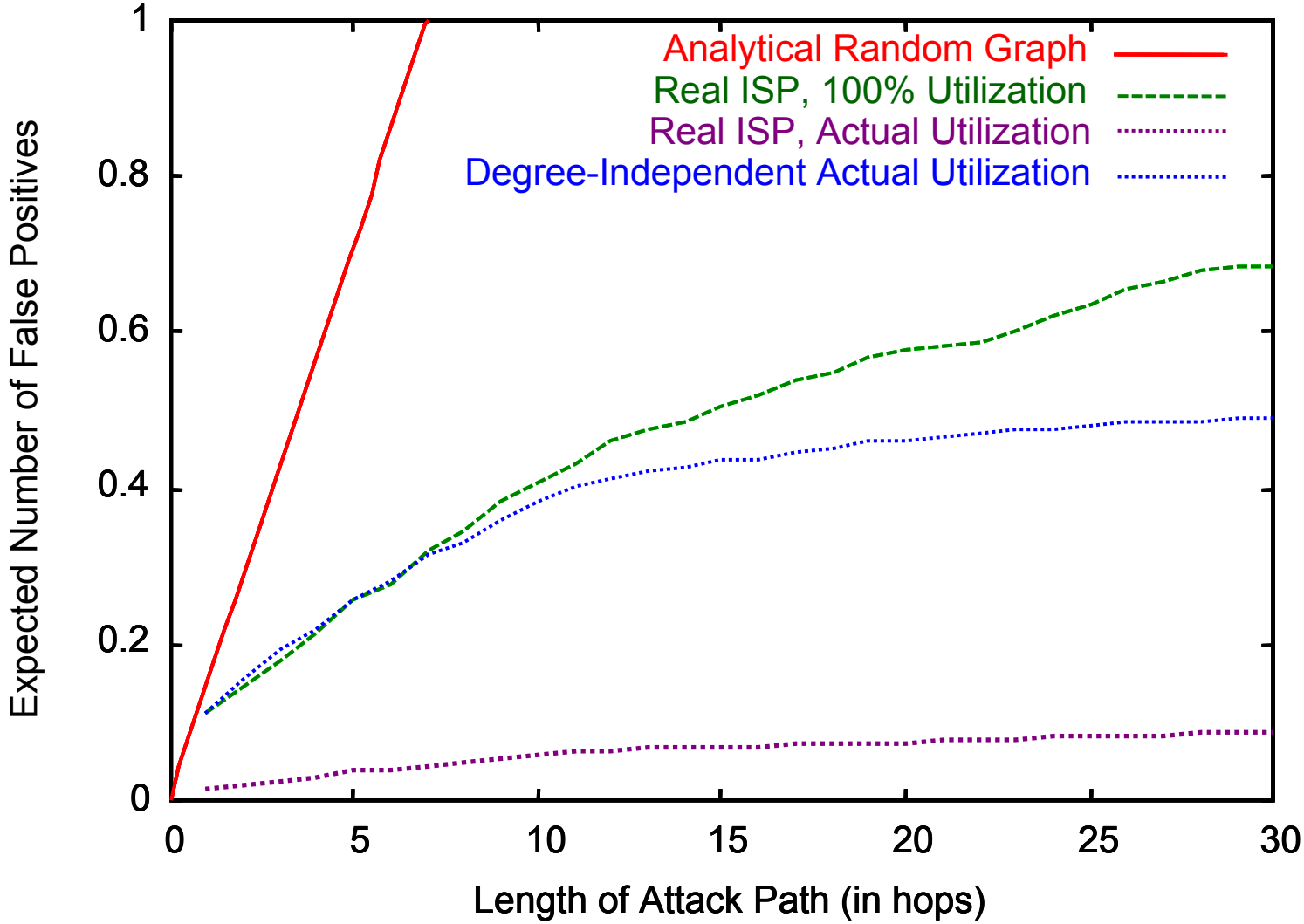
  P: =p/d, false positive rate of a single digest table

  $p=P*d$


Analytical Model False Positve Upper Bound Prediction
2.50
2.00
1.50
1.00
0.50
0.00
P1
0.14
0.29
0.43
0.57
0.71
0.86
1.00
1.14
1.29
1.43
1.57
1.71
1.86
2.00
1
2
3
4
5
6
7
8
9
10
11
12
13
14
n

P1=n*p/(1-p) p=.125

Simulation Results



False Positive Rate of a Single Digest Table

- False Positive Rate

  $P = [1-(1-1/m)^{kn}]^k = (1-e^{-kn/m})^k$

- Parameters

  m: size of bloom filter in bits.

  k: number of hash functions

  n: number of packets the table serve for

For example, when m=5n, k=3, P=0.092

when m=12n, k=8, P=0.00314

Memory Analysis

- Bloom filters require 0.5% of bandwidth.
  - 4-OC-3s require 23MB per minute
  - 32 OC-192 12GB per minute
- Access time important
  - DRAM can support 20Mpkts/sec
  - SRAM needed for OC-192

Summary/Critique

Summary

- Hash-based traceback is a viable alternative
  - Router memory and early detection of suspect packets are keys to effectiveness.
- References

  http://www.ir.bbn.com/projects/SPIE

Issues Summary

- Deployment
  - The SPIE's usefulness increases with deployment.
- Vulnerabilities
  - DDOS may slow SPIE processing time
  - Flow Amplification – duplicate packets
  - Information Leakage – passing information from IDS to SPIE.
- Transformations
  - Problematic and possible attack candidate

Critique

- Generally easy to read and well structured paper.
- Simulation discussion could be improved.
- Complex implementation.
- More discussion and a comparative analysis of the alternatives would be useful. For example packet encoding/tagging would eliminate the storage problem, but increase network bandwidth.