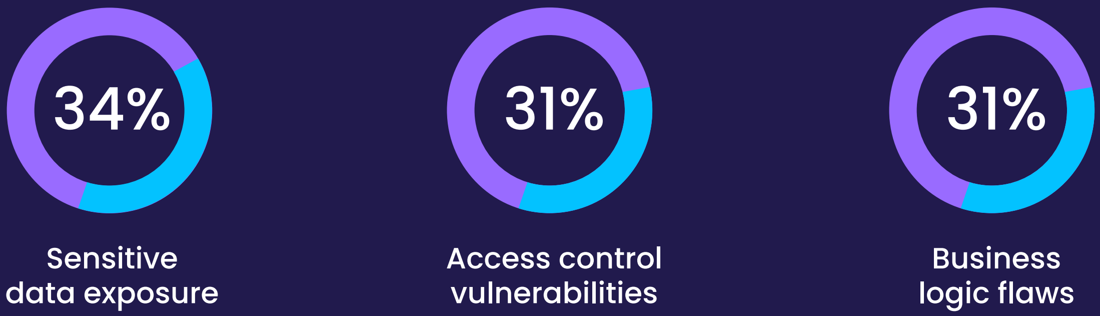
API SECURITY: Best Practices for Mitigating Risks

API Security Done Right

graylog

On the digital highway of modern business operations, application programming interfaces (APIs) are the technical bridges that link applications to one another. APIs give applications a way to talk to one another, sharing the data that makes interoperability possible. According to research from ESG, the growth of API usage is exacerbating security risk. As developers increasingly use APIs in their applications, security teams work diligently to keep pace with the expanding attack surface. According to the report, organizations worry about possible exposure and the wide range of API security susceptibilities that could expose them to attacks, rating the following as their top three concerns:



Despite these concerns, research clearly indicates that organizations believe APIs are worth the risks. However, organizations need a clear understanding of the risk to implement appropriate mitigation, remediation, detection, and response controls.

Although API and web application attacks have similar naming conventions, malicious actors use different processes to achieve their objectives. Traditional web application attacks are transaction-based, meaning malicious actors have a repeatable set of steps for gaining unauthorized access. Since these attacks follow patterns, signature-based security tools can block or detect them.

Understanding the Types of API Attacks

API attacks are part of larger application logic attacks. Threat actors engage in reconnaissance to learn *how* disrupting the API's functionality can impact the organization's systems or enable them to gain unauthorized access to data. The API attack typically involves uncovering the underlying business logic and cross-application communications.

BRUTE FORCE ATTACKS

APIs can employ different methods for authentication and authorization, and some are less secure than others. For the less secure authentication methods, attackers can send API requests for known or suspected users to determine whether a supplied password was correct. Depending on the response, the attackers may be able to make assumptions about the access and use that to gain additional information.

For example, a POST call may use a Basic Authentication header that allows the attacker to enumerate users or return a RESPONSE that gives the attacker insight into whether the user is valid.

DISTRIBUTED DENIAL OF SERVICE (DDOS) ATTACK

Threat actors send high volumes of requests through the API, typically using a network of infected devices known as a botnet. As the API forwards the requests to the backend client, both become overloaded with inbound requests, interrupting legitimate traffic and causing a service disruption.

Malicious actors generally use DDoS attacks to disrupt business operations and lose revenue.

INJECTION ATTACKS

Attackers insert malicious data from an untrusted source, hoping that when the API executes the command, it provides insight into how the application connects to the backend server or database.

With an injection attack against an API, malicious actors may gain information about the API endpoint, including:

- User ID
- FIle paths to databases
- Payment information
- Personally identifiable information

MAN IN THE MIDDLE (MITM) ATTACK

An MitM attack occurs when malicious actors intercept communications between the API endpoint and the client. When APIs lack TLS or SSL cryptographic protocols, attackers can gain access to any data traveling between the client and server. For example, a REST API that uses HTTP instead of HTTPS creates a MitM attack risk.

DATA EXPOSURE

When the API makes a request to the application, the response can include sensitive data back to the client. When attackers make these request types to an application service, they hope that the API's response includes sensitive information, like:

- Personally identifiable information
- Payment information
- User credentials

EXPOSED API KEYS

The API key authorizes an application on a remote device to connect with the backend database. However, an API key is sometimes hard-coded into an application as a string or byte array in the code or an asset file. Attackers can reverse engineer the application to scan the code and retrieve the hard-coded API key.

PARAMETER TAMPERING

Whenever the API calls the backend server, it exchanges information called parameters. Attackers can manipulate these parameters to change data, like user credentials and permissions. Doing this allows them to gain further unauthorized access to the application, network, system, or data.

Threats and Vulnerabilities: Just the Tip of the Iceberg

The Open Web Application Security Project (OWASP) publishes various lists of security risks to help organizations and software developers improve security. The OWASP Top 10 Security Risks list identifies critical API security risks based on:

- Publicly available API threat intelligence
- Community discussions and feedback
- Meetings with industry members for insights
- Review and discussion of the list to ensure continued applicability

Although many of the Top 10 API Security Risks may look similar to the Top 10 Web Application Security Risks, the two are separate because attackers can leverage the vulnerabilities differently. For example, both lists contain Security Misconfiguration. Still, they define it differently based on how attackers can exploit the vulnerability:

Application Top 10: Security Misconfiguration	API Top 10: Security Misconfiguration
Missing appropriate security hardening across any part of the application stack or improperly configured permissions on cloud services.	Appropriate security hardening is missing across any part of the API stack, or if there are improperly configured permissions on cloud services.
Unnecessary features are enabled or installed (e.g., unnecessary ports, services, pages, accounts, or privileges).	Unnecessary features are enabled (e.g., HTTP verbs, logging features).
The software is out of date or vulnerable.	The latest security patches are missing, or the systems are outdated.
Error handling reveals stack traces or other overly informative error messages to users.	Error messages include stack traces or expose other sensitive information.
Default accounts and their passwords are still enabled and unchanged.	There are discrepancies in how servers in the HTTP server chain process incoming requests.
For upgraded systems, the latest security features are disabled or not configured For upgraded systems, the latest security features are disabled or not configured securely.	Transport Layer Security (TLS) is missing.
The security settings in the application servers, application frameworks (e.g., Struts, Spring, ASP.NET), libraries, databases, etc., are not set to secure values.	Security or cache control directives are not sent to clients.
The server does not send security headers or directives, or they are not set to secure values.	A Cross-Origin Resource Sharing (CORS) policy is missing or improperly set.

The difference between application security and API security becomes clearer when looking at these vulnerabilities. Despite the similarities, like unnecessary features being enabled, application security focuses on ports or accounts, while API security focuses on HTTP verbs and logging features.

Organizations need tools that respond to APIs' unique security risks to mitigate risks adequately.

1. Broken object level authorization (BOLA)

The API shares endpoint information that attackers can use to gain unauthorized access to data.

Mitigating this risk requires engaging in object level authorization checks for every function with a user ID to access a data source.

2. Broken authentication

Attackers can use incorrectly implemented authentication mechanisms to use credential-based attacks, assuming a user's identity to gain unauthorized access.

Mitigating this risk requires understanding all policy API authentication flows and implementing multi-factor authentication (MFA) when possible.

3. Broken object property level authorization

The authentication flow exposes object properties, enabling attackers to change, add, and/or delete values.

Mitigating this risk requires limiting:

- User access to object properties
- Data that the API returns

4. Unrestricted resource consumption

Attackers can deploy a DDoS attack when resource limits are incorrectly set or missing, allowing the API to take up resources like bandwidth, CPU, memory, and storage.

Mitigating this risk requires rate limiting and throttling to prevent the API from using too many resources at any given time.

BFLA is a BOLA at the code level, meaning that user roles or permission have too much access. For example, a standard user may be able to access an administrative endpoint.

Mitigating this risk requires configuring and monitoring the application's authorization module using a deny-all-by-default model.

6. Unrestricted Access to Sensitive Business Flows

Since APIs often define and restrict how an application uses a database, excessive access to business flows can expose sensitive data or change how the application manages data.

Mitigating this risk requires identifying all business flows and choosing appropriate protection mechanisms, like device fingerprinting, CAPTCHAs, bot detection tools, or IP blocking.

7. Server Side Request Forgery (SSRF)

Attackers can "fake" a requested source when the API fetches data from a remote source without validating the user-supplied data.

Mitigating this risk requires isolating resource fetching mechanisms and using allow lists to define accepted remote resources.

8. Security Misconfiguration

Attackers can use insecure default or complex security configurations to identify unpatched flaws, common endpoints, or unprotected files and directories.

Mitigating this risk requires implementing security across the API lifecycle, including repeatable hardening processes, configuration monitoring, and automation for assessing configurations' security effectiveness.

9. Improper inventory management

As an organization adds more APIs, it can may lose visibility into the number of APIs and API endpoints creating two distinct blindspots:

- Lack of API documentation
- Lack of visibility into data flows, like where APIs share sensitive data

Mitigating this risk requires using automation to inventory and document all API hosts and integrated services.

10. Unsafe consumption of APIs

Attackers can identify third-party services to compromise Organization standards for input validation and sanitization or int without a comprehensive API inventory acts over unencrypted channels.

Mitigating this risk requires engaging in appropriatcannotAPI security due diligence, using secure communication channels, validating and sanitizing received data, and maintaining an appropriate allowlist.

API Security: 5 Best Practices for Mitigating Risk

Mitigating the Top 10 API Security Risks is fundamental for basic API security hygiene as security teams seek to fully assess and monitor their organizational attack surfaces. However, as APIs are a relatively new technology, the original 2019 list contained the following risks that are not included in the 2023 version, like:

- Excessive Data Exposure: exposing all object properties without considering sensitivity
- Mass Assignment: Failing to filter properties according to an allowlist
- Injection: Sending untrusted data to an interpreter as part of a command or query
- Improper Asset Management: Ensuring proper and updated documentation, like deprecated API versions and exposed debug endpoints
- Insufficient Logging and Monitoring: failure to collect appropriate event data coupled with missing or ineffective integration into incident response processes

OWASP notes in the introduction to the 2023 Top 10 List that API security is rapidly evolving. OWASP specifically explained that the 2023 project does not replace the 2019 list and that organizations should review both for comprehensive security.

For holistic security, organizations should be looking for security solutions focused on the unique risks that APIs pose so that they can implement the following best practices.

1. AUTOMATE API DISCOVERY

Before an organization can secure its APIs, it needs to create a comprehensive inventory and identify the APIs with sensitive data. Without a comprehensive API inventory, organizations lack visibility into all potential attack vectors. Simultaneously, without visibility into the APIs with sensitive data, organizations have no way to assess risk significantly or prioritize remediation actions.

With an automated solution that discovers active, unmanaged APIs, organizations increase productivity and reduce development costs by eliminating manual, often inaccurate processes. By creating an inventory of internal, external, third-party, managed, unmanaged, zombie, and shadow APIs, their security teams can fully assess and monitor the organizational attack surface.

When security teams have reliable insight into their API landscape, they can incorporate this data into their alerts and quickly detect incidents, especially those arising from rogue or "shadow IT" APIs.

Organizations need visibility into all APIs running in their environments and solutions that enable

- Runtime discovery all API formats
- Discovery of all API types, like internal, external, third-party, managed, unmanaged, shadow, and zombie
- Continuous API monitoring across different formats and types

When considering a solution, organizations should look for the following capabilities:

- **API Capture:** complete visibility into your API attack surface through network, gateway, and in-application capture options with full HTTP header/body capture for request and response to enable complete situational awareness
- **Asset Classification:** automatically classifying the data that flows through any API.
- **Continuous Discovery:** incorporate into overarching continuous security monitoring
- **Risk Scoring:** assess the security posture of your APIs and receive actionable intelligence into your current API security risk.

2. INTEGRATE INSIDE-PERIMETER MONITORING WITH EXTERNAL MONITORING TOOLS

Comprehensive API security requires a set of solutions that provide inside-perimeter and external monitoring. An organization needs its WAF and API gateways because they help monitor the runtime environment and can detect events like:

Attacks

Threats

Leaks

Performance weaknesses

At the same time, they need solutions that enable real-time visibility into:

- API request attacks and threats that evaded perimeter defenses
- Leaks and performance issue in API responses

By augmenting WAF and API gateway security capabilities with API security solutions focused on inside-the-perimeter monitoring, organizations can implement comprehensive and holistic monitoring across the API attack surface.

By their powers combined, WAF, API gateways, and API security monitoring solutions enable:

- Runtime scanning without impacting app performance
- Client-side request monitoring without slowing down traffic
- Robust search/reporting with detailed analysis so that security teams can investigate incidents faster

When considering a solution, organizations should look for the following capabilities:

- **Request and Response Capture:** Automatically capture the header and body data of all requests and responses for REST APIs and GraphQL queries for advanced threat hunting and API-specific remediation
- **Threat and Vulnerability Scanning:** threat signatures that go beyond OWASP for immediate risk reduction.
- **Remediation Suggestions:** detailed, targeted, and customizable instructions

3. AGGREGATE AND CORRELATE API MONITORING FOR HIGH-FIDELITY DETECTIONS

With inside-perimeter context, security teams can create high-fidelity detections for some of the most risky, challenging security issues arising from APIs. With the ability to identify authenticated attackers, security teams can reduce risk arising from attackers that easily bypass perimeter security and have access to their runtime, like:

- Fake or caustic customers,
- Disgruntled employees,
- Supply-chain spies,
- Partner contractors

When security teams integrate API context and visibility into their monitoring, they can tune alerts which reduces alert fatigue and increases response effectiveness. API security tools that provide a pre-configured base set of detections and alerts enable security teams to quickly turn search and investigation results into a customized alert unique to their API environment. By connecting all alerts back to the API and security monitoring tools, they can display the exact context that initiated the alert and include customizable response and remediation guidance.

For example, solutions should capture and analyze different characteristics beyond signatures related to API security that perimeter security typically misses, including issues like response leaks and performance issues. By mapping key header and body data to security-related fields, the teams can more easily complete follow-on security functions.

When considering a solution, organizations should look for the following capabilities:

- **Targeted Alerting:** Precise routing of alerts directly to Security and/or DevOps teams via Slack, Teams, Gchat, or Zapier.
- **Real-Time Threat Detection:** security issues at runtime, generating well-tuned alerts with full context and customized remediation guidance.
- **Integrate with SIEM:** Automatically send critical security alerts to SIEM.

4. BUILD API SECURITY INTO INCIDENT RESPONSE PLAN

Responding to API attacks can be complicated because the root cause can be any number of different events, like:

- A security policy that SecOps needs to address
- A deployment config issue DevOps needs to address
- An software defect DevOps needs to address

Additionally, the current adversary attack frameworks that security teams use to build detections and respond to incidents fail to respond to the unique issues that APIs create. Security teams need to adapt the MITRE ATT&CK framework or other kill-chain models so that they can incorporate API security into their incident response processes.

To effectively incorporate APIs into the organization’s incident response plan, security teams need API security solutions that:

- Provide real-time information about attacks
- Store all captured and analyzed API transactions

Without capturing and analyzing all API transactions, security teams struggle to engage in comprehensive, full-scale investigations for sophisticated API attacks. By building high-fidelity alerts focused on APIs, the team that needs to respond can automatically receive notification, ensuring that incident responses are fast and effective.

API security solutions that feed into a local security data lake enable the organization to fully optimize the security data to more rapidly investigate sophisticated attacks that evolve and progress over time frames measured in weeks and months.

When considering a solution, organizations should look for the following capabilities:

- **Search Capabilities:** standard SQL and regex queries
- **Integrate with SOAR:** incorporate into automated incident response
- **Feed To Local Data Lake:** enables detailed analysis and investigation to reduce Mean Time to Investigate and Mean Time to Recover

5. INCORPORATE INTO COMPLIANCE REPORTING

APIs are critical to business operations, and their security is critical to the organization's compliance posture. As legislative bodies implement more stringent data protection laws, organizations need to incorporate APIs into their compliance documentation.

Simultaneously, they need documentation that responds to different needs. Senior leadership needs high-level visibility into security posture while auditors may need more technical information. An API security solution should provide visualizations that enable at-a-glance visibility into trends while also giving the technical stakeholder a way to dig deeper into controls' effectiveness.

When considering a solution, organizations should look for the following capabilities:

- Dashboards with visualizations that provide holistic insights into trends
- Visibility into the types of data that traverse APIs and who uses them
- Mapping with attack paths to track malicious actors
- Real-time visibility into data flows to prevent violations of company policies, regulations, or industry compliance frameworks, like the OWASP Top 10
- Insight into sensitive information flowing through APIs to ensure compliance with data privacy laws

graylog

API SECURITY

CONTINUOUS API THREAT DETECTION & INCIDENT RESPONSE

Graylog API Security is continuous API security, scanning all API traffic at runtime for active attacks and threats. Mapped to security and quality rules, Graylog API Security captures complete request and response detail, creating a readily accessible datastore for attack detection, fast triage, and threat intelligence. With visibility inside the perimeter, organizations can detect attack traffic from valid users before it reaches their applications.

Graylog API Security captures details to immediately identify valid traffic from malicious actions, adding active API intelligence to your security stack. Think of it as a "security analyst in-a-box," automating API security by detecting and alerting on zero-day attacks and threats. Our pre-configured signatures identify common threats and API failures and integrate with communication tools like Slack, Teams, Gchat, JIRA or via webhooks.

ABOUT GRAYLOG

Graylog is a game-changing cybersecurity firm, revolutionizing the way organizations protect against cyber threats. Our solutions are crafted with the latest advancements in AI/ML, security analytics, and intelligent alerting, offering unparalleled threat detection and incident response capabilities. Graylog stands out by making advanced cybersecurity accessible and affordable, ensuring businesses can easily implement robust defenses against the evolving landscape of cyber threats, including the critical vulnerabilities associated with APIs. Our commitment to innovation and simplicity positions Graylog as the go-to partner for businesses seeking to enhance their cybersecurity posture without the complexity and high costs of traditional solutions. For more information on how Graylog can fortify your cybersecurity, visit our website at graylog.org.