Bloom Filter for Network Security

Haipeng Dai

haipengdai@nju.edu.cn
313 CS Building
Department of Computer Science and Technology
Nanjing University

Bloom Filters

- Given a set $S = \{x_1, x_2, x_3, \ldots, x_n\}$ on a universe U, want to answer queries of the form:

$$Is\ y \in S.$$

- Bloom filter provides an answer in
  - "Constant" time (time to hash).
  - Small amount of space.
  - But with some probability of being wrong.

- Alternative to hashing with interesting tradeoffs.

Bloom Filters

Start with an m bit array, filled with 0s.

B	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Hash each item x_j in S k times. If $H_i(x_j) = a$, set $B[a] = 1$.

B	0	1	0	0	1	0	1	0	0	1	1	1	0	1	1	0

To check if y is in S, check B at $H_i(y)$. All k values must be 1.

B	0	1	0	0	1	0	1	0	0	1	1	1	0	1	1	0

Possible to have a false positive; all k values are 1, but y is not in S.

B	0	1	0	0	1	0	1	0	0	1	1	1	0	1	1	0

n items $m = cn$ bits k hash functions

False Positive Probability

- Pr(specific bit of filter is 0) is

$$p' \equiv (1 - 1/m)^{kn} \approx e^{-kn/m} \equiv p$$

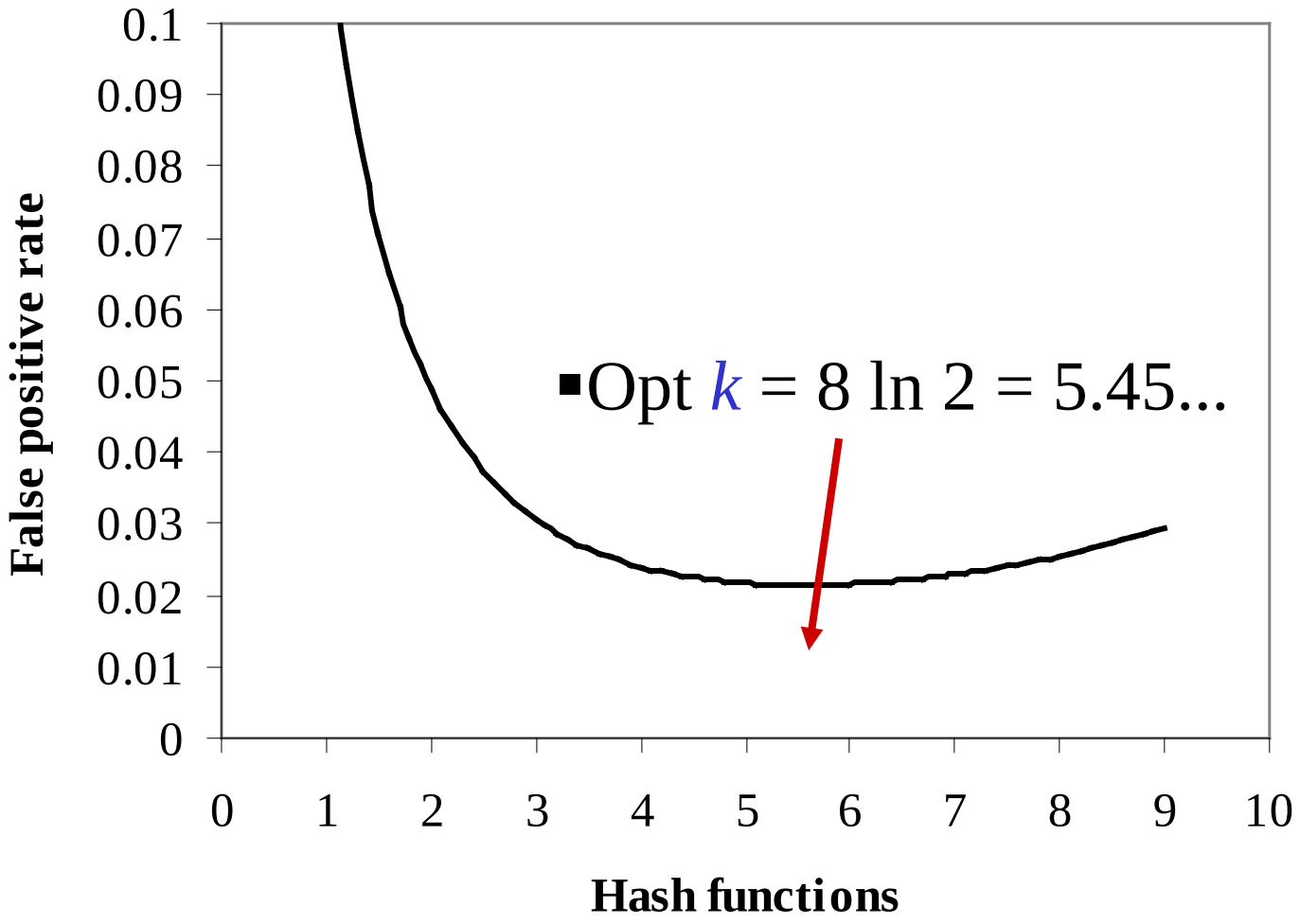
- If ρ is fraction of 0 bits in the filter then false positive probability is

$$(1-\rho)^k \approx (1-p')^k \approx (1-p)^k = (1-e^{-k/c})^k$$

- Approximations valid as ρ is concentrated around $E[\rho]$.
  - Martingale argument suffices.
- Find optimal at $k = (\ln 2)m/n$ by calculus.
  - So optimal fpp is about $(0.6185)^{m/n}$

n items $m = cn$ bits k hash functions

Example



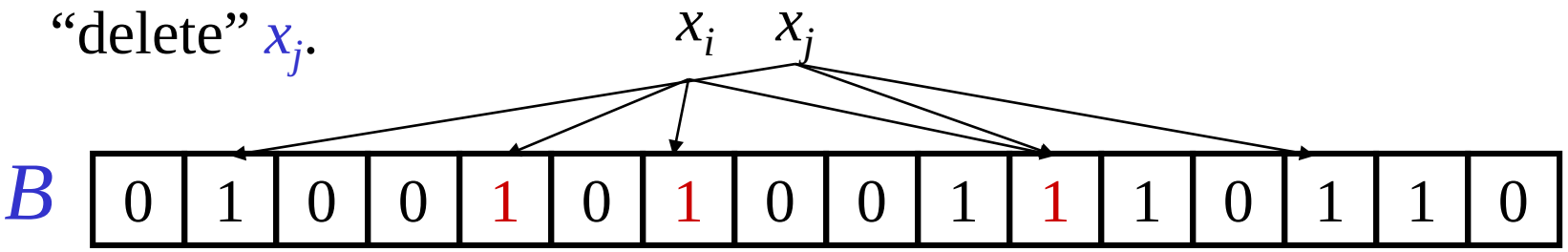
$m/n = 8$

n items $m = cn$ bits k hash functions

Handling Deletions

- Bloom filters can handle insertions, but not deletions.

- If deleting x_i means resetting 1s to 0s, then deleting x_i will "delete" x_j.



Counting Bloom Filters

Start with an m bit array, filled with 0s.

B	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

Hash each item x_j in S k times. If $H_i(x_j) = a$, add 1 to $B[a]$.

B	0	3	0	0	1	0	2	0	0	3	2	1	0	2	1	0

To delete x_j decrement the corresponding counters.

B	0	2	0	0	0	0	2	0	0	3	2	1	0	1	1	0

Can obtain a corresponding Bloom filter by reducing to 0/1.

B	0	1	0	0	0	0	1	0	0	1	1	1	0	1	1	0

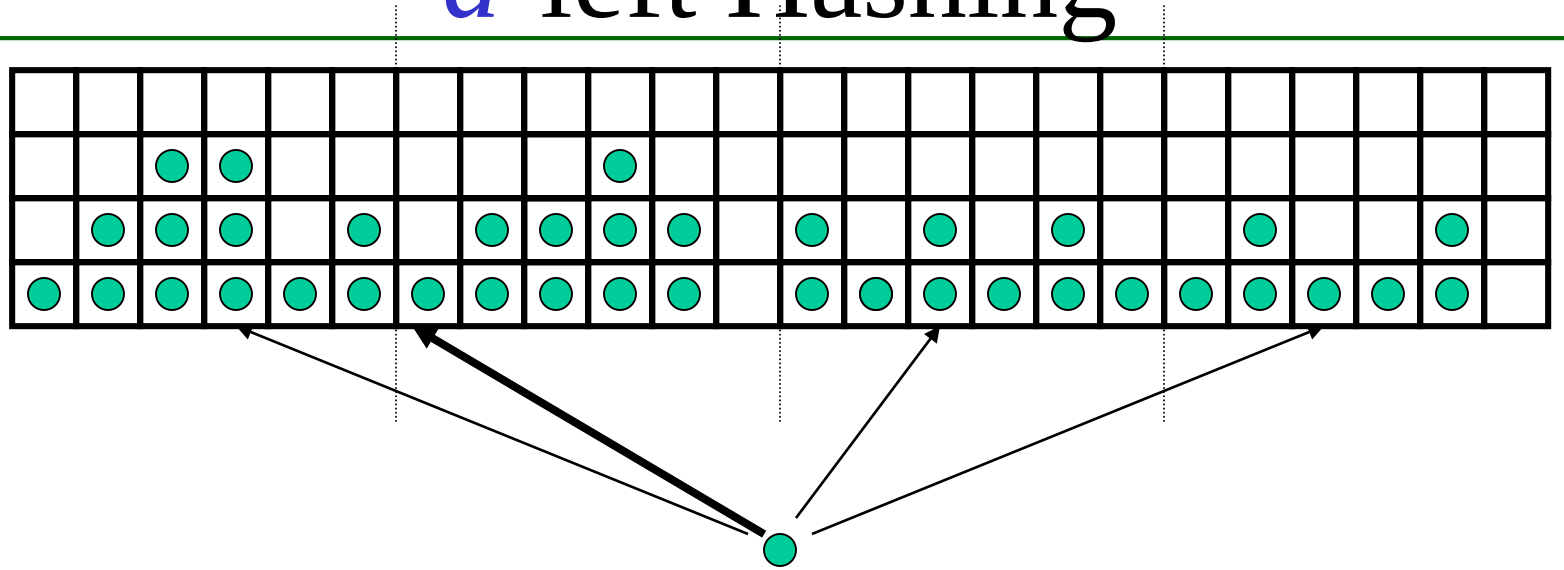
Counting Bloom Filters: Overflow

- Must choose counters large enough to avoid overflow.
- Poisson approximation suggests 4 bits/counter.
  - Average load using $k = (\ln 2)m/n$ counters is $\ln 2$.
  - Probability a counter has load at least 16:

$$\approx e^{-\ln 2}(\ln 2)^{16}/16! \approx 6.78\text{E}-17$$

- Failsafes possible.
- We assume 4 bits/counter for comparisons.

d-left Hashing



- Split hash table into d equal subtables.
- To insert, choose a bucket uniformly for each subtable.
- Place item in a cell in the least loaded bucket, breaking ties to the left.

Properties of *d*-left Hashing

- Analyzable using both combinatorial methods and differential equations.
  - Maximum load very small: $O(\log \log n)$.
  - Differential equations give very, very accurate performance estimates.
- Maximum load is extremely close to average load for small values of d.

Power of 2 choices!

Example of *d*-left hashing

- Consider 3-left performance.

Average load 4

Load 0	2.3e-05
Load 1	6.0e-04
Load 2	1.1e-02
Load 3	1.5e-01
Load 4	6.6e-01
Load 5	1.8e-01
Load 6	2.3e-05
Load 7	5.6e-31

Average load 6.4

Load 0	1.7e-08
Load 1	5.6e-07
Load 2	1.2e-05
Load 3	2.1e-04
Load 4	3.5e-03
Load 5	5.6e-02
Load 6	4.8e-01
Load 7	4.5e-01
Load 8	6.2e-03
Load 9	4.8e-15

A taxonomy of Bloom filter Application in Network Security

Environment	Application		ED	IP	ID
Wireless networks	– Authentication	– Message authentication	×	√	√
		– Node authentication	×	√	√
	– Anonymity and privacy-preserving	– Anonymous routing	×	√	√
		– privacy-preserving	√	√	√
	– Firewalling	– Mesh firewall	×	√	√
		– 3G firewall	×	×	√
	– Tracebacking		√	×	×
	– Misbehavior detection		×	√	√
	– Replay protection		×	√	√
	– Node replication detection		×	√	√
Wired networks	– String matching	– Standard BF-based schemes	√	×	√
		– Counting BF-based schemes	√	×	√
		– Bloomier based schemes	√	×	√
		– Standard and counting BF-based schemes	√	×	√
	– IP tracebacking	– Logging-based IP tracebacking	×	×	√
		– Marking-based IP tracebacking	×	√	×
		– Logging- and Marking-based IP tracebacking	×	√	√
	– Spam filtering and e-mail protection	– Spam filtering	√	×	×
		– E-mail Server protection	√	×	×
	– DoS and DDoS detection	– DoS and DDoS attacks addressing	×	√	√
		– DNS attacks addressing	×	×	√
		– SYN flooding attacks addressing	×	×	√
	– Anomaly detection		×	×	√

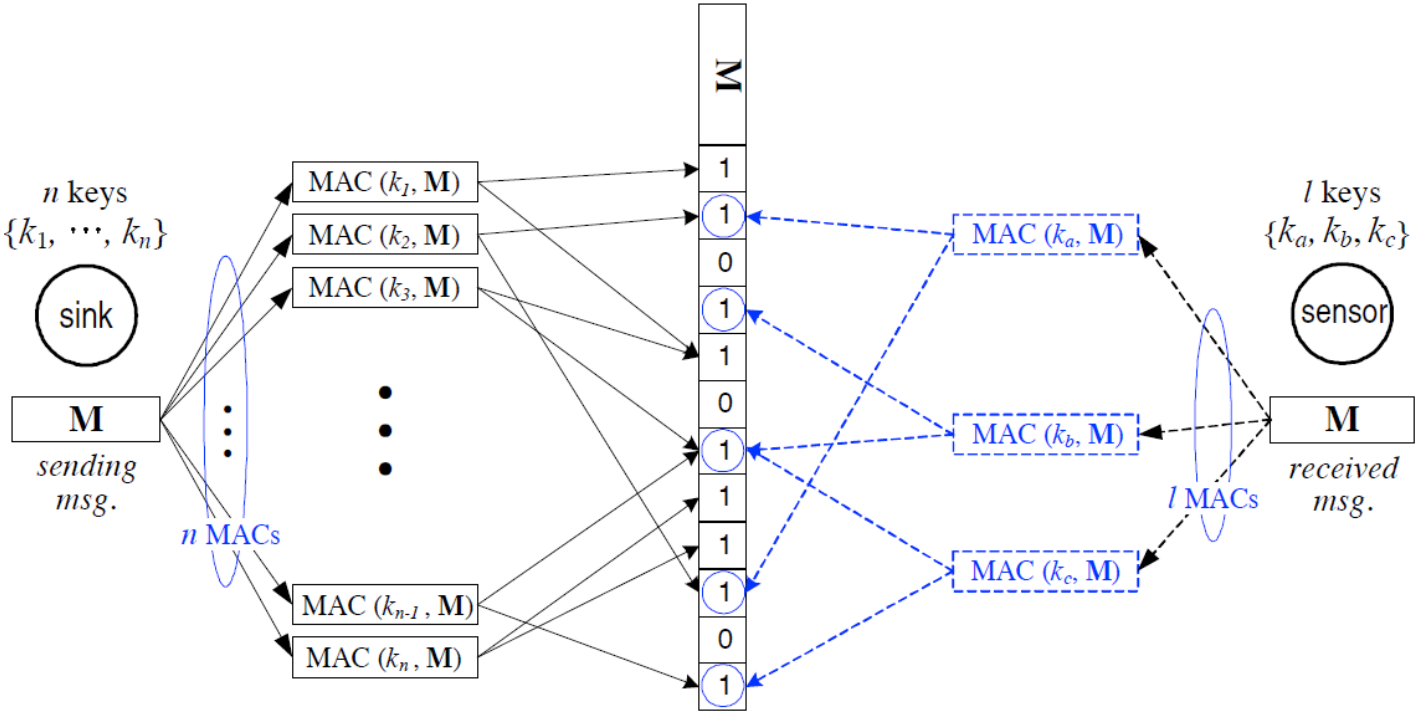
ED (end-devices), **ID** (intermediate-devices), **IP** (in-packet)

Wireless Networks: Authentication

- A protocol to authenticate messages flooded in large-scale WSNs.
- Each sensor node is preloaded with *l* symmetric keys and *k* hash functions. The sink also maintains *k* hash functions and *n* keys.
- The sink constructs *n* message authentication codes (MACs) using the *n* keys. These resulting MACs are then inserted into the BF. Subsequently, the BF is flooded along with the message in the whole network. When the message arrives at each node, *l* MACs are constructed again in the same way by using *l* keys stored in the node. These *l* MACs are sought in the arrived BF. When a zero value is found, the message is assumed to be invalid; otherwise, it is sent to the neighbor nodes. Moreover, they proposed to use compressed Bloom filters for reducing false positive rate and the size of BF.

▪J.H. Son, H. Luo, S.W. Seo, Authenticated flooding in large-scale sensor networks, in: IEEE International Conference on Mobile Adhoc and Sensor Systems (MASS), 2005, pp. 536–543.

Wireless Networks: Authentication



BMAP Bloom filter construction and verification

- J.H. Son, H. Luo, S.W. Seo, Authenticated flooding in large-scale sensor networks, in: IEEE International Conference on Mobile Adhoc and Sensor Systems (MASS), 2005, pp. 536–543.

Wireless Networks: Node Authentication

- In data-centric sensor networks, key BFs have been used for generating query messages and enhancing the privacy of information against various attacks.
- In order to avoid performing membership test by the attacker, the IDs of all storage cells are encrypted using cell keys, and then hashed and inserted into the BF. In this scheme, when the query message arrives at a cell, it is sought in the key BF. If the neighbor is in the BF, the message is sent to that neighbor.

▪M. Shao, S. Zhu, W. Zhang, G. Cao, Y. Yang, pDCS: security and privacy support for data-centric sensor networks, IEEE Transactions on Mobile Computing 8 (8) (2009) 1023–1138.

Wireless Networks: Anonymity

- A secure anonymous routing protocol for clustered ad hoc network.
- Because of using BFs, this protocol does not require any public key operation.
- In this scheme, BF has been used to both anonymous data transmission and anonymous route discovery. The identities of the nodes in the route from source to destination are mapped into the BF. Therefore, to hide the ID, the node only needs to hash its ID by k hash functions and set the corresponding bits in the BF. In the data transmission phase, the BF containing routing information is sent along with the message.

▪S. Chen, L. Xu, Z. Chen, Secure anonymous routing in trust and clustered wireless ad hoc networks, in: Second International Conference on Communications and Networking in China, 2007, pp. 994–998.

Wireless Networks: Firewall

- In mesh networks, firewall schemes are essential to classify and filter traffic.
- Each node adopts a Bloom filter to represent all packets accepted by the node, and then distributes the Bloom filter to all nodes in the network.
- When a node wants to forward a packet, it queries the packet from all Bloom filters it has received from other nodes. If it is found, the packet is forwarded; otherwise, it is discarded. In this scheme, a firewall rule is presented by the set $R = \{sourceIP, destinationIP, sourcePort, destinationPort\}$.

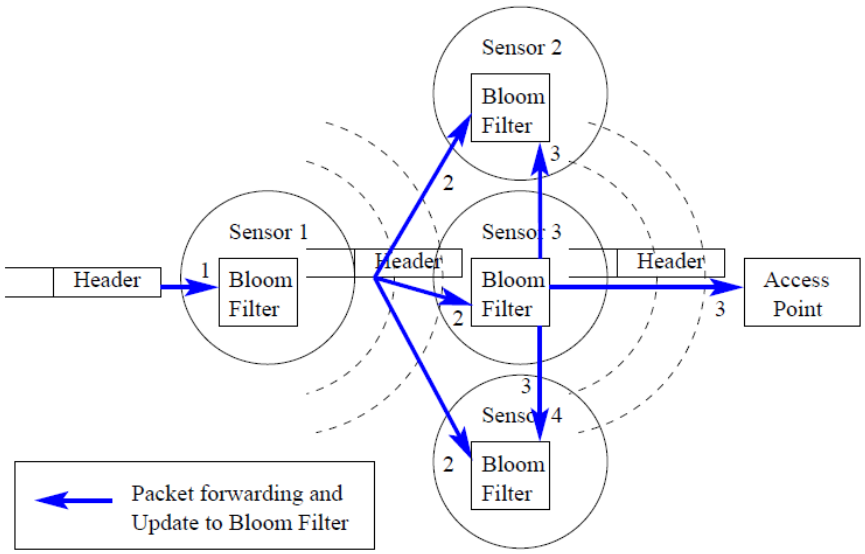
▪L. Maccari, R. Fantacci, P. Neira, R.M. Gasca, Mesh network firewalling with Bloom filters, in: IEEE International Conference on Communications (ICC), 2007, pp. 1546–1551.

Wireless Networks: Tracebacking

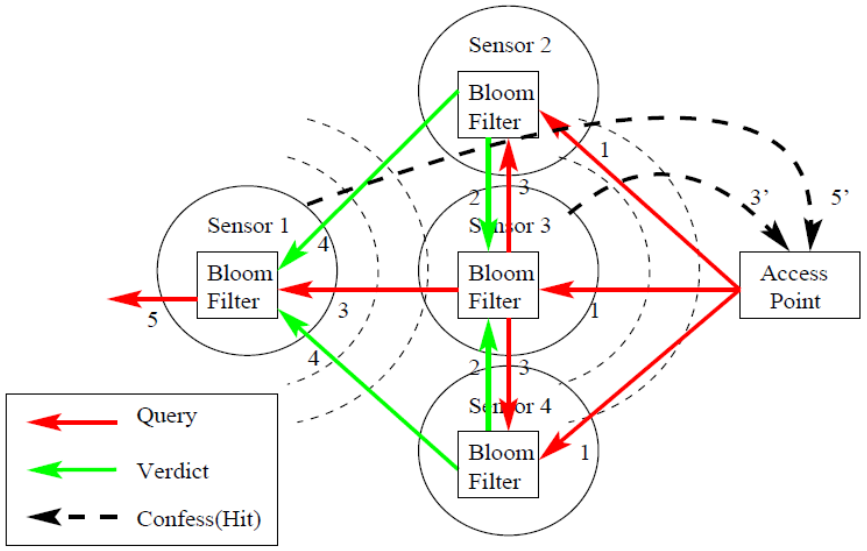
- Cooperative sensors utilize multi-dimensional BFs, named space–time Bloom filters, to maintain the attributes of the packets in order to traceback the attacker packets.
- In addition to the packet information, the ID of the forwarding node is also added to the input string of the hash functions. When passing a packet through a sensor, this packet is mapped into the BF of the sensor. Later, the BF will be used to reconstruct the attack graph. However, this scheme has been designed for a small sensor network, and it has no feature to recompute the attack path.

▪D. Sy, L. Bao, CAPTRA: coordinated packet traceback, in: Proceedings of the fifth international conference on Information Processing in Sensor Networks, 2006, pp. 124–135.

Wireless Networks: Tracebacking



Space-Time Bloom Filter for Packet Tracking



Packet Tracing Process using Space-Time Bloom Filter

▪D. Sy, L. Bao, CAPTRA: coordinated packet traceback, in: Proceedings of the fifth international conference on Information Processing in Sensor Networks, 2006, pp. 124–135.

Wireless Networks: Node replication detection

- A *cell forwarding* and *cross forwarding* to improve the node replica detection in WSNs.
- The proposed schemes use BFs to store the information stored at the sensors to reduce the memory usage of intermediate nodes in Line-Selected Multicast (LSM). These schemes use two BFs, one for storing ID of the nodes (*ID filter*) and the other one for keeping the locations (*location filter*). Subsequently, these two BFs will be utilized by the nodes to detect conflicting claims in the subsequent operations.
- These schemes are based on distributing the location claims to relay nodes in the network. Since the location claim is distributed to many nodes in the network, it increases a chance to detect the node replication.
- A lot of communication overhead, despite of using BFs, because they try to forward the location claims to intermediate nodes which act as a witness node

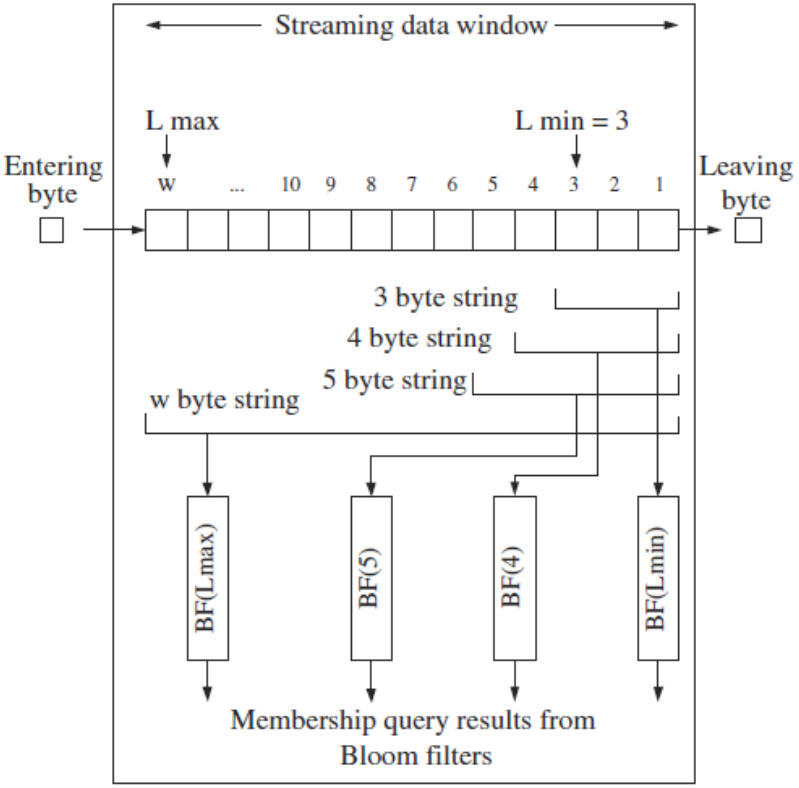
▪M. Zhang, V. Khanapure, S. Chen, X. Xiao, Memory efficient protocols for detecting node replication attacks in wireless sensor networks, in: 17th IEEE International Conference on Network Protocols (ICNP), 2009, pp. 284–293.

Wired Networks: String Matching

- A set of hardware BFs have been used in parallel to verify which input flow matches against a set of predefined signatures. In this architecture, each BF maintains the signatures of a particular length. Therefore, each BF is utilized to find the strings of a specific length in the input stream.
- In each run, a window of the data stream is inspected by the system. If each of these BFs detects a match, the string is delivered to the analyzer to perform exact matching; otherwise, the next byte of the stream is processed. If there are multiple matches for different lengths, the longest one is selected.

▪S. Dharmapurikar, P. Krishnamurthy, T. Sproull, J. Lockwood, Deep packet inspection using parallel bloom filters, IEEE Micro 24 (1) (2004) 52–61.

Wired Networks: String Matching



A window of streaming data containing strings of length from $L_{min} = 3$ to $L_{max} = w$

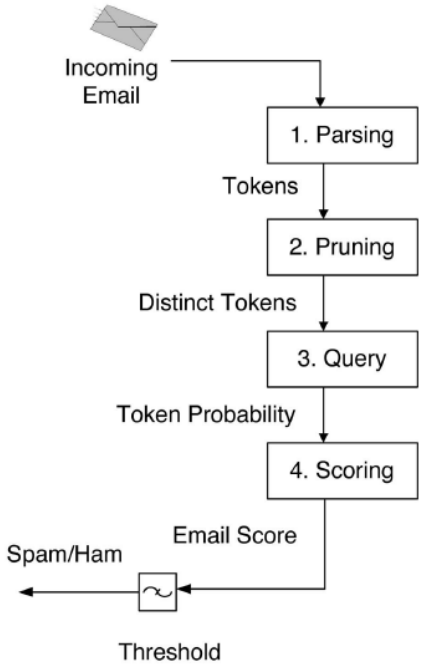
▪S. Dharmapurikar, P. Krishnamurthy, T. Sproull, J. Lockwood, Deep packet inspection using parallel bloom filters, IEEE Micro 24 (1) (2004) 52–61.

Wired Networks: Spam filtering

- An approximate method was proposed to speed up spam filter processing.
- It utilizes BFs in two techniques, called *approximate pruning* and *approximate lookup*.
- In the former case, an *m*-bit BF is used to maintain the tokens resulting from parsing each message in order to reduce the delay of searching repeated tokens when performing approximate membership test.
- In the latter case, a *2D* BF is used to reduce memory requirement by supporting information retrieval. The authors reported that the scheme has shown a factor of 6x speedup with similar false negative rates and identical false positive rates compared to the original filters.

▪Z. Zhong, K. Li, Speedup statistical spam filter by approximation, IEEE Transactions on Computers 60 (1) (2010) 120–134.

Wired Networks: Spam filtering



Stage	Bogofilter-style Implementation	Speedup
Pruning [Optional]	Traverse Token List	Membership Test with a traditional Bloom filter
Query	Database lookup	Approximate lookup with an extended Bloom filter
Scoring	Intensive arithmetic operation (both CPU and MEM)	Approximate table lookup

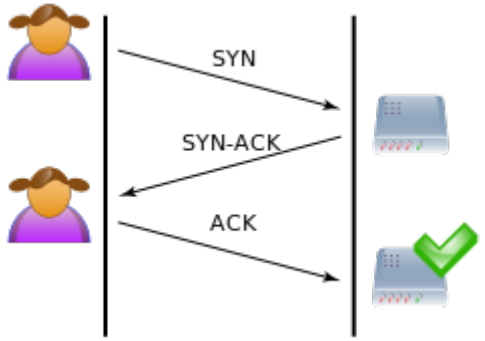
(b)

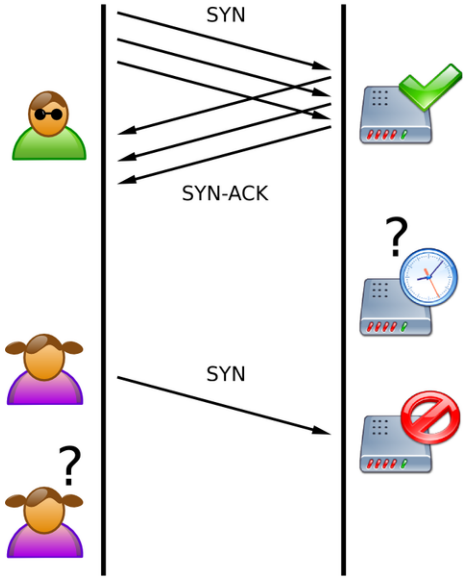
Bayesian filter stages: (a) The stage with its output and (b) the speedup techniques corresponding to the stages.

- Z. Zhong, K. Li, Speedup statistical spam filter by approximation, IEEE Transactions on Computers 60 (1) (2010) 120–134.

Wired Networks: DoS and DDoS attacks detection

- SYN Flood Attack





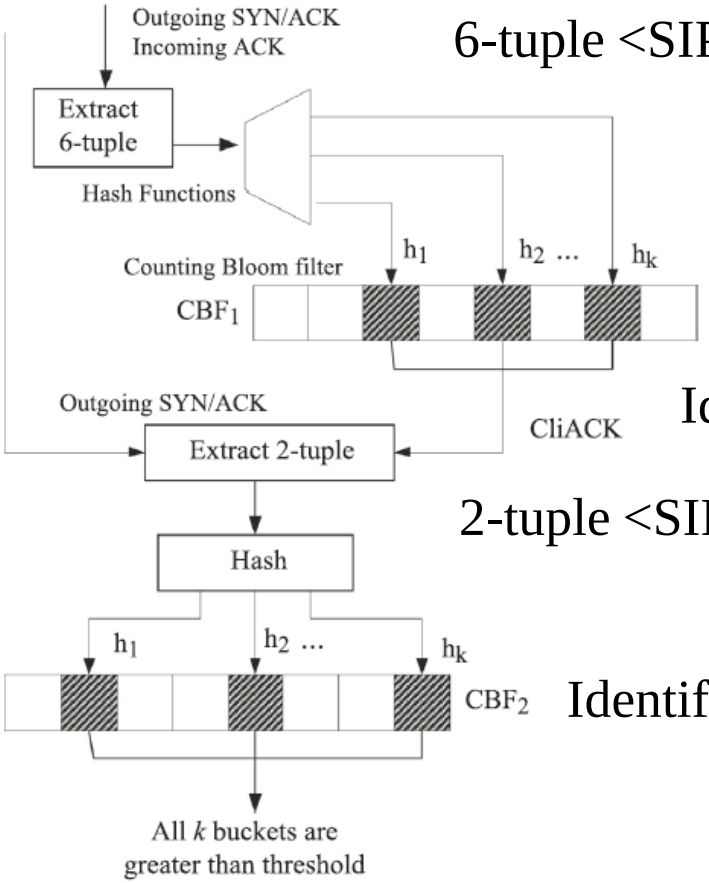
- C. Sun, C. Hu, Y. Tang, B. Liu, More accurate and fast SYN flood detection, in: Proceedings of 18th IEEE International Conference on Computer Communications and Networks (ICCCN), 2009, pp. 1–6.

Wired Networks: DoS and DDoS attacks detection

- SACK uses Client ACK (CliACK) packets to detect SYN flooding attacks for accurate and fast router-based detection.
- SACK applies SYN/ACK–CliACK pair to detect the victim server. Two CBFs are used to maintain the full information of TCP connection, including the 6-tuple of the output SYN/ACK packet, i.e., source and destination's IP addresses, source and destination's ports, sequence number and ACK sequence number, and also the same 6-tuple of the input ACK packet.

- C. Sun, C. Hu, Y. Tang, B. Liu, More accurate and fast SYN flood detection, in: Proceedings of 18th IEEE International Conference on Computer Communications and Networks (ICCCN), 2009, pp. 1–6.

Wired Networks: DoS and DDoS attacks detection



6-tuple <SIP, DIP, SP, DP, SEQ,ASEQ>

Identify Client ACK

2-tuple <SIP, SP>

Identify DDoS attacks

The logic architecture of SACK

- C. Sun, C. Hu, Y. Tang, B. Liu, More accurate and fast SYN flood detection, in: Proceedings of 18th IEEE International Conference on Computer Communications and Networks (ICCCN), 2009, pp. 1–6.

Summary

- Bloom filter variants and their contribution to network security; false positive (FP), false negative (FN).

Bloom filter	FP	FN	Security usage	Application domain
Standard Bloom filter	Yes	No	Yes	Authentication, Firewalling, Anomaly detection, Tracebacking, Node replication detection, Anonymous routing and privacy-preserving, String matching, DoS and DDoS addressing, Email protection, Misbehavior detection
Adaptive Bloom filter	Yes	No	No	
Bloomier filter	Yes	No	Yes	String matching
Compressed Bloom filter	Yes	No	Yes	Authentication, IP tracebacking
Counting Bloom filter	Yes	No	Yes	Firewalling, String matching, Email protection, SYN flooding addressing
Decaying Bloom filter	Yes	No	No	
Deletable Bloom filter	Yes	No	No	
Distance-sensitive Bloom filters	Yes	Yes	No	
Dynamic Bloom filter	Yes	No	Yes	Node replication detection
Generalized Bloom filter	Yes	Yes	Yes	IP tracebacking
Hierarchical Bloom filter	Yes	No	Yes	IP tracebacking
Retouched Bloom filter	Yes	Yes	No	
Scalable Bloom filter	Yes	No	No	
Space Code Bloom filter	Yes	No	Yes	IP tracebacking
Spectral Bloom filter	Yes	No	No	
Split Bloom filter	Yes	No	No	
Stable Bloom filter	Yes	Yes	No	
Weighted Bloom filter	Yes	No	No	