# AWS Best Practices for DDoS Resiliency

*June 2018*

© 2018, Amazon Web Services, Inc. or its affiliates.  All rights reserved.

# Notices

This document is provided for informational purposes only. It represents AWS's current product offerings and practices as of the date of issue of this document, which are subject to change without notice. Customers are responsible for making their own independent assessment of the information in this document and any use of AWS's products or services, each of which is provided "as is" without warranty of any kind, whether express or implied. This document does not create any warranties, representations, contractual commitments, conditions or assurances from AWS, its affiliates, suppliers or licensors. The responsibilities and liabilities of AWS to its customers are controlled by AWS agreements, and this document is not part of, nor does it modify, any agreement between AWS and its customers.

# Contents


Introduction 1

Denial of Service Attacks 2

- Infrastructure Layer Attacks 3
- Application Layer Attacks 5

Mitigation Techniques 7

- Infrastructure Layer Defense (BP1, BP3, BP6, BP7) 10
- Application Layer Defense (BP1, BP2, BP6) 14

Attack Surface Reduction 16

- Obfuscating AWS Resources (BP1, BP4, BP5) 16

Operational Techniques 19

- Visibility 19
- Support 22

Conclusion 24

Contributors 24

Document Revisions 24

Appendix A: Additional Resources 26

# Abstract

This paper is intended for customers who want to improve the resiliency of their applications running on Amazon Web Services (AWS) against Distributed Denial of Service (DDoS) attacks. It provides an overview of DDoS attacks, capabilities provided by AWS, mitigation techniques, and a DDoS-resilient reference architecture that can be used as a guide to help protect application availability.

The paper is intended for IT decision makers and security engineers who are familiar with the basic concepts of networking, security, and AWS. Each section has links to AWS documentation that provides more detail on the best practice or capability.

# Introduction

You work to protect your business from the impact of Distributed Denial of Service (DDoS) attacks, as well as other cyberattacks. You want to keep your customers' trust in your service by maintaining the availability and responsiveness of your application. And you want to avoid unnecessary direct costs when your infrastructure must scale in response to an attack.

AWS is committed to providing you with tools, best practices, and services to help ensure high availability, security, and resiliency to defend against bad actors on the internet.

In this whitepaper, we provide you with prescriptive DDoS guidance. We describe different attack types, such as infrastructure layer attacks and application layer attacks, and explain which best practices are most effective to manage each attack type. We also outline the services and features that fit into a DDoS mitigation strategy, and how each one can be used to help protect your applications.

# Denial of Service Attacks

A Denial of Service (DoS) attack is a deliberate attempt to make your website or application unavailable to users, such as by flooding it with network traffic. To achieve this, attackers use a variety of techniques that consume large amounts of network bandwidth or tie up other system resources, disrupting access for legitimate users. In its simplest form, a lone attacker uses a single source to execute a DoS attack against a target, as shown in the following diagram (Figure 1).

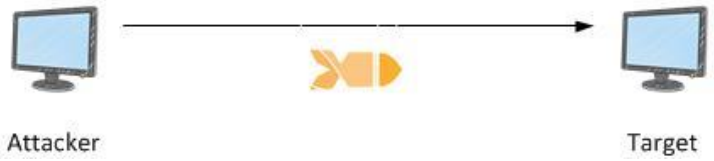


**Figure 1: Diagram of DOS attack**

But in a Distributed Denial of Service (DDoS) attack, an attacker uses multiple sources – which may be distributed groups of malware infected computers, routers, IoT devices, and other endpoints– to orchestrate an attack against a target. As illustrated in the following DDoS attack diagram (Figure 2), a network of compromised hosts participates in the attack, generating a flood of packets or requests to overwhelm the target.

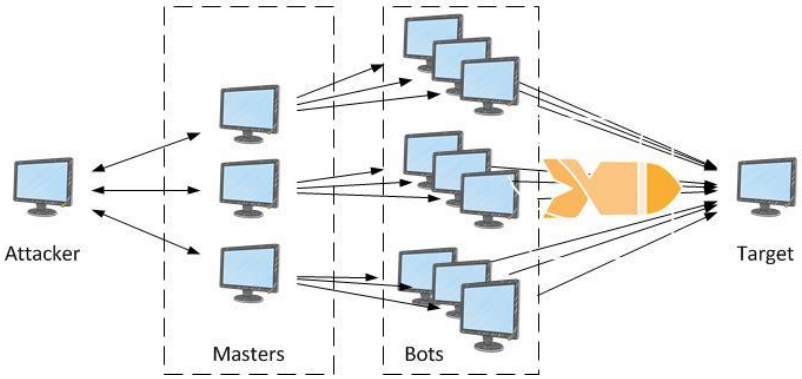


**Figure 2: Diagram of DDoS attack**

DDoS attacks are most common at layers 3, 4, 6, and 7 of the Open Systems Interconnection (OSI) model, which is described in following table (Table 1). Layer 3 and 4 attacks correspond to the Network and Transport layers of the OSI model. We'll refer to these collectively as *infrastructure layer attacks*. Layer 6 and 7 attacks correspond to the Presentation and

Application layers of the OSI model. We'll address these together as *application layer attacks*. We describe examples of these attack types in the next sections.

| # | Layer | Unit | Description | Vector Examples |
|---|---|---|---|---|
| **7** | Application | Data | Network process to application | HTTP floods, DNS query floods |
| **6** | Presentation | Data | Data representation and encryption | TLS abuse |
| ***5*** | Session | Data | Interhost communication | N/A |
| **4** | Transport | Segments | End-to-end connections and reliability | SYN floods |
| **3** | Network | Packets | Path determination and logical addressing | UDP reflection attacks |
| ***2*** | Data Link | Frames | Physical addressing | N/A |
| ***1*** | Physical | Bits | Media, signal, and binary transmission | N/A |

**Table 1: Open Systems Interconnection (OSI) Model.**

# Infrastructure Layer Attacks

The most common DDoS attacks, UDP reflection attacks and SYN floods, are infrastructure layer attacks. An attacker can use either of these methods to generate large volumes of traffic that can inundate the capacity of a network or tie up resources on systems like a server, firewall, IPS, or load balancer. While these attacks can be easy to identify, to effectively mitigate them, you must have a network or systems that scale up capacity more rapidly than the inbound traffic flood. This extra capacity is to either filter out or absorb the attack traffic enabling your system and application to respond to your legitimate customer traffic.

**UDP Reflection Attacks**

UDP reflection attacks exploit the fact that UDP is a stateless protocol. Attackers can craft a valid UDP request packet listing the attack target's IP as the UDP source IP address. The attacker has now falsified, or "spoofed", the UDP request packet's source IP. An attacker then sends the UDP packet containing the spoofed source IP to an intermediate server. The server is tricked into sending its UDP response packets to the targeted victim IP rather than back to the

attacker's IP address. The intermediate server is used because it generates a response that is several times larger than the request packet effectively amplifying the amount of attack traffic sent to the target IP address.

The amplification factor, which is the ratio of response size to request size, varies depending on which protocol the attacker uses: DNS, NTP, or SSDP. For example, the amplification factor for DNS can be 28 to 54 times the original number of bytes. So if an attacker sends a request payload of 64 bytes to a DNS server, they can generate over 3400 bytes of unwanted traffic to an attack target. The reflection tactic and amplification effect are illustrated in the following diagram (Figure 3).

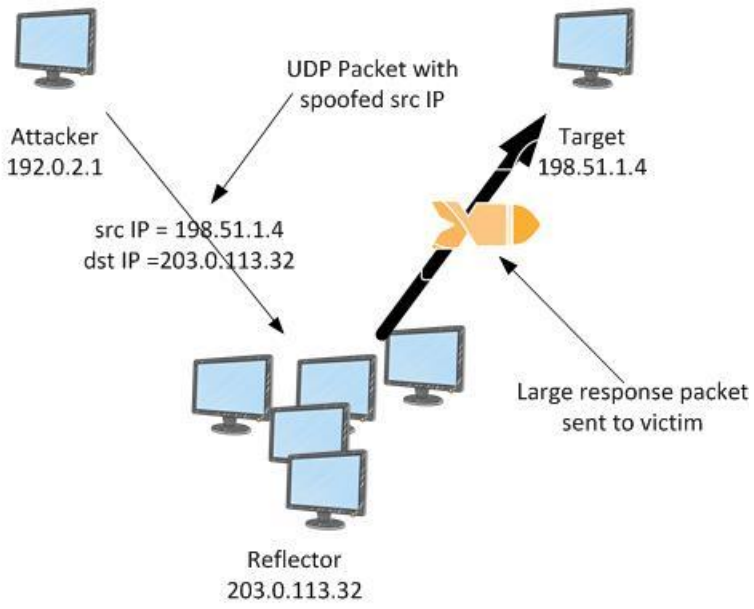


**Figure 3: UDP reflection attack**

**SYN Flood Attacks**

When a user connects to a TCP service, like a web server, their client sends a SYN (synchronization) packet. The server returns a SYN-ACK packet in acknowledgement, and, finally, the client responds with an ACK packet, which completes the expected three-way handshake. This typical handshake is illustrated in the following diagram (Figure 4):

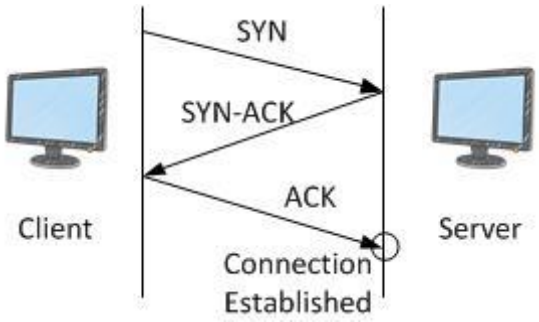


Figure 4: SYN 3-way handshake

In a SYN flood attack, a malicious client sends a large number of SYN packets, but never sends the final ACK packets to complete the handshakes. The server is left waiting for a response to the half-open TCP connections and eventually runs out of capacity to accept new TCP connections. This can prevent new users from connecting to the server. SYN floods can reach up to 100s of Gbps, but the attack is not about SYN traffic volume but rather tying up available server connections resulting in no resources for legitimate connections.

## Application Layer Attacks

An attacker may target the application itself by using a layer 7 or application layer attack. In these attacks, similar to SYN flood infrastructure attacks, the attacker attempts to overload specific functions of an application to make the application unavailable or extremely unresponsive to legitimate users. Sometimes this can be achieved with very low request volumes that generate only a small volume of network traffic. This can make the attack difficult to detect and mitigate. Examples of application layer attacks include HTTP floods, cache-busting attacks, and WordPress XML-RPC floods.

In an **HTTP flood attack**, an attacker sends HTTP requests that appear to be from a real user of the web application. Some HTTP floods target a specific resource, while more complex HTTP floods attempt to emulate human interaction with the application. This can increase the difficulty of using common mitigation techniques like request rate limiting.

**Cache-busting attacks** are a type of HTTP flood that uses variations in the query string to circumvent content delivery network (CDN) caching. Instead of being able to return cached results, the CDN must contact the origin server for every page request, and these origin fetches cause additional strain on the application web server.

With a **WordPress XML-RPC flood attack**, also known as a WordPress pingback flood, an attacker misuses the XML-RPC API function of a website hosted on the WordPress content management software to generate a flood of HTTP requests. The pingback feature allows a web site hosted on WordPress (Site A) to notify a different WordPress site (Site B) that Site A has created a link to Site B. Site B then attempts to fetch Site A to verify the existence of the link. In a pingback flood, the attacker misuses this capability to cause Site B to attack Site A. This type of attack has a clear signature: "WordPress" is typically present in the "User-Agent" of the HTTP request header.

There are also other forms of malicious traffic that can impact an application's availability. Scraper bots automate attempts to access a web application to steal content or record competitive information, like pricing. Brute force and credential stuffing attacks are programmed efforts to gain unauthorized access to secure areas of an application. These are not strictly DDoS attacks; but their automated nature can look similar to a DDoS attack and they can be mitigated by implementing some of the same best practices to be covered later in this paper.

Application layer attacks can also target domain name system (DNS) services. The most common of these attacks is a DNS query flood in which an attacker uses many well-formed DNS queries to exhaust the resources of a DNS server. These attacks can also include a cache-busting component where the attacker randomizes the subdomain string to bypass the local DNS cache of any given resolver. As a result, the resolver can't take advantage of cached domain queries and must instead repeatedly contact the authoritative DNS server, which amplifies the attack.

If a web application is delivered over TLS, an attacker can also choose to attack the TLS negotiation process. TLS is computationally expensive so an attacker can reduce a server's availability by sending unintelligible data. In a variation of this attack, an attacker completes the TLS handshake but perpetually renegotiates the encryption method. Or an attacker can attempt to exhaust server resources by opening and closing many TLS sessions.

# Mitigation Techniques

Some forms of DDoS mitigation are included automatically with AWS services. You can further improve your DDoS resilience by using an AWS architecture with specific services and by implementing additional best practices.

All AWS customers benefit from the automatic protections of AWS Shield Standard, at no additional charge. AWS Shield Standard defends against most common, frequently occurring network and transport layer DDoS attacks that target your web site or applications. This is offered on all AWS services and in every AWS Region, at no additional cost. In AWS Regions, DDoS attacks are detected by a system that automatically baselines traffic, identifies anomalies, and, as necessary, creates mitigations. This mitigation system provides protection against many common infrastructure layer attacks. You can use AWS Shield Standard as part of a DDoS-resilient architecture to protect both web and non-web applications.

Additionally, you can leverage AWS services that operate from edge locations, like Amazon CloudFront and Amazon Route 53, to build comprehensive availability protection against all known infrastructure layer attacks. Using these services – part of the AWS Global Edge Network – can improve the DDoS resilience of your application when you serve web application traffic from edge locations distributed around the world.

Several specific benefits of using Amazon CloudFront and Amazon Route 53 include the following:

- AWS Shield DDoS mitigation systems that are integrated with AWS edge services, reducing time-to-mitigate from minutes to sub-second.
- Stateless SYN Flood mitigation techniques that proxy and verify incoming connections before passing them to the protected service.
- Automatic traffic engineering systems that can disperse or isolate the impact of large volumetric DDoS attacks.
- Application layer defense when combined with AWS WAF that does not require changing your current application architecture (e.g. in an AWS Region or on-premises datacenter).

There is no charge for inbound data transfer on AWS and you do not pay for DDoS attack traffic that is mitigated by AWS Shield.

For a DDoS-resilient reference architecture that includes AWS Global Edge Network services, see the following diagram (Figure 5).

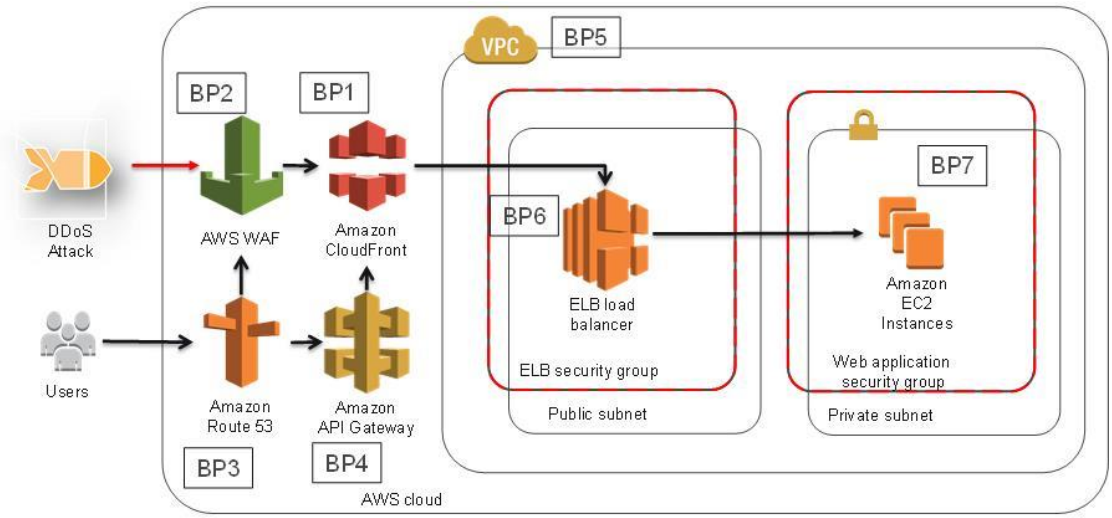


**Figure 5: DDoS-resilient reference architecture.**

This reference architecture includes several AWS services that can help you improve your web application's resiliency against DDoS attacks. For a summary of these services and the capabilities that they can provide, see the following table (Table 2). We've tagged each service with a "best practice indicator" (BP1, BP2, etc.) so that you can easily refer back to each of them here as you read the rest of this document. For example, an upcoming section discusses the capabilities provided by Amazon CloudFront and includes the best practices indicator BP1.

| | **AWS Edge Locations** | | **AWS Regions** | | | |
|---|---|---|---|---|---|---|
| | Amazon CloudFront (BP1) with AWS WAF (BP2) | Amazon Route 53 (BP3) | Elastic Load Balancing (BP6) | Amazon API Gateway (BP4) | Amazon VPC (BP5) | Amazon EC2 with Auto Scaling (BP7) |
| **Layer 3 (e.g. UDP reflection) attack mitigation** | ✓ | ✓ | ✓ | ✓ | ✓ | ✓ |

| | AWS Edge Locations | | AWS Regions | | | |
|---|---|---|---|---|---|---|
| **Layer 4 (e.g. SYN flood) attack mitigation** | ✓ | ✓ | ✓ | ✓ | | |
| **Layer 6 (e.g. TLS) attack mitigation** | ✓ | | ✓ | ✓ | | |
| **Reduce attack surface** | ✓ | ✓ | ✓ | ✓ | ✓ | |
| **Scale to absorb application layer traffic** | ✓ | ✓ | ✓ | ✓ | | ✓ |
| **Layer 7 (application layer) attack mitigation** | ✓ | ✓ | ✓ (if used with AWS WAF) | | | |
| **Geographic isolation and dispersion of excess traffic, and larger DDoS attacks** | ✓ | ✓ | | | | |

**Table 2: Summary of best practices.**

Another way that you can improve your readiness to respond to and mitigate DDoS attacks is by subscribing to AWS Shield Advanced. This optional DDoS mitigation service helps you protect an application hosted on any AWS Region or hosted outside of AWS. The service is available globally for Amazon CloudFront and Amazon Route 53. It's also available in select AWS Regions for Classic Load Balancer (CLB), Application Load Balancer (ALB), and Elastic IP Addresses (EIPs). Using AWS Shield Advanced with EIPs allows you to protect Network Load Balancer (NLBs) or Amazon EC2 instances.

With AWS Shield Advanced, you get the following additional benefits:

- Access to the AWS DDoS Response Team (DRT) for assistance in mitigating DDoS attacks that impact application availability.

- DDoS attack visibility by using the AWS Management Console, API, and Amazon CloudWatch metrics and alarms.
- Access to the Global Threat Environment dashboard, which provides an overview of DDoS attacks observed and mitigated by AWS.
- Access to AWS WAF, at no additional cost, for the mitigation of application layer DDoS attacks (when used with Amazon CloudFront or ALB).
- Automatic baselining of web traffic attributes, when used with AWS WAF.
- Access to AWS Firewall Manager, at no additional cost, for automated policy enforcement. This service lets security administrators centrally control and manage AWS WAF rules.
- Sensitive detection thresholds which routes traffic into DDoS mitigation system earlier and can improve time-to-mitigate attacks against Amazon EC2 or NLB, when used with EIP.
- Cost Protection that allows you to request a limited refund of scaling-related costs that result from a DDoS attack.
- Enhanced Service Level Agreement that is specific to AWS Shield Advanced customers.

For a complete list of AWS Shield Advanced features and to learn more about AWS Shield, see AWS Shield – Managed DDoS Protection.

In the following sections, we'll describe each of the recommended best practices for DDoS mitigation in more depth. For a quick and easy-to-implement guide on building a DDoS mitigation layer for static or dynamic web applications, see How to Help Protect Dynamic Web Applications Against DDoS Attacks by Using Amazon CloudFront and Amazon Route 53.

## Infrastructure Layer Defense (BP1, BP3, BP6, BP7)

In a traditional datacenter environment, you can mitigate infrastructure layer DDoS attacks by using techniques like overprovisioning capacity, deploying DDoS mitigation systems, or scrubbing traffic with the help of DDoS mitigation services. On AWS, DDoS mitigation capabilities are automatically provided—but you can optimize your application's DDoS-resilience by making architecture choices that best leverage those capabilities and also allow you to scale for excess traffic.

Key considerations to help mitigate volumetric DDoS attacks include ensuring that enough transit capacity and diversity is available, and protecting your AWS resources, like Amazon EC2 instances, against attack traffic.

## Instance Size (BP7)

Amazon EC2 provides resizable compute capacity so that you can quickly scale up or down as your requirements change. You can scale horizontally by adding instances to your application, and scale vertically by using larger instances. Some Amazon EC2 instance types support features that can more easily handle large volumes of traffic, for example, 25 Gigabit network interfaces and Enhanced Networking.

With 25 Gigabit network interfaces, each instance can support a larger volume of traffic. This helps prevent interface congestion for traffic that has reached the Amazon EC2 instance. Instances that support Enhanced Networking provide higher I/O performance and lower CPU utilization compared to traditional implementations. This improves the ability of the instance to handle traffic with larger packet volumes.

The 25 Gigabit feature is available on the largest sizes of instances, for example, M4. To learn more about Amazon EC2 instances that support 25 Gigabit network interfaces and Enhanced Networking, see Amazon EC2 Instance Types. To learn how to enable Enhanced Networking, see Enabling Enhanced Networking on Linux Instances in a VPC.

## Choice of Region (BP7)

AWS services are available in multiple locations worldwide. These geographically-separate areas are called regions. When architecting your application, you can choose one or more regions based on your requirements. Common considerations include performance, cost, and data sovereignty. In each region, AWS provides access to a unique set of internet connections and peering relationships to provide optimal latency and throughput to users in those areas.

It is also important to consider DDoS resiliency when you choose the regions for your application. Many regions are close to internet exchanges so they have more connectivity to major networks. Being close to exchanges where international carriers and large peers have a strong presence can help give you the internet capacity to mitigate larger attacks.

To learn more about choosing a region, see Regions and Availability Zones. You can also ask your account team about the characteristics of each region that you're considering, to help you make an informed decision.

## Load Balancing (BP6)

Large DDoS attacks can overwhelm the capacity of a single Amazon EC2 instance, so adding load balancing can help your resiliency. There are several options that you can choose from to help mitigate an attack by load balancing excess traffic. With Elastic Load Balancing (ELB), you can reduce the risk of overloading your application by distributing traffic across many backend instances. ELB can scale automatically, allowing you to manage larger volumes when you have unanticipated extra traffic, for example, due to flash crowds or DDoS attacks. For applications built within Amazon VPC, there are two types of ELBs to consider, depending on your application type: Application Load Balancer (ALB) or Network Load Balancer (NLB).

**For web applications**, you can use ALB to route traffic based on its content and accept only well-formed web requests. This means that many common DDoS attacks, like SYN floods or UDP reflection attacks, will be blocked by ALB, protecting your application from the attack. When ALB detects these types of attacks, it automatically scales to absorb the additional traffic. To learn more about protecting web applications with ALB, see Getting Started with Application Load Balancers.

**For TCP-based applications**, you can use NLB to route traffic to Amazon EC2 instances at ultra-low latency. When you create an NLB, a network interface is created for each Availability Zone (AZ) that you enable. You have the option to assign an Elastic IP (EIP) address per subnet enabled for the load balancer. One key consideration with NLB is that any traffic that reaches the load balancer on a valid listener will be routed to your Amazon EC2 instances, not absorbed. To learn more about protecting TCP applications with NLB, see Getting Started with Network Load Balancers.

## Deliver at Scale Using AWS Edge Locations (BP1, BP3)

Access to highly-scaled, diverse internet connections can significantly increase your ability to optimize latency and throughput to users, to absorb DDoS attacks, and to isolate faults while minimizing the impact on your application's availability. AWS edge locations provide an additional layer of network infrastructure that provides these benefits to any web application that uses Amazon CloudFront and Amazon Route 53. When you use these services, your content is served and DNS queries are resolved from locations that are often closer to your users.

### *Web Application Delivery at the Edge (BP1)*

Amazon CloudFront is a service that can be used to deliver your entire website, including static, dynamic, streaming, and interactive content. Persistent TCP connections and variable time-to-live (TTL) settings can be used to offload traffic from your origin, even if you are not

serving cacheable content. These features mean that using Amazon CloudFront reduces the number of requests and TCP connections back to your origin which helps protect your web application from HTTP floods. Amazon CloudFront only accepts well-formed connections, which helps prevent many common DDoS attacks, like SYN floods and UDP reflection attacks, from reaching your origin. DDoS attacks are also geographically isolated close to the source which prevents the traffic from impacting other locations. These capabilities can greatly improve your ability to continue serving traffic to users during large DDoS attacks. You can use Amazon CloudFront to protect an origin on AWS or elsewhere on the internet.

If you're using Amazon S3 to serve static content on the internet, you should use Amazon CloudFront to protect your bucket. You can use Origin Access Identify (OAI) to ensure that users only access your objects by using CloudFront URLs. To learn more about OAI, see Using an Origin Access Identity to Restrict Access to Your Amazon S3 Content.

To learn more about protecting and optimizing the performance of web applications using Amazon CloudFront, see Getting Started with CloudFront.

### *Domain Name Resolution at the Edge (BP3)*

Amazon Route 53 is a highly available and scalable domain name system (DNS) service that can be used to direct traffic to your web application. It includes advanced features like Traffic Flow, Latency Based Routing, Geo DNS, and Health Checks and Monitoring that allow you to control how the service responds to DNS requests, to improve the performance of your web application and to avoid site outages.

Amazon Route 53 uses techniques like shuffle sharding and anycast striping, that can help users access your application even if the DNS service is targeted by a DDoS attack. With shuffle sharding, each name server in your delegation set corresponds to a unique set of edge locations and internet paths. This provides greater fault tolerance and minimizes overlap between customers. If one name server in the delegation set is unavailable, users can retry and receive a response from another name server at a different edge location. Anycast striping allows each DNS request to be served by the most optimal location, spreading the network load and reducing DNS latency. In turn, this provides a faster response for users. Additionally, Amazon Route 53 can detect anomalies in the source and volume of DNS queries, and prioritize requests from users that are known to be reliable.

To learn more about using Amazon Route 53 to route users to your application, see Getting Started with Amazon Route 53.

# Application Layer Defense (BP1, BP2, BP6)

Many of the techniques discussed in this paper are effective at mitigating the impact that infrastructure layer DDoS attacks have on your application's availability. To also defend against application layer attacks requires you to implement an architecture that allows you to specifically detect, scale to absorb, and block malicious requests. This is an important consideration because network-based DDoS mitigation systems are generally ineffective at mitigating complex application layer attacks.

## Detect and Filter Malicious Web Requests (BP1, BP2)

When your application runs on AWS, you can leverage both Amazon CloudFront and AWS WAF to help defend against application layer DDoS attacks.

Amazon CloudFront allows you to cache static content and serve it from AWS edge locations, which can help reduce the load on your origin. It can also help reduce server load by preventing non-web traffic from reaching your origin. Additionally, CloudFront can automatically close connections from slow-reading or slow-writing attackers (e.g. Slowloris).

By using AWS WAF, you can configure web access control lists (Web ACLs) on your CloudFront distributions or Application Load Balancers to filter and block requests based on request signatures. Each Web ACL consists of rules that you can configure to string match or regex match one or more request attributes, such as the URI, query string, HTTP method, or header key. In addition, by using AWS WAF's rate-based rules, you can automatically block the IP addresses of bad actors when requests matching a rule exceed a threshold that you define. Requests from offending client IP addresses will receive "403 Forbidden" error responses and will remained blocked until request rates drop below the threshold. This is useful for mitigating HTTP flood attacks that are disguised as regular web traffic.

To block attacks from known bad-acting IP addresses, you can create rules using IP match conditions or use Managed Rules for AWS WAF offered by sellers in the AWS Marketplace that will block specific malicious IP addresses that are included in IP reputation lists. Both AWS WAF and Amazon CloudFront also allow you to set geo-restrictions to block or whitelist requests from selected countries. This can help block attacks originating from geographic locations where you do not expect to serve users.

To help identify malicious requests, you can review your web server logs or use WAF's Sampled Requests feature which provides details about requests that have matched one of your AWS WAF rules for a period of time in the past 3 hours. You can use this information to identify potentially malicious traffic signatures and create a new rule to deny those requests. If you see a number of requests with a random query string, you might decide to disable query

string forwarding in Amazon CloudFront. This can be helpful in mitigating a cache-busting attack against your origin.

If you are subscribed to AWS Shield Advanced, you can engage the AWS DDoS Response Team (DRT) to help you create rules to mitigate an attack that is hurting your application's availability. DRT can only gain limited access to your account, and only with your explicit authorization. For more information, see the Support section later in this document.

Using AWS Firewall Manager can help simplify managing AWS WAF rules across your organization. By using AWS Firewall Manager, you can enable AWS WAF across many accounts and resources, including creating rules that are automatically applied to existing or new accounts in your organization.

To learn more about using geo restriction to limit access to your Amazon CloudFront distribution, see Restricting the Geographic Distribution of Your Content.

To learn more about using AWS WAF, see Getting Started with AWS WAF and Viewing a Sample of the Web Requests that CloudFront has Forwarded to AWS WAF.

To learn more about configuring rate-based rules, see Protect Web Sites & Services Using Rate-Based Rules for AWS WAF.

To learn how to manage the deployment of AWS WAF rules across your AWS resources with AWS Firewall Manager, see Getting Started with AWS Firewall Manager.

## Scale to Absorb (BP6)

Another way to mitigate application layer attacks is to operate at scale. If you have web applications, you can use ELB to distribute traffic to a number of Amazon EC2 instances that are overprovisioned or configured to automatically scale. These instances can handle sudden traffic surges that occur for any reason, including a flash crowd or an application layer DDoS attack. You can set Amazon CloudWatch alarms to initiate Auto Scaling to automatically scale the size of your Amazon EC2 fleet in response to events that you define. This protects application availability when there's an unexpected increase in request volume. If you use Amazon CloudFront or Application Load Balancer (ALB) with your application, TLS negotiation is handled by the distribution or the load balancer. This helps protect your instances from being impacted by TLS-based attacks by scaling to handle legitimate requests as well as TLS-abuse attacks.

To learn more about using Amazon CloudWatch to invoke Auto Scaling, see Monitoring Your Auto Scaling Instances and Groups Using Amazon CloudWatch.

# Attack Surface Reduction

Another important consideration when architecting an AWS solution is to limit the opportunities an attacker has for targeting your application. For example, if you do not expect users to directly interact with certain resources, you can make sure that those resources are not accessible from the internet. Similarly, if you do not expect users or external applications to communicate with your application on certain ports or protocols, you can make sure that that traffic is not accepted.

This concept is known as *attack surface reduction*. In this section, we provide best practices to help you reduce your attack surface and limit your application's internet exposure. Resources that are not exposed to the internet are more difficult to attack, which limits the options an attacker has to target your application's availability.

## Obfuscating AWS Resources (BP1, BP4, BP5)

Typically, users can quickly and easily use an application without requiring that AWS resources be fully exposed to the internet. For example, when you have Amazon EC2 instances behind an ELB, the instances themselves might not need to be publicly accessible. Instead, you could provide users with access to the ELB on certain TCP ports and allow only the ELB to communicate with the instances. You can set this up by configuring Security Groups and Network Access Control Lists (NACLs) within your Amazon Virtual Private Cloud (VPC). Amazon VPC allows you to provision a logically isolated section of the AWS cloud where you can launch AWS resources in a virtual network that you define.

Security groups and NACLs are similar in that they allow you to control access to AWS resources within your VPC. But security groups allow you to control inbound and outbound traffic at the instance level, while NACLs offer similar capabilities at the VPC subnet level. There is no additional charge for using security groups or NACLs.

### Security Groups and Network Access Control Lists (NACLs) (BP5)

You can specify security groups when you launch an instance, or you can associate the instance with a security group at a later time. All internet traffic to a security group is implicitly denied unless you create an *allow* rule to permit the traffic. For example, if you have a web application that uses an ELB and a number of Amazon EC2 instances, you might decide to create one security group for the ELB ("ELB security group") and one for the instances ("web application server security group"). You can then create an *allow* rule to permit internet traffic

to the ELB security group, and another rule to permit traffic from the ELB security group to the web application server security group. This ensures that internet traffic can't directly communicate with your Amazon EC2 instances, which makes it more difficult for an attacker to learn about and impact your application.

When you create NACLs, you can specify both *allow* and *deny* rules. This is useful if you want to explicitly deny certain types of traffic to your application. For example, you can define IP addresses (as CIDR ranges), protocols, and destination ports that are denied access to the entire subnet. If your application is used only for TCP traffic, you can create a rule to *deny* all UDP traffic, or vice versa. This option is useful when responding to DDoS attacks because it lets you create your own rules to mitigate the attack when you know the source IPs or other signature.

If you are subscribed to AWS Shield Advanced, you can register Elastic IPs (EIPs) as Protected Resources. DDoS attacks against EIPs that have been registered as Protected Resources are detected more quickly, which can result in a faster time-to-mitigate. When an attack is detected, the DDoS mitigation systems read the NACL that corresponds to the targeted EIP and enforce it at the AWS network border. This significantly reduces your risk of impact from a number of infrastructure layer DDoS attacks.

To learn more about configuring Security Groups and NACLs to optimize for DDoS resiliency, see How to Help Prepare for DDoS Attacks by Reducing Your Attack Surface.

To learn more about using AWS Shield Advanced with EIPs as Protected Resources, see Enable and Configure AWS Shield Advanced.

## Protecting Your Origin (BP1, BP5)

If you're using Amazon CloudFront with an origin that is inside of your VPC, you should use an AWS Lambda function to automatically update your security group rules to *allow* only Amazon CloudFront traffic. This improves your origin's security by helping to ensure that malicious users cannot bypass Amazon CloudFront and AWS WAF when accessing your web application.

To learn more about how to protect your origin by automatically updating your security groups, see How to Automatically Update Your Security Groups for Amazon CloudFront and AWS WAF by Using AWS Lambda.

You may also want to ensure that only your Amazon CloudFront distribution can forward requests to your origin. With Edge-to-Origin Request Headers, you can add or override the value of existing request headers when Amazon CloudFront forwards requests to your origin. You can use the *X-Shared-Secret* header to help validate that requests made to your origin were sent from Amazon CloudFront.

To learn more about protecting your origin with an *X-Shared-Secret* header, see Forwarding Custom Headers to Your Origin.

## Protecting API Endpoints (BP4)

Typically, when you must expose an API to the public, there is a risk that the API frontend could be targeted by a DDoS attack. To help reduce the risk, you can use Amazon API Gateway as a "front door" to applications running on Amazon EC2, AWS Lambda, or elsewhere. By using Amazon API Gateway, you don't need your own servers for the API frontend and you can obfuscate other components of your application. By making it harder to detect your application's components, you can help prevent those AWS resources from being targeted by a DDoS attack.

When you use Amazon API Gateway, you can choose from two types of API endpoints. The first is the default option: edge-optimized API endpoints that are accessed through an Amazon CloudFront distribution. The distribution is created and managed by API Gateway, however, so you don't have control over it. The second option is to use a regional API endpoint that is accessed from the same AWS region in which your REST API is deployed. We recommend that you use the second type of endpoint, and then associate it with your own Amazon CloudFront distribution. By doing this, you have control over the Amazon CloudFront distribution and the ability to use AWS WAF for application layer protection.

When you use Amazon CloudFront and AWS WAF with Amazon API Gateway, configure the following options:

- Configure the cache behavior for your distributions to forward all headers to the API Gateway regional endpoint. By doing this, CloudFront will treat the content as dynamic and skip caching the content.
- Protect your API Gateway against direct access by configuring the distribution to include the origin custom header *x-api-key*, by setting the API key value in API Gateway.
- Protect your backend from excess traffic by configuring standard or burst rate-limits for each method in your REST APIs.

  To learn more about creating APIs with Amazon API Gateway, see Getting Started with Amazon API Gateway.

# Operational Techniques

The mitigation techniques in this paper help you architect applications that are inherently resilient against DDoS attacks. In many cases, it's also useful to know when a DDoS attack is targeting your application so you can take mitigation steps. If you experience an attack, you may also benefit from help in assessing the threat and reviewing the architecture of your application, or you might want to request other assistance. This section discusses best practices for gaining visibility into abnormal behavior, alerting and automation, and engaging AWS for additional support.

## Visibility

When a key operational metric deviates substantially from the expected value, an attacker may be attempting to target your application's availability. If you're familiar with the normal behavior of your application,  you can more quickly take action when you detect an anomaly.

By using Amazon CloudWatch, you can monitor applications that you run on AWS. For example, you can collect and track metrics, collect and monitor log files, set alarms, and automatically respond to changes in your AWS resources.

If you have architected your application by following the DDoS-resilient reference architecture, common infrastructure layer attacks will be blocked before reaching your application. If you are subscribed to AWS Shield Advanced, you have access to a number of CloudWatch metrics that can indicate that your application is being targeted. For example, you can configure alarms to notify you when there is a DDoS attack in progress, so you can check your application's health and decide whether to engage DRT. You can configure the DDoSDetected metric to tell you if an attack has been detected. If you want to be alerted based on the attack volume, you can also use the DDoSAttackBitsPerSecond, DDoSAttackPacketsPerSecond, or DDoSAttackRequestsPerSecond metrics. You can monitor these metrics by integrating Amazon CloudWatch with your own tools or by using tools provided by third-parties, such as Slack or PagerDuty.

An application layer attack can elevate many Amazon CloudWatch metrics. If you're using AWS WAF, you can use CloudWatch to monitor and alarm on increases in requests that you've set in

WAF to be allowed, counted, or blocked. This allows you to receive a notification if the level of traffic exceeds what your application can handle. You can also use Amazon CloudFront, Amazon Route 53, ALB, NLB, Amazon EC2, and Auto Scaling metrics that are tracked in CloudWatch to detect changes that can indicate a DDoS attack.

For a description of Amazon CloudWatch metrics that are commonly used to detect and react to DDoS attacks, see the following table (Table 3).

| Topic | Metric | Description |
|---|---|---|
| **AWS Shield Advanced** | DDoSDetected | Indicates a DDoS event for a specific Amazon Resource Name (ARN). |
| **AWS Shield Advanced** | DDoSAttackBitsPerSecond | The number of bytes observed during a DDoS event for a specific Amazon Resource Name (ARN). This metric is only available for layer 3/4 DDoS events. |
| **AWS Shield Advanced** | DDoSAttackPacketsPerSecond | The number of packets observed during a DDoS event for a specific Amazon Resource Name (ARN). This metric is only available for layer 3/4 DDoS events. |
| **AWS Shield Advanced** | DDoSAttackRequestsPerSecond | The number of requests observed during a DDoS event for a specific Amazon Resource Name (ARN). This metric is only available for layer 7 DDoS events and is only reported for the most significant layer 7 events. |
| **AWS WAF** | AllowedRequests | The number of allowed web requests. |
| **AWS WAF** | BlockedRequests | The number of blocked web requests. |
| **AWS WAF** | CountedRequests | The number of counted web requests. |
| **Amazon CloudFront** | Requests | The number of HTTP/S requests |
| **Amazon CloudFront** | TotalErrorRate | The percentage of all requests for which the HTTP status code is 4xx or 5xx. |
| **Amazon Route 53** | HealthCheckStatus | The status of the health check endpoint. |
| **ALB** | ActiveConnectionCount | The total number of concurrent TCP connections that are active from clients to the load balancer, and from the load balancer to targets. |
| **ALB** | ConsumedLCUs | The number of load balancer capacity units (LCU) used by your load balancer. |

| Topic | Metric | Description |
|---|---|---|
| **ALB** | HTTPCode_ELB_4XX_Count<br>HTTPCode_ELB_5XX_Count | The number of HTTP 4xx or 5xx client error codes generated by the load balancer. |
| **ALB** | NewConnectionCount | The total number of new TCP connections established from clients to the load balancer, and from the load balancer to targets. |
| **ALB** | ProcessedBytes | The total number of bytes processed by the load balancer. |
| **ALB** | RejectedConnectionCount | The number of connections that were rejected because the load balancer had reached its maximum number of connections. |
| **ALB** | RequestCount | The number of requests that were processed. |
| **ALB** | TargetConnectionErrorCount | The number of connections that were not successfully established between the load balancer and the target. |
| **ALB** | TargetResponseTime | The time elapsed, in seconds, after the request left the load balancer until a response from the target was received. |
| **ALB** | UnHealthyHostCount | The number of targets that are considered unhealthy. |
| **NLB** | ActiveFlowCount | The total number of concurrent TCP flows (or connections) from clients to targets. |
| **NLB** | ConsumedLCUs | The number of load balancer capacity units (LCU) used by your load balancer. |
| **NLB** | NewFlowCount | The total number of new TCP flows (or connections) established from clients to targets in the time period. |
| **NLB** | ProcessedBytes | The total number of bytes processed by the load balancer, including TCP/IP headers. |
| **Auto Scaling** | GroupMaxSize | The maximum size of the Auto Scaling group |
| **Amazon EC2** | CPUUtilization | The percentage of allocated EC2 compute units that are currently in use. |
| **Amazon EC2** | NetworkIn | The number of bytes received by the instance on all network interfaces. |

**Table 3: Recommended Amazon CloudWatch metrics.**

To learn more about using Amazon CloudWatch to detect DDoS attacks on your application, see Getting Started with Amazon CloudWatch.

AWS includes several additional metrics and alarms to notify you about an attack and to help you monitor your application's resources. The AWS Shield console or API provide a summary and details about attacks that have been detected. In addition, the Global Threat Environment Dashboard provides summary information about all DDoS attacks that have been detected by AWS. This can be useful in better understanding DDoS threats across a larger population of applications, understanding attack trends, and comparing with attacks that you may have observed.

Another tool that can help you gain visibility into traffic that is targeting your application is VPC Flow Logs. On a traditional network, you might use network flow logs to troubleshoot connectivity and security issues, and to make sure that network access rules are working as expected. By using VPC Flow Logs, you can capture information about the IP traffic that is going to and from network interfaces in your VPC.

Each flow log record includes the following: source and destination IP addresses, source and destination ports, protocol, and the number of packets and bytes transferred during the capture window. You can use this information to help identify anomalies in network traffic and to identify a specific attack vector. For example, most UDP reflection attacks have specific source ports, such as source port 53 for DNS reflection. This is a clear signature that you can identify in the flow log record. In response, you might choose to block the specific source port at the instance level or create a NACL rule to block the entire protocol if your application doesn't require it.

To learn more about using VPC Flow Logs to identify network anomalies and DDoS attack vectors, see VPC Flow Logs and VPC Flow Logs – Log and View Network Traffic Flows.

# Support

It is important to create a response plan for DDoS attacks before an actual event. The best practices outlined in this paper are intended to be proactive measures that you implement before you launch an application, but DDoS attacks against your application might still occur. Review the options in this section to determine the support resources that are best for your scenario. Your account team can evaluate your use case and application, and assist with specific questions or challenges that you have.

If you're running production workloads on AWS, consider subscribing to Business Support which provides you with 24 x 7 access to Cloud Support Engineers who can assist with DDoS attack issues. If you're running mission-critical workloads, consider Enterprise Support which provides you with the ability to open "Critical" cases and receive the fastest response from a

Senior Cloud Support Engineer.

If you're subscribed to AWS Shield Advanced and are also subscribed to either Business Support or Enterprise Support, you can escalate to the AWS DDoS Response Team (DRT) if you have a DDoS-related event that impacts your application's availability. If your application's responsiveness is degraded because of a DDoS attack, you can make live contact with AWS Support. Another option is to use the AWS Shield Engagement Lambda function to more quickly initiate contact with DRT. For example, you can use an AWS IoT button to trigger the AWS Lambda function if you have an emergency situation. When you press the button, a case is automatically opened with AWS Support and DRT is notified immediately. You receive a direct reply for your case that includes an Amazon Chime conference bridge that you can join to interact with AWS Support and DRT. The AWS Shield Engagement Lambda can be used with any trigger supported by AWS Lambda.

To learn more about rapid DRT engagement by using an AWS Lambda function, see AWS Shield Engagement Lambda.

DRT does not typically have access to your AWS account or AWS WAF Sampled Requests. You can authorize DRT to access AWS WAF, AWS Shield, and related API operations on your account from the AWS Shield console or API. For example, you may want to allow DRT to view your Sampled Requests or place rules to assist with the mitigation of an application layer DDoS attack. You can also authorize DRT to access Amazon S3 buckets that you specify. For example, you may have a bucket where you store web request logs and would like DRT to have access to them for analysis during an attack. DRT will only access your account or make changes during an escalated event and any changes will be subject to your consent. To learn more about granting limited account access to DRT, see Authorize the DDoS Response Team.

In some cases, DRT may learn about a DDoS attack and engage you proactively. If there are specific points of contact who should be engaged during DRT-driven escalation, you can add them in the AWS Shield console by clicking "Summary," followed by "Edit" under the "Additional contacts" section.

# Conclusion

The best practices outlined in this paper can help you to build a DDoS-resilient architecture that can protect your application's availability by preventing many common infrastructure and application layer DDoS attacks. The extent to which you follow these best practices when you architect your application will influence the type, vector, and volume of DDoS attacks that you can mitigate. You can build in resiliency without subscribing to a DDoS mitigation service. Or, optionally, you can choose to subscribe to AWS Shield Advanced to gain additional support, visibility, mitigation, and cost protection features that further protect an already resilient application architecture.

To learn more about DDoS mitigation and DDoS resiliency best practices on AWS, see Appendix A: Additional Resources.

# Contributors

The following individuals and organizations contributed to this document:

- Andrew Kiggins, AWS Solutions Architect
- Jeffrey Lyon, AWS Perimeter Protection
- Achraf Souk, AWS Solutions Architect
- Tino Tran, AWS Solutions Architect
- Yoshihisa Nakatani, AWS Solutions Architect

# Document Revisions

| Date | Description |
|---|---|
| **June 2018** | Updated to include AWS Shield, AWS WAF features, AWS Firewall Manager, and related best practices. |
| **June 2016** | Added prescriptive architecture guidance and updated to include AWS WAF. |

| Date | Description |
| --- | --- |
| **June 2015** | First publication. |

# Appendix A: Additional Resources

You can use the following resources to learn more about DDoS mitigation and DDoS-resiliency best practices with AWS:

- Best Practices for DDoS Mitigation on AWS
- SID216 – re:Invent 2017: Cloud-Native App Protection: Web Application Security at Pearson
- SID324 – re:Invent 2017: Automating DDoS Response in the Cloud
- CTD304 – re:Invent 2017: Dow Jones & Wall Street Journal's Journey to Manage Traffic Spikes While Mitigating DDoS & Application Layer Threats
- CTD310 – re:Invent 2017: Living on the Edge, It's Safer Than You Think! Building Strong with Amazon CloudFront, AWS Shield, and AWS WAF