January 2020

IDENTIFYING APPLICATION AND TRAFFIC PATTERNS CONTRIBUTING TO MICROBURSTS AND CONGESTION IN DATA CENTER NETWORKS

Saravanan Sampathkumar

Ajay Modi

Sandeep Sreerangam

Umamaheswararao Karyampudi

Ashish Singh

IDENTIFYING APPLICATION AND TRAFFIC PATTERNS CONTRIBUTING TO MICROBURSTS AND CONGESTION IN DATA CENTER NETWORKS

AUTHORS:

Saravanan Sampathkumar
Ajay Modi
Sandeep Sreerangam
Umamaheswararao Karyampudi
Ashish Singh

ABSTRACT

Microbursts are traffic events that can cause severe performance degradation in a network. With the advent of modern big data applications, microburst events are not uncommon in a data center. Rather than attempting superficial ad-hoc solutions, such as providing large buffer switches/routers, under provisioning bandwidth, etc., this proposal provides a technique to identify an offending application causing a microburst based on queue-level thresholds. Once identified, appropriate remedial action(s) (e.g., Quality of Service (QoS) actions, security actions, etc.) can be performed by a network administrator.

DETAILED DESCRIPTION

Data center traffic is bursty by nature. Server and Storage interaction in a data center has a high potential to cause congestion, especially if multiple servers interact with multiple storage devices in a single transaction. Such a traffic pattern can be frequently seen in data center networks, in particular, data center networks in which big data applications are implemented. On a network switch, queueing typically occurs at an (egress port, traffic queue) level. In a data center, a large number of flows may map to the same (egress port, traffic queue) level on each switch.

Typically, bandwidth monitoring on a network switch occurs over large time intervals at a queue-level. It is often difficult to track flows contributing to buffer drops with limited information spread over a large time interval.

This proposal provides a technique to proactively identify application(s) experiencing microbursts that could potentially exhaust a corresponding output queue's buffer.

Microburst events are highly bursty traffic events occurring over an extremely short interval of time. In some instances, an Application-Specific Integrated Circuit (ASIC) of a platform (e.g., switch, router, etc.) may support microburst event information. For such a platform, the ASIC can detect a microburst when the buffer utilization in an egress traffic queue rises above a configured up-threshold (measured in bytes). The micro burst for an egress traffic queue ends when the queue buffer occupancy falls below a configured down-threshold. Figure 1, below, is a chart illustrating example details associated with the queue level of a microburst event in an ASIC.

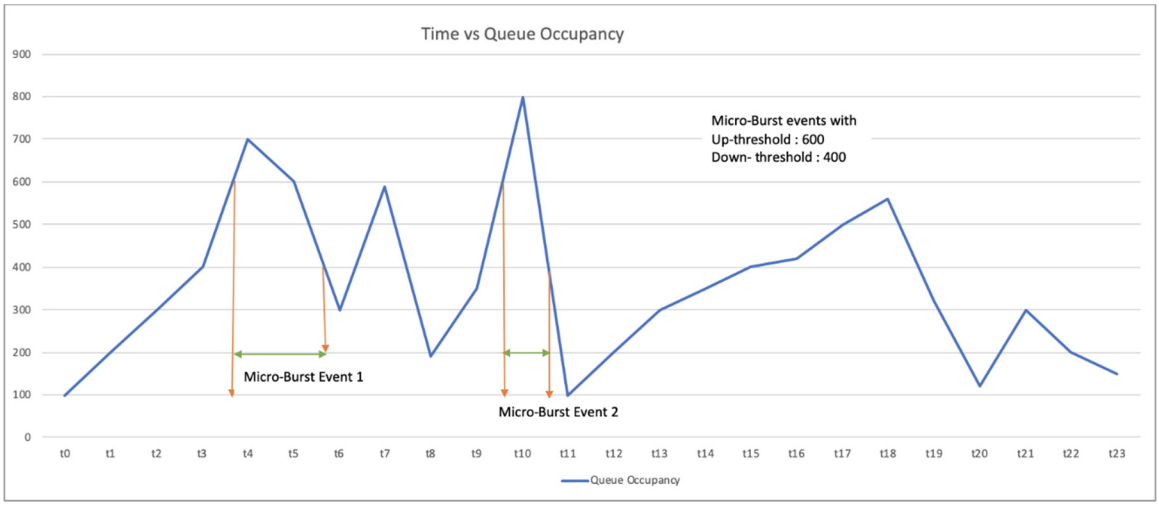

Figure 1: Queue level microburst event in an ASIC

Microburst events on all traffic queues of a switch can monitored and reported to a local Central Processing Unit (CPU) of the switch. A microburst event may include various parameters, such as:

1. Queue number (egress port, queue).
2. Time-stamp for up-threshold.
3. Time-stamp for down-threshold.
4. Peak occupancy (in bytes).
5. Time-stamp for peak occupancy.

Some platforms also support a flow table in which flow information (e.g., flow records) can be stored and exported to the local CPU. A flow record may include various information, such as:

1. Five Tuple (Source Internet Protocol (IP) address, Destination IP address, Source port, Destination port, and protocol).
2. Ingress Interface.
3. Encapsulation Virtual Local Area Network (VLAN).
4. Egress Interface.
5. Input Traffic queue.
6. Output Traffic queue.
7. Flow level micro burst.
8. Time stamp when the flow started/ended.
9. Time stamp when the microburst occurred.

As discussed in further detail below, this proposal provides a technique to determine one or more offending application(s) and generate a fault to alert a network administrator. The administrator can take corrective action, such as applying a policer, moving the application to a different QoS class, modifying a security policy, etc. The technique of this proposal is non-invasive and a user/administrator may take action at a later time with no need to debug the problem live.

Figure 2, below is a block diagram illustrating example details relating to feeding information from a switch ASIC, through the switch CPU, and into an analyzer.

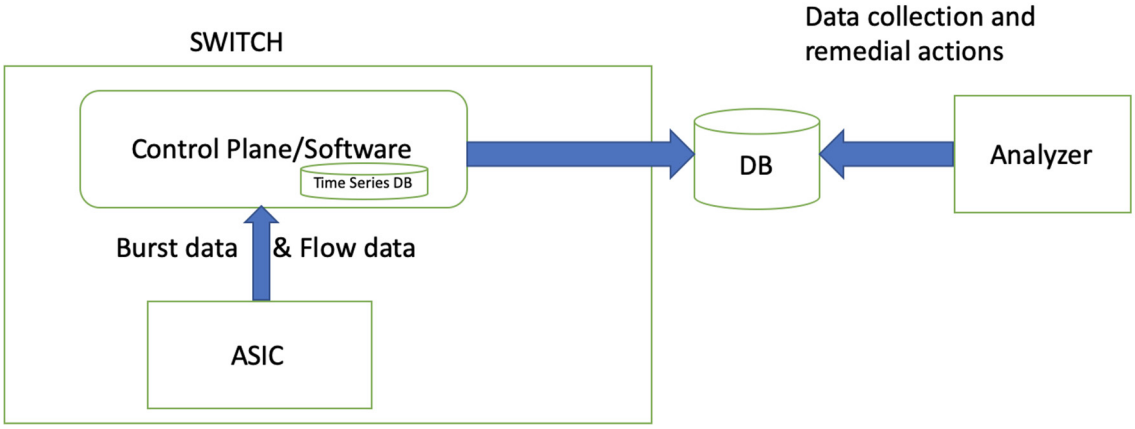

Figure 2: Switch level information path

Consider an operational example involving the switch as illustrated in Figure 2. During operation, the switch ASIC can generate an interrupt if one or more queues cross a configured microburst threshold. The local CPU handles the interrupt and stores the burst

and time stamp in a time series database. The ASIC may export all flow records to the local CPU at a regular time interval. The local CPU may also store the flow records in a time series database.

These two time series databases are exported to an analyzer (e.g., which may be running on a virtual machine (VM), may be reachable through fabric (either in-band or out-of-band), or the like). The analyzer consumes data from these two databases and identifies the offending flows and traffic pattern. An example traffic pattern is shown in Figure 3, below.

A network administrator can perform one or more action(s) based on the flow, such as applying a queue-level/port-level policer, moving the flow to a different queue, etc. In some implementations, all nodes in an Application Centric Infrastructure (ACI) fabric may feed data to one or more analyzer(s), which may analyze the information for every node.

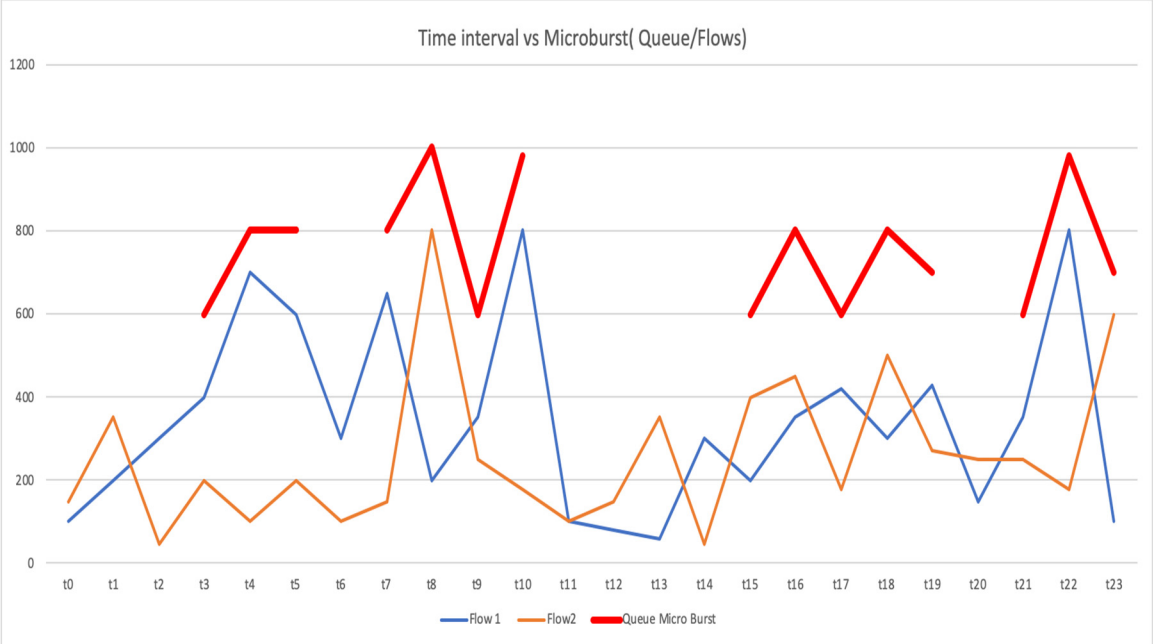

Figure 3: Analysis of offending flows by an analyzer

Figure 3, above, demonstrates example details that illustrate how an analyzer may utilize flow information to determine one or more offending flow(s). In the example shown in Figure 3, once there is queue-level microburst that crosses a given threshold of 600 bytes, the analyzer will report the offending flow for each time stamp. For example, at time

stamps t4, t5, t7, t10, and t22 as shown in Figure 3, flow1 is experiencing a burst. Further, at time stamp t9, flow2 is experiencing a burst. Based on this information, a network administrator can decide how to mitigate the impact of this bursty application.

As illustrated in Figure 3, an application can experience a burst at any time, which can increase the difficulty in monitoring application statistics. Typically, ASICs have a shared buffer architecture such that a frequently bursty application can severely impact other applications. Thus, it can be important to identify bursty applications in order to take proper action for such applications, as may be needed. However, given the difficulty that is typically involved in identifying bursty applications, it is often recommended to implement routers/switches with a large buffer capacity, which can increase cost, or over-provision bandwidth, which can lead to sacrifices in efficiency. The technique of this proposal solves the problem of identifying bursty applications without these two drawbacks.

Various other novel features may be provided by the technique of this proposal. For example, traffic may be monitored without user intervention. A network administrator may determine offending flows without any manual intervention. Further, storage and CPU processing may occur only with microburst events. Additionally, the solution provided by the technique of this proposal is event driven rather than poll-based. Still further, this technique may be useful in a data center fabric, which may include a large number of co-existing applications. Moreover, the microburst granularity of this technique is at a level of 100 microseconds (μs); thus, this technique does not suffer from the granularity issues often involved in monitoring application statistics that typically have a granularity in multiple seconds.

In summary, this proposal provides a technique to identify an offending application causing a microburst based on queue-level thresholds. Once identified, appropriate remedial action(s) can be performed by a network administrator.