AWS Best Practices for DDoS Resiliency

June 2016

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

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 (DDoS) attacks. The paper 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.

Introduction

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 that provides more detail on the best practice or capability. You can also view AWS re:Invent conference sessions SEC307 – Building a DDoS-Resilient Architecture with AWS[1] and SEC306 – Defending Against DDoS Attacks[2] for more information.

DDoS Attacks

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

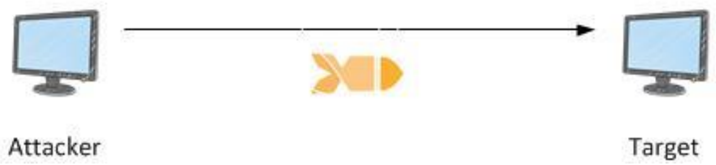

Figure 1: Diagram of a DOS attack

In the case of a Distributed Denial of Service (DDoS) attack, an attacker uses multiple sources—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.

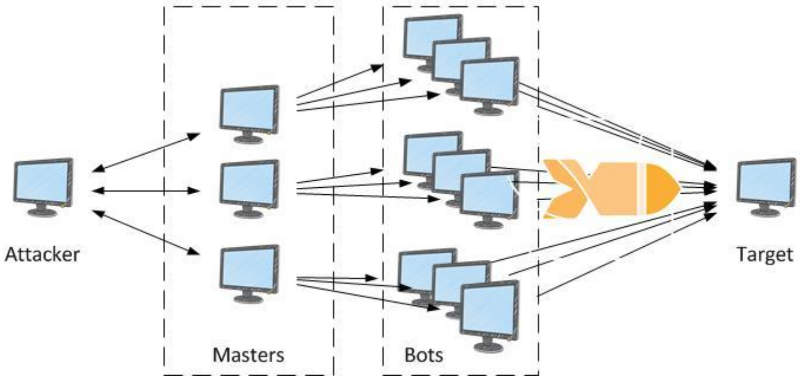

Figure 2: Diagram of a DDOS attack

DDoS attacks are most common at layers 3, 4, 6, and 7 of the Open Systems Interconnection (OSI) model, which is described in Table 1. Layer 3 and 4 attacks correspond to the Network and Transport layers of the OSI model: This document refers to them as infrastructure layer attacks. Layer 6 and 7 attacks correspond to the Presentation and Application layers of the OSI model: This document refers to them as application layer attacks.

#	Layer	Unit	Description	Vector Examples
7	Application	Data	Network process to application	HTTP floods, DNS query floods
6	Presentation	Data	Data representation and encryption	SSL 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

This distinction is important because the attack types directed at these layers are different and so different techniques are used to build resiliency.

Infrastructure Layer Attacks

The most common DDoS attacks, User Datagram Protocol (UDP) reflection attacks and synchronize (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 system like a server, firewall, IPS, or load balancer. These attacks have clear signatures that can make them easier to detect. Effective mitigation of these attacks requires network or system resources in excess of the volume that is generated by the attacker.

UDP is a stateless protocol. This can allow an attacker to spoof the source of a request sent to a server that elicits a larger response. The amplification factor, which is the ratio of request size to response size, varies depending on the protocol used, such as, Domain Name System (DNS), Network Time Protocol (NTP), or Simple Service Discovery Protocol (SSDP). For example, the 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 over 3400 bytes of unwanted traffic. This concept is illustrated in Figure 3.

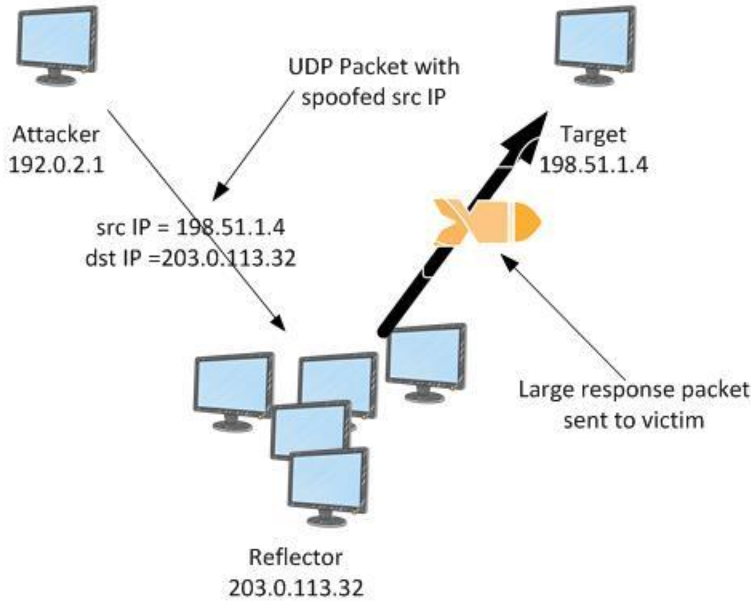

Figure 3: UDP reflection attack

SYN floods can be in the order of tens of Gbps, but the intent of the attack is to exhaust the available resources of a system by leaving connections in a half-open state. As illustrated in Figure 4, when an end user connects to a TCP service, like a web server, the client will send a SYN packet. The server will return SYN-ACK and the client will return ACK, completing the three-way handshake.

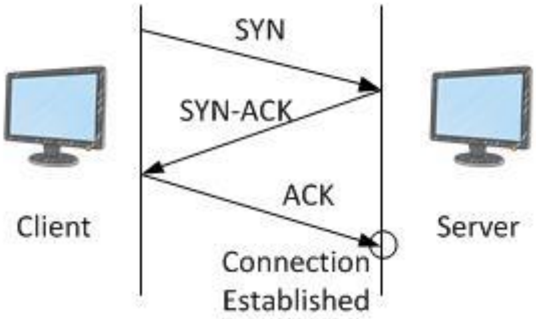

Figure 4: SYN 3-way handshake

In a SYN flood, the ACK is never returned and the server is left waiting for a response. This can prevent new users from connecting to the server.

Application Layer Attacks

Less frequently, an attacker might target the application itself with a layer 7 or application layer attack. These attacks differ from infrastructure layer attacks because the attacker is attempting to over-exercise specific functions of an application in order to render it unavailable. In some cases, this can be achieved with very low request volumes that do not generate a large volume of network traffic. This can make the attack more difficult to detect and mitigate. Examples of application layer attacks include HTTP floods, cache-busting attacks, and WordPress XML-RPC floods.

With an HTTP flood, an attacker sends HTTP requests that appear to be from a real user of the web application. Some HTTP floods will target a specific resource, while more complex HTTP floods will attempt to emulate human behavior. 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 which results in origin fetches, causing additional strain on the origin web server.

With a WordPress XML-RPC flood, also known as a WordPress pingback flood, an attacker can misuse the XML-RPC API function of a website hosted on the

WordPress brand content management software to generate a flood of HTTP requests. The pingback feature allows a website hosted on WordPress (Site A) to notify a different WordPress site (Site B) that Site A has created a link to Site B. As a result, Site B will attempt to fetch Site A to verify the existence of the link. In the case of a pingback flood, the attacker misuses this capability to cause Site B to attack Site A. This type of attack has a clear signature because "WordPress" should be present in the "User-Agent" of the HTTP request header.

Application layer attacks can also target domain name system (DNS) services. The most common of these attacks is a DNS query flood where 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 sub-domain string to bypass the local DNS cache of any given resolver. As a result, the resolver is conscripted in an attack against the authoritative DNS server.

In the case of web applications that are delivered over Secure Sockets Layer (SSL), an attacker can choose to attack the SSL negotiation process. SSL is computationally expensive which allows an attacker to impact the availability of the server by sending unintelligible data. Other variations of this attack involve an attacker who completes the SSL handshake but perpetually renegotiates the encryption method. Similarly, the attacker can choose to exhaust server resources by opening and closing many SSL sessions.

Mitigation Techniques

AWS infrastructure is DDoS-resilient by design and is supported by DDoS mitigation systems that can automatically detect and filter excess traffic. To protect the availability of your application, it is necessary to implement an architecture that allows you to take advantage of these capabilities.

One of the most common AWS use cases is a web application that serves static and dynamic content to users over the Internet. For a DDoS-resilient reference architecture commonly used with web applications, see Figure 5.

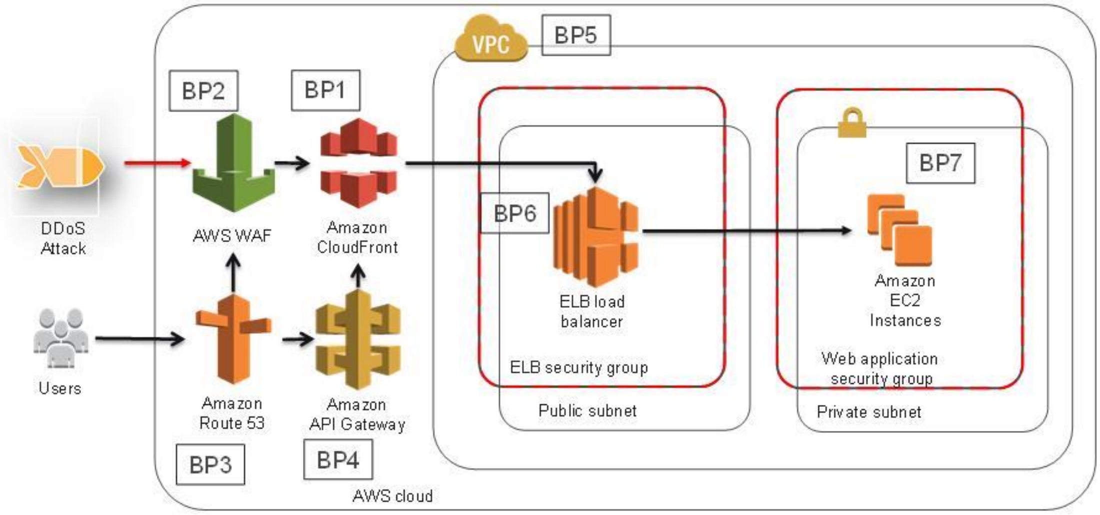

Figure 5: DDoS-resilient reference architecture

This reference architecture includes many AWS services that can help you improve the resiliency of your web application against DDoS attacks. Best practices in this architecture are enumerated for easy reference as they are discussed throughout the document. For example, a section that discusses capabilities provided by Amazon CloudFront will be referenced with a best practices indicator (e.g., BP1). For a summary of these services and the capabilities that they can provide, see Table 2.

	AWS Edge Locations			**AWS Regions**		
	Amazon CloudFront with AWS WAF (BP1, BP2)	Amazon API Gateway (BP4)	Amazon Route 53 (BP3)	Elastic Load Balancing (BP6)	Amazon VPC (BP5)	Amazon EC2 with Auto Scaling (BP7)
Layer 3 (e.g., UDP reflection) attack mitigation	✓	✓	✓	✓	✓	
Layer 4 (e.g., SYN flood) attack mitigation	✓	✓	✓	✓		
Layer 6 (e.g., SSL) attack mitigation	✓	✓	N/A	✓		
Reduce attack surface	✓	✓	✓	✓	✓	
Scale to absorb application layer traffic	✓	✓	✓	✓		✓
Layer 7 (application layer) attack mitigation	✓	✓	✓			
Geographic isolation and dispersion of excess traffic and larger DDoS attacks	✓	✓	✓			

Table 2: Summary of best practices

Services that are available within AWS Regions, like Elastic Load Balancing and Amazon Elastic Compute Cloud (EC2), allow you to build DDoS resiliency and scale to handle unexpected volumes of traffic within a given region. Services that are available in AWS edge locations, like Amazon CloudFront, AWS WAF, Amazon Route 53, and Amazon API Gateway, allow you to take advantage of a global network of edge locations that can provide your application with greater fault tolerance and increased scale for managing larger volumes of traffic. The benefits of using each these services to build resiliency against infrastructure layer and application layer DDoS attacks are discussed in the following sections.

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, you have options for architecting your application to be able to scale and absorb larger volumes of traffic without capital-intensive investments or unnecessary complexity. Key considerations in the mitigation of volumetric DDoS attacks include the availability of transit capacity and diversity and protecting AWS resources like Amazon EC2 instances against attack traffic.

Instance Size (BP7)

Many AWS customers use Amazon EC2 for resizable compute capacity, which allows you to quickly scale up or down as your requirements change. You can scale horizontally by adding instances to your application, as required. You can also choose to scale vertically by using larger instances. Some instance types support features, such as 10 Gigabit network interfaces and Enhanced Networking, that can improve your ability to handle larger volumes of traffic.

With 10 Gigabit network interfaces, each instance is able to support a larger volume of traffic. This helps prevent interface congestion for any 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 that is larger in packet volume. On AWS, you are not responsible for the cost of inbound data transfer.

To learn more about Amazon EC2 instances that support 10 Gigabit network interfaces and Enhanced Networking, see Amazon EC2 Instance Types[3]. To learn how to enable Enhanced Networking, see Enabling Enhanced Networking on Linux Instances in a VPC[4].

Choice of Region (BP7)

Many AWS services, like Amazon EC2, are available in multiple locations worldwide. These geographically separate areas are called AWS Regions. When architecting your application, you have the ability to choose one or more regions based on your own 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 that allow for optimal latency and throughput to similarly situated end-users.

It is also important to consider your choice of region in terms of DDoS resiliency. Many regions are closer to large Internet exchanges. Many DDoS attacks originate internationally, so it is helpful to be close to exchanges where international carriers and large peers frequently maintain a strong presence. This helps end-users reach your application when dealing with larger volumes of traffic.

To learn more about choosing a region, see Regions and Availability Zones[5] and ask your account team about the characteristics of each region to help you make an informed decision.

Load Balancing (BP6)

Larger DDoS attacks can exceed the size of a single Amazon EC2 instance. To mitigate these attacks, you will want to consider options for 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 of unanticipated traffic, like flash crowds or DDoS attacks.

ELB accepts only well-formed TCP connections. This means that many common DDoS attacks, like SYN floods or UDP reflection attacks will not be accepted by ELB and will not be passed to your application. When ELB detects these types of attacks, it will automatically scale to absorb the additional traffic but you will not incur any additional charges.

To learn more about using ELB to distribute load and to protect Amazon EC2 instances, see Getting Started with Elastic Load Balancing[6].

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 end-users, absorb DDoS attacks, and to isolate faults while minimizing availability impact. AWS edge locations provide an additional layer of network infrastructure that provides these benefits to web applications using Amazon CloudFront and Amazon Route

53. With these services, your content is served and DNS queries resolved from locations that are often closer to your end-users.

Web Application Delivery at the Edge (BP1)

Amazon CloudFront is a content delivery network (CDN) 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) can be used to accelerate delivery of content, even if it cannot be cached at an edge location. This allows you to use Amazon CloudFront to protect your web application, even if you are not serving static content. Amazon CloudFront only accepts well-formed connections to prevent many common DDoS attacks like SYN floods and UDP reflection attacks from reaching your origin. DDoS attacks are geographically isolated close to the source, which prevents the traffic from affecting other locations. These capabilities can greatly improve your ability to continue serving traffic to end users during larger DDoS attacks. You can use Amazon CloudFront to protect an origin on AWS or elsewhere on the Internet.

To learn more about optimizing the performance of web applications using Amazon CloudFront, see Getting Started with CloudFront[7].

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 many advanced features like traffic flow, latency-based routing, Geo DNS, health checks, and monitoring. These features allow you to control how the service responds to DNS requests in order to optimize for latency, health, and other considerations. You can use these features to improve the performance of your web application and to avoid site outages.

Amazon Route 53 uses shuffle sharding and anycast striping so end users can 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, end-users can retry and receive a response from another name server at a different edge location. Anycast striping is used so that each DNS request is served by the most optimal location. This has the effect of spreading load and reducing DNS latency, which allows end-users to receive a response more quickly. 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.

If you have many Amazon Route 53 hosted zones you can create a reusable delegation set that will provide you with the same set of authoritative name servers for each domain. This can make it easier to maintain your hosted zones. In the event of a DDoS attack, it also allows AWS to apply a single mitigation that covers any hosted zone where the reusable delegation set is used.

To learn more about using Amazon Route 53 to direct end-users to your application, see Getting Started with Amazon Route 53[8]. To learn more about reusable delegation sets, see Actions on Reusable Delegation Sets[9].

Application Layer Defense (BP1, BP2, BP6)

Many of the techniques discussed in this paper are effective at mitigating the availability impact of infrastructure layer DDoS attacks. Defending your application against application layer attacks requires you to implement an architecture that allows you to detect plus 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)

Web application firewalls (WAFs) are often used to protect web applications against attacks that attempt to exploit a vulnerability in the application. Common examples include SQL injection or cross-site request forgery. You can also use a WAF to detect and mitigate web application layer DDoS attacks.

On AWS, you can use Amazon CloudFront and AWS WAF to defend your application against these attacks. Amazon CloudFront allows you to cache static content and serve it from AWS Edge Locations that can help reduce the load on your origin. Additionally, Amazon CloudFront can automatically close connections from slow-reading or slow-writing attackers (e.g., Slowloris). You can use Amazon CloudFront geo restriction to prevent users in specific geographic locations from accessing your content. This can be useful in case you want to block attacks that are originating from geographic locations where you do not expect to serve end-users.

For other types of attacks, like HTTP floods or WordPress pingback floods, you can use AWS WAF to create your own mitigations. If you know the source IP addresses that you want to block, you can create a rule with an action to block and associate it with a web ACL. You can then create an IP address match condition in the web ACL to block the source IP addresses that are participating in the attack. You can also create rules with conditions that block by URI, query string, HTTP method, or header key. The latter is useful in case of attacks that have a clear signature. For example, a WordPress pingback attack will always have "WordPress" in the User-Agent.

It can be challenging to identify the signature of a DDoS attack or to accurately identify the IP addresses that are participating in the attack. Sometimes, it is possible to find this information by reviewing your web server logs. You can also use the AWS WAF console to view a sample of requests that Amazon CloudFront has forwarded to AWS WAF. Sampled requests can help you decide which rules might be needed to mitigate an application layer attack. If you see many 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.

Some attacks consist of web traffic that is disguised to look like regular end user traffic. To mitigate this type of attack, you can use an AWS Lambda function to implement rate-based blacklisting. With rate-based blacklisting, you can set a threshold for how many requests your web application can serve. If a bot or crawler exceeds this limit, you can use AWS WAF to automatically block any additional requests.

To learn more about using geo restriction to limit access to your Amazon CloudFront distribution, see Restricting the Geographic Distribution of Your Content[10].

To learn more about using AWS WAF, see Getting Started with AWS WAF[11] and Viewing a Sample of the Web Requests that CloudFront has Forwarded to AWS WAF[12].

To learn how to configure rate-based blacklisting with AWS Lambda and AWS WAF, see How to Configure Rate-Based Blacklisting with AWS WAF and AWS Lambda[13].

Scale to Absorb (BP6)

Another way to deal with application layer attacks is to operate at scale. In the case of web applications, you can use ELB to distribute traffic to many Amazon EC2 instances that are overprovisioned or configured to auto scale for the purpose of serving surges of traffic, whether it is the result of a flash crowd or an application layer DDoS attack. Amazon CloudWatch alarms are used to initiate Auto Scaling, which automatically scales the size of your Amazon EC2 fleet in response to events that you define. This protects application availability even when dealing with an unexpected volume of requests. By using Amazon CloudFront or ELB, SSL negotiation is handled by the distribution or load balancer, which can prevent your instances from being affected by SSL-based attacks.

To learn more about using Amazon CloudWatch to invoke Auto Scaling, see Monitoring Your Auto Scaling Instances and Groups Using Amazon CloudWatch[14].

Attack Surface Reduction

Another important consideration when architecting on AWS is to limit the opportunities that an attacker may have to target your application. For example, if you do not expect an end user to directly interact with certain resources you will want to make sure that those resources are not accessible from the Internet. Similarly, if you do not expect end-users or external applications to communicate with your application on certain ports or protocols, you will want to make sure that traffic is not accepted. This concept is known as attack surface reduction. In this section, you will find best practices that allow you to reduce your attack surface and limit the extent to which your application is exposed to the Internet. Resources that are not exposed to the Internet are more difficult to attack, which limits the options an attacker might have to target the availability of your application.

Obfuscating AWS Resources (BP1, BP4, BP5)

For many applications, your AWS resources do not need to be fully exposed to the Internet. For example, it may not be necessary for Amazon EC2 instances behind an ELB to be publicly accessible. In this scenario, you might decide to allow end-users to access the ELB on certain TCP ports and to allow only the ELB to communicate with the Amazon EC2 instances. This can be achieved 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 network ACLs are similar in that they allow you to control access to AWS resources within your VPC. Security groups allow you to control inbound and outbound traffic at the instance level and network ACLs offer similar capabilities, but at the VPC subnet level. Additionally, there is no charge for inbound data transfer on Amazon EC2 security group (SG) rules or network ACLs. This ensures that you do not incur any additional charges for traffic that is dropped by your security groups or network ACLs.

Security Groups (BP5)

You can specify security groups when launching an instance or associate the instance with a security group later on. All traffic to a security group from the Internet is implicitly denied unless you create an *allow* rule to permit the traffic. For example, if you have a web application that consists of an ELB and many 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 *allow* rules to permit traffic from the Internet to the ELB security group and to permit traffic from the ELB security group to the web application server security group. As a result, traffic from the Internet is unable to directly communicate with your Amazon EC2 instances, which makes it more difficult for an attacker to learn about your application.

Network Access Control Lists (ACLs) (BP5)

With network ACLs, you can specify both *allow* and *deny* rules. This is useful in case 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 should be denied for 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 tool is useful when responding to DDoS attacks because it can allow you to create your own rules to mitigate the attack if you know the source IP addresses or other signature.

Protecting Your Origin (BP1)

If you are 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 traffic from Amazon CloudFront. This can improve the security of your origin by helping to ensure that Amazon CloudFront and AWS WAF can't be bypassed.

To learn more about protecting 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[15].

You may also want to ensure that only your Amazon CloudFront distribution is forwarding 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 *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[16].

Protecting API Endpoints (BP4)

Ordinarily, when there is a need to expose an API to the public there is a risk that the API front end could be targeted by a DDoS attack. Amazon API Gateway is a fully managed service that allows you to create an API that acts as a "front door" to applications running on Amazon EC2, AWS Lambda, or any web application. With Amazon API Gateway, you do not need to run your own servers for the API front end, and you can obfuscate other components of your application from the public. This can help prevent those AWS resources from being targeted by a DDoS attack. Amazon API Gateway is integrated with Amazon CloudFront, which allows you to benefit from the added DDoS-resiliency that is inherent to that service. You can also 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[17].

Operational Techniques

The mitigation techniques in this paper allow you to architect applications that are inherently resilient against DDoS attacks. In many cases, it is also useful to know when DDoS attacks are targeting your application and to be able to take action on this data. You may also want to engage additional resources to assess a threat, review the architecture of your application, or 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

Understanding the normal behavior of your application can allow you to more quickly take action when you detect an anomaly. When a key metric deviates substantially from the expected value, this is an indication that an attacker may be attempting to target the availability of your application. With Amazon CloudWatch, you can monitor your applications running on AWS. It allows you to collect and track metrics, collect and monitor log files, set alarms, and automatically react to changes in your AWS resources. For a description of Amazon CloudWatch metrics that are commonly used to detect and react to DDoS attacks, see Table 3.

Topic	**Metric**	**Description**
Auto Scaling	GroupMaxSize	The maximum size of the Auto Scaling group
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 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
ELB	SurgeQueueLength	The number of requests queued by the load balancer, awaiting a back-end instance to accept connections and process the request
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 successful
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 3: Recommended Amazon CloudWatch metrics

For an application that is architected according to the DDoS-resilient reference architecture provided in Figure 5, common infrastructure layer attacks will be blocked before reaching your application. As a result, these attacks will not appear in your Amazon CloudWatch metrics.

An application layer attack may cause elevations on many of these metrics. For example, an HTTP flood may cause elevations in requests and CPU and network utilization for Amazon CloudFront, ELB, and Amazon EC2 metrics. If the backend instances are not able to serve the excess requests, you may also see elevations in TotalErrorRate on Amazon CloudFront and SurgeQueueLength, UnHealthyHostCount, Latency, BackendConnectionErrors, SpilloverCount, or HTTPCode on ELB. In this case, the volume of HTTP requests may be depressed as the application is not able to serve regular end users. You can remedy this condition by scaling the backend of your application or by blocking the excess traffic with AWS WAF as discussed earlier in this paper.

To learn more about using Amazon CloudWatch to detect DDoS attacks against your application, see Getting Started with Amazon CloudWatch[18].

Another tool that you can use to gain visibility into traffic 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. With VPC Flow Logs, you can capture information about the IP traffic going to and from network interfaces in your VPC.

Each flow log record includes the source and destination IP addresses, source and destination ports, protocol, and the number of packets and bytes transferred during the capture window. This information can be used to help identify anomalies in network traffic and to identify the specific attack vector. For example, most UDP reflection attacks will have specific source ports (e.g., 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 network ACL rule to block the entire protocol if it is not required.

To learn more about using VPC Flow Logs to identify network anomalies and DDoS attack vectors, see VPC Flow Logs[19] and VPC Flow Logs – Log and View Network Traffic Flows[20]

Support

It is important to create a plan for DDoS attacks before an actual event. The best practices outlined in this paper are intended to be proactive measures and should be implemented prior to launching an application that may be targeted by a

DDoS attack. Your account team can help review your use case and application and assist with any specific questions or challenges that you may encounter.

Sometimes, you may find it beneficial to contact AWS for additional support during a DDoS attack; your case will be answered quickly and routed to an expert who is able to assist. By subscribing to Business Support, you gain 24 x 7 access to Cloud Support Engineers by email, chat, or telephone.

If you are running mission-critical workloads on AWS, you should consider Enterprise Support. With Enterprise Support, your urgent cases receive the highest priority and are routed to Senior Cloud Support engineers. Additionally, Enterprise Support provides access to a Technical Account Manager (TAM) who is your advocate and dedicated technical point of contact. Enterprise Support also provides you with access to Infrastructure Event Management, which includes real-time operational support during planned events, product launches, and migrations.

To learn more about choosing a support plan to suit your unique needs, see Compare AWS Support Plans[21].

Conclusion

The best practices outlined in this paper can allow you to build a DDoS-resilient architecture that is capable of protecting the availability of your application against many common infrastructure and application layer DDoS attacks. The degree to which you are able to architect your application according to these best practices will influence the type, vector, and volume of DDoS attacks that you are able to mitigate. AWS encourages you to use these best practices to better protect the availability of your application against common DDoS attacks.

Contributors

The following individuals and organizations contributed to this document:

- Andrew Kiggins, AWS Solutions Architect

- Jeffrey Lyons, AWS DDoS Ops Engineering

Notes

[1] https://www.youtube.com/watch?v=OT2y3DzMEmQ

[2] https://www.youtube.com/watch?v=Ys0gG1koqJA

[3] https://aws.amazon.com/ec2/instance-types/

[4] http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/enhanced-networking.html

[5] http://docs.aws.amazon.com/AWSEC2/latest/UserGuide/using-regions-availability-zones.html

[6] http://docs.aws.amazon.com/ElasticLoadBalancing/latest/DeveloperGuide/elb-getting-started.html

[7] http://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/GettingStarted.html

[8] http://docs.aws.amazon.com/Route53/latest/DeveloperGuide/getting-started.html

[9] http://docs.aws.amazon.com/Route53/latest/APIReference/actions-on-reusable-delegation-sets.html

[10] http://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/georestrictions.html

[11] http://docs.aws.amazon.com/waf/latest/developerguide/getting-started.html

[12] http://docs.aws.amazon.com/waf/latest/developerguide/web-acl-testing.html#web-acl-testing-view-sample

[13] https://blogs.aws.amazon.com/security/post/Tx1ZTM4DT0HRH0K/How-to-Configure-Rate-Based-Blacklisting-with-AWS-WAF-and-AWS-Lambda

[14] http://docs.aws.amazon.com/autoscaling/latest/userguide/as-instance-monitoring.html

[15] https://blogs.aws.amazon.com/security/post/Tx1LPI2H6Q6S5KC/How-to-Automatically-Update-Your-Security-Groups-for-Amazon-CloudFront-and-AWS-W

[16] http://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/forward-custom-headers.html

[17] https://aws.amazon.com/api-gateway/getting-started/

[18] http://docs.aws.amazon.com/AmazonCloudWatch/latest/DeveloperGuide/GettingStarted.html

[19] http://docs.aws.amazon.com/AmazonVPC/latest/UserGuide/flow-logs.html

[20] https://aws.amazon.com/blogs/aws/vpc-flow-logs-log-and-view-network-traffic-flows/

[21] https://aws.amazon.com/premiumsupport/compare-plans/