Traceback

Dr. Yong Guan

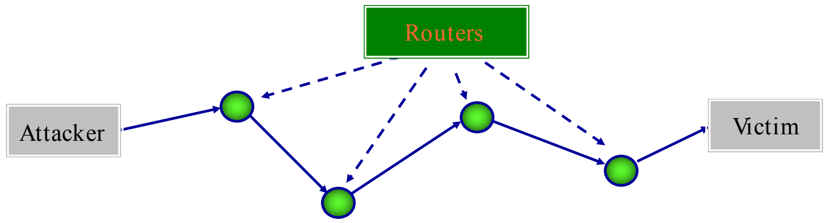
Department of Electrical and Computer Engineering
& Information Assurance Center,
Iowa State University

Overview of Network Attack Attribution

- Goal:
  - Traceback and identification of network attackers.
- Network Attack Attribution problems:
  - IP Trace-back
    - The problem is to trace the path (i.e., a sequence of routers) of a datagram traverse through the Internet.
    - Three classes of schemes:
      - Hash-based schemes
      - Probabilistic Marking Schemes
      - Algebraic Packet Marking Schemes
  - Attack Attribution (or attack traceback)
    - Stepping Stone Attack Attribution
      - The problem is to discover the real origin of the attackers
      - Stepping stones can be compromised hosts, web proxy services, anonymous communication services, etc.
    - DDoS
      - The problem is to discover the master computer
      - In DDoS, a master computer controls a number of zombie computers.

IP Traceback Problem

- Problem Definition
  - IP traceback is to identify the true origin of the attack by tracing IP packet along the path (i.e., a sequence of routers) which it traverses through the Internet.

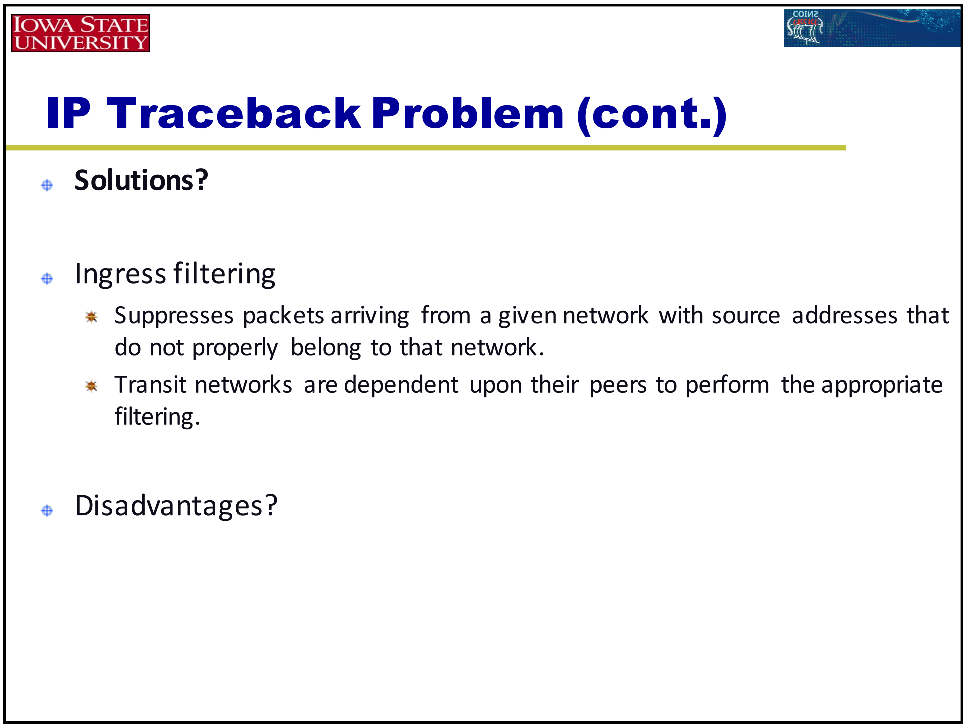


- It's not always easy to determine the source of a packet due to
  - Spoofed source addresses
  - Stateless nature of Internet routing

IP Traceback Problem (cont.)

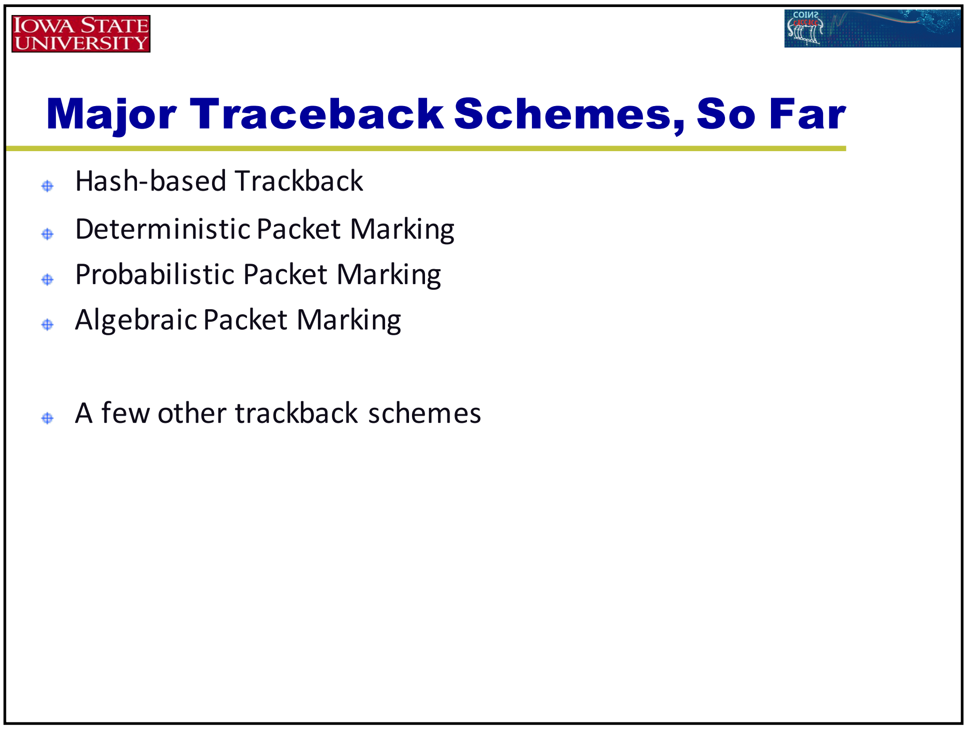
- **The design of the IP protocol makes it difficult to reliably identify the originator of an IP packet.**
  - **Deliberate attempt to disguise a packet's origin (fake source IP address)**
  - **Packet forwarding techniques, such as NAT and encapsulation**

- Accordingly, a well-placed attacker can generate offending IP
- packets that appear to have originated from almost anywhere.



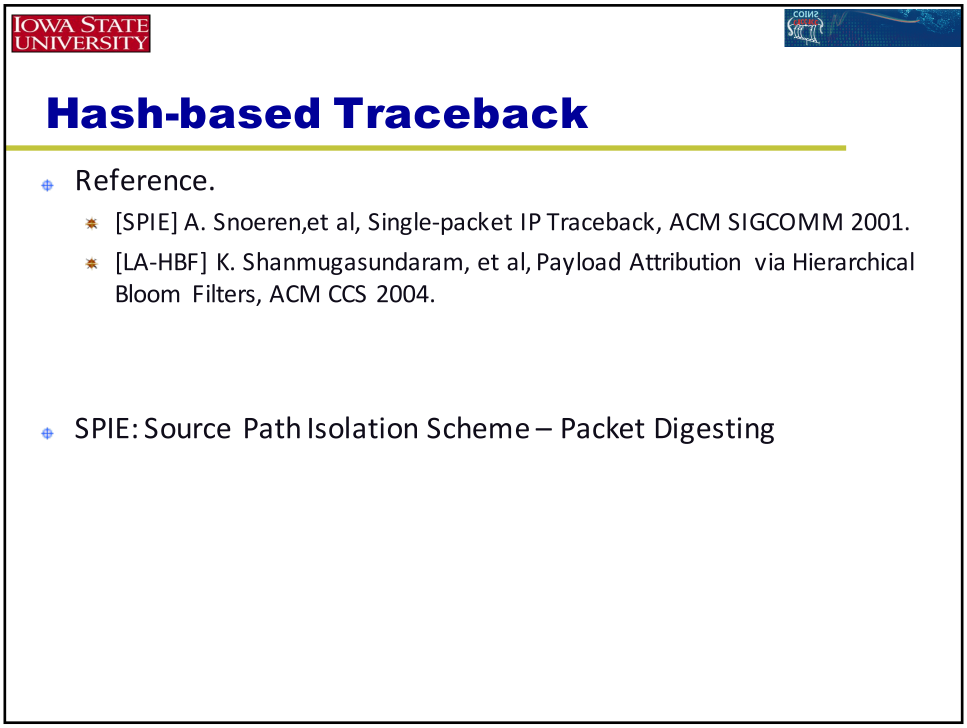
IP Traceback Problem (cont.)

- **Solutions?**
- Ingress filtering
  - Suppresses packets arriving from a given network with source addresses that do not properly belong to that network.
  - Transit networks are dependent upon their peers to perform the appropriate filtering.
- Disadvantages?



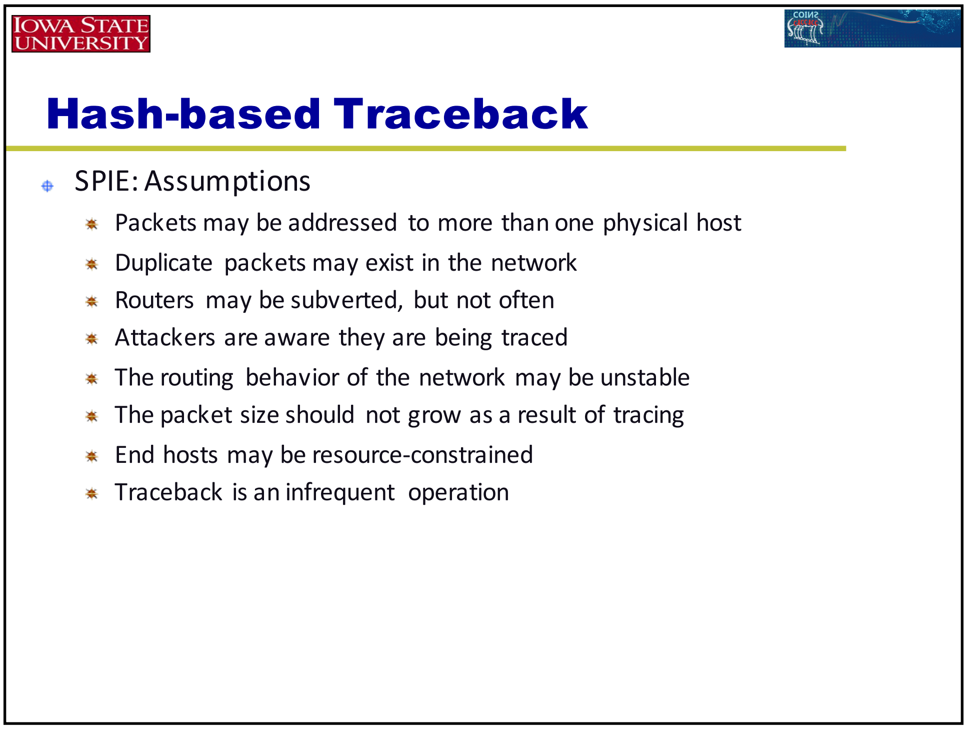
Major Traceback Schemes, So Far

- Hash-based Trackback
- Deterministic Packet Marking
- Probabilistic Packet Marking
- Algebraic Packet Marking
- A few other trackback schemes



Hash-based Traceback

- Reference.
  - [SPIE] A. Snoeren,et al, Single-packet IP Traceback, ACM SIGCOMM 2001.
  - [LA-HBF] K. Shanmugasundaram, et al, Payload Attribution via Hierarchical Bloom Filters, ACM CCS 2004.
- SPIE: Source Path Isolation Scheme – Packet Digesting



Hash-based Traceback

- SPIE: Assumptions
  - Packets may be addressed to more than one physical host
  - Duplicate packets may exist in the network
  - Routers may be subverted, but not often
  - Attackers are aware they are being traced
  - The routing behavior of the network may be unstable
  - The packet size should not grow as a result of tracing
  - End hosts may be resource-constrained
  - Traceback is an infrequent operation

Hash-based Traceback

- SPIE: Goals
  - Consider the source of a packet to be one of:
    - The ingress point to the traceback-enabled network
    - The actual host or network of origin
    - One or more compromised routers within the enabled network
  - Constructing an *attack path*, where the path consists of each router traversed by the packet on its journey from source to the victim.
  - Packet Transformation
    - Encapsulation
    - ICMP packet
    - ….

Hash-based Traceback

- SPIE: Source Path Isolation Scheme – Packet Digesting

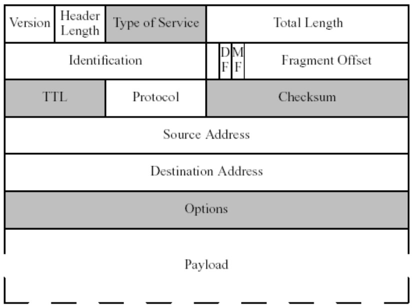


Fig. 1. The fields of an IP packet. Fields in gray are masked out before digesting, including the Type of Service, Time to Live (TTL), IP checksum, and IP options fields.

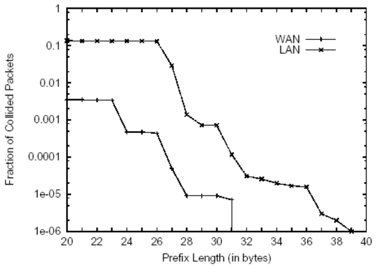
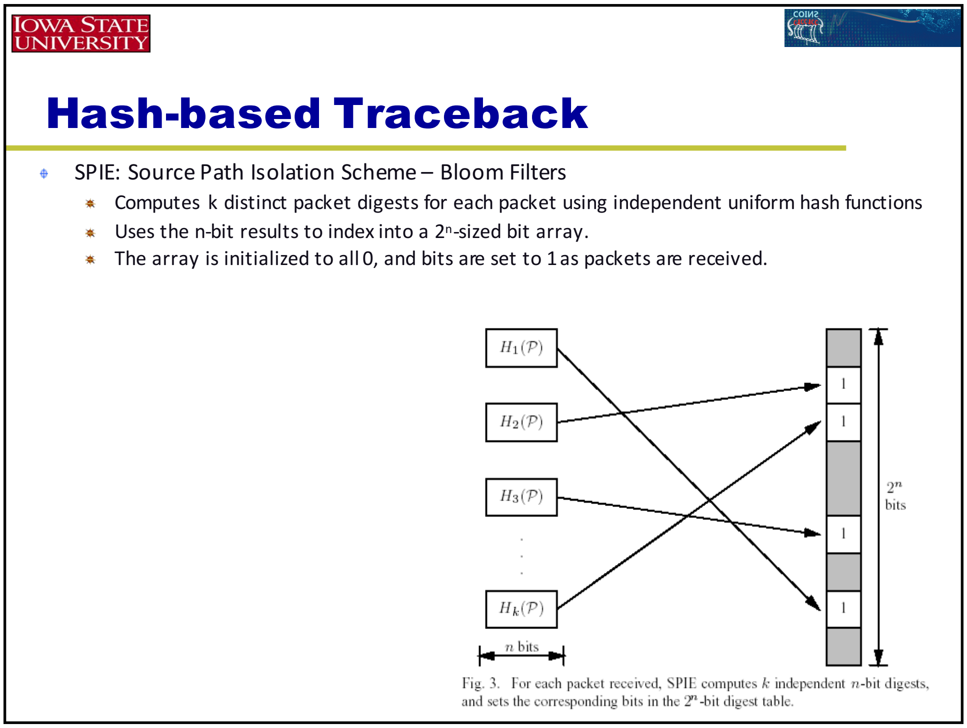


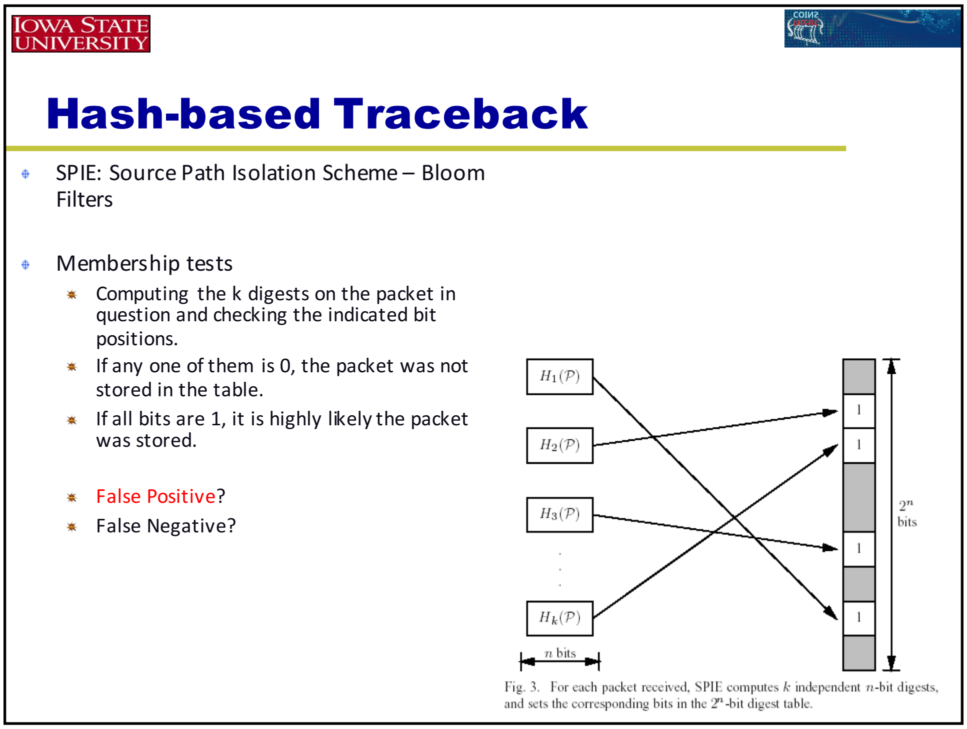
Fig. 2. The fraction of packets that collide (with ToS, TTL, and checksum fields masked out) as a function of prefix length. The WAN trace represents 985,150 packets (with 5,801 duplicates removed) between 6,031 host pairs collected on July 20, 2000 at the University of Florida OC-3 gateway. The LAN trace consists of one million packets (317 duplicates removed) between 2,879 host pairs observed on an Ethernet segment at the MIT Lab for Computer Science.



Hash-based Traceback

- SPIE: Source Path Isolation Scheme – Bloom Filters
  - Computes k distinct packet digests for each packet using independent uniform hash functions
  - Uses the n-bit results to index into a 2^n-sized bit array.
  - The array is initialized to all 0, and bits are set to 1 as packets are received.

Fig. 3. For each packet received, SPIE computes k independent n-bit digests, and sets the corresponding bits in the 2^n-bit digest table.



Hash-based Traceback

- SPIE: Source Path Isolation Scheme – Bloom Filters

- Membership tests
  - Computing the k digests on the packet in question and checking the indicated bit positions.
  - If any one of them is 0, the packet was not stored in the table.
  - If all bits are 1, it is highly likely the packet was stored.

  - False Positive?
  - False Negative?

Fig. 3. For each packet received, SPIE computes k independent n-bit digests, and sets the corresponding bits in the 2^n-bit digest table.

Hash-based Traceback

- SPIE: Source Path Isolation Scheme – Digesting Functions in its Bloom Filters
- Three restrictions:

1. Each member function must distribute a highly correlated set of input values (IP packet prefixes), *P*, as uniformly as possible over the hash's result value space.
   - For a hash function H: *P* → 2^m in *F*, for X=Y in P, Pr{H(X)=H(Y)}=2^{-m}.
2. The event that two packets collide in one hash function (*H*(*x*) = *H*(*y*) for some *H*) be independent of collision events in any other functions (*H'*(*x*) =*H'* (*y*),*H'* != *H*).
3. Member functions must be straightforward to compute at high link speeds.

Hash-based Traceback

- SPIE: Source Path Isolation Scheme – Path Construction
  - Traceback Processing
  - Transformation Processing

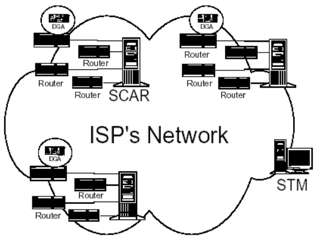


Fig. 4. The SPIE network infrastructure, consisting of Data Generation Agents (DGAs), SPIE Collection and Reduction Agents (SCARs), and a SPIE Traceback Manager (STM).

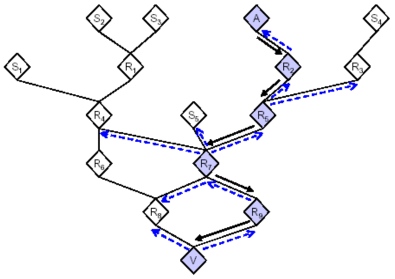


Fig. 6. Reverse path flooding, starting at the victim's router, *V*, and proceeding backwards toward the attacker, *A*. Solid arrows represent the attack path; dashed arrows are SPIE queries. Queries are dropped by routers that did not forward the packet in question.

Hash-based Traceback

SNOEREN *ET AL.*: SINGLE-PACKET IP TRACEBACK

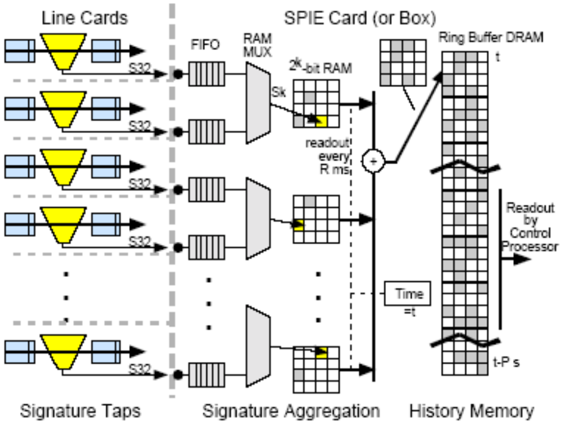


Fig. 7. A sample SPIE DGA hardware implementation for high-speed routers.

Deterministic Packet Marking

- Ref. Belenky and Ansari’s papers
  - Marking each individual packet as it enters the network
  - Using the 16-bit Packet ID field and the reserved 1-bit Flag in the IP header
  - This mark remains unchanged for as long as the packet traverses the network.
  - The packet is marked by the interface closest to the source of the packet on the edge ingress router.
  - The interface makes a distinction
  - between incoming and outgoing packets. Incoming packets are marked; outgoing packets are not marked.

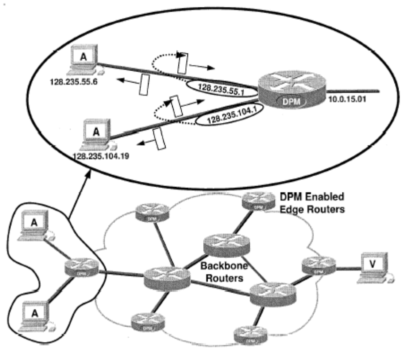
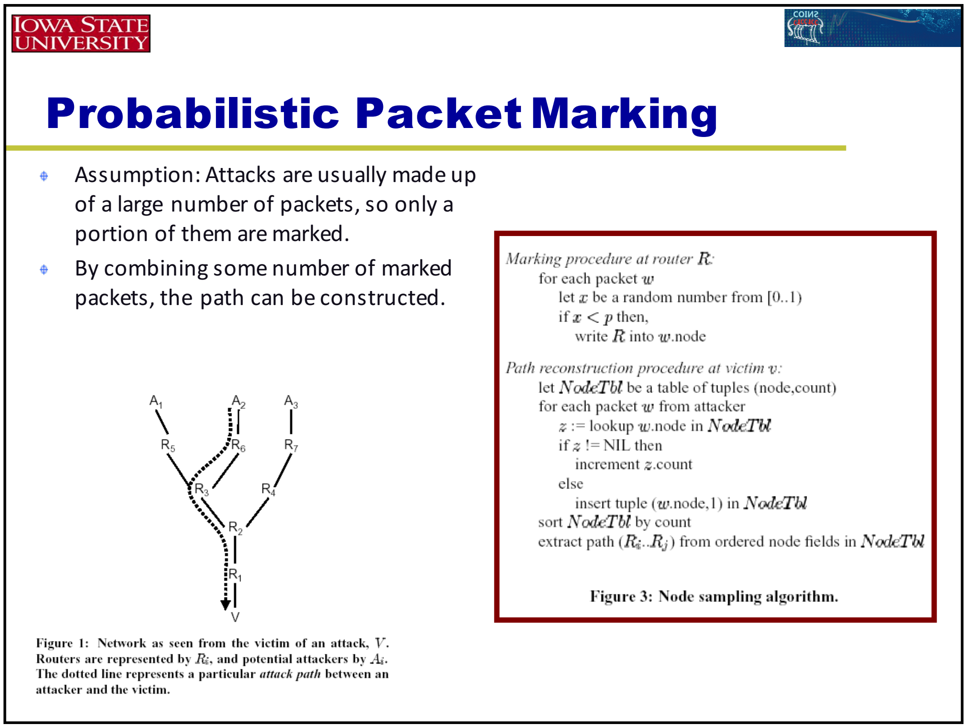


Fig. 1. Deterministic packet marking (DPM).



Probabilistic Packet Marking

- Assumption: Attacks are usually made up of a large number of packets, so only a portion of them are marked.
- By combining some number of marked packets, the path can be constructed.

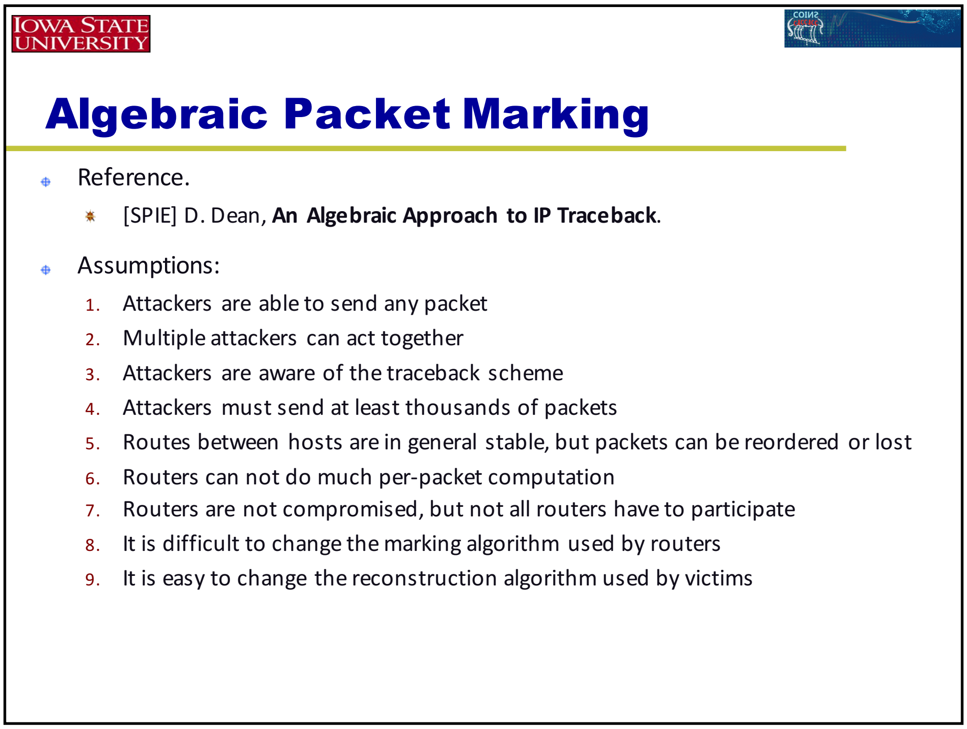
A_1
A_2
A_3
R_5
R_6
R_7
R_3
R_4
R_2
R_1
V

Figure 1: Network as seen from the victim of an attack, V. Routers are represented by R_i, and potential attackers by A_i. The dotted line represents a particular *attack path* between an attacker and the victim.

```
Marking procedure at router $R$:
    for each packet $w$
        let $x$ be a random number from [0..1)
        if $x < p$ then,
            write $R$ into $w$.node

Path reconstruction procedure at victim $v$:
    let $NodeTbl$ be a table of tuples (node,count)
    for each packet $w$ from attacker
        $z$ := lookup $w$.node in $NodeTbl$
        if $z$ != NIL then
            increment $z$.count
        else
            insert tuple ($w$.node,1) in $NodeTbl$
    sort $NodeTbl$ by count
    extract path ($R_i..R_j$) from ordered node fields in $NodeTbl$
```

Figure 3: Node sampling algorithm.



Algebraic Packet Marking

- Reference.
  - [SPIE] D. Dean, **An Algebraic Approach to IP Traceback**.
- Assumptions:
  1. Attackers are able to send any packet
  2. Multiple attackers can act together
  3. Attackers are aware of the traceback scheme
  4. Attackers must send at least thousands of packets
  5. Routes between hosts are in general stable, but packets can be reordered or lost
  6. Routers can not do much per-packet computation
  7. Routers are not compromised, but not all routers have to participate
  8. It is difficult to change the marking algorithm used by routers
  9. It is easy to change the reconstruction algorithm used by victims

Algebraic Packet Marking

Algebraic Coding of Paths

- All of these schemes are based on the principal of reconstructing a polynomial in a prime field.
- The basic idea is that for any polynomial *f* (*x*) of degree *d* in the prime field *GF p* , we can recover *f* (*x*) , given *f* (*x*) evaluated at (*d* +1) unique points.
- Let $A_1, A_2, \ldots, A_n$ be the 32-bit IP addresses of the routers on path *P*. Let $f_P(x) = A_1 x^{n-1} + A_2 x^{n-2} + \ldots + A_{n-1} x + A_n$.
- We then somehow evaluate $f_P(x)$ as the packet x travels along the path, accumulating the result of the computation in a running total along the way.
- When enough packets from the same path reach the destination, then f_P can be reconstructed by interpolation.

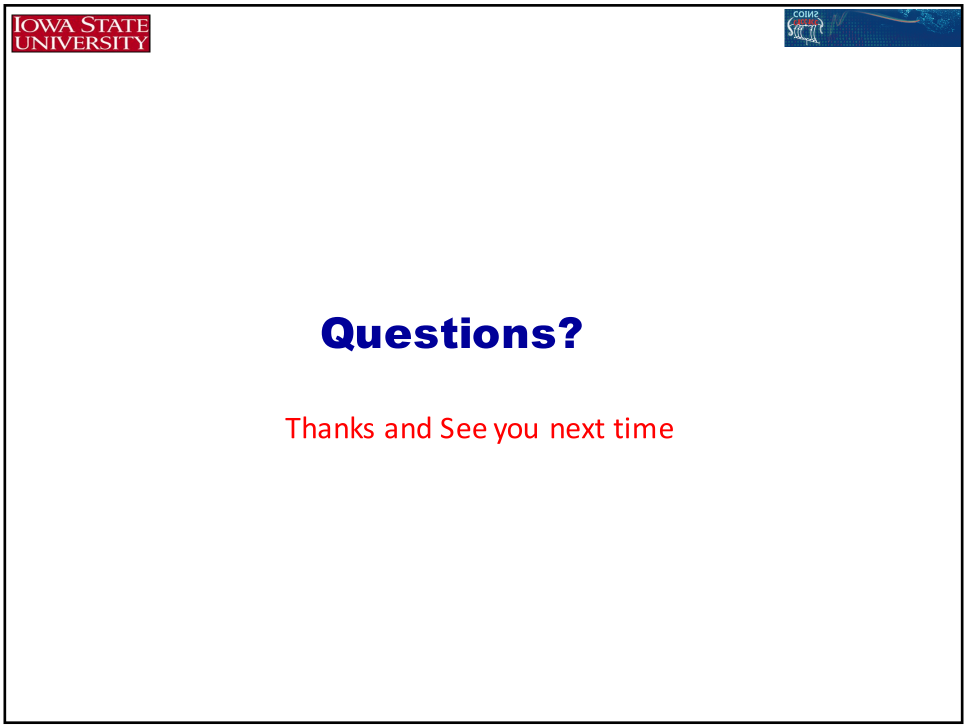
Algebraic Packet Marking

Full-path Encoding

- Similar to the technique used by Savage's PPM, with the major difference being that this scheme is based on algebraic techniques.
- *Better* filter out attacker generated noise and separate multiple paths.

$$\begin{pmatrix} 1 & x_1 & x_1^2 & \ldots & x_1^{n-1} \\ 1 & x_2 & x_2^2 & \ldots & x_2^{n-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \ldots & x_n^{n-1} \end{pmatrix} \begin{pmatrix} A_1 \\ A_2 \\ \vdots \\ A_n \end{pmatrix} = \begin{pmatrix} FullPath_{n,1} \\ FullPath_{n,2} \\ \vdots \\ FullPath_{n,3} \end{pmatrix}$$

At beginning of a path, Let FullPath0,j=0. Each router i on the path calculates $FullPath_{i,j}=(FullPath_{i-1,j} \bullet X_j + R_i)$ where X_j is a random value passed in each paket, R_i is the router's IP address. At the packet's destination FullPath will equal $(R_n X_{n-1} + R_{n-1} X_{n-2} + \ldots + R_2 X + R_1)$



Questions?

Thanks and See you next time