CyPerf Test Drive

Testing That Replicates Your Network in Action

The focus of these labs is to provide the users with hands-on experience on how to use CyPerf for a few test scenarios. These scenarios assess the performance and security efficacy of a cloud-deployed security control (in this case, it is a Web Application Firewall—WAF).

KEYSIGHT

Table of Contents

Overview—The need for Hybrid, Distributed Network Testing

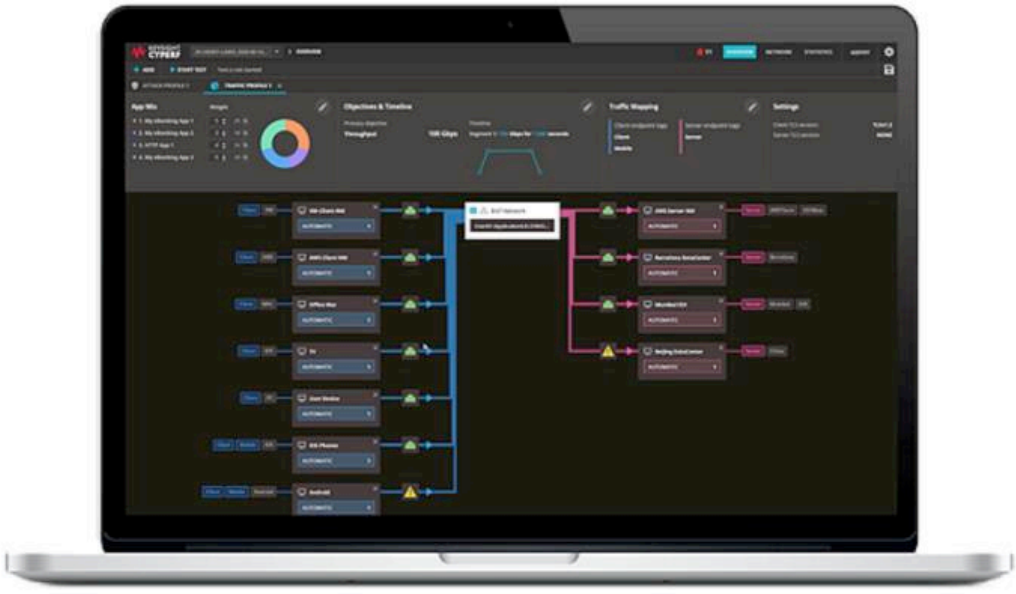
“If you can’t measure it, you can’t improve it.” This quote finds better applicability in today’s disaggregated hybrid networks, evolving applications, and service infrastructures. Digital transformation is the essence of these rapidly evolving ecosystems and key to competitiveness for most enterprises today. Success and agility in such environments depend on the ability to measure and analyze how distributed, disaggregated network infrastructures, applications, and services perform before going live, as well as during their production life cycle.

Digital business transformation and edge computing are bringing major unknowns to the performance, scalability, and threat protection of new, emerging network, and security architectures.

An enterprise faces the following challenges as it moves to more cost-effective and elastic off premises networking and storage:

- Delivering high-quality access to users and devices
- Delivering high-quality access to cloud services throughout the distributed, disaggregated networks
- Is the cybersecurity infrastructure enough to limit exposure across your on- and off-prem networking
- Are the security policies dynamically adjusting to your auto-scale events

Your perimeter-less, elastic, and dynamic network requires a new testing paradigm.



Keysight CyPerf is the industry’s first cloud-native software test solution that recreates every aspect of a realistic workload across a variety of physical and cloud environments to deliver unprecedented insights into end user experience, security posture, and performance bottlenecks of distributed, hybrid networks.

CyPerf delivers new heights in realism that comes from simultaneously generating both legitimate traffic mixes and malicious activities across a complex network of proxies, software-defined wide area

networking (SD-WAN), Secure Access Service Edge (SASE), VPN tunnels, Transport Layer Security (TLS) inspection, elastic load balancers, and web applications firewalls (WAF). Combined with the unique ability to interleave applications and attacks to model user behavior and security breaches, CyPerf enables a holistic approach in replicating distributed customer deployment environments faster and with more fidelity than other solutions.

Lab Introduction

Overview

A cloud-based setup with distributed, lightweight traffic agents that generate realistic application and malicious traffic to assess the performance and security efficacy of a cloud-deployed Web Application Firewall—WAF (that is, device under test or DUT) is used for the following labs.

A cloud deployed WAF for the device under test as a readily available security control is selected. However, following a similar approach, an extremely large set of network devices or entire infrastructure can be tested.

The main two components of the Lab environment are as follows:

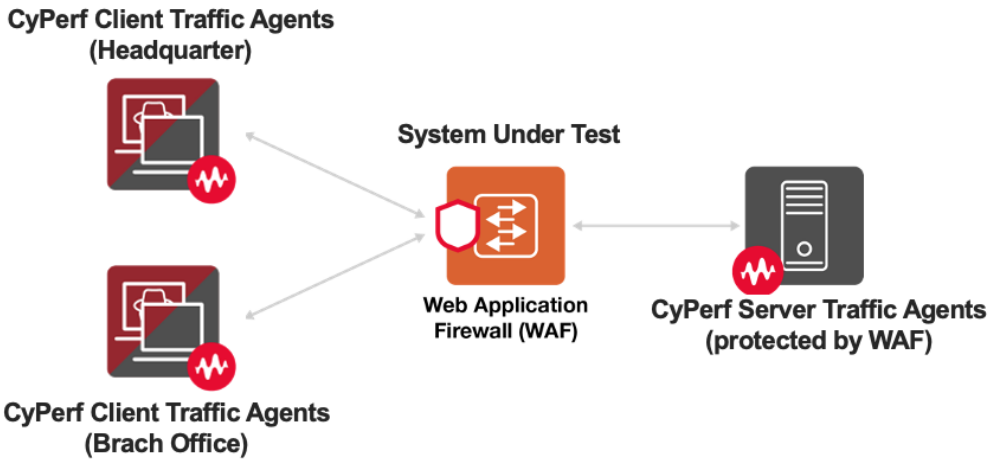
1. The **test tool**: Keysight's CyPerf emulating the malicious and legitimate traffic clients as well as traffic servers (all deployed as cloud instances)
2. The **device under test (DUT)**: cloud deployed WAF

The main components of Keysight's CyPerf (test tool) are as follows:

1. **Test Controller**: web-based UI for configuring and running tests, viewing real-time statistics, and reviewing results
    - The CyPerf Controller is deployed in the cloud and publicly available to users executing this lab as per <tba>
2. **Traffic Agents**: software agents generating test traffic
    - The CyPerf traffic agents (clients and servers) are deployed in the cloud to generate legitimate and malicious traffic going through the cloud deployed WAF (that is, DUT)

Setup

The following is the high-level diagram of the setup that is used in this lab:



The setup for all the labs consists of the following:

1. Traffic Agents:
    - **Clients**: Two CyPerf traffic agents acting as clients deployed in two different locations (in this setup, different availability zones) to model a distributed deployment. The two locations will be Headquarter and Branch Office.
    - **Servers**: One CyPerf traffic agent acting as server behind the cloud based WAF.

Important

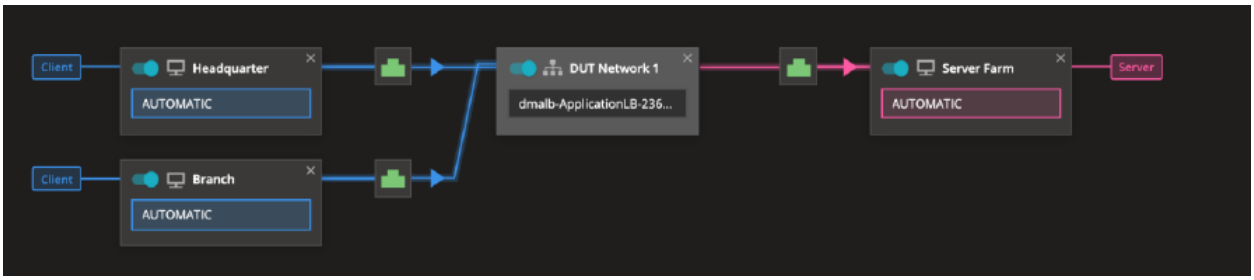
CyPerf traffic agents (clients and servers) can be virtually deployed in any Region/Zone, across a variety of public clouds (for example, Microsoft Azure, Amazon Web Services, Google Cloud Platform) as well as on on-prem machines to emulate a large-scale distributed network to test the performance and security efficacy of such infrastructures. For more details, see the product datasheet.

Caution

In this Lab, because of cost related considerations, we have forcefully kept one single server agent in the group. In addition, because of the same considerations, although the cloud instances type used for the CyPerf traffic can generate up to 10Gbps, we will limit the maximum throughput per test to 5Gbps.

CyPerf delivers elastically scaling traffic agents that can spawn and tear down dynamically during a test to validate auto-scale policies and enables customers to fine-tune the balance between user experience and security. For more examples and templates for deploying environments with multiple server agents in autoscaling groups, see the CyPerf's public GitHub repo at: https://github.com/Keysight/cyperf

2. Device Under Test (DUT):
    - Cloud-based WAF: The configuration and the DUT will mostly consist of default parameters with a default security rule to block SQLi attacks.



Resources and Prerequisites

To run this lab, users only need access to a common web browser. Everything will be run from a web interface.

All the resources for these labs can be found at the following location:
https://github.com/Keysight/cyperf/tree/main/CyPerfTestDrive

This includes the following:

- Configuration files: each lab will start from a configuration file that is available at the preceding location
- Step-by-step lab guiding document (this document)
- Intro video: a quick video that guides users on how to spin up and manage the test drive environment
- Cloud Formation templates

    - Using the Cloud Formation templates found at the preceding location, users can deploy a similar setup with the one from this lab in their own cloud account.
- Terraform script deploys the same environment as the preceding Cloud Formation templates, through a single, aggregated Terraform script

Looking for more resources? We offer a broad range of additional resources like deployment templates (for major public clouds), associated instructions and REST API wrappers at the following GitHub repository: https://github.com/Keysight/cyperf

Lab 1: Setup Baselining—Max HTTPS performance

Overview

As the very first step, the maximum performance of the System Under Test (SUT) by using HTTPS traffic is baselined.

For this scenario, the DUT configured with Load Balancing capabilities on the default WAF security rules to protect against SQLi attacks is used.

CyPerf is configured with the following HTTP profile:

- HTTP GET: to fetch a 1MB page
- HTTP POST: to send a 500KB response

Most of the traffic in today's networks is encrypted. Therefore, TLS traffic, which the Device Under Test decrypts, inspects, and re-encrypts (basically acting as Man-in-The-Middle and negotiating two different TLS handshakes on the client and server side for each client to server session) is used.

Objectives

CyPerf provides a very large set of Key Performance Indicators (KPIs), however, for this test, the following KPIs are considered:

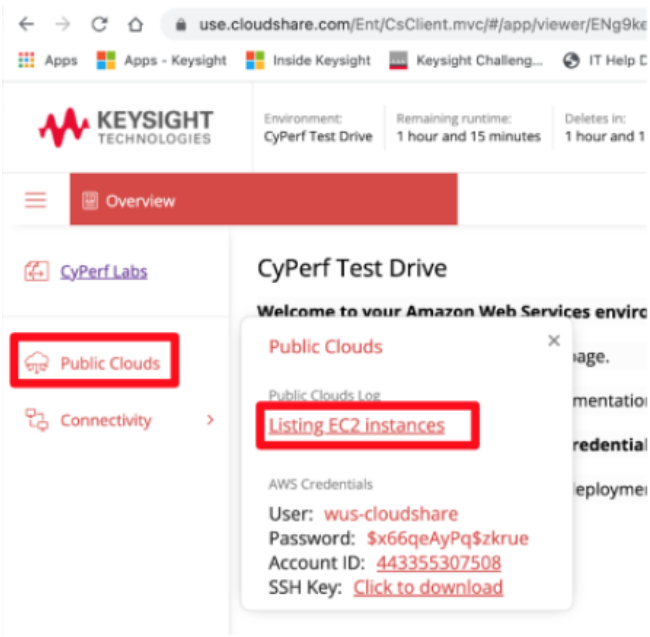
- Maximum throughput
- Latencies

As a time saving approach, import a configuration file that was created before. Nevertheless, the most important parameters are examined as per the below instructions.

Step-by-Step Instructions

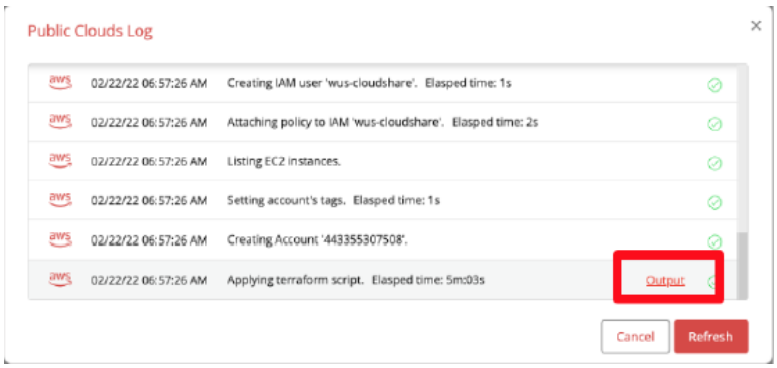
1. Connect to the CyPerf Controller by using the IP address.

   To perform the preceding step, in the CloudShare web page, select, **Public Clouds** > **Listing EC2 instance**.

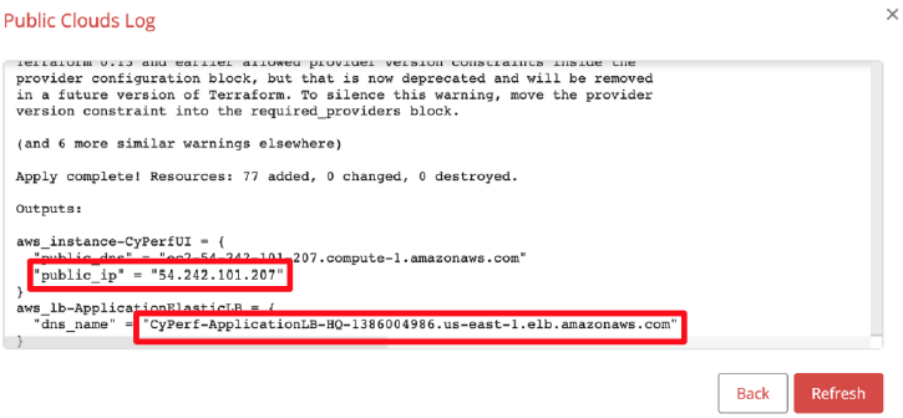


Note: If the **Listing EC2 instance** link is not available, ensure that you start the AWS deployment as per the introduction video (stream it from here).

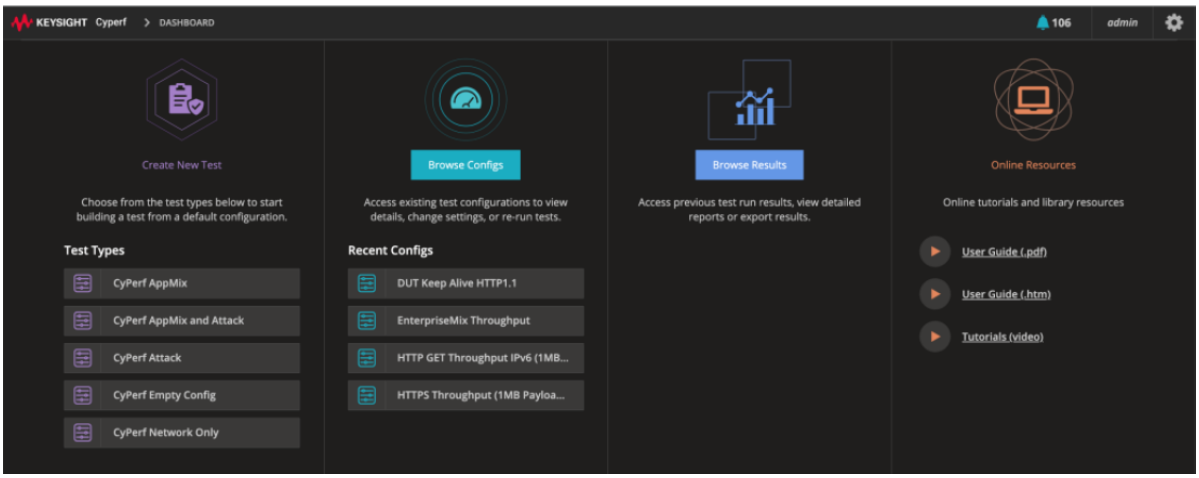
2. Select the **Output** link next to the *Applying terraform script* line as depicted in the following image:



3. Search the IP address of the CyPerf Controller (that is, aws_instance-*CyPerfUI*) as well as the DNS name of the DUT (that is, aws_lb-ApplicationElasticLB) in the Public Clouds Log as depicted in the following image. Copy both the addresses to a text file.



4. Sign in to the CyPerf Controller and start configuring the tests.
5. Connect to the CyPerf Controller: open a web browser and enter the IP address of the CyPerf Controller. Use the following credentials to sign in:
   - UserName: admin
   - Password: CyPerf&Keysight#1
6. Accept the TLS certificate of the Controller Web UI because this is a self-signed certificate.



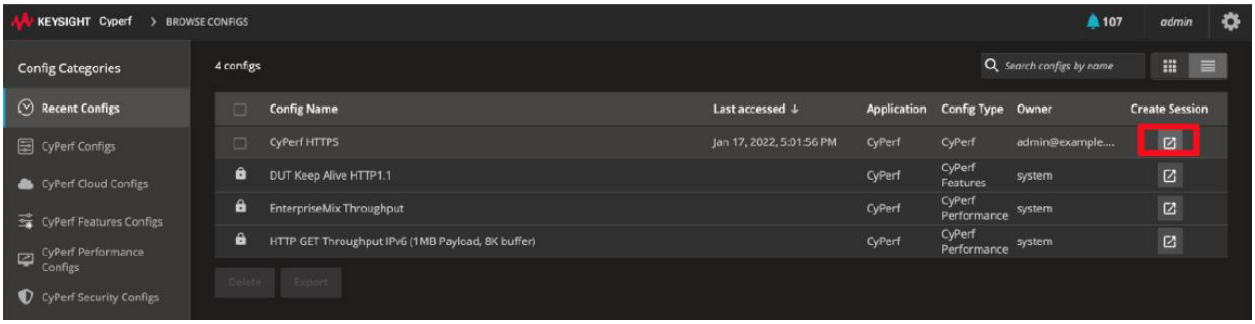
7. Import the first configuration file. Before importing the configuration file, ensure to download *CyPerf HTTPS.zip* file to your computer from the following GitHub repo:

   https://github.com/Keysight/cyperf/tree/main/CyPerfTestDrive/Configuration_Files

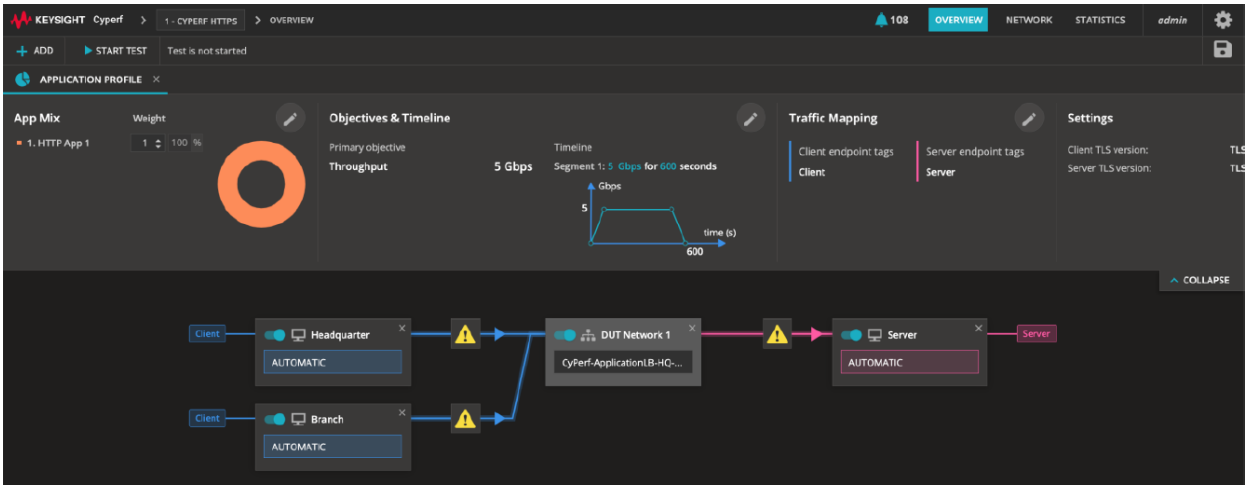
Important

CyPerf provides a list of canned configuration files optimized for various objectives and cloud environments, which are a great starting point for such scenarios. These configs are available at: Browse Configs > CyPerf Cloud Configs

8. In the CyPerf Controller's landing page, navigate to **Browse Configs**, and then select **Import** on the lower left corner.
9. Select the first configuration file (that is, *CyPerf HTTPS.zip*) from your computer and complete the import step.
10. After the new config file is imported, select the **Create Session** check box, as depicted in the following image:



The new config file loads into a new CyPerf Session.

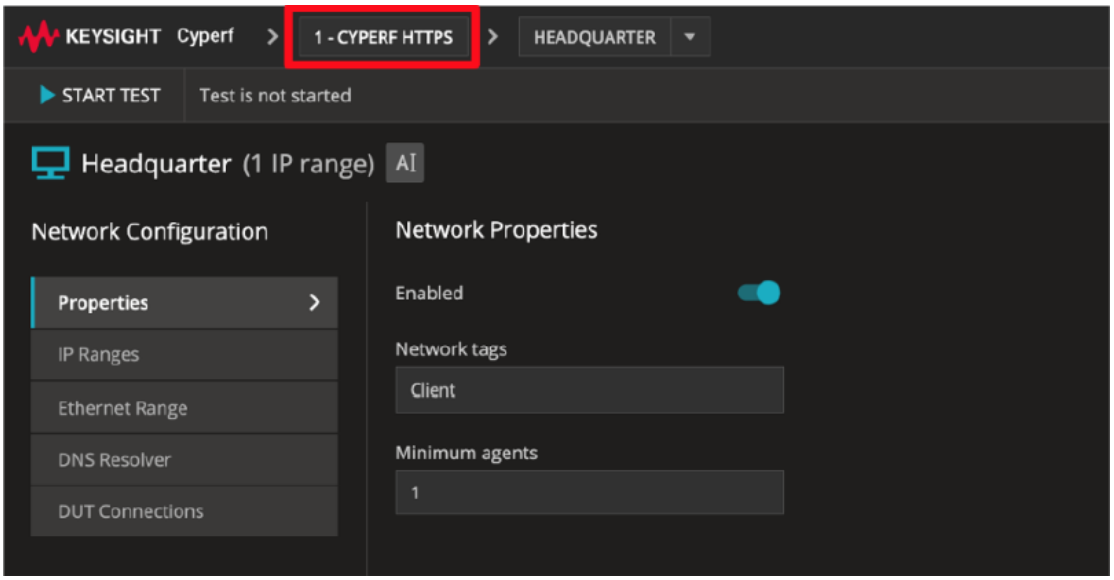


This is the CyPerf test overview page, where all the important test parameters are available in a single pane.

The lower pane is an interactive representation of the logical topology that is used in this test. This topology reflects the setup that is described in the Lab Introduction section. Select any element in the topology to view and configure the parameters.

For example, if you select the **Headquarter** network segment, all the network related parameters like configured IP addresses (in this case, it is set on Automatic, which means that the already assigned IP addresses are used), DNS Resolvers, and so on are visible.

You can select the back button in the browser or select **CyPerf HTTPS** in the breadcrumb to navigate back to the test overview page.

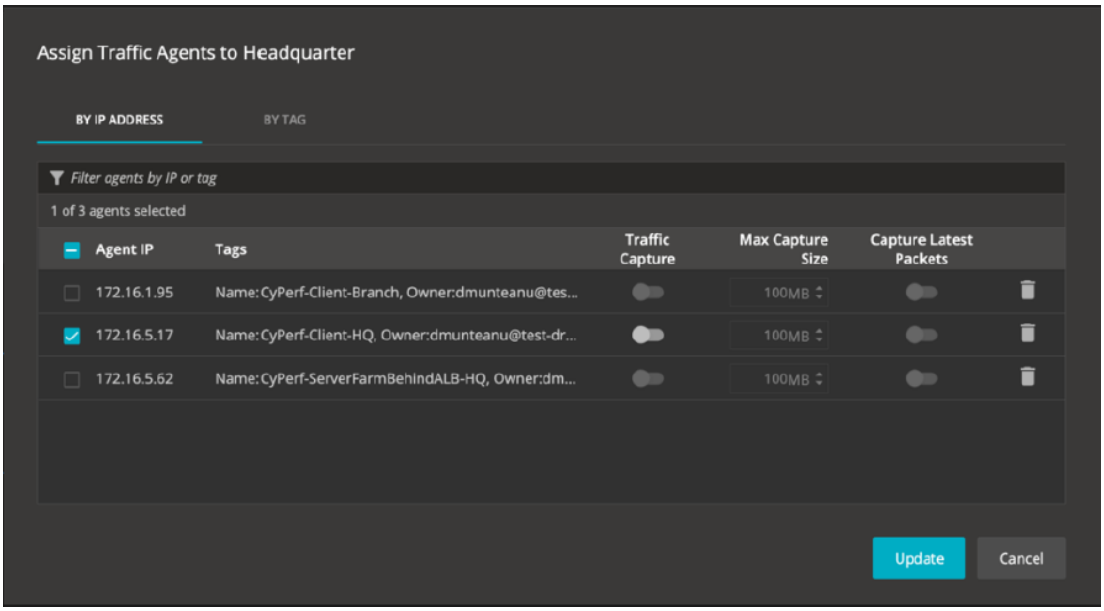


11. First, assign the already deployed CyPerf Agents (that are automatically deployed when applying the Terraform script in the CloudShare webpage) to each of the Network Segments in the test configuration:

a. Select the *yellow attention* icon ( ) corresponding to the **Headquarter** network segment.

   A new dialog box opens with all the CyPerf traffic agents connected to the CyPerf Controller.

b. For the Headquarter network segment, select the Agent that has the CyPerf-Client-HQ tag and select **Update**.

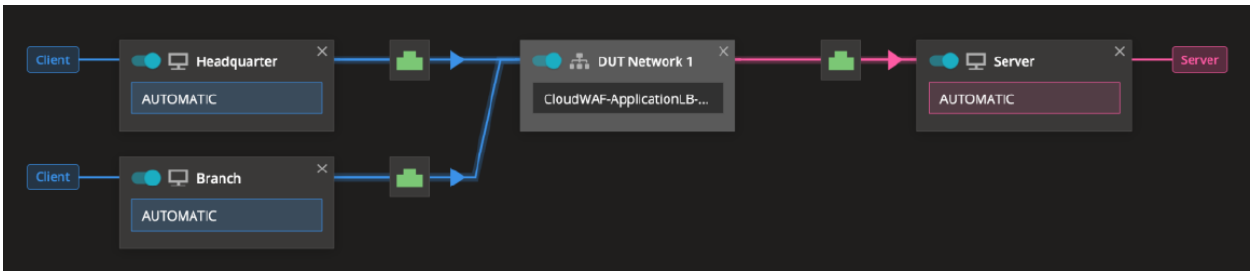


   After the new CyPerf traffic agent is assigned, the *yellow attention* icon ( ) turns into a green port ( ), which means that the **Headquarter** network segment has active CyPerf traffic agents assigned.

c. Repeat the preceding procedure and assign to the **Branch** network segment another CyPerf traffic agent with the tag *CyPerf-Client-Branch*.

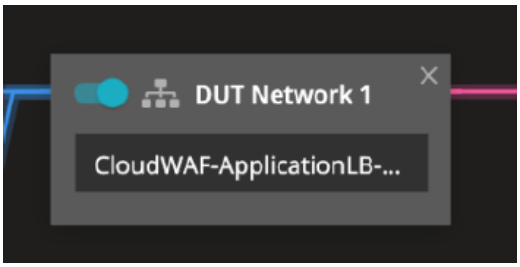
d. Lastly, for the **Server** network segment, assign the last CyPerf traffic agent with the tag *CyPerf-ServerFarmBehindALB*.

After the active CyPerf agents are assigned, ensure that all three network segments ports turn green ( ).



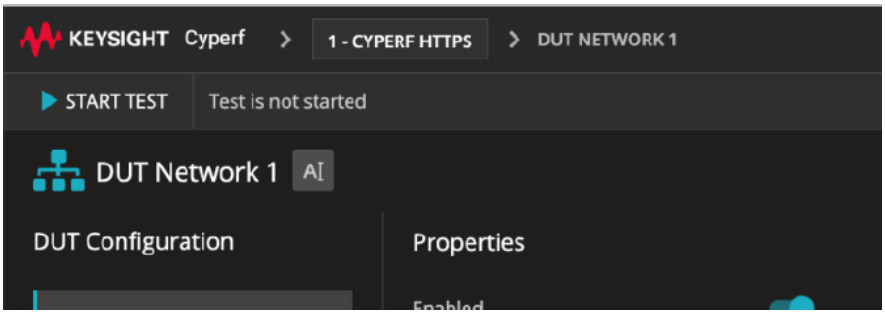
12. Next, configure the DUT address, which is basically the DNS hostname that the DUT is listening for incoming requests. The CyPerf client traffic agents use the DNS hostname to send traffic to:

    - Select the **DUT Network** object in the middle of the topology diagram:



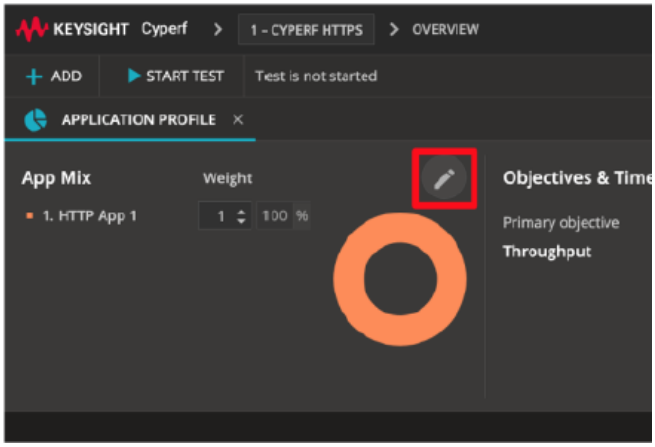
    - In the new menu, paste the DUT public DNS that was obtained from the Public Cloud Logs (at the beginning of this lab) in the **Host** field.

13. Select **CyPerf HTTPS** in the breadcrumb at the top of the page to return to the test overview page:



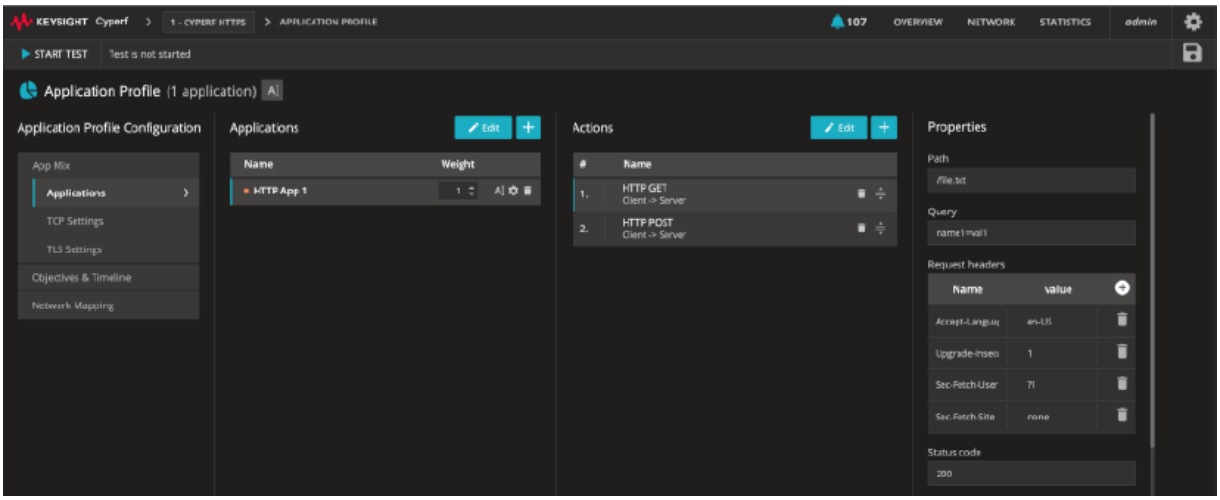
    Although the test is now ready to run, before starting it, we recommend you verify the most important test parameters.

14. **Application Profile**: Select the edit icon ( ) in the Application Profile section.



    The new menu provides access to all the L4-7 configurations options, like:

    - ***Type of traffic*** that is being sent from an L7 perspective:



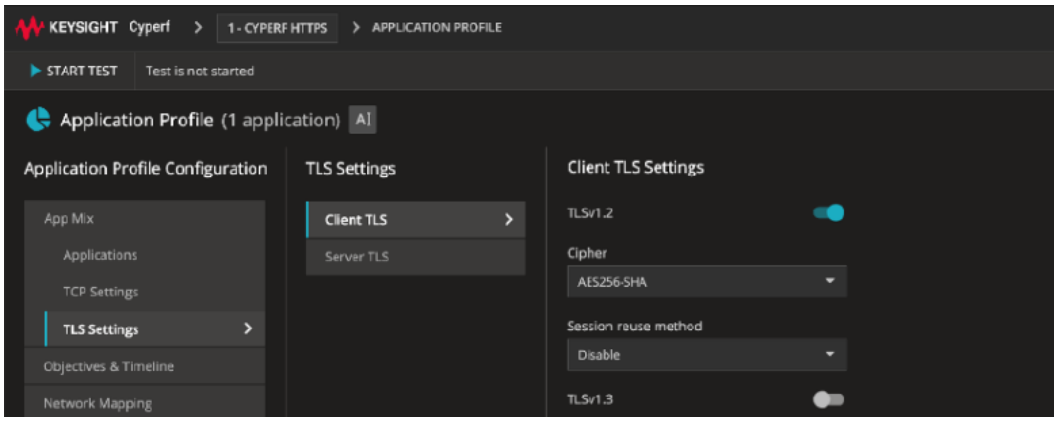
In this test, a basic HTTP profile for bi-directional traffic with the following commands is used:

- HTTP GET: to fetch a 1MB page
- HTTP POST: to send a 500KB response

Important

CyPerf offers an unmatched flexibility for configuring and parametrizing the applications and the actions, so that you can create your own unique profiles. The next lab provides more details about other apps available in the application library.

- ***TLS Settings***:



In this test, a basic yet common TLS profile with the following commands is used:

- TLS 1.2

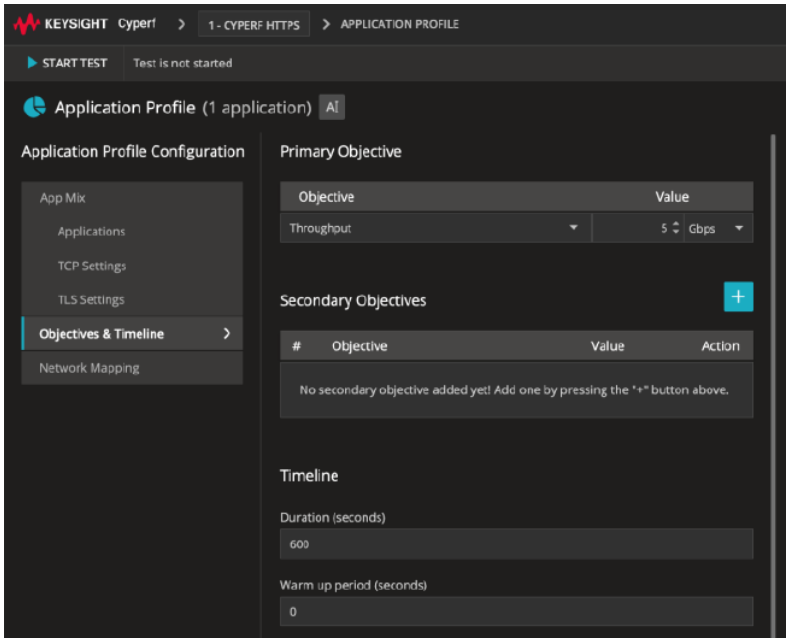
- RSA 2K key certificate
- RSA-AES256-SHA cipher

Important

Client and Server-side TLS parameters are configured independently, therefore, CyPerf can be used to test a wide range of topologies/deployments including, but not limited to:

- **TLS Offload**: to test the DUT's ability to offload TLS connections facing the served clients, while the Server side is unencrypted.
- **TLS Man-in-the-Middle** (MiTM): to test the DUT's ability to decrypt, inspect (for malicious content, filtering, and so on) and re-encrypt TLS connections. In this case, both client side and server side are TLS enabled.
- **TLS Pass-through**: in this scenario, the client side and server side are also TLS enabled, however, the DUT is not inspecting the encrypted content.

- ***Objectives and Timeline***: this is where the test load objective and duration are configured:



Important

The Objectives and Timeline menu also offers the ability to configure secondary objectives enabling use cases where for example, a certain throughput level must be achieved with a certain number of Simulated Users.

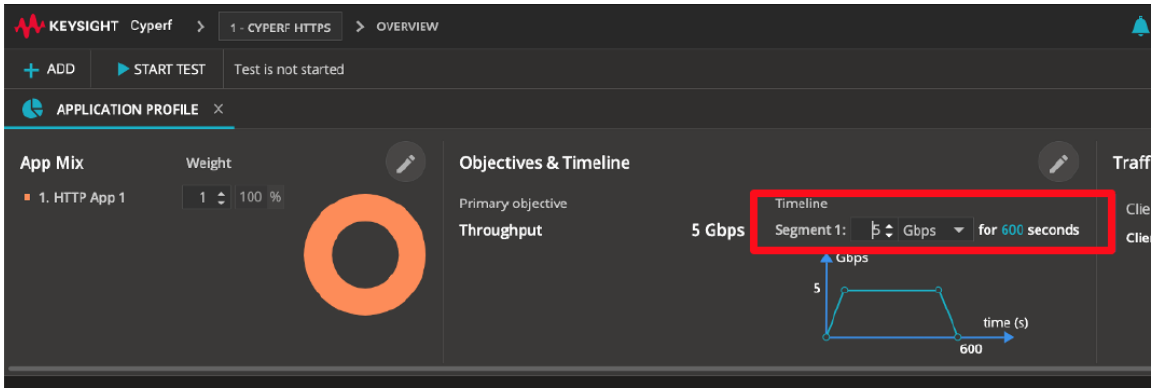
After the test is updated with the setup/agents' details assigned to the user, you can run the test and interpret the results.

Caution

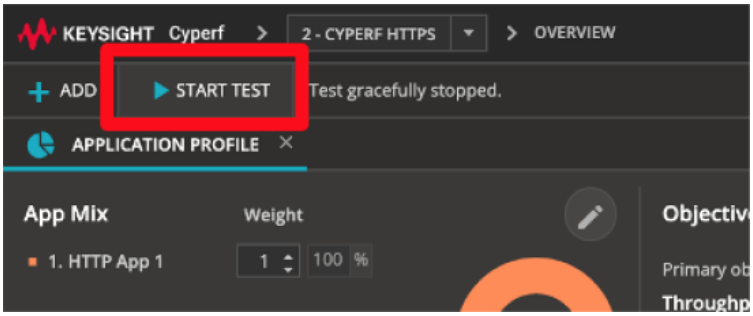
AWS Elastic Load Balancers (ELB) are able to address a broad set of use cases and traffic load profiles. However, when there is a sudden increase in traffic, we recommend that the ELBs would be pre-warmed. The general recommendation is that the traffic increases at no more than 50 percent over a five-minute interval.

For more information, see https://aws.amazon.com/articles/best-practices-in-evaluating-elastic-load-balancing/#pre-warming

Therefore, before running the first actual HTTPS performance baselining test, an initial test just to pre-warm the ELB (as our traffic load exceeds 50 percent over a five-minute interval) is run. In the test overview page, in the **Objectives and Timelines** section, ensure that the value is set to 5Gbps:

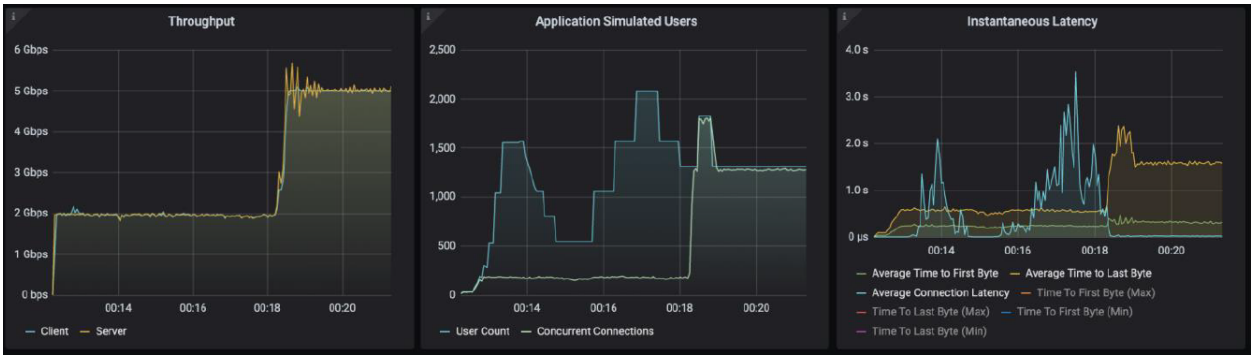


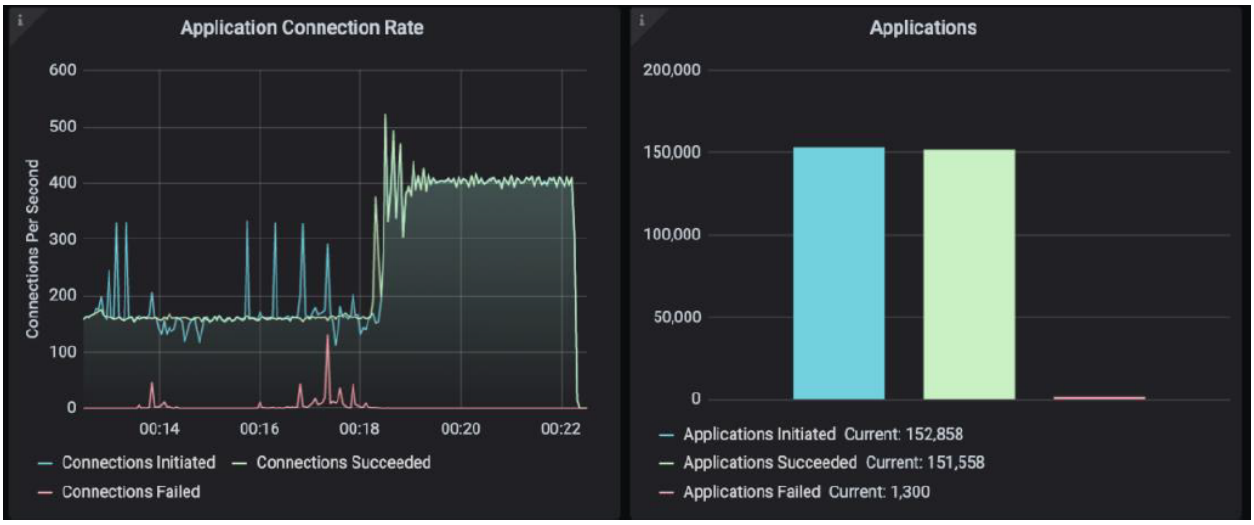
15. Select **Start Test**.



The test traffic agents are configured, and in a few moments, the traffic starts. The view automatically switches to the STATISTICS dashboard.

For this initial warm-up test, notice that the 5Gbps of throughput is achieved after a few minutes into the test (note that the exact time required to reach to 5Gbps varies). Also, depending on several factors, the test may report several Application Failures, Connection Failures, and high Latency stats as the ELB is warming up.





We will not analyze this initial warm-up test in detail as it is not the scope of this document. However, users can analyze different metrics to get an understanding of the infrastructure behavior when confronted with a spike in traffic, without a pre-warming up:

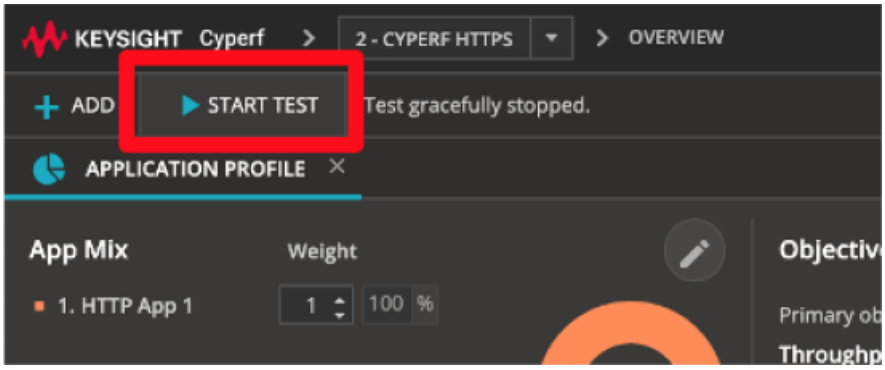
- Aside from the Application Failures, another obvious aspect is the huge spike in terms of the Connection Latency (the blue line of the Instantaneous Latency graph), especially in the first part of the test, when the configured objective of 5Gbps is not yet achieved.
- Another important metric is the Concurrent Connections levels, which is falling behind the Simulated Users, because the CyPerf traffic agents are bringing up multiple Simulated Users to reach the configured objective. However, only few of the attempted connections are being successfully established.

After the test finishes, run another test. This time the test you want to analyze in detail, keeping the same traffic load to 5Gbps.

Important

The reason why the same 5Gbps is kept as the traffic load is because of cost reasons (as the traffic is distributed and crossing through several network and security controls).

16. Select **Start Test**.



Again, the test traffic agents are configured, and, in a few moments, the traffic starts.

Result Analysis

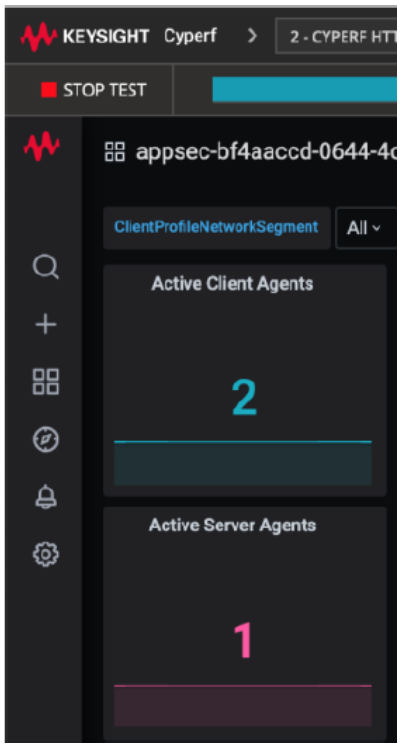
The maximum throughput performance test typically requires an iterative method in which the test is run multiple times, changing several test input parameters (which, because of time and cost constraints, is not fully covered in this lab). The most important recommendation is that the DUT must be configured as close as possible as in the production environment to obtain meaningful results, instead of datasheet-like figures, which are typically performed in an ideal scenario.

The importance of detailed and granular statistics is that it helps to identify if the device has reached its saturation point and pin-point issues. Also, interpreting the results in the correct manner ensures that transient network, device, or test tool behavior does not create a false negative condition, which is especially applicable for cloud environments.

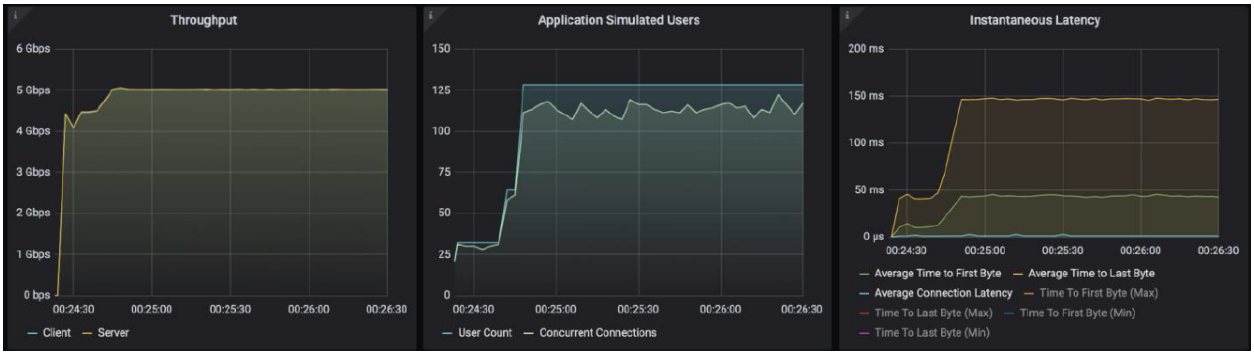
Real Time Statistics

The following graphs provide a view of the real-time statistics for the test. Real-time statistics provide instant access to key statistics that must be examined for failures at the TCP, TLS, and HTTP protocol level.

As a first thing, observe that there are two clients and one server that are active and participate in the test, which is as per the configurations and expectations.



Next, in terms of **Throughput,** the statistics indicate that the setup can easily reach the 5Gbps that is being configured:

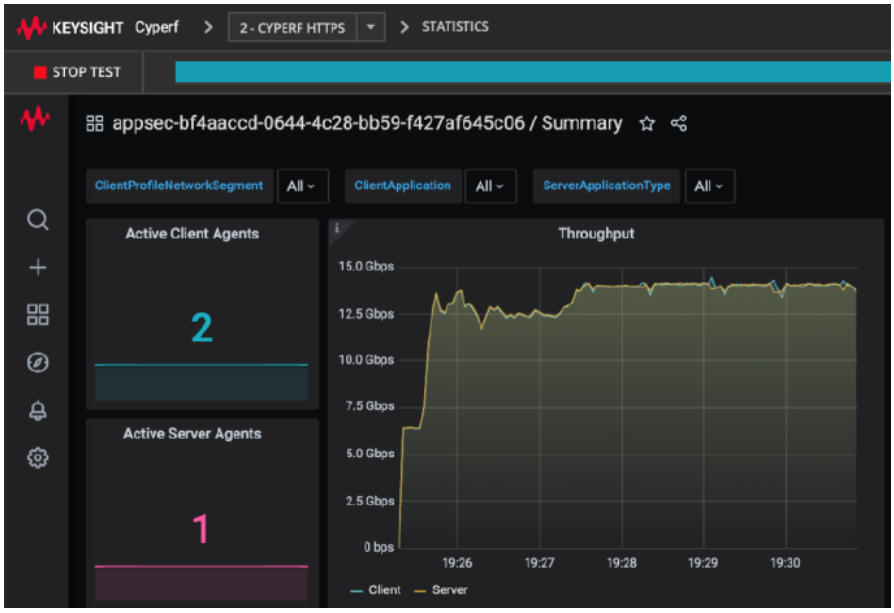


In the preceding graphs, you can observe that the number of Concurrent Connections is closely following the number of Simulated Users indicating that the infrastructure can easily cope with the generated traffic concurrency. Also, the Instantaneous Latency graphs are looking good with stable values.

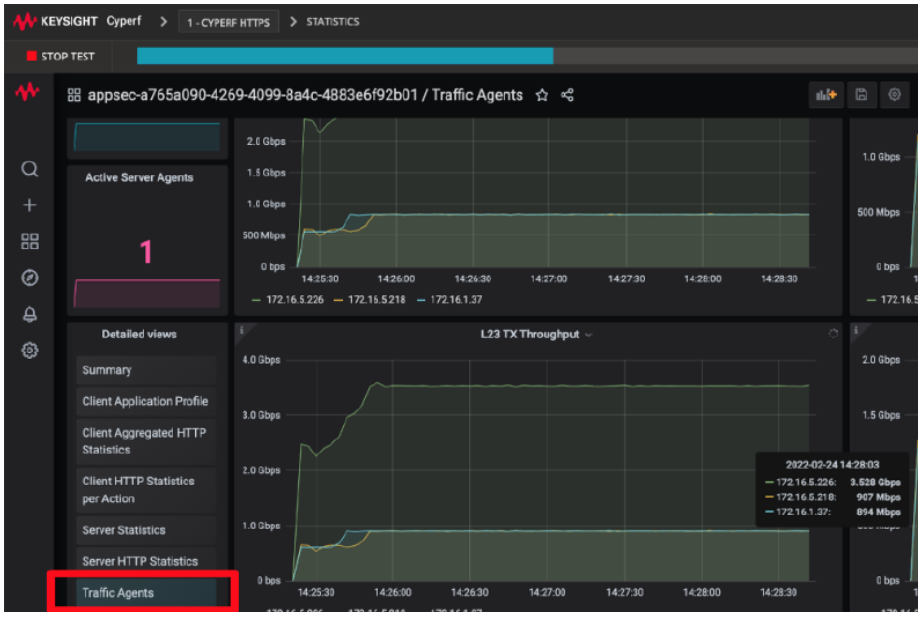
In the **Instantaneous Latency** view, you can distinguish the following three important latency related metrics:

- Connection Latency: the average time it took for the TCP connection to establish
- Time to First Byte: the average time to receive the first data byte since the request was issued
- Time to Last Byte: the average time to receive the full data object/page since the request was issued

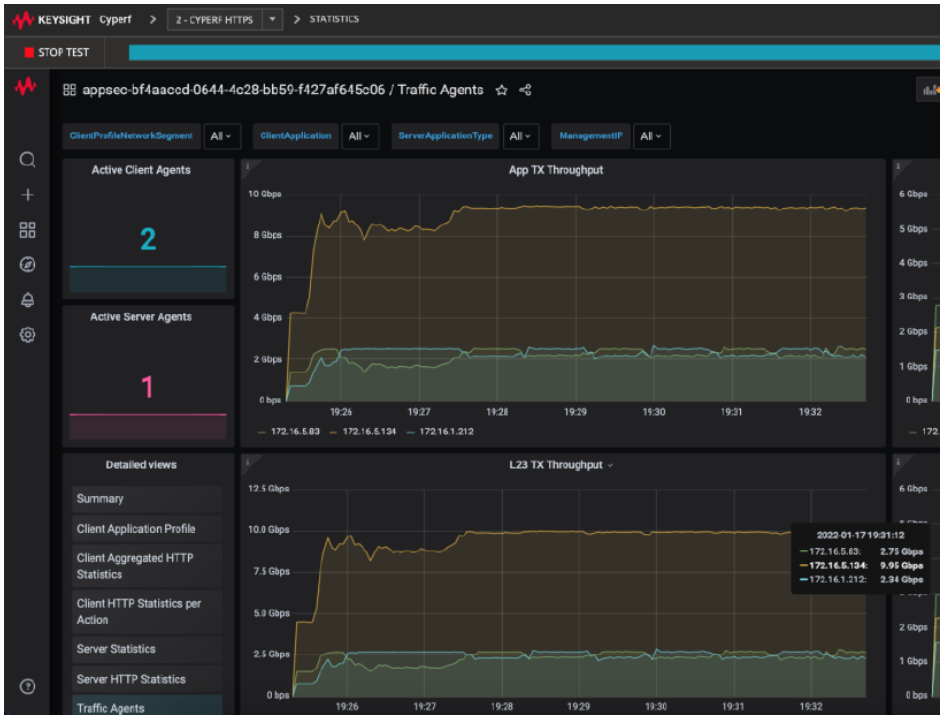
Note: The load of this test is restricted to 5Gbps because of cost related reasons. If the infrastructure is adequately warmed up to push more traffic, the maximum performance obtained on this setup would have been around 14Gbps (the following screenshot is after such a maximum performance test was ran):



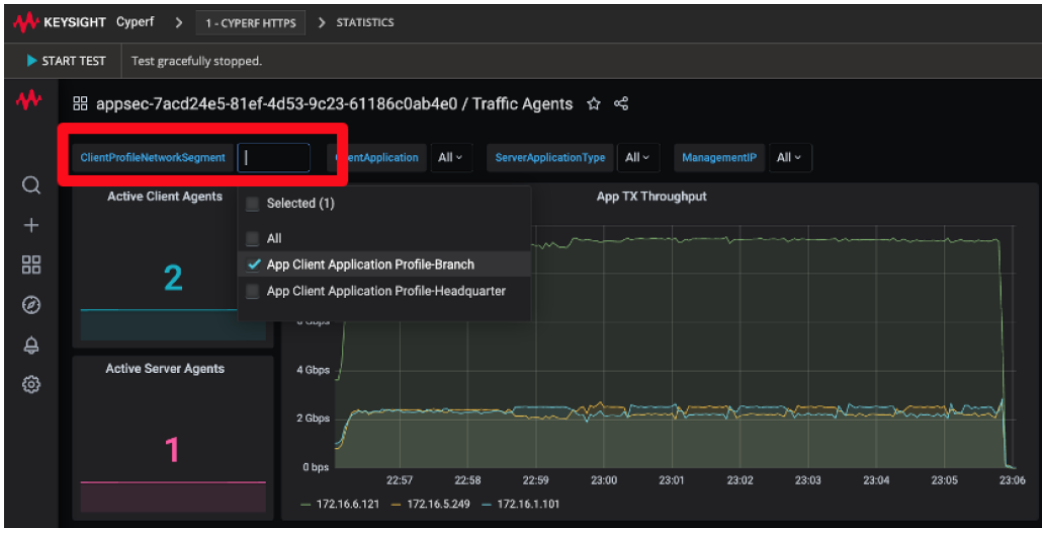
Next, for the current test, you can further investigate more granular details and see how much traffic each agent is performing. Drilling down further from the **Detailed View** pane (in the lower left corner) of the **Traffic Agents** dashboard, observe that most of the traffic is generated by the server (that is, 3.528Gbps) while the clients are balanced (generating 907Mbps and 894Mbps respectively):



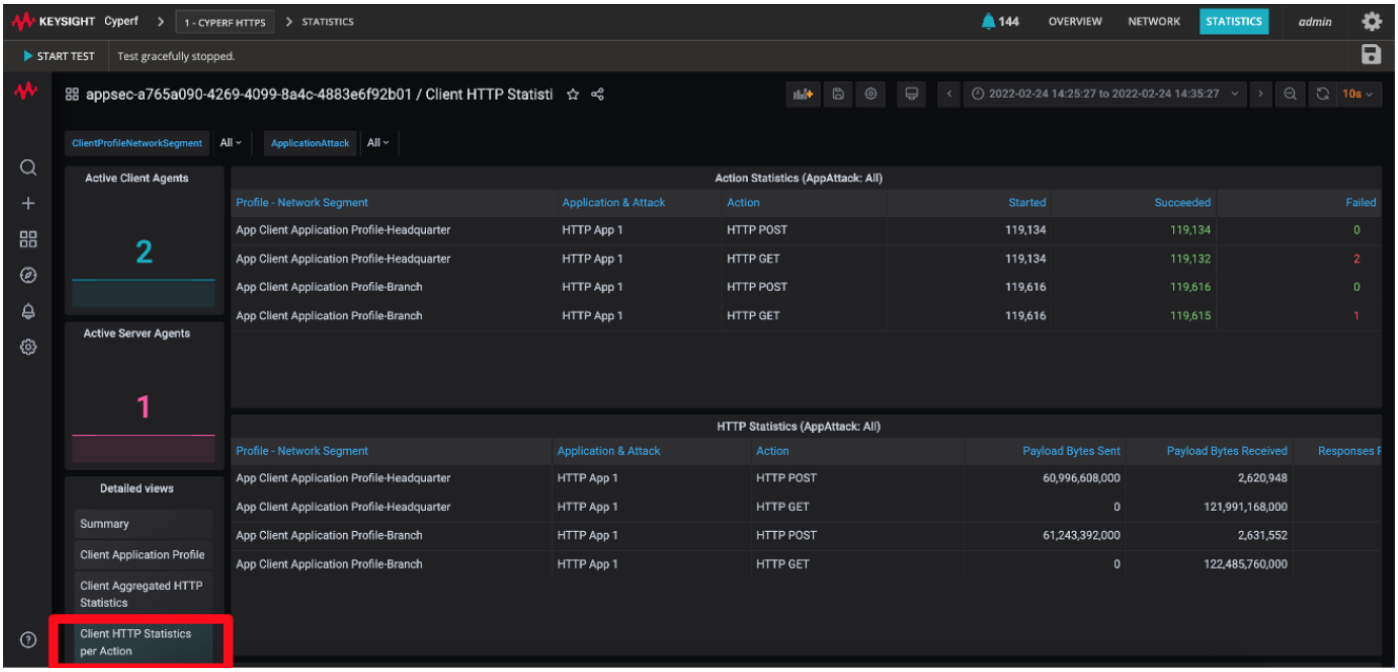
Note: when trying to push the maximum throughput the reason for performance to be up to 14Gbps (bi-directionally) is that the server agent can send traffic at up to 10Gbps (the Public Cloud provider is limiting the maximum bandwidth for this instance type to 10Gbps per direction). Also, the client agents can send traffic at 2.75Gbps and 2.34Gbps respectively for the max bandwidth test, as seen in the following image (taken for the maximum performance test):



A very useful functionality is the ability to apply filters for most of the statistic views. For example, choosing only one network segment (in the upper left corner) presents only the statistics for that respective segment that helps in pin-pointing issues with certain network segments (extremely useful when testing distributed environments):



The **Client HTTP Statistics per Action** dashboard from the **Detailed View** pane provides very granular statistics about each action configured for all the applications in the test:



From the **Detailed View** pane, the ***Client Application Profile*** view (scrolling down into the page) provides detailed TCP and TLS statistics to understand if there are TCP failures like retries or even rests as well as TLS Handshake stats.

Conclusions

The purpose of this test is to baseline the maximum performance of the Device Under Test (DUT) by using simple HTTPS traffic, which most vendors publish in their datasheets. This is a recommended first step to run to build that baseline, with the important note that the test environment as well as the DUT config must be as close as possible to the production version.

Lab 2: Realistic Enterprise traffic mix performance

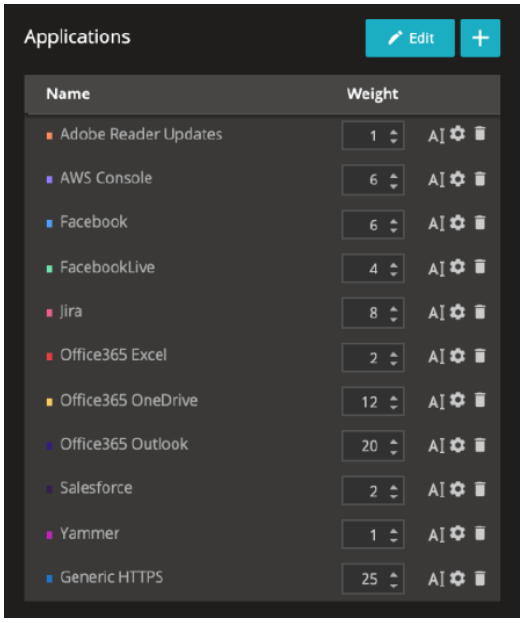
Overview

For the second Lab, instead of generic HTTPS traffic, a much more realistic traffic profile resembling the typical applications traffic flows in an enterprise network is used.

Many Network Equipment Manufacturers publish the performance figures of their devices in ideal scenarios with either generic, large packet HTTP traffic, or with application mixes that are geared, so that those devices render better performance.

Datasheets are a good starting point, but because each environment is unique, it is paramount to test with an application traffic profile resembling the closest production environment. This ensures that the test results are as relevant as possible to take informed decisions.

For this Lab, the emulated application traffic includes multiple Microsoft Office365 applications (like Outlook, OneDrive, or Excel), Salesforce, Jira, Facebook, Yammer, AWS, and so on. The complete list of the application traffic and the associated weights is emphasized next:



For this scenario, the SUT is configured as per previous LAB, with Load Balancing capabilities on as well as default WAF security rules to protect against the SQLi attacks.

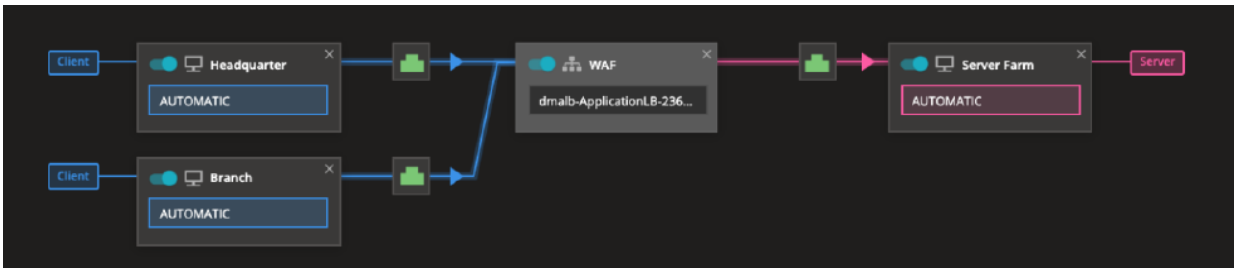
Objective

Performance metrics to be characterized:

- Max throughput: as this is a realistic AppMix, the focus is on the overall achieved throughput, but other KPI are important as well like:
  - Application Failures
  - Application Latencies

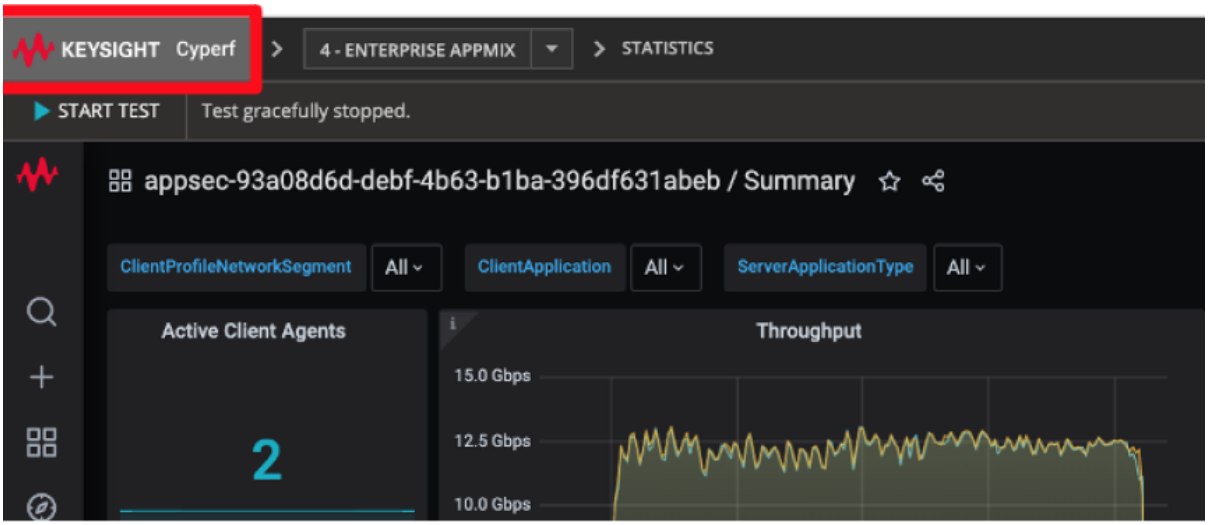
Setup

The setup for this lab is the same as the one described in the Introduction section:



Step-by-Step Instructions

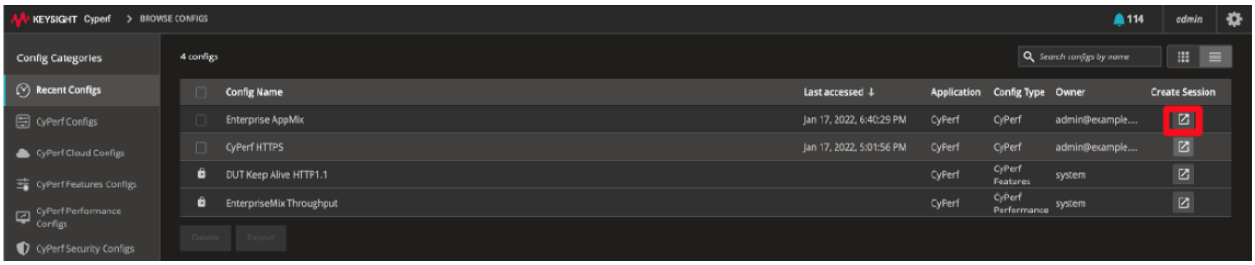
1. After completing the previous test, navigate to CyPerf Controller's landing page by selecting **Keysight CyPerf** as depicted in the following image:



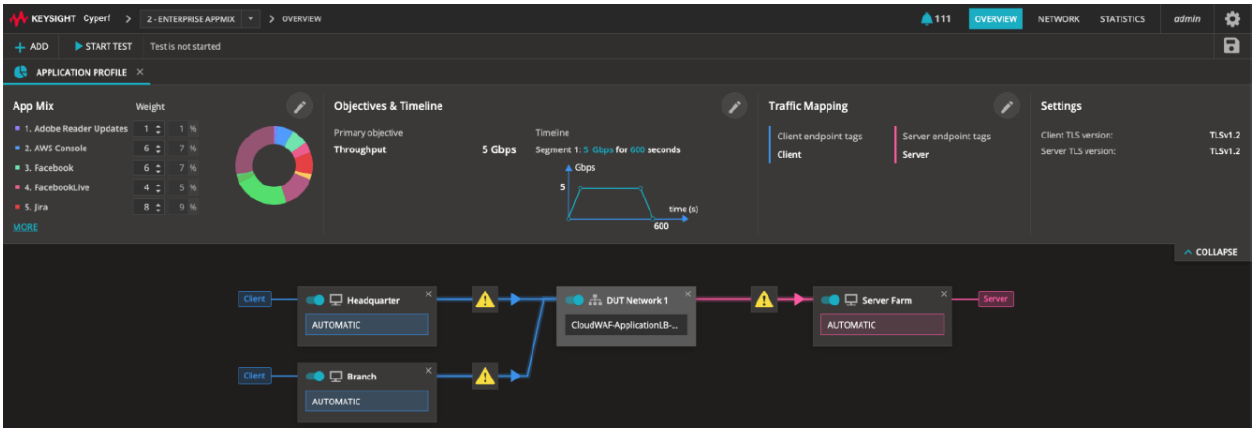
To import the second configuration file, download the Enterprise AppMix.zip file locally to your computer from the following GitHub repo: https://github.com/Keysight/cyperf/CyPerf/CyPerfTestDrive

2. In the CyPerf Controller's landing page, navigate to Browse Configs, and then select the import button on the lower left corner and select the just downloaded Enterprise AppMix.zip file.

3. After the new config file is imported, select the **Create Session** check box on the same row with the new config.



The new config file loads into a new CyPerf Session:



Before looking at the config in more details, assign the CyPerf test traffic agents to the respective Network Segments, like the previous labs:

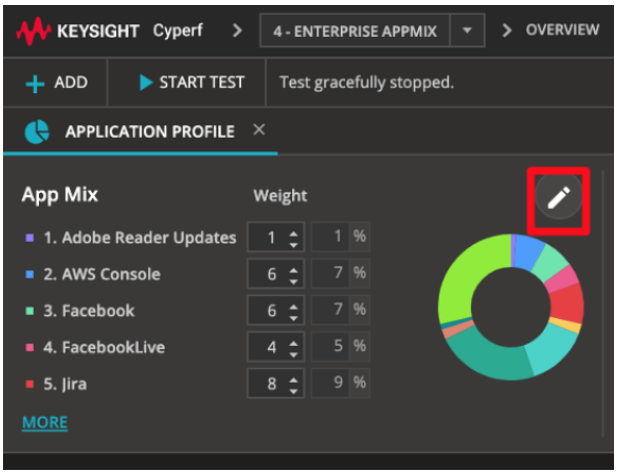
Repeat step 5 from the previous lab, and make sure that the port icon turns green for all the three Network Segments.

Similarly, repeat step 6 from the previous lab as well to configure the DUT address.

4. Now, in the CyPerf test overview page, you can see that the Application Profile component is a bit more advanced, having multiple applications instead of just HTTP as in the previous test. All the other parameters are the same.

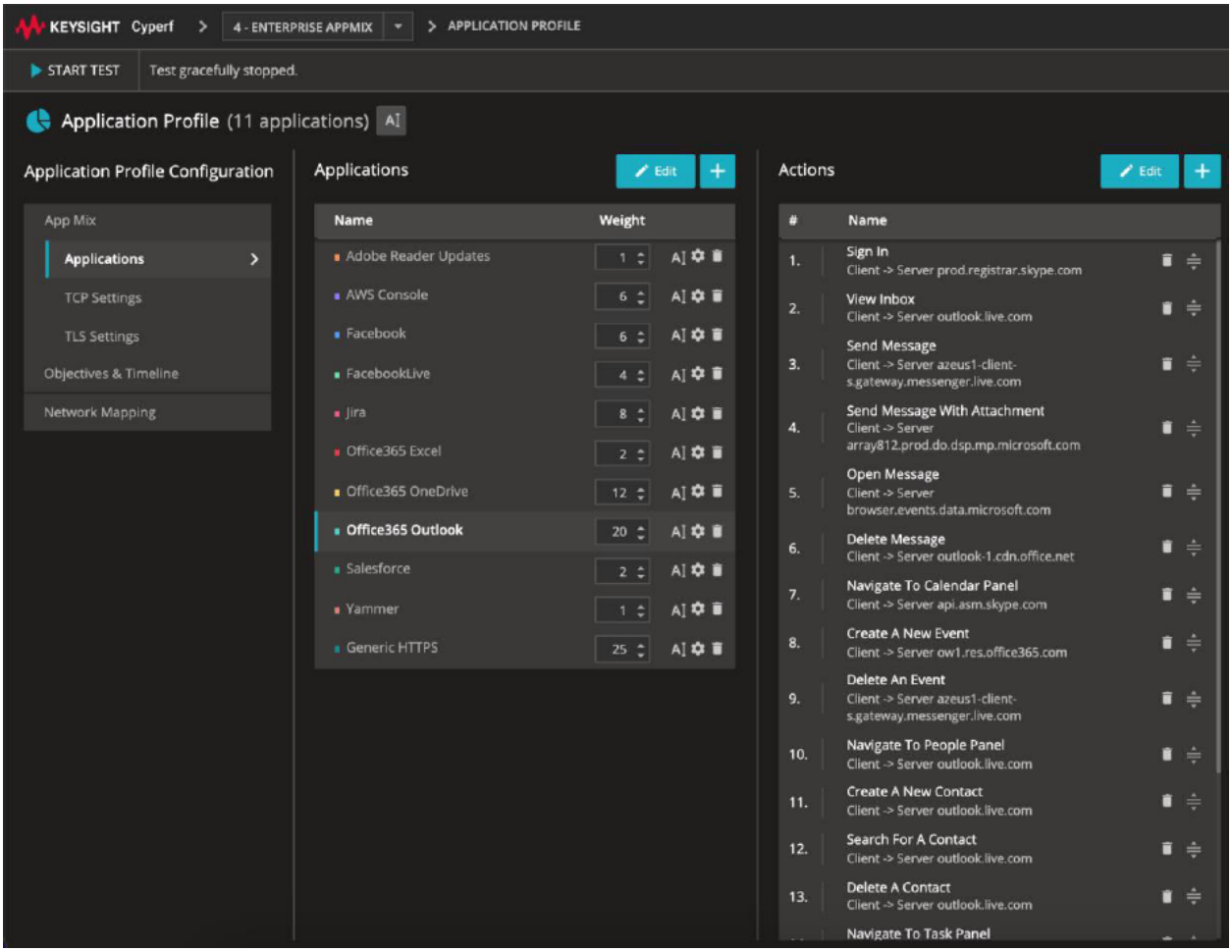
   Before running the test, verify the new Application Profile.

5. Select the edit icon from the Application Profile section as depicted in the following image:



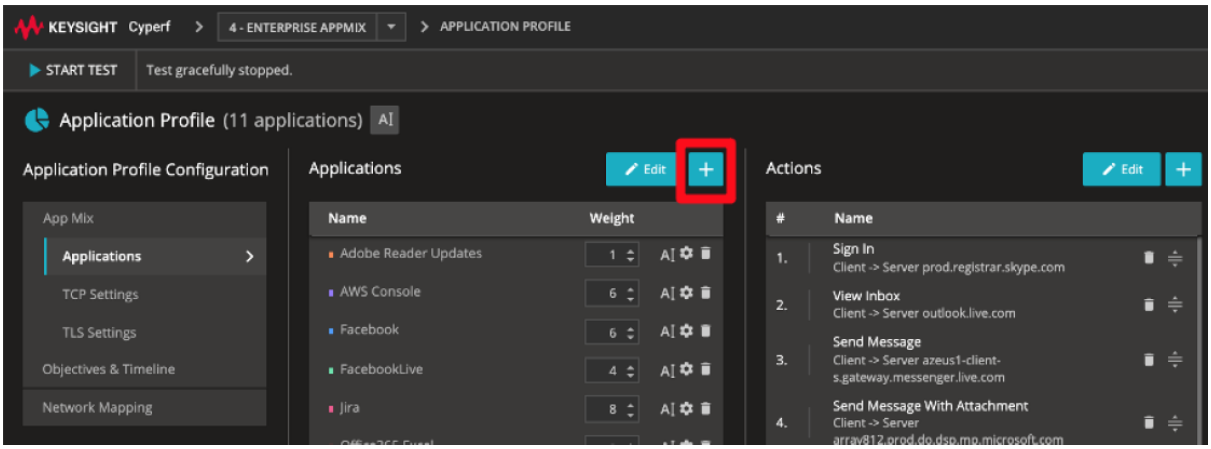
6. In the new menu, you can view the entire AppMix composition with associated weights.

   Selecting any of the application exposes one level of detail deeper, showing all the actions comprising the respective application. For example, Office365 Outlook has the following list of granular actions:

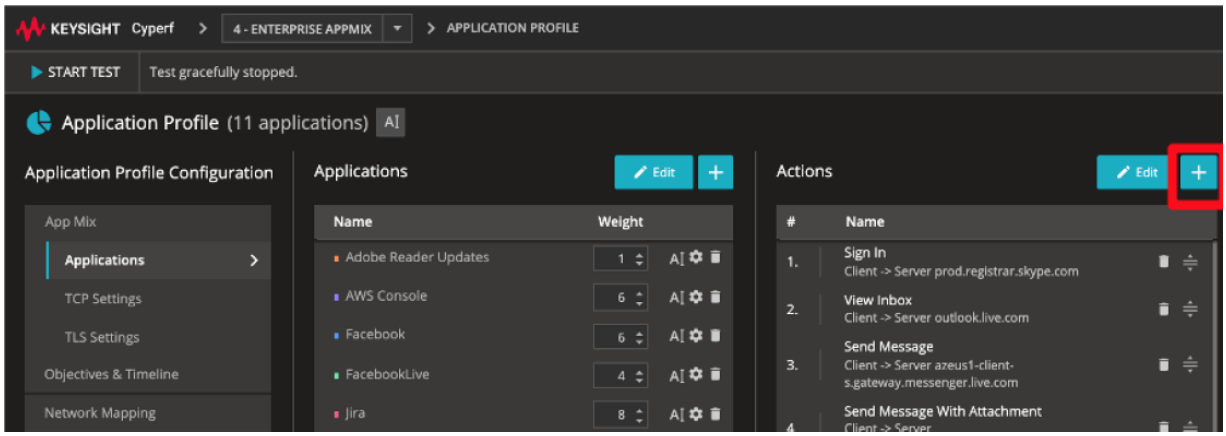


Tip: CyPerf offers the ability to create your own customer Application mixes to resemble various traffic profiles from different network environments:

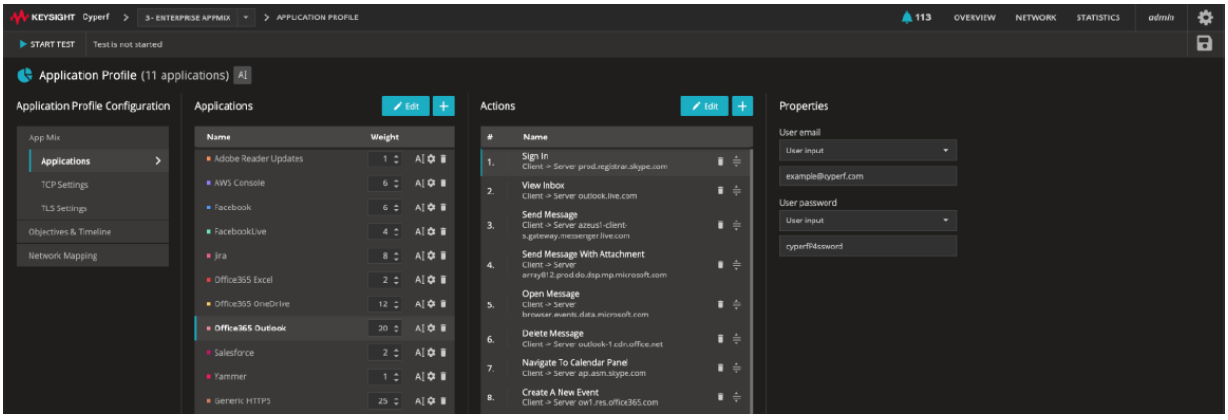
- To add a new application from the library, select the add button (+ icon):



- To customize a specific application (for example, build a custom email user action list), you can edit the application action list as well. To have access to all actions for a certain application, select the add button (+ icon):



- Last but not the least, the application's actions can be customized as well. For example, an email login action can be configured with custom username and password values or even with playlists (lists of usernames and passwords that CyPerf can iterate through to simulate signing in from multiple users):



Important

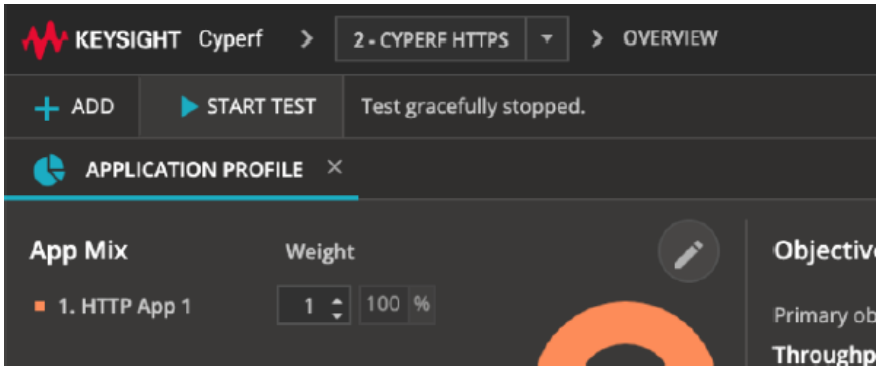
The application mix that is used earlier is an example of a representative traffic mix that can be found in production environments. However, depending on the device under test (for example, WAF, NGFW, IPS/IDS, SASE, and so on) and where it is placed in production networks, as well as what it is targeting to defend, the application mix can be modified to represent that environment. For example, for Web Application Firewalls, you can use more generic HTTP/HTTPS traffic with a various of options like:

- various GET/POST methods with different object types
- think/idle commands to mimic user's inactivity
- different types of TLS versions (for example, TLS 1.2 and 1.3), encryption algorithms

7. In terms of Timeline and Objectives, retain the same test load and duration as in the previous test because of the same reasons.

   After updating the test with the setup/agents' details assigned to the user, you can run the test and interpret the results.

8. Select **Start Test**.



The traffic starts within a few moments after the test traffic agents are configured. The view automatically switches to the STATISTICS dashboard.

Note: After starting the test, two warning messages appear informing that the Objective for the two network segments (that is, Branch and Headquarter) is uneven and this might affect the application distribution percentages: *The distribution of Branch objectives is uneven and the results from the objective distribution chart may be affected*. The reason for the error is that the overall objective is rather low (5Gbps), which needs to be distributed across two segments and across applications that have low percentages in the appmix. Hence, some applications might not hit the exact configured value.

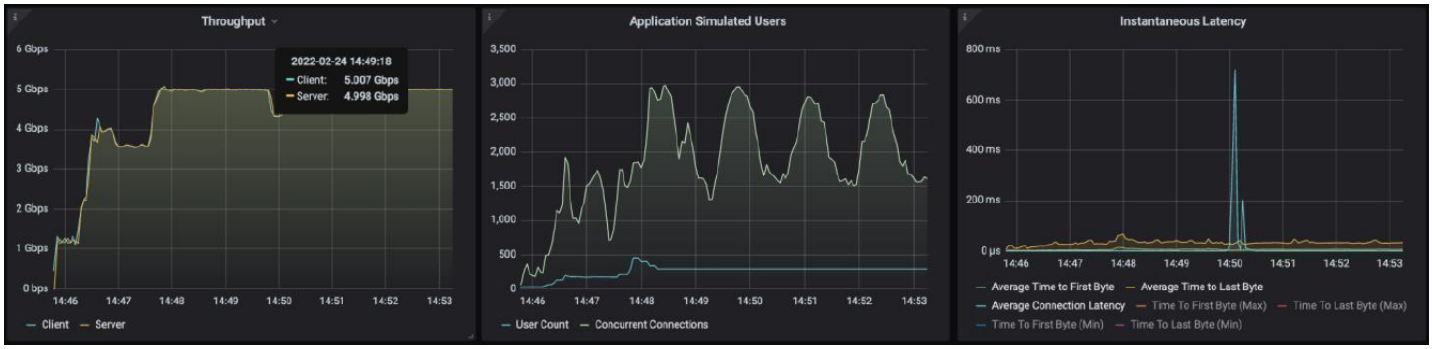
Real-Time Statistics

The following graph provides a view of the real-time statistics for the test. Real-time statistics provide instant access to key statistics that must be examined for failures at the Application, HTTP, and TCP/TLS protocol level.

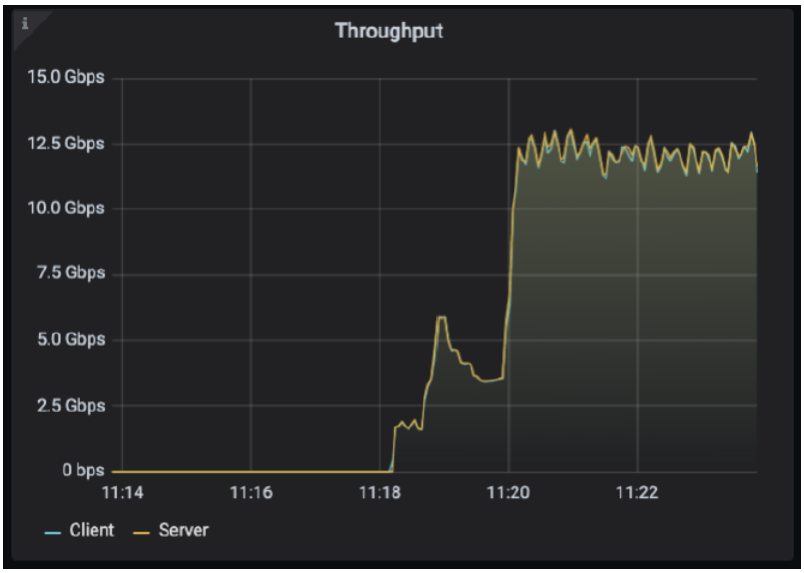
As the test is progressing, observe in the Summary dashboard that CyPerf is dynamically adapting the number of Simulated Users and generating Concurrent Connections to get to the best performance that the underlying setup permits.

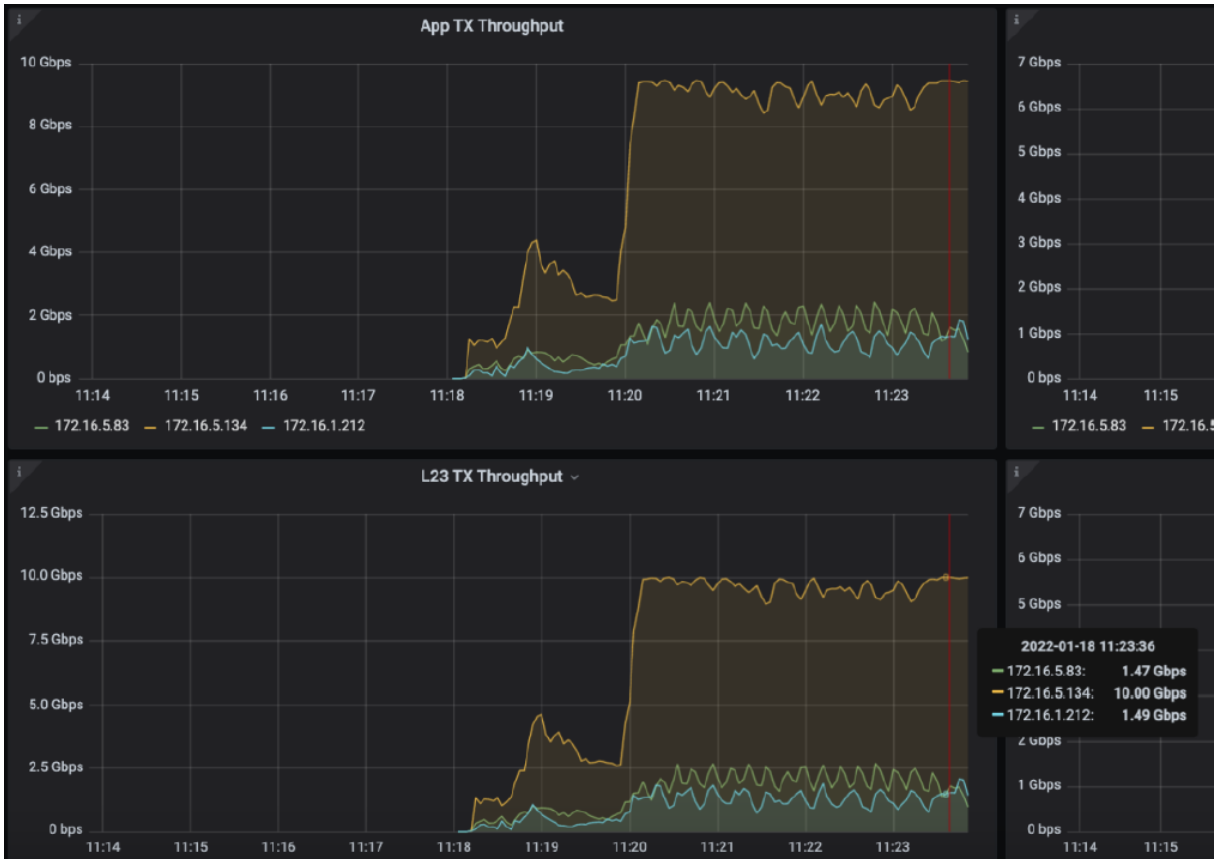
Note: CyPerf's unique ability to dynamically adapt and find the best performance that the underlying infrastructure can handle by modifying the number of users and concurrent connections is a proprietary goal-seeking mechanism allowing users to find that maximum performance much faster than using other test tools.

After a few minutes into the test (approximately 3 to 5 mins), you can see that the goal seeking mechanism stabilizes and the configured throughput is achieved (5Gbps):



Note: If you were to push the maximum throughput for this test as well, you can obtain 12.5Gbps. Therefore, one important difference (for the max throughput test) compared to the previous Lab test is that the overall achieved throughput is lower (14Gbps vs 12.5Gbps). For such a maximum AppMix throughput test, notice that the server agent is still capped at 10Gbps for the Tx traffic and the additional traffic is coming from the client agents that are generating upstream traffic, like a real environment. The following images depict the maximum AppMix throughput test:


Throughput
15.0 Gbps
12.5 Gbps
10.0 Gbps
7.5 Gbps
5.0 Gbps
2.5 Gbps
0 bps
11:14
11:16
11:18
11:20
11:22
Client
Server

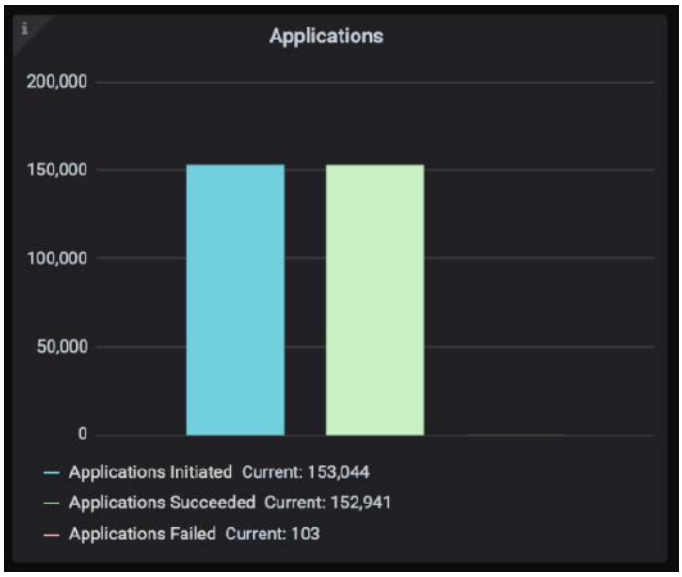

App TX Throughput
10 Gbps
8 Gbps
6 Gbps
4 Gbps
2 Gbps
0 bps
11:14
11:15
11:16
11:17
11:18
11:19
11:20
11:21
11:22
11:23
172.16.5.83
172.16.5.134
172.16.1.212
L23 TX Throughput
12.5 Gbps
10.0 Gbps
7.5 Gbps
5.0 Gbps
2.5 Gbps
0 bps
2022-01-18 11:23:36
172.16.5.83: 1.47 Gbps
172.16.5.134: 10.00 Gbps
172.16.1.212: 1.49 Gbps

Important

In most test scenarios, a realistic application traffic mix is much more resource intensive on the DUT because of the lower average packet size and various application types (stressing the DUT's inspection engine). Nonetheless, this is the recommended and relevant approach to testing Network Devices, not the ideal datasheet test conditions approach.

Other important statistics to be inspected for the current test are described next.

Total Application Initiated/Successful and Failures:

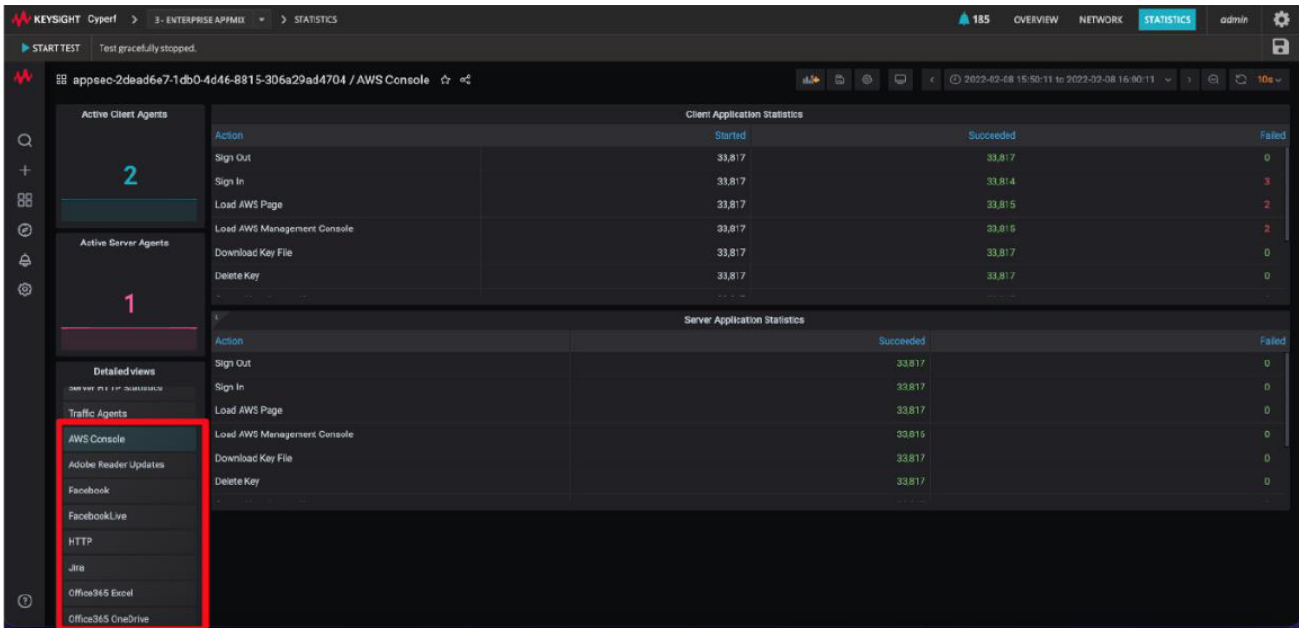


There are only 103 application failures out of over 150,000 initiated that for the most real-world requirements is an acceptable level. You can inspect further to understand the exact applications with observed failures (and even per location or Network Range). You can observe the failures in the **Client Application Profile** view (from the **Detailed View** pane) > **Application Statistics** table:

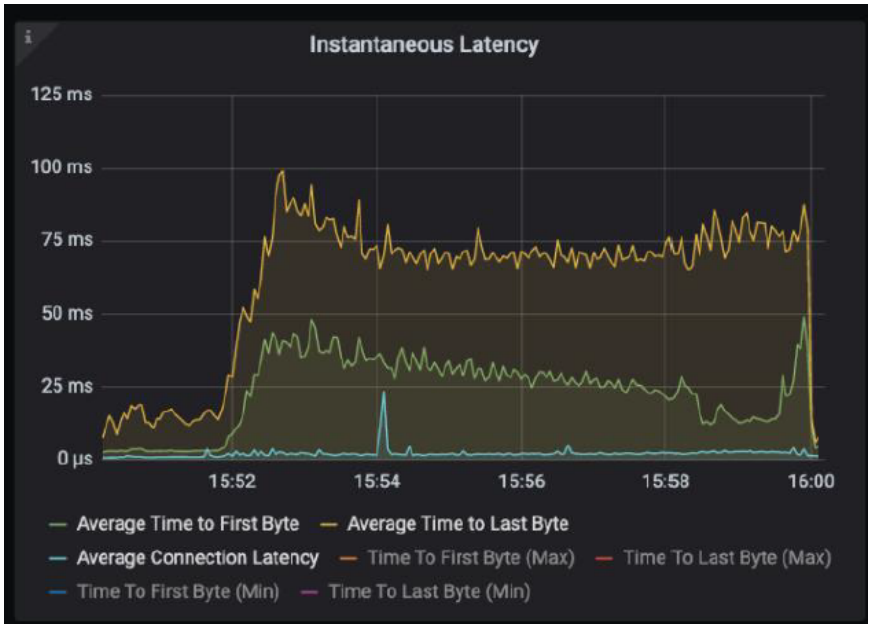
Application Statistics (App: All, NetworkSegment: All)

Application Profile - Network Segment	Application	Bytes Sent	Bytes Received	Connections Initiated	Connections Succeeded	Connections Failed	Connections Aborted
App Client Application Profile-Headquarter	Yammer	169,592,137	1,762,176,047	4,760	4,760	0	0
App Client Application Profile-Headquarter	Salesforce	1,132,557,721	2,931,792,424	3,332	3,332	0	0
App Client Application Profile-Headquarter	Office365 Outlook	10,504,832,668	28,611,867,654	204,589	204,567	22	0
App Client Application Profile-Headquarter	Office365 OneDrive	16,217,369,837	7,805,825,342	87,441	87,437	4	0
App Client Application Profile-Headquarter	Office365 Excel	2,301,217,533	1,528,079,996	35,380	35,374	6	0
App Client Application Profile-Headquarter	Jira	5,440,968,776	10,054,996,880	112,912	112,887	25	0

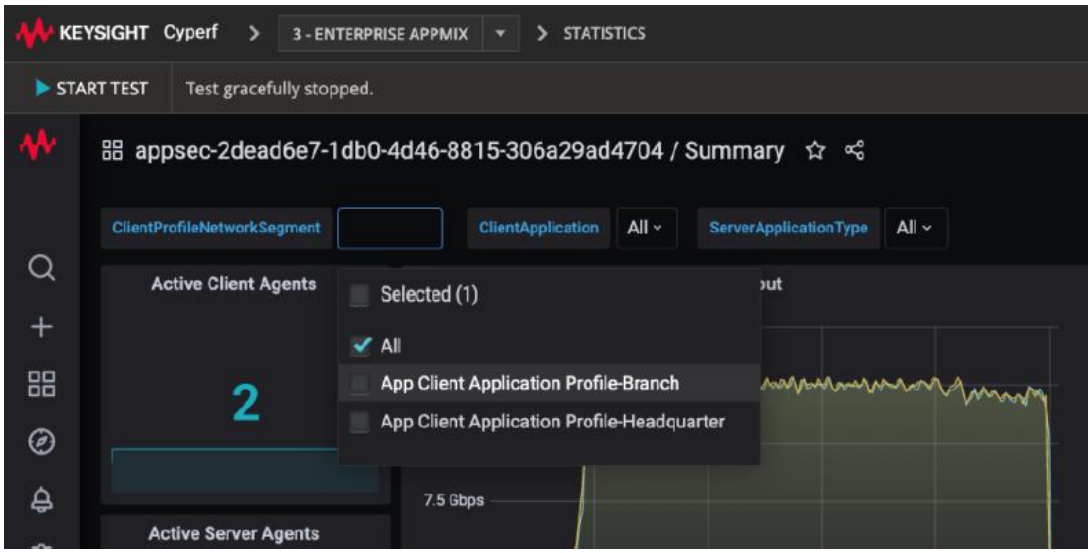
For even more granular statistics, each application in the mix has its own view where statistics for each of the action comprising the respective application is shown:



Other important metrics to be analyzed are in the Instantaneous Latency graph from the **Summary** view:



As you can observe in the preceding graph, the average latencies are stable. Furthermore, you can also inspect the latencies that are experienced by each of the locations (or Network Segments) in the test by applying the relevant filter in the upper left corner on the Statistics screen (ClientProfileNetworkSegment).



Conclusions

In this second test, the true performance of the Device Under Test (DUT) is characterized by using a much more realistic application mix, which is not typically provided in the vendors' datasheets.

This is a very important test to run because it uncovers the true capabilities of various DUTs, and helps the customers to make informed, data-driven decisions.

Lab 3: Realistic Security Efficacy

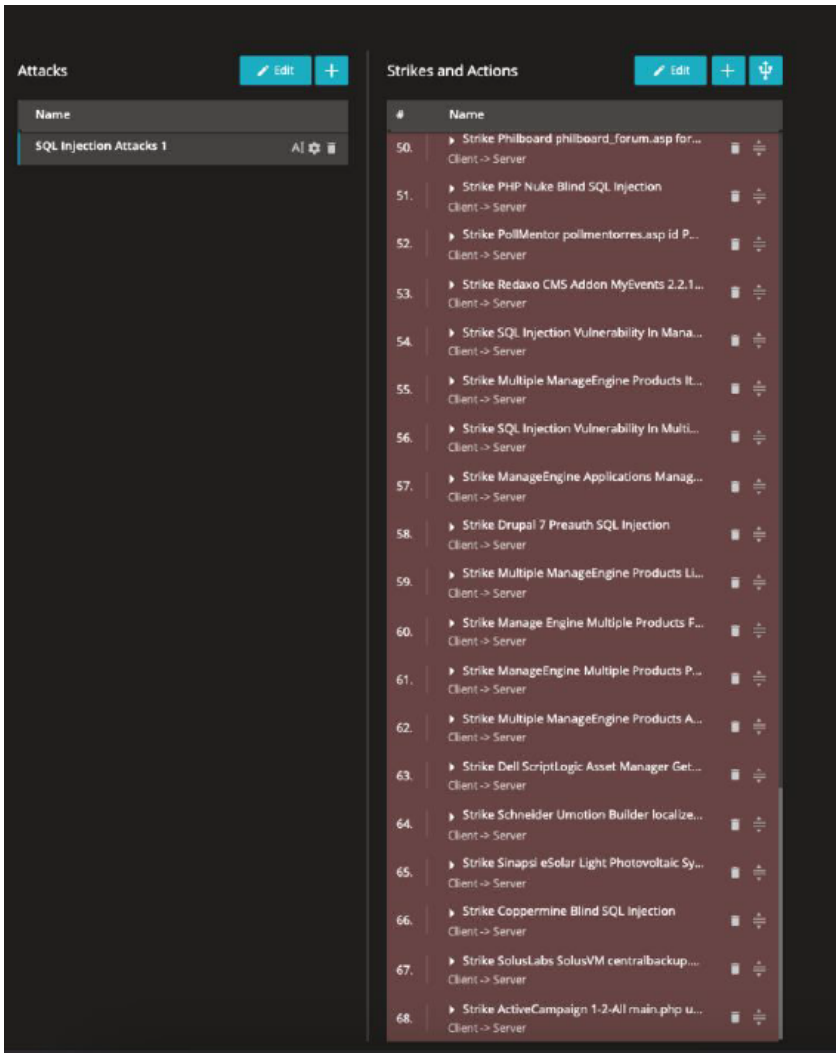
Overview

This lab is one step further to evaluate the security efficacy of the Device Under Test (DUT).

First, keep the previously used realistic traffic profile, so that all security attacks are happening while legitimate traffic is present as well. This is like what is observed in production networks as security attack never happen alone or in isolation.

The really challenging part for the security devices is to accurately identify (and block as needed) the malicious traffic, while not impacting the legitimate traffic, therefore, offering a seamless user experience.

Therefore, along with the previously used Enterprise application traffic mix, you must add a security profile consisting of 68 selected SQLi attacks.



For this scenario, the SUT is configured as per previous LAB, with Load Balancing capabilities on as well as default WAF security rules to protect against SQLi attacks.

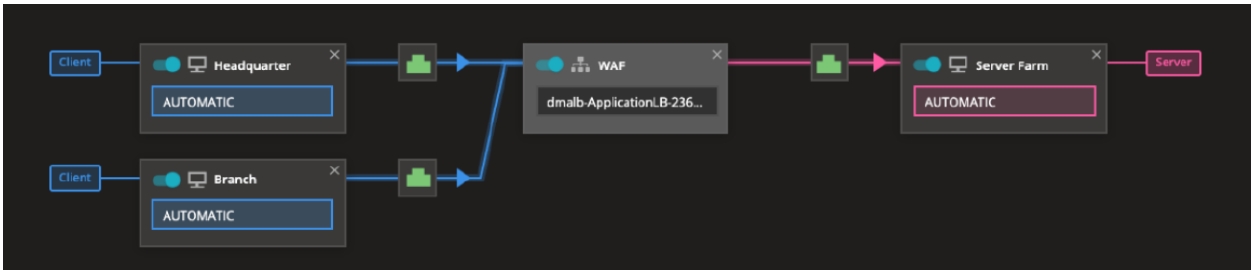
Objective

Performance metrics required:

- Legitimate traffic KPIs: throughput, latencies, application failures
- Security Efficacy: percentage of blocked security strikes

Setup

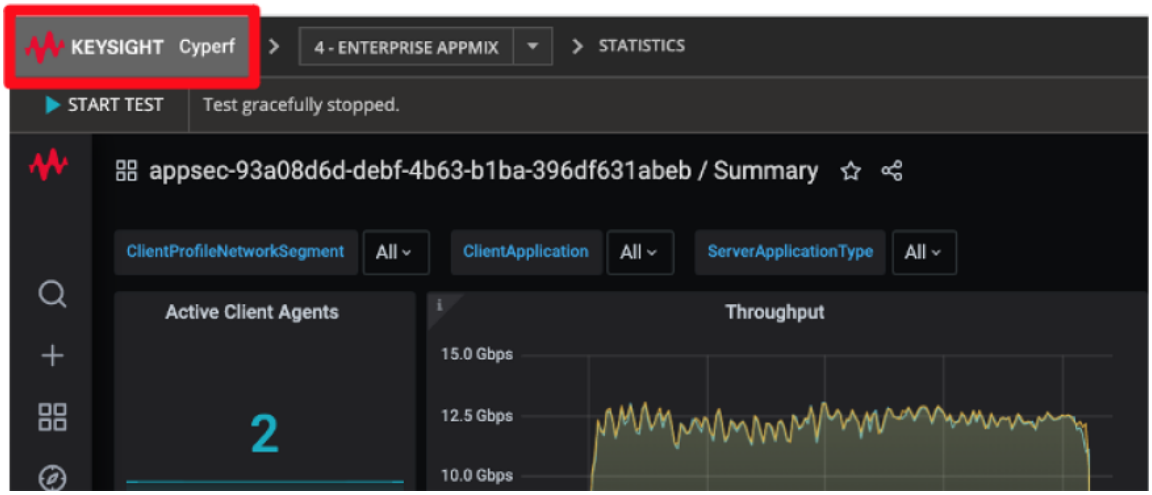
The setup for this lab is the same as the one described in the Introduction section:



The DUT is configured as in the previous tests, with Load Balancing enabled as well as the default SQLi security rules set on blocks. All the other security rules are set on allowed to make sure that the security efficacy of the default SQLi security rules (all generated security attacks will be SQLi attacks) is characterized.

Step-by-Step Instructions

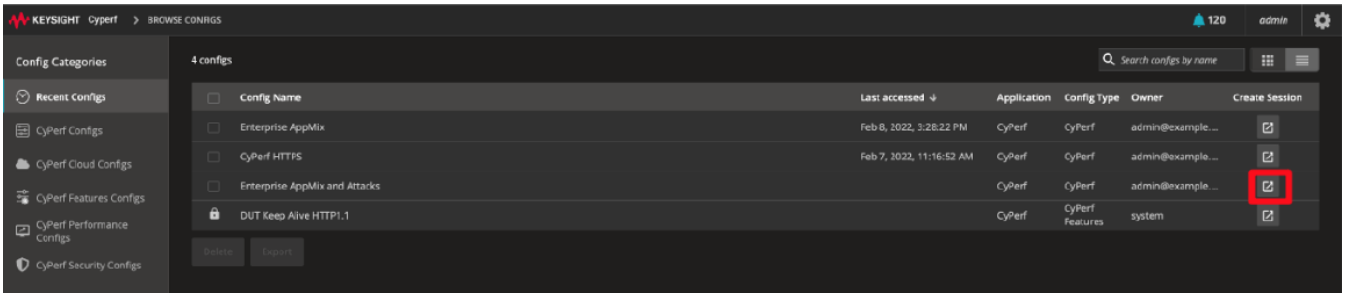
1. After completing the previous test, navigate to CyPerf Controller's landing page by selecting **Keysight CyPerf** as depicted in the following image:



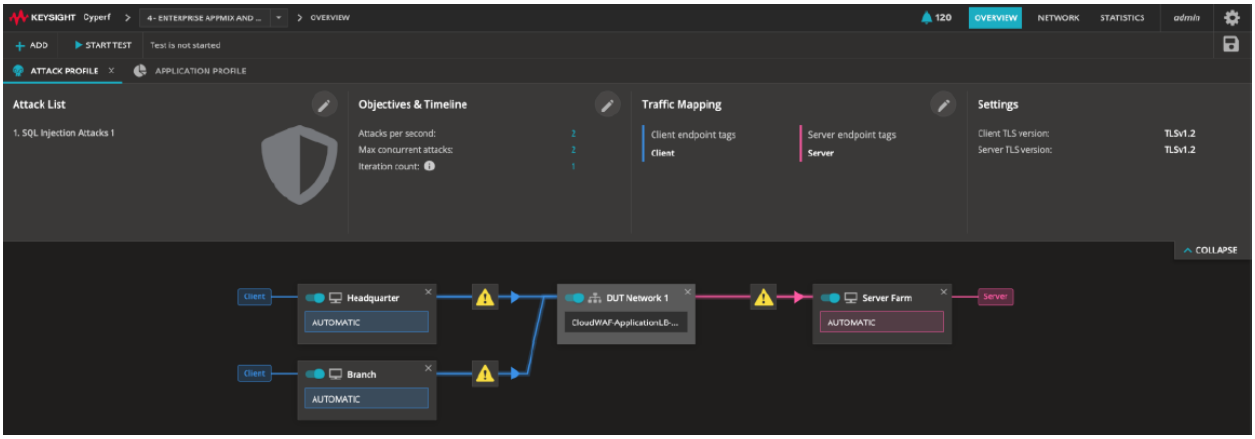
To import the third configuration file, download Enterprise AppMix and the Attacks.zip file locally to your computer from the following GitHub repo:
https://github.com/Keysight/cyperf/CyPerf/CyPerfTestDrive

2. In the CyPerf Controller's landing page, navigate to Browse Configs, and then select the import button on the lower left corner. Select the downloaded Enterprise AppMix and the Attacks.zip file.

3. After the new config file is imported, select the **Create Session** check box as depicted in the following image:



The new config file loads into a new CyPerf Session:



4. Before verifying the config for more details, assign the CyPerf test traffic agents to the respective Network Segments, as done in the first lab.

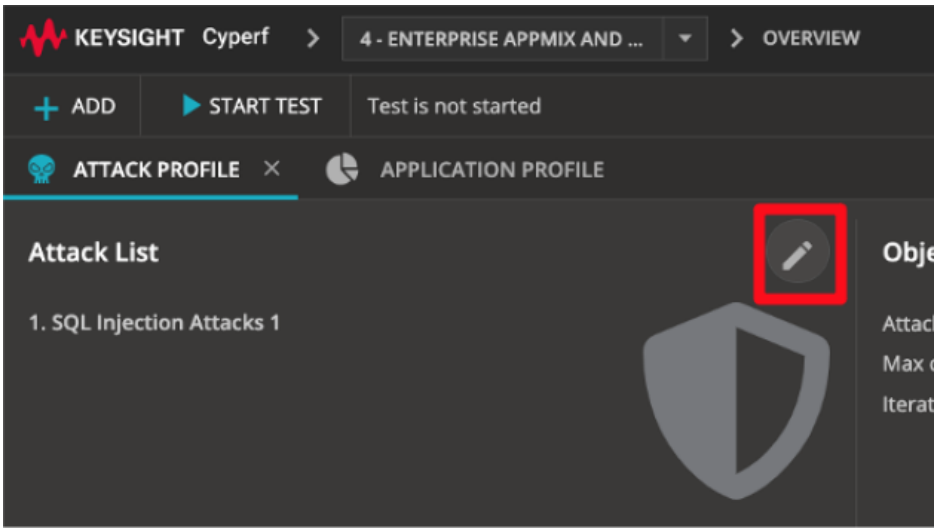
5. Repeat step 5 from the first lab and make sure that the port icon turns green for all the three Network Segments.

6. Similarly, repeat step 6 from the previous lab as well to configure the DUT address.

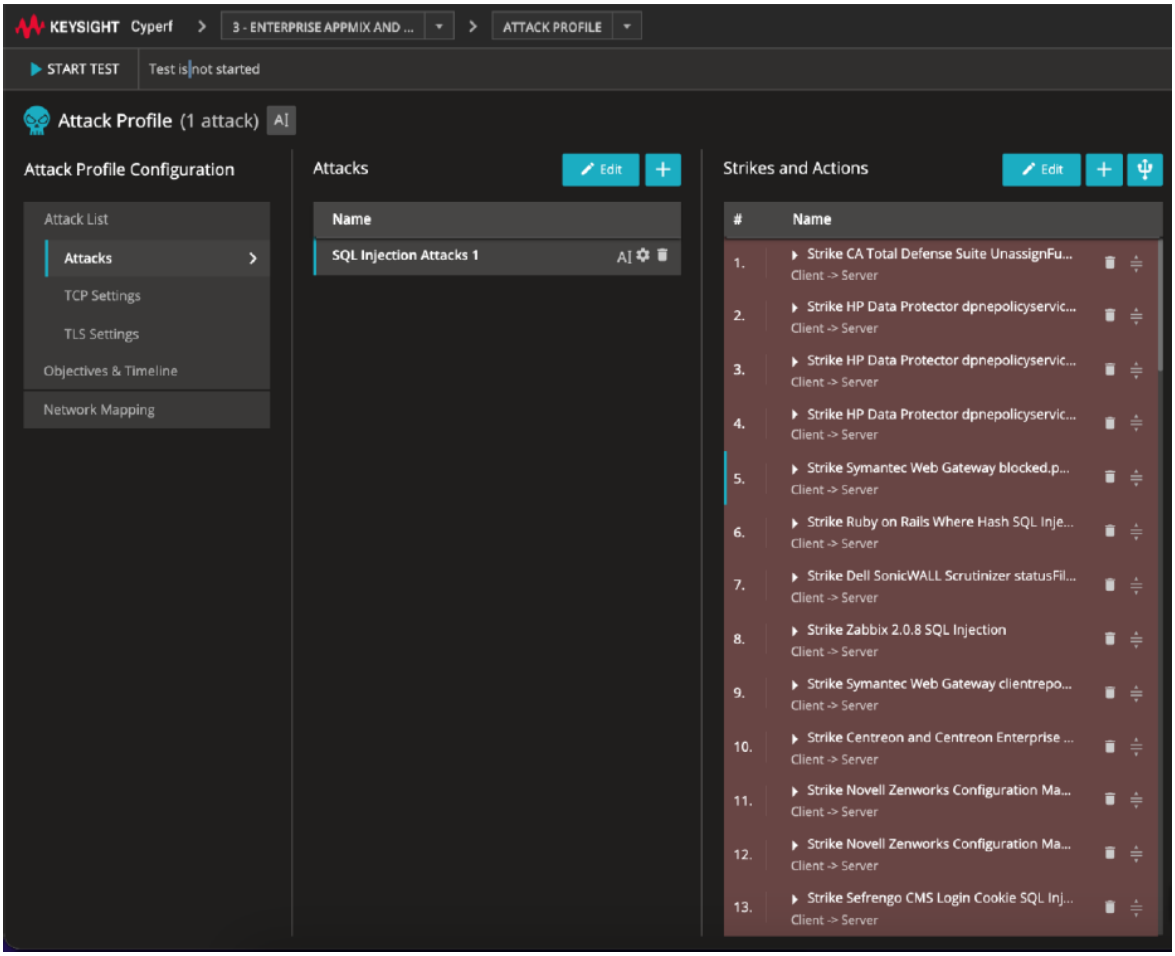
7. Now, in the CyPerf test overview page, observe that along with the Application Profile, there is a second profile—Attack Profile.
Before running the test, examine the new Attack Profile

8. Select the edit icon in the Attack Profile section as depicted in the following image:

.

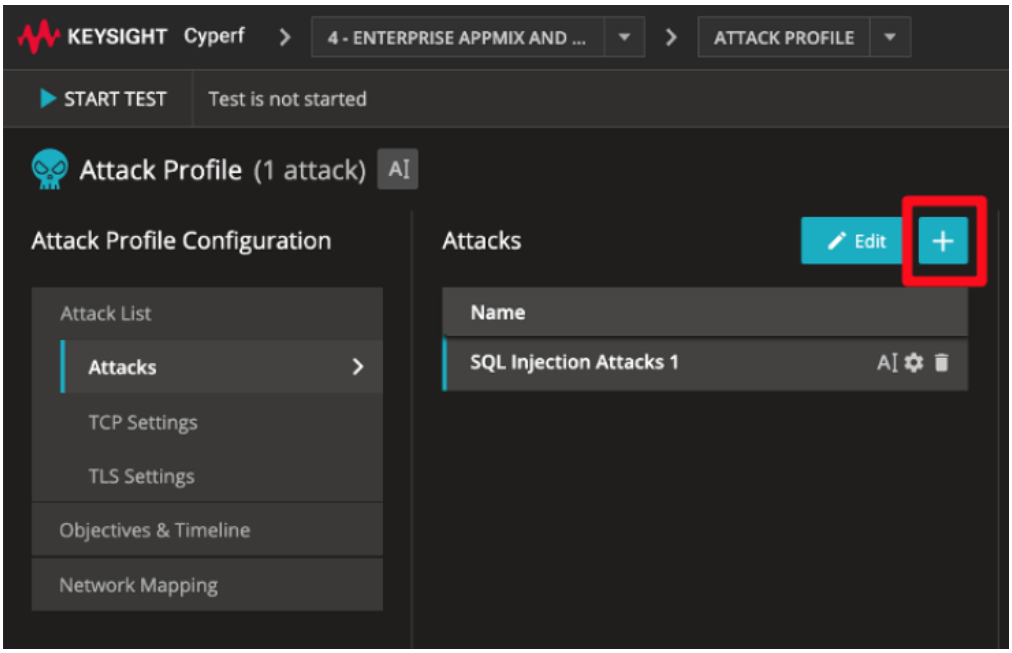


9. In the new menu, the Attack list can be seen, which has one attack with 68 individual SQLi Strikes. Select the SQL Injection Attack 1 to view all the 68 individual SQLi Strikes:

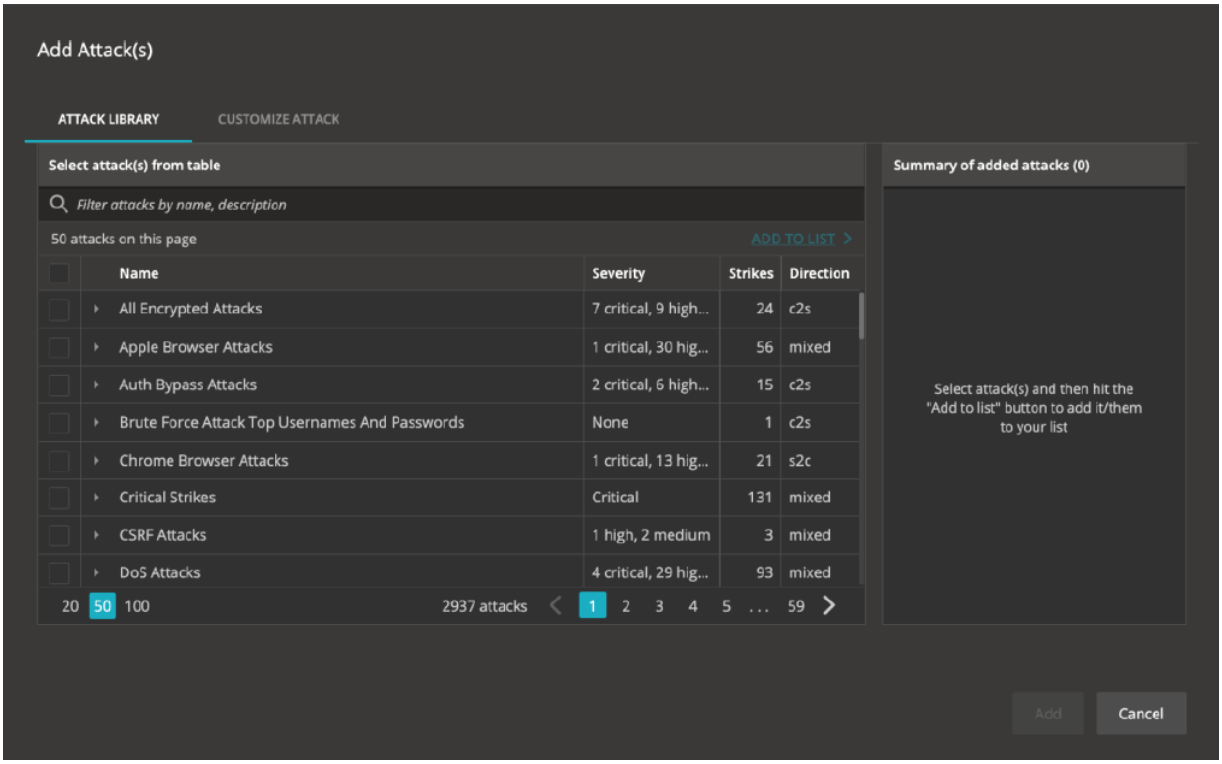


Tip: CyPerf offers you the ability to add the attacks into the test that are more relevant for the type of DUT being tested and obviously the production environment as well:

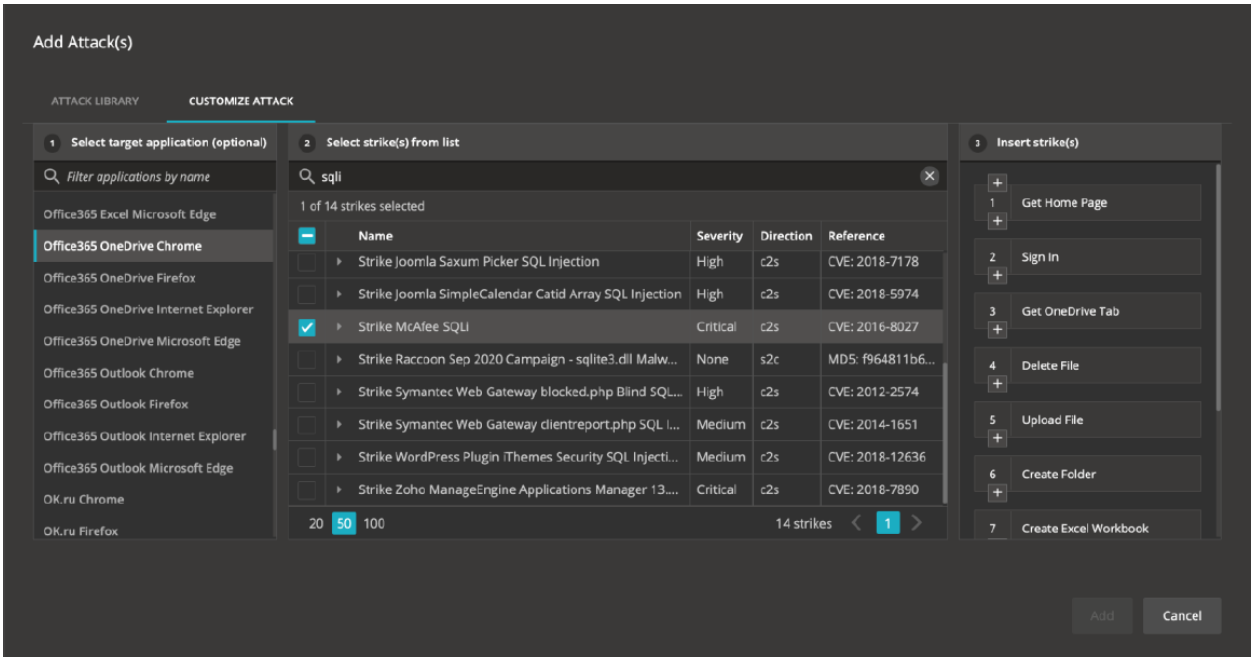
- To add a new attack from the library, select the add button (+ icon):



- The attack library comprises a comprehensive list of attacks containing multiple strikes, grouped based on certain categories, as well as individual strikes.



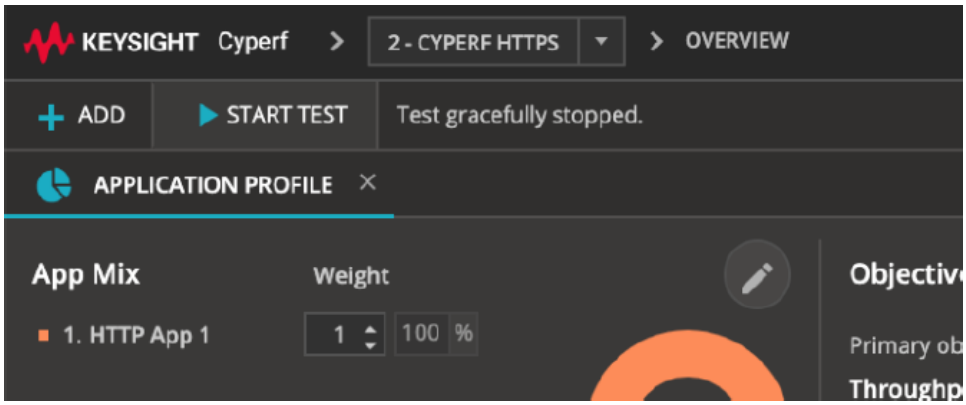
Tip: You can also create custom attacks by combining legitimate application actions with malicious strikes allowing an unprecedented level of realism. This can be done either manually by adding the required application actions to an attack or through the Customize Attack pane in the **Add Attack(s)** dialog box:



10. The Objective and Timeline of the Attack Profile is configured to two attacks per second and two maximum concurrent attacks and iterating only once through the attack list:

All the other test parameters are retained the same as in the previous lab. After the test is updated with the details of the setup/agents assigned to the user, you can run the test and interpret the results.

11. Select **Start Test** as depicted in the following image:



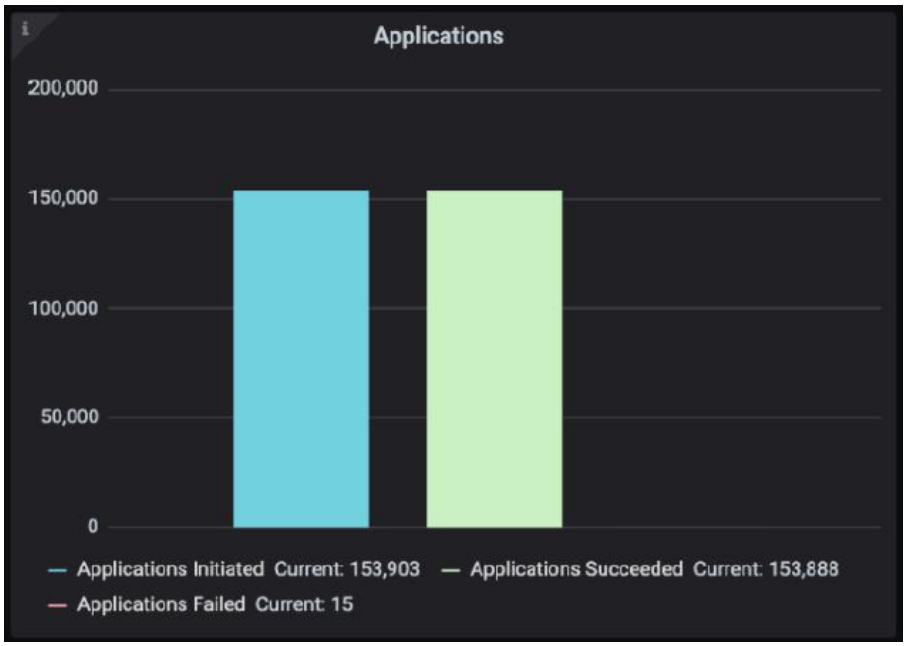
The test traffic agents are configured, and in a few moments, the traffic starts. The view automatically switches to the STATISTICS dashboard.

Real-Time Statistics

First, in terms of application traffic, a similar behavior to the previous test is seen and after a few minutes into the test (approximately 3 to 5 mins), you can see that the goal seeking mechanism stabilizes. The resulting throughput is achieved as well:

Note: One might notice that the test progress bar is not displaying the total and elapsed test duration (next to the **Start/Stop Test** button). The reason is that the test is configured to run one iteration of the attack list and the duration of running the attacks is unknown as in many cases, that depends on how the device under test responds to the attacks. Another configuration option is to set the Iteration Count parameter under the Attack Profile to 0. This configuration provides the ability to set a fixed duration for which the test iterates through the attack list.

One important thing to check is if/how many application failures are there for this test. In the Applications graphs, observe that only 15 application iterations failed out of over 150,000:



Therefore, on the legitimate traffic front, we can conclude that the behavior is like the previous test, hence, at this level of traffic, there is no significant impact when it comes to application traffic.

Important

In general, adding or enabling more policies, features, and functionalities on the Network Security devices lead to an impact to the maximum performance that the device can handle. Therefore, it is very important, when characterizing the performance of any DUT to make sure that its configuration reflects the production needs and requirements.

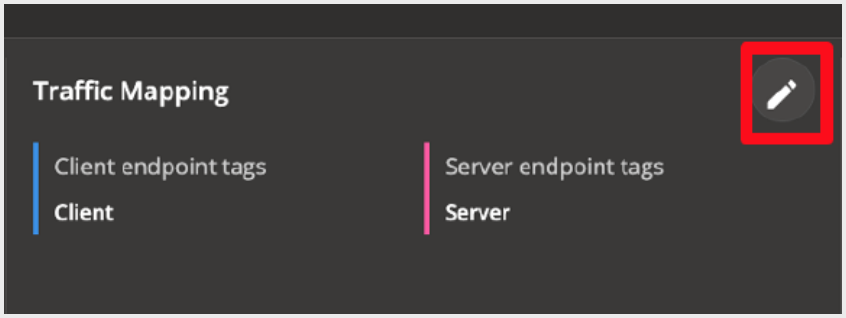
Next, examine what the security efficacy of the default SQLi security rules is relative to the SQLi attack samples being selected. From the Summary dashboard, observe that out of the total 136 strikes (134 client to server and 2 server to client), 38 are allowed:

The reason behind a total of 136 strikes that is run is that the same strike list (of 68 strikes) was run from both client locations (or Network Segments).

Therefore, it seems that the security efficacy of the WAF for default SQLi security rules (tested against the selected SQLi attack samples) **is approximately 72 percent.**

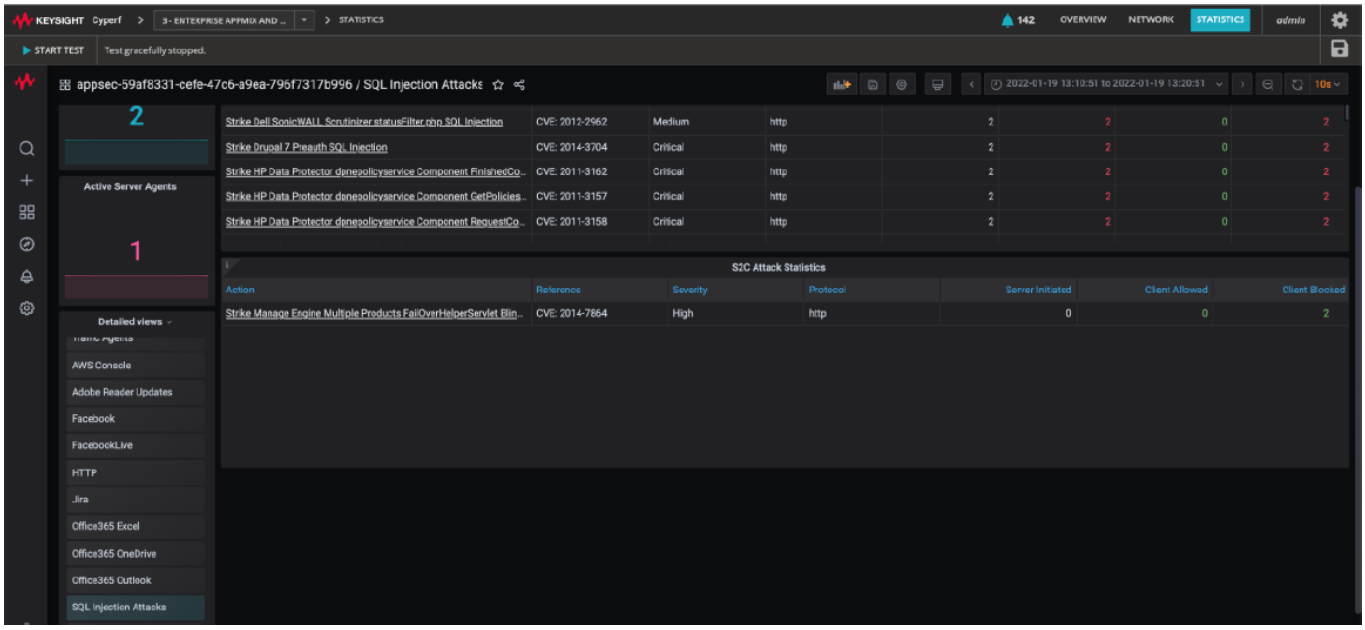
Important

Using the advanced traffic mapping functionality (accessible through the edit icon ‘ ’ from the **Traffic Mapping** section in the test overview page), you can configure the security profile to be run only from one location (for example, Branch) to emulate complex scenarios where different locations are generating different traffic patterns (same traffic mapping feature can be granularly applied to each legitimate application as well). More details on how to configure traffic mapping and Network Segments tags also available in the CyPerf’s User Guide.



Furthermore, you can also check about the exact security strike being allowed and which blocked for an even better visibility into the associated risks relative to the production environment.

In the SQL Injection Attacks detailed view, each of the security strikes are individually presented with their allowed/block status:



For example, the Drupal 7 Preauth SQL Injection strike, which is allowed in the test, can be assigned a different security risk level based on the real production environment characteristics (for example, if Drupal is being used or not and what version).

Conclusions

In this third and last test, the true performance of the System Under Test (SUT) is characterized by using malicious traffic in conjunction with a realistic application mix. First, the impact on the legitimate traffic was evaluated, and then you observed the SUT security efficacy of the default SQLi security rules relative to the SQLi attack samples being selected.

In the rush to move to cloud networks, companies are finding it difficult in rightsizing cloud and optimizing cost while minimizing user disruptions. Testing the performance of elastic environments and isolating bottlenecks is also challenging.

Install Keysight's CyPerf agents in cloud instances to benchmark application performance of cloud instances that involve switching, netting, or going through internet gateways to another cloud location (multi-cloud topologies) or on-premises (hybrid topologies).

For more information on Keysight Technologies' products, applications, or services, please visit: www.keysight.com