ExtraHop and AppDynamics Deployment Guide

This guide describes how to use ExtraHop and AppDynamics to provide real-time, per-user transaction tracing across the entire application delivery chain. ExtraHop provides wire data analytics for IT and business operational intelligence, and AppDynamics is the leader in agent data analytics. By integrating the two technology data sets, enterprises can view wire data and agent data together, maintaining consistent per-user transaction information.

This solution uses the ExtraHop Open Data Stream (ODS) capability to send real-time events and metrics to the AppDynamics REST API. While this guide focuses on real-time, per-user transaction tracing, there are many other ways to combine wire data and agent analytics not covered in this guide.

Contents

Software Prerequisites

ExtraHop:

- Version 4.1 GA or later including a license for ODS for HTTP
- AppDynamics Cookie Analysis bundle from the ExtraHop documentation site at docs.extrahop.com

AppDynamics:

- A machine agent to receive HTTP REST input
- An End User Experience license for the AppDynamics EUE system to insert and instrument the ADRUM_BT cookie
- An installed App Dynamics Machine Agent to receive HTTP messages and forward them to the AppDynamics Controller
- AppDynamics version 3.9 or later

Network Prerequisites

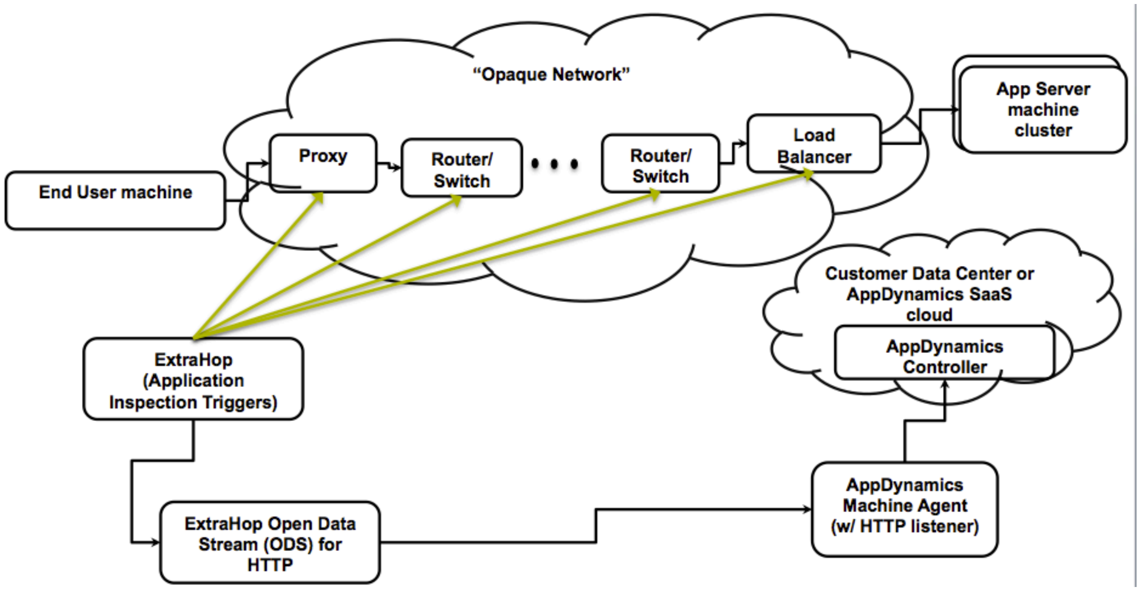
ExtraHop:

- Routing and firewall rules to allow ExtraHop devices to connect to the AppDynamics Machine Agent HTTP listener. In our example, this is TCP port 8293 connectivity (and is configurable on both ExtraHop and AppDynamics to be any unique, available TCP port (not UDP)).
- A topology that can be defined through the use of some uniquely identifiable characteristic. In order for ExtraHop to uniquely trace network segments, the ExtraHop Application Inspection (AI) triggers must be able to identify these segments. In our example, we define a firewall segment by VLAN, with VLAN 1129 being the Internet side of the connections, VLAN 1321 and 1421 being either side of the firewall, and VLAN 1121 being either side of the load balancer. These "hops" are customizable to each environment.

Refer to AppDynamics documentation for AppDynamics requirements.

Overview

The integration has three components: the ExtraHop platform, the AppDynamics controller, and the AppDynamics Machine Agent. This section provides an overview of the integration before explaining the step-by-step details.


"Opaque Network"
End User machine
Proxy
Router/ Switch
• • •
Router/ Switch
Load Balancer
App Server machine cluster
Customer Data Center or AppDynamics SaaS cloud
AppDynamics Controller
ExtraHop (Application Inspection Triggers)
ExtraHop Open Data Stream (ODS) for HTTP
AppDynamics Machine Agent (w/ HTTP listener)

This solution uses ExtraHop triggers, user-defined JavaScript code that you can write to extract custom metrics. In this case, triggers deliver the metrics that agents cannot—the "opaque network" of connectivity systems such as proxies, firewalls, routers and load balancers. Such metrics include timing deltas between hops and network anomalies. Once the metrics are extracted, they are sent to the REST interface of the AppDynamics Machine Agent. The Machine Agent then delivers the information to the AppDynamics Controller.

Using the examples in this guide, you will be capturing user traffic for one complete application environment. If you have multiple application environments, you will set up additional triggers, agents, and proxies for each additional environment.

This guide outlines four configuration tasks:

1) Defining the topology of the application environment in the trigger, giving the ExtraHop platform visibility into the flow of AppDynamics EUE cookies through the network. For this task, you will be implementing the topology by modifying the ExtraHop trigger,
2) Defining the type of metrics and events sent to AppDynamics. For this task, you will be implementing the metrics you are interested in by modifying the ExtraHop Trigger as well,
3) Configuring the ExtraHop Open Data Stream (ODS) for HTTP to send the HTTP output to the AppDynamics Machine Agent. For this task you will use the administrative interface on the ExtraHop to configure an ODS destination.
4) Configuring the AppDynamics Machine Agent to receive HTTP input. For this task, you will be modifying the AppDynamics Machine Agent configuration file.

Detailed Instructions

As you follow these steps, have the manuals ready for reference. The instructions in this guide are not a complete substitute for the entire product configuration. Instead, these instructions provide the critical pieces of information to complete the procedures.

For ExtraHop documentation, click the **Help** icon in the ExtraHop Web UI.

For AppDynamics documentation, click the **?** (Question Mark) icon in the AppDynamics user interface.

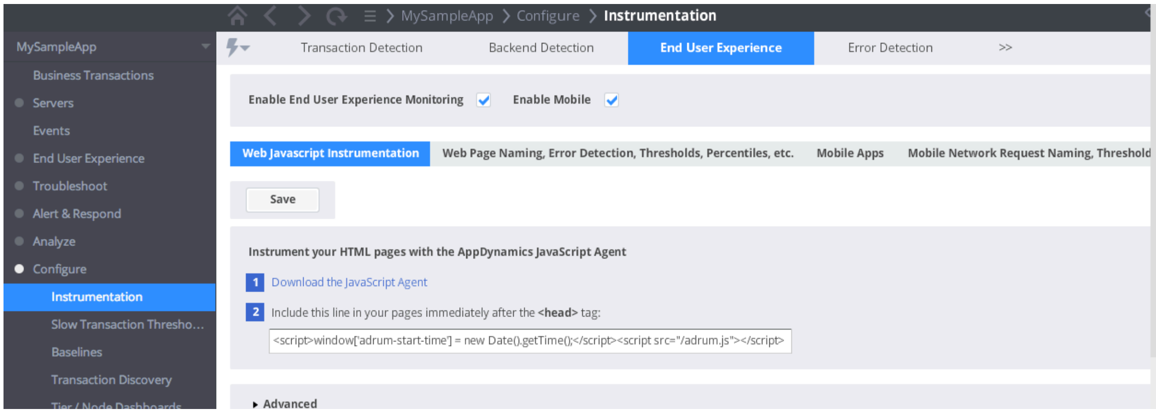
1) Configure the End User Experience (EUM) headers with the AppDynamics system

This step assumes that the application being instrumented has already has the AppDynamics application agent installed and operational. After the application agent is installed, JavaScript and cookie injection can be done automatically using the steps highlighted here.

Refer to the AppDynamics configuration guide for the complete information on this configuration. The end result should be the presence of ADRUM_BT cookies within the application stream. In general:

A) Navigate to the application you would like to instrument in the AppDynamics User Interface.
B) Click the **Configure** menu item.
C) Click the **Instrumentation** menu item.
D) Select **End User Experience** from the top navigation.
E) Check the **Enable End User Experience Monitoring** checkbox and save the changes.

The UI will look similar to the following:



Enable End User Experience Monitoring ☑ Enable Mobile ☑

Instrument your HTML pages with the AppDynamics JavaScript Agent

1 Download the JavaScript Agent

2 Include this line in your pages immediately after the **<head>** tag:

```
<script>window['adrum-start-time'] = new Date().getTime();</script><script src="/adrum.js"></script>
```

Note: These instructions are a summary and apply to this specific version of AppDynamics. There are also multiple ways, both manually and automatically, to configure EUE. Refer to AppDynamics documentation for details.

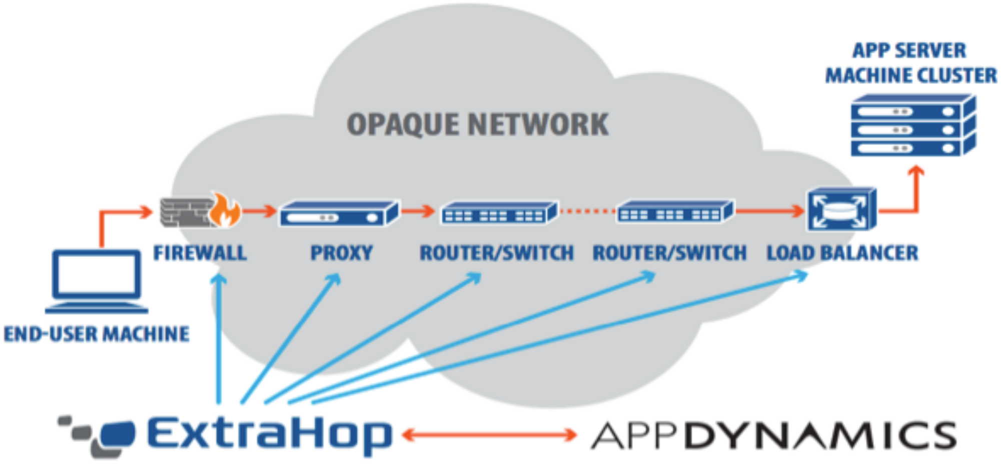
2) Explore and define the topology where the application lives

After configuring your application in AppDynamics for instrumentation, the next step is understanding the application topology in oder to implement an end-to-end monitoring solution. In this step, we will analyze and document the flow of the application through the network and the segments that we are interested in tracking.

The core function of the ExtraHop trigger is to record timing differences as the EUE cookie traverses each tier. Additionally, the trigger sends network metrics to AppDynamics. In order to accurately represent the traffic flow, we will begin by defining the "hops".

In our deployment, we are defining the topology using VLANs. It's also possible to define the topology via IP addresses or other uniquely identifiable aspects of each segment of the devices where traffic traverses.

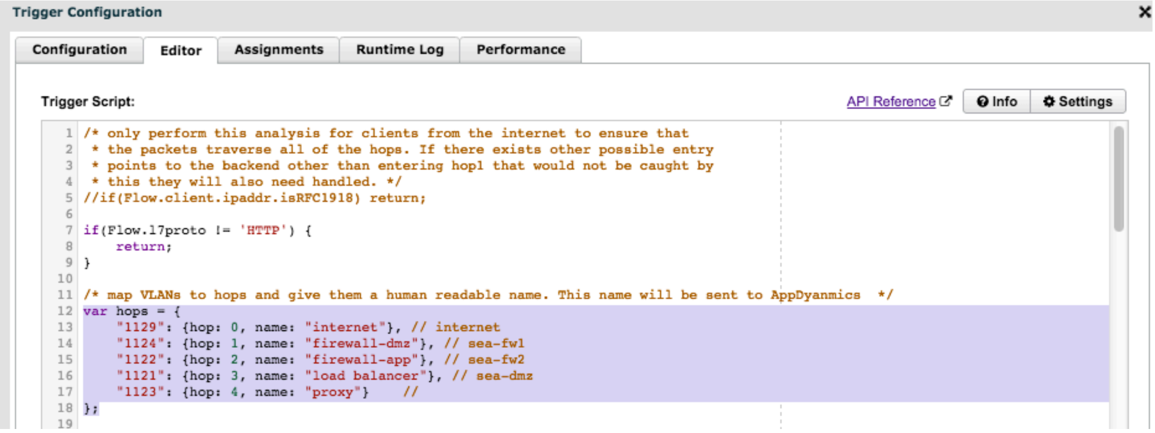
Refer to the following example:



The diagram above shows the traffic traversing a firewall, a proxy, a router, and a load balancer. If the firewall VLAN is 1000, the internal side of the first proxy is VLAN 1001, and the internal side of the load balancer is VLAN 1002, then the topology definition is as follows:

VLAN1000 :Hop1
VLAN1001: Hop2
VLAN1002: Hop3

Now, let's look at a more specific example with the trigger code.



```
/* only perform this analysis for clients from the internet to ensure that
 * the packets traverse all of the hops. If there exists other possible entry
 * points to the backend other than entering hop1 that would not be caught by
 * this they will also need handled. */
//if(Flow.client.ipaddr.isRFC1918) return;

if(Flow.l7proto != 'HTTP') {
    return;
}

/* map VLANs to hops and give them a human readable name. This name will be sent to AppDyanmics  */
var hops = {
    "1129": {hop: 0, name: "internet"}, // internet
    "1124": {hop: 1, name: "firewall-dmz"}, // sea-fw1
    "1122": {hop: 2, name: "firewall-app"}, // sea-fw2
    "1121": {hop: 3, name: "load balancer"}, // sea-dmz
    "1123": {hop: 4, name: "proxy"}    //
};
```

In the trigger snippet above, you can see that we have five hops defined: VLANs 1129, 1124, 1122, 1121, and 1123. These VLANs are defined as hops zero through four, and they have human-readable names: internet, firewall-dmz, firewall-app, load balancer and proxy.

In the next section, you will update the trigger with the information you gathered in this step.

3) Implement the topology and implement the ExtraHop trigger

Now you must define the topology in the trigger and implement it. Remember that not every hop must be defined, but you should define every hop that is practical, or particularly problematic hops. Here you will import the trigger into the environment, save it, and then edit the key pieces.

To configure the trigger, we will first classify the events. For this trigger, we are interested in HTTP_REQUEST and FLOW_TICK. Second, we will define the topology within the trigger code that we defined in step 2 above. Finally, we will assign devices to this trigger, specifically, all of the devices in the traffic path we want to capture.

A) Create or import the trigger and give it an appropriate name. In this example, the name is Cookie Analysis. Included in the bundles section of the ExtraHop documentation site (docs.extrahop.com) you will find the AppDynamics Cookie Analysis bundle. You may simply import this bundle which will install the trigger on your ExtraHop device.
B) Add the FLOW_TICK and HTTP_REQUEST events to the trigger.
C) Paste or edit the trigger code in the ExtraHop Trigger Editor. Edit the "Map VLANs to hops" section with the information you gathered above, inserting the VLAN numbers, numbering them from 0 to *n*, and giving each a human-readable name. Note that the name will be the name of the metric sent to AppDynamics.
D) Save and close the trigger.
E) Navigate to the **Devices** section of the ExtraHop Web UI, and search for the devices associated with the VLANS and servers you will be tracking. Select the checkbox next to each device and then click the **Select Action** drop-

down list in the upper right corner of the window to assign the device to the “cookie analysis” trigger.

Note: Pay particular attention to step C above. Now that you have activated the trigger, within the trigger code, ensure that you have adjusted the section “map VLANs to hops” to match your environment, changing the VLAN numbers, adding or removing hops numbers, and updating the comments about which hop you are tracking.

4) Edit the ExtraHop trigger to configure the specific metrics to send to AppDynamics

In the trigger you created above, there are two sections, one for HTTP metrics (captured in the HTTP_REQUEST function) and one for general TCP metrics (captured in the FLOW_TICK function). You will now modify the user-configurable settings in each section.

The user-configurable settings for the ODS for HTTP begin on or around line 66 with the string: `/* Start of user-configurable settings to send to AppDynamics /*`. Move your cursor to this line and edit the following variables:

A) **machineAgentEventURL:** Enter the URL of the Machine Agent Event REST interface. In our case, we enter “`/machineagent/event`. You do not need to adjust this variable unless AppDynamics changes their REST API. This document uses AppDynamics version 3.9.

B) **machineAgentMetricURL:** Enter the URL of the Machine Agent Metric REST interface. In our case, we enter `/machineagent/metrics`. You should not need to adjust this variable unless AppDynamics changes their REST API.

C) **appName:** Enter the name of the application you are sending data to for AppDynamics. This does not have to match the name exactly, but this is the name that will appear in the AppDynamics Metric Browser. In our case we enter `MySampleApp`.

D) **ExtraHop:** Enter the name of your ExtraHop appliance. This name is sent to the AppDynamics Event system and identifies the ExtraHop appliance containing the events. In our case we enter: `myExtraHopName`.

E) **timestamp** and **clientIP:** You do not need to change these but are here as an example of other types of information you can add or remove as you send metrics to AppDynamics.

Save your changes and repeat the same changes for the FLOW function, beginning on or around line 165 with the string: `/* Start of user configurable settings to send to AppDynamics /*`.

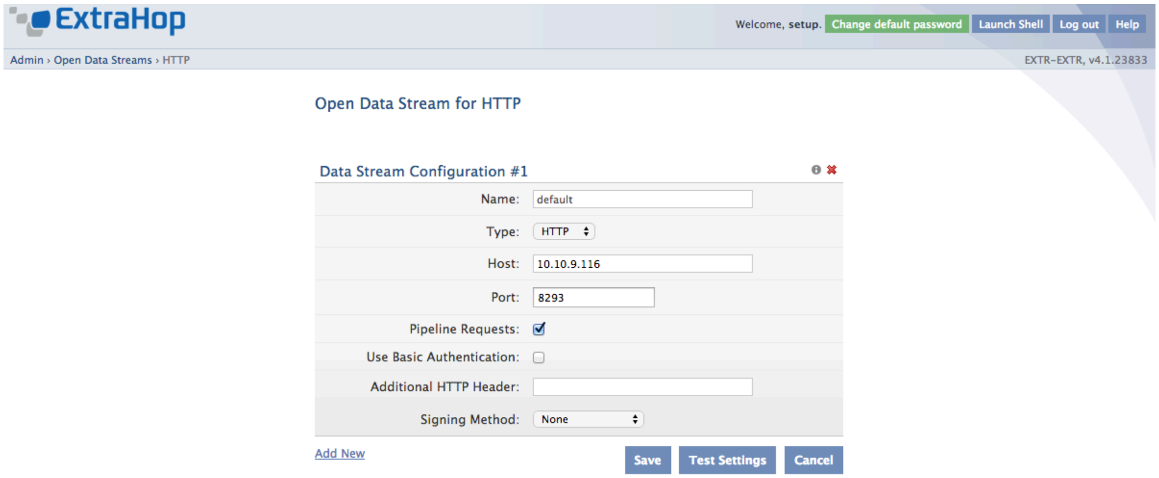
5) Configure ExtraHop ODS for HTTP output

The ExtraHop ODS for HTTP configuration is set in the ExtraHop Administration UI.

A) In the ExtraHop Web UI, click **Settings** in the navigation bar and then select **Administration**.
B) Log in with the proper credentials.
C) In the Configuration, section, click **Open Data Streams**.
D) Click **HTTP**.
E) On the Open Data Stream for HTTP, click **Add New** in the lower-left corner.
F) Configure the following settings:

- **Name:** If this is the first ODS HTTP destination, this will be the default connection. Otherwise, choose a name that will be used in your trigger. In our example, we use the first and default connection.
- **Type:** We choose HTTP.
- **Host:** Enter the IP Address of the host where the Machine Agent is installed. In our example we enter: 10.10.9.116
- **Port:** Enter the port of the AppDynamics Machine Agent. The default is 8293.
  Note: Make sure your ExtraHop management interface has the ability to reach this port, including any intermediary firewalls, routers, and host ACLs.
- For the remaining options, use the defaults.

Your configuration will look similar to the following:



Open Data Stream for HTTP

Data Stream Configuration #1

Name:	default
Type:	HTTP
Host:	10.10.9.116
Port:	8293
Pipeline Requests:	☑
Use Basic Authentication:	□
Additional HTTP Header:	
Signing Method:	None

Click **Save** to save your configuration.

You may also click **Test Settings** if you have already set up the AppDynamics Machine Agent. If you have not, return to this screen and run this at a later time. The result of the test should be `HTTP 200 OK`.

6) Implement and configure the AppDynamics Machine Agent

There are three main components to configuring the AppDynamics Machine Agent. First, install the Machine Agent must be installed. Then, configure it to activate the HTTP listener functionality. Finally, configure the controller information.

A) Refer to the AppDynamics installation instructions to obtain and install the Machine Agent.

B) After installation, enable the HTTP listener functionality. You must modify the startup script to add two values: `metric.http.listener=true` and `metric.http.listener.port`. In our installation, we add the following to values to the Machine Agent Startup script in `/etc/init.d`:

```
AGENT_OPTIONS="$AGENT_OPTIONS -Dmetric.http.listener=true"
AGENT_OPTIONS="$AGENT_OPTIONS -Dmetric.http.listener.port=8293"
```

C) Edit the controller-info.xml file to associate this Machine Agent with the application that will be receiving the metrics from ExtraHop. These are values for our implementation:
   a. controller-host: appdynamics.seadmz.example.com
   b. controller-port: 8090
   c. application-name: MySampleApp
   d. tier-name: WebTier
   e. node-name: Node1

D) Save and quit the configuration of the controller-info.xml file.

E) Start the Machine Agent and confirm that it is listening on the specific port, in this case, TCP 8293.

7) Explore the AppDynamics Event Viewer

The AppDynamics Event View is a log-based format that shows the events sent to the event URL of the Machine Agent.

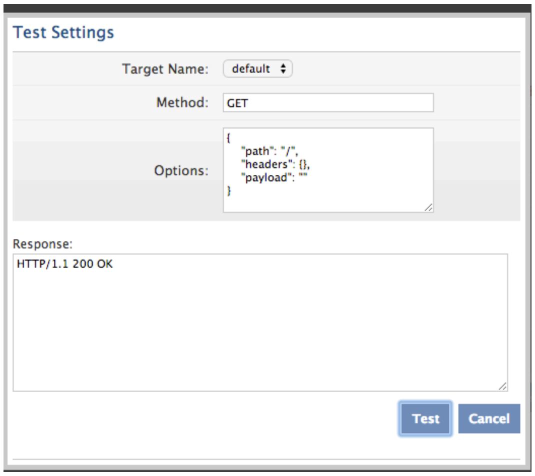
8) Explore and set up the AppDynamics Metric Browser

The AppDynamics Metric Browser allows the real-time plotting of ExtraHop metrics and agent data. The Metric Browser contains data sent to the metrics URL of the Machine Agent.

Validating Your Installation

To validate the ExtraHop ODS for HTTP connectivity to the AppDynamics Machine Agent:

1. Click **Test Connectivity** in the ExtaHop Open Data Stream (ODS) for HTTP window of the ExtraHop Admin UI while the AppDynamics Machine Agent is running. You should see an `HTTP 200 OK` message in the ExtraHop test response screen. It will look like this:



Test Settings

Target Name: default

Method: GET

Options:

```
{
    "path": "/",
    "headers": {},
    "payload": ""
}
```

Response:

```
HTTP/1.1 200 OK
```

2. Log in to AppDynamics and ensure that metrics and events are being transmitted from the Machine Agent to AppDynamics. Refer to AppDynamics documentation for more information about validating the Machine Agent.

Conclusion

The end-to-end analysis capabilities of ExtraHop and AppDynamics provide a lightweight solution to deploy and utilize, but it does have multiple moving parts. By combining the full view of both ExtraHop and AppDynamics data, developers, engineers and operators can see real-time correlated wire and agent data with visualizations provided by both vendors.

Next Steps

You have now instrumented one application environment with the combination of ExtraHop and AppDynamics.

The next steps for this deployment could take you in a variety of directions. You may choose to customize the dashboards on both ExtraHop and AppDynamics to provide information relevant to your environment. Alternately, you can set up another integration to instrument other applications with AppDynamics by repeating the steps in this document for each environment.

FAQ

Where do I run the AppDynamics Machine Agent?

The AppDynamics Machine Proxy is a lightweight Java-based proxy that gathers machine-level metrics for AppDynamics. You can run the AppDynamics Machine Agent on any server that is able to run a Java virtual machine. Some of the requirements for the Machine Agent include the following:

A) A location that can run a Java Virtual Machine
B) A location that can accept incoming TCP HTTP messages on port 8293
C) A location that can communicate with the AppDyanmics Controller on port 80 or 443

What if I cannot use the AppDynamics End User Experience header system?

Application developers may want to consider using their own unique identifiers in place of the AppDynamics End User Experience headers. Another option may be to use a system such as Boomerang. ExtraHop provides guidance on how to use Boomerang for traffic tracing here: http://www.extrahop.com/solutions/by-category/real-user-monitoring/. While metric collection with ExtraHop will be possible, using custom EUM headers changes the nature of the solution. If it is not possible to use EUE headers in your configuration, consult your ExtraHop or AppDynamics representative for more information.