

# AWS Web Application Deployment with CloudFront and WAF - Walkthrough Summary

**Author:** Tony Tran

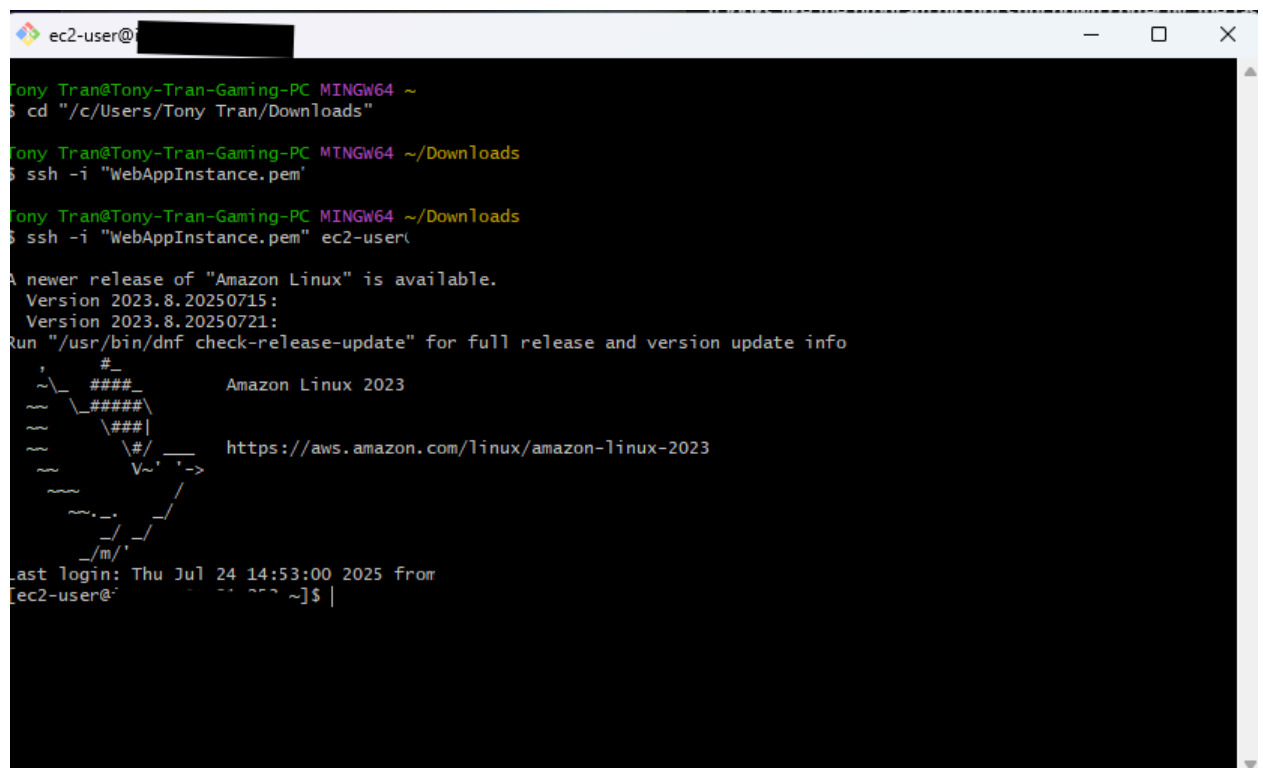
**Project Goal:** Deploy a secure web application using EC2, CloudFront, and AWS WAF with OWASP protection.

---

## 1. EC2 Setup

- **Launch EC2 instance** using Amazon Linux 2023.
- Open ports 22 (SSH) and 80 (HTTP) in the **Security Group**.
- Connect to the EC2 instance using .pem key with:

```
ssh -i "WebAppInstance.pem" ec2-user@<EC2 Public DNS>
```



The screenshot shows a terminal window titled 'ec2-user@' with a black background. The user 'Tony Tran' is at a 'MINGW64' prompt in the directory '~/Downloads'. They execute the command 'ssh -i "WebAppInstance.pem" ec2-user@' to connect to the EC2 instance. The terminal output shows the SSH connection process, including a warning about a newer release of Amazon Linux 2023 (Version 2023.8.20250715) and a message about checking for updates. The terminal then displays the Amazon Linux 2023 logo and the URL 'https://aws.amazon.com/linux/amazon-linux-2023'. The last login is recorded as 'Thu Jul 24 14:53:00 2025 from [ec2-user@'.

---

## 2. Install and Configure Web Server

- Install Apache web server:

```
sudo yum update -y  
sudo yum install httpd -y
```

```
sudo systemctl start httpd
sudo systemctl enable httpd
```

- Modify index.html:

```
echo "Hello from Tony's secure AWS web app!" | sudo tee /var/www/html/index.html
```

- Verify via EC2 public DNS:

```
curl http://<EC2 Public DNS>
```

**Output:**

```
Last login: Thu Jul 24 14:53:00 2025 from
[ec2-user@ip-172-31-21-253 ~]$ sudo systemctl start httpd
[ec2-user@ip-172-31-21-253 ~]$ sudo systemctl enable httpd
[ec2-user@ip-172-31-21-253 ~]$ curl http://<EC2 Public DNS>
Tony Tran's first AWS Project - EC2 + WAF + CloudFront + AWS CloudWatch Logs
07/2025
[ec2-user@ip-172-31-21-253 ~]$
```

---

### 3. CloudFront Configuration

- Create a **CloudFront distribution**:
  - **Origin**: EC2 Public DNS (e.g., ec2-34-228-71-196.compute-1.amazonaws.com)
  - **Origin Protocol Policy**: HTTP only
  - **Viewer Protocol Policy**: Redirect HTTP to HTTPS
  - Enable **WAF security protections**
- Wait until **Status = Deployed**
- **CloudFront URL**: <https://d1xvvj2ag2hgbf.cloudfront.net/>

```
<h1>Not Found</h1>
<p>The requested URL was not found on this server.</p>
</body></html>
[ec2-user@ip-172-31-21-253 ~]$ curl https://d1xvvj2ag2hgbf.cloudfront.net/
Tony Tran's first AWS Project - EC2 + WAF + CloudFront + AWS CloudWatch Logs
07/2025
```

---

### 4. AWS WAF Setup

- Create a **Web ACL** in AWS WAF:
    - Scope: CloudFront
    - Associate with the CloudFront distribution
    - Add managed rule group: **AWSManagedRulesCommonRuleSet** (OWASP protection)
-

## 5. Test WAF Protection

Test payloads via CloudFront URL:

XSS Payload

```
curl "https://d1xvvj2ag2hgbf.cloudfront.net/?search=<script>alert('x')</script>"
```

SQL Injection Payload:

```
curl "https://d1xvvj2ag2hgbf.cloudfront.net/?id=1' OR '1'='1"
```

Path Traversal Payload:

```
curl "https://d1xvvj2ag2hgbf.cloudfront.net/?search=../../etc/passwd"
```

Admin Enumeration Payload:

```
curl "https://d1xvvj2ag2hgbf.cloudfront.net/?admin=true"
```

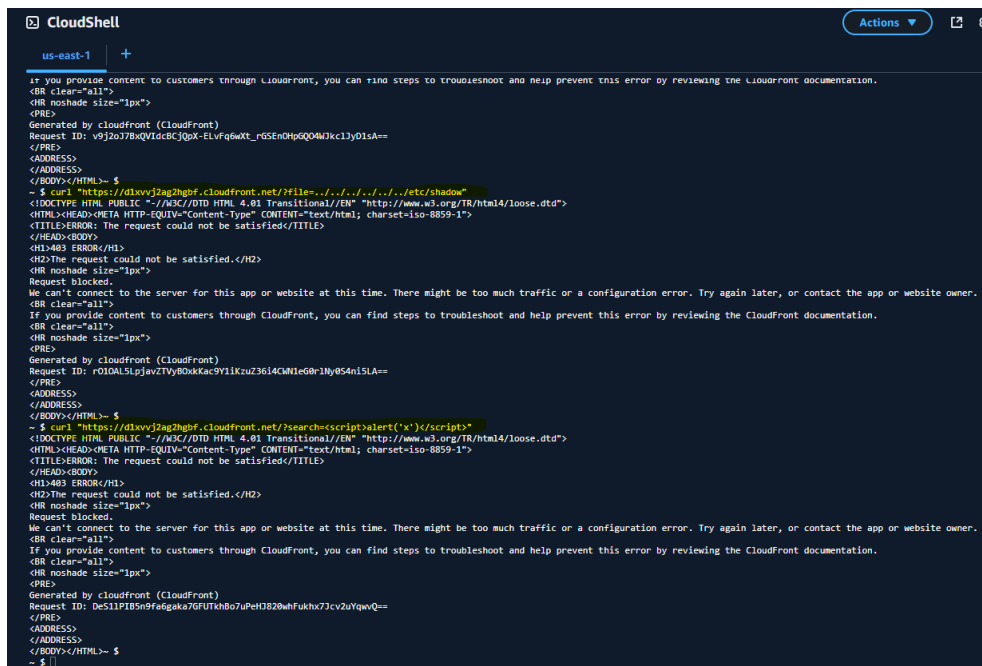
Command Injection (for completeness):

```
curl "https://d1xvvj2ag2hgbf.cloudfront.net/?cmd=ls%20-la"
```

Local File Inclusion (LFI) variant:

```
curl https://d1xvvj2ag2hgbf.cloudfront.net/?file=../../../../../../../../etc/shadow
```

All return:



```
CloudShell
us-east-1 +
if you provide content to customers through CloudFront, you can find steps to troubleshoot and help prevent this error by reviewing the CloudFront documentation.
<BR clear="all">
<HR noshade size="1px">
<PRE>
Generated by cloudfront (CloudFront)
Request ID: v9j2o7bKQVidCBCJQpX-ELvqgwXt_r05En0hpQ0Wkcl3j01sA==
</PRE>
</ADDRESS>
</ADDRESS>
</BODY></HTML>~ $
~ $ curl "https://d1xvvj2ag2hgbf.cloudfront.net/?file=../../../../../../../../etc/shadow"
<DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<HTML><HEAD><META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=iso-8859-1">
<TITLE>ERROR: The request could not be satisfied</TITLE>
</HEAD><BODY>
<H1>403 ERROR</H1>
<H2>The request could not be satisfied.</H2>
<HR noshade size="1px">
Request blocked.
We can't connect to the server for this app or website at this time. There might be too much traffic or a configuration error. Try again later, or contact the app or website owner.
<BR clear="all">
If you provide content to customers through CloudFront, you can find steps to troubleshoot and help prevent this error by reviewing the CloudFront documentation.
<BR clear="all">
<HR noshade size="1px">
<PRE>
Generated by cloudfront (CloudFront)
Request ID: r010ALS1pjavZTVy80xKac9Y11kzuZ3614CM1e60r1My054n1SLA==
</PRE>
</ADDRESS>
</ADDRESS>
</BODY></HTML>~ $
~ $ curl "https://d1xvvj2ag2hgbf.cloudfront.net/?search=<script>alert('x')</script>"
<DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<HTML><HEAD><META HTTP-EQUIV="Content-Type" CONTENT="text/html; charset=iso-8859-1">
<TITLE>ERROR: The request could not be satisfied</TITLE>
</HEAD><BODY>
<H1>403 ERROR</H1>
<H2>The request could not be satisfied.</H2>
<HR noshade size="1px">
Request blocked.
We can't connect to the server for this app or website at this time. There might be too much traffic or a configuration error. Try again later, or contact the app or website owner.
<BR clear="all">
If you provide content to customers through CloudFront, you can find steps to troubleshoot and help prevent this error by reviewing the CloudFront documentation.
<BR clear="all">
<HR noshade size="1px">
<PRE>
Generated by cloudfront (CloudFront)
Request ID: De51PIB5n9fag6ka7GfU1x80u7uPeH0820whFukhu73cv2uYqavQ==
</PRE>
</ADDRESS>
</ADDRESS>
</BODY></HTML>~ $
~ $ |
```

403 ERROR: The request could not be satisfied. Request blocked.

## 6. Troubleshooting Notes

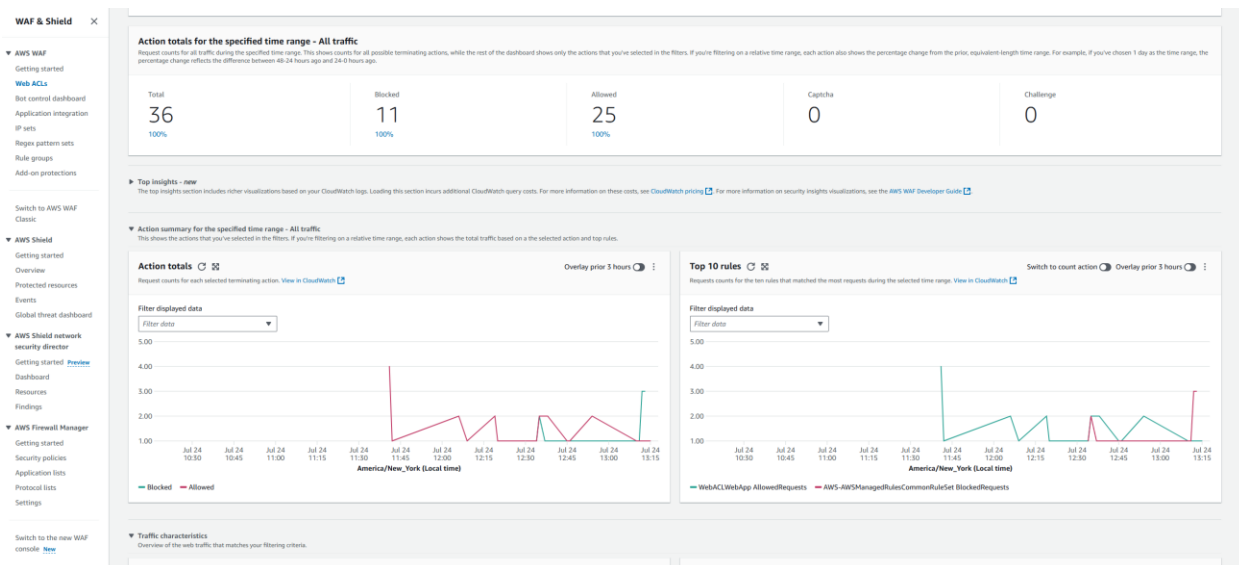
- If CloudFront returns 504 Gateway Timeout:
  - Confirm EC2 is running
  - Ensure EC2 allows traffic on port 80
  - Check CloudFront origin settings and wait for deployment

## 7. Key Services Used

- **EC2:** Hosts the web application
- **CloudFront:** CDN layer with performance and caching
- **AWS WAF:** Blocks malicious input using managed rules
- **Security Groups:** Controls inbound/outbound EC2 traffic

## 8. Outcome

- Successfully deployed a public web app with **CloudFront CDN** and **WAF protection**.
- Verified defense against **OWASP Top 10** payloads.



## ▼ Traffic characteristics

Overview of the web traffic that matches your filtering criteria.

### Sample of bot detection [Info](#) [↻](#)

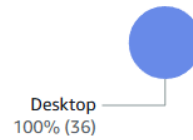
Counts for non-bot activity, and for verified and unverified bot activity. This includes only bots labeled by Bot Control for common bots [View in CloudWatch](#) [↗](#)



■ Bots:Unverified ■ Non-bots

### Client device types [↻](#)

The device types of the clients that sent the requests, obtained from the web request's user-agent header. [View in CloudWatch](#) [↗](#)



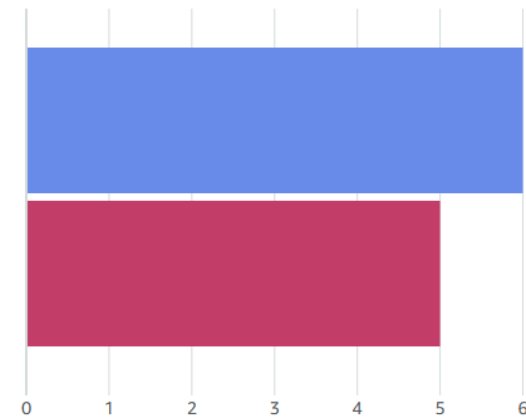
■ Desktop

### Attack types [Info](#) [↻](#)

The types of attacks identified in the requests. [View in CloudWatch](#) [↗](#)

Filter displayed data

Filter data ▼



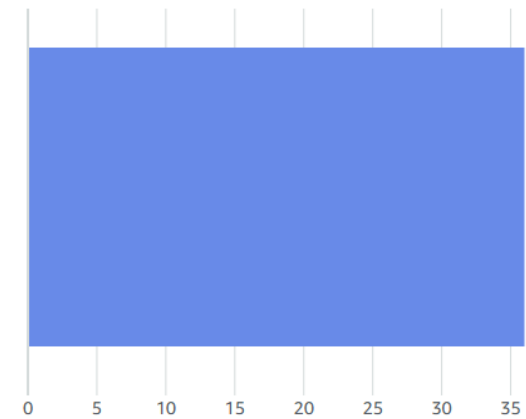
■ XSS ■ GenericLFI

### Top 10 countries [Info](#) [↻](#)

The ten countries that sent the most requests. [View in CloudWatch](#) [↗](#)

Filter displayed data

Filter data ▼



■ United States

## ▼ Managed rule groups