Bufferbloat mitigation in the Linux WiFi stack – status and ongoing work

Toke Høiland-Jørgensen

LLC 2019
Lund, Sweden, May 2019
Outline

- Background: Bufferbloat mitigation in Linux
- Fixing bloat in the WiFi stack
- Airtime fairness and policy
- Status of mainline drivers
- Future / ongoing work
Background: Bufferbloat mitigation in Linux

- What is bufferbloat?
- Effects of mitigation in the qdisc layer
- Pieces of CAKE
- Byte Queue Limits
What is bufferbloat?
Mitigation in the qdisc layer (wired links)

Mean induced latency (ms)

- pfifo_fast
- sfq
- fq_nocodel
- fq_codel
- pie
- codel
- ared

Mean TCP goodput (Mbit/s)
# Pieces of CAKE

The `sch_cake qdisc` adds:

- Traffic shaping
- Host-based fairness queueing
- DiffServ handling
- TCP ACK filtering

<table>
<thead>
<tr>
<th></th>
<th>cake</th>
<th>cake_dst</th>
<th>cake_src</th>
<th>cake_triple</th>
<th>fq_codel</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mbits/s</td>
<td>1.5</td>
<td>2.5</td>
<td>2.0</td>
<td>2.5</td>
<td>1.5</td>
</tr>
</tbody>
</table>

©️ .Immutable - Toke Høiland-Jørgensen <toke@redhat.com>
Important feature: Byte Queue Limits (BQL)

- Queueing needs to happen where it can be managed
- There’s another queue in the hardware
- For Ethernet, we have BQL
- Enabled in most drivers now
- Check your driver, it’s a pretty small patch
Fixing bloat in WiFi
The qdisc approach doesn’t work for WiFi
Fixing queueing in the WiFi stack

- Toko Høiland-Jørgensen <token@redhat.com>
Queueing latency impact

Cumulative probability vs. latency (ms)

- With our solution
- Without our solution
Throughput impact

![Throughput Impact Graph]

- **Station 1**
  - FIFO
  - FQ-CoDel
  - FQ-MAC
  - Airtime

- **Station 2**
  - FIFO
  - FQ-CoDel
  - FQ-MAC
  - Airtime

- **Station 3**
  - FIFO
  - FQ-CoDel
  - FQ-MAC
  - Airtime

- **Average**
  - FIFO
  - FQ-CoDel
  - FQ-MAC
  - Airtime

**Mbits/s**

- **Station 1**
  - Station 2
  - Station 3
  - Average

- **Throughput Impact**
  - Toke Høiland-Jørgensen <toke@redhat.com>
Airtime: fairness and policy

How can we control airtime usage of WiFi devices?
802.11 Performance Anomaly

Effective transmission time $T(i)$ and rate $R(i)$ (for station $i \in I$):

$$T(i) = \begin{cases} \frac{1}{|I|} & \text{with fairness} \\ \frac{T_{\text{data}}(i)}{\sum_{j \in I} T_{\text{data}}(j)} & \text{otherwise} \end{cases}$$

$$R(i) = T(i) R_0(i)$$

Where $R_0(i) = \frac{L_i}{T_{\text{data}}(i) + T_{oh}}$ is the effective rate of a station transmitting without collisions.
Airtime fairness scheduler

Idea: Schedule stations to equalise airtime usage

- We already have per-station queueing
- The driver schedules stations anyway
- So we can simply account airtime using DRR scheduling

First prototype: Change the ath9k driver
Airtime fairness - achieved fairness (ath9k)

<table>
<thead>
<tr>
<th>Airtime share</th>
<th>FIFO</th>
<th>FQ-CoDel</th>
<th>FQ-MAC</th>
<th>Airtime fair FQ</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fast 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fast 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Slow</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fast 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fast 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Slow</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fast 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fast 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Slow</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fast 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Fast 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Slow</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

0.0 0.2 0.4 0.6 0.8 1.0

Airtime fairness - achieved fairness (ath9k)
Airtime fairness - throughput benefit (ath9k)
Enable airtime policy enforcement, supporting:

- **Prioritising single devices**
  - Use case, e.g.: “My TV needs more airtime”

- **Balancing device groups**
  - Use case, e.g.: 5G network slicing

- **Limiting groups to a maximum capacity share**
  - Use case, e.g.: Guest network
Policy operating modes

Weights set by userspace daemon (hostapd), supporting three modes:

- **Static** mode: Specify MAC priority in config
- **Dynamic** mode: Specify weight per group (BSS)
- **Limit** mode: Like dynamic, but only limit some groups
Policy enforcement effect

![Graph showing airtime share over time](image)
Status in the mainline kernel

Everything presented here is upstream as of 5.1!
## Upstream feature support – kernel versions

<table>
<thead>
<tr>
<th>Feature</th>
<th>Version</th>
</tr>
</thead>
<tbody>
<tr>
<td>BQL</td>
<td>3.3</td>
</tr>
<tr>
<td>sch_fq_codel</td>
<td>3.5</td>
</tr>
<tr>
<td>sch_cake</td>
<td>4.19</td>
</tr>
<tr>
<td>WiFi queueing</td>
<td>4.10</td>
</tr>
<tr>
<td>Airtime fairness (ath9k)</td>
<td>4.11</td>
</tr>
<tr>
<td>Airtime fairness + policy</td>
<td>5.1</td>
</tr>
</tbody>
</table>
## Upstream feature support - WiFi drivers (5.1)

<table>
<thead>
<tr>
<th>Driver</th>
<th>WiFi-Q</th>
<th>Airtime</th>
</tr>
</thead>
<tbody>
<tr>
<td>ath9k</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>ath10k</td>
<td>✓</td>
<td>✓</td>
</tr>
<tr>
<td>mt76</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>iwlwifi</td>
<td>✓</td>
<td>✗</td>
</tr>
<tr>
<td>others</td>
<td>✗</td>
<td>✗</td>
</tr>
</tbody>
</table>

✓: Supported
✗: Not supported

(✓): See caveat in following slides

- Toke Høiland-Jørgensen <toke@redhat.com>
Future / ongoing work

- Airtime Queue Limits
- Driver support
- Virtual time-based scheduler
Airtime Queue Limits (AQL)

Just like with Ethernet drivers, there can be queueing in WiFi devices

- E.g., seconds of queueing in ath10k (hence the (✓)).
- For WiFi we know the packet duration from rate control
- AQL means limiting queueing based on airtime
  - Prototyped by Google for ath10k with great effect
Driver support

Driver support is partial. Efforts to fix this:

- Porting the **full mac80211 TX path** to TXQs
- A method to **estimate airtime** for devices w/o hw support
  - This may tie into AQL support

- Toke Høiland-Jørgensen <toke@redhat.com>
Virtual time-based scheduler

- Current airtime scheduler is DRR-based
  - Requires heavy throttling of stations to achieve fairness
- A virtual time-based scheduler doesn’t
  - And may also allow MU-MIMO
- Prototyped in RFC patch
Summary

Bufferbloat mitigation in Linux has come a long way!

- Very close to being “done” for WiFi
- Everything is upstream as of 5.1
- Missing: AQL and full driver support
References

Papers with more detail (from my PhD):

- Bufferbloat qdisc comparison: The Good, the Bad and the WiFi
- Description of CAKE: Piece of CAKE
- WiFi queueing and fairness: Ending the Anomaly
- Airtime policy: PoliFi