

Revisiting Bloom Filters

Payload attribution via Hierarchical Bloom Filters

Kulesh Shanmugasundaram, Herve Bronnimann, Nasir Memon

600.624 - Advanced Network Security

version 3

Overview

- Questions
- Collaborative Intrusion Detection
- Compressed Bloom filters

When to flush the Bloom filter?

“They said they have to refresh the filters at least every 60 seconds. Is it pretty standard?”

In general, FP chosen \Rightarrow m/n and k (minimum values)

Given $m \Rightarrow$ maxim for n

m/n	k	k=1	k=2	k=3	k=4	k=5	k=6	k=7	k=8
2	1.39	0.393	0.400						
3	2.08	0.283	0.237	0.253					
4	2.77	0.221	0.155	0.147	0.160				
5	3.46	0.181	0.109	0.092	0.092	0.101			
6	4.16	0.154	0.0804	0.0609	0.0561	0.0578	0.0638		

How many functions?

“They report using MD5 as the hashing function but only use two bytes of it to achieve the FP_0 . I don’t follow why this is the case.”

Paper says: “Each MD5 operation yields 4 32-bit integers and two of them to achieve the required FP_0 .”

m/n	k	k=1	k=2	k=3	k=4	k=5	k=6	k=7	k=8
2	1.39	0.393	0.400						
3	2.08	0.283	0.237	0.253					
4	2.77	0.221	0.155	0.147	0.160				
5	3.46	0.181	0.109	0.092	0.092	0.101			
6	4.16	0.154	0.0804	0.0609	0.0561	0.0578	0.0638		

How do we know source IP addresses?

“[...] what do they mean by source and destination? [...] the ‘use of zombie or stepping stone hosts’ makes attribution difficult”.

“[...] the attribution system needs a list of ‘candidate hostIDs’. Honestly, I am not sure what they mean by this.”

Paper says:

“For most practical purposes hostID can simply be (SourceIP, Destination IP)”

More accuracy with block digest?

“The block digest is a HBF as all the others and the number of inserted values are the same as the offset digest. Why is then the accuracy better?”

The number of entries is the same but think about how you do a query? How is FP rate influenced by that?

Query time /space tradeoff (block digest)

“[...] such an extension (block digest) would shorten query times, but increase the storage requirement. What is the tradeoff between querying time and space storage?”

What payload attribution? (aka Spoofed addresses)

“I am unsure of the specific contribution that this paper makes. The authors purport to have a method for attributing payload to source, destinations pairs, yet the system itself has no properties that allow you to correlate a payload with a specific sender”.

What would you prefer: a system like this one or one which requires global deployment (like SPIE)?

Various comments

How do you find it?

“smart and simple”

“quite ingenious with regard to storage and querying”

“The authors seem to skip any analysis that doesn’t come up in the actual implementation.”

Fabian’s answer: *“That’s fine :-)”*

“seem to be a useful construction”

“I thought this was a decent paper overall. [...] I think it is also poorly written and lacks a good number of details.”

“I liked this paper very much.”

Extensions

Ryan:

“Large Batch Authentication”

Scott:

Use a variable length block size (hm...)

Razvan:

Save the space for hostIDs using a global IP list?

Jay’s crazy idea:

Address the spoofed address problem using hop-count-filtering?

Collaborative Intrusion Detection

IDS are typically constrained within one administrative domain.

- single-point perspective cause slow scans to go undetected
- low-frequency events are easily lost

Sharing IDS alerts among sites will enrich the information on each site and will reveal more detail about the behavior of the attacker

Benefits

- Better understanding of the attacker intent
- Precise models of adversarial behavior
- Better view of global network attack activity

“Worminator” Project

Developed by IDS group at Columbia University

- Collaborative Distributed Intrusion Detection, *M. Locasto, J. Parekh, S. Stolfo, A. Keromytis, T. Malkin, V. Misra, CU Tech Report CUCS-012-04, 2004.*
- Towards Collaborative Security and P2P Intrusion Detection, *M. Locasto, J. Parekh, A. Keromytis, S. Stolfo, Workshop on Information Assurance and Security, June 2005.*
- On the Feasibility of Distributed Intrusion Detection, *CUCS D-NAD Group, Technical report, Sept. 2004.*
- Secure “Selecticast” for Collaborative Intrusion Detection System, *P. Gross, J. Parekh, G. Kaiser, DEBS 2004.*

Terminology

1. Network event
2. Alert
3. Sensor node
4. Correlation node
5. Threat assessment node

Challenges

- Large alert rates
- A centralized system to aggregate and correlate alert information is not feasible.
- Exchanging alert data in a full mesh quadratically increases bandwidth requirements
- If alert data is partitioned in distinct sets, some correlations may be lost
- Privacy considerations

Privacy Implications

Alerts may contain sensitive information: IP addresses, ports, protocol, timestamps etc.

Problem: Reveal internal topology, configurations, site vulnerabilities.

From here the idea of “anonymization”:

- Don't reveal sensitive information
- Tradeoff between anonymity and utility

Assumptions

- Alerts from Snort
- Focus on detection of scanning and probing activity
- Integrity and confidentiality of exchange messages can be addressed with IPsec, TLS/SSL & friends
- Unless compromised, any participant provides entire alert information to others (they don't disclose partial data)

Threat model

- Attacker attempts to evade the system by performing very low rate scans and probes
- Attacker can compromise a subset of nodes to discover information about the organization he is targeting

Bloom filters to the Rescue

IDS parses alerts output and hashes IP/port information into a Bloom filter. Sites exchange filters (“watchlists”) to aggregate the information

Advantages:

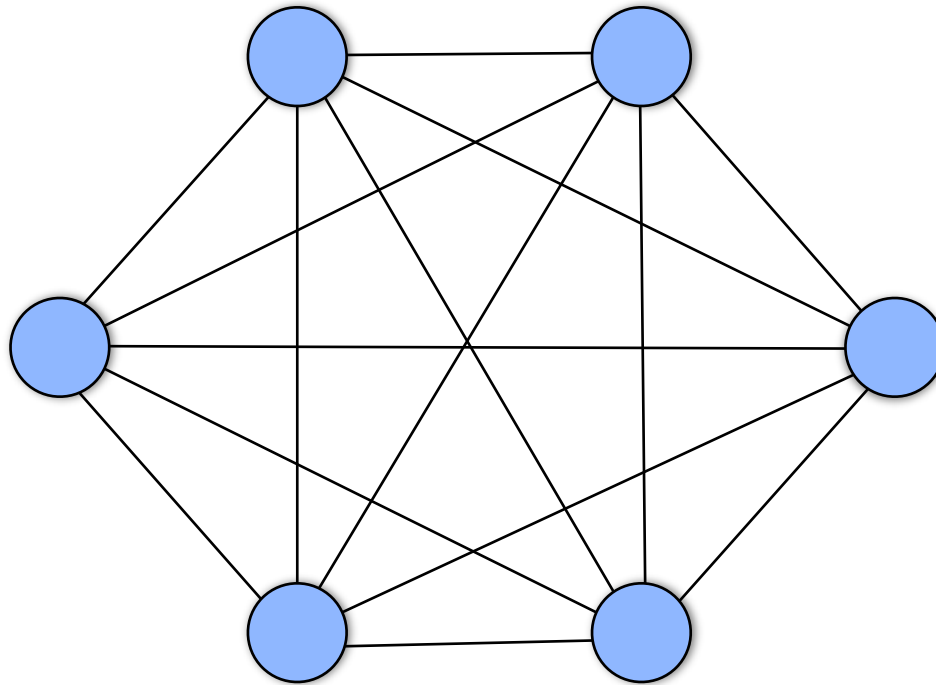
- Compactness (e.g. 10k for thousands of entries)
- Resiliency (never gives false negatives)
- Security (actual information is not revealed)

Distributed correlation

Approaches:

1. Fully connected mesh
2. DHT
3. Dynamic overlay network
 - Whirlpool

I. Fully connected mesh



Each node communicates with
each other node

2. Distributed Hash Tables

DHT design goals:

- Decentralization
- Scalability
- Fault tolerance

Idea:

Keys are distributed among the participants
Given a key, find which node is the owner

Example:

(filename, data) \Rightarrow SHA1(filename) = k , $put(k, data)$

Search: $get(k)$

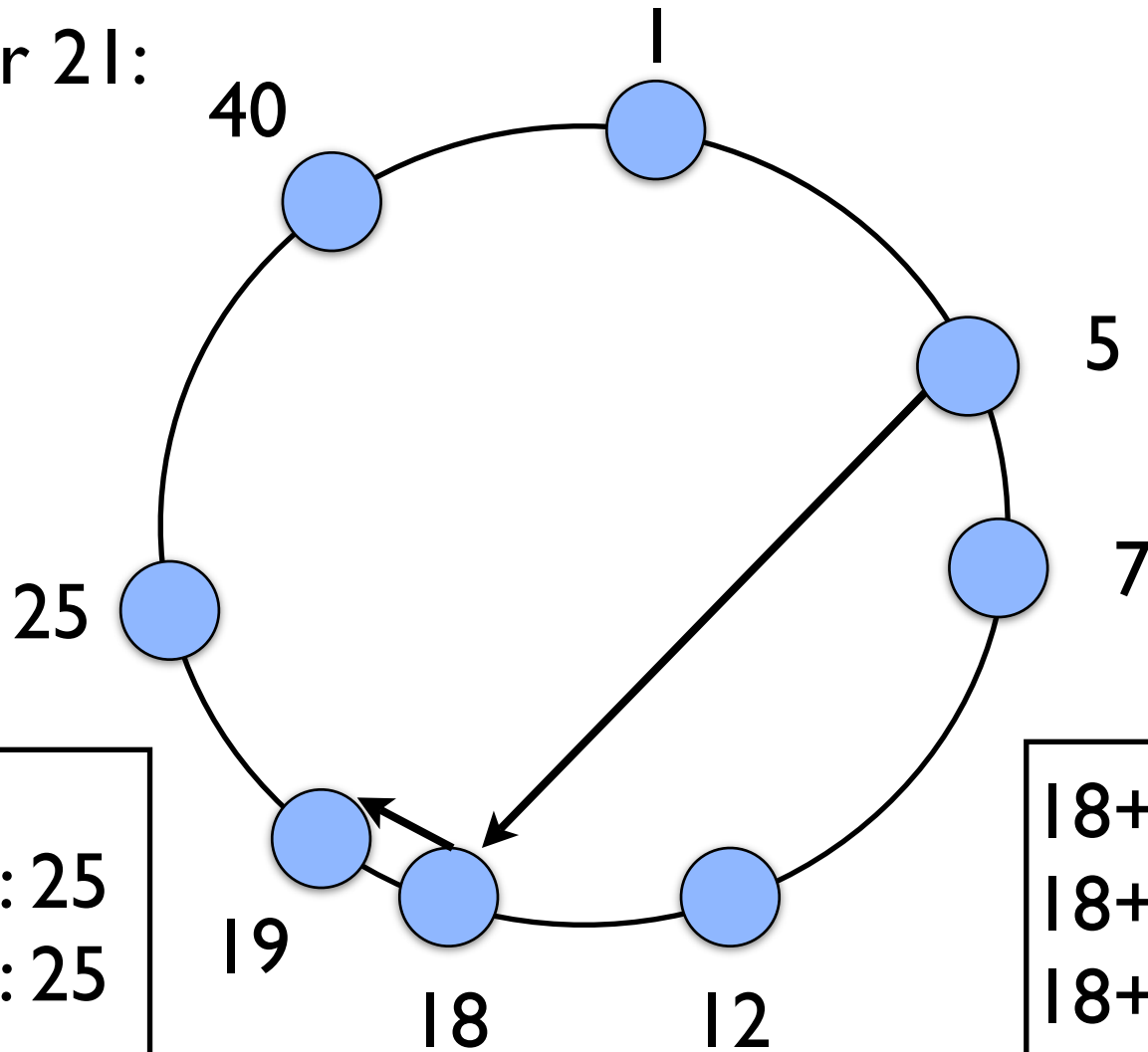
Chord

Chord: A Scalable Peer-to-peer Lookup Service for Internet Applications
Ion Stoica, Robert Morris, David Karger, M. Frans Kaashoek, and Hari Balakrishnan, MIT
ACM SIGCOMM 2001

- Each node has a unique identifier ID in range $[0, 2^m]$ (hash) and is responsible to cover objects with keys between previous ID and his own ID.
- Each node maintains a table (finger table) that stores identifiers of other m overlay nodes.
- Node s is in finger table of t if it is the closest node to $t + 2^i \pmod{2^m}$
- Lookup will take at most m steps.

Chord

Search for 21:



5+1: 7
5+2: 7
5+4: 12
5+8: 18
5+16: 25

19+1: 25
19+2: 25
...

18+1: 19
18+2: 25
18+4: 25
18+8: 40
18+16: 40

DHT for correlations

Map alert data (IP addresses, ports) to correlation nodes.

Limitations:

- nodes are single point of failure for specific IPs
- too much trust in a single node (collects highly related information at one node)

Dynamic Overlay Networks

Idea: Use a dynamic mapping between the nodes and content.

Requirement: Need to have the correct subset of nodes that must communicate given a particular alert.

There is a theoretical optimal schedule for communication information (correct subsets are always communicating).

Naive solution: pick relationships at random.

Whirlpool

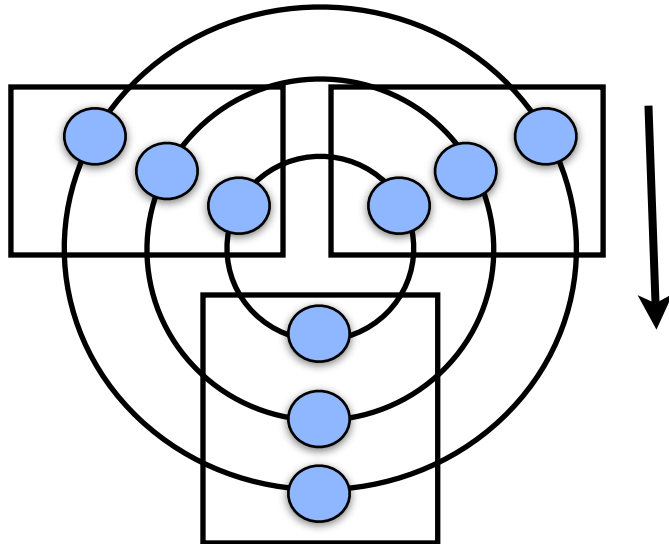
Mechanism for coordinating the exchange of information between the members of a *correlation group*.

Approximates “optimal” scheduler by using a mechanism which allows a good balance between traffic exchange and information loss.

Whirlpool

- N nodes arranged in concentric circles of size \sqrt{N}
- Inner circles spin with higher rates than outer circles
- A radius that crosses all circles will define a “family” of nodes that will exchange their filters.

Provides stability of the correlation mechanism and brings fresh information into each family.



“Practical” results

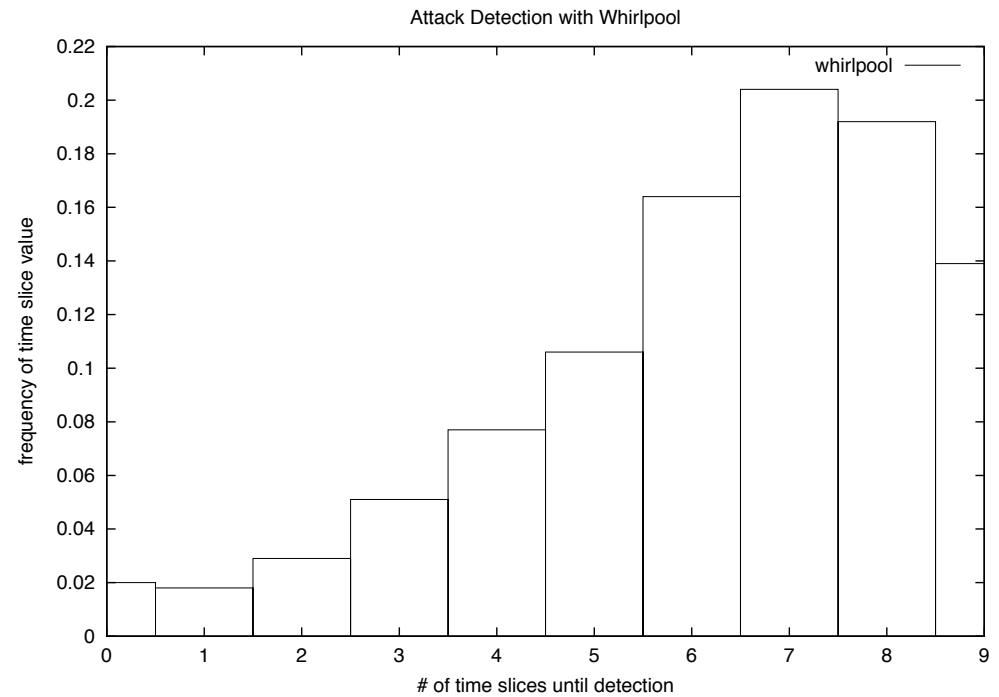
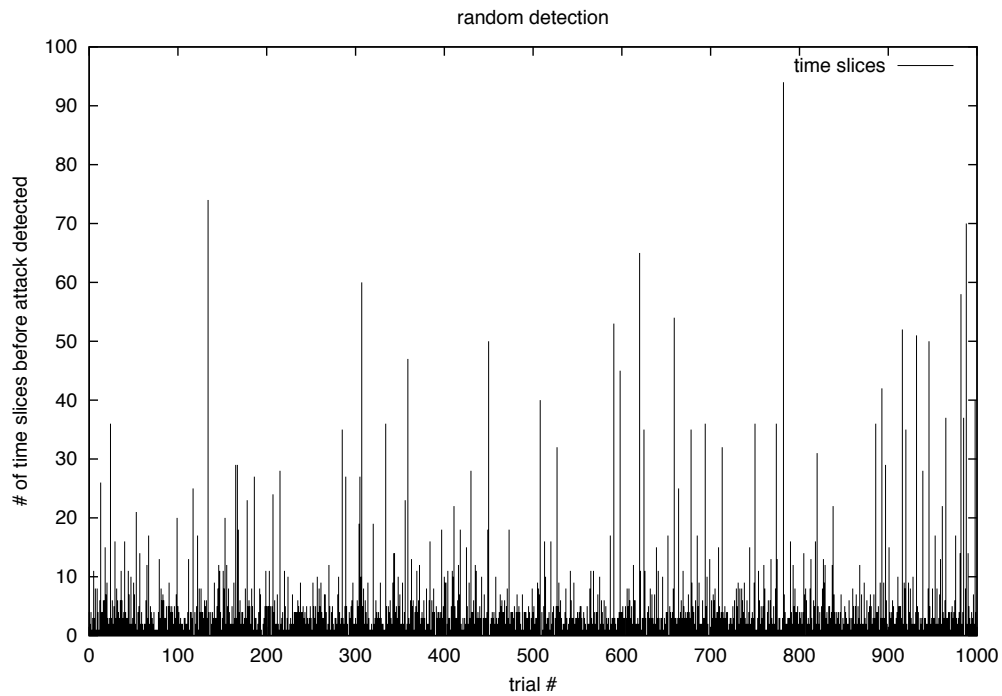
Preliminaries:

Bandwidth Effective Utilization Metric, $BEUM = \frac{1}{t * N \sqrt{N}}$

Comparison between (for 100 nodes):

- Full mesh distribution strategy, $BEUM = 1/10000$
- Randomized distribution strategy, $BEUM = 1/(t * B)$
5-6 time slots to detect an attack
- Whirlpool
6 time slots on average

“Practical” results



Whirlpool doesn't need to keep a long history (9 versus 90)

Secure "Selecticast" for Collaborative Intrusion Detection Systems

Philip Gross, Janak Parekh and Gail Kaiser, Columbia University
International Workshop on Distributed Event-Based Systems 2004

- Share intrusion detection data among organizations to predict attacks earlier.
- Participants collect lists of suspect IPs and want to be notified if others suspect the same IPs.
- Alerts regarding external probes should be visible only to participants which experienced probes from the same source address.

Selecticast

System concerns:

- size of submissions and notifications in transit
- size of the subscription representations in router memory
- speed to compute intersections
- what service to offer? (number, identities list)

Attempt #1: Plain Hash Tables

- Clients hash alerts and submit the lists to the router
- The router maintains a hash table, each entry points to the list of the clients who sent that alert
- No false positives
- Allows deletion of alerts
- Size

Attempt #1: Plain Hash Tables

1. size of submissions and notifications in transit
 - small, hashes of alerts
2. size of the subscription representations in router memory
 - takes a lot of space
3. speed to compute intersections
 - very easy, an entry contains directly the list of participants subscribed to that alert
4. service
 - notifies which participants submitted the same alert

Attempt #2: Pure Bloom Filters

- Clients submit a Bloom filter representing their alerts
- How does the router look for matches?

Attempt #2: Pure Bloom Filters

Bloom filter of size m storing n distinct values, k bits per item.

A bit is set with probability $p = 1 - (1 - 1/m)^{kn}$

One bit matches a bit from the other filter is a Bernoulli trial with chance of success p .

Expected number of successes in kn trials is knp

Ex: $k=6, n=1183, m=2^{23}, p \approx 0.0008, knp \approx 6.0$

Problem: it will require 7000+ bits/item !!!

Attempt #2: Pure Bloom Filters

1. size of submissions and notifications in transit
 - need to transmit an entire Bloom filter
2. size of the subscription representations in router memory
 - a Bloom filter for each client, but it must be big to lower the false positive rate
3. speed to compute intersections
 - easy, need to intersect a filter with everybody else's filter
4. service
 - notifies which participants submitted the same alert

Attempt #3: Hybrid Bloom Filters

- A client hashes an alert k times and submit the list of hashes to the router
- The router maintains one Bloom filter of size $8\hat{n}$ per client (we need an explicit bound of n since we cannot resize the filter)
- The router uses the hash values to check them against the others Bloom filters, updates the Bloom filter and discard the values

Attempt #3: Hybrid Bloom Filters

Small issue with transferred size:

k hashes $(0..m-1) \Rightarrow k \ln m$ hash bits per item

$m = 8n \Rightarrow k \ln m = k(3 + \ln n)$

Implication:

$k = 6$ and sets of 2,000 to 128,000 items \Rightarrow 84-120 hash bits per item

Alternative:

double hashing (32 bit for transmission then rehash to 120 bits for inserting into the filter)

Attempt #3: Hybrid Bloom Filters

1. size of submissions and notifications in transit
 - small (need to sent *one* hash)
2. size of the subscription representations in router memory
 - small (a Bloom filter for each client)
3. speed to compute intersections
 - easy, need to check k hashes in everybody else's filter
4. service
 - notifies which participants submitted the same alert

Mapping Internet Passive Monitors

Mapping Internet Sensors With Probe Response Attacks
John Bethencourt, Jason Franklin, Mary Vernon

Vulnerabilities of Passive Internet Threat Monitors
Yoichi Shinoda (JAIST),
Ko Ikay (National Police Agency, Japan),
Motomu Itoh (JPCERT/CC)

USENIX Security 2005

Mapping Internet Passive Monitors

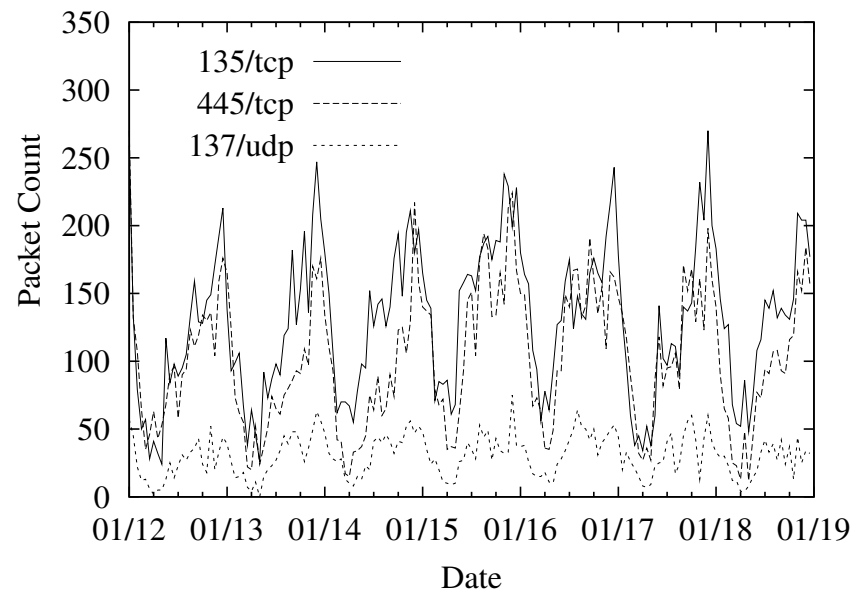
Monitors that periodically publish their results on the Internet are vulnerable to attacks that can reveal their locations.

The idea is to use the *feedback mechanism*:

- Probe an IP address with activity that will be reported if the address is monitored
- Check if the activity (TCP connection to a blocked port) is reported

Report types: Port Table, Time-Series Graph

```
% cat port-report-table-sample
# port  proto  count
8       ICMP   394
135     TCP    11837
445     TCP    11172
137     UDP    582
139     TCP    576
      ⋮
```



Port table attack

Requirements: Send enough packets on a port to be able to distinguish the probe from other activities

Port	Reports	Sources	Targets
325	99321	65722	39
1025	269526	51710	47358
139	875993	42595	180544
3026	395320	35683	40808
135	3530330	155705	270303
225	8657692	366825	268953
5000	202542	36207	37689
6346	2523129	271789	2558

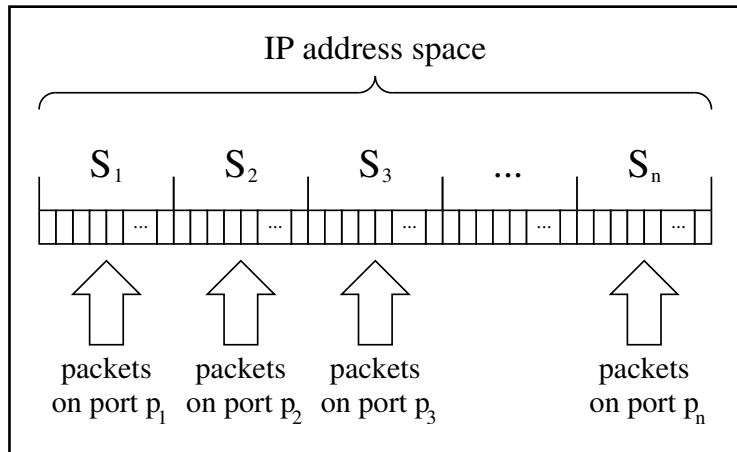
“Smart” system

Table 2: Example excerpt from an ISC port report.

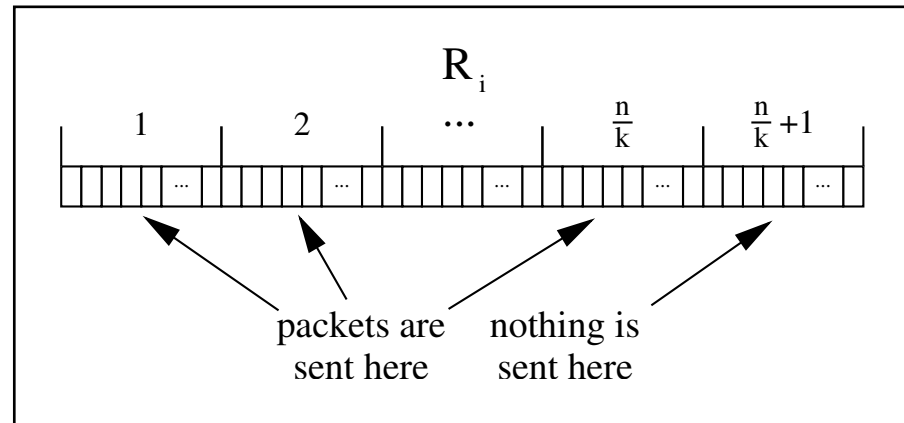
Port table attack

- *Problem:* There are too many addresses to check one after another
 - most participants only submit logs to the ISC every hour
 - there are about 2.1 billion valid, routable IP addresses
- *Alternative:* test many addresses in the same time
 - vast majority of IP addresses are not monitored
 - send probes to each address, in parallel
 - rule out if no activity is reported
 - since malicious activity is reported by port, use different ports for simultaneous tests

Basic Probe Response Attack

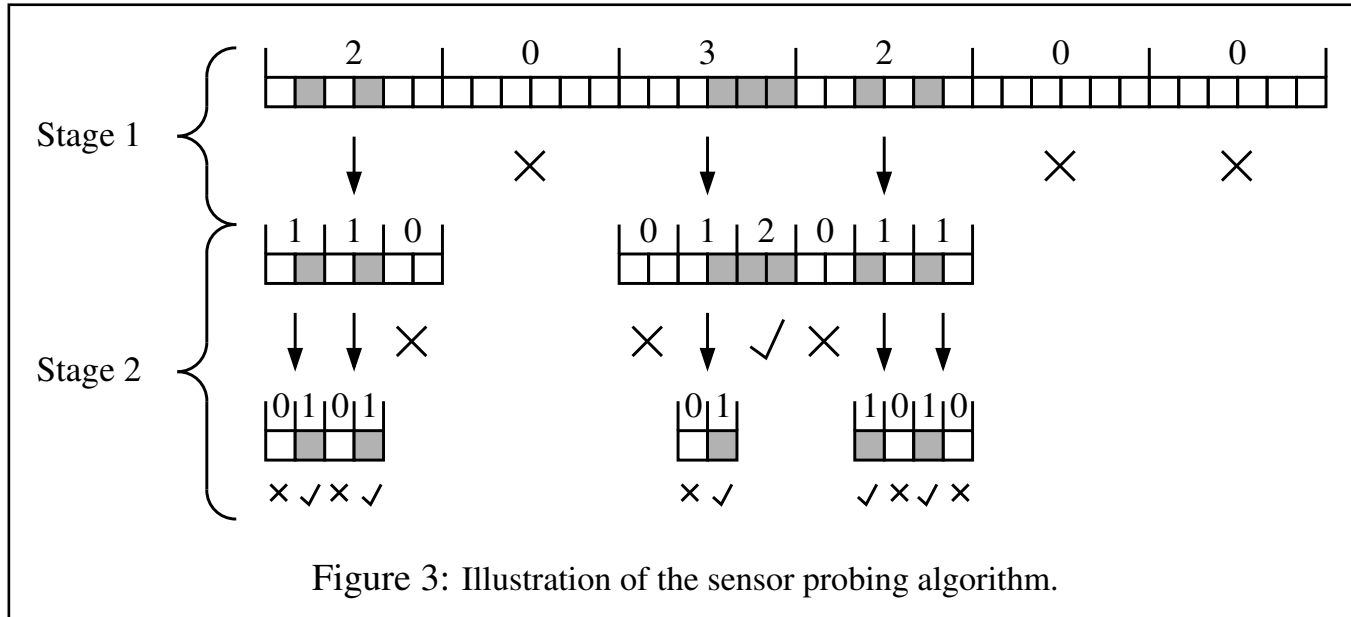


First stage



Second stage

Example



External activity?

- noise cancellation technique

Simulation

- T1 attacker 1.544 Mbps of upload bandwidth
- Fractional T3 attacker 38.4 Mbps of upload bandwidth
250 cable modems botnet
- OC6 attacker 384 Mbps of upload bandwidth
2,500 cable modems botnet

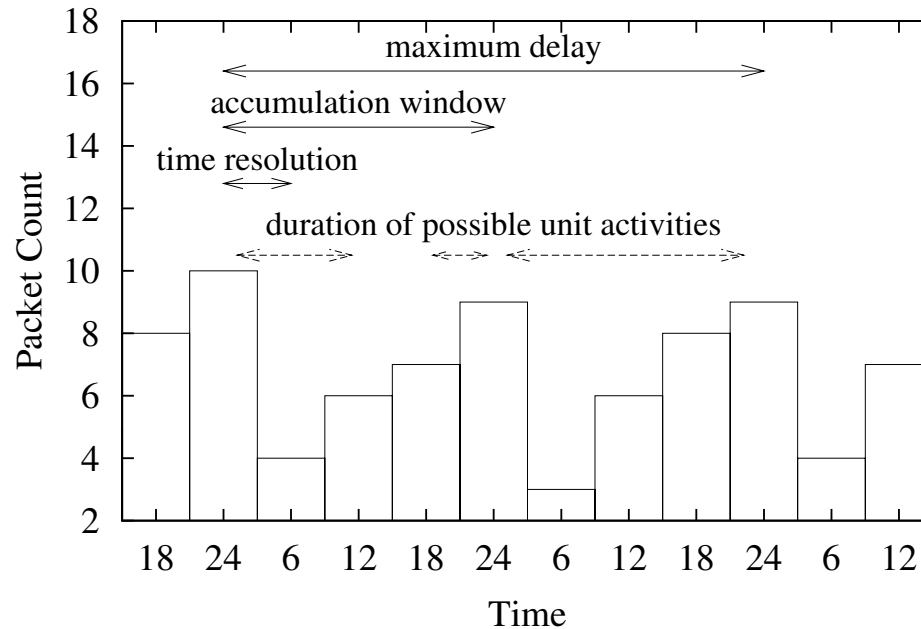
type of mapping	bandwidth available	data sent	false positives	false negatives	correctly mapped addresses	time to map
exact	OC6	1,300GB	0	0	687,340	2 days, 22 hours
exact	T3	687GB	0	0	687,340	4 days, 16 hours
exact	T1	440GB	0	0	687,340	33 days, 17 hours
superset	T3	683GB	3,461,718	0	687,340	3 days, 6 hours
subset	T1	206GB	0	182,705	504,635	15 days, 18 hours

Table 4: Time to map sensor locations. (ISC sensor distribution)

Feedback mechanism is changed

Feedback properties:

Accumulation window	Type Sensitivity
Time Resolution	Dynamic Range
Feedback Delay	Counter Resolution / Level Sensitivity
Retention Time	Cut-off and Capping



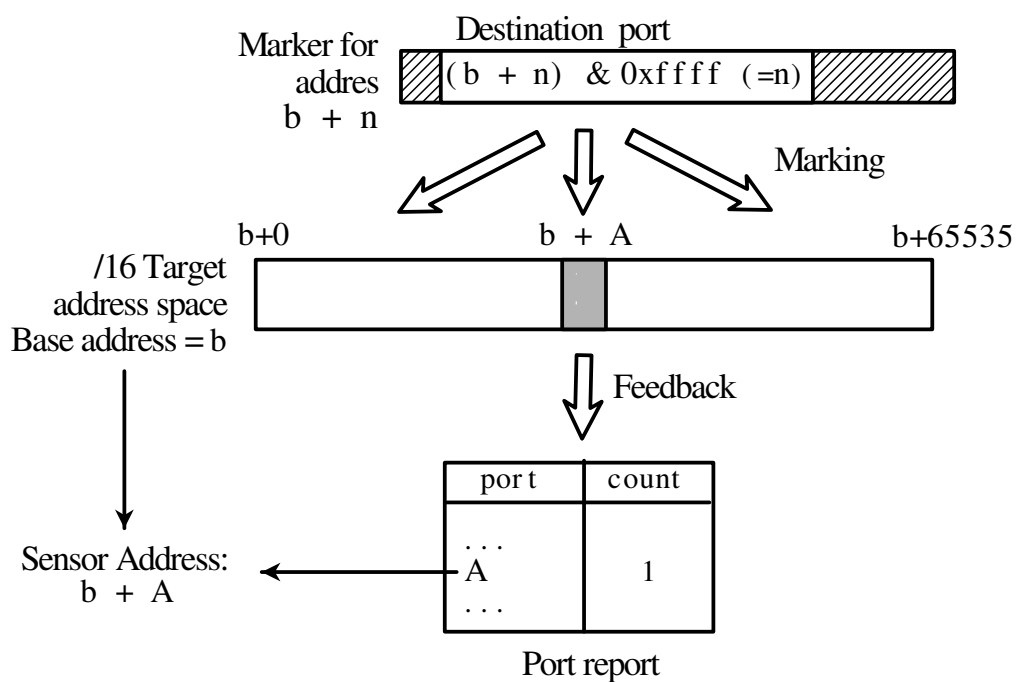
Possible Marking Strategies

- Advanced-Encoded-Port Marking
- Time Series Marking
- Uniform Intensity Marking
- Radix-Intensity Marking
- Radix-Port Marking
- Delayed Development Marking

Address-Encoded-Port Marking

Destination port is derived from address bits.

Limitation: not all 16-bit port space is useable. We can use redundant marking to increase accuracy.



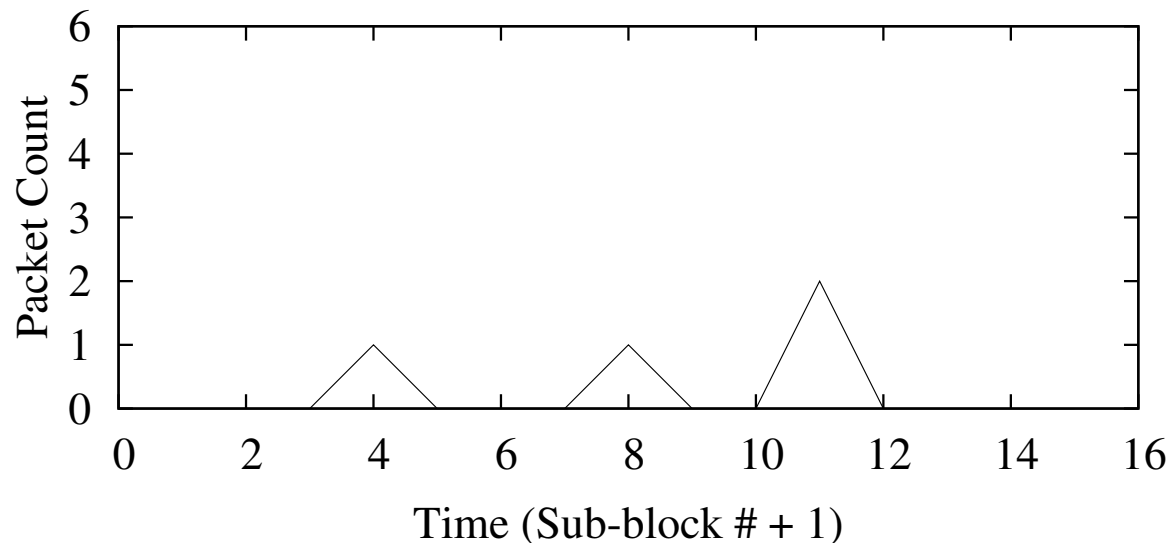
Time Series Marking

- Used in conjunction with other marking mechanisms.
- Each sub-block are marked within time resolution window to allow recovering the sub-block from the feedback.

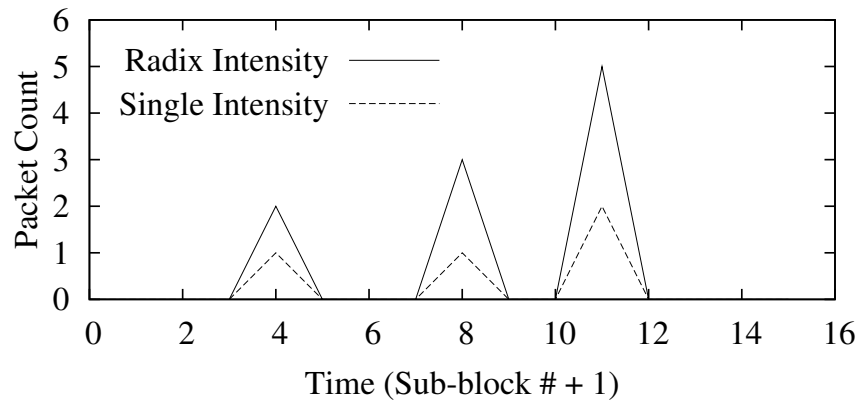
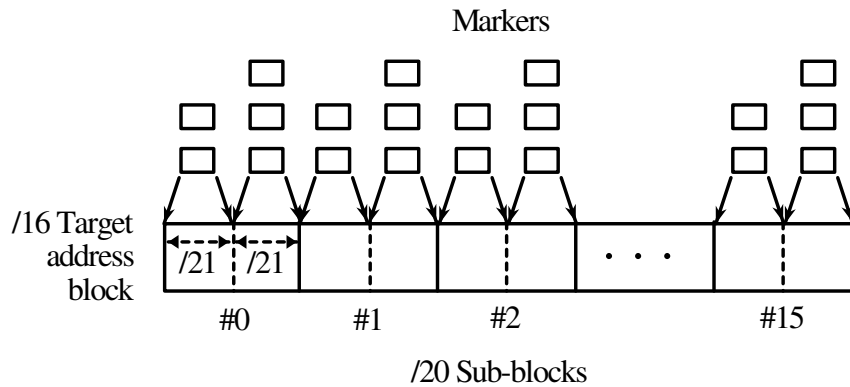
Uniform Intensity Marking

Addresses are marked with the same intensity.

Mark one sub-block per time unit, marking all addresses from it with a single marker.



Radix-Intensity Marking



Sensor Count	Sensor Location (Block #)			Feedback Intensity
	first sensor	second sensor	third sensor	
0	—	—	—	0
1	0	—	—	2
	1	—	—	3
2	0	0	—	4
	0	1	—	5
	1	1	—	6
3	0	0	0	6
	0	0	1	7
	0	1	1	8
	1	1	1	9

Radix Port Marking

If multiple ports are available for marking, a port pair can be assigned to toggle an address bit on or off.

Delayed Development Marking

Used for “Top-N” reports.

2 phases:

- exposure
 - leave hidden traces in feedback using minimal intensity marking
- development
 - high intensity marking (within the retention time)

Obvious Countermeasures

- Provide less information
- Throttle the information
- Introducing explicit noise
- Disturbing Mark-Examine-Update Cycle
- Marking detection
- Sensor scale and placement

Conclusions

- Secrecy of the monitored addresses is essential to the effectiveness of the sensor network.
- Passive Internet threat monitors are subject of detection attacks that can uncover their locations.
- *“Continuing efforts to better understand and protect passive threat monitors are essential for the safety of the Internet”.*

Can we do this without
“summaries”?