AWS Best Practices for DDoS Resiliency

June 2015

Notices

Contents

Abstract

This paper is intended for customers who want to improve resiliency of their applications running on Amazon Web Services (AWS) against Distributed Denial of Service attacks. The paper provides an overview of Distributed Denial of Service attacks, techniques that can help maintain availability, and reference architectures to provide architectural guidance with the goal of improving your resiliency.

The paper is targeted at IT decision makers and security personnel who are familiar with basic concepts in the area of networking, security, and AWS. Each section has links to AWS documentation with specific information on how to perform the tasks listed. You can also view AWS re:Invent sessions Sec305 and SEC307 for more information.

Introduction

A Denial of Service (DoS) attack is an attack that attempts to make your website or application unavailable to your end users. To achieve this, attackers use a variety of techniques that consume network or other resources, interrupting access for legitimate end users. In its simplest form, a DoS attack against a target is executed by a lone attacker via a single host, as shown in Figure 1.

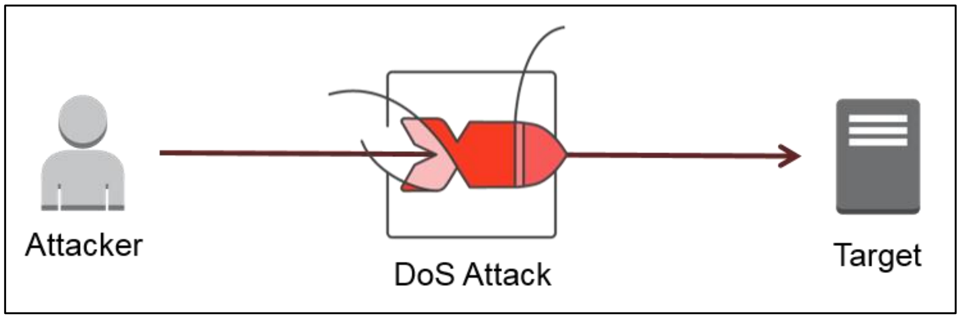

Figure 1: Diagram of DoS Attack

Distributed Denial of Service Attacks

In the case of a Distributed Denial of Service (DDoS) attack, the attackers use multiple hosts – which may be compromised or controlled by a group of collaborators – to orchestrate an attack against a target. As illustrated in Figure 2, in a DDoS attack, each of the collaborators or compromised hosts participates in the attack, generating a flood of packets or requests to overwhelm the intended target.

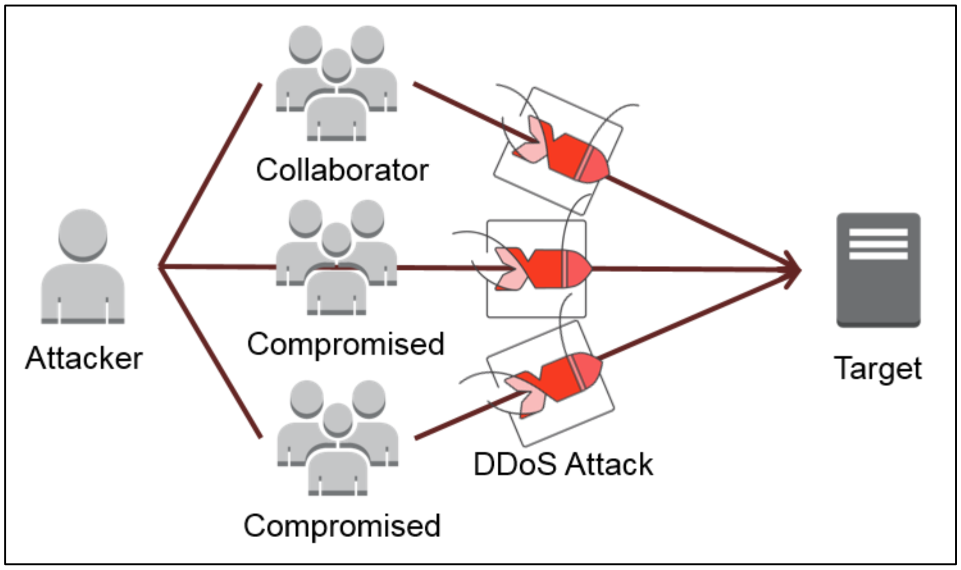

Figure 2: Diagram of DDoS Attack

There are many ways attackers can overwhelm a target. For example, attackers can generate large packet floods by using a combination of reflection and amplification techniques or use large botnets. Reflection attacks involve eliciting a response from a server to a spoofed IP address where the compromised server acts like a reflector.

Amplification attacks work by attackers sending a small packet or request that elicits a large response. The amplification factor, which is the ratio of request size to response size, varies depending on the protocol used (e.g., DNS, NTP, and SSDP). For example, the average amplification factor for DNS can be in the 28 to 54 range – which means an attacker can send a request payload of 64 bytes to a DNS server and generate 3,456 bytes of unwanted traffic.[1]

Reflection attacks in conjunction with amplification attacks result in a compromised server sending a response that is disproportionate to the original request. As shown in Figure 3, by combining these techniques, attackers can turn a small amount of bandwidth coming from a limited number of hosts into a large amount of traffic hitting the target.

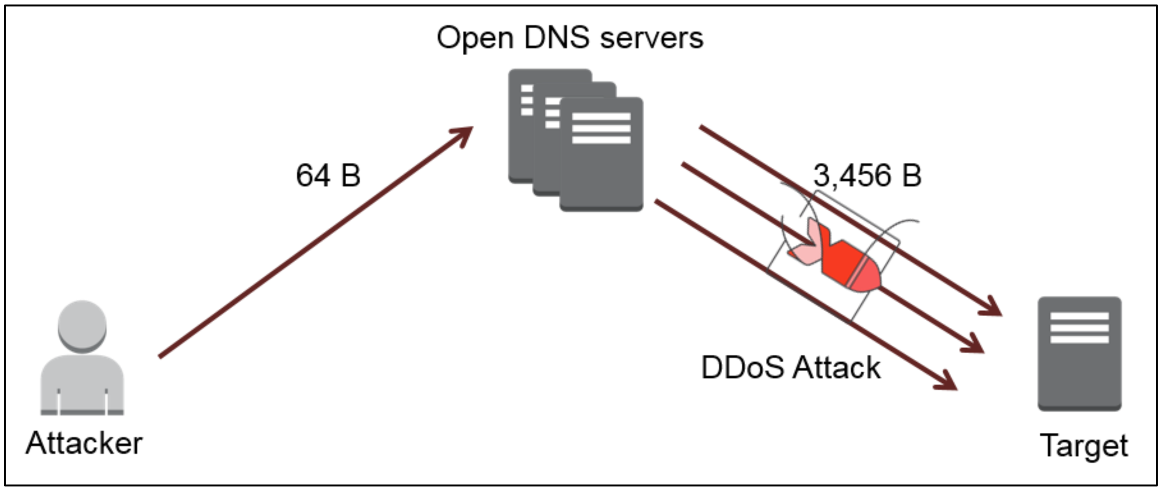

Figure 3: DDoS Flood, Reflection, and Amplification Attack

As the unwanted traffic reaches the target, the traffic traverses multiple layers of networking hardware, operating systems, and application layers and can exhaust resources at any of these layers. Depending on the consumed resources, the attack is

[1] https://www.us-cert.gov/ncas/alerts/TA14-017A

classified as bandwidth exhausting (e.g., UDP floods), protocol exhausting (e.g., SYN flood) or application exhausting (e.g., HTTP GET/POST floods).

Irrespective of the attack type, DDoS attacks at their core create an availability problem, as the goal of attackers is to render resources unusable for legitimate end users. Consequently, you can leverage failover capabilities within AWS to reduce your vulnerability to availability problems caused by DDoS attacks.

The amount of resiliency to DDoS attacks may vary depending on the services used and how they are configured; as such, the techniques described here do not guarantee a specific level of availability. When using the techniques listed in this paper, please keep in mind that AWS pricing is based on usage.

Mitigation Techniques

Traditionally, DDoS security involves the creation of filters or barriers to prevent attackers from accomplishing their goals. In addition to filtering and blocking techniques, you can employ a flexible architecture that can scale with the changing landscape. The remainder of this paper discusses five techniques that can be used to reduce your vulnerability to DDoS attacks:

- Minimize the Attack Surface Area
- Be Ready to Scale to Absorb the Attack
- Safeguard Exposed Resources
- Learn Normal Behavior
- Create a Plan for Attacks

This paper discusses each of these techniques along with a reference architecture as shown in Figure 4 to demonstrate how to build resiliency using AWS. You can also follow the general guidelines in the AWS Architecture Center for building systems that quickly failover as shown in Fault Tolerance and High Availability.

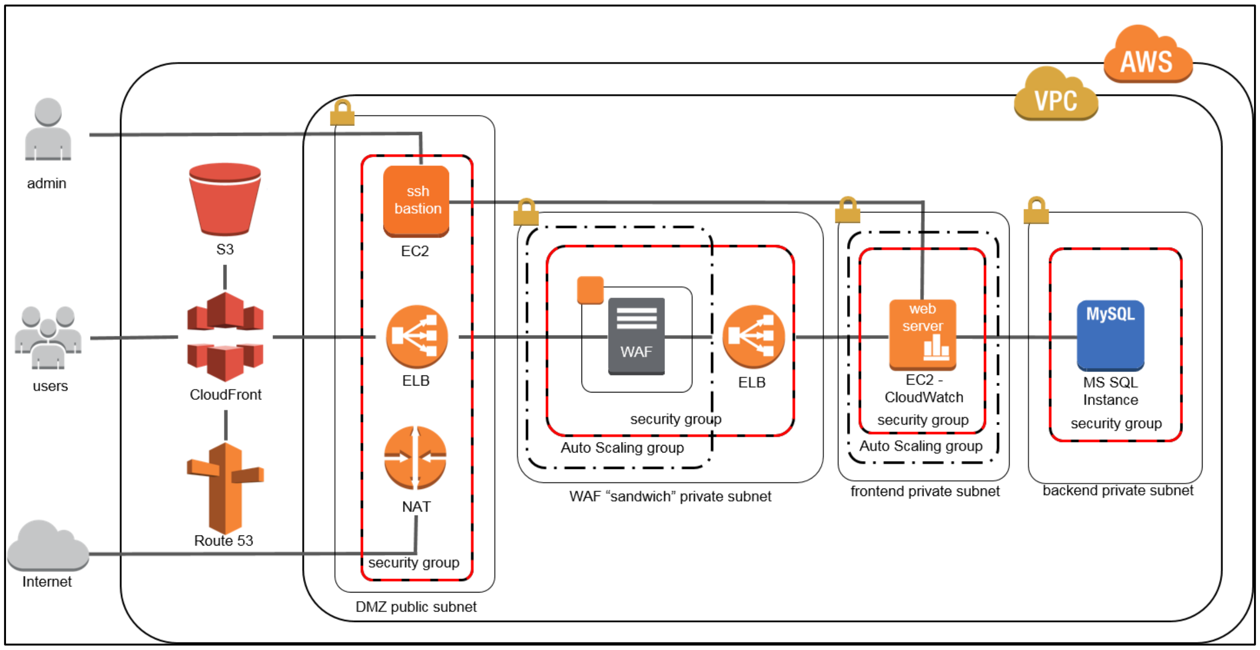

Figure 4: DDoS Resilient Reference Architecture

Minimize the Attack Surface Area

The attack surface is comprised of the different Internet entry points that allow access to your application. The strategy to minimize the attack surface area is to (a) reduce the number of necessary Internet entry points, (b) eliminate non-critical Internet entry points, (c) separate end user traffic from management traffic, (d) obfuscate necessary Internet entry points to the level that untrusted end users cannot access them, and (e) decouple Internet entry points to minimize the effects of attacks. This strategy can be accomplished with Amazon Virtual Private Cloud.

Amazon Virtual Private Cloud

Amazon Virtual Private Cloud (VPC) enables you to define a virtual network in your own logically isolated area within AWS, known as a virtual private cloud. VPC allows you to configure route tables, network gateways, and security settings in order to launch AWS resources like EC2 instances in a secured environment.

More importantly, VPC allows you to hide instances from the Internet to ensure non-public instances are only available on a private subnet and private DNS entries are only accessible by internal applications. This can be accomplished by creating Security Groups and Network Access Control Lists (ACLs). Security Groups act as a firewall for associated EC2 instances while Network ACLs act as a firewall for associated subnets. You can secure your VPC instances using security groups as the first layer of defense and add Network ACLs as the second layer. For more information on Security Groups and Networks ACLs, you can review Security in Your VPC.

Use the following links to configure VPC in the Amazon Management Console. If you're unsure which instance to select in the Step 3, proceed to the section entitled Amazon Elastic Compute Cloud before following the next steps.

Step 1: Set Up the VPC and Internet Gateway

Step 2: Set Up a Security Group for Your VPC

Step 3: Launch an Instance into Your VPC

Step 4: Assign an Elastic IP Address to Your Instance

Step 5: Set Up a Network Access Control List

You can also follow the steps for Amazon VPC common configuration scenarios listed below.

Scenario 1: VPC with a Public Subnet Only

Scenario 2: VPC with Public and Private Subnets

Scenario 3: VPC with Public and Private Subnets and Hardware VPN Access

Scenario 4: VPC with a Private Subnet Only and Hardware VPN Access

Once you've followed the instructions in the links, you can connect to your EC2 instances in VPC. For information on how to connect to a Linux instance, see Connect to Your Linux Instance in the *Amazon EC2 User Guide for Linux Instances*. For information about how to connect to a Windows instance, see Connect to Your Windows Instance in the *Amazon EC2 User Guide for Microsoft Windows Instances*.

Be Ready to Scale and Absorb the Attack

DDoS attacks are about scale. Most attackers achieve their purpose by sending a level of traffic that the application cannot accommodate. By implementing an architecture that can out-scale the attack, you create a barrier that requires more time and resources on the part of the attacker, thereby making your application more resilient.

Within AWS, you can take advantage of two forms of scaling: horizontal and vertical scaling. Horizontal scaling works by adding more instances or services to your infrastructure. Vertical scaling works by selecting instances with more memory, CPU, and capacity. Using these two dimensions of scale provides four direct benefits against DDoS attacks:

- The attack is dispersed over a wider area, minimizing the blast radius.
- Attackers have to expend more resources to scale up the attack.
- Scaling buys you time to analyze the DDoS attack and respond with countermeasures.
- Scaling provides an additional layer of redundancy for other failure scenarios.

In terms of DDoS, there are three ways to take advantage of scaling in AWS: (1) select the appropriate instance types for your application, (2) configure services such as Elastic Load Balancing and Auto Scaling to automatically scale, and (3) use the inherent scale built into the AWS global services like Amazon CloudFront and Amazon Route 53.

Amazon Elastic Compute Cloud

Instance Types

Amazon Elastic Compute Cloud (EC2) provides scalable compute capacity in the cloud while eliminating the need to invest in hardware up front. You can use EC2 to launch virtual servers called instances, where you can scale up or down to handle changes in traffic spikes. When you launch an instance, the instance type that you specify determines the hardware components such as compute, memory, and storage of the host computer used for your application.

To increase the resiliency of your application, you should select an instance type that can scale to support your application as well as unanticipated spikes. Some EC2 instances that are optimized for cost do not offer guaranteed network resources and may be more susceptible to the availability impact of DDoS attacks. For those applications that are critical to your infrastructure, you should select an EBS-optimized instance or an instance type with 10 Gigabit network connectivity as the host computer. For more information, you can review Amazon EC2 Instance Configuration.

Enhanced Networking

With C3, C4, R3, D2, and I2 instances, you can enable Enhanced Networking capabilities, which provides higher network performance (packets per second). This feature uses a network virtualization stack that provides higher I/O performance and lower CPU utilization compared to traditional implementations. With Enhanced Networking, your application can benefit from features that can aid in building resiliency against DDoS attacks, such as high packet-per-second performance, low latency networking, and improved scalability.

In order to enable Enhanced Networking, you must launch an Amazon Machine Image (AMI) that is the appropriate Hardware assisted Virtual Machine (HVM) with a Single Root I/O Virtualization (SR-IOV) driver. The Amazon HVM Linux AMI includes the SR-IOV driver by default and. For AMIs that do not contain the SR-IOV driver by default, you will need to download and install the appropriate driver. You can use the following links for instructions on how to download and enable Enhanced Networking for the following AMIs.

Enabling Enhanced Networking on Amazon Linux

Enabling Enhanced Networking on Ubuntu

Enabling Enhanced Networking on Other Linux Distributions

Enabling Enhanced Networking on Windows

Elastic Load Balancing

When you use Elastic Load Balancing (ELB) to manage traffic to your infrastructure, you get the benefit of distributing traffic to several EC2 instances across multiple Availability Zones (AZ) so that the risk of overloading one single instance is minimized. With ELB, you can add and remove EC2 instances as your needs change without disrupting the overall flow of traffic. For example, if one EC2 instance fails, ELB automatically reroutes traffic to the remaining running EC2 instances. When the failed EC2 instance is restored, ELB restores traffic to that instance.

ELB also offers a single point of management and can serve as the first line of defense against attacks on your network. You can place all your EC2 instances behind ELB and only expose ELB to the Internet, which helps in minimizing the attack surface area. You can point instances at ELB and scale as needed (either adding or removing capacity) without having to manage each individual instance. When used with VPC, you can associate Security Groups and Network ACLs with ELB to provide additional layers of resiliency. Because ELB only supports valid TCP requests, DDoS attacks such as UDP and SYN floods are not able to reach your instances.

The next section will show how to configure two basic load balancers: an Internet-facing load balancer and an internal load balancer. These two load balancers (shown in Figure 4) are required for scaling layer 7 protection detailed in the section Web Application Firewall.

Step 1: Create a Basic Load Balancer in Default VPC

Step 2: Create a Basic Internal Load Balancer in Amazon VPC

Repeat these steps for other regions where you have EC2 instances.

Auto Scaling

Auto Scaling helps you maintain application availability and allows you to scale your EC2 capacity up or down automatically according to conditions you define. For example, you can set a condition to incrementally add new instances to the Auto Scaling group when network traffic is high (typical of DDoS attacks). You can also set a condition to remove instances in the same increments when network traffic is low. You can use Amazon CloudWatch to trigger scaling activities and ELB to distribute traffic to your instances within Auto Scaling groups.

There are actions that you need to consider before you put your first Auto Scaling group into production. Before you start, take the time to review your application thoroughly as it runs in the AWS cloud and make note of the following:

- How long it takes to launch and configure your servers? *If you're application takes more than five minutes to start, we recommend having multiple instances already running your application or low thresholds for scaling.*
- What metrics have the most relevance to your application's performance? *Example metrics for DDoS attacks are* `CPUUtilization`, `NetworkIn`, *and* `StatusCheckFailed`.
- What existing resources (such as EC2 instances or AMIs) you might want to use as part of your Auto Scaling group? *You'll want the same type of instance or higher capacity running the application under attack for your Auto Scaling group.*

- To how many AZs do you want the Auto Scaling group to span? *We recommend a minimum of two AZs.*
- How fast should you scale up and down? *Keep in mind that DDoS attacks can come in waves. You don't want to scale down after the initial wave only to find out you have to scale back up again.*
- What is the maximum amount of EC2 instances for the Auto Scaling group? *Additional instances may increase your costs. When you create your Auto Scaling policy, you can set maximum number of instances. You can also set an alarm when this maximum number has been reached. See Amazon CloudWatch for steps on setting alarms.*

The better you understand your infrastructure and application, the more effective your implementation of Auto Scaling becomes. When you have enough information about your infrastructure and application, you can start creating Auto Scaling groups.

Step 1: Create a Launch Configuration

Step 2: Create an Auto Scaling Group

Step 3: Verify Your Auto Scaling Group

Amazon CloudFront

Amazon CloudFront is a Content Delivery Network (CDN) that integrates with other Amazon services to provide an easy and effective way to distribute content to end users. CDNs work by acting as a proxy layer between your origin and end users. CDNs improve performance by caching content and optimizing connections to your site from multiple Points of Presence (PoPs). The effect is that instead of sending all traffic requests back to your origin, requests are spread across multiple PoPs that respond directly to your end users.

By using multiple PoPs, Amazon CloudFront has the inherent ability to help mitigate against both infrastructure and some application layer DDoS attacks by dispersing the traffic across multiple locations. At each of these locations, AWS has multiple Internet connections for capacity and redundancy, which allows Amazon CloudFront to isolate attack traffic while serving content to legitimate end users.

Amazon CloudFront also has filtering capabilities to ensure that only valid TCP connections and HTTP requests are made while dropping invalid requests. This takes the burden of handling invalid traffic (commonly used in UDP floods, SYN floods, and slow reads) off your origin.

To use Amazon CloudFront, you create a distribution and specify your origin (which can be an EC2 instance, S3 bucket, ELB, or custom web server). After you've configured a distribution, Amazon CloudFront will begin responding to end user requests and caching content to improve performance.

The following instructions show how to create an Amazon CloudFront distribution with ELB as the origin (shown in Figure 4). You can configure Amazon CloudFront for different distributions by reviewing Requirements and Recommendations for Using Amazon EC2 and Other Custom Origins.

Step 1: Create a Web Distribution

Step 2: Test Your Web Distribution

Amazon Route 53

One of the most common targets of DDoS attacks is the Domain Name System (DNS). Because DNS is used to locate and translate domain names to IP addresses, attackers frequently see DNS as a single point of failure. For example, your application may actually be available, but if DNS for your application is down end users are not routed correctly. This makes your application effectively unavailable. Because DNS uses UDP and is used to respond to queries, the service is susceptible to reflection and amplification attacks at higher volumes. For these reasons, you'll need resources to increase resiliency for DNS.

Amazon Route 53 is a highly available and scalable DNS service designed to route end users to infrastructure running inside or outside of AWS. Amazon Route 53 makes it possible to manage traffic globally through a variety of routing types, including Latency Based Routing, Geo DNS, and Weighted Round Robin. These routing types can also be combined with Route 53 Failover to enable a low-latency, fault-tolerant architecture.

Amazon Route 53 has two capabilities that work together to help ensure end users can access your application even under DDoS attack: shuffle sharding and anycast routing.

Shuffle Sharding

Shuffle sharding is similar to the concept of database sharding, where horizontal partitions of data are spread across separate database servers to spread load and provide redundancy. Similarly, Amazon Route 53 uses shuffle sharding to spread DNS requests over numerous PoPs, thus providing multiple paths and routes for your application.

Anycast Routing

Anycast routing increases redundancy by advertising the same IP address from multiple PoPs. In the event that a DDoS attack overwhelms one endpoint, shuffle sharding isolate failures while providing additional routes to your infrastructure.

Amazon Route 53 also provides Health Checks with DNS failover to ensure DNS queries use only healthy resources. For example, suppose example.com is hosted on 10 instances with two instances located in different AZ around the world. You can configure Amazon Route 53 to check the health of those instances and to respond to DNS queries for example.com using only healthy instances. You can set up a variety of failover configurations using Amazon Route 53 alias, Weighted Round Robin, Latency Based Routing, Geo DNS, and failover resource record sets.

To use Amazon Route 53 for DNS, you can register or migrate domain names and subdomains to Amazon Route 53. This section will use Amazon CloudFront distribution as the ALIAS record set for Amazon Route 53. You can route traffic to other services by following the steps listed in Routing Queries to AWS Resources.

Step 1: Register a Domain Name and Configure Amazon Route 53 as the DNS Service

Step 2: Route Queries to an Amazon CloudFront Distribution (Public Hosted Zones Only)

Step 3: Create Health Checks and DNS Failover

You can refer to the following links for additional information on using Amazon Route 53.

- Migrating DNS Service for an Existing Domain to Amazon Route 53
- Creating a Subdomain That Uses Amazon Route 53 as the DNS Service without Migrating the Parent Domain
- Migrating DNS Service for a Subdomain to Amazon Route 53 without Migrating the Parent Domain

Safeguard Exposed Resources

In situations where you cannot eliminate Internet entry points to your applications, you'll need to take additional measures to restrict access and protect those entry points without interrupting legitimate end user traffic. Three resources that can provide this control and flexibility are Amazon CloudFront, Amazon Route 53, and web application firewalls (WAFs).

Amazon CloudFront

Amazon CloudFront provides two mechanisms to restrict access to content: *Geo Restriction* and *Origin Access Identity*.

Geo Restriction

Amazon CloudFront supports Geo Restriction, also known as Geo Blocking, which prevents access to your content from specific geographic locations. When an end user requests your content, Amazon CloudFront typically serves it regardless of where the request originated. However, if your end user base is located in specific countries, you can use Geo Restriction to reduce your level of exposure to other countries where you do not normally serve end users. Geo Restriction supports the following scenarios:

- Allow access to content based on a whitelist of approved countries
- Prevent access to content based on a blacklist of banned countries

To use Geo Restriction, you can use the built-in Amazon CloudFront feature or use a third-party geolocation service that provides finer granularity than the country level. To use the built-in Amazon CloudFront feature, you can follow the steps listed in Restrict the Geographic Distributions of Your Content. You can read Using a Third-Party Geolocation Service for information on configuring third-party services.

Origin Access Identity (OAI)

Typically, if you are using an Amazon S3 bucket as the origin for Amazon CloudFront, you grant everyone permission to read the objects in your bucket. This allows anyone to access your content using either the Amazon CloudFront URL or the Amazon S3 URL. Amazon CloudFront does not expose Amazon S3 URLs, but attackers may discover those URLs if your application serves any content directly from Amazon S3 or if anyone gives out direct links.

You can restrict access to Amazon S3 by creating an OAI, which is a special Amazon CloudFront user. You change Amazon S3 permissions to give the OAI permission to Amazon CloudFront and remove all other permissions. When your end users access your Amazon S3 objects using Amazon CloudFront, OAI gets the objects on their behalf. If end users try to access objects using Amazon S3 URLs, they are denied access.

Step 1: Create a CloudFront OAI and Add it to Your Distribution

Step 2: Grant the OAI Permission to Read Objects in Your Amazon S3 Bucket

Step 3: Edit S3 Bucket Permissions

Amazon Route 53

Amazon Route 53 has two features that make it easier to scale your infrastructure and respond to DDoS attacks. These features are *Alias Record Sets* and *Private DNS*.

Alias Record Sets

Unlike ordinary Amazon Route 53 resource record sets, Alias Record Sets provide an Amazon Route 53 – specific extension to DNS functionality. Instead of an IP address or a domain name, an Alias Record Set contains a pointer to an Amazon CloudFront distribution, an ELB load balancer, an Amazon S3 bucket, or another Amazon Route 53 resource record set in the same hosted zone.

Alias Record Sets can save you time and provide additional tools while under attack. For example, suppose an Alias Record Set for example.com points to an ELB load balancer, which is distributing traffic across several EC2 instances running your application. If your application came under attack, you could change the Alias Record Set to point to an Amazon CloudFront distribution or to a different ELB load balancer with higher capacity EC2 instances running WAFs or your own security tools. Amazon Route 53 would then automatically reflect those changes in DNS answers for example.com without any changes to the hosted zone that contains Alias Record Sets for example.com.

Creating an Alias Record Set provides flexibility where you can redirect traffic through additional resources that may not be required all the time but are useful when under attack. For more information on Alias Records Sets refer to Choosing Between Alias and Non-Alias Resource Record Sets.

To create Alias Record Sets, see Create an Alias Record Set by Using Amazon Route 53.

Private DNS

Private DNS allows you to manage internal DNS names for your application resources (web servers, application servers, databases, etc.) without exposing this information to the public Internet. For example, if you have internal resources that you'd like to route using CNAMEs but don't need to expose them to the outside world, you can use Private DNS within VPCs.

You can also use Amazon Route 53 to configure split-view DNS, also known as split-horizon DNS. If you want to maintain internal and external versions of the same website or application (e.g., separating admin and end user traffic), you can configure public and private hosted zones to return different internal and external IP addresses for the same domain name. Just create a public hosted zone and a private hosted zone that have the same domain name, and create the same subdomains in both hosted zones.

Step 1: Create a Private Hosted Zone

Step 2: List Private Hosted Zone

Step 3: Associate Amazon VPCs with a Private Hosted Zone

Web Application Firewall

DDoS attacks that happen at the application layer, commonly target web applications with lower volumes of traffic compared to infrastructure attacks. To mitigate these types of attacks, you'll want to include a WAF as part of your infrastructure.

WAFs act as filters that apply a set of rules to web traffic. Generally, these rules cover exploits like cross-site scripting (XSS) and SQL injection (SQLi) but can also help build resiliency against DDoS by mitigating HTTP GET or POST floods.

HTTP works as a request-response protocol between end users and applications where end users request data (GET) and submit data to be processed (POST). GET floods work by requesting the same URL at a high rate or requesting all objects from your application. POST floods work by finding expensive application processes, i.e., logins or database searches, and triggering those process to overwhelm your application.

WAFs have several features that may prevent these types of attacks from affecting your application availability. One feature is HTTP rate limiting where you can establish a threshold of supported HTTP requests per end user for a certain time period. Once an end user has exceeded that threshold, WAFs can block or buffer new requests to ensure other end users have access to your application.

WAFs can also inspect HTTP requests and identify those that don't confirm to normal patterns, e.g., prevent logins that are over character limits, prevent searches for all items, etc. Other WAF features that can help with application layer DDoS are Completely Automated Public Turning test to tell Computers and Humans Apart (CAPTCHA) tests and IP reputation lists.

To deploy a WAF in your AWS infrastructure, first you'll need to select from one of the available WAF solutions in the AWS Marketplace. The AWS Marketplace is an online store that helps AWS customers find, buy, and install software to run on EC2 instances. WAF solutions can be found under the AWS Marketplace Security Category or by searching for Web Application Firewall.

Once you have selected your WAF solution, you'll need to deploy the software on an EC2 instance and configure the instance to scale with your traffic. Before doing that, let's discuss the additional requirements for scaling AWS Marketplace WAFs.

In order to inspect all HTTP requests, WAFs sit in-line with your application traffic. Unfortunately, this creates a scenario where WAFs can become a point of failure or bottleneck. To mitigate this problem, you'll need the ability to run multiple WAFs on demand during traffic spikes. This type of scaling for WAF is done via a "WAF sandwich."

In the "WAF sandwich," the EC2 instance running your WAF software is included in an Auto Scaling group and placed in between two ELB load balancers. Recall from the Elastic Load Balancing section, you created two load balancers: a basic load balancer in the default VPC, and an internal load balancer. The basic load balancer in your default

VPC will be the frontend, public facing load balancer that will distribute all incoming traffic to the WAF EC2 instance. By running the WAF EC2 instance in an Auto Scaling group behind ELB, the instance can scale and add additional WAF EC2 instances should the traffic spike to elevated levels.

Once the traffic has been inspected and filtered, the WAF EC2 instance forwards traffic to the internal, backend load balancer which then distributes traffic across your application EC2 instance. This configuration (shown in Figure 5) allows the WAF EC2 instances to scale and meet capacity demands without affecting the availability of your application EC2 instance.

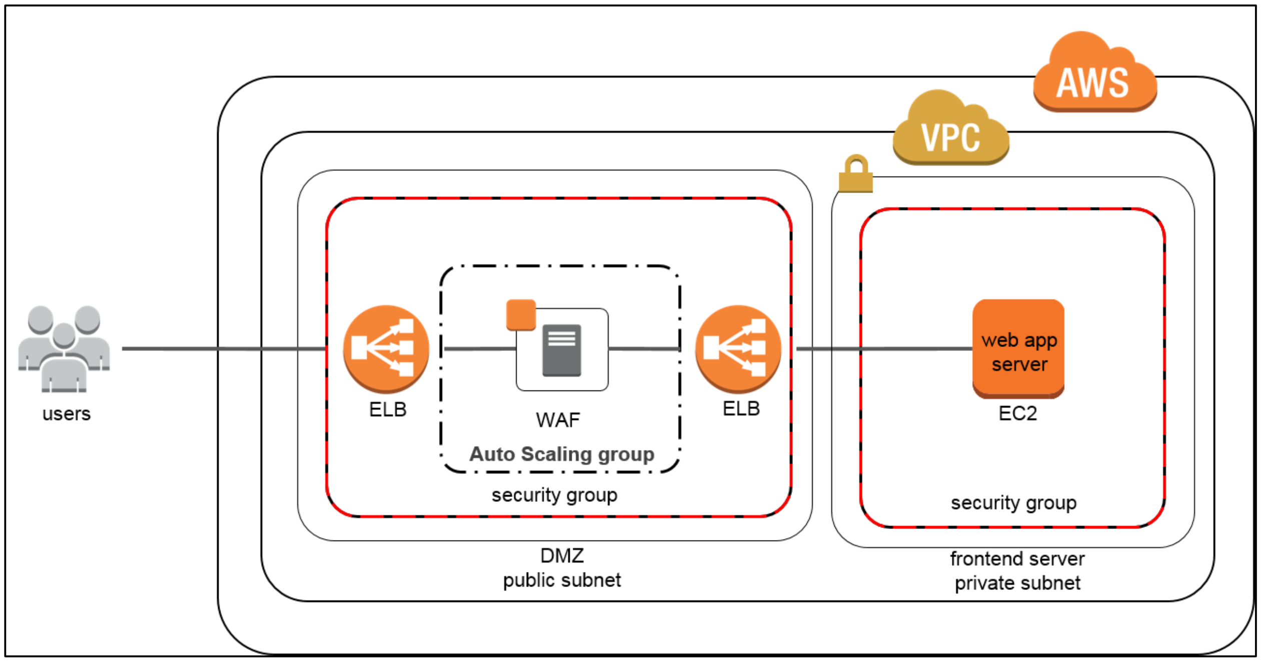

Figure 5: "WAF Sandwich"

Since the basic and internal ELB load balancers have already been created, the next step is to deploy and configure your WAF on an EC2 instance in between the two ELB load balancers. AWS Marketplace solutions employ different configurations to support Auto Scaling for the "WAF sandwich." After you have selected a WAF from the AWS Marketplace, you should contact the AWS partner and request documentation or assistance with deploying and scaling their software. You can also contact AWS Professional Services for help deploying WAF in your infrastructure.

Learn Normal Behavior

One way to ensure a resilient architecture is to know when your application or infrastructure is under attack. Frequently, customers determine the status of an attack by whether their application is "up or down." However, a better strategy is to understand the levels and patterns of traffic that are expected for your application and use that as a benchmark for identifying abnormal levels or patterns.

DDoS attackers typically probe or test your application in an attempt to determine thresholds. Once they have completed several probing attacks, attackers can estimate the amount of traffic or vectors that can be used to successfully impact the availability of your application. In these circumstances, knowing what you expect and what you don't expect will help you to spot and alert on anomalies to help establish situational awareness.

Amazon CloudWatch

You can use Amazon CloudWatch to monitor your infrastructure and applications running in AWS. Amazon CloudWatch can collect metrics, log files, and set alarms for when these metrics have passed predetermined thresholds. You can also use Amazon CloudWatch to gain system-wide visibility into resource utilization, application performance, and operational health if you're unsure what a normal day should look like. You can use these insights to react to DDoS attacks and evaluate when changes are needed.

Before creating alerts, you'll want to become familiar with how your application typically performs. You can do this by viewing, selecting, and graphing the Amazon CloudWatch metrics.

Step 1: View Available Metrics

Step 2: Search for Available Metrics

Step 3: Select Metrics

Step 4: Get Statistics for a Metric

Step 5: Graph Metrics

Amazon CloudFront reports and Amazon Route 53 Health Checks have separate metrics you can review to better understand your traffic and application.

Once you have established a baseline for your application, you can use Amazon CloudWatch to create alarms on theses metrics and alert you to potential attacks or other irregularities. The following table contains recommended alert metrics for DDoS attacks.

Topic	Metric	Description
Auto Scaling	GroupMaxSize	The maximum size of the Auto Scaling group.
AWS Billing	EstimatedCharges	The estimated charges for your AWS usage.
Amazon CloudFront	Requests	The number of requests for all HTTP/S requests.
Amazon CloudFront	TotalErrorRate	The percentage of all requests for which the HTTP status code is 4xx or 5xx.
Amazon EC2	CPUUtilization	The percentage of allocated EC2 compute units that are currently in use.
Amazon EC2	NetworkIn	The number of bytes received on all network interfaces by the instance.
Amazon EC2	StatusCheckFailed	A combination of of StatusCheckFailed_Instance and StatusCheckFailed_System that reports if either of the status checks has failed.

ELB	UnHealthyHostCount	The number of unhealthy instances in each Availability Zone.
ELB	RequestCount	The number of completed requests that were received and routed to registered instances.
ELB	Latency	The time elapsed, in seconds, after the request leaves the load balancer until a response is received.
ELB	HTTPCode_ELB_4xx HTTPCode_ELB_5xx	The number of HTTP 4XX or 5XX error codes generated by the load balancer.
ELB	BackendConnectionErrors	The number of connections that were not successfully.
ELB	SpilloverCount	The number of requests that were rejected because the queue was full.
Amazon Route 53	HealthCheckStatus	The status of the health check endpoint.

Table 1: Recommended CloudWatch Metrics

In addition to these default metrics, you can use Amazon CloudWatch Logs to monitor and access log files from your applications running on EC2 instances. Amazon CloudWatch Logs is an installable agent available for Amazon Linux, Ubuntu, and Windows that sends logs to Amazon CloudWatch. Amazon CloudWatch Logs can track the number of errors that occur in your application logs and send you a notification whenever the rate of errors exceeds a threshold you specify.

You can also monitor application logs for specific literal terms (such as "NullReferenceException") or count the number of occurrences of a literal term at a particular position in log data (such as "404" status codes in an Apache access log). When you find the desired term, Amazon CloudWatch Logs reports the data to an Amazon CloudWatch metric that you specify.

This section describes the steps for installing the Amazon CloudWatch Logs agent on an Amazon EC2 instance running Amazon Linux or Ubuntu Server. To get started with Amazon CloudWatch Logs on an Amazon EC2 instance running Windows, see Sending Performance Counters to CloudWatch and Logs to CloudWatch Logs in the *Amazon EC2 User Guide for Microsoft Windows Instances*.

Step 1: Quick Start: Install and Configure the CloudWatch Logs Agent on an Existing EC2 Instance

Step 2: Set Up Amazon Simple Notification Service

Step 3: Create an Alarm

For more information on available alarms, see Amazon CloudWatch Namespaces, Dimensions, and Metrics Reference.

Create a Plan for Attacks

Developing a response strategy while under attack is unlikely to be effective. Having a plan in place before an attack ensures that:

- You've validated the architecture and selected the techniques that work for your infrastructure and application.
- You've evaluated the costs for increased resiliency and understand the goals of your defense.
- You know who to contact when an attack happens.

As part of creating this plan, you should consider the level of support that you will require. AWS provides a highly personalized level of support for all customers. However, for customers looking for a higher level of support during DDoS attacks, you should consider the Enterprise tier support based on the following benefits:

- Technical Account Manager (TAM): a TAM provides technical expertise for the full range of AWS services and obtains a detailed understanding of your architecture. TAMs work with AWS Solution Architects to help you follow best practices and provide a direct telephone for primary point of contact for ongoing support needs.
- White-Glove Case Routing: cases are routed directly to specially trained engineers to help ensure fast, accurate resolution of critical issues.

For more information about these support features and different levels of support, visit the AWS Support Center.

Conclusion

AWS provides a number of services and features that you can use to improve your resiliency against DDoS attacks. You are responsible for using these and other measures, as appropriate, to protect your infrastructure and application in the cloud and meet your requirements for DDoS protection. AWS encourages you to build a set of security policies and responses using the best practices from this paper which can help improve your resiliency against DDoS attacks.