****

DeepakVakkala  
 **DevOpsIntern**

**OVERVIEW**

AWS WAF is a web application firewall that lets you monitor the HTTP(S) requests that are forwarded to your protected web application resources. You can protect the following resourcetypes:

* Amazon CloudFront distribution
* Amazon API Gateway REST API
* Application Load Balancer
* AWS AppSync GraphQL API
* Amazon Cognito user pool
* AWS App Runner service
* AWS Verified Access instance
* AWS Amplify.
* It is supported in Asia Pacific (Mumbai) - (ap-south-1), and its endpoint is
* **Endpoint in ap-south-1 (Mumbai)**:

wafv2.ap-south-1.amazonaws.com

wafv2-fips.ap-south-1.amazonaws.com

* AWS WAF can protect our applications that are hosted in Amazon Elastic Container Service (Amazon ECS) containers via API Gateway / ALB (Any one is Madatory, Without this connection is not possible)

**BEHAVIOUR**

1. Allow all, block some – Let all traffic through except specific ones you block (like bad IPs or attackers).
2. Block all, allow some – Block everything except specific traffic you allow (like known users).
3. Count only – Track requests that match certain rules without allowing or blocking them (used for testing or monitoring).
4. CAPTCHA or challenge – Add a CAPTCHA or silent challenge to suspicious traffic to reduce bots.

**BENEFITS**

* Protects against web attacks using rules based on:
  + IP Addresses
  + Countries
  + Request Headers (Key-Value Pair)
  + Specific words or patterns
  + Request size
  + Malicious Code(Like SQL injection or XSS)
* Flexible rules – We can allow, block or count requests, even based on how many requests happen in a short time.
* Reusable rules – Use the same rules for different applications.
* Managed rule groups – Use pre-built security rules from AWS or third-party providers.
* Real-time data – See live metrics and examples of requests.
* Easy automation – Use APIs to manage everything programmatically.

**HOW IT WORKS**

* You create a Web ACL (Access Control List) – It’s like a set of security rules.
* You attach this Web ACL to services like CloudFront, API Gateway, or Load Balancer.
* When a request comes in, AWS WAF checks it using your rules.
* Based on the rules, WAF can:
  + Allow the request
  + Block the request
  + Count it for monitoring
  + Challenge it using CAPTCHA to stop bots

**COMPONENTS**

* Web ACL: Holds all your rules and decides what happens to matching/non-matching traffic.
* Rules: Define what to look for in web requests (like IP, patterns, country, rate limits).
* Rule Groups: A group of reusable rules – can be created by you or provided by AWS/Marketplace.
* WCUs (Web ACL Capacity Units): A way AWS measures how "heavy" your rules are, to manage performance and costs.

**CONFIGURATION**

Step 1: Set up AWS WAF

Go to https://console.aws.amazon.com/wafv2/  
This is where we will manage your firewall settings.

Step 2: Create a Web ACL (Access Control List)

A Web ACL is like a set of rules that tells AWS which web requests to allow or block.

How we create one:

1. Go to the AWS WAF console.
2. Click Create web ACL.
3. Give your Web ACL a name.
4. (Optional) Add a description.
5. Set a name for CloudWatch metrics (for monitoring).
6. Choose the resource type to protect (e.g., CloudFront, ALB, or API Gateway).
7. If you want to link resources now, you can do that. Otherwise, skip.
8. Click Next.

Step 3: Add a String Match Rule

Here we can add rules to look for specific content in requests, like a header or IP address.

Example: Block requests with a specific User-Agent string

1. Click Add rules → Add my own rules → Rule builder.
2. Name your rule.
3. Choose Regular rule.
4. Set it to trigger if a request matches the rule.
5. Select which part of the request to inspect (like a header).
6. Choose User-Agent header and look for the string MyAgent.
7. Choose what happens when it matches (for now, set it to Count – just monitor).
8. Add the rule.

Step 4: Add AWS Managed Rule Group

AWS offers prebuilt rule sets to protect against common threats like SQL injection or bots.

To add these:

1. On the same page, choose Add managed rule groups.
2. Expand the AWS list.
3. Pick a rule group you want to use.
4. Set action to Count to see how it behaves first.
5. Save and add it to your Web ACL.

Step 5: Finish Web ACL Configuration

1. We can now set the priority order of rules (AWS processes them top to bottom).
2. On the metrics page, we will see options for CloudWatch monitoring and sampling.
3. Review all your settings.
4. Click Create Web ACL.

Web ACL is now ready and can be attached to a service like API Gateway, ALB, or CloudFront.

Step 6: Clean Up (Optional but Recommended)

To avoid charges after trying this:

1. Go back to the Web ACL list.
2. Click your Web ACL → Edit.
3. Disassociate it from any resources.
4. Go through the steps and finally Delete the Web ACL.

**IAM**

* By default, users and roles lack permission to create or modify AWS WAF resources; an IAM administrator can create IAM policies and assign them to roles, enabling users to assume the roles and perform necessary actions
* To manually delete the service-linked role using IAM - AWSServiceRoleForWAFV2Logging
* AWS WAF does not allow editing or renaming of the AWSServiceRoleForWAFV2Logging service-linked role, as it's referenced by multiple AWS entities.
* If a service-linked role is no longer needed, delete it after cleaning up related resources, ensuring AWS WAF isn’t actively using it to avoid deletion failure.
* AWS WAF automatically creates the service-linked role when logging is enabled via Console, CLI, or API, but you must have the iam:CreateServiceLinkedRole permission.
* AWS WAF uses the AWSServiceRoleForWAFV2Logging service-linked role, attached to the WAFV2LoggingServiceRolePolicy, to write logs to Amazon Data Firehose

**FEATURES**

* Identity-based policies
* Resource-based policies
* Policy actions
* Policy condition keys (service-specific)
* ACLs
* ABAC (tags in policies)
* Temporary credentials
* Forward access sessions (FAS)
* Service roles
* Service-linked roles

**POLICIES**

* WAFV2LoggingServic eRolePolicy
* AWSServiceRoleForW AFV2Logging
* AWSWAFFullAccess
* AWSWAFReadOnlyAccess
* AWSWAFConsoleFullAccess
* AWSWAFConsoleReadOnlyAccess

**ARN**

* The ARNs of AWS WAF wafv2 resources have the following format:
* arn:partition:wafv2:region:account-id:scope/resource-type/resource-name/resource-id

**API GATEWAY REST API**

* This IAM policy defines the permissions required to associate an AWS WAF WebACL with an API Gateway REST API. It allows users or roles to use AWS WAF and API Gateway to set up and manage security rules for the web application
* Associate a WebACL with AWS WAF (wafv2:AssociateWebACL)
* Set the WebACL on API Gateway stages (apigateway:SetWebACL)

**MONITERING**

* Monitoring is an important part of maintaining the reliability, availability, and performance of AWS WAF and your AWS solutions
* By Monitering we can more easily debug a multi-point failure if one occurs.

1. Amazon CloudWatch alarms

* We can watch a single metric over a time period that we specify
* If the metric exceeds a given threshold, CloudWatch sends a notification to an Amazon SNS topic or AWS Auto Scaling policy.

1. AWS CloudTrail logs

* It provides logs detailed records of requests to AWS WAF, including who made the request, when, from which IP, and what actions were taken.

1. AWS WAF web ACL traffic logging

* AWS WAF logging captures detailed request data, timestamps, and matched rule actions for traffic analyzed by your web ACLs.

**SUPPORTED HTTP STATUS CODES**

* 2xx – Success:  
  200, 201, 202, 204, 206
* 3xx – Redirection:  
  300, 301, 302, 303, 304, 307, 308
* 4xx – Client Errors:  
  400, 401, 403, 404, 405, 408, 409, 411, 412, 413, 414, 415, 416, 421, 429
* 5xx – Server Errors:  
  500, 501, 502, 503, 504, 505

**MITIGATING BOT TRAFFIC WITH CAPTCHA & CHALLENGE MECHANISM**

To reduce bot traffic and improve application security, AWS WAF offers a CAPTCHA feature that can challenge suspicious requests. When configured, AWS WAF can automatically present a CAPTCHA test to clients, ensuring only legitimate human users can access your resources.

* Enable CAPTCHA in AWS WAF: Set up custom rules in AWS WAF to detect bot-like behavior, such as high request rates or unusual traffic patterns. When these rules are triggered, a CAPTCHA challenge is presented to verify the requester's identity.
* Rate-Based Rules: Use rate-based rules to detect excessive requests from the same IP in a short time and trigger a CAPTCHA challenge for potential bot traffic.
* Integration with AWS Shield and Firewall Manager: AWS Shield Standard provides DDoS protection, and when combined with AWS WAF's CAPTCHA mechanism, it helps mitigate application-layer attacks. Additionally, AWS Firewall Manager can enforce consistent CAPTCHA protection across multiple accounts.

This CAPTCHA challenge mechanism provides an extra layer of defense against bots, helping secure your web applications from malicious automated traffic

**USING AWS\_WAF WITH AMAZON\_CLOUDFRONT**

AWS WAF (Web Application Firewall) helps protect your web applications by filtering traffic. You can use it with Amazon CloudFront to secure content delivery.

When you create a Web ACL (Access Control List), you can attach it to one or more CloudFront distributions. AWS WAF will inspect incoming web requests and take action based on your rules.

1. Custom Error Pages with CloudFront

By default, when AWS WAF blocks a request, it sends a 403 Forbidden status. CloudFront passes this to users, showing a basic error message. If you want to show a more user-friendly message (like one that matches your website’s style), you can:

* Configure CloudFront custom error pages to show your own HTML message.
* Or, configure AWS WAF rules to return a custom response.

Note: If both AWS WAF and CloudFront are set to return custom messages, the AWS WAF response will be used.

However, CloudFront cannot tell if the 403 error is from WAF or your origin server—so you can't show different pages for each.

2**.** Using AWS WAF for Your Own HTTP Server

You can use AWS WAF with CloudFront even if your web server is hosted outside AWS (like in your own data center or EC2).

* Secure communication:  
  You can require HTTPS between:
  + CloudFront and your server (by setting origin settings and using a trusted SSL certificate).
  + Viewers and CloudFront (by setting the Viewer Protocol Policy).  
    You can also use your own domain name with HTTPS using a custom SSL certificate.

3. Choosing HTTP Methods CloudFront Accepts

When setting up your CloudFront distribution, you can choose which HTTP methods it supports:

* GET, HEAD – For reading data only.
* GET, HEAD, OPTIONS – For reading and checking supported options.
* All methods (GET, HEAD, OPTIONS, PUT, POST, PATCH, DELETE) – For full access including adding, updating, and deleting data.

You can also use AWS WAF rules to allow or block certain methods. For example:

* If you want only GET and HEAD, just set that in CloudFront.
* If you want GET, HEAD, and POST, set CloudFront to allow all methods, then use WAF to block the ones you don’t want.

**INTEGRATION WITH AWS\_FIREWALLMANAGER & AWS\_SHIELD**

Set up AWS Firewall Manager

* Enable AWS Firewall Manager: Go to the AWS Firewall Manager console and enable the service in your organization.
* Create WAF Policy in Firewall Manager: Create a WAF policy in Firewall Manager to enforce the use of AWS WAF Web ACLs across multiple accounts and resources.
  + Select the resources you want to protect, such as CloudFront distributions, Application Load Balancers (ALB), or API Gateway.

2. Integrate AWS WAF with Firewall Manager

* Assign WAF Web ACL to Resources: Within the WAF policy in Firewall Manager, assign the Web ACL (created in AWS WAF) to the resources in your AWS environment.
* Automate WAF Deployment: AWS Firewall Manager will automatically deploy and enforce the Web ACL across multiple accounts and regions, ensuring consistent security configurations.

3. Integrate AWS Shield Standard

* AWS Shield Standard is automatically enabled for AWS services like ALB, CloudFront, and Route 53 at no extra cost. There is no manual configuration required to integrate Shield Standard with these resources.
* AWS Shield Standard will automatically protect these resources from DDoS attacks.

4. Combine AWS WAF and Shield Standard

* Leverage Both for Layered Security: While AWS Shield Standard provides DDoS protection, AWS WAF can be used to filter traffic at the application layer for threats like SQL injection and cross-site scripting (XSS).
* AWS Firewall Manager ensures that WAF policies are consistently applied to resources, integrating seamlessly with Shield Standard protection.

5. Monitor and Respond to Threats

* Monitor Traffic and Attacks: Use CloudWatch and AWS Shield metrics to monitor for security incidents.
* Automate Threat Mitigation: Configure automatic responses in CloudWatch or Lambda functions to respond to attacks detected by WAF or Shield.

**QUOTAS**

General Limits (Per Account, Per Region)

* Web ACLs: 100
* Rule Groups: 100
* IP Sets: 100
* Requests/sec per Web ACL: 25,000
* Custom Request/Response Headers: 100 each
* Custom Response Bodies: 50
* Token Domains per List: 10
* Regex Sets: 10

Fixed Limits (Cannot Be Increased)

* Web ACL WCUs: 5,000
* Rule Group WCUs: 5,000
* Reference Statements per Web ACL or Rule Group: 50
* IP Addresses per IP Set: 10,000
* Rate-based Rules per Web ACL: 10
* Rate-based Rules per Rule Group: 4
* Minimum Rate Limit: 10 req/sec
* Unique IPs Rate Limited: 10,000
* String/Regex Lengths: 200 characters
* Regex Patterns per Set: 10

Request Body Size for Inspection

* ALB, AppSync: 8 KB
* CloudFront, API Gateway, Cognito, etc.: up to 64 KB (default 16 KB)

API Call Limits (Per Account, Per Region)

* Associate/Disassociate Web ACL: 1 every 2 seconds
* Get/List Actions: 1–5 requests per second
* Create/Update Actions: 1 request per second

API Call Limits (Across AWS Organization)

* For ListResourcesForWebACL:
  + us-east-1, us-west-2, eu-west-1: 12 requests/sec total
  + Other Regions: May vary

**LIMITATIONS**

AWS WAF: Limitations for Oversize Web Request Components

1) Inspection Size Limits:

* Request Body:
* AWS WAF inspects the first 8 KB for Application Load Balancer and AWS AppSync.
* For CloudFront, API Gateway, Amazon Cognito, App Runner, and Verified Access, the default is 16 KB, configurable up to 64 KB.
* Headers:
* Inspects only the first 8 KB or the first 200 headers, whichever limit is reached first.
* Cookies:
* Inspects only the first 8 KB or the first 200 cookies, whichever limit is reached first.

2) Uninspected Content:

- If a request exceeds these limits, only the portion within the limit is evaluated by WAF.

- The entire request is still forwarded to the protected resource, even if some parts weren’t inspected

3) Oversize Handling Options in Rules:

- When defining rules that inspect limited-size components, AWS WAF requires an oversize handling setting:

\* Continue (default) – Inspect what’s within size limits.

\* Match – Treat as a match without evaluating contents.

\* No Match – Skip evaluation and proceed with other rules.

4) Console Requirement:

- You must explicitly choose one of the above options when using the AWS Console.

- Default behavior outside the Console is Continue.

5) Managed and Shared Rule Groups:

- Oversize component handling in AWS Managed Rules or shared rule groups cannot be changed.

- It's important to verify how these rules handle oversized request components.

* Block Oversized Web Request
* Edit your Web ACL and go to Add rules > Add my own rules > Rule builder > Rule visual editor.
* Name the rule and keep the type as Regular rule.

1. In match settings

* Choose what to inspect: Body, Headers, or Cookies.
* Set Match type to Size greater than.
* Enter the size limit:
* For Headers/Cookies: enter 8192.
* For Body:
* Use 8192 for Application Load Balancer or AWS AppSync.
* Use 16384 (or your configured limit) for CloudFront, API Gateway, Cognito, App Runner, or Verified Access.
* For Oversize handling, select Match.

1. Set the Action to Block.
2. Add the rule.
3. On the priority page, move this rule above any similar rules so it runs first.

**BEST PRACTICES**

- Start with 'Count' mode to observe traffic without blocking.  
- Use AWS Managed Rules for basic protection.  
- Regularly review CloudWatch logs and metrics.  
- Group common rules into reusable rule groups.  
- Limit request body inspection size to reduce cost.  
- Monitor WCU (Web ACL Capacity Units) to optimize performance and cost.

--- THE END ---