cpacket NETWORKS

Application Note

Troubleshooting A Multi-Hop Service for User Experience Issues

How to Identify Chokepoints through Multi-Hop Analysis to Avoid Service Deterioration

www.cpacket.com

Highlights

- Learn how to design and build an always on network-centric observability architecture
- Learn how to troubleshoot a multi-hop service chain for rapid root cause analysis
- Learn about network-centric service level indicators (SLI) to observe for user experiences

Why It Matters?

This Application Note is focused on troubleshooting a service chain where the end-users are experiencing deterioration in a particular service. Still, it is harder to find out where precisely the issue may be. If the root-cause-analysis (RCA) is not done in a timely or systematic way, the result is a long mean-time-to-resolution (MTTR), which means the trouble ticket will stay open, and customers will keep calling, resulting in customer dissatisfaction and customer churn. If the service impacted is business-related, this will also result in revenue loss – which would at least mean losing your job.

The fact is that more and more services are becoming multi-hop and distributed across the data center and public cloud-hosted applications, or even multi-cloud. And more and more applications are using short-lived microservices. The challenge is getting complex due to distributed digital hybrid environments under different domains of control, short-lived microservices or containers, and a higher experience bar under competitive pressures. Examples include banking, financial services, insurance (BFSI) services, healthcare services, and many other services to which this application note, and the suggested methodology may apply.

A Network-Centric Approach to RCA

In a multi-hop path between a service and the users, it is primarily the network that gets blamed for a slowdown. As a Network Engineer or Network Operations person, you can wait and let the issue come down to you, starting with top-down troubleshooting, or you can take a proactive approach based on the nature of the issue and start bottom-up, which is the approach we are recommending in this application note.

These days the network is usually over-provisioned and reliable enough that you hardly get into packet loss issues due to congestion on the network. But you can still see the packet loss on a middlebox such as a firewall, load-balancer, WAN optimizer, proxy, etc. AWS direct connect is another example.

Out of the four “W” cPacket helps narrow down, the most useful in this case is *where* it happens or *what* entity or components in the service chain are causing the problem. Not only it helps with avoiding the existing service downtime, but also with service agility in terms of rolling out the new applications by the application team with buy-in from the network and security operations teams.

Commonly, the traffic flow between single or multiple users and an application or service usually goes

through multiple "hops" in the middle. For example, large banks use Virtual Desktop Infrastructure (VDI) platforms such as Citrix to provide secure access across hundreds or thousands of branch offices to conduct their daily business. Tellers from the branch offices log in at the start of their workday in order to process transactions such as check deposits and cash withdrawals for the clients. Yet, another example of such an application is booking airline tickets from a travel agency location or processing claims from an insurance company's field offices. These types of services typically span multiple hops or mid-points, such as a branch firewall, a colocation firewall, a proxy, a WAN augmentation system (WAAS), an authentication system, and then a server running the application such as Citrix. This is shown in diagram 1.

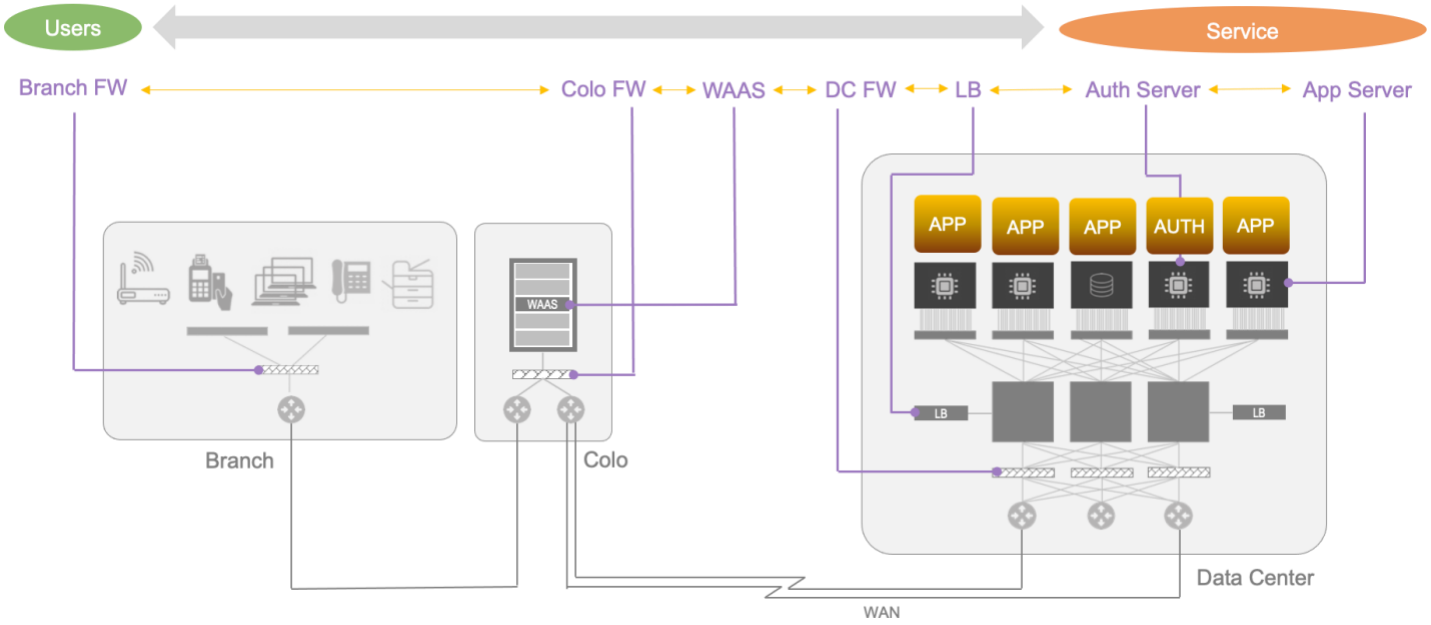

Diagram-1: A multi-hop service flow example

It is expected that the overall service gets impacted and user experience deteriorates due to an issue such as a slow-down at one or more of these mid-points, stalling the business and resulting in loss of revenue. The more locations impacted, the higher the loss. Therefore, narrowing down *what* the issue is and *where* it is precisely as fast as possible becomes essential. This requires a proactive observability practice and instrumentation set up ahead of time so that when things go south, and the phone starts ringing, you do not need the truck roll, and everything can be troubleshot remotely.

An Always-On Observability Design

Fortunately, cPacket has a tap-to-dashboard full-stack observability solution for troubleshooting root-cause-analysis in a multi-hop service chain. As shown in diagram 2, the design strategy starts with determining which strategic locations (branch offices, data centers, clouds) to set up for observability and which mid-points to monitor in the service chain. This would require deploying a balanced

combination of cPacket cTap® series passive TAPs (Test Access Points) and SPAN (Switch Port Analyzer) for the best economics. You may not need to monitor anything on the WAN or within the service provider network, but the LANs on either side are in your control. One of the starting points of observability thinking is to determine where you need to deploy cTap TAPs vs. using the SPAN feature on the switches.

The next you need is a cPacket cVu® NG series packet brokering and monitoring observability node to aggregate and broker those tap points. But the cVu node does a few additional important jobs. It:

- It helps dynamically select the monitoring point (port) for observability
- Helps dynamically turn the monitoring on/off or on-demand
- Tags those monitoring points as distinct spots for identification
- Provides clarity that the same packet is not viewed as a duplicate
- Provides microburst-level microscopic detail about the connected device's throughput

The correct monitoring point identification is a critical step. One way to do it is to use different VLANs or cPacket timestamping features on each side of the "mid-point" device under suspicion of causing a slowdown. Deduplication is turned off in this case on the cVu device. It could also be physically different locations with more than one cVu node deployed.

The next layer needed is the cPacket cStor® series packet capture and analysis observability node. Once the cVu node brokers the packet data tagged with the correct information to the cStor, it performs some critical value-adding roles:

- Saves the data permanently on the built-in storage in PCAP format
- Provide the analysis using cPacket cClear® series single-pane-of-glass extractions, view, and correlation with integrated Wireshark
- Perform TCP and Layer-4 analysis such as packet loss, connection issues, gap detection, round-trip time, one-way latency, and more.

The cStor node selected should be carefully considered based on the storage requirements if you need to capture the last few hours of data on-demand when there is an issue or capture and retain the data based on an all-time-on strategy. Also, based on the number of locations being aggregated.

Once the above observability nodes have been deployed, any-to-any connectivity could be achieved remotely through cVu, and analysis can be performed quickly using the cStor. A recommended strategy by cPacket is to have always-on observability to keep an eye on the services' health at the NOC (Network Operations Center) level before the users start complaining.

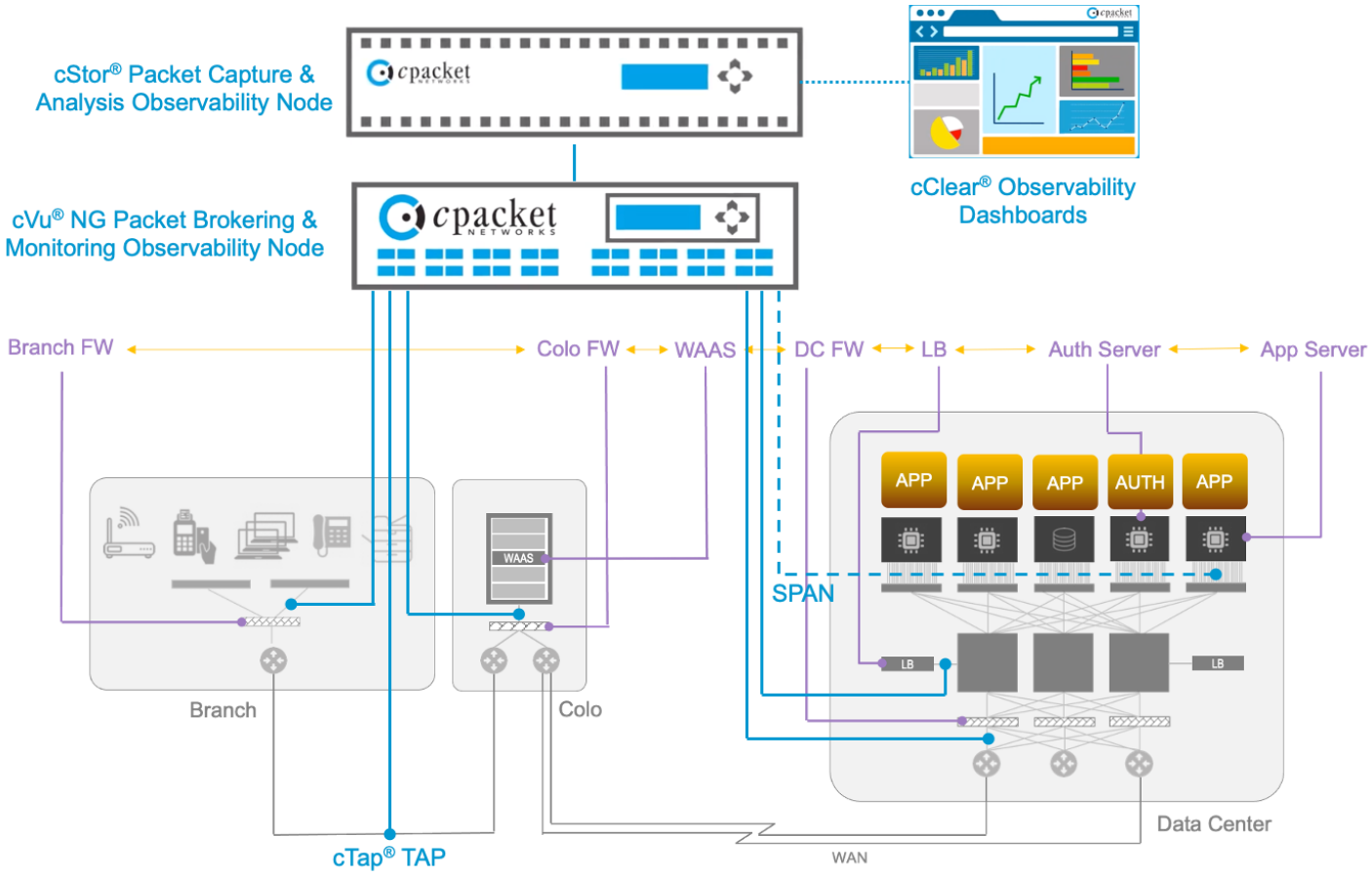

Diagram-2: Multi-hop service observability with cPacket

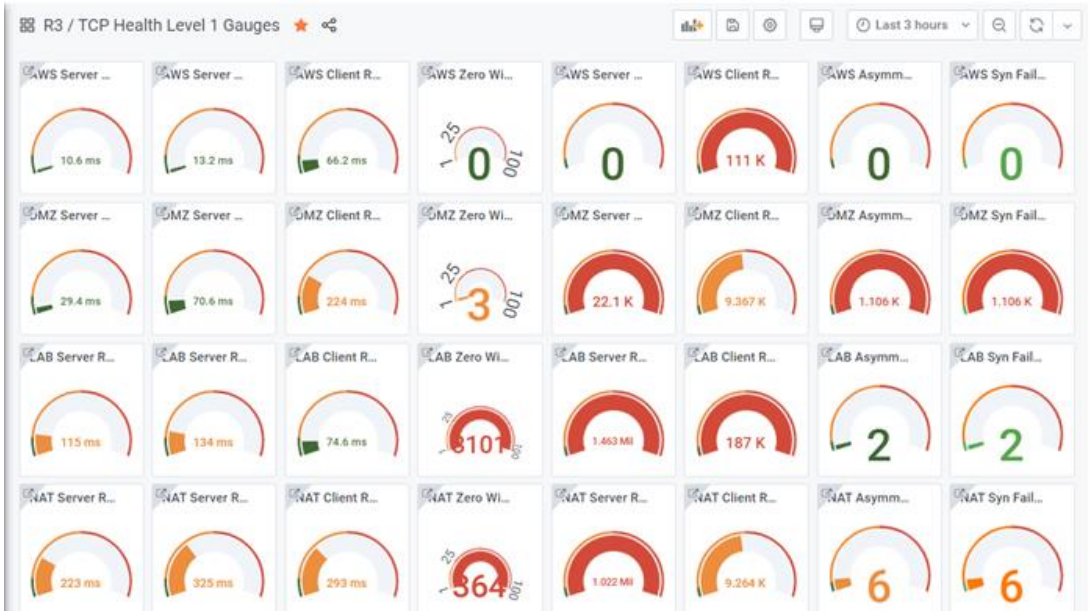

Diagram-3: Key service-level indicators observability using cClear

Deep-Dive Common Issues

In the above multi-hop service chain example, it is usually a single device in the middle that is usually causing the issue. The first step is isolating which one it is, or in order words, *where* the problem is. For example, it could commonly be a firewall or a load-balancer that is being saturated, hitting its peak performance, and dropping the packets. The issue is difficult to isolate if you use the standard GUI interface with those devices because they will report an "all good" picture. This is typically called the "watermelon dashboarding" problem, in which situation you see all good (green) pictures, but there is something wrong under the hood (red) that you cannot find out, at least not in time. You need the right tools to see the microscopic details with high-resolution data over a shorter interval of time.

For example, a branch or data center firewall with a peak load performance benchmark of 10Gbps will report that it is operating below 10Gbps over 1-minute intervals reporting. But if you use the cPacket cVu node to monitor the microbursts using its cBurst® feature, you may see that it is exceeding its throughput well above that over several millisecond intervals, hence dropping packets, as shown in Diagram-4. It does not matter if it is an active/active or active/passive design. The same may apply to a load balancer.

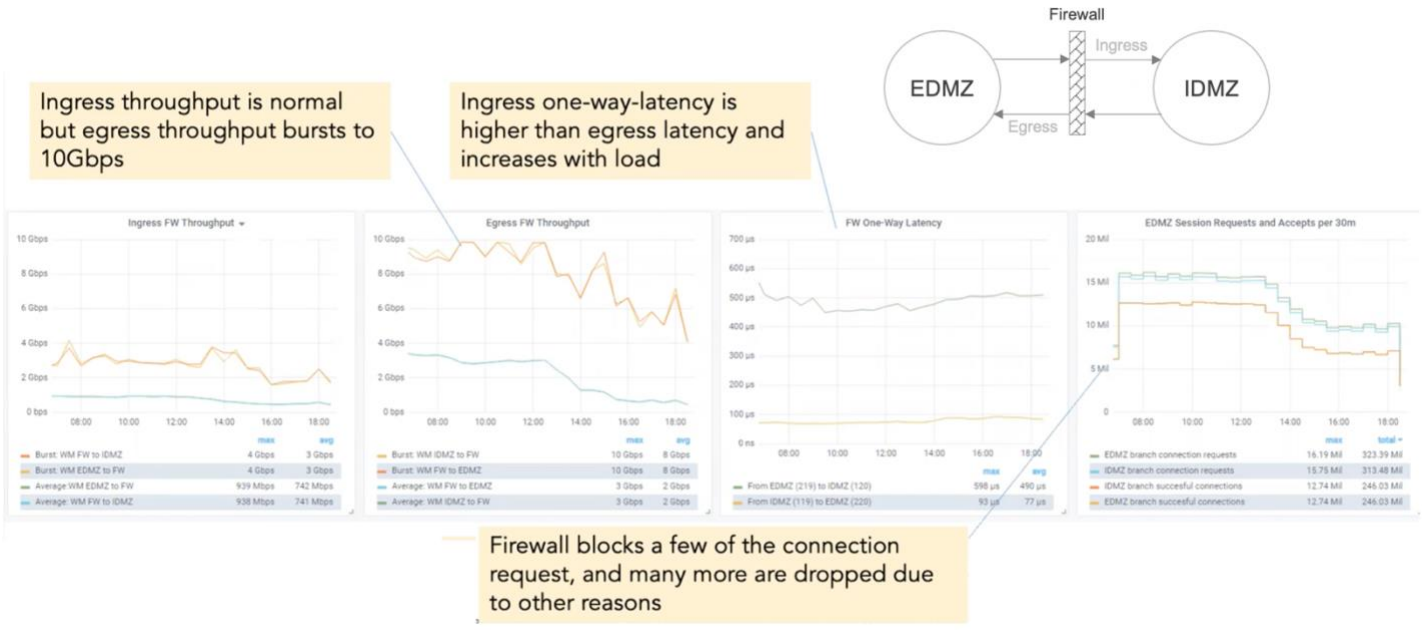

Diagram-4: Actual throughput and latency reporting using cPacket

Due to this overload problem, the device throughput will be impacted, and the result will be higher latency and response time, as reported by the cStor in diagram 4. Any type of mid-point device faces this capacity overload issue which results in adding latency. The above capabilities of the cPacket solution can help benchmark the network and security device's capacity ahead of time and plan upgrades in time, avoiding many headaches later.

Summary

Keeping your networks performing smoothly is essential to ensuring an optimal user experience and keeping revenue losses at bay. Unfortunately, identifying the source of network issues can be a stressful struggle that requires timely resolution. With cPacket's observability solution, IT teams are able to pinpoint problems quickly - whether on-premises, in the data center, or cloud - so downtime is minimized and services run without disruption.