PROBLEM:

Organizations, even those not typically associated with technology, are migrating to the cloud. This trend is growing because the cloud offers increased flexibility and agility. With this mass migration, organizations have more segments to manage and more potential blind spots in their networks. Regardless of where infrastructure and applications reside, security and compliance needs remain the same. Organizations are finding that their traditional network visibility solutions are unable to meet their needs for visibility of cloud-based data.

SOLUTION:

CloudLens™, Ixia’s platform for public, private and hybrid cloud visibility addresses the challenges of granular data access in the cloud. CloudLens is the first network-level solution that provides Visibility-as-a-Service (VaaS) through a Software-as-a-Service (SaaS). It is also the industry’s first cloud service-provider agnostic visibility platform.

KEY FEATURES:

- Elastically scales on-demand – so visibility auto-scales horizontally along with the instances monitored and the cluster of instances that are needed to do the monitoring
- Automates cloud visibility management by providing it as a service – so no architectural changes required
- Reduces errors by eliminating manual configuration
- Easy to use and setup with a drag and drop interface
- Reduces bandwidth to tools by filtering packets at the source instances, eliminating unwanted traffic so tools operate optimally
* Shown above is a sample deployment, while destination components need to in the same subnet, monitored sources instances can be located in any subnet, VPC, or AWS Region. CloudLens Sensors run on customer AWS instances, register up to the CloudLens SaaS which manages them and forwards desired traffic to the destination.
PREPARE AWS ENVIRONMENT

NOTE: IN THIS EXAMPLE WE ARE ASSUMING THE TWO SOURCE INSTANCES DO NOT ALREADY EXIST, IF THEY ALREADY EXIST YOU DO NOT NEED TO CREATE THEM, YOU CAN ATTACH THOSE AND ANY OTHER SOURCE INSTANCES TO CLOUDLENS AS DESCRIBED ON P 6-11 OF THIS DOCUMENT. YOU DO ALWAYS NEED TO CREATE ONE LINUX INSTANCE, THIS ONE WILL BE USED FOR FORWARDING TRAFFIC FROM CLOUDLENS TO RIVERBED VIA GRE AS DESCRIBED ON P 12. ALSO, WE ARE ASSUMING IN THIS EXAMPLE THAT AN INSTANCE OF RIVERBED APPRESPONSE IS ALREADY INSTALLED IN AWS, PLEASE CONTACT RIVERBED FOR ASSISTANCE IF NEEDED.

In this sample set up we will be creating one sample Windows 2012 R2 instance and two sample Ubuntu 16.04 Linux instances (other Linux types are also supported). We will use one Ubuntu 16.04 Linux instance as the source instance and one Ubuntu 16.04 Linux instance to forward traffic from CloudLens P2P VPN Tunnels to GRE Tunnel origination point. The GRE Tunnel will terminate on Riverbed Steelcentral Appresponse 11.

Step 1 – Log into the AWS Portal. Click “Launch Instance” within the EC2 service.

Create Instance

To start using Amazon EC2 you will want to launch a virtual server, known as an Amazon EC2 instance.

Launch Instance

Note: Your instances will launch in the US East (N. Virginia) region

Step 2 – Choose Windows 2012 R2 Server. Click “Select”

Windows

Microsoft Windows Server 2012 R2 Base - ami-f6529b8c
Microsoft Windows 2012 R2 Standard edition with 64-bit architecture. [English]
Root device type: ebs
Virtualization type: hvm
ENA Enabled: Yes

and

Ubuntu Server 16.04 LTS (HVM), SSD Volume Type - ami-da05a4a0
Ubuntu Server 16.04 LTS (HVM),EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www Canonical.com/cloud/services).
Root device type: ebs
Virtualization type: hvm
ENA Enabled: Yes

Step 3 – Enter Virtual Machine instance type (e.g. t2.xlarge)

| General purpose | t2.xlarge | 4 | 10 | EBS-only | - | Moderate | Yes |

Step 4 – Select configuration details

Note: Recommended - choosing an IAM role (e.g. ec2_metadata_access) which allows metadata access will permit AWS Tags to be shared with CloudLens
### Step 5 – Add storage

<table>
<thead>
<tr>
<th>Volume Type</th>
<th>Device</th>
<th>Snapshot</th>
<th>Size (GiB)</th>
<th>Volume Type</th>
<th>IOPS</th>
<th>Throughput (MiB/s)</th>
<th>Delete on Termination</th>
<th>Encrypted</th>
</tr>
</thead>
<tbody>
<tr>
<td>Root</td>
<td>/dev/sda1</td>
<td>snp-06e063e2ab024e9507</td>
<td>30</td>
<td>General Purpose SSD (GP1)</td>
<td>100 / 3000</td>
<td>N/A</td>
<td>Not Encrypted</td>
<td></td>
</tr>
</tbody>
</table>

**Add New Volume**

### Step 6 – Add Tags as desired, allows for easier identification and grouping of instances in CloudLens

<table>
<thead>
<tr>
<th>Key (127 characters maximum)</th>
<th>Value (255 characters maximum)</th>
<th>Instances</th>
<th>Volumes</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name</td>
<td>Demo Windows 2012 R2 Server</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CreatedBy</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Add another tag** (Up to 50 tags maximum)

### Step 7 – Assign a security group

Please see list of CloudLens required port numbers on P 15-16 of this document for guidance when creating or editing your security group.
Step 8 – Launch the instance with the correct key pair

A key pair consists of a **public key** that AWS stores, and a **private key file** that you store. Together, they allow you to connect to your instance securely. For Windows AMIs, the private key file is required to obtain the password used to log into your instance. For Linux AMIs, the private key file allows you to securely SSH into your instance.

**Note:** The selected key pair will be added to the set of keys authorized for this instance. Learn more about **removing existing key pairs from a public AMI**.

- **Choose an existing key pair**
- **Select a key pair**
  - `skx_key`
  - I acknowledge that I have access to the selected private key file (`skx_key.pem`), and that without this file, I won’t be able to log into my instance.

[Launch Instances]
INSTALL CLOUDLENS CONTAINER IN UBUNTU 16.04 VM

Note: Customers are assumed to have already created and account at http://ixia.cloud before completing the next step. If you don't have an account, you can sign up for a 45-day free trial. After login please create or view your CloudLens Project, and make note of the Project Key (aka API key) which you will need in step on page 7

Step 0 – If Docker Engine is not already present, install Docker from https://docs.docker.com/install/

  e.g. (commands will vary by OS type and version, see link above for more details)
  sudo apt update
  sudo apt-get install -y docker.io

Step 1 – Download CloudLens container from Docker Hub

The following command downloads the latest version of CloudLens Public container; sudo docker pull ixiacom/cloudlens-agent:latest
LAUNCH CLOUDLENS CONTAINER IN UBUNTU 16.04 VM

Step 1 – Launch CloudLens container with the following parameters. You will need to substitute your CloudLens Project Key (aka API key) here.

```
sudo docker run \
--name CloudLens \
-v /:/host \
-v /var/run/docker.sock:/var/run/docker.sock \
-d --restart=always \
--net=host \
--privileged \
ixiacom/cloudlens-agent:latest \
--server agent.ixia.cloud \
--accept_eula y \
--apikey <Project Key from CloudLens Portal> \
--custom_tags CloudServiceProvider=AWS \
Location=Oregon \
DeviceName=Riverbed-AR11
```
INSTALL CLOUDLENS AGENT IN WINDOWS SERVER VM

Step 1 – Download Ixia’s CloudLens agent from the link provided
https://agent.ixia.cloud/updates/windows/latest

Step 2 – Install CloudLens agent

Step 3 – Installation wizard goes through the CloudLens agent installation and all dependent package installations.
Step 4 – Accept End User License Agreement

Step 5 – Accept End User License Agreement

Step 6 – Click "Install"
Step 7 – The Windows instance should be associated with a Project created in https://ixia.cloud. The value of Host: agent.ixia.cloud. You must also specify your own Project Key (aka API key)

CloudLens Connection

Step 8 – Finish CloudLens sensor installation

Step 9 – Restart the instance and verify the instance is associated with the CloudLens project created.
Ixial CloudLens Setup

Ixial CloudLens

Installation Successfully Completed

You must restart your computer before you can use the software.

Restart  Close
CREATE GRE INTERFACE ON ONE OF THE UBUNTU 16.04 INSTANCE.

Step 1 – Log into AWS EC2 instance
Step 2 – Switch to super user mode.
Step 3 – ip link provides the ability to display link layer information, activate an interface, deactivate an interface, change link layer state flags, change MTU, the name of the interface etc. Create a GRETAP interface.

```bash
ip link add gre2 type gretap local <local Private ipv4 address> remote <Riverbed Steelcentral Private ipv4 address> dev eth0 ttl 255 key 1
```

ip link set gre2 up

Step 4 – Verify "gre2" interface is available and operational.

```
gre2      Link encap:Ethernet  HWaddr ee:3e:5d:63:06:10
inet6 addr: fe80::ec3e:5dff:fe63:610/64 Scope:Link
UP BROADCAST RUNNING MULTICAST  MTU:8959  Metric:1
RX packets:0 errors:0 dropped:0 overruns:0 frame:0
TX packets:16885852 errors:0 dropped:0 overruns:0 carrier:0
collisions:0 txqueuelen:1000
RX bytes:0 (0.0 B)  TX bytes:3864521055 (3.8 GB)
```

Note: above commands will not persist after reboot. There are several ways to automate persistence, here is one example (details may vary by OS type/version, as well as interfaces names)

Edit the interface config file e.g.
```
vi /etc/network/interfaces.d/50-cloud-init.cfg
```
and add the following lines e.g.
```
down ip link set gre2 down

down ip link delete gre2

up ip link add gre2 type gretap local <local Private ipv4 address> remote <Riverbed Steelcentral Private ipv4 address> dev eth0 ttl 255 key 1

up ip link set gre2 up
```
USING CLOUDLENS SAAS PORTAL

Step 1 – Verify the VMs are reflected in the CloudLens SaaS portal once they are launched with the correct project key.

Step 2 - Use CloudLens tags ingested from AWS to create

- Windows Source Group
- Ubuntu Source Group

CloudLens to GRE Tool Group
Step 3 – Create secure visibility path between source and tool groups

![Image of a diagram showing CloudLens integration with Riverbed AppResponse Cloud in AWS]

Step 4 – Login to Riverbed AppResponse Cloud hosted in AWS

Verify traffic from windows and ubuntu source instances are available in Riverbed Steelcentral Appresponse Cloud.

![Image of traffic summary and bandwidth usage from the Riverbed AppResponse Cloud interface]

Summary: All Traffic

Busyest Apps > User Response Time

<table>
<thead>
<tr>
<th>Application</th>
<th>Server Traffic [%]</th>
<th>User Response Time [ms]</th>
</tr>
</thead>
<tbody>
<tr>
<td>HTTP</td>
<td>1,000</td>
<td>0.02</td>
</tr>
<tr>
<td>SSL</td>
<td>198</td>
<td>123.73</td>
</tr>
<tr>
<td>Amazon Web Services</td>
<td>53</td>
<td>17.09</td>
</tr>
<tr>
<td>TOV532208</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>TOV45962905</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>TOV45962905</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>TOV45962905</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Busyest Server IPs > Server Response Time

<table>
<thead>
<tr>
<th>Top Server</th>
<th>Server Traffic [%]</th>
<th>User Response Time [ms]</th>
</tr>
</thead>
<tbody>
<tr>
<td>54.240.201.195</td>
<td>27</td>
<td>13.12</td>
</tr>
<tr>
<td>54.240.201.493</td>
<td>24</td>
<td>13.12</td>
</tr>
<tr>
<td>e2-34-251-195-232.compute-1.amazonaws.com</td>
<td>21</td>
<td>12.22</td>
</tr>
<tr>
<td>e2-34-251-195-232.compute-1.amazonaws.com</td>
<td>21</td>
<td>12.22</td>
</tr>
<tr>
<td>e2-34-251-195-232.compute-1.amazonaws.com</td>
<td>18</td>
<td>39.51</td>
</tr>
<tr>
<td>e2-34-251-195-232.compute-1.amazonaws.com</td>
<td>15</td>
<td>16.06</td>
</tr>
<tr>
<td>e2-34-3-123-32.compute-1.amazonaws.com</td>
<td>12</td>
<td>36.29</td>
</tr>
</tbody>
</table>

Most Active Client IPs > User Response Time

<table>
<thead>
<tr>
<th>Top Client</th>
<th>Active Connections [%]</th>
<th>User Response Time [ms]</th>
</tr>
</thead>
<tbody>
<tr>
<td>198.198.198.198</td>
<td>40</td>
<td>1,167</td>
</tr>
<tr>
<td>146.33.142.142</td>
<td>2</td>
<td>10,121</td>
</tr>
<tr>
<td>146.183.232.232</td>
<td>2</td>
<td>10,121</td>
</tr>
<tr>
<td>189.209.209.9</td>
<td>2</td>
<td>10,121</td>
</tr>
<tr>
<td>189.35.18.62</td>
<td>1</td>
<td>10,121</td>
</tr>
<tr>
<td>42.119.200.184</td>
<td>1</td>
<td>10,121</td>
</tr>
</tbody>
</table>
CLOUDLENS APPRESPONSE DEPLOYMENT GUIDE FOR AWS

FIREWALL PORTS TO OPEN FOR CLOUDLENS

Note: default Security rule settings for AWS Instances is Outbound is open for All Traffic. But for Inbound a few ports numbers need to be explicitly opened:

Source Instances:
- UDP 19993 (CloudLens Tunnel) *
- TCP 22 (if Linux) **
- TCP 3389 (if Windows) **

GRE Intermediary Node:
- UDP 19993 (CloudLens Tunnel) *
- GRE Protocol *
- TCP 22 **

Riverbed SteelCentral Instance:
- GRE Protocol *
- TCP 22 **
- TCP 443 **

* Leave open all IP addresses, however if stricter controls are required contact Ixia support  
** Specify IP addresses of customer administrators

WHERE TO GET HELP

If you experience technical difficulties, please email cloudlens@keysight.com for assistance

© Keysight Technologies, 2017