Incident Response Playbook Template

Incident Type

Web Application Dos/DDoS Attack

Introduction

This playbook is provided as a template to customers using AWS products and who are building their incident response capability. You should customize this template playbook to suit your particular needs, risks, available tools and work processes.

Security and Compliance is a shared responsibility between you and AWS. AWS is responsible for “Security of the Cloud”, while you are responsible for “Security in the Cloud”. For more information on the shared responsibility model, please review our documentation (https://aws.amazon.com/compliance/shared-responsibility-model/).

Summary

This Playbook

This playbook outlines response steps for Web Application Dos/DDoS Attack incidents. These steps are based on the [NIST Computer Security Incident Handling Guide] (https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-61r2.pdf (https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-61r2.pdf)) (Special Publication 800-61 Revision 2) that can be used to:

- Gather evidence
- Contain and then eradicate the incident
- recover from the incident
- Conduct post-incident activities, including post-mortem and feedback processes

Interested readers may also refer to the AWS Security Incident Response Guide (https://docs.aws.amazon.com/whitepapers/latest/aws-security-incident-response-guide/welcome.html) which contains additional resources.

Once you have customized this playbook to meet your needs, it is important that you test the playbook (e.g., Game Days) and any automation (functional tests), update as necessary to achieve the desired results, and then publish to your knowledge management system and train all responders.

Note that some of the incident response steps noted in each scenario may incur costs in your AWS account(s) for services used in either preparing for, or responding to incidents. Customizing these scenarios and testing them will help you to determine if additional costs will be incurred. You can use AWS Cost Explorer (https://aws.amazon.com/aws-cost-management/aws-cost-explorer/) and look at costs incurred over a particular time frame (such as when running Game Days) to establish what the possible impact might be.

In reviewing this playbook, you will find steps that involve processes that you may not have in place today. Proactively preparing for incidents means you need the right resource configurations, tools and services in place that allow you to respond to an incident.

The next section will provide a summary of this incident type, and then cover the five steps (parts 1 - 5) for handling DoS and DDoS attacks.

This Incident Type

DoS and DDoS attacks are “denial of service” attacks, designed to prevent legitimate users from accessing your web applications. They target different layers of the OSI model and often exploit protocol weaknesses. Other attacks may simply use legitimate requests to your web application, but increase the volume of those requests to a point where your web application is overwhelmed and no longer able to service legitimate requests. When you use AWS services, you benefit from protections built in to those services. Additionally, if you follow the AWS Best Practices for DDoS Resiliency (https://docs.aws.amazon.com/whitepapers/latest/aws-best-practices-ddos-resiliency/aws-best-practices-ddos-resiliency.pdf), you will increase the availability of your application and its ability to withstand DDoS attacks.

Incident Response Process

Part 1: Acquire, Preserve, Document Evidence

1. You become aware that your application is not meeting availability requirements. This may mean that the application is unavailable, is providing a degraded service, or even appears to be functioning, but application performance metrics or logging are suggesting that there may be an issue. This information could come via different means, for example:

- Customer contact/feedback/support calls
- An internal ticketing system (the sources of the ticket are varied and could include any of the means below)
- A message from a contractor or third-party service provider

- From an alert in one of your own monitoring systems either internal or external to AWS (for example, in AWS, in this particular situation, it could be via CloudWatch metrics triggering an alarm from Amazon EC2, AWS Application Load Balancer, Amazon CloudFront, AWS WAF or other services that you have configured CloudWatch alarms for)
- Via an anonymous tip
- Via independent or external security researchers

2. Confirm an internal ticket/case has been raised for the incident within your organization. If not, manually raise one.
3. Determine and begin to document end-user impact/experience of the issue. This should be documented in the ticket/case related to the incident
4. In the case of automatically created tickets/cases, determine what internal alarms/metrics are currently indicating an issue (what caused the ticket to be created?)
5. Determine the application impacted (this may be done quickly via Resource Tags)
6. Determine if there are any known events that could be causing disruption to your application (increased traffic due to a sales event, or similar)
7. Check your Configuration Management Database (CMDB) to determine if there were any deployments to your production environment (code, software or infrastructure updates) that may result in an increase in traffic to the affected URL(s)
8. Incident Communications:
    1. Identify stakeholder roles from the application entry in the CMDB entry for that application, or via the application’s risk register
    2. Open a conference bridge war room for the incident
    3. Notify identified stakeholders including (if required) legal personnel, public relations, technical teams and developers and add them to the ticket and the war room, so they are updated as the ticket is updated
9. External Communications:
    1. Ensure your organizations legal counsel is informed and is included in status updates to internal stakeholders and especially in regards to external communications.
    2. For colleagues in the organization that are responsible for providing public/external communication statements, ensure these internal stakeholders are added to the ticket so they receive regular status updates regarding the incident and can complete their own requirements for communications within and external to the business.
    3. If there are regulations in your jurisdiction requiring reporting of such incidents, ensure the people in your organization responsible for notifying local or federal law enforcement agencies are also notified of the event/added to the ticket. Consult your legal advisor and/or law enforcement for guidance on collecting and preserving the evidence and chain of custody.
    4. There may not be regulations, but either open databases, government agencies or NGOs may track this type of activity. Your reporting may assist others
10. Determine the resources involved in serving the application (start with front-end resources, including CDNs, load balancers and front-end application servers, then move to supporting services, such as databases, caches, etc.)
11. Obtain the application’s documented baseline and locate the metrics for standard application performance from the CMDB.
12. Determine if the application’s resources are currently displaying metrics outside the baseline:
    1. For Amazon CloudFront, check the following metrics:

        1. Confirm if there is a significant variation in:
            1. Data transferred (https://aws.amazon.com/cloudfront/reporting/) over HTTP and/or HTTPS
            2. Total number (https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/monitoring-using-cloudwatch.html) of HTTP requests
        2. If monitoring CloudFront access logs, determine if there has been a spike in 4xx or 5xx HTTP return codes (in Amazon CloudWatch, All >> CloudFront >> Per-Distribution Metrics) with a Metric Name of 5xxErrorRate or 4xxErrorRate
    2. For Application Load Balancers, check the following metrics (https://docs.aws.amazon.com/elasticloadbalancing/latest/application/load-balancer-cloudwatch-metrics.html), which may give further information on the type of attack, and how it is impacting your application:
        1. Confirm if there is a significant variation in:
            1. HTTPCode_ELB_4xx_Count (indicators of attacks that may involve credential stuffing, malformed or incomplete requests, or overly aggressive website scraping, etc.)
            2. HTTPCode_ELB_2xx_Count (a dramatic increase may indicate a HTTP flood attack)
            3. ActiveConnectionCount (indicates total concurrent TCP connections to ALB, when used with SUM)
            4. ClientTLSNegotiationErrorCount (this may indicate a protocol exploitation attack by abusing the SSL handshake process, such as sending only weak ciphers in a request)
            5. RequestCount (a significant increase in requests may indicate a HTTP flood attack)
            6. HTTPCode_Target_5XX_Count (as this is a response dedicated by a host or container behind the load balancer, this may indicate that the server is unable to respond to the number of received requests, this may also be the result of impacted services downstream, such as a supporting database)
            7. TargetResponseTime (increasing response times may indicate server resources are not available to serve requests from the load balancer)
            8. ELBAuthFailure (indicates that user authentication could not be completed, as the IdP denied access to the user, or a user attempted to replay an already used auth token)
        2. Note the metrics of concern
    3. For Amazon Elastic Compute Cloud (Amazon EC2) instances, check the following metrics (https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/monitoring_ec2.html):
        1. Confirm if there is significant variation in:
            1. CPU utilization
            2. Network utilization
            3. Disk performance
            4. Memory utilization (if using the CloudWatch Logs Agent (https://docs.aws.amazon.com/AmazonCloudWatch/latest/monitoring/Install-CloudWatch-Agent.html))
        2. Note the metrics of concern

13. Obtain logs relevant to the incident (these will be the logs from the resources identified above, in point 4), these should include the following (note that these logs should be already configured to be sent to CloudWatch Logs):
    1. Web server access logs
        1. RHEL-based - /var/log/apache/access.log, or

        2. Debian-based - /var/log/apache2/access.log
    2. Load balancer logs:
        1. Application Load Balancer Logs will be configured for the ALB and sent to an Amazon Simple Storage Service (Amazon S3) bucket (https://docs.aws.amazon.com/elasticloadbalancing/latest/application/load-balancer-access-logs.html). If you are not sure which bucket the logs are being sent to, use this document to help you determine the bucket name (https://docs.aws.amazon.com/elasticloadbalancing/latest/application/load-balancer-access-logs.html#enable-access-logging).
    3. CloudFront Distribution logs. CloudFront will be configured to send its access logs to an S3 bucket (https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/AccessLogs.html).

14. Once you have verified you have access to the logs, use your preferred tool (for example, Amazon Athena, internal, or third-party tools) to review the logs and determine if there are any identifiable attack signatures. Such signatures may be the following:
    1. An unusual number of requests within a given time period
    2. An unusual set of source IPs
    3. A set of unrecognized URLs in the request section
    4. POST or Query parameters that are not expected or supported by the application
    5. Unexpected HTTP verbs (GET, where we expect POST, etc)
    6. An unusually high number of 2xx, 4xx or 5xx responses
15. From the log dive outlined in previous steps, determine the likely attack vector (HTTP flood, SYN flood, credential stuffing, UDP reflection attack, etc.)
16. What resources are being targeted? Note the Amazon resource names (ARN)
17. Note specific information about the service hosted on the AWS resources For example, what type of application is it? What is the provenance of the application code? Are there other dependencies on this application?

Part 2: Contain the Incident

From the Acquire, Preserve, Document Evidence phase, you should now have a good idea of the attack vector. The following steps describe how to help to mitigate the attack.

1. If it is a single EC2 instance serving the public content:
    1. determine from the application's developer whether the application is "cloud native" (can handle things such as Amazon EC2 Auto Scaling , for example as application instances spin up and down).
    2. Determine the AMI ID for the EC2 instance(s) identified during the evidence phase and determine if this is an approved image for this application
    3. If there is no pre-defined image for spinning up additional EC2 instances, consult with the application developer whether an image can be created from the existing instance, and additional instances spun up from that image. If this is possible, proceed with those steps
    4. From EC2 metrics in CloudWatch outlined in the evidence phase (Part 1), determine if the most suitableEC2 instance type (https://aws.amazon.com/ec2/instance-types/) is in use and/or if additional instances should be added behind an AWS elastic load balancer (https://aws.amazon.com/elasticloadbalancing/) to spread the load

        5. If so, this will allow you to scale additional EC2 instances using Amazon EC2 Auto Scaling (https://docs.aws.amazon.com/autoscaling/ec2/userguide/GettingStartedTutorial.html), and put those instances behind an appropriate load balancer
        6. Update DNS records in your Hosted Zone to point to the load balancer. For security groups on the EC2 instance, allow access only from the load balancer's security group, and do not allow direct public access from any port or protocol
2. If it is a group of EC2 instances behind a load balancer:
    1. Determine if Amazon EC2 Auto Scaling is in use.
        1. If not, create an Auto Scaling Group, define a Launch Configuration (https://docs.aws.amazon.com/autoscaling/ec2/userguide/create-launch-config.html) or Launch Template (https://docs.aws.amazon.com/AWSEC2/latest/UserGuide/ec2-launch-templates.html) and add the existing instances to the Auto Scaling Group
        2. If it is in use, move to step (2b).
    2. Confirm with the relevant stakeholders that CloudFront can be used and;
    3. Deploy a CloudFront distribution (https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/distribution-web-creating-console.html), using the load balancer as an origin for the distribution
3. If it is a load balancer (or EC2 instance) behind a CloudFront distribution
    1. Determine if it is a single EC2 instance serving as an origin:
        1. If so, go to step 1 and work through those steps
        2. If it is not, verify the following:
            1. There is a load balancer
            2. The target groups are spread across multiple Availability Zones
            3. The EC2 instances are managed by EC2 Auto Scaling
        3. If any of the points in (ii) are not true, go back and work through step 2, above
        4. Finally, ensure that the security groups for the load balancer are appropriately configured. This can be done automatically by using AWS Lambda and the AWS published IP ranges for CloudFront. There is a Git Repository with code that will set this up for you (https://github.com/aws-samples/aws-cloudfront-samples) and maintain the ranges if there are changes to the public IP ranges published.

Part 3: Eradicate the Incident

By following steps 1 through to 4 in PART 2, the service is now behind a CloudFront distribution, in addition to the standard protections provided by AWS Shield (https://docs.aws.amazon.com/whitepapers/latest/aws-best-practices-ddos-resiliency/aws-best-practices-ddos-resiliency.pdf) . Now further refine protections by using AWS WAF.

1. Determine if the CloudFront distribution or Application Load Balancer are protected by an AWS WAF WebACL.

    1. For a CloudFront distribution:
        1. Go to the CloudFront console and click on the name of the relevant distribution
        2. Under the **General** tab, make note of the value in the AWS WAF Web ACL field
        3. Go to the AWS WAF and Shield console and select Web ACLs from the navigation pane
        4. Locate the Web ACL with the same name as noted from step (ii) and click on the name

2. For an Application Load Balancer:
    1. Go to the EC2 console and select “Load Balancers” from the “Load Balancing” menu in the navigation pane
    2. Select the name of the relevant Application Load Balancer and in the tabs below, select “Integrated services”
    3. Note the name of the Web ACL listed under **AWS WAF**

2. If it is determined that the CloudFront distribution or Application Load Balancer are not protected by a Web ACL, take one of the two following steps:

    1. If a Web ACL already exists and your CMDB shows that this Web ACL should be associated to either the CloudFront Distribution or the Application Load Balancer, go to the AWS console or AWS Command Line Interface (AWS CLI) tools and associate that resource to the Web ACL.
    2. If no reference exists to a Web ACL in the CMDB that should be associated to either the CloudFront Distribution or Application Load Balancer, there are now two possible options:
        1. If the CMDB contains references to known Web ACL rules that should be contained in a Web ACL, go ahead and create those rules and add them to a new Web ACL. Finally, associate that Web ACL to the resources to be protected.
        2. If the CMDB does not contain references to Web ACL rules that should be deployed, the quickest solution is to deploy a set of AWS WAF Managed Rules. These are available in the AWS Marketplace for third-party providers (https://aws.amazon.com/marketplace/solutions/security/waf-managed-rules). There are also AWS-Managed rules available directly from AWS when you create a new Web ACL (https://aws.amazon.com/blogs/aws/announcing-aws-managed-rules-for-aws-waf/).
        3. Alternatively, if the application developers know what type of rules should be deployed to protect the application, work with those developers to create the rules (https://docs.aws.amazon.com/waf/latest/developerguide/classic-web-acl-rules-creating.html), deploy the Web ACL and then go back to step (a), above.

Part 4: Recover from the Incident

The next step is to determine if the attack has been mitigated, or if additional tuning is needed to eradicate the attack. This includes reviewing pre- and post-mitigation logs to:

- Determine the impact of the mitigations already performed
- Identify attack signatures to attempt to eradicate or further mitigate the attack

From points (9) and (10) in PART 1, review the logs obtained once the incident was detected. Now that mitigations are in place, you need to go back and obtain those metrics and logs again, and compare them to the incident metrics and logs obtained earlier. If request activity has returned to baseline levels, and the service is confirmed to be available, the attack has been mitigated: Continue to monitor the service post-attack. If the service request rate is still elevated and the service is available, the attack has been mitigated: Continue to monitor the service for potential change in attack method. If the service is unavailable, return to Part 1 and review logs and related data to determine if you have correctly identified the attack vector, or if the attacker has changed the vector in response to the mitigation.

Part 5: Post-Incident Activity

This “sharpen the saw” activity helps teams to assess their response to the actual incident, determine what worked and what didn’t, update the process based on that information and record these findings.

1. Review the incident handling and the incident handling process with key stakeholders identified in Part 1, Step 6.
2. Document lessons learned, including attack vector(s) mitigation(s), misconfiguration, etc.
3. Store the artifacts from this process with the application information in the CMDB entry for the application and also in the CMDB entry for the DDoS Response process.
4. Update risk documents based on any newly discovered threat/vulnerability combinations that were discovered as a result of lessons learned
5. If new application or infrastructure configuration is required to mitigate any newly identified risks, conduct these change activities and update application and component configuration in the CMDB