Implementing a bot control strategy on AWS

AWS Prescriptive Guidance

AWS Prescriptive Guidance: Implementing a bot control strategy on AWS

Table of Contents

Implementing a bot control strategy on AWS

Amazon Web Services (AWS)

February 2024 (document history)

The internet as we know it would not be possible without bots. *Bots* run automated tasks over the internet and simulate human activity or interaction. They allow businesses to build efficiency into processes and tasks. Useful bots, like web crawlers, index information on the internet and help us quickly find the most relevant information for our search queries. Bots are a good mechanism to improve business and provide value to companies. However, with time, bad actors started using bots as a means to abuse existing systems and applications in new and creative ways.

Botnets are the best-known mechanism to scale bots and their impact. *Botnets* are networks of bots that are infected by malware and are under the control of a single party, known as the *bot herder* or *bot operator*. From one central point, the operator can command every computer on its botnet to simultaneously carry out a coordinated action, which is why botnets are also referred to as *command-and-control (C2) systems*.

The scale of a botnet can be many millions of bots. A botnet helps the operator to perform large-scale actions. Because botnets remain under the control of a remote operator, infected machines can receive updates and change their behavior on the fly. As a result, for significant financial gain, C2 systems can rent access to segments of their botnet on the black market.

The prevalence of botnets has continued to grow. It is considered by experts to be the favorite tool of bad actors. Mirai is one of the biggest botnets. It emerged in 2016, is still operational, and is estimated to have infected up to 350,000 Internet of Things (IoT) devices. This botnet has been adapted and used for many types of activities, including distributed denial of service (DDoS) attacks. More recently, bad actors tried to further obfuscate their activity and source their traffic by obtaining IP addresses through the use of residential proxy services. This creates a legitimate interconnected, peer-to-peer system that adds sophistication to the activity and makes it more challenging to detect and mitigate.

This document focuses on the bot landscape, its effect on your applications, and the available strategies and mitigation options. This prescriptive guidance and its best practices help you understand and mitigate different types of bot attacks. In addition, this guide describes the AWS services and features that support a bot mitigation strategy and how each one can help you protect your applications. It also includes an overview of bot monitoring and best practices for optimizing solution costs.

Understanding bot threats and operations

According to Security Today, more than 47% of all traffic on the internet is due to bots. This includes the helpful portion of bots, those that self-identify and provide value. About 30% of bot traffic is unidentified bots that are performing malicious activities, such as DDoS attacks, ticket scalping, inventory scraping, or hoarding. Security Magazine reports a 300% increase in volumetric DDoS events during the first half of 2023. This makes this topic more relevant, and it makes knowledge about the available preventative and protective tools and technology all the more important.

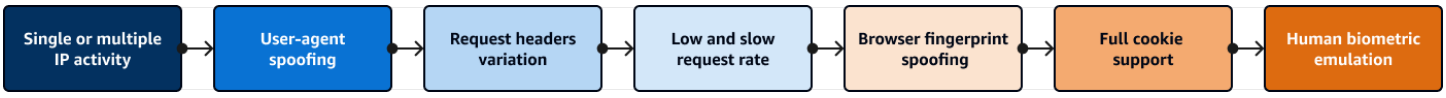
The following table categorizes the different types of bot activity and the business impact each one can have. This is not meant to be an extensive list; it is a summary of the most common bot activities. It highlight the importance of monitoring and mitigation controls. For an extensive list of bot threats, visit the OWASP Automated threats to applications handbook (OWASP website).

Bot activity type	Description	Potential impact
Content scraping	Copying of proprietary content for use by third-party sites	Impact to your SEO due to content duplication, brand impact, and performance problems caused by aggressiv e scrapers
Credential stuffing	Testing of stolen credentia l databases in your website to obtain access or validate information	Problems for users, such as fraud and account lockouts, which increase support queries and decrease brand trust
Card cracking	Testing databases of stolen credit card data to validate or complement missing information	Problems for users, such as identity theft and fraud, and damage to your fraud score
Denial of service	Increasing traffic to a specific website to slow down	Loss of revenue and damage to reputation

Bot activity type	Description	Potential impact
	response or make it unavailab le for legitimate traffic	
Account creation	Creation of multiple accounts with the purpose of misuse or financial gain	Hindered growth and skewed marketing analytics
Scalping	Obtaining limited availability goods, frequently tickets, over genuine consumers	Loss of revenue and problems for users, such as lack of access to goods being sold

How botnets operate

The tactics, techniques, and procedures (TTP) of botnet operators have evolved substantially over time. They have had to keep up with the detection and mitigation technologies developed by companies. The following figure shows this evolution. Botnets started simply by using IP addresses as a means of operation, and they eventually evolved to use sophisticated, human biometric emulation. This sophistication is expensive, and not all botnets use the most advanced tools. There are a mix of operators in the internet, and they likely evaluate the best tool for the job to provide a good return on investment. One goal in bot defense is to make the botnet activity expensive so that the target is no longer viable.



Generally, bots are categorized as common or targeted:

- **Common bots** – These bots self-identify and won't attempt to emulate browsers. Many of these bots perform useful tasks, such as content crawling, search engine optimization (SEO), or aggregation. It's important to identify and understand which of these common bots come to your site and the effect they have on your traffic and performance.
- **Targeted bots** – These bots try to evade detection by emulating browsers. They use browser technology, such as headless browsers, or they fake browser fingerprints. They have the ability to execute JavaScript and support cookies. Their intent is not always clear, and the traffic they generate can look like normal user traffic.

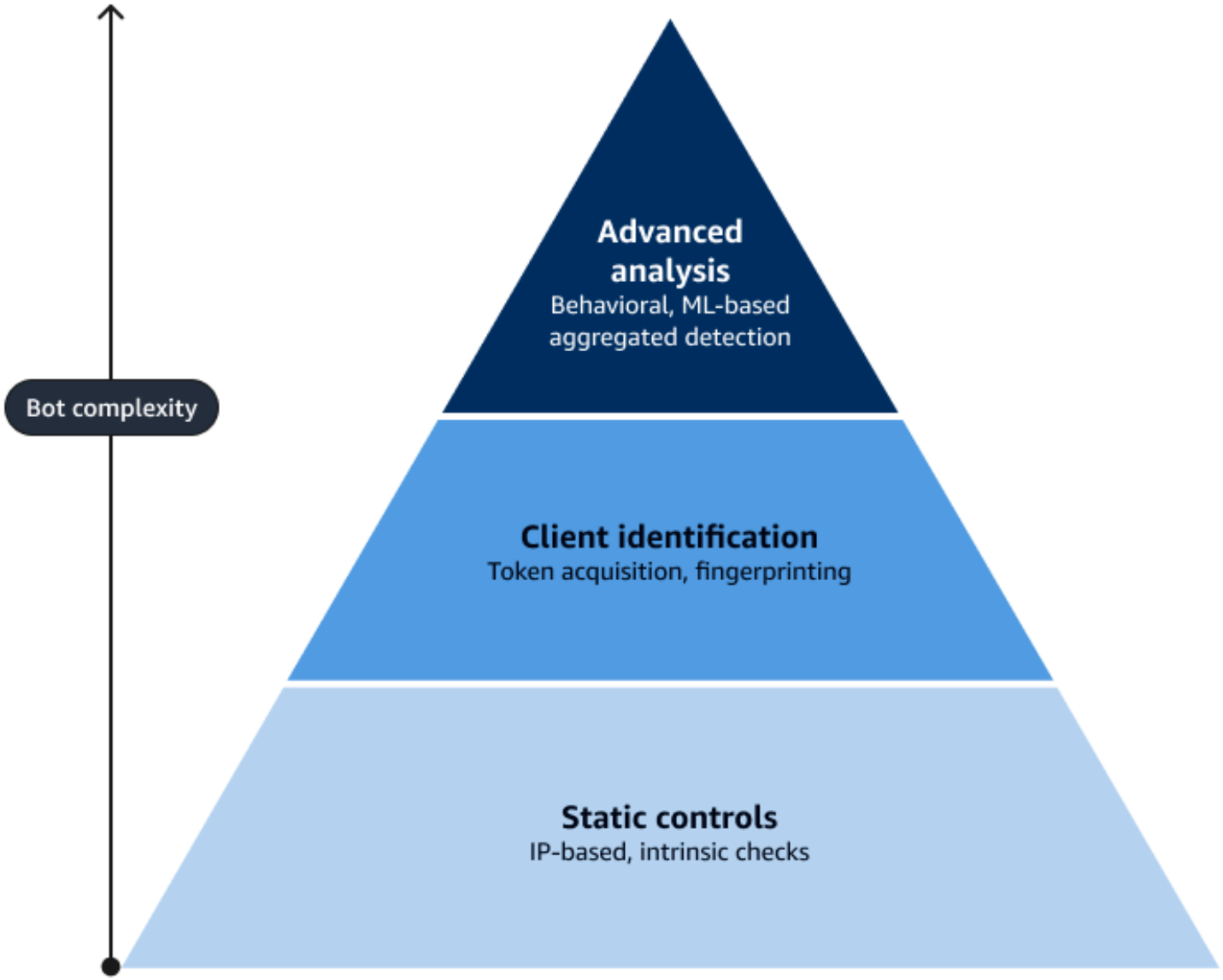
The most advanced and persistent targeted bots emulate human behavior by generating human-like mouse movements and clicks on a website. They are the most sophisticated and difficult to detect, but they are also the most expensive to operate.

Often, an operator combines these techniques. This creates a game of constant pursuit, where you have to frequently change the protection and mitigation approach in order to adapt to the operator's latest techniques. These bots are considered to be an *advanced persistent threat (APT)*. For more information, see Advanced persistent threat in the NIST resource center.

Techniques for bot control

The main goal of bot mitigation is limiting the negative impact of automated bot activity on an organization's web sites, services, and applications. The technology and techniques used depend on the type of traffic or activity you want to defend against. Understanding the application and its traffic is key to accomplishing this. For more information on where to start, see the Guidelines for monitoring your bot control strategy section in this guide.

In general, the controls that bot mitigation solutions provide can be grouped into the following high-level categories: static, client identification, and advanced analysis. The following figure shows the different techniques available and how they can be used depending on the bot activity complexity. This highlights how the base, or the broadest mitigation, can be obtained through the use of static controls, such as allow listing and intrinsic checks. The smallest portion of bots is always the most advanced, and mitigating against these bots requires more advanced technology and a combination of controls.



Next, this guide explores each category and its techniques. It also describes the options that are available in AWS WAF to implement these controls:

- Static controls for managing bots
- Client identification controls for managing bots
- Advanced analysis controls for managing bots

Static controls for managing bots

To take an action, *static controls* evaluate static information from the HTTP(S) request, such as its IP address or its headers. These controls can be useful for low-sophistication bad bot activity or for

expected beneficial bot traffic that needs to be verified and managed. Static control techniques include: allow listing, IP-based controls, and intrinsic checks.

Allow listing

Allow listing is a control that allows identified friendly traffic through existing bot mitigation controls. There are a variety of ways of accomplishing this. The simplest is to use a rule that matches a set of IP addresses or a similar match condition. When a request matches a rule that is set to an `Allow` action, it is not evaluated by subsequent rules. In some cases, you need to prevent only certain rules from being acted on; in other words, you need to allow list for one rule but not all rules. This is a common scenario for handling false positives for rules. Allow listing is considered a broad-scope rule. To reduce the potential for false negatives, we recommend that you pair it with another option that is more granular, such as a path or header match.

IP-based controls

Single IP address blocks

A commonly used tool to mitigate the impact of bots is to limit requests from a single requestor. The simplest example is to block the source IP address of the traffic if its requests are malicious or high in volume. This uses AWS WAF IP set match rules to implement IP-based blocks. These rules match on IP addresses and apply an action of `Block`, `Challenge`, or `CAPTCHA`. You can determine when too many requests are coming in from an IP address by looking at Content Delivery Network (CDN), a web application firewall, or application and service logs. However, in most cases, this control is impractical without automation.

Automating IP address block lists in AWS WAF is commonly done with rate-based rules. For more information, see Rate-based rules in this guide. You can also implement the Security Automations for AWS WAF solution. This solution automatically updates a list of IP addresses to block, and an AWS WAF rule denies requests that match those IP addresses.

One way to recognize a bot attack is if a multitude of requests from the same IP address focus on a small number of web pages. This indicates that the bot is price scrapping or repeatedly attempting logins that fail at a high percentage. You can create automations that immediately recognize this pattern. The automations block the IP address, which reduces the efficacy of the attack by quickly identifying and mitigating it. Blocking specific IP addresses is less effective when an attacker has a large collection of IP addresses to launch attacks from or when the attacking behavior is difficult to recognize and separate from normal traffic.

IP address reputation

An *IP reputation service* provides intelligence that helps evaluate the trustworthiness of an IP address. This intelligence is commonly derived by aggregating IP-related information from past activity from that IP address. Prior activity helps indicate how likely an IP address is to generate malicious requests. The data is added to managed lists that track the IP address behavior.

Anonymous IP addresses are a specialized case of IP address reputation. The source IP address originates from known sources of easily acquired IP addresses, such as cloud-based virtual machines, or from proxies, such as known VPN providers or Tor nodes. The AWS WAF Amazon IP reputation list and Anonymous IP list managed rule groups use Amazon internal threat intelligence to help identify these IP addresses.

The intelligence provided by these managed lists can help you act on activity identified from these sources. Based on this intelligence, you can create rules that directly block traffic or rules that limit the number of requests (such as rate-based rules). You can also use this intelligence to evaluate the source of the traffic by using the rules in `COUNT` mode. This examines the match criteria and applies labels that you can use to create custom rules.

Rate-based rules

Rate-based rules can be a valuable tool for certain scenarios. For example, rate-based rules are effective when bot traffic reaches high volumes compared to users in sensitive uniform resource identifiers (URIs) or when the traffic volume begins to affect normal operations. Rate limiting can keep requests at manageable levels and limit and control access. AWS WAF can implement rate-limiting rule in a web access control list (web ACL) by using a rate-based rule statement. A recommended approach when using rate-based rules is to include a blanket rule that covers the whole site, URI-specific rules, and IP reputation rate-based rules. IP reputation rate-based rules combines the intelligence of IP address reputation with rate-limiting functionality.

For the whole site, a blanket IP reputation rate-based rule creates a ceiling that prevents unsophisticated bots from flooding a site from a small number of IPs. Rate limiting is especially recommended for protecting URIs that have high cost or impact, such as login or account-creation pages.

Rate-limiting rules can provide a cost-efficient first layer of defense. You can use more advanced rules to protect sensitive URIs. URI-specific rate-based rules can limit the impact on critical pages or on APIs that affect the backend, such as database access. Advanced mitigations to protect certain URIs, which are discussed later in this guide, often incur additional costs, and these URI-

specific rate-based rules can help you control costs. For more information about commonly recommended rate-based rules, see The three most important AWS WAF rate-based rules in the AWS Security Blog. In some situations, it is useful to limit what type of request is evaluated by a rate-based rule. You can use scope-down statements to, for example, limit rate-based rules by the geographic area of the source IP address.

AWS WAF offers an advanced capability for rate-based rules through the use of aggregation keys. With this functionality, you can configure a rate-based rule to use various other aggregation keys and key combinations, aside from the source IP address. For example, as a single combination, you can aggregate requests based on a forwarded IP address, the HTTP method, and a query argument. This helps you configure more fine-grained rules for sophisticated volumetric traffic mitigation.

Intrinsic checks

Intrinsic checks are various types of internal or inherent validations or verifications within a system or process. For bot control, AWS WAF performs an intrinsic check by validating that the information sent in the request matches the system signals. For example, it performs reverse DNS lookups and other system verifications. Some automated requests are necessary, such as SEO-related requests. Allow listing is a way to permit good, expected bots through. But sometimes, malicious bots emulate good bots, and it can be challenging to separate them. AWS WAF provides methods to accomplish this through the managed AWS WAF Bot Control rule group. The rules in this group provide verification that self-identified bots are who they say they are. AWS WAF checks the details of the request against the known pattern of that bot, and it also performs reverse DNS lookups and other objective verifications.

Client identification controls for managing bots

If attack-related traffic cannot be easily recognized through static attributes, then detection needs to be able to accurately identify the client making the request. For example, rate-based rules are often more effective and harder to evade when the attribute being rate-limited is application-specific, such as a cookie or token. Using a cookie tied to a session prevents botnet operators from being able to duplicate similar request flows across many bots.

Token acquisition is commonly used for client identification. For token acquisition, a JavaScript code collects information to generate a token that is evaluated on the server side. The evaluation can range from verifying that JavaScript is running on the client to collecting device information for fingerprinting. Token acquisition requires integrating a JavaScript SDK into the site or application, or it requires that a service provider does the injection dynamically.

Requiring JavaScript support adds an additional hurdle for bots attempting to emulate browsers. When an SDK is involved, such as in a mobile application, token acquisition verifies the SDK implementation and prevents bots from mimicking the application's requests.

Token acquisition requires the use of SDKs implemented on the client side of the connection. The following AWS WAF features provide a JavaScript-based SDK for browsers and a application-based SDK for mobile devices: Bot Control, Fraud Control account takeover prevention (ATP) and Fraud Control account creation fraud prevention (ACFP).

The techniques for client identification include CAPTCHA, browser profiling, device fingerprinting, and TLS fingerprinting.

CAPTCHA

Completely automated public Turing test to tell computers and humans apart (CAPTCHA) is used to distinguish between robotic and human visitors and to prevent web scraping, credential stuffing, and spam. There are a variety of implementations, but they often involve a puzzle that a human can solve. CAPTCHAs offer an additional layer of defense against common bots and can reduce the false positives in bot detection.

AWS WAF allows rules to run a CAPTCHA action against web requests that match a rule's inspection criteria. This action is the result of the evaluation of client identification information collected by the service. AWS WAF rules can require CAPTCHA challenges to be solved for specific resources that are frequently targeted by bots, such as login, search, and form submissions. AWS WAF can directly serve CAPTCHA through interstitial means or by using an SDK to handle it on the client side. For more information see CAPTCHA and Challenge in AWS WAF.

Browser profiling

Browser profiling is a method of collecting and evaluating browser characteristics, as part of token acquisition, to distinguish real humans using an interactive browser from distributed bot activity. You can perform browser profiling passively through headers, header order, and other characteristics of requests that are inherent to how browsers work.

You can also perform browser profiling in code by using token acquisition. By using JavaScript for browser profiling, you can quickly determine if a client supports JavaScript. This helps you detect simple bots that do not support it. Browser profiling checks more than just HTTP headers and JavaScript support; browser profiling makes it difficult for bots to fully emulate a web browser.

Both browser profiling options have the same goal: to find patterns in a browser profile that indicate inconsistency with how a real browser behaves.

AWS WAF bot control for targeted bots provides an indication, as part of token evaluation, of whether a browser shows evidence of automation or inconsistent signals. AWS WAF flags the request in order to take the action specified in the rule. For more information, see Detect and block advanced bot traffic in the AWS Security Blog.

Device fingerprinting

Device fingerprinting is similar to browser profiling, but it is not limited to browsers. Code running on a device (which can be a mobile device or a web browser) collects and reports details of the device to a backend server. The details can include system attributes, such as memory, CPU type, operating system (OS) kernel type, OS version, and virtualization.

You can use device fingerprinting to recognize if a bot is emulating an environment or if there are direct signs that automation is in use. Beyond this, device fingerprinting can also be used to recognize repeated requests from the same device.

Recognizing repeated requests from the same device, even if the device tries to change some characteristics of the request, allows a backend system to impose rate-limiting rules. Rate-limiting rules that are based on device fingerprinting are typically more effective than rate-limiting rules based on IP addresses. This helps you mitigate against bot traffic that is rotating between VPNs or proxies but is sourced from a small number of devices.

When used with application integration SDKs, AWS WAF bot control for targeted bots, can aggregate client session request behavior. This helps you detect and separate legitimate client sessions from malicious client sessions, even when both originate from the same IP address. For more information about AWS WAF bot control for targeted bots, see Detect and block advanced bot traffic in the AWS Security Blog.

TLS fingerprinting

TLS fingerprinting, also known as *signature-based rules,* are commonly used when bots originate from many IP addresses but exhibit similar characteristics. When using HTTPS, the client and server sides exchange messages to acknowledge and verify one another. They establish cryptographic algorithms and sessions keys. This is called a *TLS handshake*. How a TLS handshake is implemented is a signature that is often valuable for recognizing large attacks spread across many IP addresses.

TLS fingerprinting enables web servers to determine a web client's identity with a high degree of accuracy. It requires only the parameters in the first packet connection, before any application data exchange occurs. In this case, *web client* refers to the application initiating a request, which might be a browser, CLI tool, script (bot), native application, or other client.

One SSL and TLS fingerprinting approach is JA3 fingerprint. JA3 fingerprints a client connection based on fields in the Client Hello message from the SSL or TLS handshake. It helps you profile specific SSL and TLS clients across different source IP addresses, ports, and X.509 certificates.

Amazon CloudFront supports adding JA3 headers to requests. A `CloudFront-Viewer-JA3-Fingerprint` header contains a 32-character hash fingerprint of the TLS Client Hello packet of an incoming viewer request. The fingerprint encapsulates information about how the client communicates. This information can be used to profile clients that share the same pattern. You can add the `CloudFront-Viewer-JA3-Fingerprint` header to an origin request policy and attach the policy to a CloudFront distribution. You can then inspect the header value in origin applications or in Lambda@Edge and CloudFront Functions. You can compare the header value against a list of known malware fingerprints to block the malicious clients. You can also compare the header value against a list of expected fingerprints to allow requests only from known clients.

Advanced analysis controls for managing bots

Some bots employ advanced deception tools to actively evade detection. These bots mimic human behavior in order to perform a specific activity, such as scalping. These bots have a purpose, and it is usually linked to a big monetary reward.

These advanced, persistent bots use a mix of technologies to evade detection or blend in with regular traffic. In turn, this also requires a mix of different detection technologies to accurately identify and mitigate the malicious traffic.

Targeted use cases

Use-case data can provide bot-detection opportunities. *Fraud detections* are special use cases where special mitigation is warranted. For example, to help prevent account takeovers, you can compare a list of compromised account usernames and passwords against login or account creation requests. This helps website owners to detect login attempts that use compromised credentials. Use of compromised credentials can indicate bots trying take over an account, or it could be users who are unaware their credentials are compromised. In this use case, website owners can take additional steps to verify the user and then help them change their password. AWS WAF provides the Fraud Control account takeover prevention (ATP) managed rule for this use case.

Application-level or aggregated bot detection

Some use cases require combining data about requests from the content delivery network (CDN), AWS WAF, and the backend of the application or service. Sometimes, you even need to integrate third-party intelligence to be able to make high-confidence decisions about bots.

Features in Amazon CloudFront and AWS WAF can send signals to the backend infrastructure, or they can subsequently aggregate rules through headers and labels. CloudFront exposes JA3 fingerprint headers, as previously mentioned. This is an example of CloudFront providing such data through a header. AWS WAF can send labels when it matches on a rule. Subsequent rules can use these labels to make better decisions about bots. When multiple rules are combined, you can implement highly granular controls. A common use case is to match on parts of a managed rule through a label and then combine it with other request data. For more information, see Label match examples in the AWS WAF documentation.

Machine learning analysis

Machine leaning (ML) is a powerful technique for dealing with bots. ML can adapt to changing details, and when combined with other tools, provides the most robust and complete way to mitigate bots with minimal false positives. The two most common ML techniques are *behavioral analysis* and *anomaly detection*. With behavioral analysis, a system (in the client, server, or both) monitors how a user interacts with the application or website. It monitors mouse movement patterns or frequency of click and touch interactions. The behavior is then analyzed with a ML model to recognize bots. Anomaly detection is similar. It focuses on detecting behavior or patterns that are significantly different from a baseline that is defined for the application or website.

AWS WAF targeted controls for bots provides predictive ML technology. This technology helps defend against distributed, proxy-based attacks that are made by bots designed to evade detection. The managed AWS WAF Bot Control rule group uses automated, ML analysis of website traffic statistics to detect anomalous behavior that is indicative of distributed, coordinated bot activity.

Deployment and implementation of your bot control strategy

There are multiple factors to consider when planning a bot control deployment strategy. In addition to the unique characteristics of web applications, environment size, development process, and organizational structure affect the deployment strategy. Depending on your environment and application characteristics, a centralized or decentralized deployment strategy can be used:

- **Centralized deployment strategy** – A centralized approach enables a higher degree of control when you want strict enforcement of bot control. This approach is well suited if application teams prefer to offload management. A centralized approach is most effective when web applications share similar characteristics. In this case, the applications benefit from a common set of bot control rules and bot mitigation actions.
- **Decentralized deployment strategy** – A decentralized approach provides application teams with autonomy to define and implement bot control configurations independently. This approach is common for smaller environments or when application teams need to retain control over their bot control policies. Due to the nature of many web applications, it is often necessary to maintain independent bot control policies that are tailored for unique application characteristics, resulting in a decentralized approach.
- **Combined strategy** – A combination of these two approaches is appropriate for a mix of web applications. For example, this might entail a set of base rules that applies to all web ACLs, while management of more specific bot control policies is delegated to application teams.

You can use AWS Firewall Manager to centralize and automate deployment of AWS WAF web ACLs that define bot control policies. When using Firewall Manager, consider whether it is appropriate to centralize bot control policies, including if they should be delegated to application teams. With Firewall Manager, you can use tagging to allow application teams to opt-in for AWS WAF policies. This provides AWS WAF with intelligent threat mitigation functionality. You can also enable centralized AWS WAF logging for application and security operations.

Regardless of the deployment strategy used, it is recommended to define and manage the onboarding process through infrastructure as code (IaC)-based frameworks, such as AWS CloudFormation or the AWS Cloud Development Kit (AWS CDK). This helps you configure source control to store and version configuration objects. For more information, see AWS WAF configuration samples for AWS CDK (GitHub) and CloudFormation (AWS documentation).

Implementation strategy

After you have selected a deployment strategy, implementation can begin. The deployment strategy defines how rules are rolled out to different applications. In the implementation strategy, the focus is on the iterative process of adding controls, testing, continuously monitoring, and then evaluating their effects.

Understanding traffic patterns

To really understand traffic patterns, it is important to familiarize yourself with the application's business function and expected attributes, such as usage patterns, key resources, and user personas. Incorporate production traffic and traffic generated during testing against the application to establish a baseline for the evaluation. Make sure that the timeframe includes traffic data that sufficiently represents multiple usage peaks.

Using your preferred tool, review the traffic logs and metrics over the representative usage period. Analyze the AWS WAF log data for anomalous requests by filtering on log fields such as `headers` (for example, `User-Agent` and `Referer`), `country`, and `clientIp`. Make note of uniform resource identifiers (URIs) and their access frequency. Categorize traffic, such as identifying good bots. For example, permit access for beneficial bots, such as search engine crawlers and monitors.

In the AWS WAF console, on the **Bot control dashboard**, a sample of bot activity is available for any active web ACL. Although this provides an initial perspective of common bot request volumes, perform further configuration and analysis to better understand bot activity.

For an effective implementation, you must have a good understanding of bot traffic, its effects, and which bot requests are beneficial vs. malicious. This helps with the next phase, selecting controls, and helps you evaluate bot traffic in parallel.

Selecting and adding controls

The initial traffic analysis helps determine which bot controls to use and what actions to select for each. You might also choose to log and monitor activity for potential future action. The initial traffic analysis help you select the best control to manage the traffic. For more information about the available controls, see Techniques for bot control in this guide.

Consider including additional SDK implementations during this step. This helps you test and complete SDK implementations in all required applications. AWS WAF bot control and fraud control rules provide a full token evaluation benefit when you implement JavaScript SDK or mobile SDK.

For more information, see Why you should use the application integration SDKs with Bot Control in the AWS WAF documentation.

We recommend implementing token acquisition for different application types as follows:

- **Single-page application (SPA)** – JavaScript SDK (no redirect)
- **Mobile browser** – JavaScript SDK or rule actions (CAPTCHA or Challenge)
- **Web views** – JavaScript SDK or rule actions (CAPTCHA or Challenge)
- **Native applications** – Mobile SDK
- **iFrames** – JavaScript SDK

For more information about how to implement the SDKs, see AWS WAF client application integration in the AWS WAF documentation.

Testing and deploying to production

The controls should be initially deployed in a non-production environment where you can perform testing to verify that the expected web application functionality is preserved. Always perform a thorough validation in a test environment prior to production deployment.

After testing and validation in a non-production environment, the production release can proceed. Select a date and time with the lowest expected user traffic. Before deployment, the application and security teams should review operational readiness, discuss how to roll back changes, and review dashboards to ensure all required metrics and alarms are configured.

With Amazon CloudFront continuous deployment, you can send a small amount of traffic to a staging distribution that has an AWS WAF web ACL configured specifically for bot control evaluation. AWS WAF provides version management of any new or updated managed rules so that you can test and approve changes before they start evaluating production traffic.

Evaluating and tuning controls

Implemented controls can provide further insight and visibility into traffic activity and patterns. Frequently monitor and analyze application traffic in order to add or adjust security controls. There is normally a phase of tuning to mitigate potential false negatives and false positives. *False negatives* are attacks that were not caught by your controls and require you to harden your rules. *False positives* represent legitimate requests that were incorrectly identified as attacks and blocked as a consequence.

The analysis and tuning can be done manually or with the help of tools. A Security Information and Event Management (SIEM) system is a common tool that helps provide metrics and intelligent monitoring. There are many available with varying degrees of sophistication, but they all provide a good starting point to obtain traffic insights.

Defining important key performance indicators (KPIs) for websites and applications can help you more quickly identify when things are not working as expected. For example, you can use credit card charge backs, sales per account, or conversion rates as indicators of business anomalies that can be generated by bots. Defining and understanding which metrics and KPIs are valuable to monitor is even more important than just the act of monitoring.

Understanding how to get the right metrics and logs from a bot control solution is just as important as identifying the metrics to monitor. The next section, Guidelines for monitoring your bot control strategy, details monitoring and visibility options to consider.

Guidelines for monitoring your bot control strategy

For bot traffic and web application traffic, monitoring and visibility is of great importance. It helps you prioritize activities as well as security operations. If detailed logging or using a SIEM system are not possible, then a good starting point is monitoring basic metrics provided by your selected solution or vendor.

This visibility is useful for threat intelligence, hardening rules, troubleshooting false positives, and responding to an incident. There are multiple monitoring options available with AWS WAF. For high-level monitoring, AWS WAF provides traffic overview information in the AWS Management Console. This is available for all traffic as well as a detailed view for bot traffic, when the Bot Control rule group is enabled in your web ACL.

AWS WAF provides different options for detailed logging of web ACL traffic. You can also add labels to requests, which you can use to facilitate log analysis and configure bot evaluation rules. By integrating Amazon CloudWatch Logs Insights, you can query the AWS WAF logs and visualize the results.

If you turn on detailed logging, AWS WAF provides additional visibility beyond the preconfigured **Bot control dashboard**. Using AWS WAF logs to visualize traffic, as well as ad-hoc investigations, can provide in-depth understanding of traffic patterns and options for mitigation for a web application.

You can integrate AWS WAF log data with Amazon CloudWatch Logs, Amazon Simple Storage Service (Amazon S3), or Amazon Data Firehose. For more information, see Turn on AWS WAF logging and send logs to CloudWatch, Amazon S3, or Amazon Data Firehose. You can also send logs to various targets for analysis, including to Amazon OpenSearch Service or an AWS Marketplace solution. For more information, see Destination settings in the Firehose documentation. If multiple log sources are used, a centralized logging solution is recommended to correlate sources.

Next, this guide provides recommendations for how to start monitoring bot traffic and gain visibility by using Amazon CloudWatch.

Tracking top rules

Tracking the top-hit rules can highlight trends and potentially anomalous activities. Increased rates for a specific rule might indicate a potential false positive or targeted activity that you should

investigate. The most common rule for tracking would be IP-based controls, geo-blocking rules (a spike here can show traffic from unusual countries, which might not be automatically blocked), and Rate-based rules. These rules would always have inherent variation, but an anomaly in the traffic pattern can be indicative of bot activity. Take this into consideration if you're manually setting the thresholds.

Tracking top labels and namespaces

By using CloudWatch metrics to track the top labels, you can see which AWS WAF rules are frequently being invoked. This helps you detect anomalies, such as an increase in scraper activity, traffic from suspicious sources, or attempted abuse of the application login page or API.

The following are example labels that might be of interest:

- `awswaf:managed:aws:bot-control:signal:non_browser_user_agent`
- `awswaf:managed:aws:bot-control:bot:category:http_library`
- `awswaf:managed:aws:bot-control:bot:name:curl`
- `awswaf:managed:aws:atp:signal:credential_compromised`
- `awswaf:managed:aws:core-rule-set:NoUserAgent_Header`
- `awswaf:managed:token:rejected`

The following are example label namespaces that might be of interest:

- `awswaf:managed:aws:bot-control:`
- `awswaf:managed:aws:atp:`
- `awswaf:managed:aws:anonymous-ip-list:`

Creating math expressions

In Amazon CloudWatch, you can create math expressions for any or all rules. If you set alerts on math expressions, you will be notified regarding anomalies in rates, not quantities, of certain metrics. This is an important tool to reduce alert fatigue.

Create a custom metric built out of a math expression. Look at the relative rates for rules, out of the overall number of requests to an application. The following is a common math expression:

```
[ruleX count * 100]/[All allowed requests + All blocked requests]
```

This math expression provides a percentage so that you can track a specific rule and visualize its trend over time.

Using anomaly detection

Using CloudWatch anomaly detection on any CloudWatch metric can provide alerts on abnormally low or high trends, without setting up the actual threshold manually. These algorithms continuously analyze metrics of systems and applications, determine normal baselines, and surface anomalies with minimal user intervention. CloudWatch applies statistical and ML algorithms in its anomaly detection feature.

Using Amazon CloudWatch metrics

AWS WAF processes traffic and adds labels to requests that match the rules defined in the web ACL. Each label creates a metric in CloudWatch. At the same time, each web ACL rule also creates metrics for each of its possible actions. Use these label and action metrics to gain a high-level understanding of bot traffic. This is a cost-effective approach to visualizing trends. For more information, see View available metrics and Graphing metrics in the CloudWatch documentation.

CloudWatch provides the option to send data to a log collector or aggregator, be it an AWS service or a third-party solution. Ingesting data from CloudWatch can provide a more consolidated security observability experience, where you can correlate data from multiple sources. This can help you investigate, view, or set up your alerts and security automations.

Building a dashboard

After identifying the important metrics to track, create a dashboard that contains the most relevant metrics. Displaying them side-by-side, under a single pane of glass can provide additional visibility and control.

It is always preferable to configure alerts and automation rules for anomalous metric values. Do not rely on humans to identify anomalies by looking at a dashboard. However, dashboards can be useful for investigation purposes after an alert has been received.

Optimizing costs for your bot control strategy

The nature of web traffic is dynamic. This means that the technology and services used to mitigate threats can vary and be tuned over time. This is key when considering a bot control strategy and the controls included in it. Optimization over time is the main principle to keep in mind, and it comes from the cost optimization pillar of the AWS Well-Architected Framework.

AWS WAF web ACLs can be dynamic, especially when new features are released or you are trying to mitigate a new threat. Keeping an eye on your costs involves understanding the cost dimensions of the AWS WAF service and how each affects your final spend. The main driving cost is the number of requests evaluated by the service. There are additional charges if you use the Bot Control and account takeover prevention (ATP) managed rule groups or if you use advanced actions, such as CAPTCHA or challenge.

Because specialized bot controls come at a premium cost, the primary cost optimization goal is to reduce the number of requests inspected by these advanced controls. Applicable techniques include separating high-value content, applying lower-cost measures first, scoping down the area of evaluation, and combining bot protection with other types of controls. Cost monitoring techniques provide additional visibility across your organization.

Separating dynamic and static content

One cost reduction technique is isolating the static content from the dynamic application. The majority of requests to typical web applications are requests to static objects. A common method to reduce the load on application servers is to move static content to its own URL, such as `static.example.com`. This is often achieved by creating a unique content delivery distribution with the caching configuration optimized for static content. This technique can also help lower bot control costs if static content is not commonly targeted in the site or application. Separating the static content from the dynamic application can allow for a more precise application of advanced bot controls.

Applying lower-cost rules first

Another technique is to apply lower-cost, baseline rules that filter out unwanted traffic before using advanced controls, which are more expensive. In practice, this commonly means placing bot control mitigations as a last layer of defense and using preceding controls to filter out unwanted

traffic. This pyramid approach was previously discussed in Techniques for bot control in this guide. The main goal is to use these lower-cost options to stop unwanted traffic, which reduces the number of requests processed by advanced, higher-cost mitigation techniques.

Scoping down the area of evaluation

AWS WAF scope-down statements provide a powerful technique for reducing the number of requests inspected by advanced rules. If separating static content into its own URL cannot be implemented, then scope-down statements are another method to filter out requests that do not require advanced mitigation techniques. This can be done by defining a specific application path, an HTTP method (such as POST), or a similar combination.

Combining bot protection with other controls

Additional cost-control considerations should be reviewed when protecting applications against multiple threats in addition to unwanted bot traffic. For example, protecting against distributed denial of service (DDoS) attacks and against account takeover require additional configuration that can affect costs. Shield Advanced is recommended to help protect applications against DDoS attacks. In particular, its application-layer mitigations can automatically address request floods, thus reducing the number of requests that may be processed by the AWS WAF Bot Control rule group, when placing the rule ahead in the evaluation order. Shield Advanced has an additional benefit; standard managed and custom AWS WAF rules come at no additional cost for resources protected by Shield Advanced. Note that intelligent threat mitigation rule groups, including Bot Control, do incur additional costs, even for resources protected by Shield Advanced.

Applications that require account takeover prevention can use the AWS WAF Fraud Control account takeover prevention (ATP) rule group. ATP rule group's per-request inspection cost is higher than that of the Bot Control rule group. That higher cost makes it critical to apply the ATP rule group as precisely as possible. Using the Bot Control rule group in conjunction with ATP can help achieve this objective. The Bot Control rule group should be placed ahead of ATP in the web ACL to filter out bot requests and reduce the number of requests inspected by ATP.

For continuous optimization, the most significant activity is monitoring CloudWatch metrics associated with the Bot Control rule group. The goal over time is to drive down the number of requests evaluated by the Bot Control rule group to only those that target the resources you need to protect against unwanted bot activity. Building CloudWatch dashboards provides visibility of most critical metrics for applications, including AWS WAF costs and usage.

Monitoring costs

AWS Cost Explorer is a tool that enables you to view and analyze your costs and usage. Cost Explorer facilitates analysis of AWS costs, including AWS WAF costs incurred. The tool provides cost information for the most recent 12 months and forecasts future spend for the next 12 months.

AWS Cost Anomaly Detection is another cost management control tool that can be useful for monitoring AWS WAF costs. It uses advanced ML technologies to identify anomalous spend and root causes. This helps you quickly take action or receive alerts if there's an unexpected increase in cost. To receive an alert when a specific cost threshold is reached,AWS Budgets can provide that tracking and monitoring functionality.

Resources

AWS documentation

- AWS WAF developer guide
- AWS Best Practices for DDoS Resiliency (AWS Whitepapers)
- Guidelines for implementing AWS WAF (AWS Whitepapers)

Other AWS resources

- Analyzing AWS WAF Logs in Amazon CloudWatch Logs (AWS blog post)
- Deploy a dashboard for AWS WAF with minimal effort (AWS blog post)
- Security Automations for AWS WAF (AWS Solutions Library)
- The three most important AWS WAF rate-based rules (AWS blog post)
- Visualize AWS WAF logs with an Amazon CloudWatch dashboard (AWS blog post)

Contributors

Authoring

- Diana Alvarado, Senior Solutions Architect, AWS
- Cameron Worrell, Enterprise Architect, AWS
- Geary Scherer, Solutions Architect, AWS
- Tzoori Tamam, Principal Solutions Architect, AWS

Reviewing

- Jess Izen, Senior Software Development Engineer, AWS
- Kaustubh Phatak, Senior Product Manager, AWS
- Vikramaditya Bhatnagar, Senior Security Consultant, AWS

Technical writing

- Lilly AbouHarb, Senior Technical Writer, AWS

Document history

The following table describes significant changes to this guide. If you want to be notified about future updates, you can subscribe to an RSS feed.

Change	Description	Date
Initial publication	—	February 21, 2024

AWS Prescriptive Guidance glossary

The following are commonly used terms in strategies, guides, and patterns provided by AWS Prescriptive Guidance. To suggest entries, please use the **Provide feedback** link at the end of the glossary.

Numbers

7 Rs

Seven common migration strategies for moving applications to the cloud. These strategies build upon the 5 Rs that Gartner identified in 2011 and consist of the following:

- Refactor/re-architect – Move an application and modify its architecture by taking full advantage of cloud-native features to improve agility, performance, and scalability. This typically involves porting the operating system and database. Example: Migrate your on-premises Oracle database to the Amazon Aurora PostgreSQL-Compatible Edition.
- Replatform (lift and reshape) – Move an application to the cloud, and introduce some level of optimization to take advantage of cloud capabilities. Example: Migrate your on-premises Oracle database to Amazon Relational Database Service (Amazon RDS) for Oracle in the AWS Cloud.
- Repurchase (drop and shop) – Switch to a different product, typically by moving from a traditional license to a SaaS model. Example: Migrate your customer relationship management (CRM) system to Salesforce.com.
- Rehost (lift and shift) – Move an application to the cloud without making any changes to take advantage of cloud capabilities. Example: Migrate your on-premises Oracle database to Oracle on an EC2 instance in the AWS Cloud.
- Relocate (hypervisor-level lift and shift) – Move infrastructure to the cloud without purchasing new hardware, rewriting applications, or modifying your existing operations. You migrate servers from an on-premises platform to a cloud service for the same platform. Example: Migrate a Microsoft Hyper-V application to AWS.
- Retain (revisit) – Keep applications in your source environment. These might include applications that require major refactoring, and you want to postpone that work until a later time, and legacy applications that you want to retain, because there's no business justification for migrating them.

- Retire – Decommission or remove applications that are no longer needed in your source environment.

A

ABAC

See attribute-based access control.

abstracted services

See managed services.

ACID

See atomicity, consistency, isolation, durability.

active-active migration

A database migration method in which the source and target databases are kept in sync (by using a bidirectional replication tool or dual write operations), and both databases handle transactions from connecting applications during migration. This method supports migration in small, controlled batches instead of requiring a one-time cutover. It's more flexible but requires more work than active-passive migration.

active-passive migration

A database migration method in which in which the source and target databases are kept in sync, but only the source database handles transactions from connecting applications while data is replicated to the target database. The target database doesn't accept any transactions during migration.

aggregate function

A SQL function that operates on a group of rows and calculates a single return value for the group. Examples of aggregate functions include `SUM` and `MAX`.

AI

See artificial intelligence.

AIOps

See artificial intelligence operations.

anonymization

The process of permanently deleting personal information in a dataset. Anonymization can help protect personal privacy. Anonymized data is no longer considered to be personal data.

anti-pattern

A frequently used solution for a recurring issue where the solution is counter-productive, ineffective, or less effective than an alternative.

application control

A security approach that allows the use of only approved applications in order to help protect a system from malware.

application portfolio

A collection of detailed information about each application used by an organization, including the cost to build and maintain the application, and its business value. This information is key to the portfolio discovery and analysis process and helps identify and prioritize the applications to be migrated, modernized, and optimized.

artificial intelligence (AI)

The field of computer science that is dedicated to using computing technologies to perform cognitive functions that are typically associated with humans, such as learning, solving problems, and recognizing patterns. For more information, see What is Artificial Intelligence?

artificial intelligence operations (AIOps)

The process of using machine learning techniques to solve operational problems, reduce operational incidents and human intervention, and increase service quality. For more information about how AIOps is used in the AWS migration strategy, see the operations integration guide.

asymmetric encryption

An encryption algorithm that uses a pair of keys, a public key for encryption and a private key for decryption. You can share the public key because it isn't used for decryption, but access to the private key should be highly restricted.

atomicity, consistency, isolation, durability (ACID)

A set of software properties that guarantee the data validity and operational reliability of a database, even in the case of errors, power failures, or other problems.

attribute-based access control (ABAC)

The practice of creating fine-grained permissions based on user attributes, such as department, job role, and team name. For more information, see ABAC for AWS in the AWS Identity and Access Management (IAM) documentation.

authoritative data source

A location where you store the primary version of data, which is considered to be the most reliable source of information. You can copy data from the authoritative data source to other locations for the purposes of processing or modifying the data, such as anonymizing, redacting, or pseudonymizing it.

Availability Zone

A distinct location within an AWS Region that is insulated from failures in other Availability Zones and provides inexpensive, low-latency network connectivity to other Availability Zones in the same Region.

AWS Cloud Adoption Framework (AWS CAF)

A framework of guidelines and best practices from AWS to help organizations develop an efficient and effective plan to move successfully to the cloud. AWS CAF organizes guidance into six focus areas called perspectives: business, people, governance, platform, security, and operations. The business, people, and governance perspectives focus on business skills and processes; the platform, security, and operations perspectives focus on technical skills and processes. For example, the people perspective targets stakeholders who handle human resources (HR), staffing functions, and people management. For this perspective, AWS CAF provides guidance for people development, training, and communications to help ready the organization for successful cloud adoption. For more information, see the AWS CAF website and the AWS CAF whitepaper.

AWS Workload Qualification Framework (AWS WQF)

A tool that evaluates database migration workloads, recommends migration strategies, and provides work estimates. AWS WQF is included with AWS Schema Conversion Tool (AWS SCT). It analyzes database schemas and code objects, application code, dependencies, and performance characteristics, and provides assessment reports.

B

bad bot

A bot that is intended to disrupt or cause harm to individuals or organizations.

BCP

See business continuity planning.

behavior graph

A unified, interactive view of resource behavior and interactions over time. You can use a behavior graph with Amazon Detective to examine failed logon attempts, suspicious API calls, and similar actions. For more information, see Data in a behavior graph in the Detective documentation.

big-endian system

A system that stores the most significant byte first. See also endianness.

binary classification

A process that predicts a binary outcome (one of two possible classes). For example, your ML model might need to predict problems such as “Is this email spam or not spam?" or "Is this product a book or a car?"

bloom filter

A probabilistic, memory-efficient data structure that is used to test whether an element is a member of a set.

blue/green deployment

A deployment strategy where you create two separate but identical environments. You run the current application version in one environment (blue) and the new application version in the other environment (green). This strategy helps you quickly roll back with minimal impact.

bot

A software application that runs automated tasks over the internet and simulates human activity or interaction. Some bots are useful or beneficial, such as web crawlers that index information on the internet. Some other bots, known as *bad bots*, are intended to disrupt or cause harm to individuals or organizations.

botnet

Networks of bots that are infected by malware and are under the control of a single party, known as a *bot herder* or *bot operator*. Botnets are the best-known mechanism to scale bots and their impact.

branch

A contained area of a code repository. The first branch created in a repository is the *main branch*. You can create a new branch from an existing branch, and you can then develop features or fix bugs in the new branch. A branch you create to build a feature is commonly referred to as a *feature branch*. When the feature is ready for release, you merge the feature branch back into the main branch. For more information, see About branches (GitHub documentation).

break-glass access

In exceptional circumstances and through an approved process, a quick means for a user to gain access to an AWS account that they don't typically have permissions to access. For more information, see the Implement break-glass procedures indicator in the AWS Well-Architected guidance.

brownfield strategy

The existing infrastructure in your environment. When adopting a brownfield strategy for a system architecture, you design the architecture around the constraints of the current systems and infrastructure. If you are expanding the existing infrastructure, you might blend brownfield and greenfield strategies.

buffer cache

The memory area where the most frequently accessed data is stored.

business capability

What a business does to generate value (for example, sales, customer service, or marketing). Microservices architectures and development decisions can be driven by business capabilities. For more information, see the Organized around business capabilities section of the Running containerized microservices on AWS whitepaper.

business continuity planning (BCP)

A plan that addresses the potential impact of a disruptive event, such as a large-scale migration, on operations and enables a business to resume operations quickly.

C

CAF

See AWS Cloud Adoption Framework.

canary deployment

The slow and incremental release of a version to end users. When you are confident, you deploy the new version and replace the current version in its entirety.

CCoE

See Cloud Center of Excellence.

CDC

See change data capture.

change data capture (CDC)

The process of tracking changes to a data source, such as a database table, and recording metadata about the change. You can use CDC for various purposes, such as auditing or replicating changes in a target system to maintain synchronization.

chaos engineering

Intentionally introducing failures or disruptive events to test a system's resilience. You can use AWS Fault Injection Service (AWS FIS) to perform experiments that stress your AWS workloads and evaluate their response.

CI/CD

See continuous integration and continuous delivery.

classification

A categorization process that helps generate predictions. ML models for classification problems predict a discrete value. Discrete values are always distinct from one another. For example, a model might need to evaluate whether or not there is a car in an image.

client-side encryption

Encryption of data locally, before the target AWS service receives it.

Cloud Center of Excellence (CCoE)

A multi-disciplinary team that drives cloud adoption efforts across an organization, including developing cloud best practices, mobilizing resources, establishing migration timelines, and leading the organization through large-scale transformations. For more information, see the CCoE posts on the AWS Cloud Enterprise Strategy Blog.

cloud computing

The cloud technology that is typically used for remote data storage and IoT device management. Cloud computing is commonly connected to edge computing technology.

cloud operating model

In an IT organization, the operating model that is used to build, mature, and optimize one or more cloud environments. For more information, see Building your Cloud Operating Model.

cloud stages of adoption

The four phases that organizations typically go through when they migrate to the AWS Cloud:

- Project – Running a few cloud-related projects for proof of concept and learning purposes
- Foundation – Making foundational investments to scale your cloud adoption (e.g., creating a landing zone, defining a CCoE, establishing an operations model)
- Migration – Migrating individual applications
- Re-invention – Optimizing products and services, and innovating in the cloud

These stages were defined by Stephen Orban in the blog post The Journey Toward Cloud-First & the Stages of Adoption on the AWS Cloud Enterprise Strategy blog. For information about how they relate to the AWS migration strategy, see the migration readiness guide.

CMDB

See configuration management database.

code repository

A location where source code and other assets, such as documentation, samples, and scripts, are stored and updated through version control processes. Common cloud repositories include GitHub or AWS CodeCommit. Each version of the code is called a *branch*. In a microservice structure, each repository is devoted to a single piece of functionality. A single CI/CD pipeline can use multiple repositories.

cold cache

A buffer cache that is empty, not well populated, or contains stale or irrelevant data. This affects performance because the database instance must read from the main memory or disk, which is slower than reading from the buffer cache.

cold data

Data that is rarely accessed and is typically historical. When querying this kind of data, slow queries are typically acceptable. Moving this data to lower-performing and less expensive storage tiers or classes can reduce costs.

computer vision (CV)

A field of AI that uses machine learning to analyze and extract information from visual formats such as digital images and videos. For example, AWS Panorama offers devices that add CV to on-premises camera networks, and Amazon SageMaker provides image processing algorithms for CV.

configuration drift

For a workload, a configuration change from the expected state. It might cause the workload to become noncompliant, and it's typically gradual and unintentional.

configuration management database (CMDB)

A repository that stores and manages information about a database and its IT environment, including both hardware and software components and their configurations. You typically use data from a CMDB in the portfolio discovery and analysis stage of migration.

conformance pack

A collection of AWS Config rules and remediation actions that you can assemble to customize your compliance and security checks. You can deploy a conformance pack as a single entity in an AWS account and Region, or across an organization, by using a YAML template. For more information, see Conformance packs in the AWS Config documentation.

continuous integration and continuous delivery (CI/CD)

The process of automating the source, build, test, staging, and production stages of the software release process. CI/CD is commonly described as a pipeline. CI/CD can help you automate processes, improve productivity, improve code quality, and deliver faster. For more information, see Benefits of continuous delivery. CD can also stand for *continuous deployment*. For more information, see Continuous Delivery vs. Continuous Deployment.

CV

See computer vision.

D

data at rest

Data that is stationary in your network, such as data that is in storage.

data classification

A process for identifying and categorizing the data in your network based on its criticality and sensitivity. It is a critical component of any cybersecurity risk management strategy because it helps you determine the appropriate protection and retention controls for the data. Data classification is a component of the security pillar in the AWS Well-Architected Framework. For more information, see Data classification.

data drift

A meaningful variation between the production data and the data that was used to train an ML model, or a meaningful change in the input data over time. Data drift can reduce the overall quality, accuracy, and fairness in ML model predictions.

data in transit

Data that is actively moving through your network, such as between network resources.

data mesh

An architectural framework that provides distributed, decentralized data ownership with centralized management and governance.

data minimization

The principle of collecting and processing only the data that is strictly necessary. Practicing data minimization in the AWS Cloud can reduce privacy risks, costs, and your analytics carbon footprint.

data perimeter

A set of preventive guardrails in your AWS environment that help make sure that only trusted identities are accessing trusted resources from expected networks. For more information, see Building a data perimeter on AWS.

data preprocessing

To transform raw data into a format that is easily parsed by your ML model. Preprocessing data can mean removing certain columns or rows and addressing missing, inconsistent, or duplicate values.

data provenance

The process of tracking the origin and history of data throughout its lifecycle, such as how the data was generated, transmitted, and stored.

data subject

An individual whose data is being collected and processed.

data warehouse

A data management system that supports business intelligence, such as analytics. Data warehouses commonly contain large amounts of historical data, and they are typically used for queries and analysis.

database definition language (DDL)

Statements or commands for creating or modifying the structure of tables and objects in a database.

database manipulation language (DML)

Statements or commands for modifying (inserting, updating, and deleting) information in a database.

DDL

See database definition language.

deep ensemble

To combine multiple deep learning models for prediction. You can use deep ensembles to obtain a more accurate prediction or for estimating uncertainty in predictions.

deep learning

An ML subfield that uses multiple layers of artificial neural networks to identify mapping between input data and target variables of interest.

defense-in-depth

An information security approach in which a series of security mechanisms and controls are thoughtfully layered throughout a computer network to protect the confidentiality, integrity, and availability of the network and the data within. When you adopt this strategy on AWS, you add multiple controls at different layers of the AWS Organizations structure to help secure resources. For example, a defense-in-depth approach might combine multi-factor authentication, network segmentation, and encryption.

delegated administrator

In AWS Organizations, a compatible service can register an AWS member account to administer the organization's accounts and manage permissions for that service. This account is called the *delegated administrator* for that service. For more information and a list of compatible services, see Services that work with AWS Organizations in the AWS Organizations documentation.

deployment

The process of making an application, new features, or code fixes available in the target environment. Deployment involves implementing changes in a code base and then building and running that code base in the application's environments.

development environment

See environment.

detective control

A security control that is designed to detect, log, and alert after an event has occurred. These controls are a second line of defense, alerting you to security events that bypassed the preventative controls in place. For more information, see Detective controls in *Implementing security controls on AWS*.

development value stream mapping (DVSM)

A process used to identify and prioritize constraints that adversely affect speed and quality in a software development lifecycle. DVSM extends the value stream mapping process originally designed for lean manufacturing practices. It focuses on the steps and teams required to create and move value through the software development process.

digital twin

A virtual representation of a real-world system, such as a building, factory, industrial equipment, or production line. Digital twins support predictive maintenance, remote monitoring, and production optimization.

dimension table

In a star schema, a smaller table that contains data attributes about quantitative data in a fact table. Dimension table attributes are typically text fields or discrete numbers that behave like text. These attributes are commonly used for query constraining, filtering, and result set labeling.

disaster

An event that prevents a workload or system from fulfilling its business objectives in its primary deployed location. These events can be natural disasters, technical failures, or the result of human actions, such as unintentional misconfiguration or a malware attack.

disaster recovery (DR)

The strategy and process you use to minimize downtime and data loss caused by a disaster. For more information, see Disaster Recovery of Workloads on AWS: Recovery in the Cloud in the AWS Well-Architected Framework.

DML

See database manipulation language.

domain-driven design

An approach to developing a complex software system by connecting its components to evolving domains, or core business goals, that each component serves. This concept was introduced by Eric Evans in his book, *Domain-Driven Design: Tackling Complexity in the Heart of Software* (Boston: Addison-Wesley Professional, 2003). For information about how you can use domain-driven design with the strangler fig pattern, see Modernizing legacy Microsoft ASP.NET (ASMX) web services incrementally by using containers and Amazon API Gateway.

DR

See disaster recovery.

drift detection

Tracking deviations from a baselined configuration. For example, you can use AWS CloudFormation to detect drift in system resources, or you can use AWS Control Tower to detect changes in your landing zone that might affect compliance with governance requirements.

DVSM

See development value stream mapping.

E

EDA

See exploratory data analysis.

edge computing

The technology that increases the computing power for smart devices at the edges of an IoT network. When compared with cloud computing, edge computing can reduce communication latency and improve response time.

encryption

A computing process that transforms plaintext data, which is human-readable, into ciphertext.

encryption key

A cryptographic string of randomized bits that is generated by an encryption algorithm. Keys can vary in length, and each key is designed to be unpredictable and unique.

endianness

The order in which bytes are stored in computer memory. Big-endian systems store the most significant byte first. Little-endian systems store the least significant byte first.

endpoint

See service endpoint.

endpoint service

A service that you can host in a virtual private cloud (VPC) to share with other users. You can create an endpoint service with AWS PrivateLink and grant permissions to other AWS accounts or to AWS Identity and Access Management (IAM) principals. These accounts or principals can connect to your endpoint service privately by creating interface VPC endpoints. For more information, see Create an endpoint service in the Amazon Virtual Private Cloud (Amazon VPC) documentation.

enterprise resource planning (ERP)

A system that automates and manages key business processes (such as accounting, MES, and project management) for an enterprise.

envelope encryption

The process of encrypting an encryption key with another encryption key. For more information, see Envelope encryption in the AWS Key Management Service (AWS KMS) documentation.

environment

An instance of a running application. The following are common types of environments in cloud computing:

- development environment – An instance of a running application that is available only to the core team responsible for maintaining the application. Development environments are used to test changes before promoting them to upper environments. This type of environment is sometimes referred to as a *test environment*.
- lower environments – All development environments for an application, such as those used for initial builds and tests.
- production environment – An instance of a running application that end users can access. In a CI/CD pipeline, the production environment is the last deployment environment.
- upper environments – All environments that can be accessed by users other than the core development team. This can include a production environment, preproduction environments, and environments for user acceptance testing.

epic

In agile methodologies, functional categories that help organize and prioritize your work. Epics provide a high-level description of requirements and implementation tasks. For example, AWS CAF security epics include identity and access management, detective controls, infrastructure security, data protection, and incident response. For more information about epics in the AWS migration strategy, see the program implementation guide.

ERP

See enterprise resource planning.

exploratory data analysis (EDA)

The process of analyzing a dataset to understand its main characteristics. You collect or aggregate data and then perform initial investigations to find patterns, detect anomalies, and check assumptions. EDA is performed by calculating summary statistics and creating data visualizations.

F

fact table

The central table in a star schema. It stores quantitative data about business operations. Typically, a fact table contains two types of columns: those that contain measures and those that contain a foreign key to a dimension table.

fail fast

A philosophy that uses frequent and incremental testing to reduce the development lifecycle. It is a critical part of an agile approach.

fault isolation boundary

In the AWS Cloud, a boundary such as an Availability Zone, AWS Region, control plane, or data plane that limits the effect of a failure and helps improve the resilience of workloads. For more information, see AWS Fault Isolation Boundaries.

feature branch

See branch.

features

The input data that you use to make a prediction. For example, in a manufacturing context, features could be images that are periodically captured from the manufacturing line.

feature importance

How significant a feature is for a model's predictions. This is usually expressed as a numerical score that can be calculated through various techniques, such as Shapley Additive Explanations (SHAP) and integrated gradients. For more information, see Machine learning model interpretability with :AWS.

feature transformation

To optimize data for the ML process, including enriching data with additional sources, scaling values, or extracting multiple sets of information from a single data field. This enables the ML model to benefit from the data. For example, if you break down the "2021-05-27 00:15:37" date into "2021", "May", "Thu", and "15", you can help the learning algorithm learn nuanced patterns associated with different data components.

FGAC

See fine-grained access control.

fine-grained access control (FGAC)

The use of multiple conditions to allow or deny an access request.

flash-cut migration

A database migration method that uses continuous data replication through change data capture to migrate data in the shortest time possible, instead of using a phased approach. The objective is to keep downtime to a minimum.

G

geo blocking

See geographic restrictions.

geographic restrictions (geo blocking)

In Amazon CloudFront, an option to prevent users in specific countries from accessing content distributions. You can use an allow list or block list to specify approved and banned countries. For more information, see Restricting the geographic distribution of your content in the CloudFront documentation.

Gitflow workflow

An approach in which lower and upper environments use different branches in a source code repository. The Gitflow workflow is considered legacy, and the trunk-based workflow is the modern, preferred approach.

greenfield strategy

The absence of existing infrastructure in a new environment. When adopting a greenfield strategy for a system architecture, you can select all new technologies without the restriction of compatibility with existing infrastructure, also known as brownfield. If you are expanding the existing infrastructure, you might blend brownfield and greenfield strategies.

guardrail

A high-level rule that helps govern resources, policies, and compliance across organizational units (OUs). *Preventive guardrails* enforce policies to ensure alignment to compliance standards. They are implemented by using service control policies and IAM permissions boundaries. *Detective guardrails* detect policy violations and compliance issues, and generate alerts

for remediation. They are implemented by using AWS Config, AWS Security Hub, Amazon GuardDuty, AWS Trusted Advisor, Amazon Inspector, and custom AWS Lambda checks.

H

HA

See high availability.

heterogeneous database migration

Migrating your source database to a target database that uses a different database engine (for example, Oracle to Amazon Aurora). Heterogeneous migration is typically part of a re-architecting effort, and converting the schema can be a complex task. AWS provides AWS SCT that helps with schema conversions.

high availability (HA)

The ability of a workload to operate continuously, without intervention, in the event of challenges or disasters. HA systems are designed to automatically fail over, consistently deliver high-quality performance, and handle different loads and failures with minimal performance impact.

historian modernization

An approach used to modernize and upgrade operational technology (OT) systems to better serve the needs of the manufacturing industry. A *historian* is a type of database that is used to collect and store data from various sources in a factory.

homogeneous database migration

Migrating your source database to a target database that shares the same database engine (for example, Microsoft SQL Server to Amazon RDS for SQL Server). Homogeneous migration is typically part of a rehosting or replatforming effort. You can use native database utilities to migrate the schema.

hot data

Data that is frequently accessed, such as real-time data or recent translational data. This data typically requires a high-performance storage tier or class to provide fast query responses.

hotfix

An urgent fix for a critical issue in a production environment. Due to its urgency, a hotfix is usually made outside of the typical DevOps release workflow.

hypercare period

Immediately following cutover, the period of time when a migration team manages and monitors the migrated applications in the cloud in order to address any issues. Typically, this period is 1–4 days in length. At the end of the hypercare period, the migration team typically transfers responsibility for the applications to the cloud operations team.

I

IaC

See infrastructure as code.

identity-based policy

A policy attached to one or more IAM principals that defines their permissions within the AWS Cloud environment.

idle application

An application that has an average CPU and memory usage between 5 and 20 percent over a period of 90 days. In a migration project, it is common to retire these applications or retain them on premises.

IIoT

See industrial Internet of Things.

immutable infrastructure

A model that deploys new infrastructure for production workloads instead of updating, patching, or modifying the existing infrastructure. Immutable infrastructures are inherently more consistent, reliable, and predictable than mutable infrastructure. For more information, see the Deploy using immutable infrastructure best practice in the AWS Well-Architected Framework.

inbound (ingress) VPC

In an AWS multi-account architecture, a VPC that accepts, inspects, and routes network connections from outside an application. The AWS Security Reference Architecture recommends

setting up your Network account with inbound, outbound, and inspection VPCs to protect the two-way interface between your application and the broader internet.

incremental migration

A cutover strategy in which you migrate your application in small parts instead of performing a single, full cutover. For example, you might move only a few microservices or users to the new system initially. After you verify that everything is working properly, you can incrementally move additional microservices or users until you can decommission your legacy system. This strategy reduces the risks associated with large migrations.

Industry 4.0

A term that was introduced by Klaus Schwab in 2016 to refer to the modernization of manufacturing processes through advances in connectivity, real-time data, automation, analytics, and AI/ML.

infrastructure

All of the resources and assets contained within an application's environment.

infrastructure as code (IaC)

The process of provisioning and managing an application's infrastructure through a set of configuration files. IaC is designed to help you centralize infrastructure management, standardize resources, and scale quickly so that new environments are repeatable, reliable, and consistent.

industrial Internet of Things (IIoT)

The use of internet-connected sensors and devices in the industrial sectors, such as manufacturing, energy, automotive, healthcare, life sciences, and agriculture. For more information, see Building an industrial Internet of Things (IIoT) digital transformation strategy.

inspection VPC

In an AWS multi-account architecture, a centralized VPC that manages inspections of network traffic between VPCs (in the same or different AWS Regions), the internet, and on-premises networks. The AWS Security Reference Architecture recommends setting up your Network account with inbound, outbound, and inspection VPCs to protect the two-way interface between your application and the broader internet.

Internet of Things (IoT)

The network of connected physical objects with embedded sensors or processors that communicate with other devices and systems through the internet or over a local communication network. For more information, see What is IoT?

interpretability

A characteristic of a machine learning model that describes the degree to which a human can understand how the model's predictions depend on its inputs. For more information, see Machine learning model interpretability with AWS.

IoT

See Internet of Things.

IT information library (ITIL)

A set of best practices for delivering IT services and aligning these services with business requirements. ITIL provides the foundation for ITSM.

IT service management (ITSM)

Activities associated with designing, implementing, managing, and supporting IT services for an organization. For information about integrating cloud operations with ITSM tools, see the operations integration guide.

ITIL

See IT information library.

ITSM

See IT service management.

L

label-based access control (LBAC)

An implementation of mandatory access control (MAC) where the users and the data itself are each explicitly assigned a security label value. The intersection between the user security label and data security label determines which rows and columns can be seen by the user.

landing zone

A landing zone is a well-architected, multi-account AWS environment that is scalable and secure. This is a starting point from which your organizations can quickly launch and deploy workloads and applications with confidence in their security and infrastructure environment. For more information about landing zones, see Setting up a secure and scalable multi-account AWS environment.

large migration

A migration of 300 or more servers.

LBAC

See label-based access control.

least privilege

The security best practice of granting the minimum permissions required to perform a task. For more information, see Apply least-privilege permissions in the IAM documentation.

lift and shift

See 7 Rs.

little-endian system

A system that stores the least significant byte first. See also endianness.

lower environments

See environment.

M

machine learning (ML)

A type of artificial intelligence that uses algorithms and techniques for pattern recognition and learning. ML analyzes and learns from recorded data, such as Internet of Things (IoT) data, to generate a statistical model based on patterns. For more information, see Machine Learning.

main branch

See branch.

malware

Software that is designed to compromise computer security or privacy. Malware might disrupt computer systems, leak sensitive information, or gain unauthorized access. Examples of malware include viruses, worms, ransomware, Trojan horses, spyware, and keyloggers.

managed services

AWS services for which AWS operates the infrastructure layer, the operating system, and platforms, and you access the endpoints to store and retrieve data. Amazon Simple Storage Service (Amazon S3) and Amazon DynamoDB are examples of managed services. These are also known as *abstracted services*.

manufacturing execution system (MES)

A software system for tracking, monitoring, documenting, and controlling production processes that convert raw materials to finished products on the shop floor.

MAP

See Migration Acceleration Program.

mechanism

A complete process in which you create a tool, drive adoption of the tool, and then inspect the results in order to make adjustments. A mechanism is a cycle that reinforces and improves itself as it operates. For more information, see Building mechanisms in the AWS Well-Architected Framework.

member account

All AWS accounts other than the management account that are part of an organization in AWS Organizations. An account can be a member of only one organization at a time.

MES

See manufacturing execution system.

Message Queuing Telemetry Transport (MQTT)

A lightweight, machine-to-machine (M2M) communication protocol, based on the publish/subscribe pattern, for resource-constrained IoT devices.

microservice

A small, independent service that communicates over well-defined APIs and is typically owned by small, self-contained teams. For example, an insurance system might include

microservices that map to business capabilities, such as sales or marketing, or subdomains, such as purchasing, claims, or analytics. The benefits of microservices include agility, flexible scaling, easy deployment, reusable code, and resilience. For more information, see Integrating microservices by using AWS serverless services.

microservices architecture

An approach to building an application with independent components that run each application process as a microservice. These microservices communicate through a well-defined interface by using lightweight APIs. Each microservice in this architecture can be updated, deployed, and scaled to meet demand for specific functions of an application. For more information, see Implementing microservices on AWS.

Migration Acceleration Program (MAP)

An AWS program that provides consulting support, training, and services to help organizations build a strong operational foundation for moving to the cloud, and to help offset the initial cost of migrations. MAP includes a migration methodology for executing legacy migrations in a methodical way and a set of tools to automate and accelerate common migration scenarios.

migration at scale

The process of moving the majority of the application portfolio to the cloud in waves, with more applications moved at a faster rate in each wave. This phase uses the best practices and lessons learned from the earlier phases to implement a *migration factory* of teams, tools, and processes to streamline the migration of workloads through automation and agile delivery. This is the third phase of the AWS migration strategy.

migration factory

Cross-functional teams that streamline the migration of workloads through automated, agile approaches. Migration factory teams typically include operations, business analysts and owners, migration engineers, developers, and DevOps professionals working in sprints. Between 20 and 50 percent of an enterprise application portfolio consists of repeated patterns that can be optimized by a factory approach. For more information, see the discussion of migration factories and the Cloud Migration Factory guide in this content set.

migration metadata

The information about the application and server that is needed to complete the migration. Each migration pattern requires a different set of migration metadata. Examples of migration metadata include the target subnet, security group, and AWS account.

migration pattern

A repeatable migration task that details the migration strategy, the migration destination, and the migration application or service used. Example: Rehost migration to Amazon EC2 with AWS Application Migration Service.

Migration Portfolio Assessment (MPA)

An online tool that provides information for validating the business case for migrating to the AWS Cloud. MPA provides detailed portfolio assessment (server right-sizing, pricing, TCO comparisons, migration cost analysis) as well as migration planning (application data analysis and data collection, application grouping, migration prioritization, and wave planning). The MPA tool (requires login) is available free of charge to all AWS consultants and APN Partner consultants.

Migration Readiness Assessment (MRA)

The process of gaining insights about an organization's cloud readiness status, identifying strengths and weaknesses, and building an action plan to close identified gaps, using the AWS CAF. For more information, see the migration readiness guide. MRA is the first phase of the AWS migration strategy.

migration strategy

The approach used to migrate a workload to the AWS Cloud. For more information, see the 7 Rs entry in this glossary and see Mobilize your organization to accelerate large-scale migrations.

ML

See machine learning.

modernization

Transforming an outdated (legacy or monolithic) application and its infrastructure into an agile, elastic, and highly available system in the cloud to reduce costs, gain efficiencies, and take advantage of innovations. For more information, see Strategy for modernizing applications in the AWS Cloud.

modernization readiness assessment

An evaluation that helps determine the modernization readiness of an organization's applications; identifies benefits, risks, and dependencies; and determines how well the organization can support the future state of those applications. The outcome of the assessment is a blueprint of the target architecture, a roadmap that details development phases and

milestones for the modernization process, and an action plan for addressing identified gaps. For more information, see Evaluating modernization readiness for applications in the AWS Cloud.

monolithic applications (monoliths)

Applications that run as a single service with tightly coupled processes. Monolithic applications have several drawbacks. If one application feature experiences a spike in demand, the entire architecture must be scaled. Adding or improving a monolithic application's features also becomes more complex when the code base grows. To address these issues, you can use a microservices architecture. For more information, see Decomposing monoliths into microservices.

MPA

See Migration Portfolio Assessment.

MQTT

See Message Queuing Telemetry Transport.

multiclass classification

A process that helps generate predictions for multiple classes (predicting one of more than two outcomes). For example, an ML model might ask "Is this product a book, car, or phone?" or "Which product category is most interesting to this customer?"

mutable infrastructure

A model that updates and modifies the existing infrastructure for production workloads. For improved consistency, reliability, and predictability, the AWS Well-Architected Framework recommends the use of immutable infrastructure as a best practice.

O

OAC

See origin access control.

OAI

See origin access identity.

OCM

See organizational change management.

offline migration

A migration method in which the source workload is taken down during the migration process. This method involves extended downtime and is typically used for small, non-critical workloads.

OI

See operations integration.

OLA

See operational-level agreement.

online migration

A migration method in which the source workload is copied to the target system without being taken offline. Applications that are connected to the workload can continue to function during the migration. This method involves zero to minimal downtime and is typically used for critical production workloads.

OPC-UA

See Open Process Communications - Unified Architecture.

Open Process Communications - Unified Architecture (OPC-UA)

A machine-to-machine (M2M) communication protocol for industrial automation. OPC-UA provides an interoperability standard with data encryption, authentication, and authorization schemes.

operational-level agreement (OLA)

An agreement that clarifies what functional IT groups promise to deliver to each other, to support a service-level agreement (SLA).

operational readiness review (ORR)

A checklist of questions and associated best practices that help you understand, evaluate, prevent, or reduce the scope of incidents and possible failures. For more information, see Operational Readiness Reviews (ORR) in the AWS Well-Architected Framework.

operational technology (OT)

Hardware and software systems that work with the physical environment to control industrial operations, equipment, and infrastructure. In manufacturing, the integration of OT and information technology (IT) systems is a key focus for Industry 4.0 transformations.

operations integration (OI)

The process of modernizing operations in the cloud, which involves readiness planning, automation, and integration. For more information, see the operations integration guide.

organization trail

A trail that's created by AWS CloudTrail that logs all events for all AWS accounts in an organization in AWS Organizations. This trail is created in each AWS account that's part of the organization and tracks the activity in each account. For more information, see Creating a trail for an organization in the CloudTrail documentation.

organizational change management (OCM)

A framework for managing major, disruptive business transformations from a people, culture, and leadership perspective. OCM helps organizations prepare for, and transition to, new systems and strategies by accelerating change adoption, addressing transitional issues, and driving cultural and organizational changes. In the AWS migration strategy, this framework is called *people acceleration*, because of the speed of change required in cloud adoption projects. For more information, see the OCM guide.

origin access control (OAC)

In CloudFront, an enhanced option for restricting access to secure your Amazon Simple Storage Service (Amazon S3) content. OAC supports all S3 buckets in all AWS Regions, server-side encryption with AWS KMS (SSE-KMS), and dynamic `PUT` and `DELETE` requests to the S3 bucket.

origin access identity (OAI)

In CloudFront, an option for restricting access to secure your Amazon S3 content. When you use OAI, CloudFront creates a principal that Amazon S3 can authenticate with. Authenticated principals can access content in an S3 bucket only through a specific CloudFront distribution. See also OAC, which provides more granular and enhanced access control.

ORR

See operational readiness review.

OT

See operational technology.

outbound (egress) VPC

In an AWS multi-account architecture, a VPC that handles network connections that are initiated from within an application. The AWS Security Reference Architecture recommends

setting up your Network account with inbound, outbound, and inspection VPCs to protect the two-way interface between your application and the broader internet.

P

permissions boundary

An IAM management policy that is attached to IAM principals to set the maximum permissions that the user or role can have. For more information, see Permissions boundaries in the IAM documentation.

personally identifiable information (PII)

Information that, when viewed directly or paired with other related data, can be used to reasonably infer the identity of an individual. Examples of PII include names, addresses, and contact information.

PII

See personally identifiable information.

playbook

A set of predefined steps that capture the work associated with migrations, such as delivering core operations functions in the cloud. A playbook can take the form of scripts, automated runbooks, or a summary of processes or steps required to operate your modernized environment.

PLC

See programmable logic controller.

PLM

See product lifecycle management.

policy

An object that can define permissions (see identity-based policy), specify access conditions (see resource-based policy), or define the maximum permissions for all accounts in an organization in AWS Organizations (see service control policy).

polyglot persistence

Independently choosing a microservice's data storage technology based on data access patterns and other requirements. If your microservices have the same data storage technology, they can encounter implementation challenges or experience poor performance. Microservices are more easily implemented and achieve better performance and scalability if they use the data store best adapted to their requirements. For more information, see Enabling data persistence in microservices.

portfolio assessment

A process of discovering, analyzing, and prioritizing the application portfolio in order to plan the migration. For more information, see Evaluating migration readiness.

predicate

A query condition that returns `true` or `false`, commonly located in a `WHERE` clause.

predicate pushdown

A database query optimization technique that filters the data in the query before transfer. This reduces the amount of data that must be retrieved and processed from the relational database, and it improves query performance.

preventative control

A security control that is designed to prevent an event from occurring. These controls are a first line of defense to help prevent unauthorized access or unwanted changes to your network. For more information, see Preventative controls in *Implementing security controls on AWS*.

principal

An entity in AWS that can perform actions and access resources. This entity is typically a root user for an AWS account, an IAM role, or a user. For more information, see *Principal* in Roles terms and concepts in the IAM documentation.

Privacy by Design

An approach in system engineering that takes privacy into account throughout the whole engineering process.

private hosted zones

A container that holds information about how you want Amazon Route 53 to respond to DNS queries for a domain and its subdomains within one or more VPCs. For more information, see Working with private hosted zones in the Route 53 documentation.

proactive control

A security control designed to prevent the deployment of noncompliant resources. These controls scan resources before they are provisioned. If the resource is not compliant with the control, then it isn't provisioned. For more information, see the Controls reference guide in the AWS Control Tower documentation and see Proactive controls in *Implementing security controls on AWS*.

product lifecycle management (PLM)

The management of data and processes for a product throughout its entire lifecycle, from design, development, and launch, through growth and maturity, to decline and removal.

production environment

See environment.

programmable logic controller (PLC)

In manufacturing, a highly reliable, adaptable computer that monitors machines and automates manufacturing processes.

pseudonymization

The process of replacing personal identifiers in a dataset with placeholder values. Pseudonymization can help protect personal privacy. Pseudonymized data is still considered to be personal data.

publish/subscribe (pub/sub)

A pattern that enables asynchronous communications among microservices to improve scalability and responsiveness. For example, in a microservices-based MES, a microservice can publish event messages to a channel that other microservices can subscribe to. The system can add new microservices without changing the publishing service.

Q

query plan

A series of steps, like instructions, that are used to access the data in a SQL relational database system.

query plan regression

When a database service optimizer chooses a less optimal plan than it did before a given change to the database environment. This can be caused by changes to statistics, constraints, environment settings, query parameter bindings, and updates to the database engine.

R

RACI matrix

See responsible, accountable, consulted, informed (RACI).

ransomware

A malicious software that is designed to block access to a computer system or data until a payment is made.

RASCI matrix

See responsible, accountable, consulted, informed (RACI).

RCAC

See row and column access control.

read replica

A copy of a database that's used for read-only purposes. You can route queries to the read replica to reduce the load on your primary database.

re-architect

See 7 Rs.

recovery point objective (RPO)

The maximum acceptable amount of time since the last data recovery point. This determines what is considered an acceptable loss of data between the last recovery point and the interruption of service.

recovery time objective (RTO)

The maximum acceptable delay between the interruption of service and restoration of service.

refactor

See 7 Rs.

Region

A collection of AWS resources in a geographic area. Each AWS Region is isolated and independent of the others to provide fault tolerance, stability, and resilience. For more information, see Specify which AWS Regions your account can use.

regression

An ML technique that predicts a numeric value. For example, to solve the problem of "What price will this house sell for?" an ML model could use a linear regression model to predict a house's sale price based on known facts about the house (for example, the square footage).

rehost

See 7 Rs.

release

In a deployment process, the act of promoting changes to a production environment.

relocate

See 7 Rs.

replatform

See 7 Rs.

repurchase

See 7 Rs.

resiliency

An application's ability to resist or recover from disruptions. High availability and disaster recovery are common considerations when planning for resiliency in the AWS Cloud. For more information, see AWS Cloud Resilience.

resource-based policy

A policy attached to a resource, such as an Amazon S3 bucket, an endpoint, or an encryption key. This type of policy specifies which principals are allowed access, supported actions, and any other conditions that must be met.

responsible, accountable, consulted, informed (RACI) matrix

A matrix that defines the roles and responsibilities for all parties involved in migration activities and cloud operations. The matrix name is derived from the responsibility types defined in the

matrix: responsible (R), accountable (A), consulted (C), and informed (I). The support (S) type is optional. If you include support, the matrix is called a *RASCI matrix*, and if you exclude it, it's called a *RACI matrix*.

responsive control

A security control that is designed to drive remediation of adverse events or deviations from your security baseline. For more information, see Responsive controls in *Implementing security controls on AWS*.

retain

See 7 Rs.

retire

See 7 Rs.

rotation

The process of periodically updating a secret to make it more difficult for an attacker to access the credentials.

row and column access control (RCAC)

The use of basic, flexible SQL expressions that have defined access rules. RCAC consists of row permissions and column masks.

RPO

See recovery point objective.

RTO

See recovery time objective.

runbook

A set of manual or automated procedures required to perform a specific task. These are typically built to streamline repetitive operations or procedures with high error rates.

S

SAML 2.0

An open standard that many identity providers (IdPs) use. This feature enables federated single sign-on (SSO), so users can log into the AWS Management Console or call the AWS API

operations without you having to create user in IAM for everyone in your organization. For more information about SAML 2.0-based federation, see About SAML 2.0-based federation in the IAM documentation.

SCADA

See supervisory control and data acquisition.

SCP

See service control policy.

secret

In AWS Secrets Manager, confidential or restricted information, such as a password or user credentials, that you store in encrypted form. It consists of the secret value and its metadata. The secret value can be binary, a single string, or multiple strings. For more information, see What's in a Secrets Manager secret? in the Secrets Manager documentation.

security control

A technical or administrative guardrail that prevents, detects, or reduces the ability of a threat actor to exploit a security vulnerability. There are four primary types of security controls: preventative, detective, responsive, and proactive.

security hardening

The process of reducing the attack surface to make it more resistant to attacks. This can include actions such as removing resources that are no longer needed, implementing the security best practice of granting least privilege, or deactivating unnecessary features in configuration files.

security information and event management (SIEM) system

Tools and services that combine security information management (SIM) and security event management (SEM) systems. A SIEM system collects, monitors, and analyzes data from servers, networks, devices, and other sources to detect threats and security breaches, and to generate alerts.

security response automation

A predefined and programmed action that is designed to automatically respond to or remediate a security event. These automations serve as detective or responsive security controls that help you implement AWS security best practices. Examples of automated response actions include modifying a VPC security group, patching an Amazon EC2 instance, or rotating credentials.

server-side encryption

Encryption of data at its destination, by the AWS service that receives it.

service control policy (SCP)

A policy that provides centralized control over permissions for all accounts in an organization in AWS Organizations. SCPs define guardrails or set limits on actions that an administrator can delegate to users or roles. You can use SCPs as allow lists or deny lists, to specify which services or actions are permitted or prohibited. For more information, see Service control policies in the AWS Organizations documentation.

service endpoint

The URL of the entry point for an AWS service. You can use the endpoint to connect programmatically to the target service. For more information, see AWS service endpoints in *AWS General Reference*.

service-level agreement (SLA)

An agreement that clarifies what an IT team promises to deliver to their customers, such as service uptime and performance.

service-level indicator (SLI)

A measurement of a performance aspect of a service, such as its error rate, availability, or throughput.

service-level objective (SLO)

A target metric that represents the health of a service, as measured by a service-level indicator.

shared responsibility model

A model describing the responsibility you share with AWS for cloud security and compliance. AWS is responsible for security *of* the cloud, whereas you are responsible for security *in* the cloud. For more information, see Shared responsibility model.

SIEM

See security information and event management system.

single point of failure (SPOF)

A failure in a single, critical component of an application that can disrupt the system.

SLA

See service-level agreement.

SLI

See service-level indicator.

SLO

See service-level objective.

split-and-seed model

A pattern for scaling and accelerating modernization projects. As new features and product releases are defined, the core team splits up to create new product teams. This helps scale your organization's capabilities and services, improves developer productivity, and supports rapid innovation. For more information, see Phased approach to modernizing applications in the AWS Cloud.

SPOF

See single point of failure.

star schema

A database organizational structure that uses one large fact table to store transactional or measured data and uses one or more smaller dimensional tables to store data attributes. This structure is designed for use in a data warehouse or for business intelligence purposes.

strangler fig pattern

An approach to modernizing monolithic systems by incrementally rewriting and replacing system functionality until the legacy system can be decommissioned. This pattern uses the analogy of a fig vine that grows into an established tree and eventually overcomes and replaces its host. The pattern was introduced by Martin Fowler as a way to manage risk when rewriting monolithic systems. For an example of how to apply this pattern, see Modernizing legacy Microsoft ASP.NET (ASMX) web services incrementally by using containers and Amazon API Gateway.

subnet

A range of IP addresses in your VPC. A subnet must reside in a single Availability Zone.

supervisory control and data acquisition (SCADA)

In manufacturing, a system that uses hardware and software to monitor physical assets and production operations.

symmetric encryption

An encryption algorithm that uses the same key to encrypt and decrypt the data.

synthetic testing

Testing a system in a way that simulates user interactions to detect potential issues or to monitor performance. You can use Amazon CloudWatch Synthetics to create these tests.

T

tags

Key-value pairs that act as metadata for organizing your AWS resources. Tags can help you manage, identify, organize, search for, and filter resources. For more information, see Tagging your AWS resources.

target variable

The value that you are trying to predict in supervised ML. This is also referred to as an *outcome variable*. For example, in a manufacturing setting the target variable could be a product defect.

task list

A tool that is used to track progress through a runbook. A task list contains an overview of the runbook and a list of general tasks to be completed. For each general task, it includes the estimated amount of time required, the owner, and the progress.

test environment

See environment.

training

To provide data for your ML model to learn from. The training data must contain the correct answer. The learning algorithm finds patterns in the training data that map the input data attributes to the target (the answer that you want to predict). It outputs an ML model that captures these patterns. You can then use the ML model to make predictions on new data for which you don't know the target.

transit gateway

A network transit hub that you can use to interconnect your VPCs and on-premises networks. For more information, see What is a transit gateway in the AWS Transit Gateway documentation.

trunk-based workflow

An approach in which developers build and test features locally in a feature branch and then merge those changes into the main branch. The main branch is then built to the development, preproduction, and production environments, sequentially.

trusted access

Granting permissions to a service that you specify to perform tasks in your organization in AWS Organizations and in its accounts on your behalf. The trusted service creates a service-linked role in each account, when that role is needed, to perform management tasks for you. For more information, see Using AWS Organizations with other AWS services in the AWS Organizations documentation.

tuning

To change aspects of your training process to improve the ML model's accuracy. For example, you can train the ML model by generating a labeling set, adding labels, and then repeating these steps several times under different settings to optimize the model.

two-pizza team

A small DevOps team that you can feed with two pizzas. A two-pizza team size ensures the best possible opportunity for collaboration in software development.

U

uncertainty

A concept that refers to imprecise, incomplete, or unknown information that can undermine the reliability of predictive ML models. There are two types of uncertainty: *Epistemic uncertainty* is caused by limited, incomplete data, whereas *aleatoric uncertainty* is caused by the noise and randomness inherent in the data. For more information, see the Quantifying uncertainty in deep learning systems guide.

undifferentiated tasks

Also known as *heavy lifting*, work that is necessary to create and operate an application but that doesn't provide direct value to the end user or provide competitive advantage. Examples of undifferentiated tasks include procurement, maintenance, and capacity planning.

upper environments

See environment.

V

vacuuming

A database maintenance operation that involves cleaning up after incremental updates to reclaim storage and improve performance.

version control

Processes and tools that track changes, such as changes to source code in a repository.

VPC peering

A connection between two VPCs that allows you to route traffic by using private IP addresses. For more information, see What is VPC peering in the Amazon VPC documentation.

vulnerability

A software or hardware flaw that compromises the security of the system.

W

warm cache

A buffer cache that contains current, relevant data that is frequently accessed. The database instance can read from the buffer cache, which is faster than reading from the main memory or disk.

warm data

Data that is infrequently accessed. When querying this kind of data, moderately slow queries are typically acceptable.

window function

A SQL function that performs a calculation on a group of rows that relate in some way to the current record. Window functions are useful for processing tasks, such as calculating a moving average or accessing the value of rows based on the relative position of the current row.

workload

A collection of resources and code that delivers business value, such as a customer-facing application or backend process.

workstream

Functional groups in a migration project that are responsible for a specific set of tasks. Each workstream is independent but supports the other workstreams in the project. For example, the portfolio workstream is responsible for prioritizing applications, wave planning, and collecting migration metadata. The portfolio workstream delivers these assets to the migration workstream, which then migrates the servers and applications.

WORM

See write once, read many.

WQF

See AWS Workload Qualification Framework.

write once, read many (WORM)

A storage model that writes data a single time and prevents the data from being deleted or modified. Authorized users can read the data as many times as needed, but they cannot change it. This data storage infrastructure is considered immutable.

Z

zero-day exploit

An attack, typically malware, that takes advantage of a zero-day vulnerability.

zero-day vulnerability

An unmitigated flaw or vulnerability in a production system. Threat actors can use this type of vulnerability to attack the system. Developers frequently become aware of the vulnerability as a result of the attack.

zombie application

An application that has an average CPU and memory usage below 5 percent. In a migration project, it is common to retire these applications.