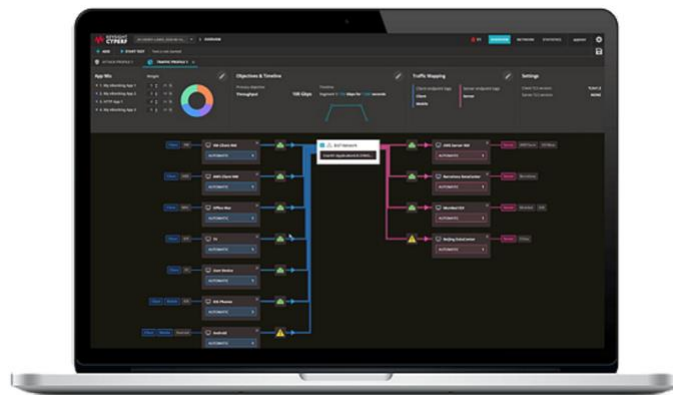


# CyPerf Test Drive

Testing That Replicates Your Network in Action



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

# CyPerf Test Drive

## Table of Contents

Overview - The need for Hybrid, Distributed Network Testing .....	3
Lab Introduction .....	4
Overview .....	4
Setup .....	5
Resources and Prerequisites.....	6
Lab 1: Setup Baselineing – Max HTTPS performance .....	6
Overview .....	6
Objectives .....	7
Step-by-Step Instructions.....	7
Result Analysis.....	17
Real Time Statistics .....	17
Conclusions.....	21
Lab 2: Realistic Enterprise traffic mix performance.....	22
Overview .....	22
Objective .....	22
Setup .....	23
Step-by-Step Instructions.....	23
Real-Time Statistics .....	27
Conclusions.....	31
Lab 3: Realistic Security Efficacy .....	32
Overview .....	32
Objective .....	33
Setup .....	33
Step-by-Step Instructions.....	33
Real-Time Statistics .....	38
Conclusions.....	40

## Overview - The need for Hybrid, Distributed Network Testing

“If you can’t measure it, you can’t improve it.” This quote could not find better applicability than 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. As an enterprise moving to more cost-effective and elastic off premises networking and storage, you face new challenges— are you delivering high-quality access to users, devices, and cloud services everywhere in your distributed, disaggregated networks? Is your cybersecurity infrastructure enough to limit exposure across your on- and off-prem networking? Are your security policies dynamically adjusting to your auto-scale events? Your perimeter-less, elastic, 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

For the following Labs we will be using a cloud-based setup with distributed, lightweight traffic agents which will generate realistic application and malicious traffic to assess the performance and security efficacy of a cloud-deployed Web Application Firewall - WAF (i.e. DUT, or device under test).

We have selected a cloud-deployed WAF for the device under test as a readily available security control 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:

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

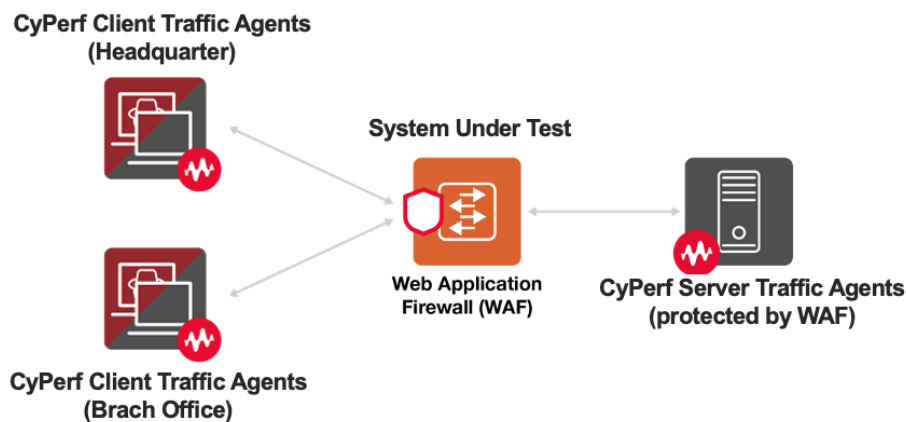


The main components of Keysight's CyPerf (test tool) are:

1. **Test Controller**: web-based UI for configuring and running tests, viewing real-time statistics and reviewing results
  - The CyPerf Controller will be deployed in the cloud and will be 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) will be deployed in the cloud to generate legitimate and malicious traffic going through the cloud-deployed WAF (i.e. DUT).

## Setup

Below is the high-level diagram of the setup 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 particular 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 (e.g., 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, please refer to the [product datasheet](#).

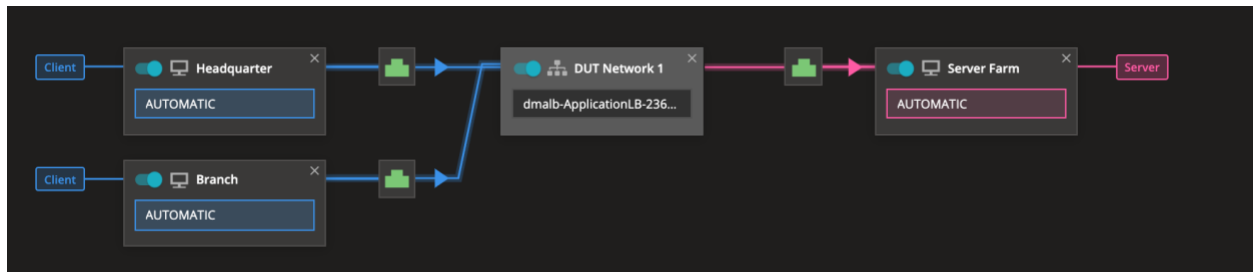
### CAUTION

In this particular Lab, due to cost related considerations, we have forcefully kept **one single server agent** in the group. Also due to the same considerations, although the cloud instances type used for the CyPerf traffic are capable of generating 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, please refer to 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:

- Configuration files: each lab will start from a configuration file that is available at above 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 above 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 above 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 Baseline – Max HTTPS performance

### Overview

As the very first step, we will baseline the maximum performance of the System Under Test (SUT) using HTTPS traffic.

For this scenario we will have the DUT configured with Load Balancing capabilities on as well as default WAF security rules to protect against SQLi attacks.

CyPerf will be configured with the following HTTP profile:

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

Also, as most traffic in today's networks is encrypted, we will be using TLS traffic, which the Device Under Test will need to decrypt, inspect and re-encrypt (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).

## Objectives

CyPerf provides a very large set of Key Performance indicators, however for this test we will focus on the following:

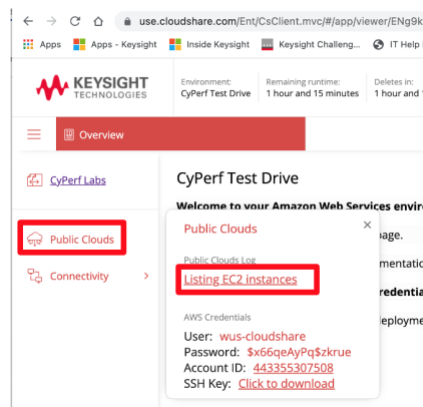
- Maximum throughput
- Latencies

For the test, we will import a configuration file that was already created beforehand to save time. Nevertheless, we will look at the most important parameters as per the step-by-step instructions below.

## Step-by-Step Instructions

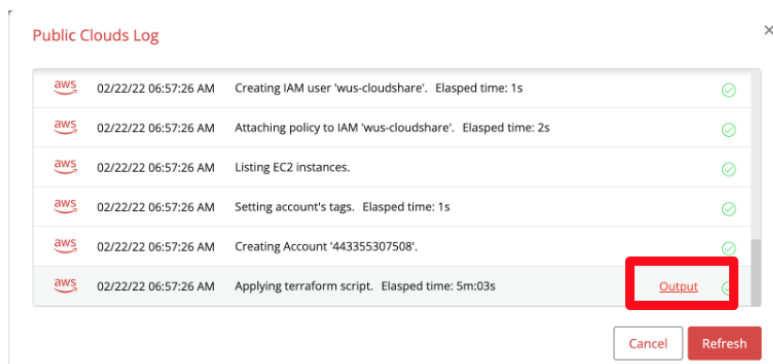
Initially we will need to connect to the CyPerf Controller, and for this we will need its IP address.

For this, in your CloudShare web page, please click on **Public Clouds** menu and then on **Listing EC2 instance**:

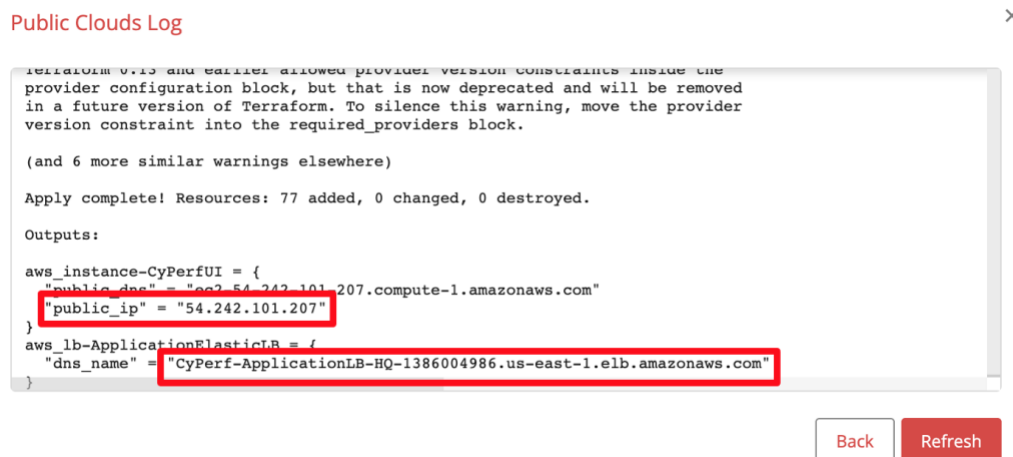


**Note:** if the **Listing EC2 instance** link is not available, please make sure that you started the AWS deployment as per the introduction video (stream it from [here](#)).

In the new dialog, click on the **Output** link next to the *Applying terraform script* line:



Scroll all the way down through the Logs and at the bottom, the IP address of the CyPerf Controller (i.e., `aws_instance-CyPerfUI`) as well as the DNS name of the DUT (i.e., `aws_lb-ApplicationElasticLB`). Please copy both in a text file.



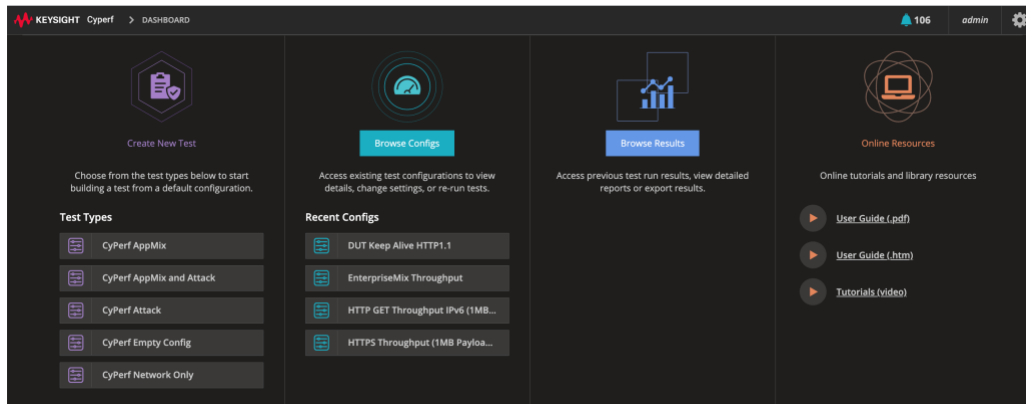
Next, we will log in to the CyPerf Controller and start configuring the tests:

1. Connect to the CyPerf Controller: open a Web Browser and type the IP address of the CyPerf Controller. Use the following credentials to log in:
  - UserName: admin
  - Password: CyPerf&Keysight#1

Congratulations! You have logged in for the first time in your own CyPerf instance.



Note: You will need to accept the TLS certificate of the Controller Web UI since this is a self-signed certificate.



Next, let's import the first configuration file. Before importing the configuration file, please make sure to download "CyPerf HTTPS.zip" file on your local machine from the following GitHub repo:

[https://github.com/Keysight/cyperf/tree/main/CyPerfTestDrive/Configuration\\_Files](https://github.com/Keysight/cyperf/tree/main/CyPerfTestDrive/Configuration_Files)

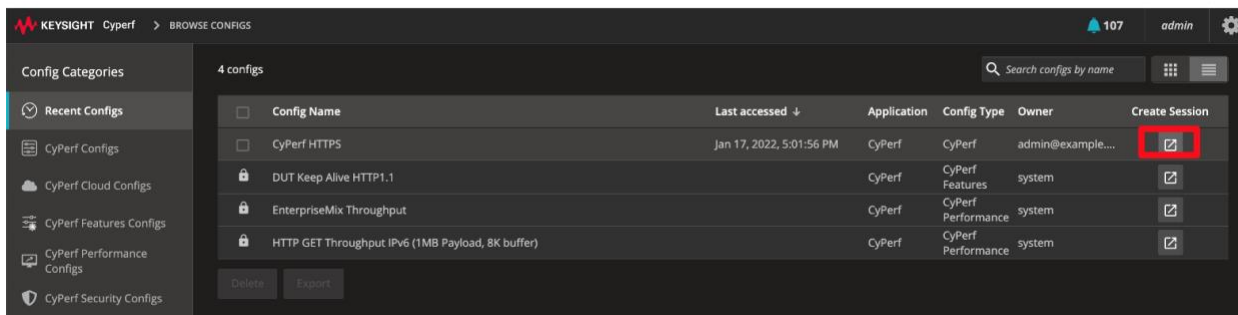
## IMPORTANT

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 can be found under Browse Configs menu -> CyPerf Cloud Configs

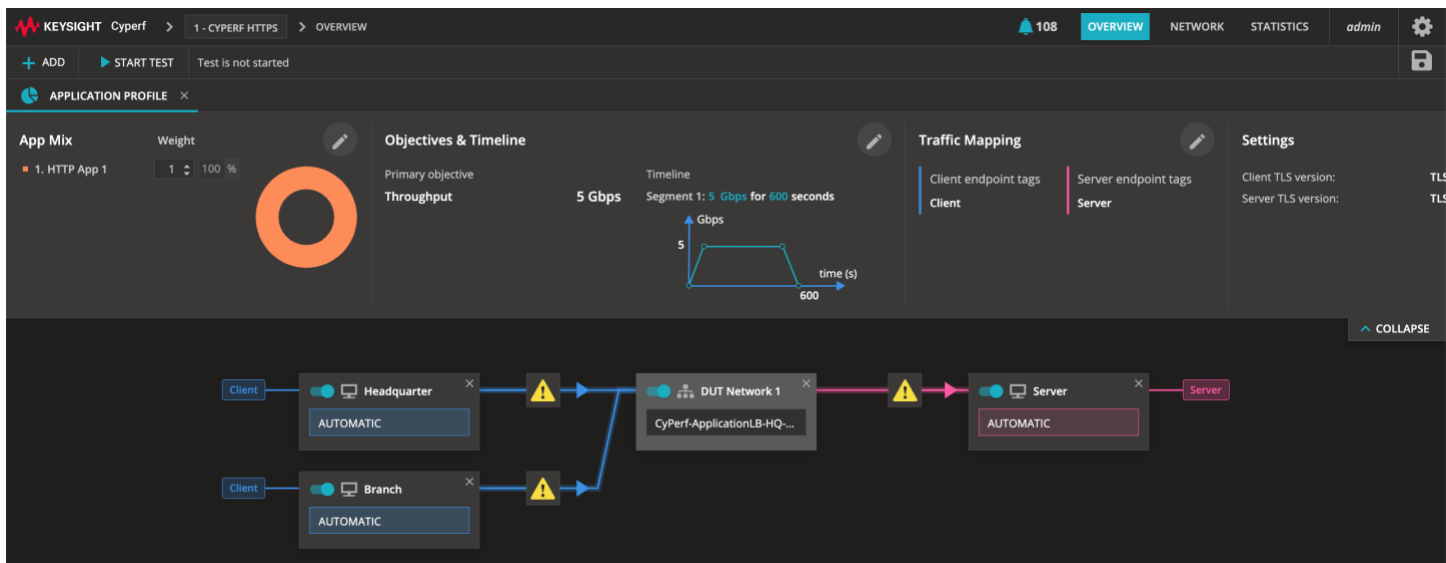
2. In the CyPerf Controller's landing page, please navigate to **Browse Configs** and then click on the **Import** button on the lower left corner.

Select the first configuration file (i.e., "CyPerf HTTPS.zip") from your local machine and complete the import step.

3. After the new config file is imported, please click on the "create session" icon, on the right side of the screen, on the same row with the new config:



4. The new config file will be loaded into a new CyPerf Session:

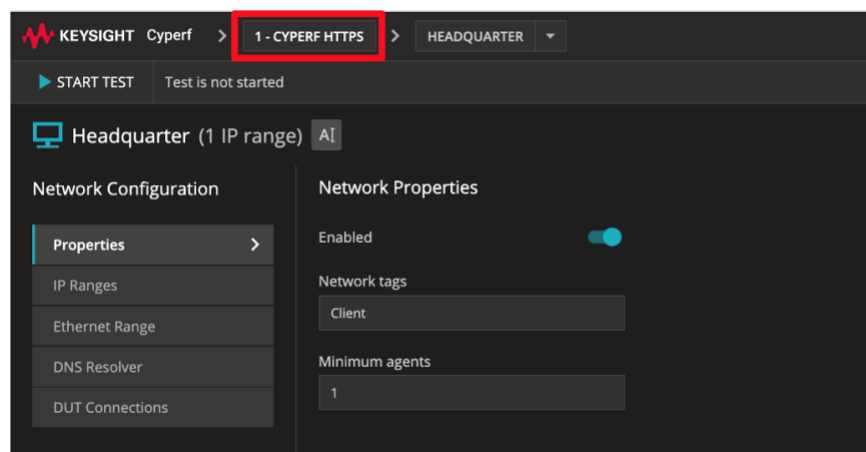


This is the CyPerf test overview page, where all the most important test parameters can be seen in a single pane of glass.

The bottom half of the page is an interactive representation of the logical topology used in this test. This topology reflects the setup that we described in the above “Lab Introduction” section. Upon clicking on any element in the topology, the menu with all the configurable parameters for the respective object will be showed.

For example, if you click on the **Headquarter** network segment you can see 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 will be used), DNS Resolvers etc.

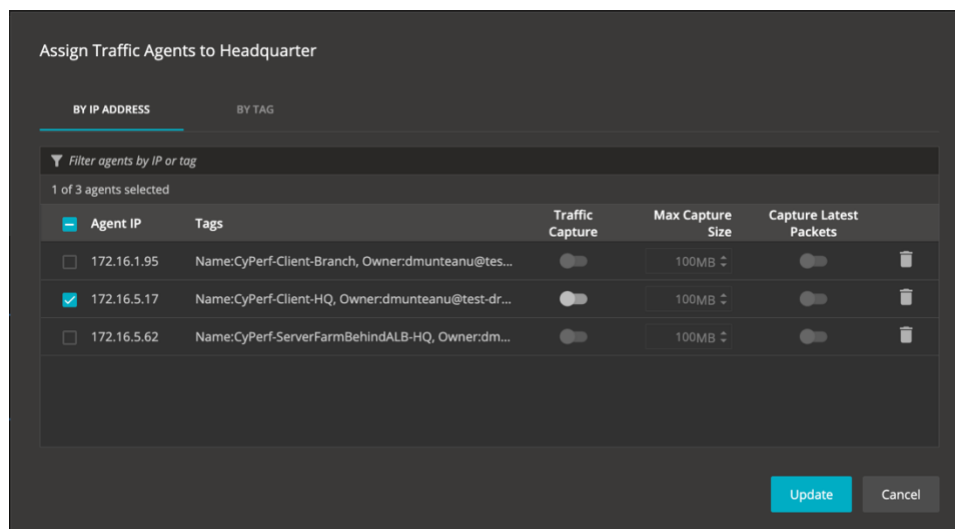
You can click the “back” browser button or click on the **CyPerf HTTPS** button in the breadcrumb to navigated back to the test overview page:



5. First, let's assign the already deployed CyPerf Agents (which got automatically deployed when Applying the Terraform script in the CloudShare webpage) to each of the Network Segments in our test configuration:

- a. Click on the “yellow attention” icon (⚠️) corresponding to the **Headquarter** network segment.

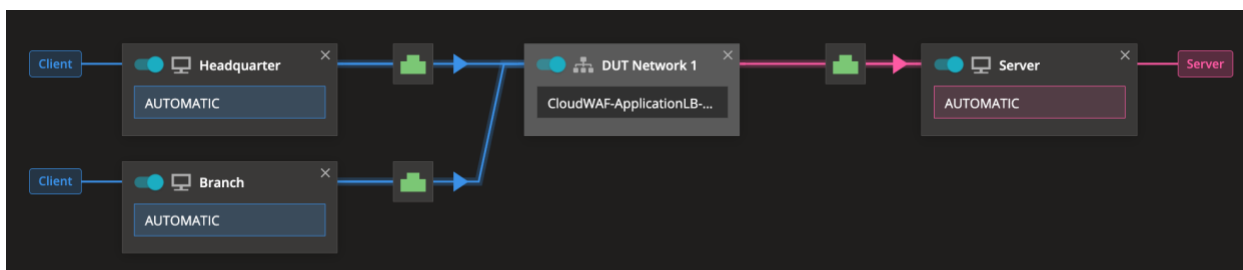
A new dialog will open with all the CyPerf traffic agents connected to your CyPerf Controller. For the **Headquarter** network segment, select the Agent that has the “*CyPerf-Client-HQ*” tag and click on the **Update** button:



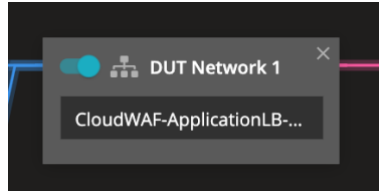
Once the new CyPerf traffic agent is assigned, the “yellow attention” icon (⚠️) will turn into a green port (🟢) which means that the **Headquarter** network segment has active CyPerf traffic agents assigned.

- b. Next, repeat the above procedure and assign to the **Branch** network segment another CyPerf traffic agent with the tag “*CyPerf-Client-Branch*”.
- c. Lastly, for the **Server** network segment, assigned the last CyPerf traffic agent with the tag “*CyPerf-ServerFarmBehindALB*”.

Now, that active CyPerf agents have been assigned, all three network segments should have a green port (🟢):

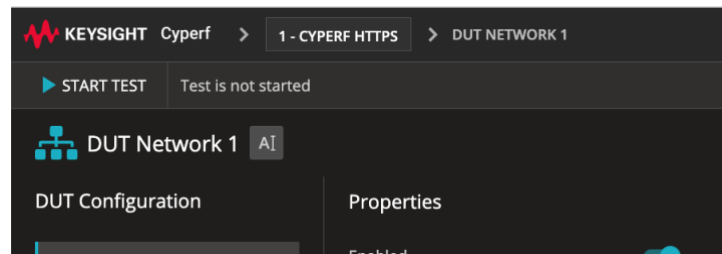


6. Next, let's configure the DUT address, which is basically the DNS Hostname that the DUT is listening for incoming requests, and which will be used by the CyPerf client traffic agents to send traffic to:
- a. Click on the **DUT Network** object in the middle of the topology diagram:



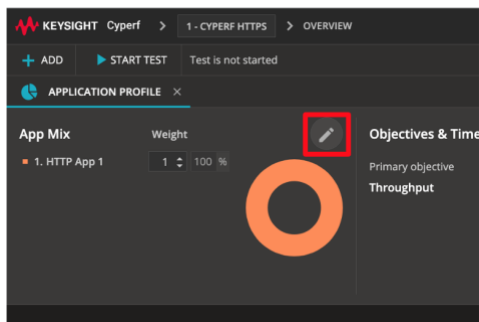
- b. In the new menu, paste the DUT public DNS that was obtained from the Public Cloud Logs (at the beginning of this lab) in the **Host** field.

7. Click on the **CyPerf HTTPS** button in the breadcrumb at the top of the page to go back to the test overview page:



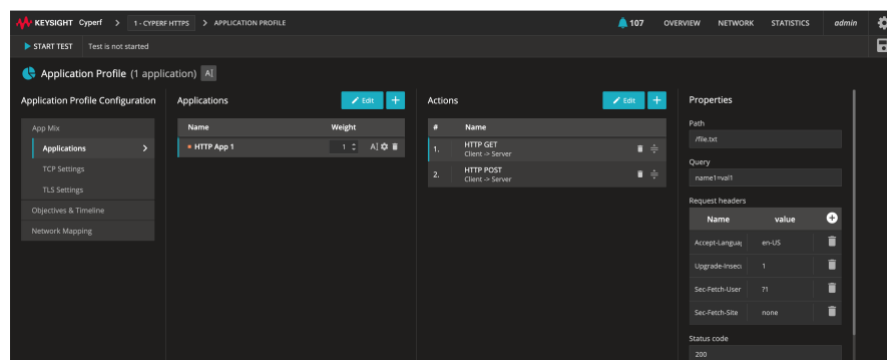
Although the test is now ready to run, before starting it, let's have a look at the most important test parameters.

8. **Application Profile:** Click on the pencil icon (  ) in the Application Profile section:



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

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



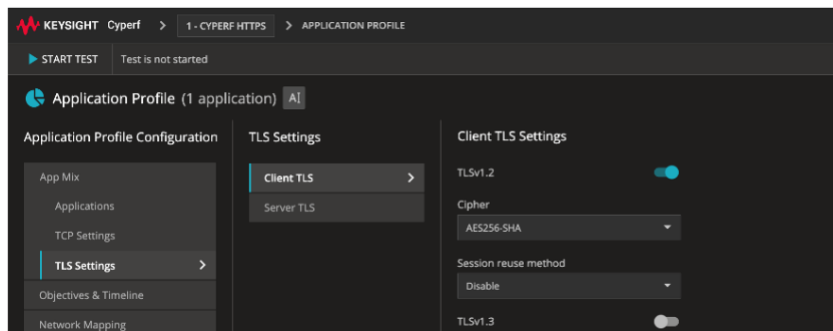
In this test we are using a basic HTTP profile for bi-directional traffic with the following commands:

- 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 users can create their own unique profiles. In the next lab we will go in more details into what other apps are available into the application library.

### b. TLS Settings:



In this test we are also using a basic yet common TLS profile with the following commands:

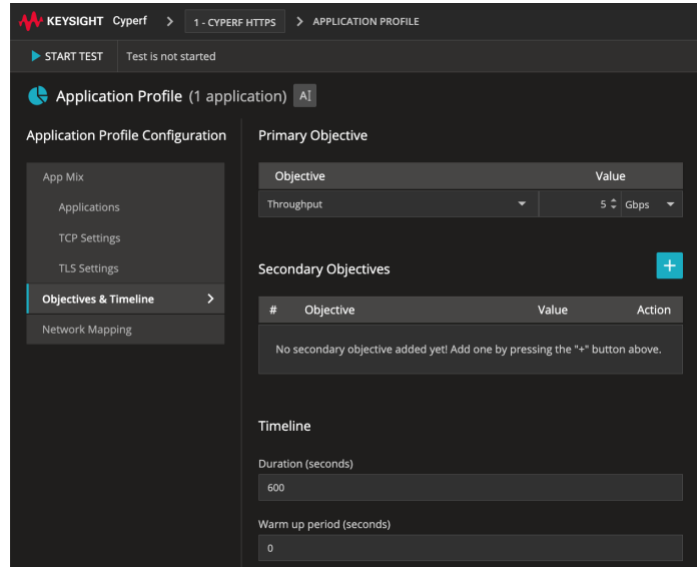
- TLS 1.2
- RSA 2K key certificate
- RSA-AES256-SHA cipher

## 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 etc) 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.

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



### IMPORTANT

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

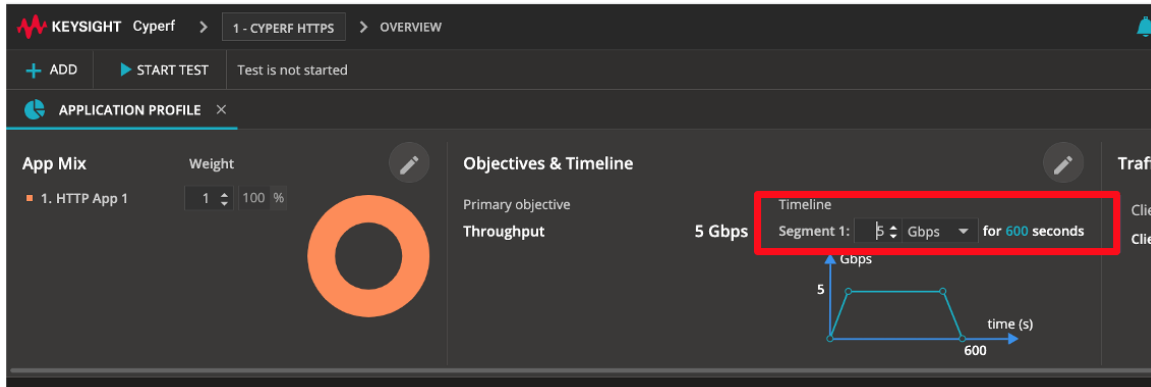
Now that the test has been updated with the details of the setup/agents assigned to your user, we can run the test and interpret the results.

### CAUTION

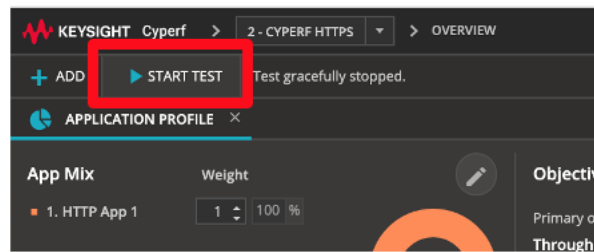
AWS Elastic Load Balancers (ELB) are able to address a broad set of use cases and traffic load profiles however, when there is a sudden increase of traffic it is recommended that ELBs would be pre-warmed. The general recommendation is that “*the traffic increases at no more than 50 percent over a five-minute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

For more information, please refer to: <https://aws.amazon.com/articles/best-practices-in-evaluating-elastic-load-balancing/#pre-warming>

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

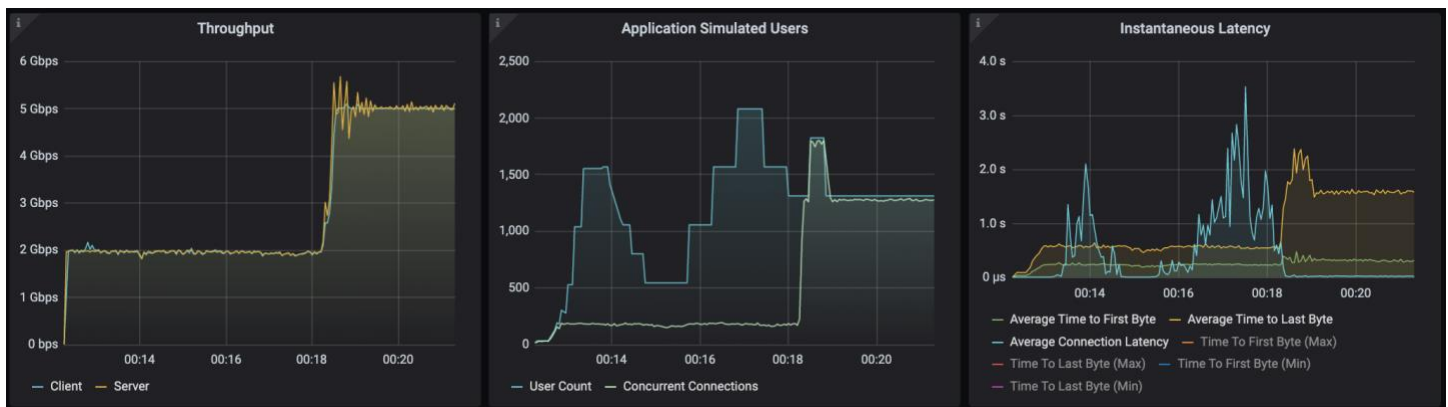


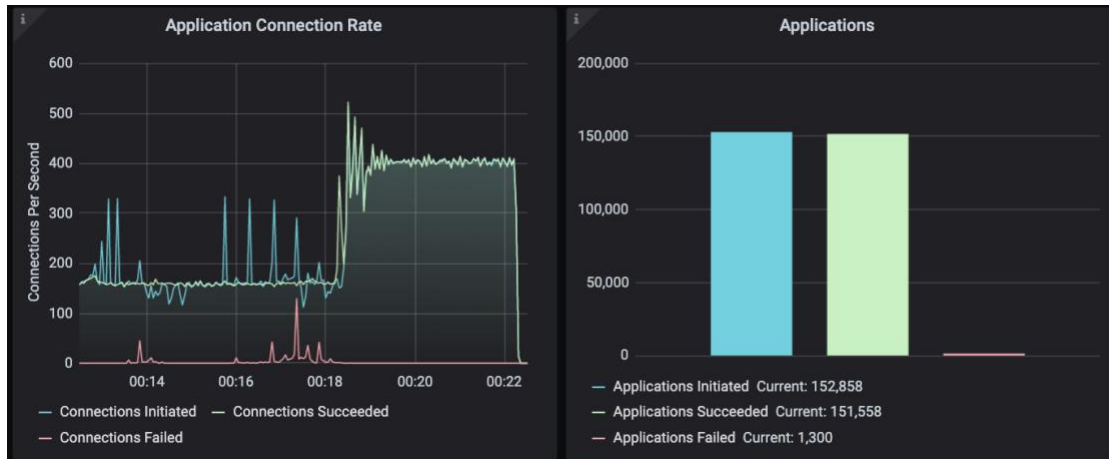
9. Click on the “Start Test” button in the upper left corner:



The test traffic agents will get configured and in a few moments the traffic will start. The view will automatically switch to the “STATISTICS” dashboard.

For this initial warm-up test, we will notice that the 5Gbps of throughput will be achieved after a few good minutes into the test (please note that the exact time required to reach to 5Gbps will vary). Also, depending on a number of factors, it is possible that the test will report a number of Application Failures, Connection Failures and high Latency stats as the ELB is warming up.





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

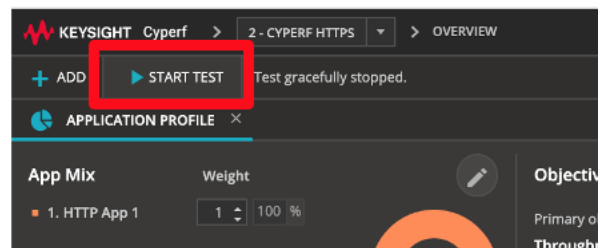
- Aside from the Application Failures, another obvious aspect is the huge spike in terms of the Connection Latency (the blue line of the Instantaneous Latency graph), especially in the first part of the test, when the configured objective of 5Gbps is not yet achieved.
- Another important metric is the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

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

## IMPORTANT

The reason why we keep the same 5Gbps as the traffic load is due to cost reasons (as the traffic is distributed and crossing through a number of network and security controls)

10. Click on the “Start Test” button in the upper left corner:



Again, the test traffic agents will get configured and in a few moments the traffic will start.



## Result Analysis

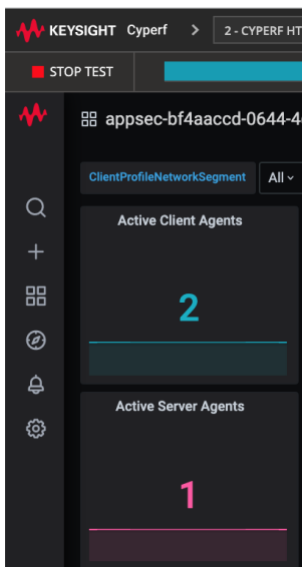
The maximum throughput performance test typically requires an iterative method in which the test is run multiple times, changing a number of test input parameters (which, due to time and cost constraints we would not be fully covering in this lab). The most important recommendation is that the DUT needs to 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 identify if the device has reached its saturation point and pin-point issues. Also interpreting the results in the correct manner will ensure that transient network, device or test tool behavior do not create a false negative condition, which is especially applicable for cloud environments.

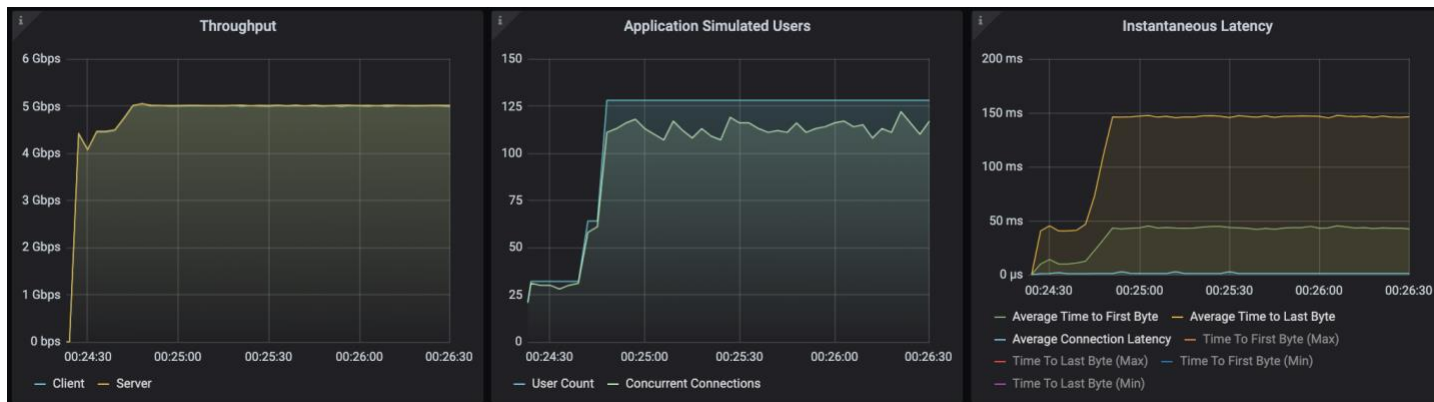
## Real Time Statistics

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

The first thing that can be noticed is that in the test there are two clients and one server that are being active and participating into the test, which is per our configuration and expectations.



Next, in terms of **Throughput** the statistic indicates that the setup is able to easily reach the 5Gbps we have configured:

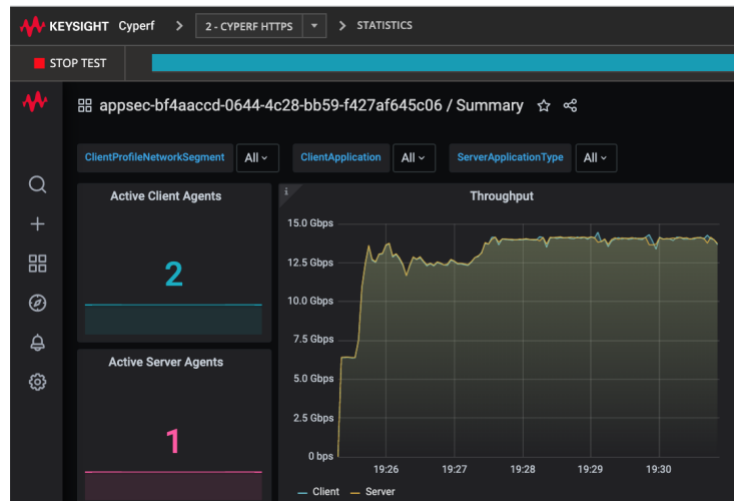


Also, per above graphs, we can see that the number of Concurrent Connections is closely following the number of Simulated Users which indicates that the infrastructure can easily cope with the generated traffic concurrency. Also, the Instantaneous Latency graphs are looking very good and stable values.

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

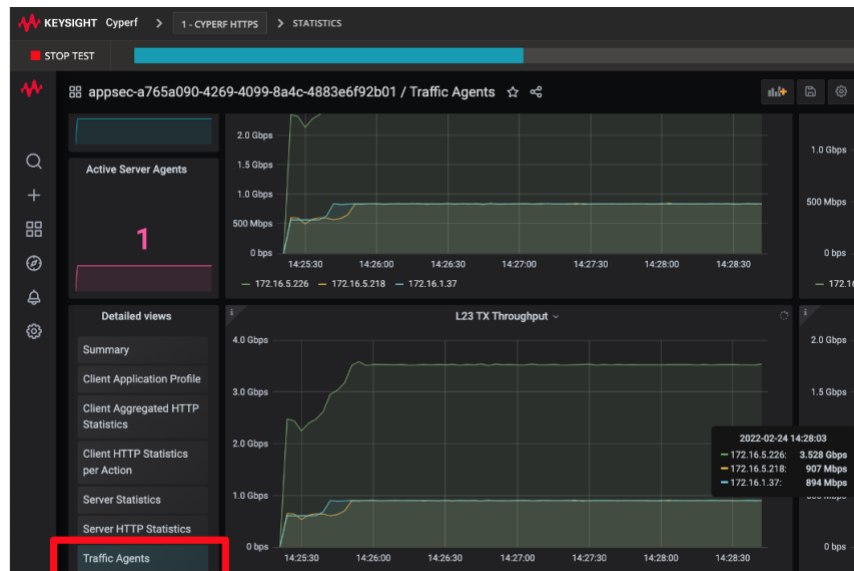
- Connection Latency: the average time it took for the TCP connection to establish
- Time to First Byte: the average time to receive the first data byte since the request was issued
- Time to Last Byte: the average time to receive the full data object/page since the request was issued

**Note:** Keep in mind that we restricted the load of this test to 5Gbps due to cost related reasons. If we would have properly warmed up the infrastructure and tried to push more traffic, the maximum performance obtained on this setup would have been around 14Gbps (below screenshot is after such a maximum performance test was ran):

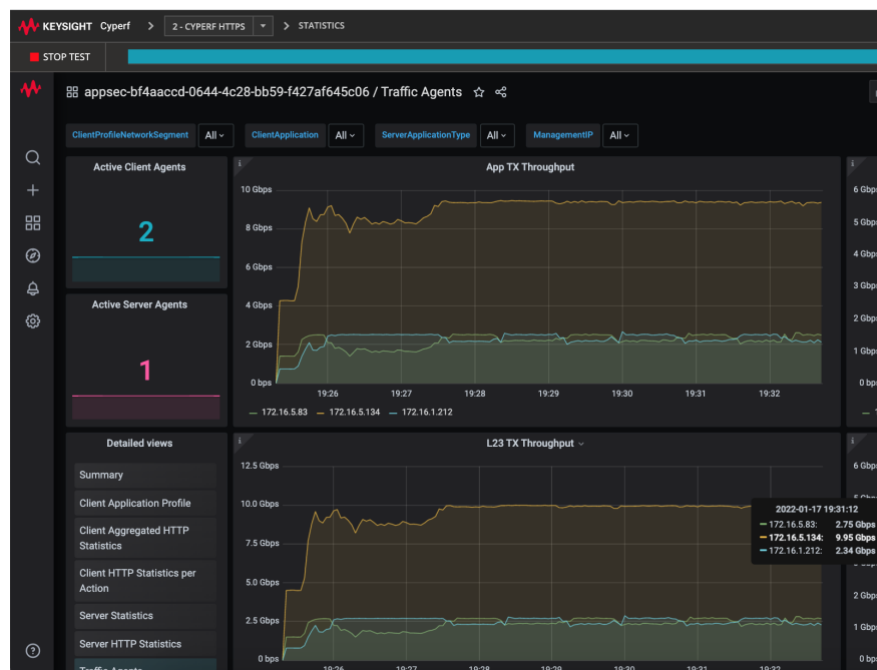


Next, for our current test, we can further look into more granular details and see how much traffic each agent is performing: if we look deeper and select from the **Detailed View** pane (in the lower left corner) the **Traffic Agents**

dashboard we see that most of the traffic is generated by the server (i.e. 3.528Gbps) while the clients are fairly balanced (generating 907Mbps and 894Mbps respectively):

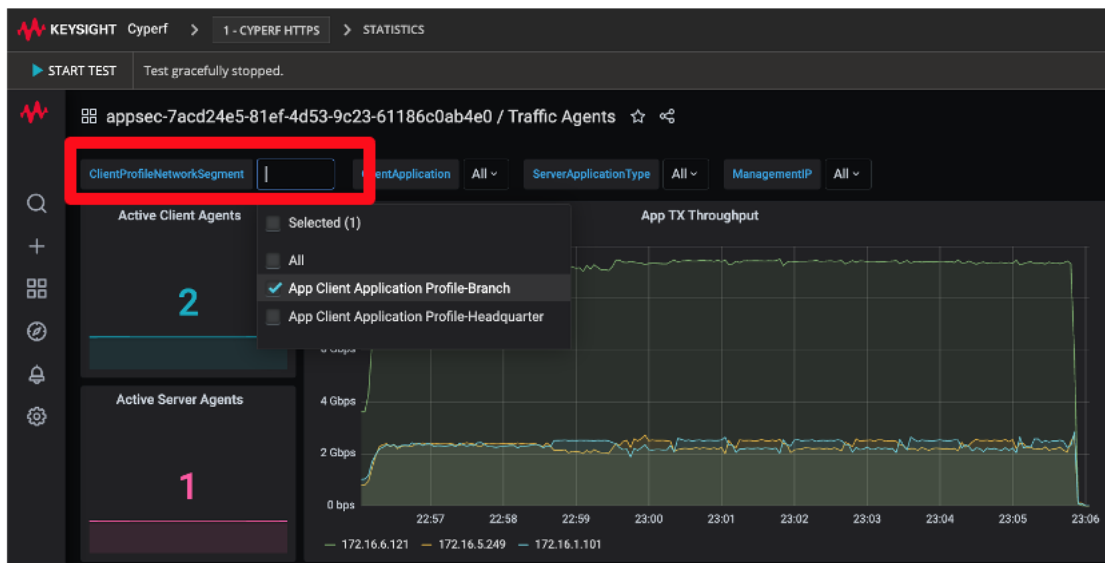


**Note:** when trying to push the maximum throughput the reason why we were getting up to 14Gbps (bi-directionally) was that the server agent could send traffic at up to 10Gbps (the Public Cloud provider is limiting the maximum bandwidth for this particular instance type to 10Gbps per direction). Also, the client agents could send traffic at 2.75Gbps and 2.34Gbps respectively for the max bandwidth test, as seen in below screenshot (taken for the maximum performance test):



A very useful functionality is the ability to apply filters for most of the stat views. For example, choosing only one network segment (in the upper left corner) would present only the stats for that respective

segment which would help 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 stats about each action configured for all the applications in the test:

Profile - Network Segment	Application & Attack	Action	Started	Succeeded	Failed
App Client Application Profile-Headquarter	HTTP App 1	HTTP POST	119,134	119,134	0
App Client Application Profile-Headquarter	HTTP App 1	HTTP GET	119,134	119,132	2
App Client Application Profile-Branch	HTTP App 1	HTTP POST	119,616	119,616	0
App Client Application Profile-Branch	HTTP App 1	HTTP GET	119,616	119,615	1

Profile - Network Segment	Application & Attack	Action	Payload Bytes Sent	Payload Bytes Received	Responses F
App Client Application Profile-Headquarter	HTTP App 1	HTTP POST	60,996,608,000	2,620,948	
App Client Application Profile-Headquarter	HTTP App 1	HTTP GET	0	121,991,168,000	
App Client Application Profile-Branch	HTTP App 1	HTTP POST	61,243,392,000	2,631,552	
App Client Application Profile-Branch	HTTP App 1	HTTP GET	0	122,485,760,000	

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 resets as well as TLS Handshake stats.

TLS Statistics (NetworkSegment: All) ^							
Application Profile - Network Segment	Handshake Started		Handshake Succeeded		Handshake Failed	Session Reuse Succeeded	Session Reuse Failed
App Client Application Profile-Headquarter	119,134		119,134		0	0	0
App Client Application Profile-Branch	119,616		119,616		0	0	0

## Conclusions

The purpose for this first test, was to baseline the maximum performance of the Device Under Test (DUT) using simple HTTPS traffic, which is what most vendors will 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 should be as close as possible to the production version.

## Lab 2: Realistic Enterprise traffic mix performance

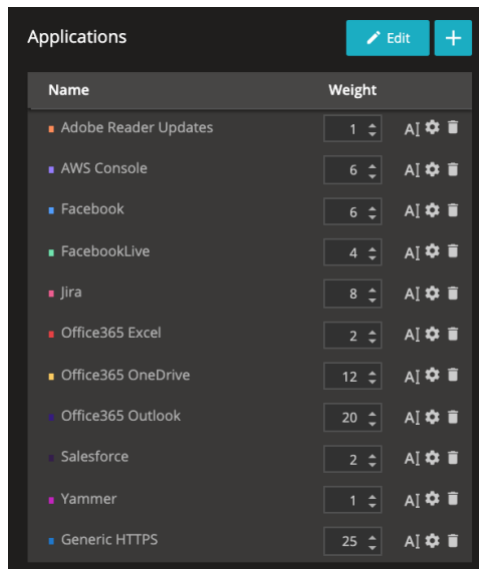
### Overview

For the second Lab, instead of generic HTTPS traffic, we will use a much more realistic traffic profile which resembles the typical applications traffic flows in an enterprise network.

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 perform as good as possible.

Datasheets are a good starting point but since each environment is unique, it is paramount to test with an application traffic profile that resembles as close as possible the production environment. This will ensure that the test results will be as relevant as possible so that informed decisions can be made.

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



Name	Weight	
Adobe Reader Updates	1	A [ ⚙️ 🗑️ ]
AWS Console	6	A [ ⚙️ 🗑️ ]
Facebook	6	A [ ⚙️ 🗑️ ]
FacebookLive	4	A [ ⚙️ 🗑️ ]
Jira	8	A [ ⚙️ 🗑️ ]
Office365 Excel	2	A [ ⚙️ 🗑️ ]
Office365 OneDrive	12	A [ ⚙️ 🗑️ ]
Office365 Outlook	20	A [ ⚙️ 🗑️ ]
Salesforce	2	A [ ⚙️ 🗑️ ]
Yammer	1	A [ ⚙️ 🗑️ ]
Generic HTTPS	25	A [ ⚙️ 🗑️ ]

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

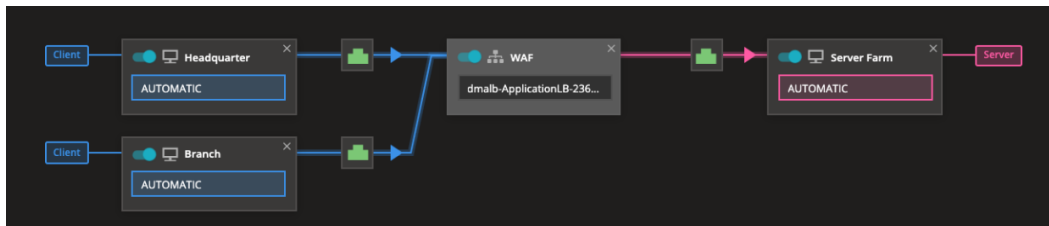
### Objective

Performance metrics to be characterized:

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

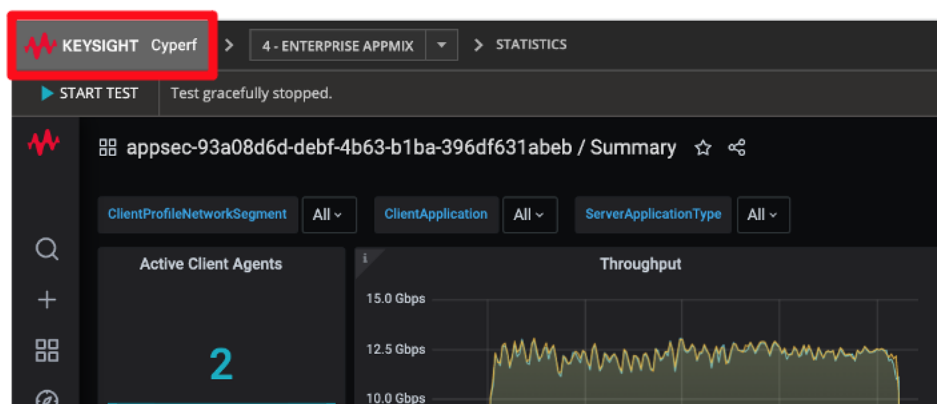
## Setup

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



## Step-by-Step Instructions

1. Once the previous test has been completed, navigate to CyPerf Controller's landing page by clicking on the "KEYSIGHT CyPerf" button in the upper left corner.



To import the second configuration file, please first download "Enterprise AppMix.zip" file locally on your PC from the following GitHub repo: <https://github.com/Keysight/cyperf/CyPerf/CyPerfTestDrive>

2. In the CyPerf Controller's landing page, please navigate to Browse Configs and then click on 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, please click on the "create session button", on the right side of the screen, on the same row with the new config.

KEYSIGHT Cyperf > BROWSE CONFIGS

114 admin

Config Categories

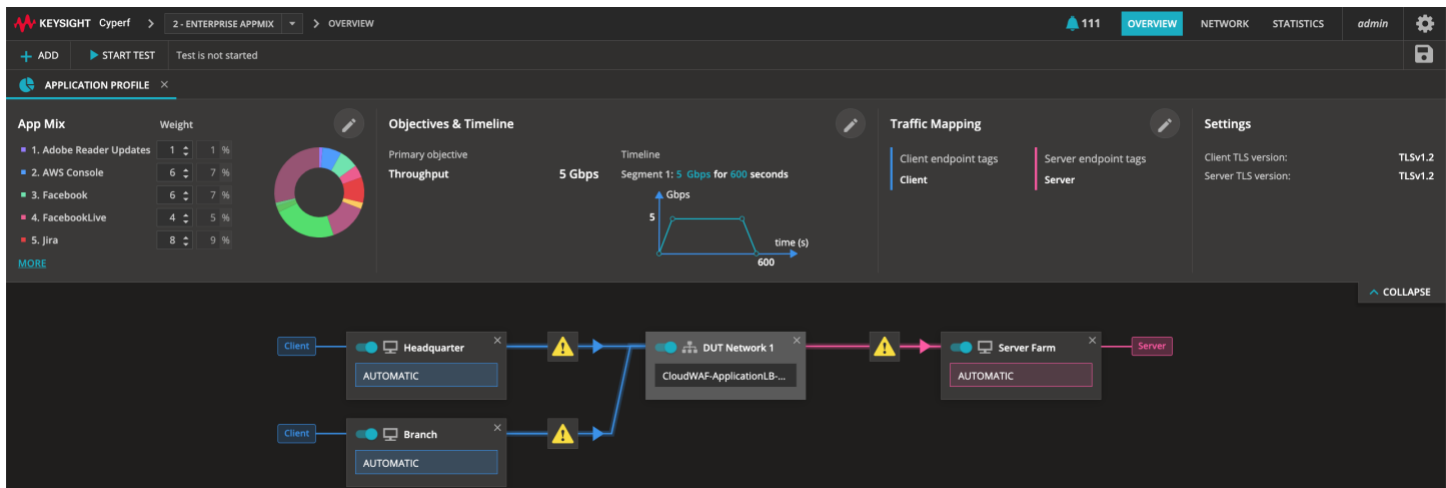
- Recent Configs
- CyPerf Configs
- CyPerf Cloud Configs
- CyPerf Features Configs
- CyPerf Performance Configs
- CyPerf Security Configs

4 configs

Config Name	Last accessed ↓	Application	Config Type	Owner	Create Session
Enterprise AppMix	Jan 17, 2022, 6:40:29 PM	CyPerf	CyPerf	admin@example...	
CyPerf HTTPS	Jan 17, 2022, 5:01:56 PM	CyPerf	CyPerf	admin@example...	
DUT Keep Alive HTTP1.1		CyPerf	CyPerf Features	system	
EnterpriseMix Throughput		CyPerf	CyPerf Performance	system	

Delete Export

4. The new config file will be loaded into a new CyPerf Session:



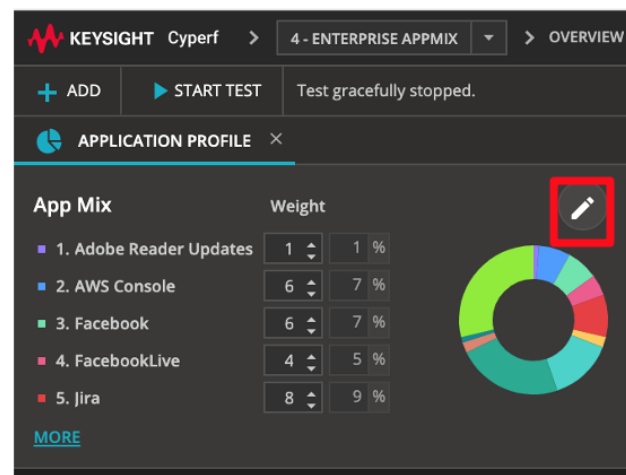
Before looking at the config in more details, we need to assign the CyPerf test traffic agents to the respective Network Segments, as we did in the previous labs:

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

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

- Now, in the CyPerf test overview page, we 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 rest of the parameters are basically the same.  
Before running the test, let's have a look at the new Application Profile.

- Click on the "pencil" right above the pie chart graph from the "Application Profile" section.





7. In the new menu, the entire AppMix composition with associated weights can be seen.

Clicking on any of the application will expose one level of detail deeper, showing all the actions comprising the respective application. For example, “Office365 Outlook” has the following list of granular actions.

The screenshot displays the 'Application Profile (11 applications)' configuration in the Keysight Cyperf interface. The left sidebar shows the 'Applications' tab selected under 'App Mix'. The main area is divided into two panels: 'Applications' and 'Actions'.

**Applications Panel:**

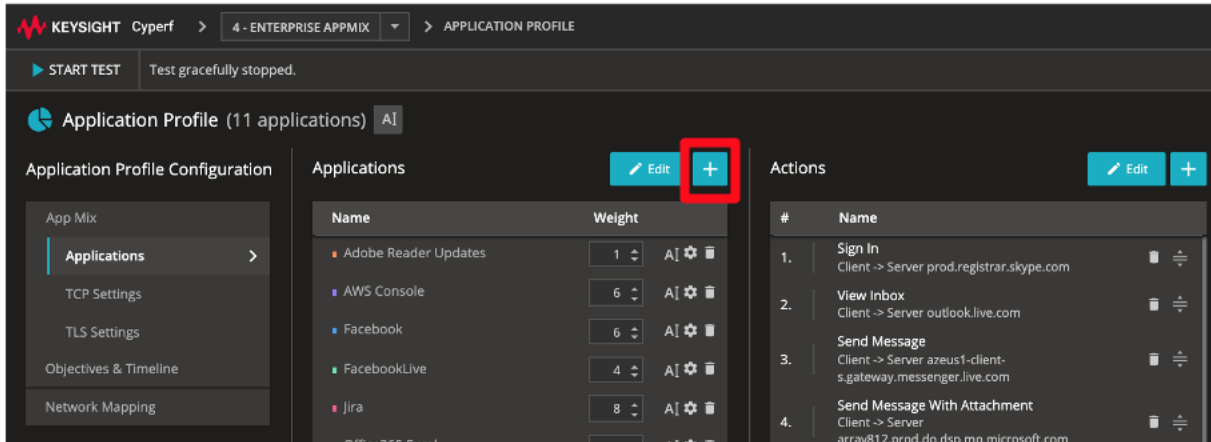
Name	Weight	AI	Settings	Remove
Adobe Reader Updates	1	A		
AWS Console	6	A		
Facebook	6	A		
FacebookLive	4	A		
Jira	8	A		
Office365 Excel	2	A		
Office365 OneDrive	12	A		
<b>Office365 Outlook</b>	<b>20</b>	<b>A</b>		
Salesforce	2	A		
Yammer	1	A		
Generic HTTPS	25	A		

**Actions Panel (for Office365 Outlook):**

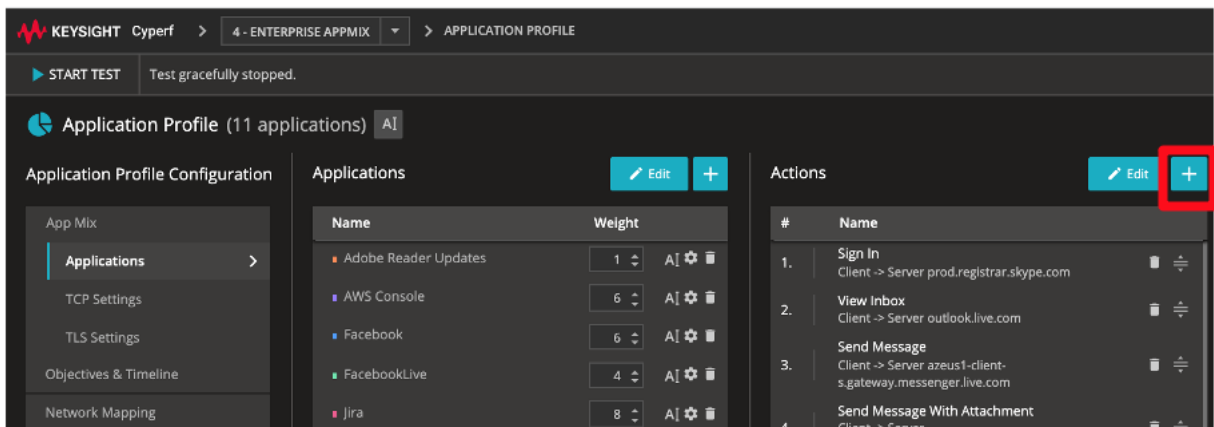
#	Name	Remove	Order
1.	Sign In Client -> Server prod.registrar.skype.com		
2.	View Inbox Client -> Server outlook.live.com		
3.	Send Message Client -> Server azeus1-client-s.gateway.messenger.live.com		
4.	Send Message With Attachment Client -> Server array812.prod.do.dsp.mp.microsoft.com		
5.	Open Message Client -> Server browser.events.data.microsoft.com		
6.	Delete Message Client -> Server outlook-1.cdn.office.net		
7.	Navigate To Calendar Panel Client -> Server api.asm.skype.com		
8.	Create A New Event Client -> Server ow1.res.office365.com		
9.	Delete An Event Client -> Server azeus1-client-s.gateway.messenger.live.com		
10.	Navigate To People Panel Client -> Server outlook.live.com		
11.	Create A New Contact Client -> Server outlook.live.com		
12.	Search For A Contact Client -> Server outlook.live.com		
13.	Delete A Contact Client -> Server outlook.live.com		
	Navigate To Task Panel		

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

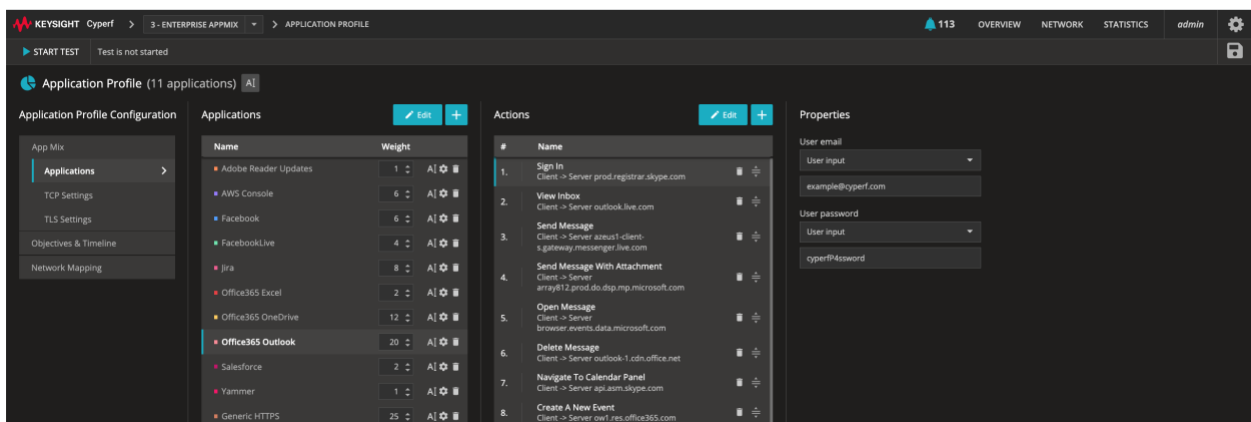
- To add a new application from the library just click on the add button (+ icon):



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



- Last but not the least, 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 & passwords that CyPerf can iterate through to simulate logging in from multiple users):



## IMPORTANT

The application mix that has been used above is just an example of a representative traffic mix that can be found in production environments.

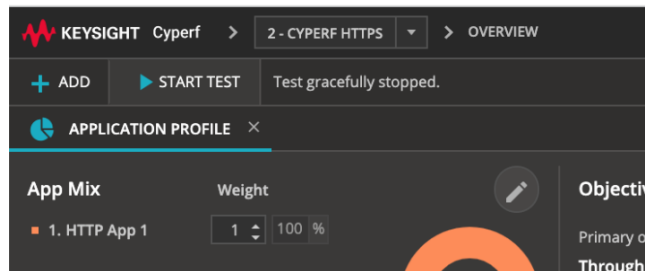
However, depending on the device under test (e.g. WAF, NGFW, IPS/IDS, SASE etc) and where it is placed in production networks, as well as what it is targeting to defend, the application mix can and should be tweaked to represent that particular environment. For example, for Web Application Firewalls users might want to 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 (e.g., TLS 1.2 and 1.3), encryption algorithms and SNI

8. In terms of Timeline and Objectives we will keep the same test load and duration as in the previous test due to the same reasons.

Now that the test has been updated with the details of the setup/agents assigned to your user, we can run the test and interpret the results.

9. Click on the “Start Test” button in the upper left corner:



The test traffic agents will get configured and in a few moments the traffic will start. The view will automatically switch to the “STATISTICS” dashboard.

**Note:** After starting the test, two harmless warning messages will pop-up notifying the users that Objective for the two network segments (i.e., 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 (i.e., 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.

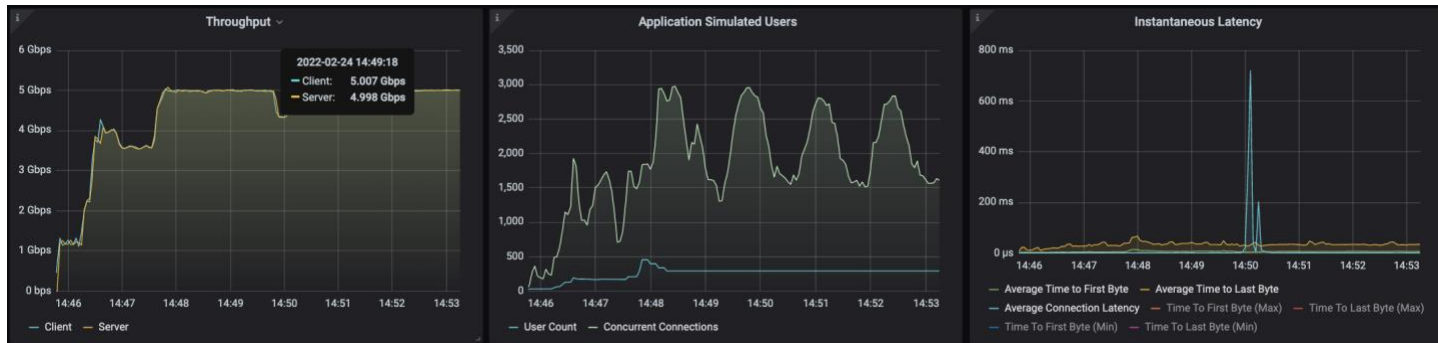
### Real-Time Statistics

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

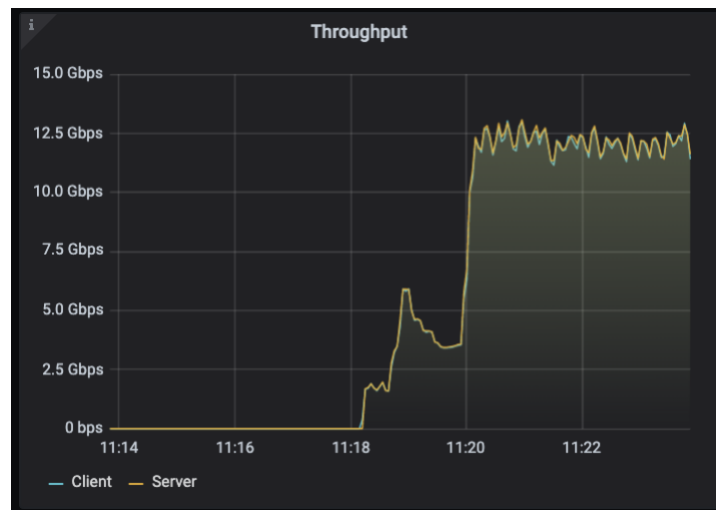
As the test is progressing, we will notice in the Summary dashboard that CyPerf is dynamically adapting the number of Simulated Users and generated Concurrent Connections to get to the best performance that the underlying setup will permit.

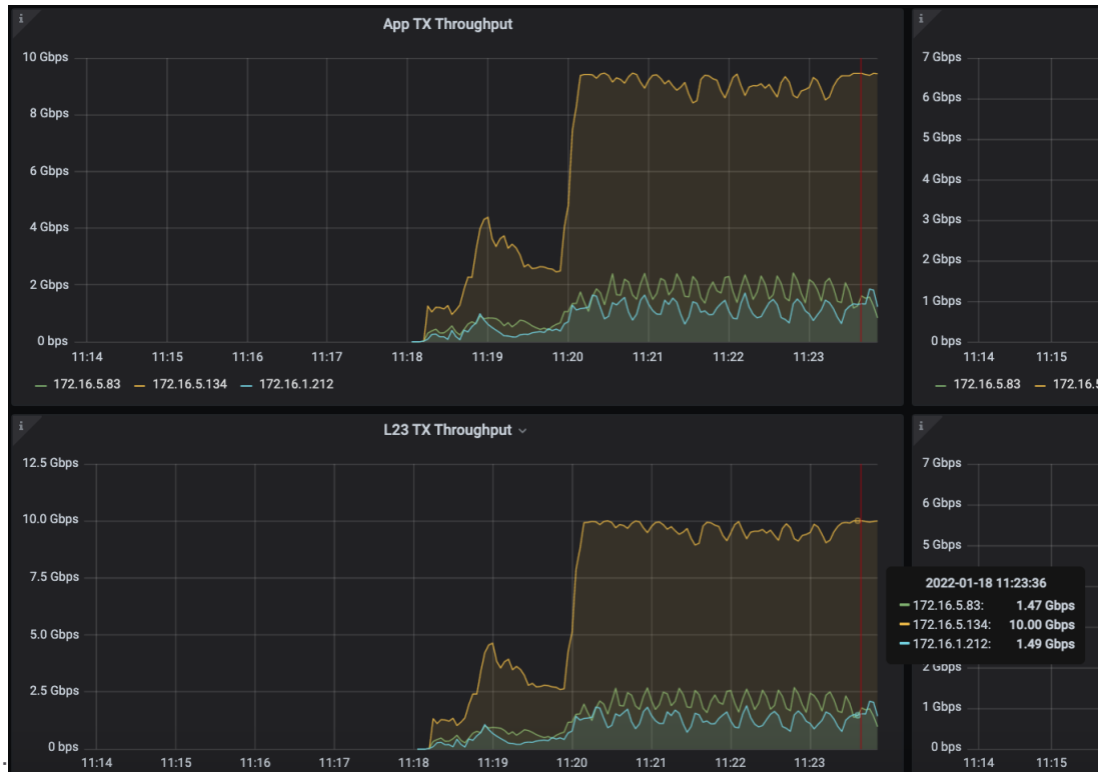
Note: CyPerf's unique ability to dynamically adapt and find the best performance that the underlying infrastructure can handle by tweaking the number of users and concurrent connections is a proprietary goal-seeking mechanism allowing users to find that maximum performance much faster than using other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

After a few minutes into the test (approx. 3-5 mins) we can see that the goal seeking mechanism stabilizes and the configured throughput is achieved (5Gbps):



**Note:** If we were to push the maximum throughput for this test as well, we would have obtained 12.5Gbps. Therefore, one important difference (for the max throughput test) that we would have seen 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 we would have noticed 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, similar to a real environment. Below are two screenshots taken from such a maximum AppMix throughput test:



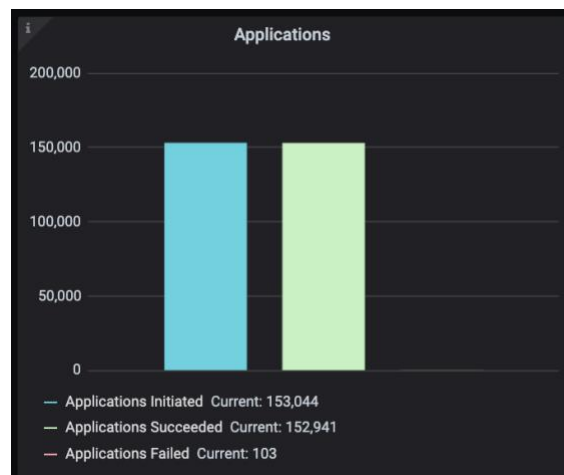


## IMPORTANT

In most test scenarios, a realistic application traffic mix is much more resource intensive on the DUT due to 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 below.

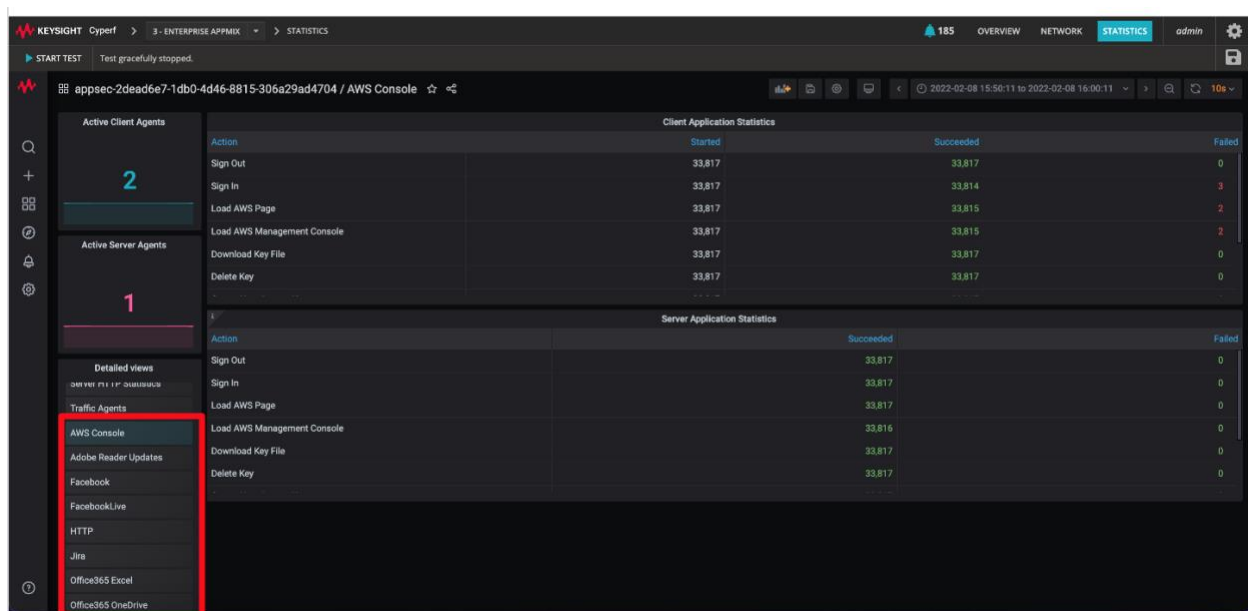
Total Application Initiated/Successful and Failures:



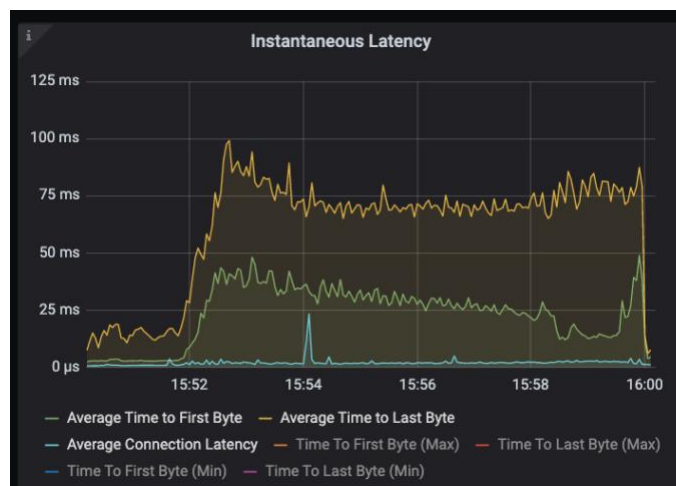
There are only 103 application failures out of over 150,000 initiated which, for most real-world requirements is an acceptable level. We can inspect further to understand which exact applications have observed failures (and even per location or Network Range) - we can see this in the **Client Application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

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

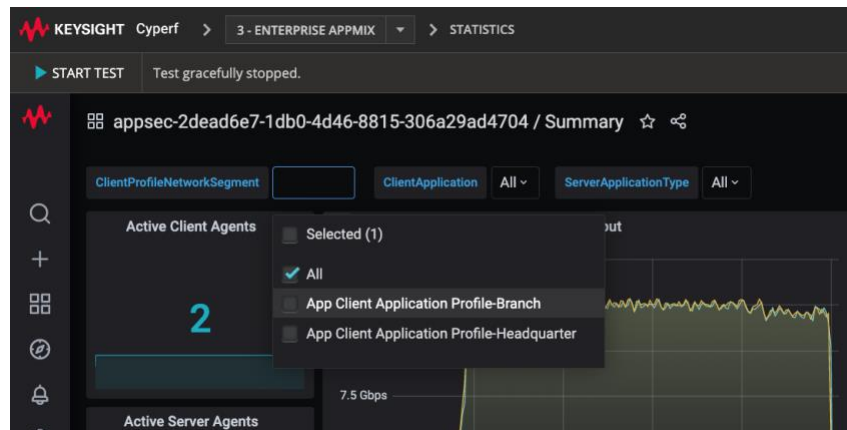
For an even more granular stats, each application in the mix has its own view where stats for each of the action comprising the respective application is shown:



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



As can be seen in the above graph the average latencies are fairly stable. Furthermore, we 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, we characterized the true performance of the Device Under Test (DUT) using a much more realistic application mix, which is not typically provided in the vendors' datasheets.

This is a very important test to run as it will uncover the true capabilities of various DUTs, and will help customers make informed, data-driven decisions.

## Lab 3: Realistic Security Efficacy

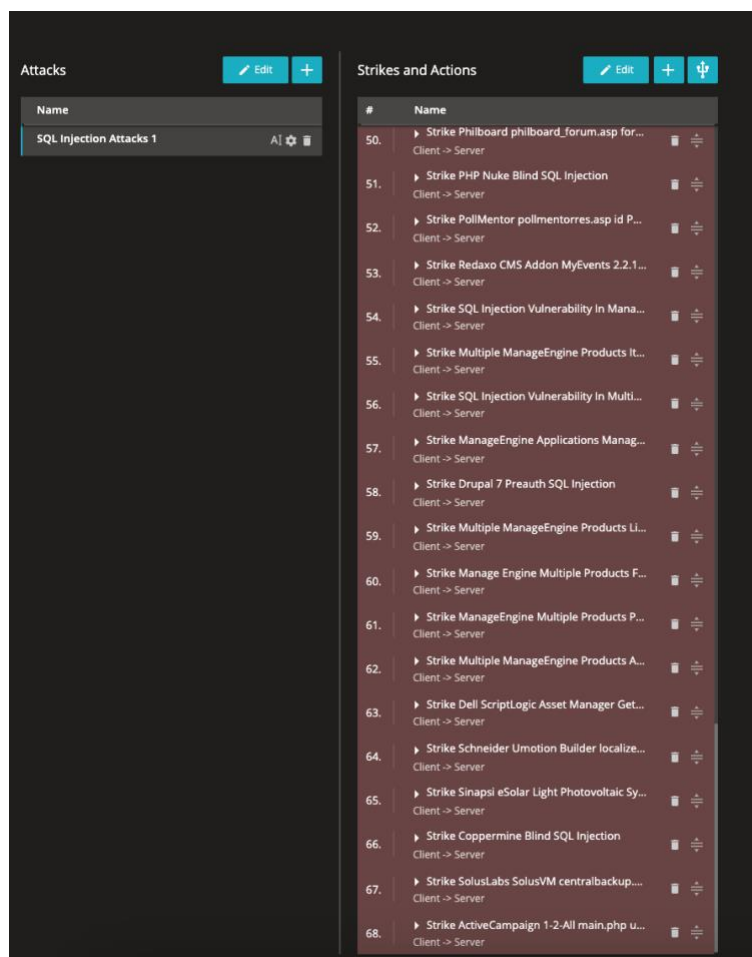
### Overview

In this Lab, we will go one step further and will evaluate the security efficacy of the Device Under Test (DUT).

First, we will keep the previously used realistic traffic profile, so that all security attacks are happening while legitimate traffic is present as well. This is similar to 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, we will be added a security profile that will consists of 68 selected SQLi attacks.



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



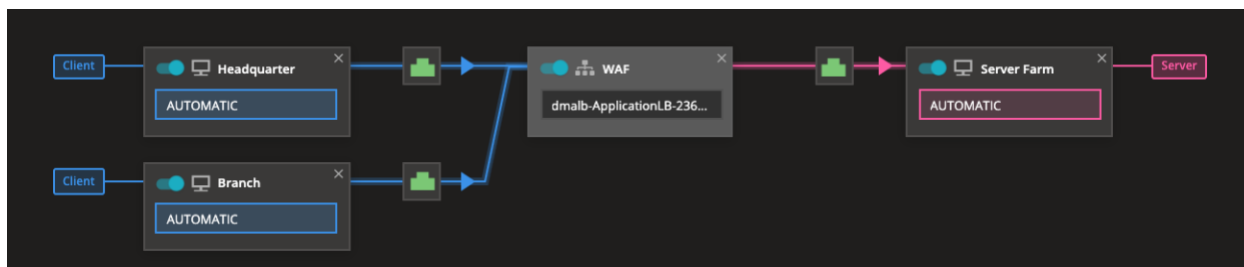
## Objective

Performance metrics required:

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

## Setup

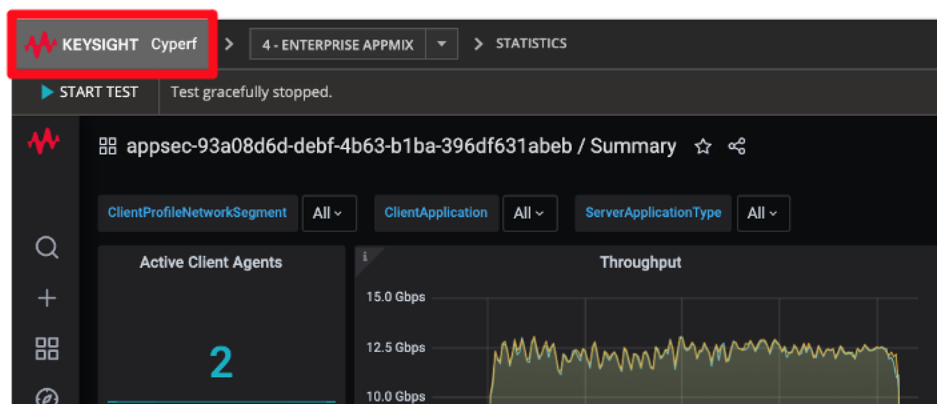
The setup for this lab will be the same as the one described in the Introduction section:



The DUT will be 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 we are characterizing the security efficacy of the default SQLi security rules (all generated security attacks will be SQLi attacks).

## Step-by-Step Instructions

1. Once the previous test has been completed, navigate to CyPerf Controller's landing page by clicking on the "KEYSIGHT CyPerf" button in the upper left corner.

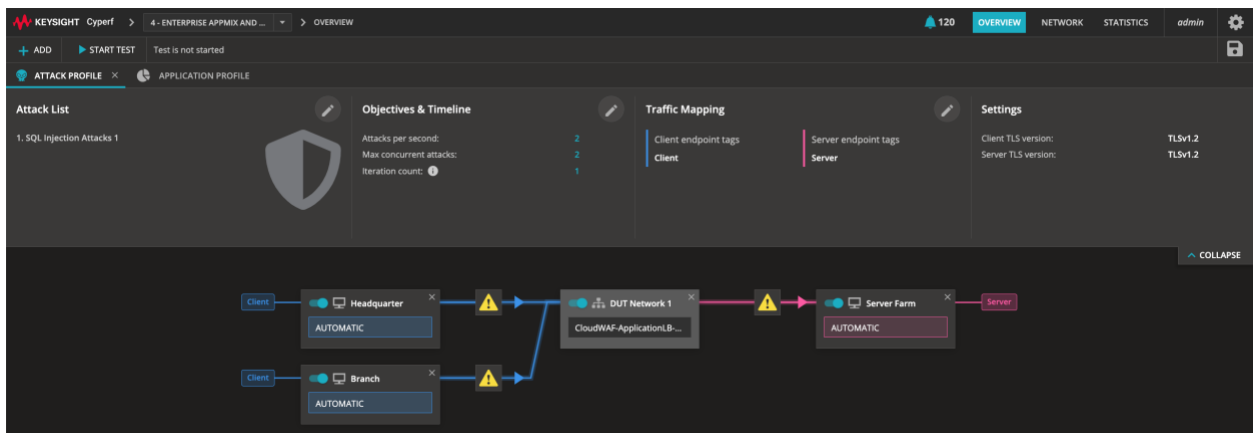


To import the third configuration file, please first download "Enterprise AppMix and Attacks.zip" file locally on your PC from the following GitHub repo: <https://github.com/Keysight/cyperf/CyPerf/CyPerfTestDrive>

2. In the CyPerf Controller's landing page, please navigate to Browse Configs and then click on the import button on the lower left corner and select the just downloaded "Enterprise AppMix and Attacks.zip" file.
3. After the new config file is imported, please click on the "create session button", on the right side of the screen, on the same row with the new config.



4. The new config file will be loaded into a new CyPerf Session:

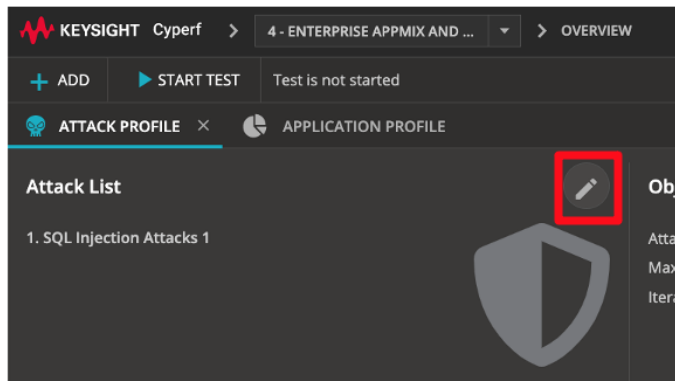


Before looking at the config in more details, we need to assign the CyPerf test traffic agents to the respective Network Segments, as we did in the first lab.

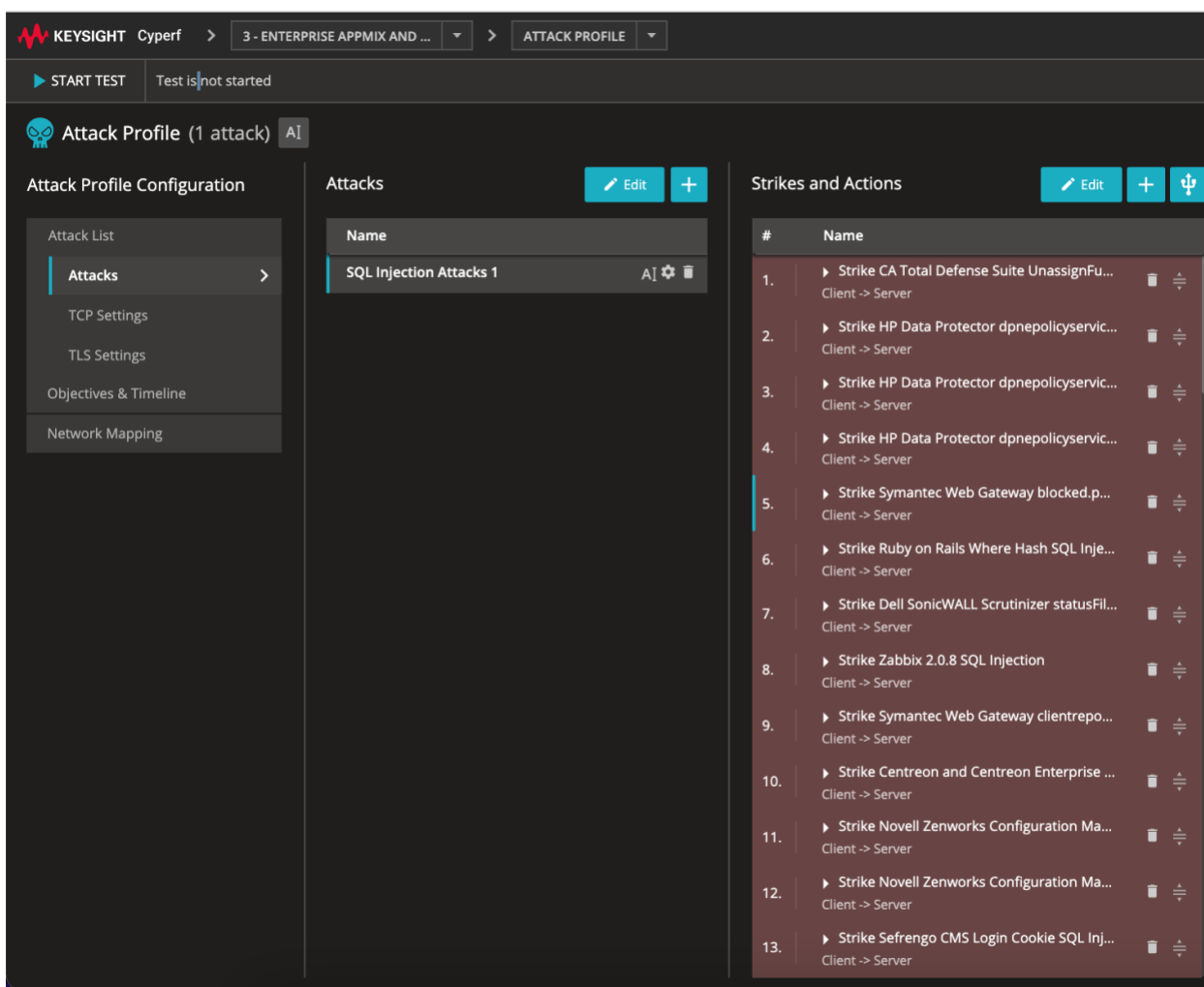
Please repeat step #5, from the first lab and make sure that the port icon turns green for all the three Network Segments.

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

5. Now, in the CyPerf test overview page, we can see that along with the Application Profile there is a second profile – Attack Profile.  
Before running the test, let's have a look at the new Attack Profile
6. Click on the "pencil" right above the shield graph from the "Attack Profile" section.

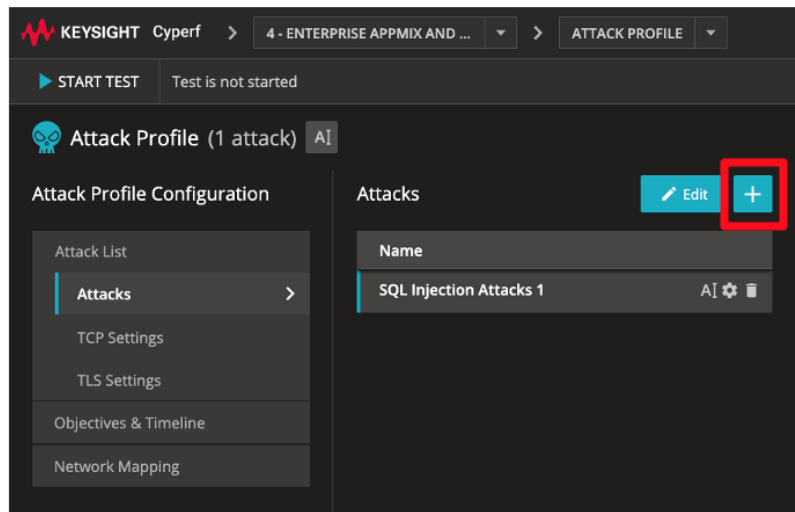


7. In the new menu, the Attack list can be seen which has one attack with 68 individual SQLi Strikes. Clicking on the “SQL Injection Attack 1” will expose all the 68 individual SQLi Strikes:

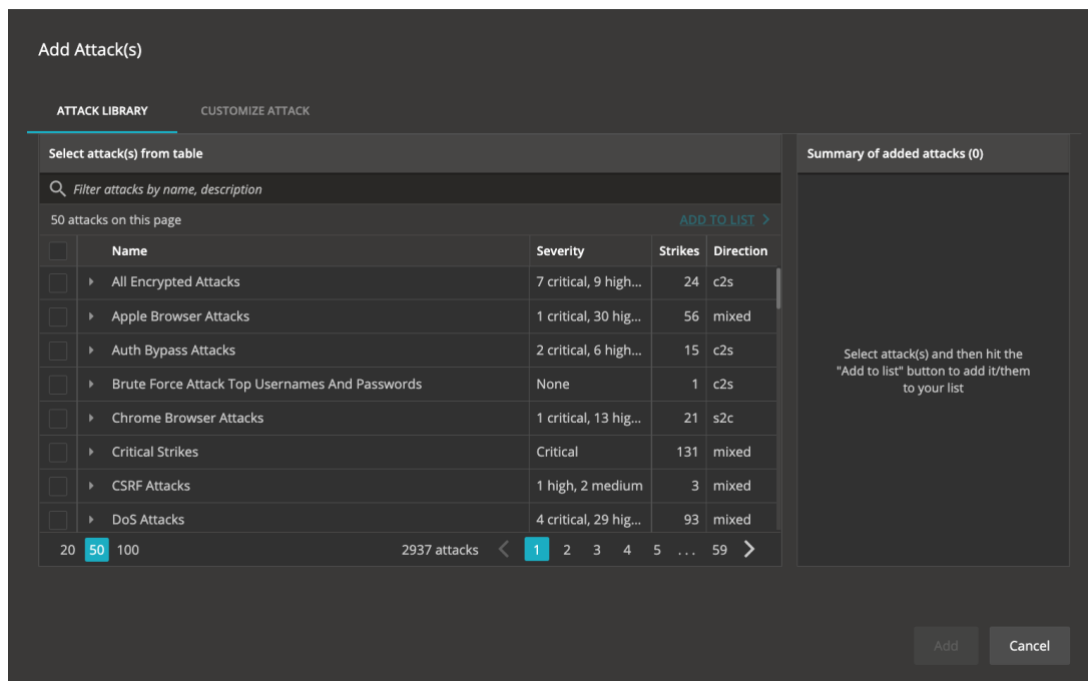


Tip: CyPerf offers users the ability to add into the test those attacks that are more relevant for the type of DUT they are testing and obviously the production environment as well:

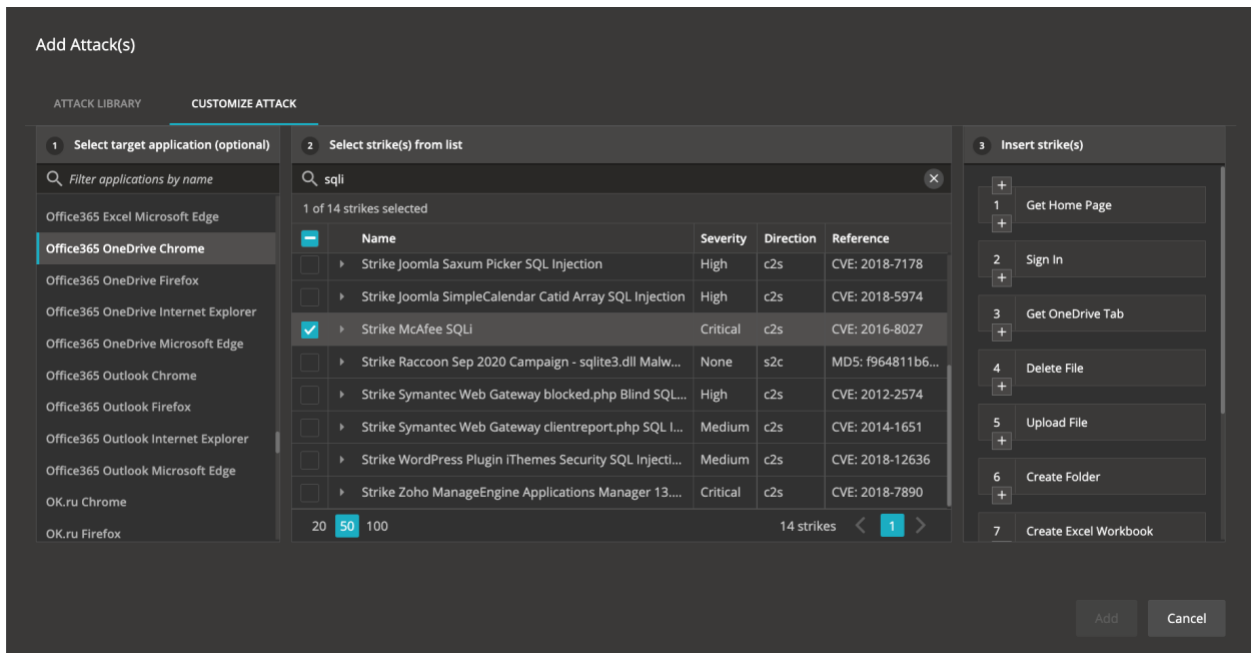
- To add a new attack from the library just click on the add button (+ icon):



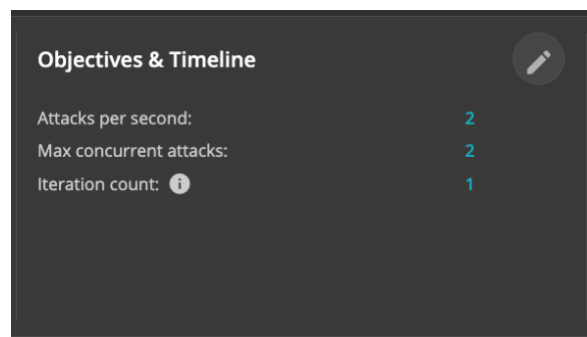
- The attack library comprises a comprehensive list of attacks containing multiple strikes, grouped based on certain categories, as well as individual strikes



Tip: Users 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 desired application actions to an attack or through the Customize Attack pane in the Add Attack dialog:



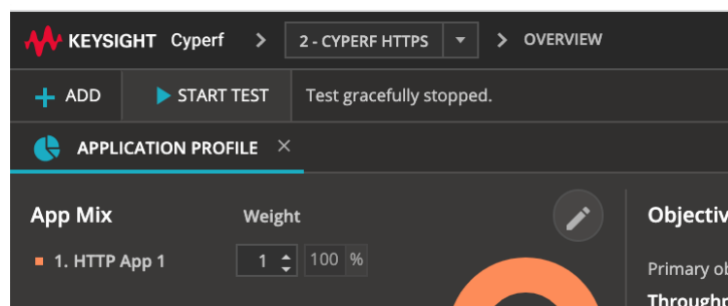
8. The Objective and Timeline of the Attack Profile is configured to 2 attacks per second and 2 maximum concurrent attacks and iterating only once through the attack list:



All the other test parameters are kept the same as in the previous lab.

Now that the test has been updated with the details of the setup/agents assigned to your user, we can run the test and interpret the results.

9. Click on the “Start Test” button in the upper left corner:



The test traffic agents will get configured and in a few moments the traffic will start. The view will automatically switch to the “STATISTICS” dashboard.

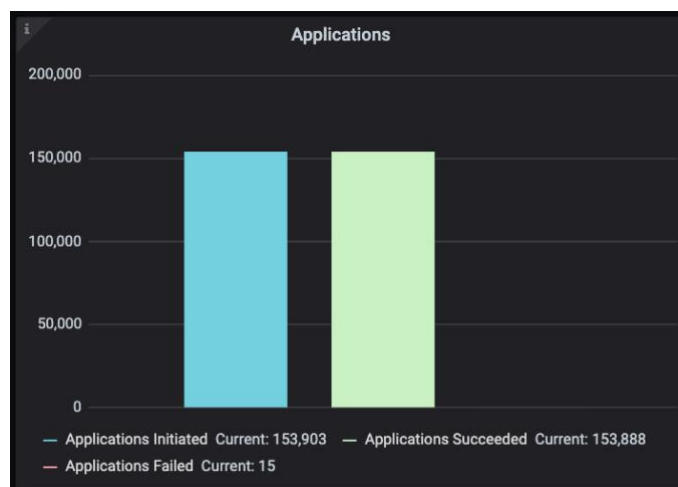
### Real-Time Statistics

First, in terms of application traffic we see a similar behavior to the previous test and after a few minutes into the test (approx. 3-5 mins) we 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 execute one iteration of the attack list and the duration of executing the attacks is unknown as in many cases that will depend on how the device under test will respond to the attacks. Another configuration option is to set the **Iteration Count** parameter under the **Attack Profile** to 0 which will then give users the ability to set a fixed duration for which the test will iterate through the attack list.

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



Therefore, on the legitimate traffic front we can conclude that the behavior is similar to 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 Network Security devices would lead to an impact to the maximum performance 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, let's see what is the security efficacy of the default SQLi security rules relative to the SQLi attack samples we selected. From the Summary dashboard, we see that out of the total 136 strikes (134 client to server and 2 server to client), there are 38 which have been allowed:

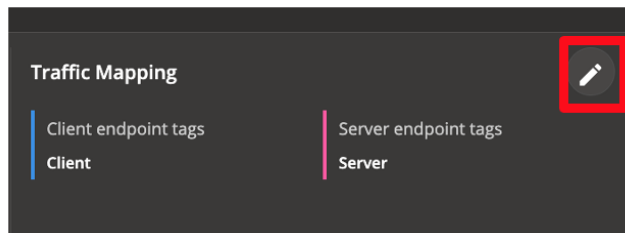


The reason why there have been a total of 136 strikes executed is that the same strike list (of 68 strikes) was executed from both client locations (or Network Segments).

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

## IMPORTANT

Using the advanced traffic mapping functionality (accessible via the pencil icon "✎" from the **Traffic Mapping** section in the test overview page), users can configure the security profile to be executed only from one location (e.g., 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, users can also check what exact security strike have been 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:

KEYSIGHT

Cyberf

3 - ENTERPRISE APPMIX AND ...

STATISTICS

142

OVERVIEW

NETWORK

STATISTICS

admin

START TEST

Test gracefully stopped.

appsec-59af8331-cefe-47c6-a9ea-796f7317b996 / SQL Injection Attacks

2022-01-19 13:10:51 to 2022-01-19 13:20:51

10s

2

Active Server Agents

1

Detailed views

aws-logs

AWS Console

Adobe Reader Updates

Facebook

FacebookLive

HTTP

Jira

Office365 Excel

Office365 OneDrive

Office365 Outlook

SQL Injection Attacks

Strike Dell SonicWALL Scrutinizer statusFilter.php SQL Injection	CVE: 2012-2962	Medium	http	2	2	0	2
Strike Drupal 7 Preauth SQL Injection	CVE: 2014-3704	Critical	http	2	2	0	2
Strike HP Data Protector donepollservice Component FinishedCo	CVE: 2011-3162	Critical	http	2	2	0	2
Strike HP Data Protector donepollservice Component GetPolicies	CVE: 2011-3157	Critical	http	2	2	0	2
Strike HP Data Protector donepollservice Component RequestCo	CVE: 2011-3158	Critical	http	2	2	0	2

S2C Attack Statistics

Action	Reference	Severity	Protocol	Server Initiated	Client Allowed	Client Blocked
Strike Manage Engine Multiple Products FailOverHelperServlet Bin	CVE: 2014-7864	High	http	0	0	2

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

Conclusions

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

In the rush to move to cloud networks, organizations are finding difficulty 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, natting, or going through internet gateways to another cloud location (multi-cloud topologies) or on-premises (hybrid topologies).