High-Accuracy Updatable Bloom Filters for Robust Network Security in Programmable Networks

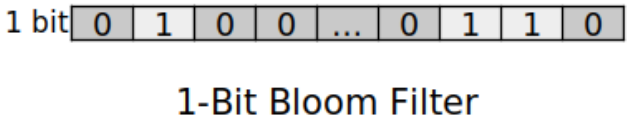
Mehmet Emin Şahin

THE LINUX FOUNDATION

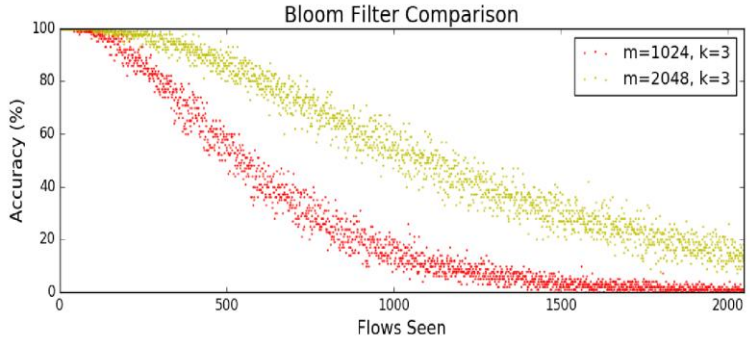
Why Bloom Filter (BF) ?

- Programmable data planes are powerful in terms of flexibility, scalability, and line-rate packet processing capabilities.
- But they offer limited memory space due to high-cost restrictions, known as TCAM / SRAM.
- Bloom filter (BF) is a probabilistic data structure that works space-efficient and enables fixed query time.
- But BF may generate false-positive results.

Bloom Filters in Network Flow Tracking



1-Bit Bloom Filter

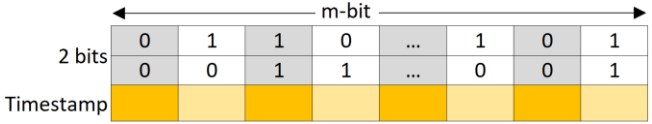


*Tracking network flows with P4 (2018)

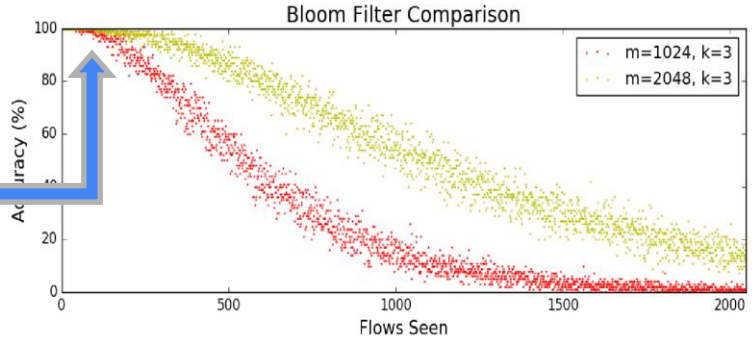
As new elements are added, the accuracy decreases since the BFs fill up over time.

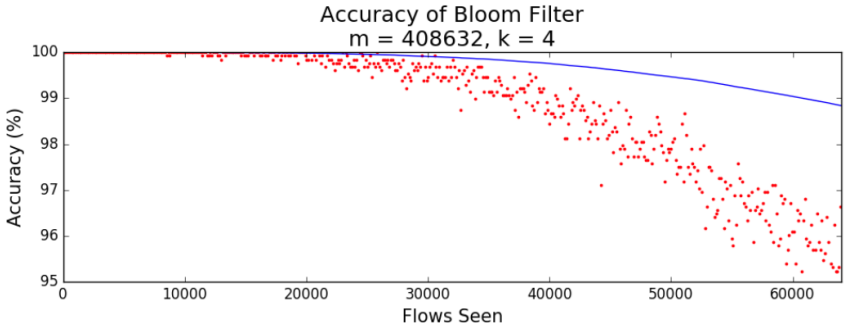
Each protocol in the network is associated with predefined timeout values, why don't we construct a data structure that ensures time-based up-to-dateness and efficiently removes outdated entries from the Bloom Filter?

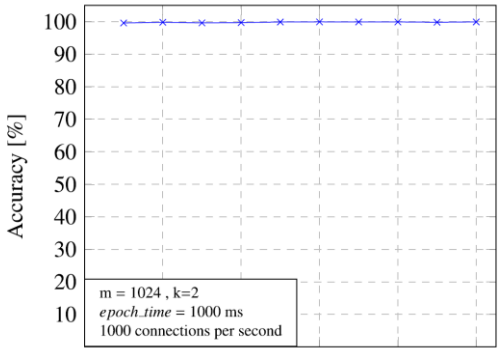
Updatable Bloom Filter (UBF)



Example of 2-bit wide, m-bit long UBF

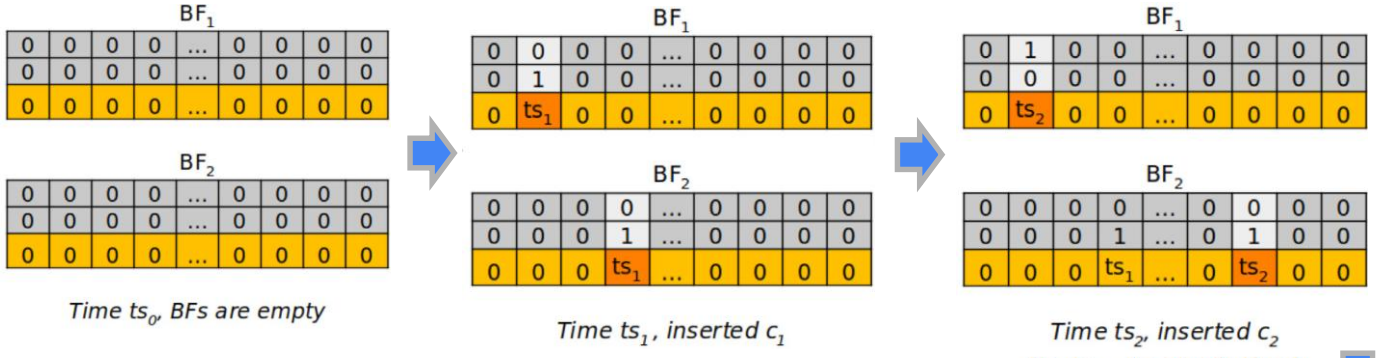






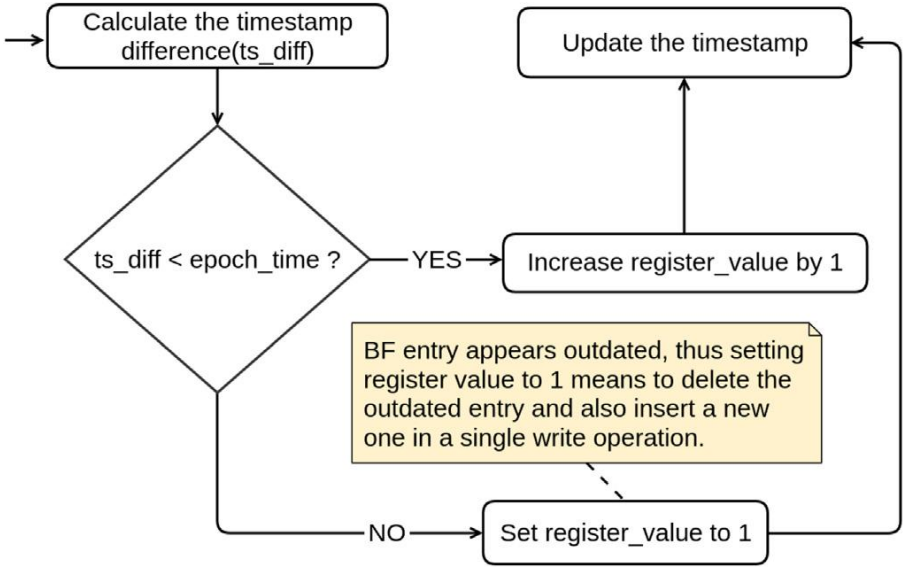
How UBF Works ?

Time	Time Stamp	Connections
0	ts_0	0
t	ts_1	c_1
$2t$	ts_2	c_2
$3t$	ts_3	c_3
$5t$	ts_4	c_4
$6t$	ts_5	c_5

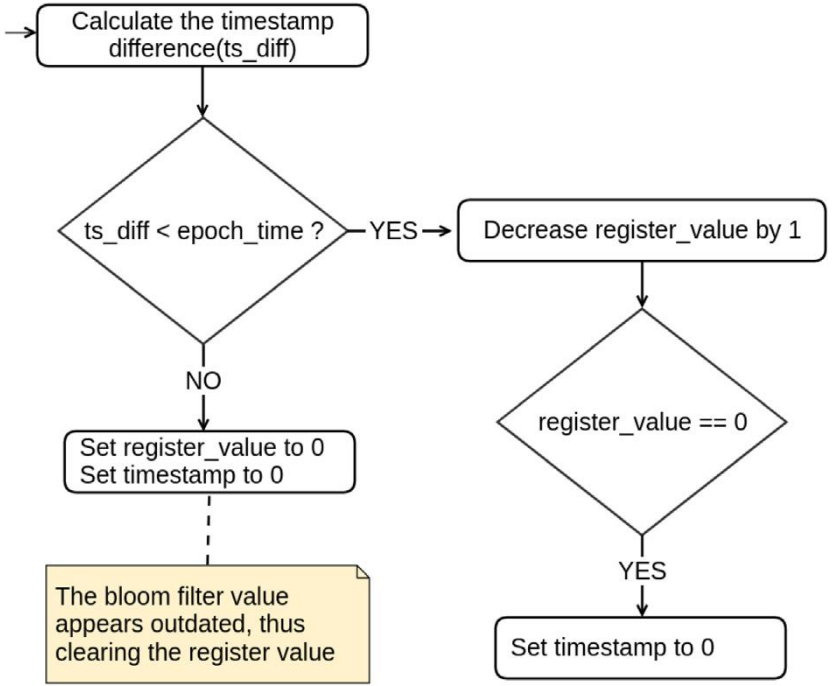


$t < epoch\_time < 2t$

Add and Delete Operations in UBF

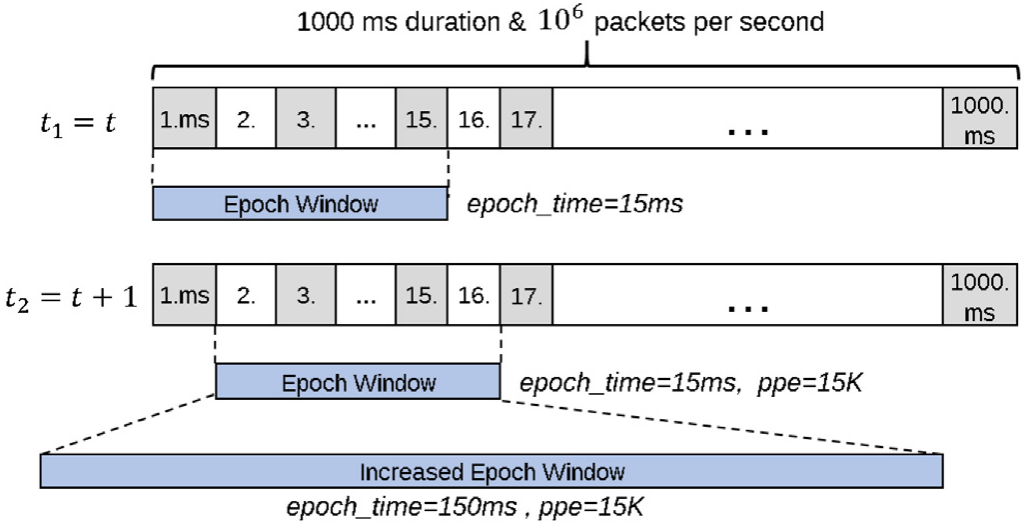


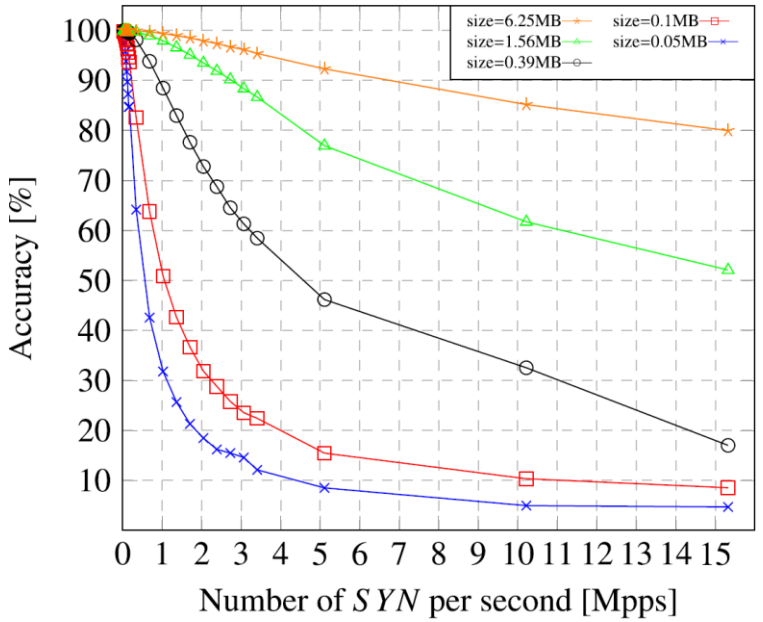
Add new element to UBF



Delete element from UBF

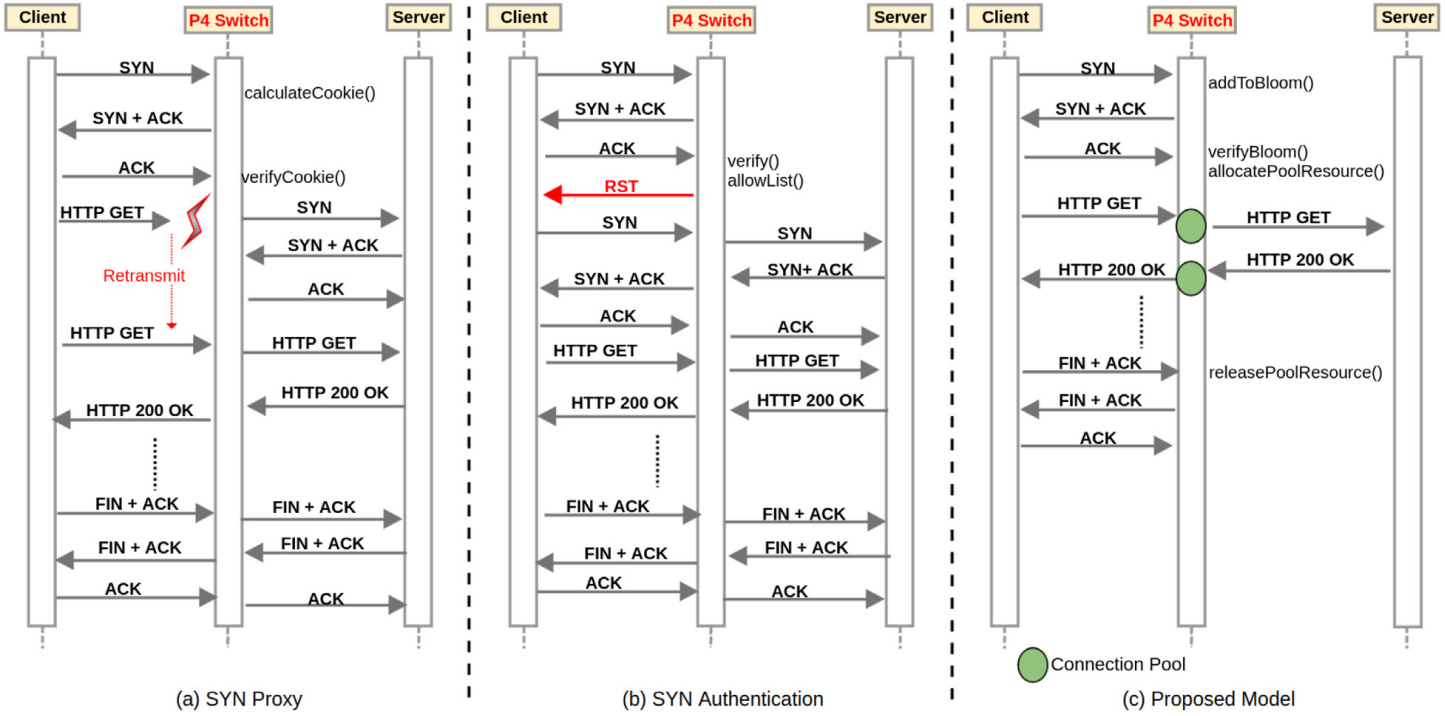
Performance Measurement of UBF





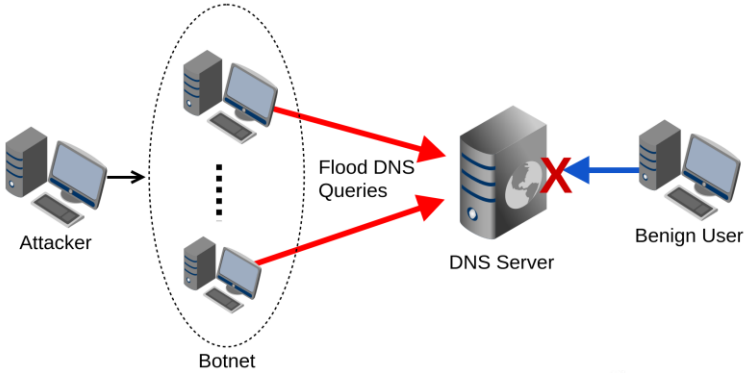
Example illustration of increasing epoch time for 1Mpps

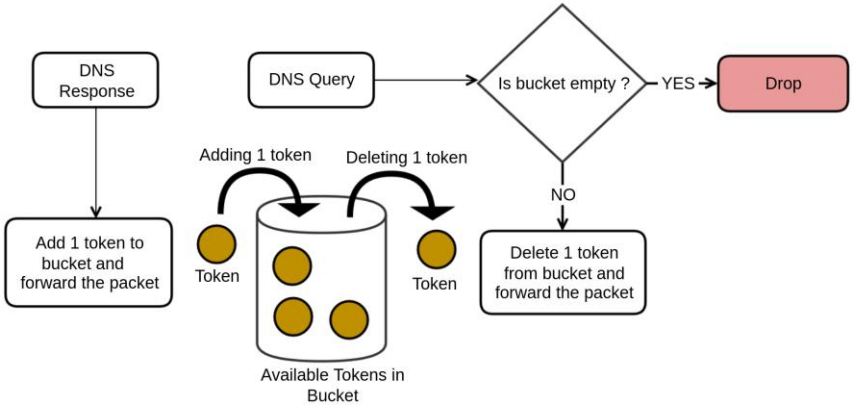
Use Case 1: SYN Flood Attacks



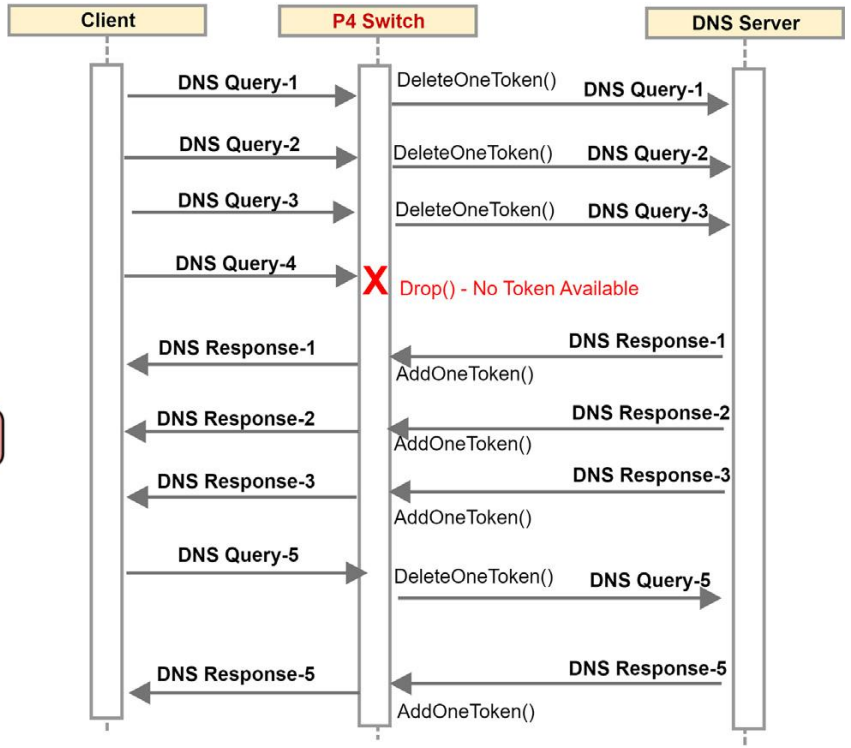
* ConPoolUBF: Connection pooling and updatable Bloom filter based SYN flood defense in programmable data planes

Use Case 2: DNS Query Flood Attack



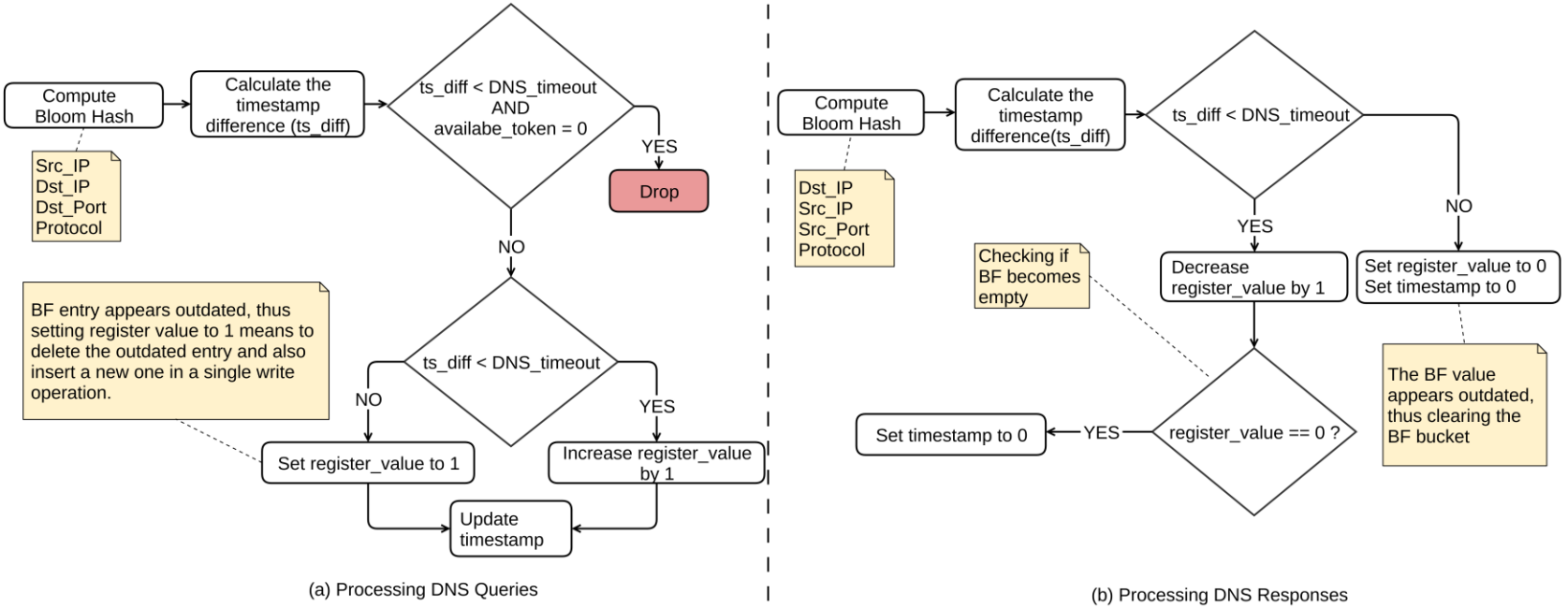


Modified Token Bucket Algorithm



cql = 3

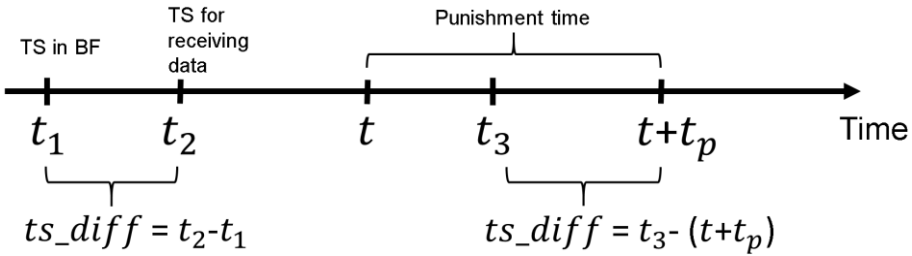
Adaptive DNS Rate Limiter

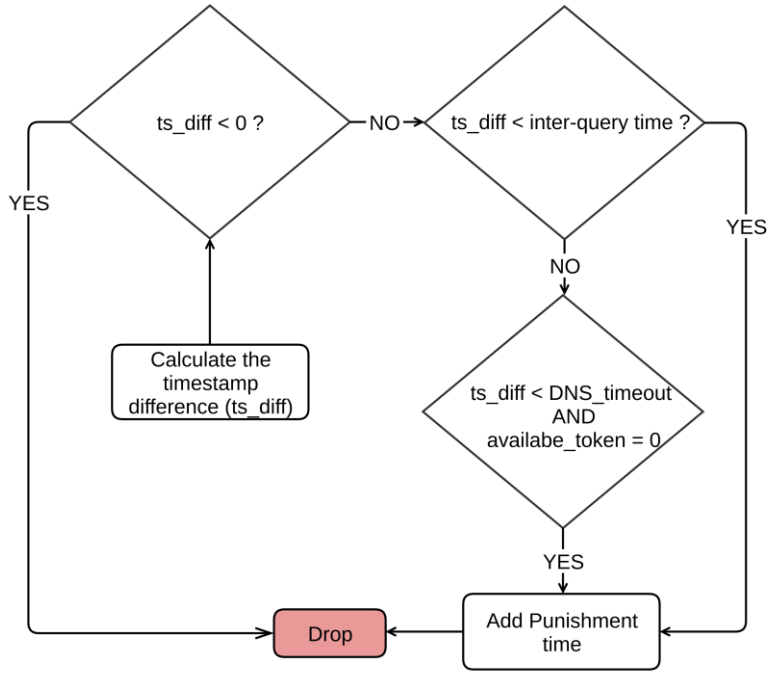


Time-Based Punishment with UBF

Attacker Detection Methods

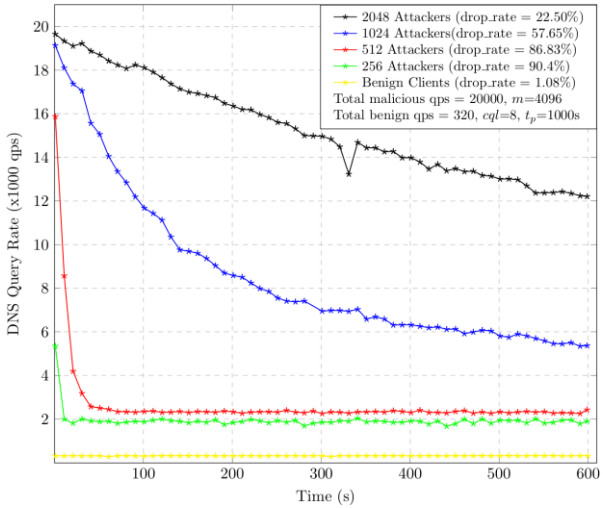
- Attackers continue querying despite having no token
- Sending DNS queries more frequently than the inter-query time threshold



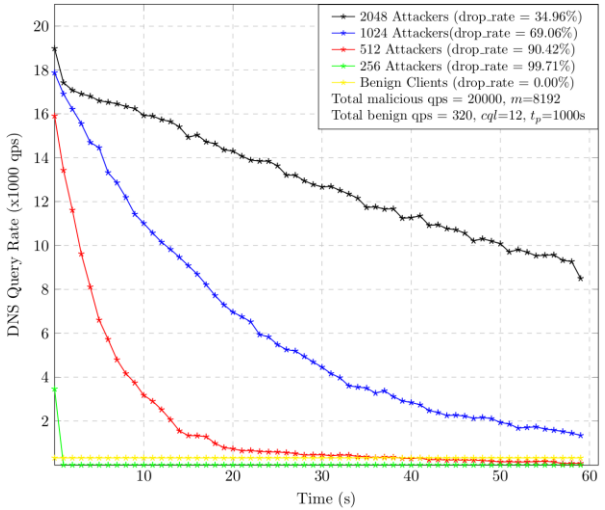


* t_p = *punishment time*

Rate Limiter Tests

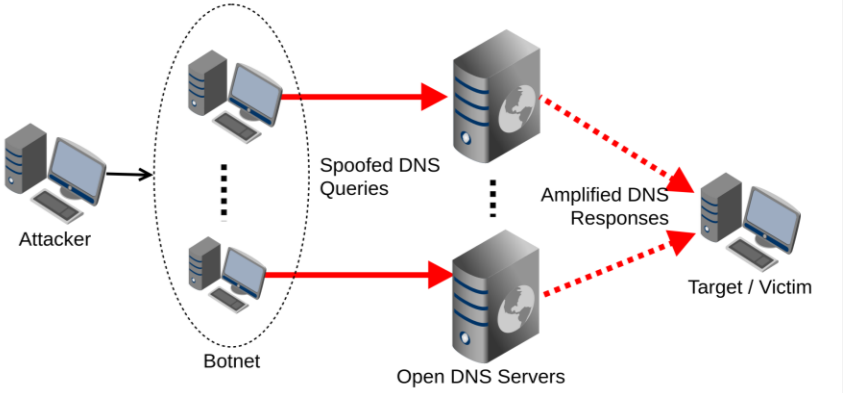


DDoS Performance for Authoritive Name Server
(μ_{ys}=0,04ms , σ_{ys}=0,045ms)

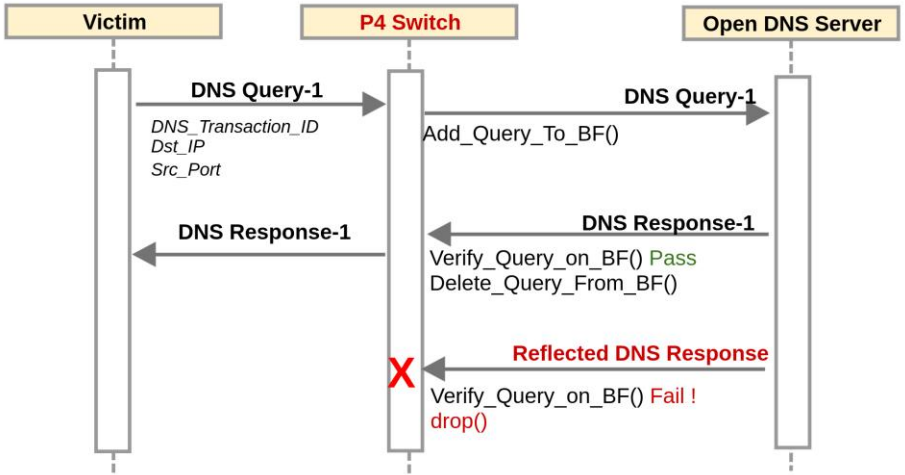


DDoS Performance for Recursive Name Server
(μ_{ys}=405ms , σ_{ys}=612ms)

Use Case 3: DNS Firewall



DNS Amplification

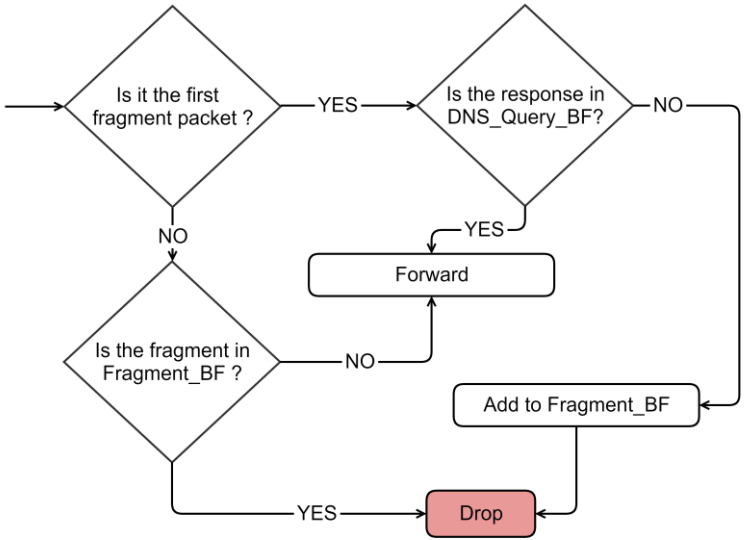


UBF Based DNS Firewall Packet Processing (*one-to-one mapping*)

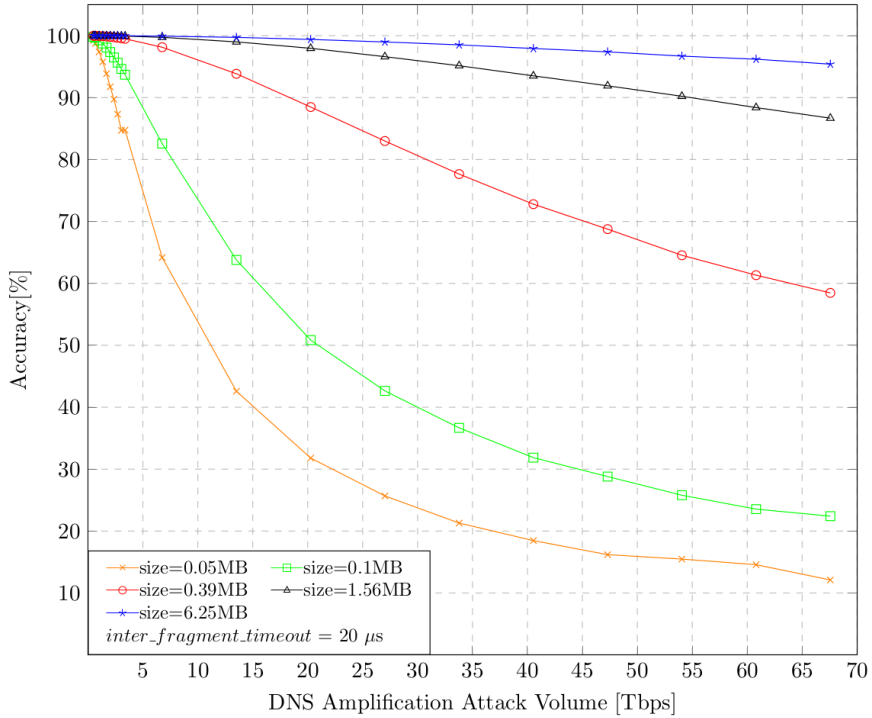
Tracking of Fragmented DNS Responses

How to incorporate fragmented DNS packets into stateful tracking ?

- Extension Mechanisms for DNS (EDNS0) enables a DNS server to send DNS responses larger than 1500 bytes through IP fragmentation.
- Fragmented DNS responses do not adhere to a one-to-one mapping. Also the upper headers only appear in the first fragment in IP fragmentation
- Solution: A second UBF was created to track fragmented packets named *Fragment_BF*.



UBF Performance on DNS Amplification



UBF accuracy for varying DNS amplification attack volumes

* HELP4DNS: Leveraging the programmable data plane for effective and robust defense against DDoS attacks on DNS

Updatable Bloom Filter in Action

- Tracking of SYN packets in TCP 3-way handshake
- Adaptive rate limiting of DNS queries
- One-to-one mapping based tracking of DNS packets
- Tracking Fragmented DNS responses
- Time based punishment of attackers
- Measuring inter-query time

UBF is

- Simple in design
- Efficient in execution
- Robust against diverse of attack scenarios