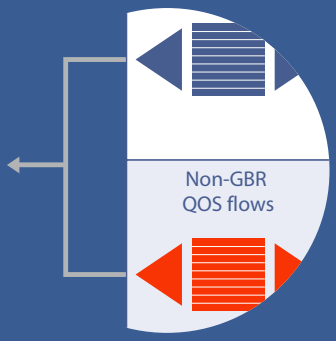
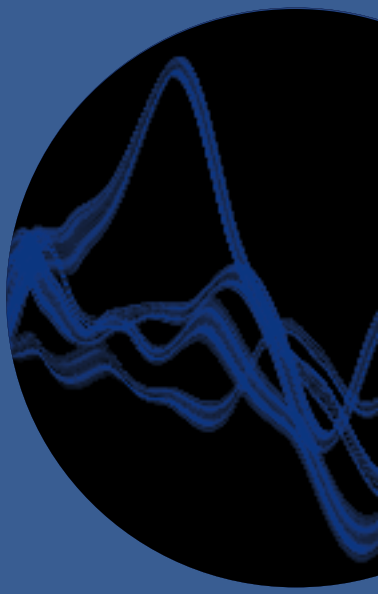
CHARTING THE FUTURE OF INNOVATION | #3·2018

ERICSSON TECHNOLOGY Review





VIRTUAL ACTIVE QUEUE MANAGEMENT

ERICSSON

LOW LATENCY, HIGH FLEXIBILITY

Virtual AQM

Virtual Active Queue Management is an innovative approach to buffer management that does not require deployment in the network nodes where the buffering takes place. Instead, it is deployed as a centralized network function, which allows for simpler configuration and increased flexibility compared to traditional Active Queue Management deployments.

MARCUS IHLAR,
ALA NAZARI,
ROBERT SKOG

To minimize the effects of bandwidth fluctuations in mobile networks, radio base stations are typically provisioned with large buffers. However, since excessive buffering of packets causes a large perceived delay for the application consuming the data, it is important for the network to detect and properly manage cases in which buffering becomes excessive.

Active Queue Management (AQM) is a well-established technique for managing the frequent and substantial fluctuations in bandwidth and delay that are common in networks. It has been applied to network buffers for many years with the purpose of detecting and informing data senders about incipient traffic congestion. Traditional AQM tends to work well in fixed networks, where bottleneck link bandwidth varies little over short time durations. However, the deployment of traditional AQM is not widespread in mobile networks.

The problem of bufferbloat

Bufferbloat is an undesirable phenomenon prevalent in packet networks in which excessive buffering results in unnecessarily large end-to-end latency, jitter and throughput degradation. It occurs due to large queues that absorb a huge amount of traffic in a congested path and usually causes a long queuing delay. To prevent packet losses, networks nodes rely on large buffers.

Bufferbloat occurs mostly at edge network nodes and defeats the built-in TCP congestion avoidance mechanism, which relies on dropped packets to

find the ideal send rate for a given end-to-end link. Mobile networks are provisioned with large buffers to handle bursty traffic, user fairness and radio channel variability. Even though these oversized buffers prevent packet loss, the overall performance degrades, as bufferbloat has major impacts on congestion control at endpoints. It misleads loss-based congestion control algorithms, resulting in packet overshooting on the sender side. Over-buffering also results in large jitter of end-to-end latency, which potentially brings in timeout events and eventually leads to throughput degradation.

Application impact

High latency and jitter cause significant problems for interactive applications such as VoIP, gaming and videoconferencing. Web browsing is also highly dependent on latency, and page load times grow linearly as round-trip time (RTT) increases.

Video streaming with adaptive bitrate through a continuous sequence of small HTTP downloads suffers when packets are lost. Under good conditions, video is transferred over the network at the same rate as it is played back. If the network can't keep up with video playback, the player usually switches to a lower video rate representation and reasonable video quality can be achieved with less bandwidth.

The main challenges in video streaming are packet loss and excessive buffering. When a packet is lost, the video player stops receiving data until the lost packet has been retransmitted, even though data packets continue to arrive. Video players normally buffer video segments, allowing them to continue playing from their buffer while downloading video segments that might involve retransmissions. Buffering is normally used for video on demand and timeshifted video but is limited for live video.

Terms and abbreviations

ACK – acknowledgement | **AMBR** – aggregate maximum bitrate | **AQM** – Active Queue Management | **CoDel** – Controlled Delay | **CPE** – Customer-premises equipment | **DN** – Data Network | **DNS** – Domain Name System | **DSCP** – Differentiated Services Code Point | **DRR** – Deficit Round Robin | **EPC** – Evolved Packet Core | **EPS** – Evolved Packet Switched System | **FQ** – Flow Queue | **FQ-CoDel** – Flow Queue Controlled Delay | **GBR** – guaranteed bitrate | **IETF** – Internet Engineering Task Force | **PDN GW**– public data network gateway | **PDU** – protocol data unit | **QFI** – QoS Flow Identifier | **QUIC** – Quick UDP Internet Connections | **RTT**– round-trip time | **SDF** – Service Data Flow | **SMF** – Session Management Function | **SYN** – synchronization | **TCP** – Transmission Control Protocol | **UE** – user equipment | **UPF** – user plane function | **vAQM** – virtual Active Queue Management

However, if multiple TCP packets are lost, TCP may slow down its sending rate to well below the data rate of the video, forcing the player into the rebuffering state.

In short, bufferbloat degrades application performance and negatively impacts the user experience. There is therefore a pressing need to control latency and jitter in order to provide desirable QoS to users.

Mitigation

Traditional AQM mitigates bufferbloat by controlling queue length through dropping or marking packets from the bottleneck buffer when it becomes full or when the queuing delay exceeds a threshold value. AQM reacts to congestion by dropping packets and thereby signaling to the sender that it should restrict the sending data rate due to the imminent overload. The short wait time also improves the response time for the transport protocol error handling.

> WITH FQ-CODEL, IT IS POSSIBLE TO REDUCE BOTTLENECK DELAYS BY SEVERAL ORDERS OF MAGNITUDE

Controlled Delay

Controlled Delay (CoDel) is an emerging AQM scheme designed by the IETF (Internet Engineering Task Force) [1]. Its goal is to contain queuing latency while maximizing the throughput. CoDel attempts to limit bufferbloat and minimize latency in saturated network links by distinguishing good queues (the ones that empty quickly) from bad queues that stay saturated and slow. CoDel features three major innovations that distinguish it from prior AQMs. First, it uses the local minimum queue as a measure of the standing/persistent queue. Second, it uses a single state-tracking variable of the minimum delay to see where it is relative to the standing queue delay. Third, instead of measuring queue size in bytes or packets, it measures the packet sojourn time in the queue and uses it as a metric to detect incipient congestion.

CoDel works as follows: upon arrival, a packet is enqueued if there is room in the queue. While enqueuing, a timestamp is added to the packet. When dequeuing, the packet sojourn time is calculated. CoDel is either in a dropping state or a non-dropping state. If the packet sojourn time remains above the target value (default 5ms) for a specified interval of time (default 100ms), CoDel enters the dropping state and starts dropping packets at the head of queue. While CoDel is in dropping state, if the packet sojourn time falls below the target or if the queue does not have sufficient packets to fill the outgoing link, CoDel leaves the dropping state.

The CoDel parameters target and interval are fixed parameters and their values have been chosen based on the observations from several experiments. Interval is used to ensure that the measured minimum delay does not become too stale. It should be set at about the worst-case RTT through the bottleneck to give endpoints sufficient time to react. The default value is 100ms. The target parameter is the acceptable queue delay for the packets, including capacity to cope with traffic spikes. A target of 5ms works well in most situations; lower values can reduce the throughput. If the target is more than 5ms, there is minor or no improvement in utilization.

Flow Queue Controlled Delay

Flow Queue Controlled Delay (FQ-CoDel) [2] is a standard AQM algorithm that ensures fairness in CoDel. It is useful because AQM algorithms working on single queues exacerbate the tendency of unfairness between the flows. FQ-CoDel is a hybrid scheme that classifies flows into one of multiple queues, applying CoDel on each queue and using modified Deficit Round Robin (DRR) scheduling to share link capacity between queues.

With FQ-CoDel, it is possible to reduce bottleneck delays by several orders of magnitude. It also provides accurate RTT estimates to elephant flows, while allowing priority access to shorter flow packets [3].

In combination with improvements in Linux network stacks, CoDel and FQ-CoDel can eliminate the problem of bufferbloat on Ethernet, cable, digital subscriber lines and fiber, resulting in network latency reductions of two to three orders of magnitude, as well as improvements in goodput [4].

In mobile networks, however, AQM needs to adjust to link bandwidth fluctuation that varies for different users. This is due to the fact that bottleneck link bandwidth can vary significantly in mobile networks even over short time durations, which causes queue build-up that can add significant delays when link capacity decreases.

Our innovative concept for virtual AQM (vAQM) is applied per user and per flow, adapting to link rate fluctuation and considering the current bottleneck link bandwidth to the user. Bottleneck link bandwidth is determined by correlating the number of bytes in flight with RTT measurements. vAQM is similar to FQ-CoDel but it uses the measured time-varying link bandwidth to the user as the dequeuing rate instead of non-varying deficit/quantum as in FQ-CoDel.

Virtual AQM

Traditional AQM is typically designed to operate on queues directly. However, due to deployment constraints and other factors, it is not always straightforward to deploy new AQMs at the network edges where congestion is likely to be experienced. One way of getting around this problem is to move the bottleneck to a more central point in the network and apply the AQM there. In a fixed network, it is easy to achieve this by shaping traffic to a rate slightly below the bottleneck peak capacity. Since mobile networks exhibit vastly different properties of fluctuating bandwidth, a more refined approach is required to apply centralized AQM.

> “OUR INNOVATIVE CONCEPT FOR VIRTUAL AQM IS APPLIED PER USER AND PER FLOW”

Virtual AQM consists of two separate but interrelated components: a bottleneck modeler and a queue manager. A separate vAQM instance is required for each bottleneck. In LTE, this corresponds roughly to a bearer (dedicated or default), while in 5G it corresponds to a Protocol Data Unit (PDU) session or a QoS flow. To effectively model the bottleneck, the vAQM measures the RTT of packets between itself and the user equipment (UE), as well as the amount of inflight data.

Virtual AQM maintains four variables: Path RTT, Average RTT, Target Queue Delay and Target RTT. Path RTT represents the propagation delay between the vAQM and the UE at the receiving end, without queuing delay. It is calculated by feeding RTT measurement samples to a min filter. Due to the varying nature of radio networks, the Path RTT must be occasionally revalidated.

Average RTT is an exponentially weighted moving average of the measured RTT, which represents the current observed delay. The use of a weighted moving average reduces the impact of inherent noise in the RTT signal, which typically arises from path layer characteristics such as Radio Link Control layer retransmissions.

A certain bottleneck queue delay must be tolerated to ensure smooth delivery of data over the radio. Target Queue Delay represents the delay that is tolerated on top of Path RTT before vAQM takes some sort of action. Target Queue Delay is defined as a fraction of Path RTT. The sum of the Path RTT and the Target Queue Delay is the Target RTT.

These four variables are used to determine whether queuing is taking place in network bottlenecks. The information can be used in two distinct ways: either as direct input to an AQM algorithm or as input to a bitrate shaper.

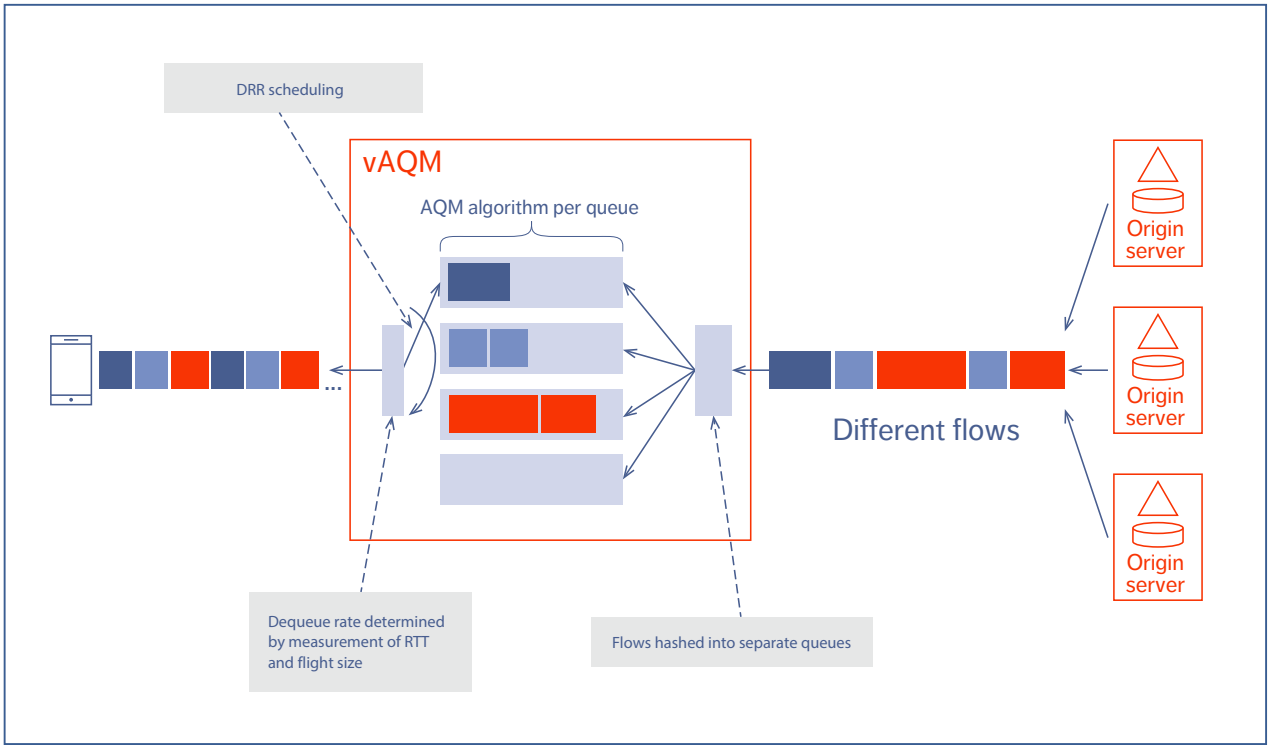


Figure 1: vAQM with dynamic bitrate adjustment and flow separation

vAQM with dynamic bitrate adjustment

In vAQM with dynamic bitrate adjustment, the traffic shaper dynamically adjusts its dequeuing rate based on the measured sending rate and the relation between Average RTT and Path RTT. The rate of incoming data is measured and applied to a max filter, while the average RTT is less than or equal to Target RTT. When Average RTT exceeds the Target RTT, the shaper will set the dequeue rate to the largest bitrate observed while Average RTT was below target. Furthermore, the difference between Average RTT and Target RTT is used as an additional bounding factor on the dequeue rate.

Internal queues will start to form if packets arrive at the vAQM at a higher rate than the dequeue rate. In such cases, a standard AQM algorithm such as FQ-CoDel can be applied to those queues. Path RTT can be used as input to the AQM algorithm to determine the CoDel interval and other variables.

Figure 1 demonstrates how vAQM assigns different flows to different queues managed by CoDel and dequeues with DRR using the current rate to the UE.

vAQM without bitrate adjustment

The vAQM without bitrate adjustment approach sacrifices the flow isolation that is achieved by FQ for reduced implementation and runtime complexity. This approach uses the RTT measurements as direct input to an AQM algorithm without applying the shaping step.

The AQM makes its drop/mark decisions based on the relation between Average RTT, Path RTT and the Target Queue Delay. If the Average RTT continuously exceeds the Target RTT for an interval amount of time, the AQM enters a dropping state, like that in CoDel. The interval is determined as a function of Path RTT. If multiple flows are traversing the same bottleneck, a drop will occur on the flow that is determined to be the most significant contributor of queue delay. This is effectively determined by comparing the flight sizes of the respective flows.

Handling of non-responsive flows

The basic principle of AQM is that it provides feedback to the congestion control mechanism at the sending endpoint. However, there is a possibility that senders will not react as expected to the signals that the AQM provides. The vAQM can detect such flows and apply flow-specific shaping to a level where the flow does not generate congestion in the bottleneck. This feature makes it safe to deploy and experiment with new congestion control algorithms, as it reduces the potential harm to the network and other flows sharing the same bottleneck.

Rate and RTT measurement

Virtual AQM relies on passive measurements of throughput and RTT to detect congestion. TCP is unencrypted at the transport level and can be measured trivially by storing sequence numbers and observing acknowledgements (ACKs) on the reverse path.

When the transport layer network protocol QUIC (Quick UDP Internet Connections) is used, a cryptographic envelope hides most transport mechanisms. Most notably, the ACK frame is completely hidden from on-path observers. A specific mechanism that allows passive on-path measurement of RTT is being proposed in QUIC standardization [5]. This mechanism makes use of a single bit in the QUIC short header, the value of the bit cycles with a frequency corresponding to the RTT between two communications endpoints. However, at the time of writing, neither the QUIC protocol nor the described mechanism is a finished standard.

The additional benefits of vAQM

The benefits of traditional AQM are well known, namely: fast delivery of short-lived flows and increased performance of latency-critical applications. There is an increased responsiveness to short-lived flows due to scheduling at flow level and minimal queue delay. For example, Domain Name System (DNS) responses and TCP SYN/ACK packets do not have to share a queue with a TCP download of bulk data.

In recent years, it has become increasingly common for different types of applications to share a single bottleneck buffer. Latency-critical applications that run in parallel with bulk download applications such as video streaming or file downloads typically experience severe performance degradation. By ensuring that the standing queues at the bottleneck are small or non-existent, large flows will not degrade latency.

In addition to the benefits of traditional AQM, vAQM is adaptive to link rate fluctuation and considers the current bottleneck link bandwidth to the user. It also provides the benefits of centralized AQM and modularity for increased flexibility. vAQM is able to simplify the deployment of AQM in an operator network because embedding it in a single (aggregation) node is sufficient to enable the management of the entire network downstream from the point of deployment. This means that legacy routers and radio schedulers do not need to be upgraded to include AQM. While vAQM currently uses a scheme resembling FQ-CoDel, it is built to support pluggable AQM modules, so that advances in the field of AQM and congestion control can have rapid impact in live networks.

vAQM testing

We have tested our vAQM concept on the Ericsson lab LTE network, embedding it in a multi-service proxy that inspects traffic to determine optimal bottleneck link capacity to UEs.

Figure 2: vAQM in full signal strength testing scenario (top) and in weak signal strength scenario (bottom)

We conducted the testing according to two multiuser cell scenarios: at full-signal strength and at weak-signal strength.

The test application consisted of loading a Google Drive page and downloading a 16MB image. We used three mobile devices to generate the background traffic for the multiuser cell, with each of them downloading 1GB files, so that we could test latency under load for multiple streams. The page was fetched using both TCP and QUIC, and the eNodeB was configured with a drop-tail queue. The server was configured to use the CUBIC congestion control algorithm for both TCP and QUIC.

We ran the tests for each scenario in three different ways:

- TCP with vAQM disabled
- QUIC with vAQM disabled
- TCP with vAQM enabled

The results, shown in *Figure 2*, demonstrate a reduction of the median RTT by roughly three times with vAQM enabled in the strong-signal scenario. In the weak-signal scenario, the median RTT with vAQM is not much different from the strong-signal scenario, whereas the traffic without AQM displays increased latency with a 95th percentile RTT of approximately one second for both QUIC and TCP traffic.

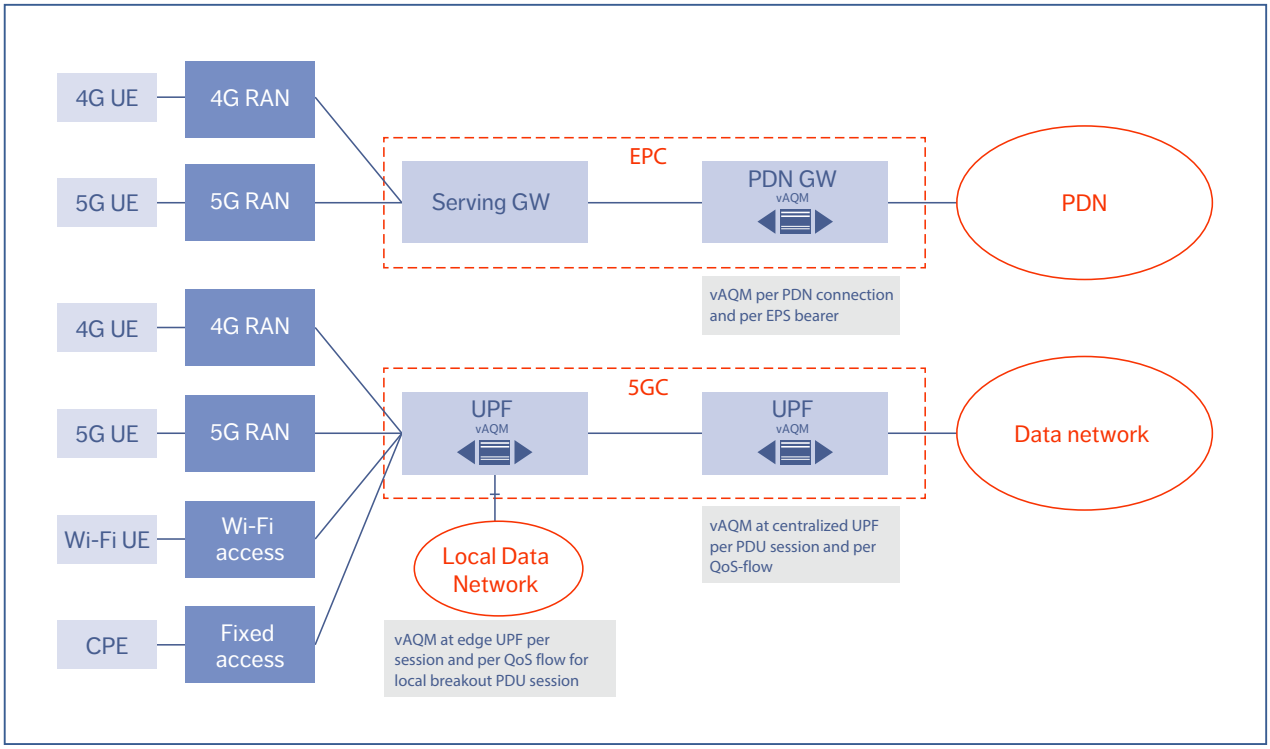


Figure 3: vAQM at PDN GW of EPC and at UPF of 5GC

vAQM in packet core

Figure 3 illustrates vAQM in two scenarios: Packet Data Network Gateway (PDN GW) of Evolved Packet Core (EPC) in 4G and in the User Plane Function (UPF) of 5G core.

In 5G, vAQM will be deployed as a network function by the UPF of the 5G core. It will be configured per PDU session, per SDF (Service Data Flow)/QoS flow – that is, the guaranteed bitrate (GBR) or non-GBR QoS flow. The non-GBR QoS flows of one PDU session will be aggregated and handled by one vAQM, whereas each GBR QoS flow will be allocated a vAQM with a dedicated virtual buffer.

The Session Management Function (SMF) will configure vAQM and its virtual buffers for the QoS flows during the PDU session establishment/modification phase. Configuration will occur when SMF provides the UPF with a QoS Enforcement Rule that contains information related to QoS enforcement of traffic including an SDF template (in other words, packet filter sets), the QoS-related information including GBR and maximum bitrate, and the corresponding packet marking information – in other words, the QoS Flow Identifier (QFI), the transport-level-packet-marking value. vAQM is self-learning, making it fully automated without any parametrization. When sending the packets on the PDU session to the UE, vAQM will read out the packets from the queues, recalculate the rate to the UE and adjust the bottleneck. In cases

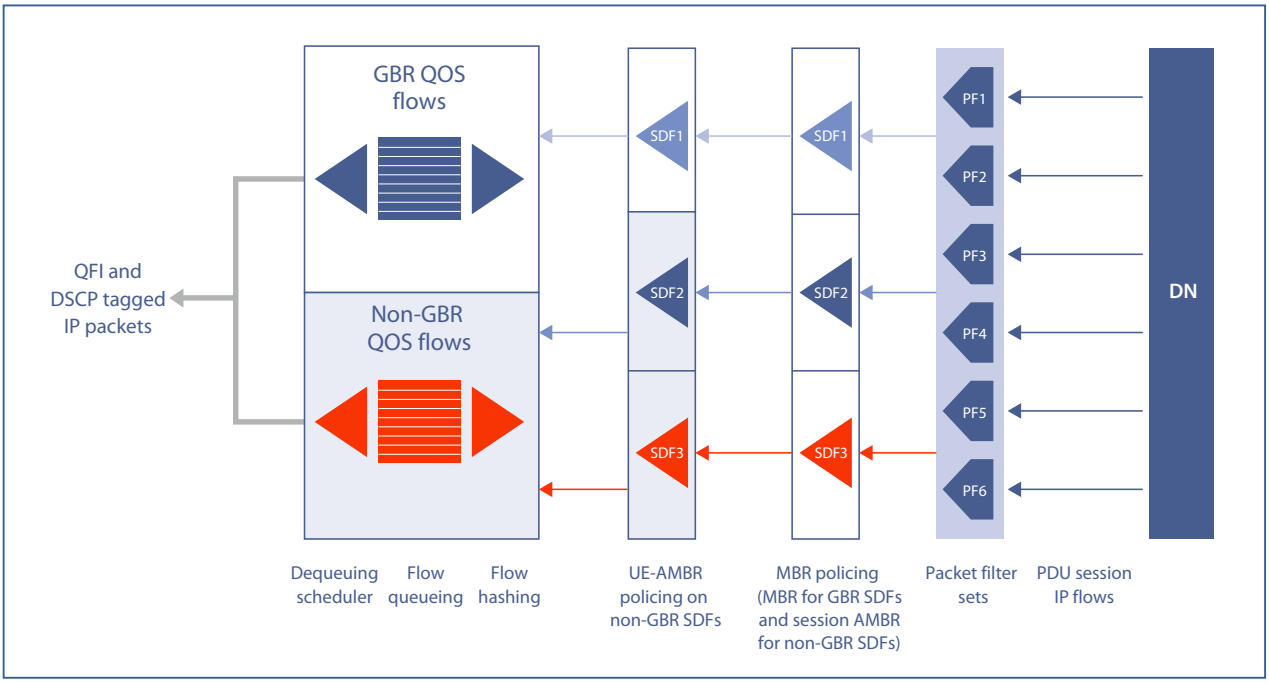


Figure 4: vAQM for the PDU session QoS flows at the UPF

where the PDU session has a local breakout, vAQM is configured at the edge UPF serving the local breakout; in other words, at the uplink classifier or branching point.

In downlink, as shown in *Figure 4*, the UPF binds the IP flows of the PDU session to SDFs/QoS flows based on the SDF templates using the IP packet filter set (for example, 5 tuples). The UPF then enforces the PDU session aggregate maximum bitrate (AMBR) across all non-GBR QoS flows of the PDU session. It also enforces the GBR for the GBR QoS flows. The UPF finally tags the packets with QoS Flow Identities (QFIs) and hands over the packets to vAQM, which is responsible for the QoS flow.

The different vAQM instances associated with QoS flows can contain implementations of completely different AQM algorithms or be configured with different default parameters.

Conclusion

The ongoing evolution of networks and applications is increasing the requirements on end-to-end performance. Specifically, it is important to be able to serve data at high rates without causing unnecessary delay due to bloated buffers. Our testing has shown it is possible to mitigate bufferbloat substantially through the use of a virtual form of AQM (vAQM) that is centralized upstream rather than being deployed in the bottleneck nodes, as it is in classic AQM. This approach greatly simplifies the deployment and configuration of AQM in mobile networks, and it ensures consistent behavior across bottleneck nodes, which in many cases are supplied by a multitude of vendors. In light of these benefits, vAQM will be an important user plane function in 5G core.

THE AUTHORS

Marcus Ihlar

◆ is a system developer in the field of traffic optimization and media delivery. He joined Ericsson in 2013 and initially did research on information-centric networking. Since 2014, he has been working on transport layer optimization and media delivery. Ihlar also participates actively in IETF standardization in the Transport Area Working Group. He holds a B.Sc. in computer science from Stockholm University in Sweden.

Ala Nazari

◆ is a media delivery architecture expert. He joined Ericsson in 1998 as a specialist in datacom, working with GPRS, 3G and IP transport. More recently, he has worked as a senior solution architect and engagement manager. Prior to joining Ericsson, Nazari spent several years at Televerket Radio working with mobile and fixed broadband and transport. He holds an M.Sc. in computer science from Uppsala University in Sweden

Robert Skog

◆ is a senior expert in the field of media delivery. After earning an M.Sc. in electrical engineering from KTH Royal Institute of Technology in Stockholm in 1989, he joined Ericsson's two-year trainee program for system engineers. Since then, he has mainly been working in

the service layer and media delivery areas, on everything from the first WAP solutions to today's advanced user plane solutions. In 2005, Skog won Ericsson's prestigious Inventor of the Year Award.

References

1. **Controlled Delay Active Queue Management, RFC8289, January 2018, available at:** *https://tools.ietf.org/html/rfc8289*
2. **The Flow Queue CoDel Packet Scheduler and Active Queue Management Algorithm, RFC8290, January 2018, available at:** *https://www.ietf.org/mail-archive/web/ietf-announce/current/msg17315.html*
3. **CoDel Overview, available at:** *https://www. bufferbloat.net/projects/codel/wiki*
4. **Making WiFi Fast (blog), Dave Täht, available at:** *http://blog.cerowrt.org/post/make-wifi-fast/*
5. **The Addition of a Spin Bit to the QUIC Transport Protocol, IETF Datatracker (2017), available at:** *https://datatracker.ietf.org/doc/draft-trammell-quic-spin*

Further reading

- **Ending the Anomaly: Achieving Low Latency and Airtime Fairness in WiFi (conference paper), Toke Høiland-Jørgensen et al., USENIX ATC (2017), available at:** *https://www.usenix.org/system/files/conference/atc17/atc17-hoiland-jorgensen.pdf*
- **Ending the Anomaly: Achieving Low Latency and Airtime Fairness in WiFi (slide set), Toke Høiland-Jørgensen et al., USENIX ATC (2017), available at:** *https://datatracker.ietf.org/meeting/99/materials/slides-99-iccrg-iccrg-presentation-1/*

ISSN 0014-0171
284 23-3313 | Uen

Ericsson
SE-164 83 Stockholm, Sweden
Phone: +46 10 719 0000