UNIVERSITY OF SAN FRANCISCO

TarProbe: Detecting Strict Priority Queueing

Vahab Pournaghshband (vahab.p@usfca.edu)
Manali Patil (mmpatil@dons.usfca.edu)

Network and Security Research Laboratory
University of San Francisco

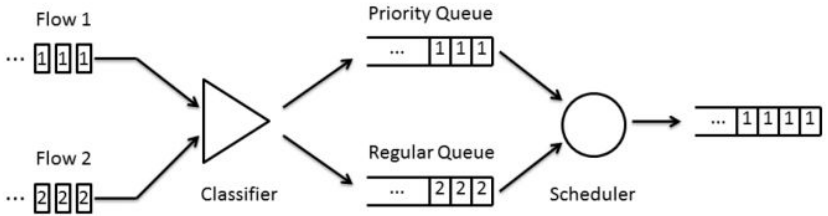
The 13th International Conference on Advanced Communications and Computation
INFOCOMP 2023

Introduction

- Priority queueing is a Quality-of-Service (QoS) feature available in most routers, for network operators to use.
- Net neutrality states that “All the Data Packets are equal and must be treated equally.”
- There are certain Internet service providers that can install packet discriminators or middleboxes such as priority queueing on routers that affect the end users’ traffic. This information is often hidden from end users.
- These middleboxes should not go unnoticed or undetected due to their significant impact on the performance of network applications, the accuracy of Internet measurements, and the effectiveness of network troubleshooting procedures.
- We propose *TarProbe*, a client/server network tool, that detects middleboxes with priority queueing enabled. Our proposed tool incorporates a loss-based approach utilizing a relative discrimination technique.
- The accuracy of TarProbe is verified in a simulated environment (ns-3).

What is Strict Priority Queueing?

- Strict Priority Queueing (SPQ) schedules traffic such that the high priority queue always get serviced first.



- If the priority queue is non-empty, the scheduler will service it first.
- The regular queue is serviced only when there are no packets in priority queue.
- Strict priority queueing may cause the regular priority queue traffic to starve.

Detection Methodology

- Each measurement round consists of two priority phases: high priority and low priority. In each priority phase, the sender sends a packet train that consists of carefully positioned sequence of high priority and low priority packets. The receiver records the lost packets in each train and reports any discrimination observed.
- In *detection phase*, if the median of the difference of the loss ratio between high priority packets in high priority phase and low priority packets in low priority phase is greater than some threshold, then prioritization is detected.

Detection Algorithm

Algorithm 2 Detection Engine

detectionEngine($\tau, \Gamma, \gamma, P, (\mathcal{P}_{L_j})_{j=1}^{P}, (\mathcal{P}_{H_j})_{j=1}^{P}, ((\mathcal{P}_{L_j}^{i}{}')_{j=1}^{P}, (\mathcal{P}_{H_j}^{i}{}')_{j=1}^{P})_{i=1}^{\Gamma}$)

1: **for** $i = 1$ **to** Γ **do**
2: $\quad \hat{p}_L^i \leftarrow$ estimateLossRatio$(n_I, P, (\mathcal{P}_{L_j}^{i}{}')_{j=1}^{P})$)
3: $\quad \hat{p}_H^i \leftarrow$ estimateLossRatio$(n_I, P, (\mathcal{P}_{H_j}^{i}{}')_{j=1}^{P})$)
4: **end for**
5:
6: **if** $median_{i\in\{1,2,\ldots,\Gamma\}}(\hat{p}_L^i) - median_{i\in\{1,2,\ldots,\Gamma\}}(\hat{p}_H^i) \geq \tau$ **then**
7: $\quad$ return **true**
8: **else**
9: $\quad$ return **false**
10: **end if**
11:

Algorithm 1 Detecting Network Discriminators

detectDiscriminator(τ, Γ, γ, P)

1: **for** $i = 1$ **to** Γ **do**
2: $\quad$ # Pre-Probing Phase
3: $\quad n_I \leftarrow$ estimateInitialPacketTrainLength()
4: $\quad (\mathcal{P}_{L_j})_{j=1}^{P} \leftarrow$ constructAProbeSequence(n_I, P)
5: $\quad (\mathcal{P}_{H_j})_{j=1}^{P} \leftarrow$ constructBProbeSequence(n_I, P)
6:
7: $\quad$ # Probing Phase
8: $\quad$ send$((\mathcal{P}_{L_j})_{j=1}^{P})$
9: $\quad$ pause for Λ
10: $\quad$ send$((\mathcal{P}_{H_j})_{j=1}^{P})$
11:
12: $\quad$ # Post-Probing Phase
13: $\quad ((\mathcal{P}_{L_j}^{i}{}')_{j=1}^{P}, (\mathcal{P}_{H_j}^{i}{}')_{j=1}^{P}) \leftarrow$ receivedPackets()
14:
15: $\quad$ pause for γ
16: **end for**
17:
18: # Detection Phase
19: return detectionEngine$(\tau, \Gamma, \gamma, P, (\mathcal{P}_{L_j})_{j=1}^{P}, (\mathcal{P}_{H_j})_{j=1}^{P}, ((\mathrm{P}_{L_j}^{i}{}')_{j=1}^{P}, (\mathcal{P}_{H_j}^{i}{}')_{j=1}^{P})_{i=1}^{\Gamma}$)

Detection Probe

Algorithm 3 Construct Low Priority Probe Sequence

constructLProbeSequnce(n_I, P)

1: $(\mathcal{P}_{L_j})_{j=1}^{P} \leftarrow \emptyset$
2: **for** $i = 1$ **to** n_I **do**
3: $\quad (\mathcal{P}_{L_j})_{j=1}^{P} \leftarrow (\mathcal{P}_{L_j})_{j=1}^{P} \,||\, (P_H)$
4: **end for**
5: **for** $i = 1$ **to** P **do**
6: $\quad$ **for** $k = 1$ **to** N' **do**
7: $\quad\quad (\mathcal{P}_{L_j})_{j=1}^{P} \leftarrow (\mathcal{P}_{L_j})_{j=1}^{P} || \, (P_H)$
8: $\quad$ **end for**
9: $\quad (\mathcal{P}_{L_j})_{j=1}^{P} \leftarrow (\mathcal{P}_{L_j})_{j=1}^{P} \,||\, (P_{L_i})$
10: **end for**
11:
12: return $(\mathcal{P}_{L_j})_{j=1}^{P}$

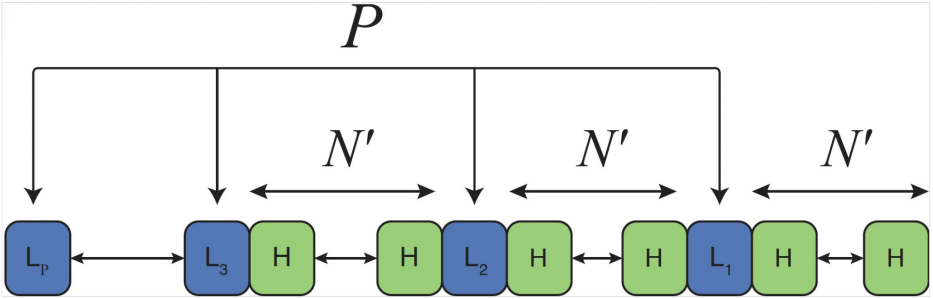
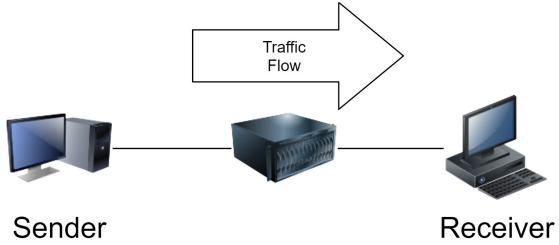


Illustration of detection probe (packet train) constructed by ConstructLProbe-Sequence function. *N'* is the number of packets in the *separation packet train*. *P* is the total number of *priority packets*. (Green) High priority packets. (Blue) Low priority packets. The Algorithm 3 demonstrates the creation of the LProbe specified in the above figure. HProbe is constructed in a similar fashion.

Parameters

- **P**: Number of priority packets. Depending on the phase this can be high priority packet or low priority packets. Only these packets are considered for detection.
- **N'**: Separation packet train length. Depending on the phase this can be high priority packet or low priority packets, but they have the opposite priority of the apt priority packets.
- **Γ, γ**: The number of times the measurement is run and the time between each measurement. The presence of cross traffic, on average, differs at certain time of the day and follows a particular pattern. By performing measurement throughout the day, we hope to capture the effects of time-dependant cross traffic variation. Hence, we choose (Γ, γ) to be (24, 1 hour).
- τ : Threshold, 20% . Research findings show that normal network packet loss rates are typically below 20%. We use this threshold as the minimum difference in loss rate between high priority and low priority packets needed to detect priority queueing.

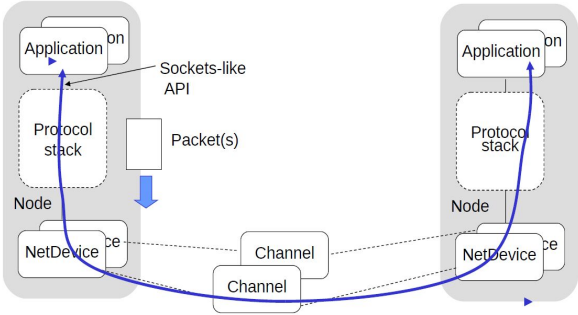
Validation Environment



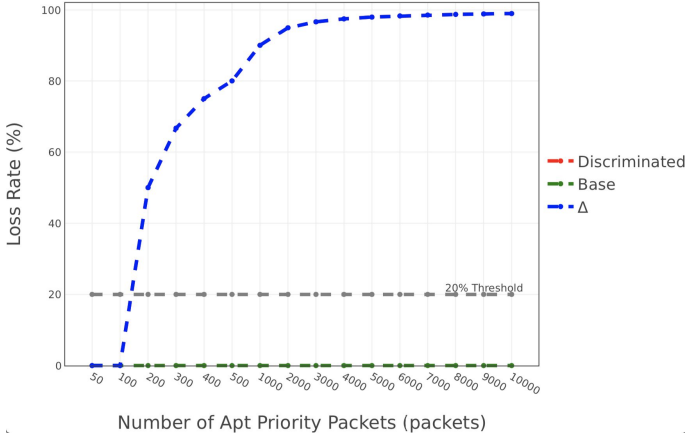
- The validation environment consists of sender, receiver and a router capable of performing strict priority queueing.
- We ran series of experiments based on selected scenarios with predictable outcomes for a given set of parameters in our simulated environment, specifically in ns-3.

Network Simulator 3 (ns-3)

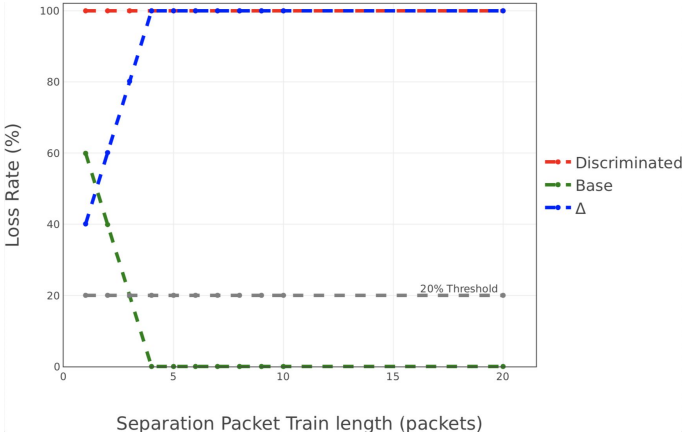
- A discrete-event open source simulator for network research and education.
- ns-3 simulates networking architecture in Linux.
- Complex simulation features include extensive parameterization system, configurable embedded tracing system , standard outputs to text logs or pcap (tcpdump/wireshark).



Validation Results



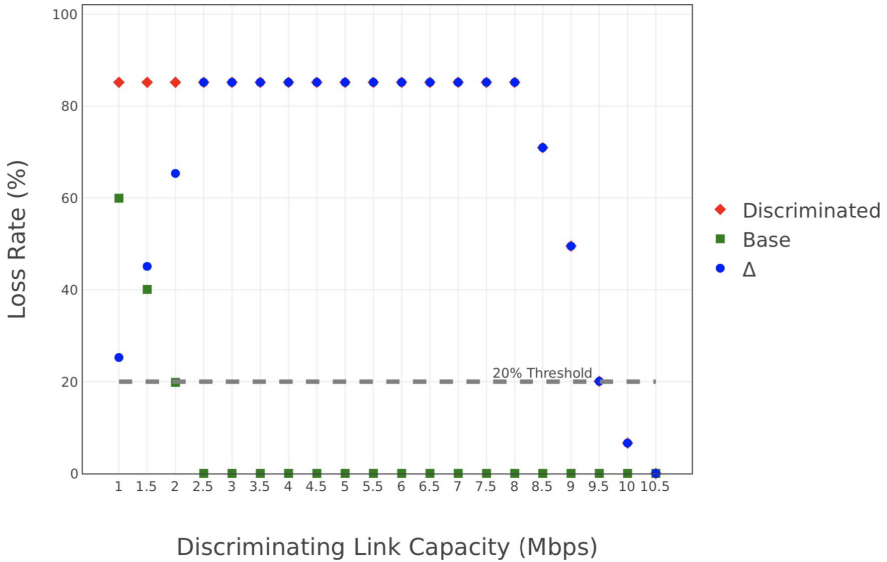
This plot demonstrates the effect of the number of priority packets (**P)** in detection accuracy, by comparing the loss rate of the high priority packets in **Base*** phase and low priority packets in **Discriminated*** phase**.** The plot shows that is no loss for **Base*** and **Discriminated*** until '**P**' is larger than 100. This is due to the fact the queue size are 100. At '**P**' = 200, there is a loss in Discriminated phase where the number of high priority packets increase thus dropping the low priority packets in the **Discriminated*** phase. There is no loss for the high priority packets in the **Base*** phase. The plot demonstrates that increasing the number of '**P**' packets for each of the phase does not affect the loss rate after a particular number. In this scenario, separation packet train, N`, is 3.



This plot shows the significance of the number of packets in separation packet train in **Base*** and **Discriminated***. In Discriminated/Low priority phase, the loss rate is high because the separation packet train contains the high priority packets and as the number of separation packet train (N`) increase the loss rate also increases. The plot suggests 4 is a suitable lower bound for N'.

Base* : High Priority Phase
Discriminated* : Low Priority Phase

Validation Results



This plot illustrates the changing loss rate of the high priority packets in Base phase and low priority packets in Discriminated phase with respect to the changes in the bottleneck capacity. As the bottleneck increases from 1Mbps to 10Mbps. During the **Base*** phase as the bottleneck increases, the loss rate of the high priority packets drops to zero. During the **Differentiated*** phase as the bottleneck capacity increases the loss rate of the low priority packet will decrease. When the sending and receiving data rate are the same, i.e., 10Mbps in this case, we notice the loss rate for both the phases is zero, since there is essentially no effective priority queueing takes place.

Base* : High Priority Phase
Discriminated* : Low Priority Phase

Conclusions

- Priority queueing is available in many routers and often enabled by network operators.
- We presented TarProbe that detects priority queueing effectively.
- While simulation confirmed the accuracy of our proposed detection tool, we will further validate our proposed approach on the Internet through live experiments. A real commercial middlebox with priority queueing feature such as Cisco Catalyst 3750 switch will be used in our Internet experiments.
- Future work also includes detecting priority queueing in an adversarial environment, where the intermediary actively tries to evade detection to conceal their discriminatory behavior.