CLOUDFLARE

2024 API Security & Management Report

Table of Contents

Click section to skip to its page

Executive Summary

The Internet is an endless flow of conversations between computers. These conversations often take place using application programming interfaces (APIs), which allow us to interact with software and apps in new ways. For instance, OpenAI's ChatGPT API enables Slack to streamline chat-based workflows, and Booking.com to deliver more personalized trip planning experiences.

Today, APIs outpace other Internet traffic, **comprising more than half (57%) of the dynamic Internet traffic** processed by Cloudflare[1] last year.

However, as explored in this **2023 API Security and Management Report**, APIs are increasingly complex to manage and protect against abuse.

For instance, many organizations lack accurate information on their APIs. Cloudflare found 30.7% more API endpoints through machine learning-based discovery, compared to what organizations self-reported.[2]

30.7%

more API endpoints

Unfortunately, organizations cannot properly defend what they cannot see.

Those that implement API security without an accurate, real-time picture of their API landscape **can unintentionally block legitimate traffic.**

Take the **"too many requests" (429) error code — the #1 API client error category** Cloudflare mitigated in 2023. A 429 code does not automatically mean *too many requests from an attacker*. If the rate limits responsible for the errors were originally put in place due to a Distributed Denial of Service (DDoS) attack, for example, imposing overly broad, incorrect rate limits can still block legitimate users. (Of note, **DDoS protection was the #1 API mitigation method** for Cloudflare customers).

The goal of this report is to provide a valuable benchmark for organizations to **holistically assess the health of their API endpoint management**. After all, API security must also incorporate data to manage visibility, performance, and risks.

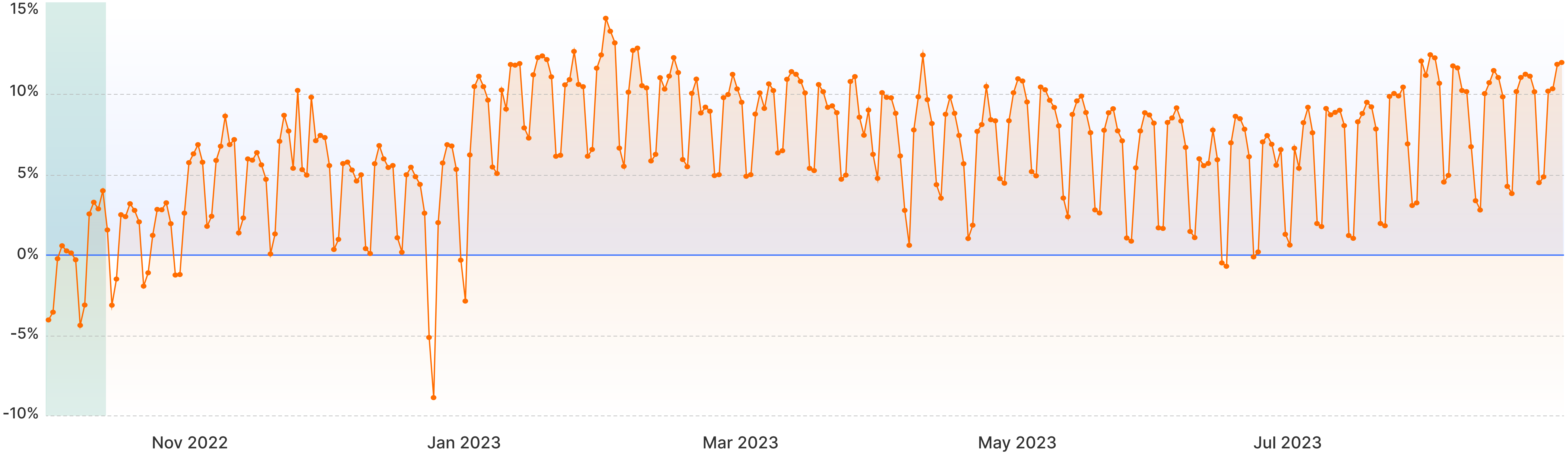
Methodology

The findings in this report are based on aggregated traffic patterns observed by Cloudflare's global network (including Cloudflare's web application firewall, DDoS protection, bot management, and API gateway services) between Oct. 1, 2022 and Aug. 31, 2023. Cloudflare serves over 50 million HTTP requests per second on average, and blocks an average of 170 billion cyber threats each day.

Snapshot: API-related traffic worldwide

Worldwide growth of API traffic over time

Baseline is highlighted, response codes of 200 and dynamic cache only



Between Oct 1, 2022 and August 31, 2023, API traffic with successful responses (200 status code) represented between 53.1% to 60.1% of Cloudflare's dynamic HTTP traffic. Dynamic content is content that changes based on factors specific to the user, such as time of visit, location, and device.

Key findings

APIs outpace other Internet traffic

Successful API requests accounted for **57% of Internet traffic** (dynamic HTTP traffic) processed by Cloudflare.[1]

#1 mitigation method: DDoS protection

One-third **(33%) of API mitigations comprised blocking Distributed Denial of Service (DDoS)** attacks.[4]

Unknown attack surfaces

Machine learning models discovered nearly **one-third (30.7%) more API endpoints** than what organizations self-reported.[2]

Industry variations

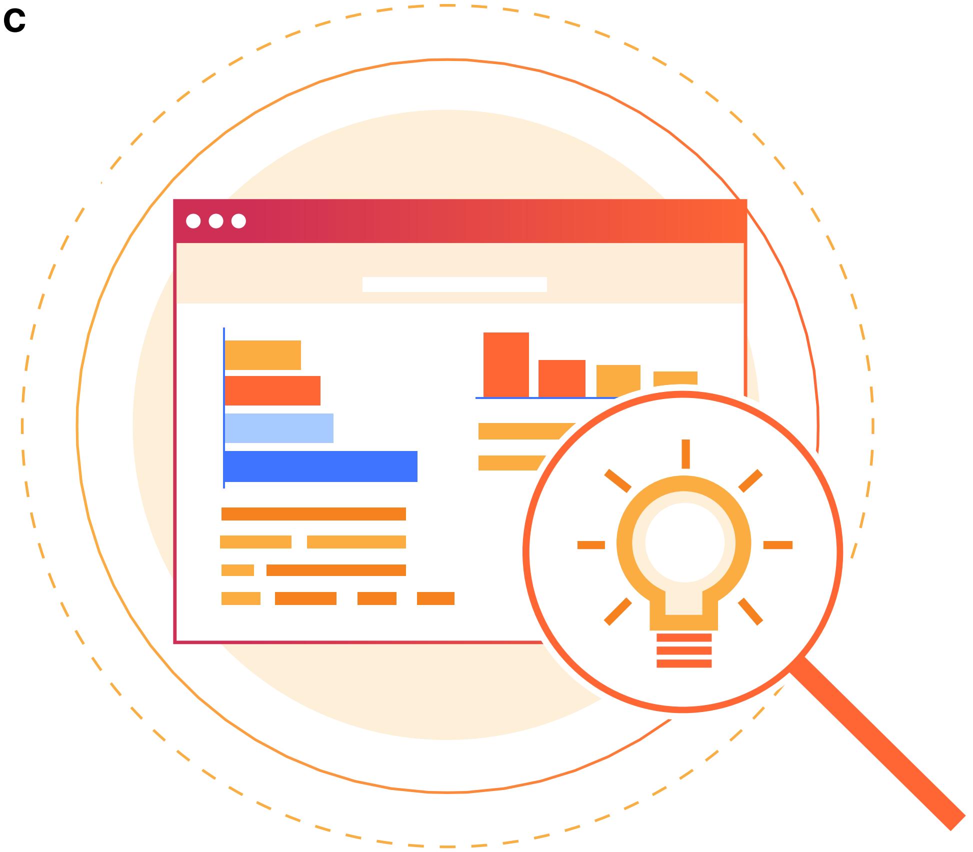
Industries with the **highest share of API traffic included the IoT platform, rail/bus/taxi, legal services, multimedia/games, and logistics/ supply chain** industries.[5]

#1 error: Too many requests

Over half **(51.6%) of API error rates comprised "Too Many Requests"** (429 errors).[3]

Regional variations

API traffic share was highest in **Africa and Asia**. API traffic **varied the most in the Middle East**.[6]



Hidden attack surfaces

Table of contents

For businesses, APIs fuel competitive advantages — greater business intelligence, swifter cloud deployments, integration of new AI capabilities, and more. But, the first step to optimizing APIs is to have a complete inventory of the hostnames and all API endpoints exposed to the Internet.

Organizations cannot manage or protect an API if they do not know it exists. And, **as it turns out, many organizations lack a full inventory of their APIs**.

- **Cloudflare discovered nearly 31% *more* API REST endpoints through machine learning** than were discovered via customer-provided session identifiers.[2]
- ***More than 15,000 accounts*** using Cloudflare had API endpoints discovered through machine learning methods only.[7]

APIs that have not been managed or secured by the organization using it — also known as 'Shadow' APIs, are often introduced by developers or individual users to run specific business functions.

While they are not inherently malicious, **shadow APIs are essentially unprotected attack surfaces that introduce new risks**.

If exploited, shadow APIs can lead to data exposure, unpatched vulnerabilities, data compliance violations, lateral movement, and other threats.

VISIBILITY CHECK

How do you discover and catalog APIs today?

An organization's or developer's API inventory is captured through API schemas — the metadata that defines the specifications of a valid API request and response. These API schemas (often documented with OpenAPI specifications) include the API host, HTTP method, path, and other requirements established by developers, such as path and query variables.

< Table of contents >

The risk of shadow APIs

Cloudflare often sees organizations that are early in their API management journey use an "email and ask" approach, which creates a point-in-time inventory that is likely to change with the next code release. But this manual approach typically relies on tribal knowledge, and is prone to manual error.

Say a healthcare organization's IT team is not aware that an API gives vendors access to certain systems. If the vendor is compromised, an attacker could abuse the API to exfiltrate patient data.

The 2019 Quest Diagnostics data breach, for example, exposed the data of nearly 12 million patients, when an unauthorized user gained access to an API that was sending information to billing vendors.

In 2022, Australian telecom provider Optus was breached, reportedly due to an attacker accessing its customer database via an unauthenticated API.

As the API economy grows, so do the problems of loss and control and complexity with API development, management, and security.

SECURITY CHECK

How are you monitoring which of your APIs allow 'write' access?

When aggregated across all account APIs, Cloudflare found that **59.2% of organizations permitted 'write' access to at least half of their APIs.**[8]

'Read-only' (GET) access APIs pull and ingest information from a system. However, 'Write' (POST, PUT, DELETE) APIs also permit users and other apps to push updates (change) a system.

Many API breaches happen due to permissive authorization: users being granted too many privileges, or allowed access to other users' data. When an API provides 'write' access to the wrong person, it can lead to attacks such as those described in this report.

Common API Errors

Once organizations have accurately discovered (then saved or removed) API endpoints, they need to know what's going well — and what's not. API errors can signal cyber attacks or app performance issues, which - in turn - prevent legitimate business.

HTTP status codes are the 3-digit codes most often used to indicate whether an app is performing successfully or if there's an error.

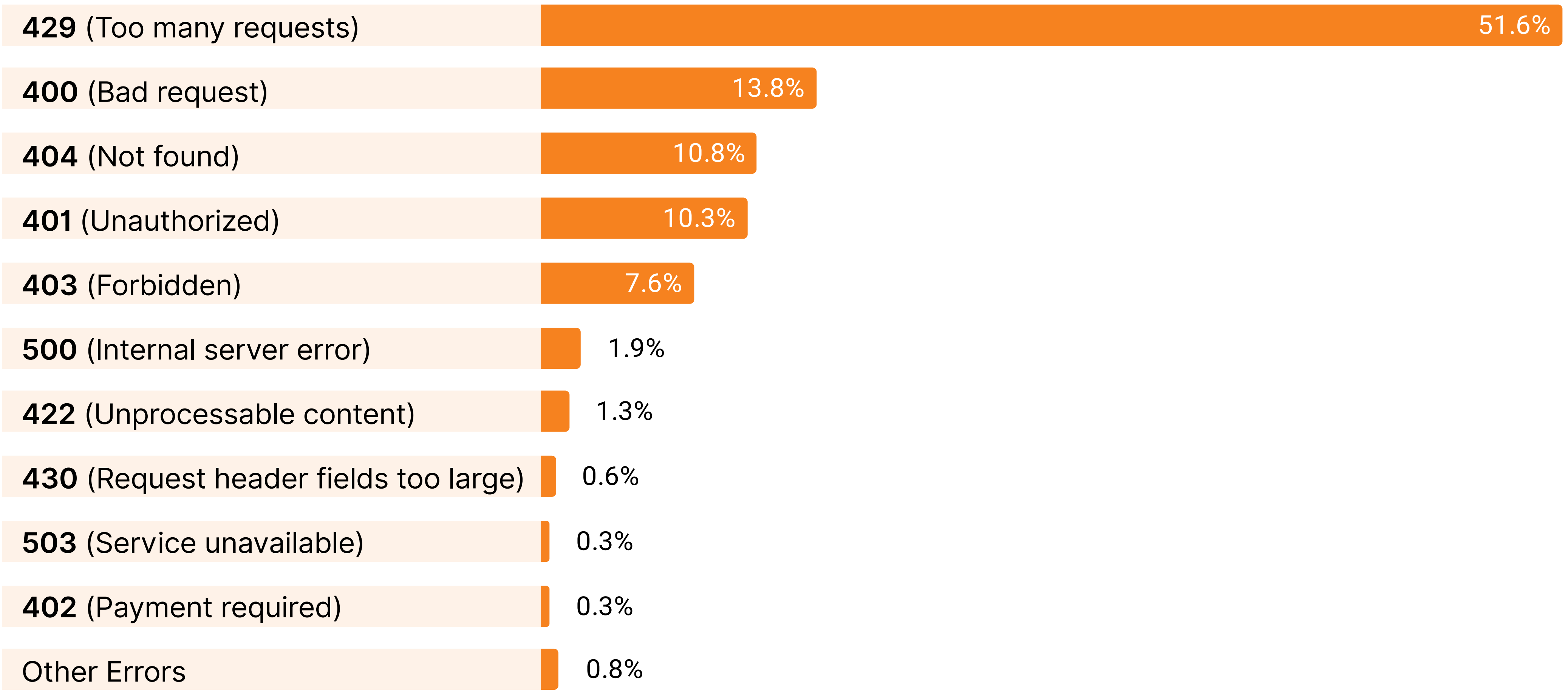
For APIs and other HTTP requests, a status code starting with '2' (2xx success codes) indicates that a client's action was received, understood, and accepted (i.e., successful).

However, when app visitors fail to reach their intended destination, they may, instead, get redirected (3xx redirection), or experience 4xx client errors or 5xx server-side errors.

Cloudflare observed trillions of traffic errors from API origins, of which **over half (51.6%) comprised '429' codes: "Too Many Requests."**[3]

Also known as "rate limiting," the 429 error occurs when the client has sent too many requests in a specific amount of time, according to the server.

Common API Errors



Please refer to the Appendix for error descriptions

The risk of misdiagnosing API errors

A **429 error (the most frequent API error**, as noted above) means the server has automatically throttled API traffic when a certain action takes place (such as a certain IP address exceeding a certain number of requests per minute to a `/login` endpoint).

However, if an organization uses manually-set rate limiting (instead of adaptive rate limiting), those can quickly go out-of-date. What if the `/login` endpoint is experiencing higher-than-average traffic because of a successful marketing campaign — and not an attack? In that scenario, manual rate limiting could prevent legitimate transactions.

Another example error whose cause is often 'misdiagnosed' is the **401 "Unauthorized" error (the fourth-most common error Cloudflare observed in API traffic)**.

A 401 means that the user's credentials did not exist or did not contain appropriate levels of access for the requested resource. But – like other HTTP error codes, the code may be due to a threat (such as an attempted Broken Object Level Authorization attack that can result in full account takeover) – or it could simply be due to a legitimate user accidentally entering the wrong credentials.

In one example of 'misdiagnosing' API traffic, in early 2023, Google warned website owners and some content delivery networks against using the wrong status errors for limiting their (legitimate) Googlebot's crawl rates.

As Google reminded users, *"Client errors are just that: client errors ... They simply mean that the client's request was bad in some way."*

PERFORMANCE CHECK

How are you monitoring and assessing API errors?

Not all API errors are caused by attacks. Understanding the root cause of API errors (and the trends behind those issues) requires consistent logging of API traffic, and analyzing trends over time.

Do you know how much of your API traffic is rate limited? How much of it is forbidden (due to bad authorization)? Are you confirming that errors are due to attacks, versus users with expired (or inaccurately-entered) credentials?

< Table of contents >

Top API security vulnerabilities

APIs are challenging to protect from abuse. They require deeper business context, discovery methods, and access verification controls compared to other web application security services.

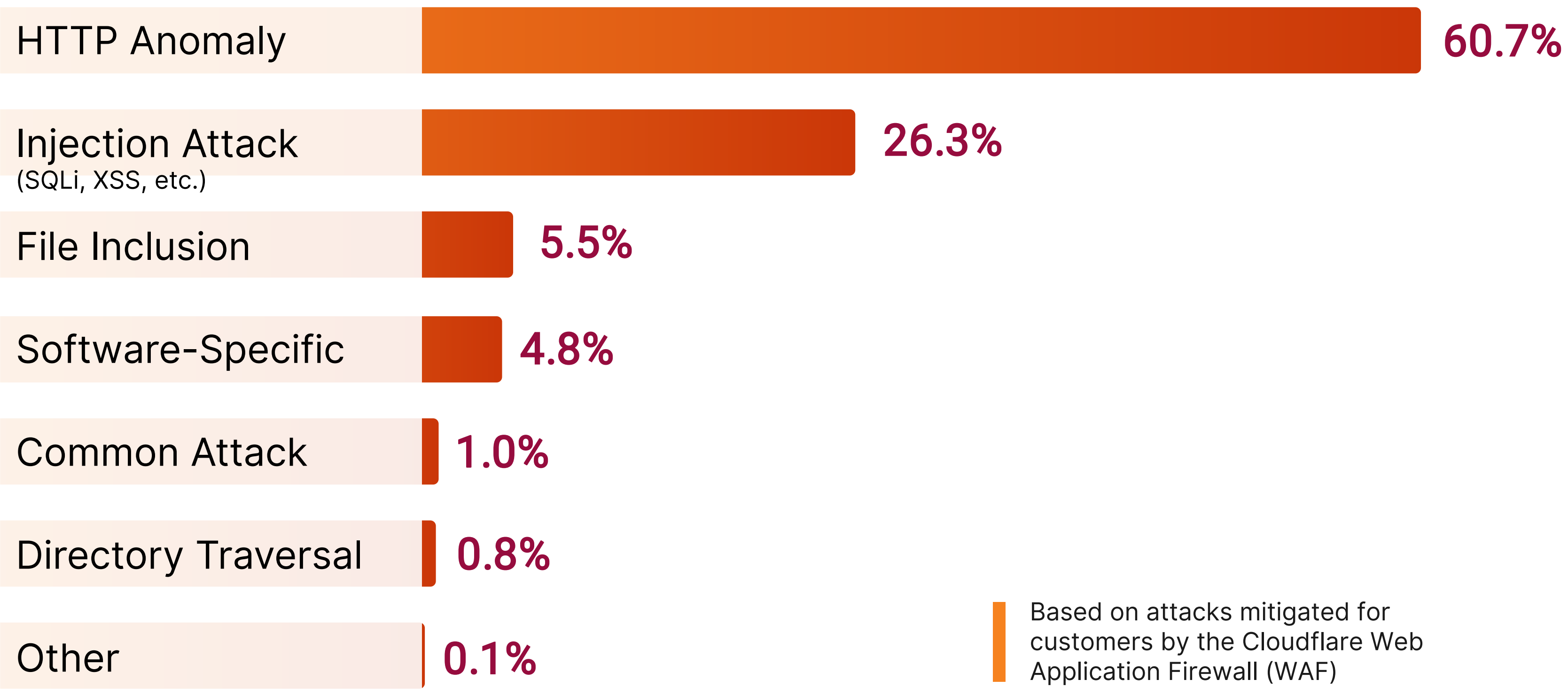
Consider, for example, that:

A web application	An API
Is **visible** to the end user	Is **invisible** to the app user
Is **accessed by end users** through a web browser	Lets **systems and apps** exchange data
Visualizes data from the backend (using HTML, CSS, and JavaScript) with varied user interactions.	**Transports data** by accessing servers and applications using a defined format (most commonly RESTful JSON, gRPC, XML, GraphQL).
Is typically protected through a “**negative security**” model that blocks known malicious traffic.	Is more effectively protected through a “**positive security**” model that only allows validated, authenticated traffic.

For those (and other) reasons, regular and automated monitoring of APIs is crucial for promptly identifying and addressing security threats.

Below is a snapshot of the most frequent threats toward APIs that Cloudflare mitigated for customers in 2023[9]:

Top API Threats



Based on attacks mitigated for customers by the Cloudflare Web Application Firewall (WAF)

A more detailed description of these attack types can be found in the Appendix.

The role of API vulnerabilities in one MDM attack

Mobile device management (MDM) helps organizations manage all of their geographically-dispersed devices from a single platform. With MDM, IT teams can deploy and control apps on the managed devices with the device's built-in APIs.

However, the ease and convenience of MDM systems must be weighed against the risks: MDM systems are attractive targets because they can provide attackers elevated access to thousands of mobile devices.

In August 2023, the Cybersecurity and Infrastructure Security Agency (CISA) and Norwegian National Cyber Security Centre (NCSC-NO) issued a joint Cybersecurity Advisory warning of attackers chaining two vulnerabilities to **exploit the Ivanti Endpoint Manager Mobile (EPMM), formerly known as MobileIron Core**.

The attackers used multiple methods such as the MITRE ATT&CK® techniques outlined in the following chart. Adherence to frameworks like MITRE ATT&CK and the OWASP API Security Top 10 help provide a strong foundation for more resilient cybersecurity — including stronger API defense.

Example Technique *(full list here)*	**Use**
Exploit Public-Facing Application	The attackers exploited CVE-2023-35078 in public-facing Ivanti EPMM appliances since at least April 2023.
Command and Scripting Interpreter	The attackers may have exploited CVE-2023-35081 to upload webshells on the EPMM device and run commands.
Account Discovery: Domain Account	The attackers exploited CVE-2023-35078 to gather EPMM device users and administrators. **In this scenario, they used API path** `/mifs/aad/api/v2/authorized/users` **to list users and administrators on the EPMM device**.
Remote System Discovery	The attackers retrieved LDAP endpoints.
Server Software Component: Web Shell	The attackers implanted webshells on the compromised infrastructure.
Proxy	The attackers leveraged compromised SOHO routers to proxy to and compromise infrastructure.

Two ways to mitigate common API vulnerabilities

1. Schema validation

HTTP anomalies such as missing user agents (the software that retrieves Internet content for end users), malformed method names, non-standard ports, and more are common signals of malicious requests. And, as noted above, these types of HTTP anomalies comprised the majority of Cloudflare-mitigated API threats.

Schema validation is a valuable way to identify HTTP anomalies in order to only allow "clean" traffic for each API to your API servers. The API schema defines which API requests are valid based on several request properties like target endpoint, path or query variable format, and HTTP method.

2. Tackle authentication loopholes

Lack of (or broken) authentication in public APIs is another serious problem, as seen in past news headlines regarding API-related data breaches.

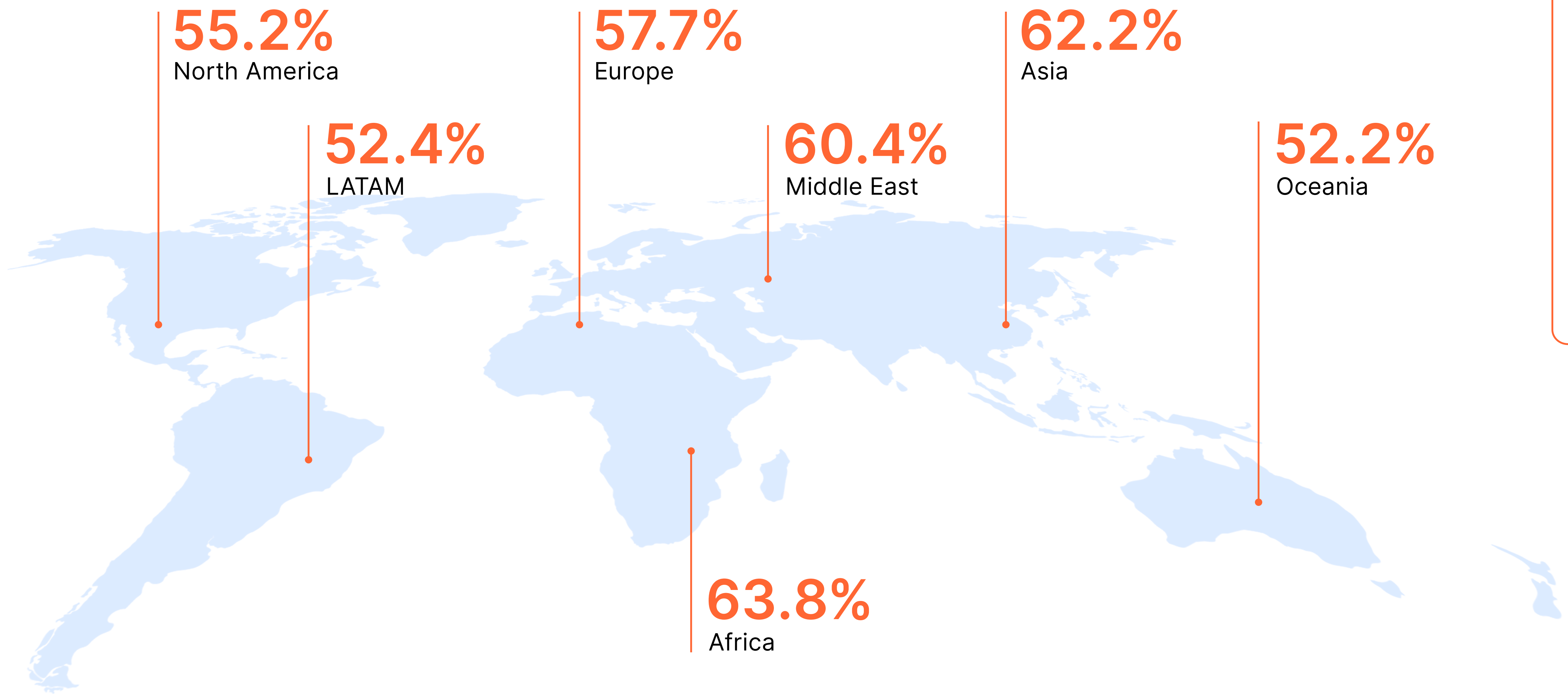
Here are four ways to tackle authentication loopholes that can expose sensitive data through your APIs:

- First, enforce authentication on each publicly accessible API unless there's a business approved exception
- Limit the speed of API requests to your servers to slow down potential attackers
- Block abnormal volumes of sensitive data outflow
- Block attackers from skipping legitimate sequences of API requests

The API-centric world

Regional Trends

Across each region Cloudflare protects, API traffic represented over half of that region's dynamic HTTP traffic[10]:



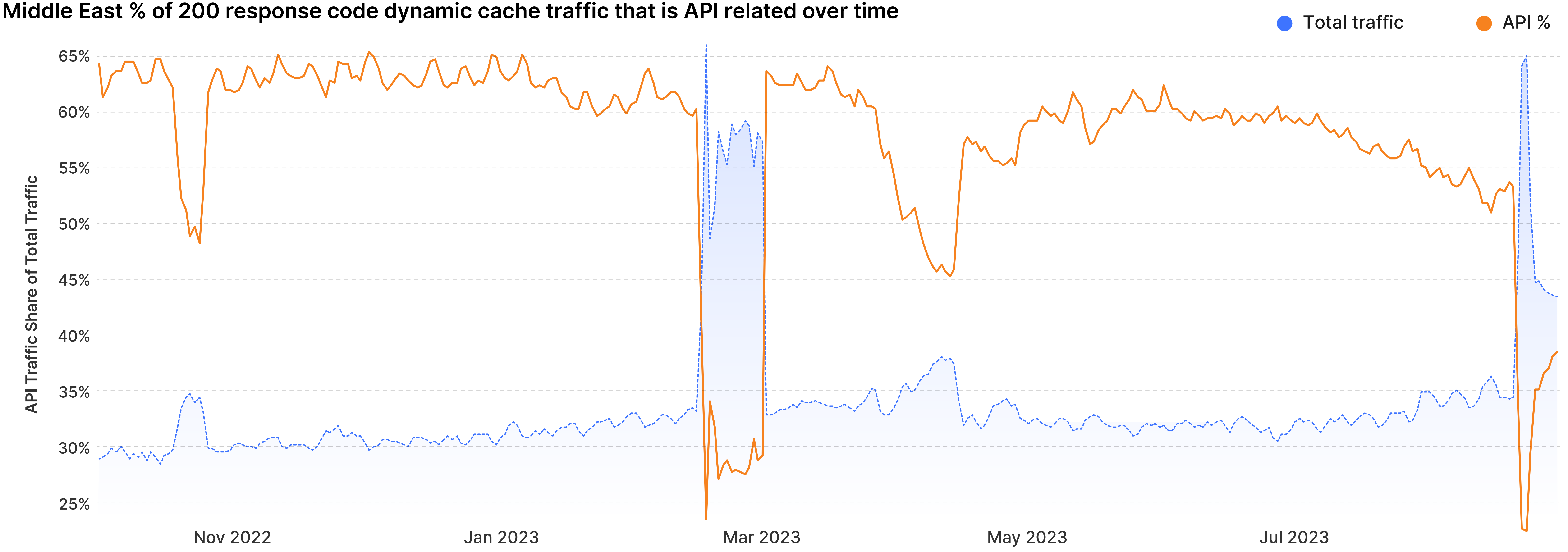
As a whole, total API traffic grew steadily throughout 2023. However, there were noticeable fluctuations in the following regions:

- In **LATAM**, API traffic represented **between 46.1% and 58.6%** of dynamic HTTP traffic
- In **Oceania**, API traffic represented **between 44.1% 57.4%** of dynamic HTTP traffic
- And in the **Middle East**, **API traffic varied the most**, as explored in the next section.

Middle East traffic spikes

The wide swing in API traffic in the Middle East coincided with a sudden, temporary upsurge in **overall traffic to an anonymity tool** known for helping circumvent network restrictions. Cloudflare observed that the 2023 spikes in traffic to the anonymity tool came shortly after government-directed Internet shutdowns.

Middle East % of 200 response code dynamic cache traffic that is API related over time

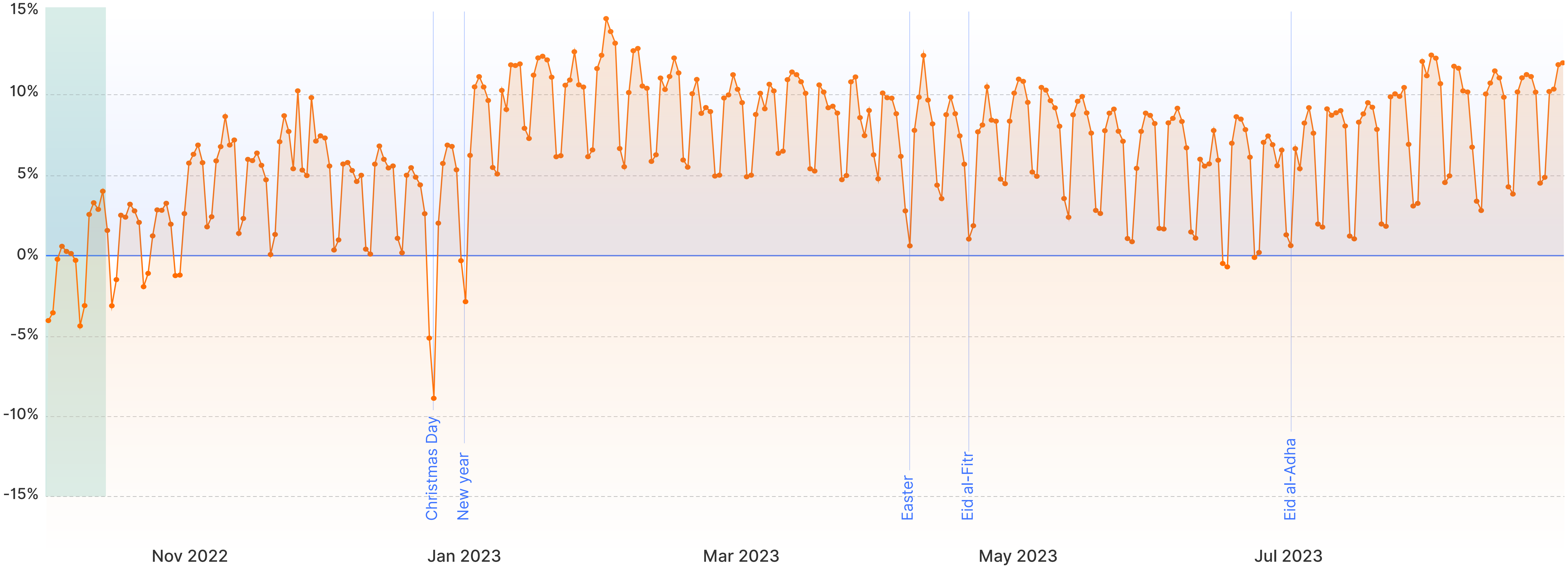


< Table of contents >

Does API traffic slow down?

Although API traffic is often thought of as conversations between bots, Cloudflare's data showed clear spikes and dips in API traffic throughout the year — particularly around major holidays.

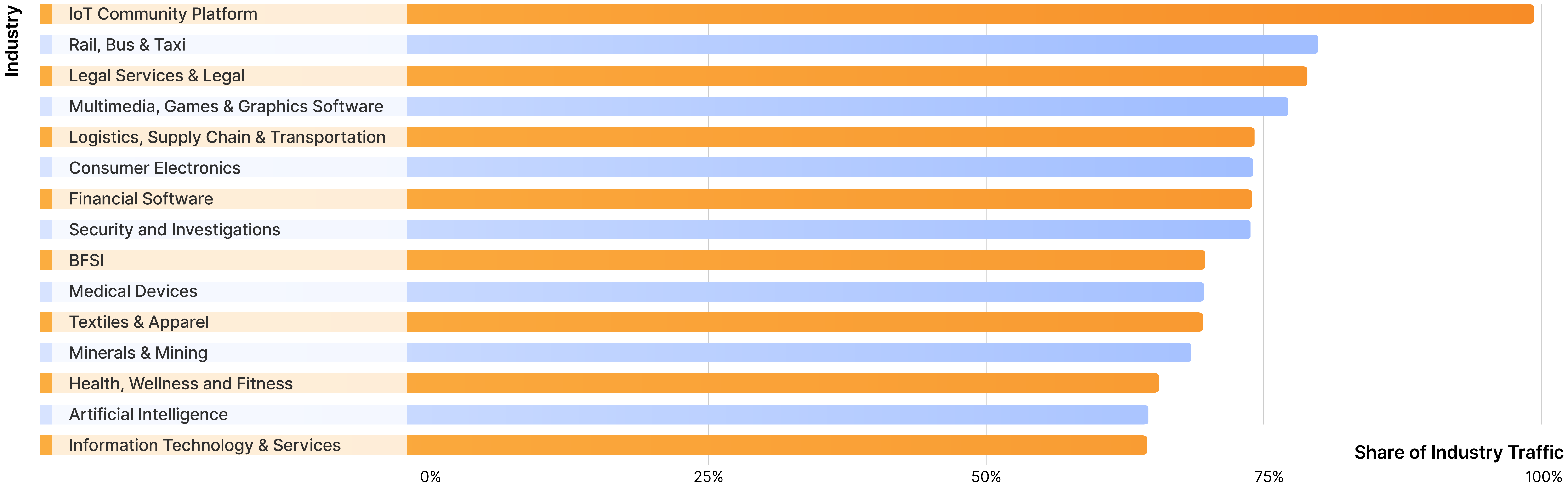
It seemed that when people were more likely to be offline (for example, **coinciding with Dec. 25th (Christmas), April 9th (Easter) or April 22nd (Eid al-Fitr), API traffic noticeably declined.**[11]



API traffic across industries

In addition to geographic variations, **certain industries had a greater share of API traffic compared to other industries**.

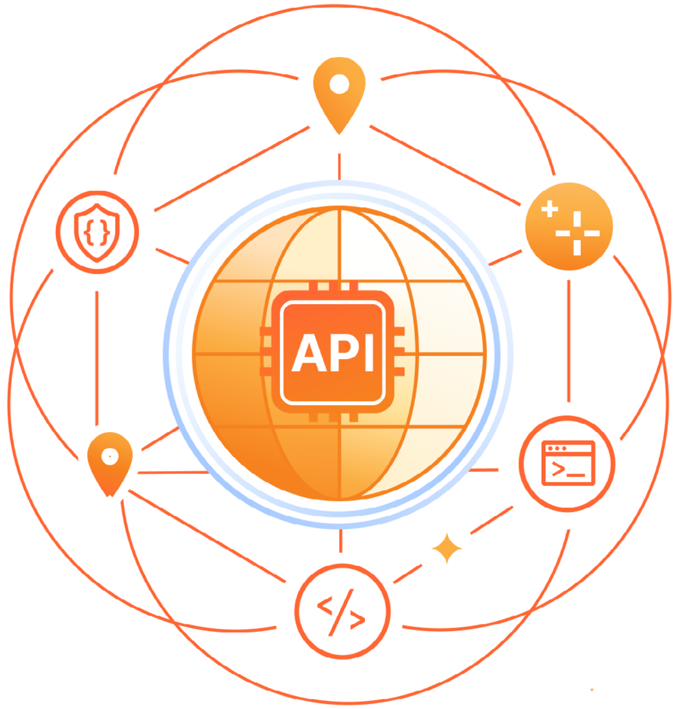
The top 15 industries where Cloudflare saw a higher volume of API-driven traffic (compared to that industry's total dynamic HTTP traffic)[12] were as follows:



Industry benchmarks

Instead of creating new features from scratch, any application, website, or mobile app can enrich the user experience by adding functionalities via APIs.

For instance, rather than creating their own payment services from scratch, a ride sharing app can add payment via the APIs of payment companies; retail APIs help personalize customer experiences with virtual fitting rooms, product recommendations, and order status.



APIs are useful within any industry, anywhere.

Here are the industries with the highest share of API traffic across specific regions[12]:

Africa

1. **Facilities Services**
2. Minerals and Mining
3. Capital Markets
4. Fundraising
5. Credit Cards and Transaction Processing

Asia

1. **IoT Community Platforms**
2. Minerals and Mining
3. Textiles and Apparel
4. Banking, Financial Services and Insurance
5. Artificial Intelligence

Europe

1. **Multimedia, Games and Graphics Software**
2. Content and Collaboration Software
3. Medical Devices
4. Textiles and Apparel
5. Legal Services

Latin America

1. **Minerals & Mining**
2. Financial Software
3. Multimedia, Games and Graphics Software
4. Capital Markets
5. Law Practice

Middle East

1. **Fundraising**
2. Legal Services
3. Wireless
4. Capital Markets
5. Transportation/Trucking/Railroad

Middle East

1. **Legal Services**
2. Rail, Bus and Taxi
3. Consumer Electronics
4. Security and Investigations
5. Logistics, Supply Chain and Transportation

North America

1. **Minerals and Mining**
2. Textiles and Apparel
3. Capital Markets
4. Security and Investigations
5. Pharmaceuticals and Biotech, Health

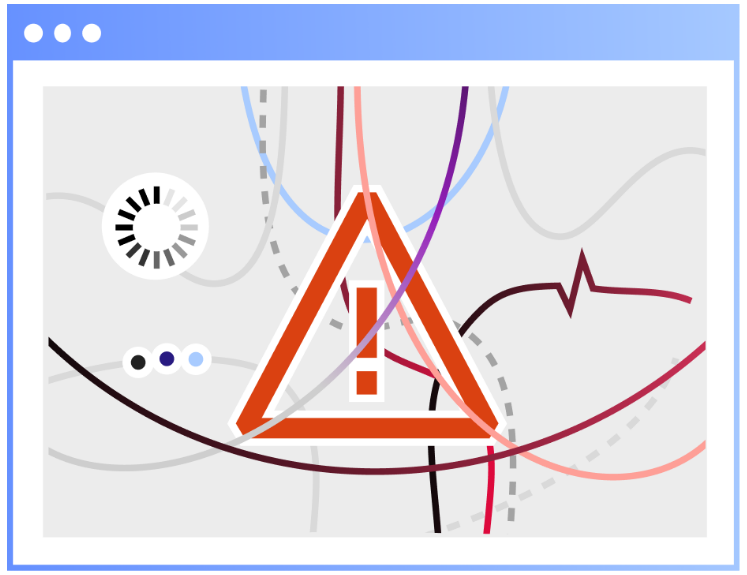
Predictions for 2024 and beyond

As consumers and end users continue to expect faster, more dynamic web and mobile experiences, development and API teams will come under more pressure to deploy and maintain many more APIs. These well-meaning app developers will continue to deploy APIs fast — sometimes without consulting other IT and security stakeholders.

This lack of a cohesive approach will force enterprises into difficult corners as they face the following challenges:

< Table of contents >

1 Increased loss of control and complexity



IT decision makers say that the #1 factor contributing to the loss of control over IT and security environments is the "overall number of applications" — followed by "an increase in locations for applications."

Yet, at most organizations, these teams remain siloed:

73% of developers say that the work or tools their security team requires them to use "interferes with their productivity and innovation."

87% of CIOs believe software engineers and developers "compromise on security policies and controls to get new products and services to market faster."

<50% of CISOs feel that developers are "very familiar" with the security risks of development and workflow tools.

Unless they fix the disconnect between IT, security, and app development with automated API protection, **businesses can expect API risks and management complexity to rise**.

IT, security, and app development teams all share responsibility for protecting the massive attack surface involved with having thousands of API-supported assets.

Unless they fix the disconnect between IT, security, and app development with automated API protection, businesses can expect API risks and management complexity to rise.

Table of contents

2 Easier access to AI leading to more API risks

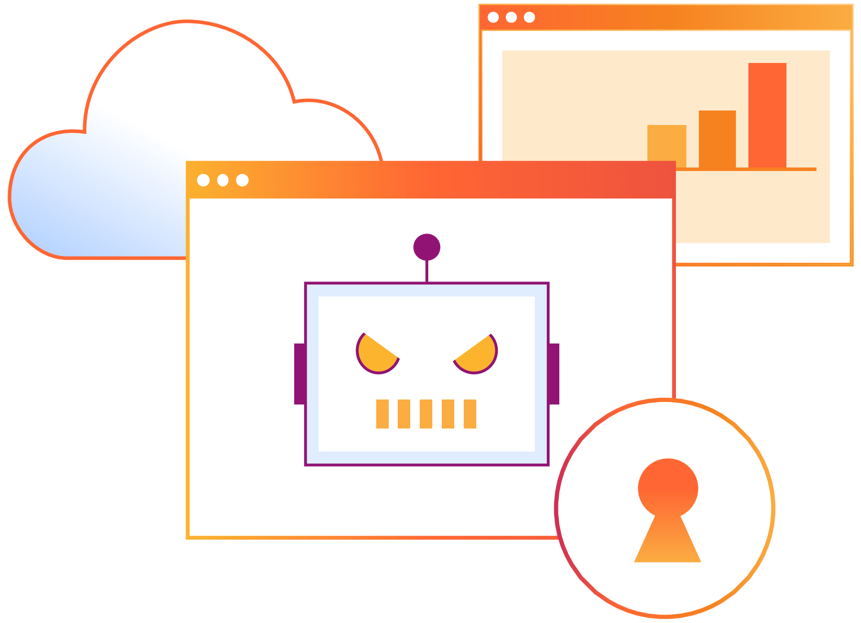
Analysts predict that by 2026, more than 80% of enterprises will have used generative artificial intelligence (GenAI) APIs or models, and/or deployed GenAI-enabled applications (such as ChatGPT) in production environments.

GenAI models (if there is no web app frontend) are generally accessed either directly as an internal function or by other apps and users through public APIs, such as OpenAI's ChatGPT and Whisper APIs. **Because GenAI usage dramatically increases API usage, GenAI services will also attract more API-related attacks against their APIs.**

Consider, for example, that competitors or attackers 'calling' a product API millions of times to scrape and steal data would have a relatively negligible direct cost to a victim's infrastructure bill. But, an attacker leveraging a target victim's generative models via API would cost much more — cents per call. If an attacker makes millions of abusive calls to an AI app's API, it would lead to immediate financial loss.

And, even when GenAI is leveraged for good intentions, it is still uncharted (i.e., risky) territory for many developers. Forrester predicts that, in 2024, without proper guardrails, "at least three data breaches will be publicly blamed on insecure AI-generated code – either due to security flaws in the generated code itself or vulnerabilities in AI-suggested dependencies."

3 Increase in business logic-based fraud attacks



The 2020s saw bot operators targeting web apps using instrumented versions of web browsers to create sophisticated browser-based bots. In parallel, most modern apps use APIs behind the scenes to complete user actions such as account creation, login, form fill and monetary transaction workflows.

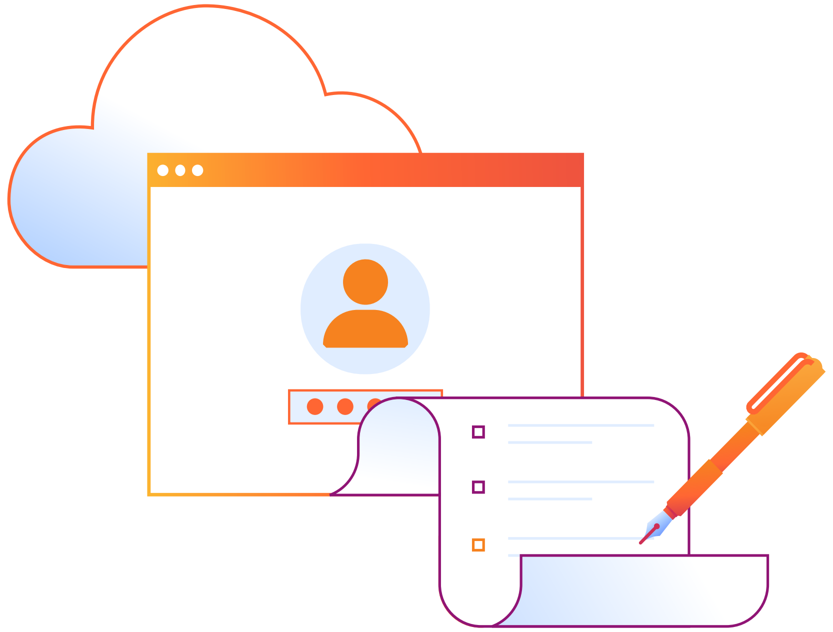
We expect in 2024 that bot operators will increasingly attack the APIs behind those workflows directly, as such attacks are more efficient (APIs tend to change less often than web app UIs) and less protected (compared to web apps).

Take the example of fake account creation in sports betting and fantasy leagues. One person having multiple accounts for different bets and team lineups gives them an increased probability of winning, which can often translate to monetary rewards. Thus, the incentive to automate the creation of new accounts at scale is even more lucrative.

Similar incentives play out with credential stuffing attacks (in which multi-factor authentication is not present or easy to get around) and fraudulent purchasing of limited supply items.

In such cases, organizations need business logic-based intelligence in their API security tooling. For example, identify abnormal sequences attempted by attackers to fast-track their fraud attempt(s). And identify when the API call has abnormal behavior characteristics, such as attempting to complete transactions faster than the baseline transaction volume for that API.

4 Growing regulation and governance



Organizations should also expect **greater governance and efforts to regulate API-related security and privacy**.

For example, the PCI DSS ("Payment Card Industry Data Security Standard") is a framework to guide businesses with processes for managing cardholder transaction and payment authentication data. On March 31, 2024, **the new PCI DSS v4.0 requirements — the first version to explicitly address API security — will go into effect**.

With the release of PCI DSS v4.0, any organization that transmits or processes card payments will be required to address API vulnerabilities, ensure appropriate API authentication, and more. Failure to comply with PCI DSS requirements can lead to steep fines and other penalties.

Healthcare is another highly-regulated industry that can expect more scrutiny around APIs, given their ability to transmit electronic protected health information (ePHI) between systems.

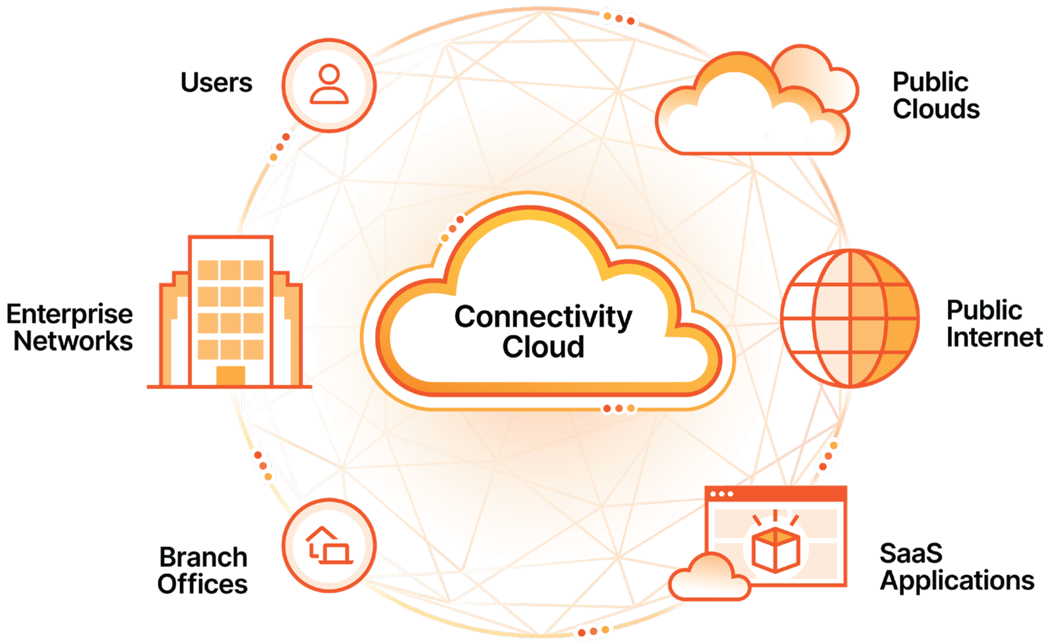
In July 2023, the US Federal Trade Commission and the Health and Human Services Department’s Office for Civil Rights (OCR) stepped up its scrutiny over the privacy risks of health apps — warning of financial penalties for failing to disclose the breach of personal health data.

Recommendations

Like any piece of software, API vulnerabilities will happen. While no one can stop attackers from constantly trying new tactics to break applications and APIs, organizations can identify, protect, and manage their APIs with a holistic approach that incorporates the following best practices:

Table of contents

1 Unify management of app development, visibility, performance, and security with a connectivity cloud



In many businesses, proprietary infrastructure, unique compliance needs, and semi-compatible processes and configurations make it hard to connect SaaS apps, web apps, and other IT infrastructure. Those domains simply weren't built to work together easily and securely.

A connectivity cloud is a new approach for delivering the many services companies need to secure and connect their digital environments. It is an intelligent platform of programmable, cloud-native services that enables any-to-any connectivity between networks, cloud environments, apps, and users.

A connectivity cloud provides the connective tissue between app deployment and API defense-in-depth services that include:

- **Automated API discovery and visibility** that gives organizations a clear inventory of their API estate
- **Modern authentication and authorization** processes built-in from the outset
- **API endpoint management** to monitor metrics such as latency, errors and error rates, and response size for API-driven domains
- **API** Layer 7 (L7) **protections**, including advanced rate limiting and DDoS protection against denial-of-service attacks, brute-force login attempts, and other API abuse
- **Detection of zero-day** (new vulnerabilities found in software that dont have a patch or fix) before a zero-day attack occurs

Table of contents

2 Move toward a "positive security" model with an API gateway

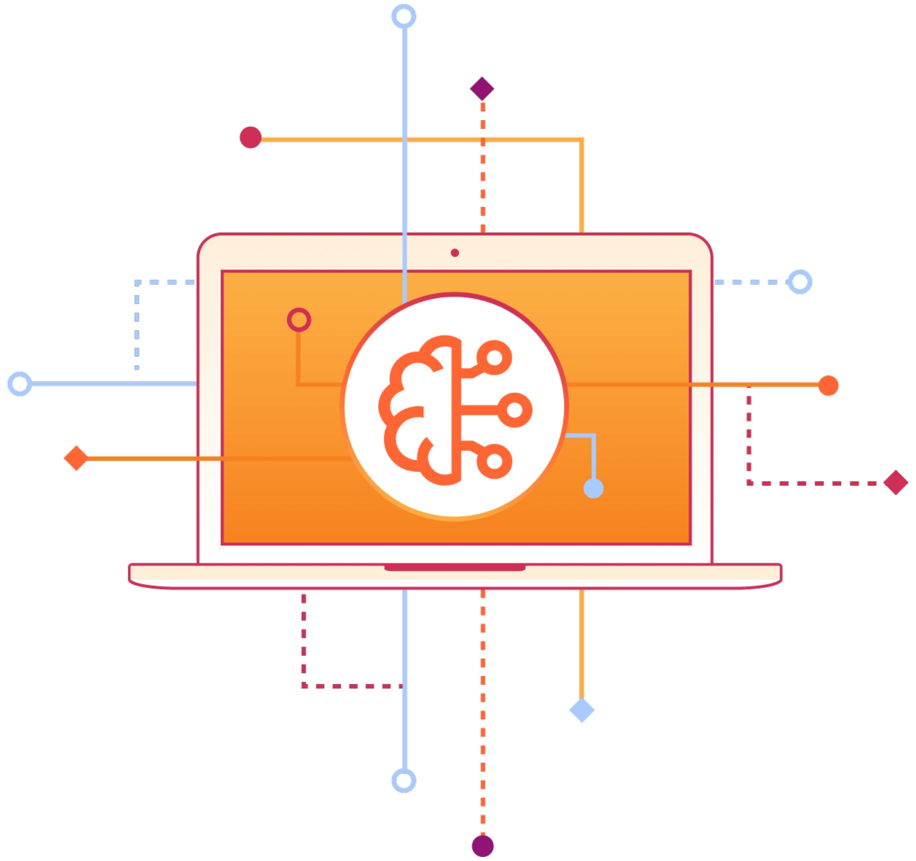
There are an estimated 200 million public and private APIs in use (and growing), and IT and security leaders cannot realistically "keep up" with the performance, behavior, and risk exposure from each API.

Traditionally, web applications are protected with a "negative security" model enforced by a web application firewall (WAF) that allows everything except for requests coming from problematic IPs, ASNs, countries, or requests with problematic signatures (such as SQLi attempts). This is because web apps can be accessed and interacted with by users in a number of ways. In such a model, the WAF will block the "known bad" while permitting all other traffic.

In contrast, a "positive security" model for APIs is more appropriate as APIs have a structured format to interact with them. As the opposite of a negative security approach, **the positive security model permits only "known good" behavior and identities (with "good" defined by the API schema)**, while rejecting everything else.

Organizations using the positive security model protect their APIs by only accepting traffic that conforms to their schemas. They can more effectively block malformed requests and HTTP anomalies, such as credential stuffing attacks and automated scanning tools.

3 Use machine learning technologies that free up resources and reduce costs

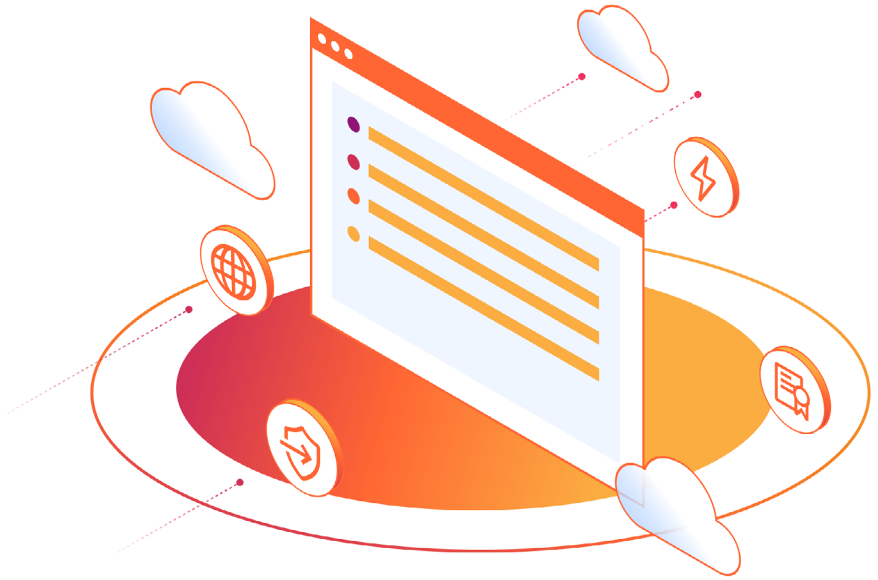


Without automation and purpose-built API tools, IT and security stakeholders have no chance of keeping up with their API teams.

However, organizations can make API visibility and security management more efficient by using machine learning-based security services. For instance, machine learning makes it possible to quickly:

- **Uncover** all API traffic (including unauthenticated APIs) to a domain, regardless of session identifier-based data
- **Detect** attack variations in RCE, XSS, and SQLi attacks on APIs
- **Train** a classifier to distinguish between various traffic types and API attack vectors
- **Differentiate** between legitimate spikes in app user traffic, versus spikes from potentially malicious bot traffic

4 Measure and improve your organization’s API maturity level over time



The most comprehensive approach for protecting APIs is to implement a holistic web application and API protection (WAAP) platform. But, an organization that is just beginning to acknowledge their API exposure may not find this feasible overnight.

However, all progress needs to start somewhere. Once an organization understands what needs to be protected, they can progress toward comprehensive API management and security:

Level 1: Visibility

Companies must first track and formally manage all their API endpoints, including any shadow APIs. However, many organizations cannot find their APIs as fast as their developers build them. And, when they do find APIs, it is difficult to accurately build a unique schema for each of potentially hundreds of API endpoints.

With an API visibility service, organizations can both automatically discover API endpoints and identify who owns that API and how that API should be used.

Level 2: General web attack protection

Web applications and APIs often work together (for example, an ecommerce website using an API to process payments). However, the global nature of the Internet exposes websites and other applications to attacks from many locations, at various levels of scale and complexity.

The following are examples of ‘table stakes’ services (covered in more detail here) to directly protect web applications and the APIs behind them from DoS and DDoS attacks, credential stuffing, zero-day vulnerabilities, and other threat types:

- **DDoS mitigation services** sit between a server and the public Internet to prevent surges of malicious traffic from overwhelming the server
- A **Web Application Firewall (WAF)** filters out traffic known (or suspected) to be taking advantage of web application vulnerabilities
- **Encryption certification management** helps manage key elements of the SSL/TLS encryption process
- **Advanced rate limiting** protects endpoints from DoS attacks, brute-force login attempts, and other API traffic surges — without penalizing legitimate users.

Level 3 — API-specific attack protection

Tools like WAFs and DDoS are critical for web security and the (human) app user's experience, but these services were designed to protect applications — not APIs specifically.

As an organization exposes more services via APIs, they should augment web app security with purpose-built API security and management.

Advanced API security, using unsupervised machine learning, is capable of developing separate baselines for each API, and predicting the intent of API requests (whether legitimate or malicious) as they are made.

Organizations understand API security is new to many in their teams. Security cannot be achieved for its own sake, but for improved and better business outcomes. Some of those benefits are faster product delivery, fewer security loopholes in published APIs, more efficient security teams, and - ultimately - more productive developers and API teams.

Protect APIs as they drive business

Powered by the Cloudflare connectivity cloud, Cloudflare's Web Application and API Protection (WAAP) portfolio integrates leading application security capabilities to keep applications and APIs secure and productive, stop DDoS attacks, block bots, and more.

Learn More

about Cloudflare's API discovery, OWASP API Top 10 protection, mutual TLS, and protecting APIs without compromising innovation.

API security glossary

API call or API request: A message sent to a server asking an API to provide a service or information.

API discovery: API discovery is the process of cataloging all internal and third-party APIs used within an organization.

API endpoint: The place where API requests (also known as API calls) are fulfilled. The API endpoint is almost always hosted on a server.

API traffic: Any HTTP request with a response content type of `XML`, `JSON`, `gRPC`, or similar. Where the response content type is not available, such as for mitigated requests, the equivalent Accept content type (specified by the user agent) is used instead. In this latter case API traffic won't be fully accounted for, but for insight purposes it still provides a good representation.

Bot traffic/automated traffic: Any HTTP request identified by Cloudflare's Bot Management system as being generated by a bot.

Broken object level authorization (BOLA): BOLA is manipulation of object IDs within a request to gain unauthorized access to sensitive data. With BOLA, attackers access objects (data) that they should not have access to, by merely changing the IDs.

Broken user authentication: If authentication is implemented incorrectly, attackers may be able to impersonate API users, enabling them to access confidential data.

Client: The party making the HTTP request. Typically an end user accessing a site on a browser, but may also be an API client or anyone requesting resources from the site.

Directory traversal: Also known as a path traversal attack, directory traversal aims to access files and directories that are stored outside the web root folder.

Distributed denial-of-service (DDoS) attack: A DDoS attack is a malicious attempt to disrupt normal traffic of a targeted server, service, or network by overwhelming the target or its surrounding infrastructure with a flood of Internet traffic.

File inclusion: This vulnerability allows an attacker to include a file in the target application. The vulnerability occurs due to the use of user-supplied input without proper validation.

HTTP anomaly: HTTP anomalies, such as malformed method names, null byte characters in headers, non-standard ports, or content length of zero with a POST request, are common indicators of attacks that are mitigated by Cloudflare's Managed WAF Rules. Detailed descriptions of example HTTP anomaly rules can be found in the Cloudflare blog here.

Examples of **injection attack** types include:

- **Command injection**: When an attacker executes arbitrary commands on the host operating system via a vulnerable application.
- **Cross site scripting (XSS):** XSS is a vulnerability that allows an attacker to inject client-side scripts into a web app in order to access important information directly, impersonate the user, or trick the user into revealing important information.
- **SQL injection (SQLi):** A method by which an attacker exploits vulnerabilities in the way a database executes search queries. Attackers use SQLi to gain access to unauthorized information, modify or create new user permissions, or otherwise manipulate or destroy sensitive data.

HTTP request: The way Internet communications platforms such as web browsers and apps ask for the information they need to load a resource.

Mitigated traffic: Any eyeball HTTP* request that had a “terminating” action applied to by the Cloudflare platform. These include actions such as `BLOCK`, `CHALLENGE` (such as captchas or JavaScript based challenges). This does not include requests that had the following actions applied: `LOG`, `SKIP`, `ALLOW`.

Rate limiting: A technique used in computer systems to control the rate at which requests are processed. It can be used as a security measure to prevent API attacks, or to limit resource usage in your origin servers.

Remote code execution (RCE): Where an attacker runs malicious code on an organization’s computers or network. The ability to execute attacker-controlled code can be used for various purposes, including deploying additional malware or stealing sensitive data.

Schema validation: If an API request does not conform to the API's schema, the API may react in unexpected ways — by revealing confidential data, for instance. Schema validation enables an API to drop such requests.

Zero-day vulnerabilities: These are vulnerabilities unknown to an application’s makers, and which thus do not have a fix available. Attackers look to exploit these vulnerabilities as quickly as possible.

< Table of contents >

HTTP status code descriptions

The following example status codes below (which were the most common API errors as explained in section 8) detail how Cloudflare interprets the Internet standards track protocol for HTTP response codes. Please refer to the current edition of the “Internet Official Protocol Standards” (STD 1) for the standardization state and status of this protocol.

429 means **Too many requests**. Client has sent too many requests in the specified amount of time according to the server. Often known as “rate-limiting”. Server may respond with information allowing the requester to retry after a specific period of time.

400 means **Bad request**. The client did not send a correct request to the server. This is a client error: malformed request syntax, invalid request, message framing, or deceptive request routing.

404 means **Not found**. Origin server was unable or unwilling to find the resource requested. This usually means the host server did not recognize the API URL, which can be caused by a number of different reasons.

401 means **Unauthorized**. The user’s credentials did not exist or did not contain appropriate levels of access for the requested resource.

403 means **Forbidden**. Cloudflare will serve 403 responses if the request violated either a default WAF managed rule enabled for all orange-clouded Cloudflare domains or a WAF managed rule enabled for that particular zone. If you’re seeing a 403 error without Cloudflare branding, this is always returned directly from the origin web server, not Cloudflare, and is generally related to permission rules on your server.

500 means **Internal Server Error**. A generic error message for unexpected errors on the server side.

422 means **Unprocessable Content**. There were semantic errors in the request.

503 means **Service Unavailable**. Server could be down due to maintenance or when your origin web server is overloaded.

430 means **Request Header Fields Too Large**. This error code is not official but is used by Shopify to mean that the request was not accepted because the request might be malicious, and Shopify has responded by rejecting it to protect the app from any possible attacks.

402 means **Payment Required**. Not widely used, but some platforms use it when daily limits have been exceeded, or there was a problem with a payment.

Endnotes

< Table of contents >

1. Cloudflare's global network serves 50 million HTTP requests per second on average, with more than 70 million HTTP requests per second at peak. Between Oct 1, 2022 and August 31, 2023, API traffic with successful responses (200 status code) represented between 53.1% to 60.1% of Cloudflare's dynamic HTTP traffic. Dynamic content is content that changes based on factors specific to the user, such as time of visit, location, and device.
2. For REST API endpoints, Cloudflare's API Discovery found on median 30.7% more endpoints (260 vs. 199) through machine learning than we discovered via customer-provided session identifiers across all customers' domains/zones, per account.
3. Based on most common non-2xx HTTP status codes (including 4xx and 5xx errors) as a percentage of all HTTP errors for APIs (dynamic cache status) between Oct. 1, 2022 and Aug. 31, 2023.
4. For calculating mitigated API traffic, Cloudflare calculated the daily percentage of API traffic mitigated per Cloudflare product source, as well as daily percentage of traffic mitigated by managed rules per the Web Application Firewall (WAF) rule category.
5. Top industries (per the organization's Salesforce industry category) where API traffic comprised over 70% of the overall industry's dynamic HTTP traffic.
6. Based on share of API traffic across North America, Europe, Latin America, Oceania, Asia, Africa, and the Middle East that returned successful (200) response codes out of all dynamic HTTP traffic processed by Cloudflare's network.
7. Cloudflare has two methods of doing API discovery: looking at traffic that contains a session identifier, and using its ML-based discovery engine that doesn't need a session identifier. There were 15,431 accounts that had endpoints only discovered by ML.
8. Based on the aggregated number of APIs per account, broken down by endpoints with write access (PUT, POST, PATCH,DELETE), vs. 'Information-only' (GET) access. For this report's purpose, Cloudflare calculated the percentage of accounts in which GET APIs made up at least 50% of each customer's total number of APIs.
9. Based on API traffic mitigated for customers per the Cloudflare WAF managed rule category
10. Based on the median daily API percentage calculated from the number of API requests within the client's country region that returned 200 response codes and dynamic cache (out of all 200 response code traffic with dynamic cache).
11. Based on per day percent change of API traffic compared to the baseline (average) daily API traffic worldwide.
12. Based on that industry's total dynamic HTTP traffic compared to other industries, where 'industry' is defined by the customer account's Salesforce industry category.

Table of contents

Call: 1 888 99 FLARE
Email: enterprise@cloudflare.com
Visit: www.cloudflare.com