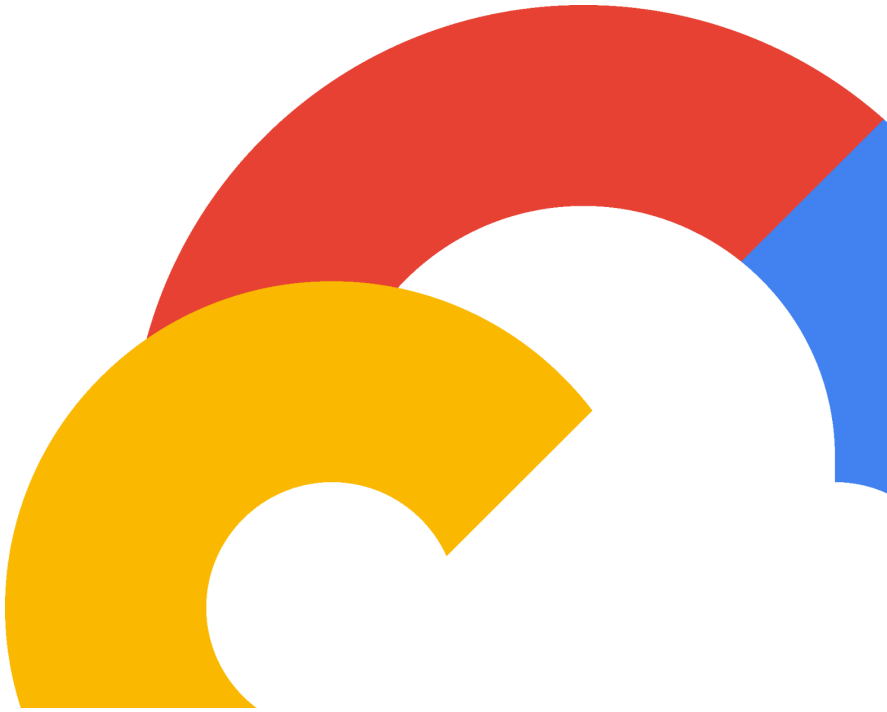
Google Cloud Security

Apigee

Mandiant

Mitigating OWASP Top 10 API Security Risks with Apigee and Advanced API Security

December 2024



For more information visit **cloud.google.com**

Keep your APIs secure.

OWASP's 2023 Top 10 for API Security list outlines the latest cybersecurity risks associated with APIs. By using Apigee and Advanced API Security together, you can improve your security posture and better protect your organization against these common attack types.

Executive Summary

APIs are essential to modern software, enabling communication between systems and facilitating digital partnerships and ecosystems. But the widespread use of APIs makes them attractive targets for cyberattacks. This document describes the key risks associated with APIs, relying heavily on work done by the OWASP project. This document also describes some best practices for addressing those risks using Apigee, Google Cloud's API management platform, and Advanced API Security, a security add-on for Apigee.

By implementing the recommendations outlined in this whitepaper and taking advantage of both Apigee's native security capabilities and additional capabilities provided in Advanced API Security, organizations can establish a comprehensive defense against the OWASP Top 10 API Security Risks and other emerging threats. This proactive approach can significantly reduce the risk of successful attacks, safeguard sensitive data, and maintain the trust of customers and partners. A well-designed and implemented security strategy is crucial for protecting your API ecosystem and ensuring the long-term success of your digital initiatives.

This document includes the following information:

01

Overview of the OWASP top 10 API security risks

The most recent list of OWASP Top 10 API security risks, updated in 2023.

02

Details and potential impact on your organization

More information on each security risk and how it can impact your organization.

03

Mitigation strategies with Apigee

Ways to mitigate these security risks using Apigee and Advanced API Security.

About Apigee and Advanced API Security

Apigee is Google Cloud's native API management platform that can be used to build, manage, and secure APIs — for any use case, environment, or scale. Apigee offers high performance API proxies to create a consistent, reliable interface for your backend services. The proxy layer provides granular control over security, rate limiting, quotas, analytics, and more for all of your services. Apigee supports REST, gRPC, SOAP, and GraphQL, providing the flexibility to implement any API architectural style.

Advanced API Security is an add-on to Apigee that provides additional security checks and controls for APIs. It integrates with existing Apigee environments and native security features, including OAuth 2.0, OpenID Connect, and JSON Web Tokens (JWTs), and validates they have been configured properly in the customer's environment. It also detects undocumented and unmanaged APIs linked to Google Cloud L7 Load Balancers. With Apigee Advanced API Security, you can regularly assess the risk of your APIs; surface API proxies that do not meet your security standards; and get recommended actions for how to mitigate detected issues. Advanced API Security's abuse detection uses machine learning models to detect patterns that are a sign of malicious activity, including API scraping and anomalies, and cluster events together based on similar patterns.

About Mandiant, now part of Google Cloud

Mandiant (now part of Google Cloud) delivers dynamic cyber defense solutions by combining consulting services, threat intelligence, attack surface management, and validation powered by industry-leading expertise, intelligence, and innovative technology. Part of Mandiant's consulting services include Incident Response Services to help Google Cloud customers investigate and remediate incidents faster.

About OWASP's Top 10 API Security Risks

The Open Worldwide Application Security Project, or OWASP, is a nonprofit organization that works to improve software security through its community-led open source software projects. OWASP publishes multiple "Top 10" lists outlining security risks, in categories spanning web applications, mobile applications, databases, and more. Each list summarizes a broad consensus about the most critical security risks in the respective domain. The OWASP "Top 10" list for API security, initially published in 2019 and revised multiple times since then, describes the exploitability, prevalence, and impact for acute API risks, and guidance for developers to avoid these problems. This list, like all the OWASP Top 10 lists, is intended to be a way to promote awareness and education among developers, to allow improvements in security.

Disclaimer

The information provided in this document is for general informational purposes only and should not be considered as legal, financial, or security advice. Organizations should conduct their own risk assessments and consult with qualified professionals to determine the appropriate security measures for their specific needs.

Note: The product "Apigee" referenced in this whitepaper refers to Apigee X and Apigee hybrid, and does not extend to Apigee Edge or Apigee Edge for Private Cloud (OPDK).

Table of Contents: OWASP Top 10 API Security Risks

1 - Broken Object Level Authorization (BOLA)

Background

Broken Object Level Authorization (BOLA) is a critical security vulnerability that arises when an API fails to adequately enforce access controls on individual data objects. This oversight allows malicious actors to gain unauthorized access to, manipulate, or even delete data they should not have permission to, leading to severe consequences such as unauthorized data exposure, modification, or deletion.

BOLA vulnerabilities can have severe consequences for organizations. Unauthorized data exposure can lead to sensitive information being leaked to the public, which could damage an organization's reputation and result in legal liability. Unauthorized data modification can disrupt an organization's operations and lead to financial losses. Unauthorized data deletion can result in the permanent loss of critical information.

BOLA vulnerabilities typically occur when an API does not properly validate the user's authorization before performing operations on specific data objects. For example, an API might allow a user to modify a data object without checking if the user has the necessary permissions. This could enable an attacker to modify sensitive data, such as financial records or personal information, without authorization.

BOLA vulnerabilities can also occur when an API allows users to access data objects through indirect references. For example, an API might allow a user to access a data object by providing its identifier. If the API does not properly validate the user's authorization before allowing access, an attacker could use this indirect reference to access data objects they should not have permission to read or modify.

To mitigate BOLA vulnerabilities, organizations should implement strong access controls on individual data objects. This can be achieved by using various techniques, such as role-based access control (RBAC), attribute-based access control (ABAC), or mandatory access control (MAC). Additionally, organizations should regularly review their API security configurations to ensure that access controls are properly enforced.

Preventing the BOLA vulnerability with Apigee

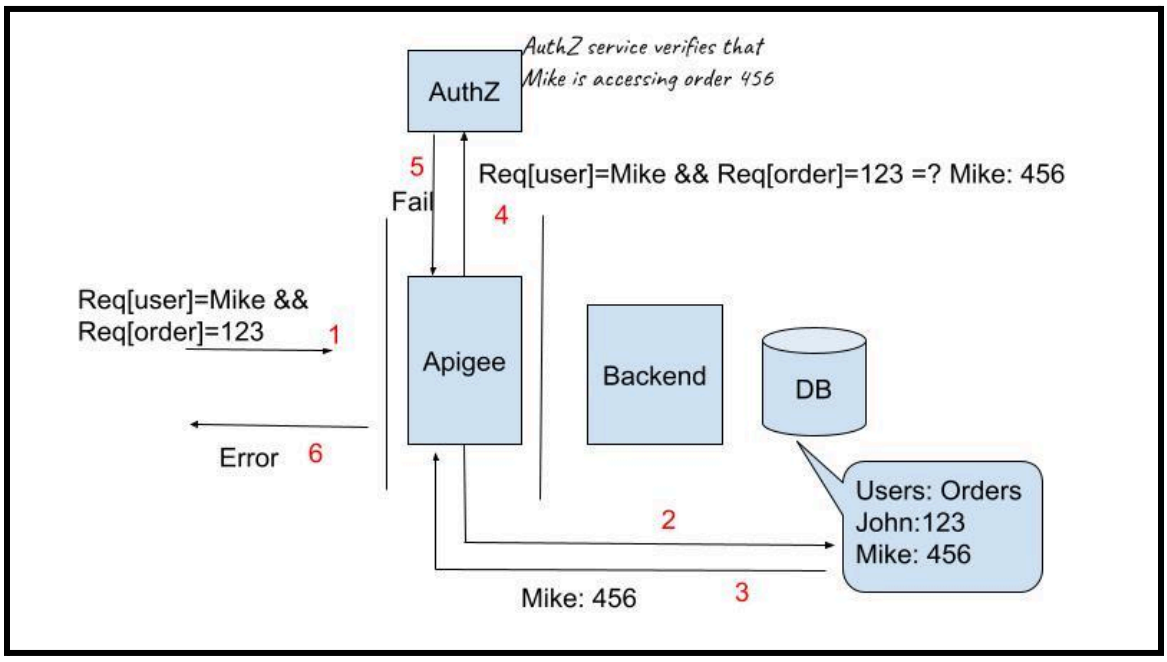


Figure 1: BOLA attack workflow

In Figure 1 above, Mike is trying to access John's order:

1. The client Mike is initiating a request to access an order with ID 123
2. The Apigee proxy sends a request to the backend, asking for Mike's orders
3. The backend responds that Mike owns order 456
4. Apigee validates with AuthZ service whether Mike can access order ID 123, given the response from the backend
5. AuthZ service answer is a failure, which means Mike can't access order 123
6. Apigee returns an error to the user

Apigee: Native security features for mitigation

Feature	Description
Quota management	Implementing quota management can prevent excessive or abusive API consumption that could potentially exploit BOLA vulnerabilities. Quota management involves setting limits for each API endpoint or resource, monitoring API usage, and enforcing the defined quotas. This helps organizations control API usage patterns, optimize resource allocation, and prevent potential security threats arising from uncontrolled API consumption. If your API is vulnerable to a Broken Object Level Authorization (BOLA) attack, attackers might figure out how it works and use stolen secrets to access it repeatedly. To protect against this, users should set usage quotas on their APIs. Apigee can then monitor these quotas and alert users if their APIs are being called excessively, which could indicate an

	attack. Apigee X allows administrators to define usage limits or quotas for each API. These quotas can be configured based on various metrics such as the number of requests per minute, hour, or day, or the total data transfer volume. For Generative AI use cases, Apigee can even enforce quotas based on token count in the prompt, or in the response.
OAuth 2.0 and OpenID Connect	Apigee X uses industry-standard protocols for secure authorization and authentication. OAuth 2.0 enables delegated access, allowing users to grant third-party applications access to their resources without sharing their credentials. OpenID Connect extends OAuth 2.0 by adding an identity layer, enabling single sign-on (SSO) across multiple applications and services. These protocols provide a comprehensive framework for protecting user identities and managing API access.
Sensitive data protection	By protecting sensitive data, organizations can reduce the impact of BOLA attacks and comply with data protection regulations including GDPR and PCI DSS. All sensitive object access needs to go through an API gateway with a non-passthrough URL. There needs to be a level of indirection over the storage layer so the authentication checks can remain independent of the raw storage resource. Apigee users can implement Sensitive Data classification through the Service Callout policy. That way, the proxies can identify sensitive data such as PII, financial information, and healthcare data.

Advanced API Security: Additional features to mitigate BOLA vulnerabilities

Feature	Description
Risk assessment	The Risk Assessment feature in Advanced API Security identifies API proxies with missing or inadequate security policies, including missing authentication verification policies that leave them exposed to potential Broken Object Level Authorization (BOLA) attacks.

	It also provides security teams with actionable recommendations to improve their security posture. These recommendations include suggestions for hardening API proxies, and implementing proper authentication and authorization mechanisms.
ML-powered abuse detection	The Abuse Detection feature empowers security teams with valuable insights into irregular proxy traffic by leveraging advanced machine learning algorithms. It uses anomaly detection algorithms to identify unusual patterns in API requests, such as sudden spikes in traffic volume or requests from unfamiliar IP addresses; these could indicate BOLA attacks. These detections can be effectively consolidated, automatically clustering similar attacks into an incident. This capability aids security teams in understanding whether an attack is isolated or more widespread, ensuring an appropriate and effective response to potential threats.
Security actions	With Advanced API Security, users can create automated security actions to immediately flag or block suspicious traffic identified through Apigee's Abuse Detection models.

Examples of supported Apigee mitigations to prevent BOLA vulnerabilities

- **Detection of anomalous access patterns:** Apigee Advanced API Security can be used to detect different types of anomalous behavior, such as sudden spikes in API traffic or requests coming from unexpected geographic locations. This can help identify potential attacks or security breaches in progress.
- **Rate limiting:** Apigee X can impose limits on the number of requests that can be made from a single IP address or account within a given time frame using a rate limiting policy. This can help prevent attackers from overwhelming your API with requests or using automated tools to brute-force authentication attempts.
- **Data encryption:** Apigee X can encrypt sensitive data in transit with Transport Layer Security (TLS). You can then use Advanced API Security's Risk Assessment feature to check that TLS and mTLS are configured on your API Proxy targets.

2 - Broken Authentication

Background

Broken Authentication, a serious security vulnerability, arises from flaws in authentication mechanisms, enabling attackers to bypass security measures and impersonate legitimate API clients. This can have devastating consequences for organizations, as attackers can gain unauthorized access to sensitive information, disrupt operations, or steal valuable assets.

Badly implemented authentication policies, including weak credential management and flawed session management, allow attackers to exploit implementation flaws and impersonate legitimate users or clients. Some key authentication principles to keep in mind are:

- Always authenticate both the user agent (the app) and the requesting user or client.
- Use delegated authentication and authorization patterns, avoiding direct password transmission within API requests.
- Validate access credential signatures and enforce defined expiry times for all credentials.
- Prevent brute force attacks with rate limiting functionality.

The Importance of Outside-In Design

An outside-in approach to API design prioritizes consumer use cases and security, especially when exposing backend systems to public networks. Traditionally, backend systems lack the robust authentication needed for public exposure. This is where Apigee, integrated with an identity and access management solution, provides strong protection. Key elements to consider include:

- **Security Design:** Leveraging Apigee features for effective authentication patterns.
- **Governance:** Ensuring consistent use of designed authentication patterns across all APIs.
- **Operational Security:** Detecting and responding to suspicious behavior and brute force attempts.

Security Design with Apigee

The goal of security design is to correctly implement authentication flows, often integrating with third-party identity tools, in order to ensure consistency in the authentication, authorization, and data protection approach across all of your APIs, and thereby reduce risk. This critical phase begins with providing for the appropriate delegated authentication flows, which may vary depending on the type of the consuming application. Collaboration with your identity team is crucial for defining integration patterns with the identity solutions you use for various constituencies that access your APIs - internal, external, partner, or public users..

Using Open Standards

OpenID Connect and OAuth RFCs offer a wide array of delegated authentication and authorization flows. However, the complexity of these standards contributes to broken authentication being a prevalent OWASP API threat. Apigee provides resources like example implementations to aid in understanding and correctly implementing OAuth flows.

Apigee: Native security features for mitigation

Feature	Description
Authentication	Apigee offers specific capabilities to address identity and authentication concerns. Apigee can: • Act as a token dispensary, generating access tokens and refresh tokens via the various OAuth2.0 grant types, and including custom grant types. Apigee can also verify such tokens, with very high performance. • Verify or decode identity or access tokens in JWT format as issued by OpenID Connect identity providers, such as Entra ID, Okta, Ping Identity, and more. • Act as an IETF RFC 8693-compliant Token Exchange endpoint, allowing you to support multiple distinct third-party identity providers in your API program. • Generate or verify JWS, or encrypted JWT. • Generate and validate SAML assertions. • VerifyIAM policy, for IAM-based authentication and authorization
OAuth 2.0 and OpenID Connect	Apigee X uses industry-standard protocols for secure authorization and authentication. OAuth 2.0 enables delegated access, allowing users to grant third-party applications access to their resources without sharing their credentials. OpenID Connect extends OAuth 2.0 by adding an identity layer, enabling single sign-on (SSO) across multiple applications and services. These protocols provide a comprehensive framework for protecting user identities and managing API access.
API key verification	For simple authentication, you can configure Apigee X to require API keys for authentication, providing a layer of protection against unauthorized access. API keys are unique identifiers that are associated with specific users or applications. When using API keys, Apigee ensures that only authorized entities can access the API. You can compose API Key verification with JWT verification to provide stronger controls for access tokens issued by third-party

	identity providers.
JWT (JSON Web Token) Support	Apigee X can validate and manage JSON Web Tokens (JWTs). JWTs are a compact and self-contained way to securely transmit information between parties. They are signed using a cryptographic algorithm, ensuring that the information has not been tampered with. Or, a JWT can be encrypted, ensuring the encapsulated information is private. Apigee's support for JWTs allows developers to use JWTs for authentication and authorization, or simple data protection, simplifying integration with third-party services and enhancing the security of API interactions.

Advanced API Security: Additional features for mitigation

Feature	Description
Threat protection	Advanced API Security offers threat protection capabilities to safeguard APIs from a range of attacks. It employs sophisticated algorithms to detect suspicious traffic patterns, including those indicative of brute force attacks, OAuth Abuser, or scraping. This comprehensive protection helps prevent unauthorized access, data breaches, and other malicious activities.
Anomaly detection	Advanced API Security utilizes advanced anomaly detection techniques to identify unusual behavior that may indicate unauthorized access attempts. By analyzing API traffic patterns, it can detect deviations from established norms, and issue the proper alerts. This enables organizations to promptly identify and respond to potential threats before they escalate into major security incidents.
Risk assessment	Advanced API Security continuously assesses the security posture of API proxies and traffic. It identifies misconfigurations that could cause vulnerabilities and weaknesses in authentication mechanisms, authorization policies, and data handling practices. This customizable risk assessment process provides actionable recommendations to improve the overall security posture of APIs, helping organizations prioritize and address critical security issues.

3 - Broken Object Property Level Authorization (BOPLA)

Background

Broken Object Property Level Authorization (BOPLA) is a critical security vulnerability that arises when an API fails to effectively enforce access control measures on individual properties or attributes within a data object. This vulnerability allows attackers to view, modify, or delete sensitive properties that they should not have access to, even if they have legitimate access to the data object itself.

BOPLA attacks often occur due to insufficient validation and authorization checks on API calls. When an API fails to properly validate the permissions of a user before allowing them to access specific properties within a data object, attackers can exploit this vulnerability to gain unauthorized access to sensitive information. For example, consider a social media platform that allows users to update their profile information, including their name, email address and membership level. If the API responsible for handling profile views only does not enforce granular access controls, an attacker could potentially modify their membership level and access higher-tier privileges without being eligible for that access.

To address BOPLA vulnerabilities, API developers must implement access control mechanisms that validate user permissions for each property or attribute within a data object. This can be achieved by utilizing fine-grained authorization policies that define the specific operations that users are allowed to perform on each property.

Additionally, APIs should employ input validation techniques to ensure that attackers cannot manipulate request parameters to bypass authorization checks. By implementing these security measures, organizations can mitigate the risk of BOPLA attacks and protect sensitive data from unauthorized access and modification.

Apigee: Native security features for mitigation

Feature	Description
Data masking	Users can use Data Classification services with Apigee to allow for the masking of sensitive data in API responses. This prevents unauthorized exposure of specific object properties. Masking can be applied to any field or property within an API response. This ensures that sensitive data, such as personally identifiable information (PII) or financial information, is not inadvertently disclosed to unauthorized users.
OpenAPI specification	By defining a clear OAS, Apigee can enforce data validation and

validation	prevent unauthorized modification of object properties. For example, if an API only exposes the user's name and email address, Apigee can block attempts to access or modify other properties, such as the user's password or credit card number. The OAS is a machine-readable definition of an API's structure and behavior. It describes the API's operations, parameters, data types, and security requirements. By validating incoming API requests against the OAS, Apigee can ensure that the requests are properly formatted and that the data provided is valid—and can do the same for responses, too. This prevents unauthorized users from modifying or manipulating object properties in a way that could compromise the security or integrity of the API.

Advanced API Security: Additional features for mitigation

Feature	Description
Proactive protection	Advanced API Security automatically checks proxy definitions for security policies that, if absent, could leave APIs susceptible to Broken Object Property Level Authorization (BOPLA) attacks. This helps the security team proactively address vulnerabilities and prevent exploitation. Apigee Advanced API Security utilizes advanced threat detection techniques to identify and act on BOPLA risks. This proactive approach helps detect and act on data breaches and unauthorized access to sensitive object properties, ensuring the confidentiality and integrity of your data.
Traffic analysis and alerting	Advanced API Security provides detailed analytics and reporting, continuously analyzing API traffic to identify unusual activity that might indicate a BOPLA attack, using ML-based Anomaly detection models. By recognizing deviations from normal behavior, it can alert security teams to quickly identify malicious API access attempts.

4 - Unrestricted Resource Consumption

Background

Unrestricted Resource Consumption (URC) vulnerabilities pose a significant threat to the security and reliability of API-driven applications. These vulnerabilities allow attackers to bypass the intended resource limits of an API, leading to excessive consumption of critical resources such as CPU, memory, or bandwidth. This can have severe consequences for both the availability of the API and the overall operational costs of the organization.

One of the primary risks associated with URC vulnerabilities is the potential for denial-of-service (DoS) attacks. By flooding the API with excessive requests, attackers can overwhelm its resources and render it unavailable to legitimate users. This can result in significant downtime and disruption of critical business processes that rely on the API. For example, an e-commerce website that experiences a DoS attack due to URC vulnerability may be unable to process orders or allow customers to access their accounts, leading to lost revenue and reputational damage.

In a less acute but still important form of unrestricted use, legitimate users of the API may inadvertently overuse an API, which can increase latency or reduce performance for other legitimate users of the API. Less a security risk, and more of an experience pitfall, this is still an important consideration in API design.

Another consequence of URC vulnerabilities is the financial burden they can impose on organizations. Excessive resource consumption can lead to increased cloud computing costs, as providers typically charge based on the amount of resources utilized. This is especially relevant for API invocations that can be relatively expensive, such as those that support generative AI models. Overuse of an API can put a strain on the IT budget and divert resources that could otherwise be allocated to strategic initiatives. Additionally, organizations may need to invest in additional infrastructure and security measures to mitigate the impact of URC vulnerabilities, further increasing their expenses.

To address URC vulnerabilities, organizations should implement a combination of technical and organizational measures. These include resource quotas and throttling mechanisms; rate limiting; usage monitoring; security patching; and educating developers and users about URC vulnerabilities and best practices for responsible API consumption.

By proactively addressing URC vulnerabilities, organizations can safeguard the availability, security, and cost-effectiveness of their API-driven systems, ensuring they remain resilient in the face of potential attacks.

Apigee: Native security features for mitigation

Feature	Description
Quota management	Apigee's quota management feature enables you to define rate limits and usage quotas for your APIs. This allows you to control the number of requests that can be made to your API within a specified time window. By setting quotas, you can prevent excessive or

	abusive access to your API that could lead to resource exhaustion or performance degradation. Apigee is flexible enough to allow rate limits that vary depending on the API consumer, or the type of application. Apigee can also enforce dynamic quotas, which can vary depending on the current load of the system, the reputation of the caller, the current day of the week, and many other factors.
Spike Arrest policy	The spike arrest feature in Apigee protects your APIs against sudden spikes in traffic. When a spike in traffic is detected, Apigee temporarily throttles requests to your API to prevent it from being overwhelmed.
Caching	Apigee provides a caching mechanism that can significantly improve the performance of your APIs. Frequently accessed data can be cached in memory, reducing the load on your backend services. This can result in faster response times for your API users and reduced infrastructure costs. You can even implement semantic caching in APIs that front Large Language Models (LLMs). API calls to LLMs can be significantly more expensive in terma of computer resource and financial cost, so intelligent rate limiting in this scenario is essential.

Advanced API Security: Additional features for mitigation

Feature	Description
Threat protection	Advanced API Security proactively detects malicious traffic patterns that could indicate a distributed denial of service (DoS) attack or attempts to exploit unrestricted resource consumption vulnerabilities. It employs sophisticated algorithms and machine learning techniques to analyze API traffic in real-time, identifying anomalies and suspicious patterns that may indicate malicious intent. Upon detection, Security Actions can be used to promptly block such traffic, safeguarding your APIs and preventing disruption to your services.

Anomaly detection	By continuously analyzing API traffic patterns and resource usage, Advanced API Security can identify unusual activity that may signal a potential DoS attack or resource exhaustion. It monitors key metrics such as API call volume, response times, and many other attributes to detect deviations from normal behavior patterns. When anomalies are detected, Advanced API Security generates alerts and provides detailed insights into the suspicious activity, enabling security teams to promptly investigate and take appropriate action.
Risk assessment	Advanced API Security continuously evaluates API proxies and traffic to identify misconfigurations that could lead to unrestricted resource consumption—for example, detecting that no Spike Arrest policies is defined on incoming requests. It performs comprehensive security scans to detect potential issues and based on its findings, Advanced API Security provides actionable recommendations to improve your security posture, such as tightening access controls and enforcing rate limits.
Prevention of resource exhaustion	Advanced API Security helps prevent resource exhaustion attacks by analyzing resource usage patterns and identifying potential issues. For example, if an API endpoint is experiencing a high volume of requests that could potentially exhaust available resources, users can then take preventative measures such as adjusting rate limits or optimizing resource allocation. By implementing these measures, organizations can ensure that their APIs have sufficient resources to handle normal traffic loads and are not vulnerable to resource exhaustion attacks.

5 - Broken Function Level Authorization (BFLA)

Background

Broken Function Level Authorization (BFLA) is a severe security vulnerability that undermines the integrity and security of web applications and services. It arises when an API fails to enforce appropriate access control mechanisms for specific functions or operations within its system. This oversight creates a window of opportunity for malicious actors to bypass intended authorization boundaries, enabling them to execute functions and perform actions that they should not have the privilege to.

At its core, BFLA stems from inadequate input validation and authorization checks within the API's code. When a request is made to the API, the API implementation should validate the credentials presented with the call, and ensure that those credentials convey the necessary permissions to execute the

requested function. In cases of BFLA, this validation process is either absent or insufficient, allowing unauthorized users to manipulate input parameters and gain access to functions that are beyond their authorized scope.

The consequences of BFLA can be severe, ranging from unauthorized data modification to privilege escalation attacks. By exploiting this vulnerability, attackers can gain access to sensitive information, such as customer records, financial data, or intellectual property. They can also modify critical data, such as product prices or inventory levels, to disrupt operations or commit fraud. Furthermore, attackers can elevate their privileges to higher levels within the system, potentially gaining administrative access and compromising the entire application or service.

To mitigate the risk of BFLA, organizations must implement stringent access control mechanisms within their APIs. This includes implementing fine-grained authorization checks for each function, ensuring that each function is only accessible to authorized users. Additionally, organizations should validate input parameters thoroughly, rejecting any requests that contain invalid or unexpected values. Furthermore, organizations should employ strong encryption techniques to protect sensitive data in transit and at rest, preventing attackers from accessing or modifying this data even if they manage to exploit BFLA.

In addition to implementing technical safeguards, organizations should also establish and maintain a comprehensive security program that includes regular security audits and penetration testing. This will help to identify and address any potential vulnerabilities that could lead to BFLA, as well as other security threats. By taking these steps, organizations can significantly reduce the risk of BFLA and protect their web applications and services from unauthorized access and compromise.

However, it is important to note that addressing BFLA requires a multi-faceted approach. Organizations should also focus on educating developers and API consumers about the importance of proper access control and input validation. By raising awareness and fostering a culture of security, organizations can further strengthen their defenses against BFLA and other API-related vulnerabilities.

Apigee: Native security features for mitigation

Feature	Description
API products	In Apigee, you create API products to bundle your APIs and make them available to app developers for consumption. Specifically, an API product bundles together one or more operations. An operation specifies an API proxy and resource paths that can be accessed on that proxy. An operation can also limit access by HTTP methods and by quota. API products are the central mechanism for access control to your APIs. By defining one or more API products in a developer app, you can restrict access to proxies and bundles of operations with an API key.

Quota management and Spike Arrest	Apigee's quota management and spike arrest features can indirectly help mitigate BFLA exploits by limiting the overall impact of potential attacks. By setting API usage quotas and implementing spike arrest mechanisms, organizations can control the volume and rate of API requests. This can help prevent sudden spikes in traffic and reduce the potential damage caused by unauthorized function execution.
OpenAPI Specification (OAS) validation	A well-defined OpenAPI Specification (OAS) can play a crucial role in enforcing function-level authorization. The OAS document describes the structure and behavior of an API, including the allowed operations for each endpoint. By validating API requests against the OAS, organizations can ensure that only authorized functions are being executed. This helps prevent unauthorized access to sensitive data and functionality.
API keys and OAuth	Apigee supports the use of API keys and OAuth for API authentication. Apigee can check not only that an OAuth token is valid, but also that it is correctly scoped for the requested function. These mechanisms are critical in helping ensure that only authorized clients can access specific functions within APIs, reducing the risk of unauthorized access. Apigee can enforce limited lifetime on tokens, shrinking any window of vulnerability, if a token should be inadvertently leaked. Apigee can apply additional checks, enforcing that OAuth tokens are usable only from particular IP addresses or from particular TLS sessions, for added protection against token hijack or leakage.
Request validation	Apigee provides built-in request validation capabilities that can help detect and block malicious requests. This can help prevent attackers from exploiting vulnerabilities in API endpoints.

Advanced API Security: Additional features to mitigate BFLA

Feature	Description
Risk assessment	Apigee Advanced API Security continuously evaluates API proxies and traffic, identifying misconfigurations that could lead to Broken

	Function Level Authorization (BFLA). It provides actionable recommendations to improve your security posture and put missing security policies in place, like access control mechanisms and Apigee authorization policies. This risk assessment capability is critical because BFLA attacks often target misconfigurations or vulnerabilities in API proxies, which can allow attackers to bypass security controls and access unauthorized functions. Advanced API Security's risk assessment helps organizations identify and address these weaknesses before they can be exploited. Advanced API Security allows you to create security profiles that define the expected configuration of your APIs. These profiles include information such as the expected encryption level and the allowed authentication schemes.
Threat protection and anomaly detection	Advanced API Security can detect suspicious traffic patterns that may signal an unauthorized function execution. This threat protection capability is essential for preventing BFLA attacks, as it can identify and block malicious traffic before it can reach your APIs. By setting up anomaly detection algorithms and monitoring function usage patterns, Advanced API Security can detect suspicious behavior and trigger alerts for further investigation.

6 - Unrestricted Access to Sensitive Business Flows

Background

Unrestricted Access to Sensitive Business Flows is a critical API vulnerability. It occurs when APIs expose sensitive business processes or workflows without adequate protection. This allows unauthorized individuals to manipulate these processes, potentially leading to severe consequences for the organization. This can happen due to improper API design, lack of proper security measures, or vulnerabilities in the underlying infrastructure.

The impact of this vulnerability can be significant. Attackers can exploit exposed business flows to manipulate critical operations, such as using an inventory API to artificially inflate or deplete stock, disrupting operations and impacting revenue. Attackers might also exploit vulnerabilities to perform

unauthorized actions, like bypassing payment verification in an e-commerce application to obtain goods or services without paying. Furthermore, attackers can exploit vulnerabilities in APIs to extract sensitive data, such as customer records and financial information, leading to identity theft and fraud. They can also manipulate business workflows through exposed APIs to perform unauthorized transactions, such as transferring funds or making purchases, resulting in financial losses and reputational damage to the organization. Additionally, attackers can disrupt business operations by manipulating or sabotaging critical business processes exposed through APIs. This can lead to downtime, lost productivity, and customer dissatisfaction.

To mitigate this risk, organizations should conduct thorough risk assessments to identify and assess the sensitivity of all business processes exposed through APIs. They should also implement comprehensive access controls to ensure only authorized users and systems can access sensitive API endpoints. Employing rate limiting and throttling can prevent abuse by limiting the number of requests to sensitive functions. Additionally, organizations should monitor API activity to track usage patterns and identify anomalies that could indicate malicious activity. Regular API security testing should be conducted to identify and fix vulnerabilities before attackers can exploit them. Finally, organizations must prioritize security in API design, considering potential misuse scenarios throughout the development lifecycle.

By understanding the risks of Unrestricted Access to Sensitive Business Flows and taking proactive measures to mitigate them, organizations can strengthen their API security and protect their critical business operations.

Apigee: Native security features for mitigation

Feature	Description
Role-based access control (RBAC)	Apigee implements a comprehensive RBAC system that enables organizations to define granular access controls for their APIs. Administrators can assign roles to users and groups, which specify the operations that each entity is authorized to perform. For example, an "editor" role might be granted permission to create and modify APIs, while a "viewer" role might only be granted permission to view API documentation. By implementing RBAC, organizations can ensure that only authorized individuals have access to sensitive business flows.
API key verification	Apigee requires API keys for authentication, which adds an additional layer of security to protect against unauthorized access. API keys can be generated and managed within the Apigee platform, and they can be associated with specific roles and permissions. When a client application makes a request to an API, it must include the API key in the request header. Apigee then validates the API key

	against the stored list of valid keys and grants or denies access accordingly. When using API keys, Apigee helps organizations protect their APIs from unauthorized access and abuse.
Threat protection	**Rate limiting:** Apigee can limit the number of requests that a client application can make to an API within a specified time period. This can help prevent denial-of-service (DoS) attacks. **Payload inspection:** Apigee can inspect the payloads of requests and responses for suspicious content, such as Structured Query Language (SQL) injection attacks or cross-site scripting (XSS) attacks.

Advanced API Security: Additional features for mitigation

Feature	Description
Risk assessment	Apigee Advanced API Security continuously evaluates API proxies and traffic, identifying misconfigurations that could expose sensitive business flows. It provides actionable recommendations to improve your security posture. While Apigee provides policies to enforce correct access to business logic, Advanced API Security (via the Risk Assessment feature) can identify when these policies are not in place, or if they change while deployed.
Traffic analysis and alerting	Advanced API Security can detect and flag unusual activity within business flows, such as sudden spikes in API traffic or changes in the types of API calls being made. This information can be used to identify potential attacks or security breaches. Once detected, users can choose how to react to these traffic patterns and block requests from specific IP addresses that are known to be malicious.

7 - Server-Side Request Forgery (SSRF)

Background

Server-Side Request Forgery (SSRF) is a critical security vulnerability that arises when an attacker can manipulate a server-side application to make HTTP requests to arbitrary locations. This typically occurs when an application fetches a resource from a user-supplied URL without proper validation or sanitization. Essentially, the attacker tricks the server into acting as a proxy, allowing them to access resources that should be inaccessible.

An attacker can exploit SSRF to gain access to internal resources that should not be exposed to the public internet, such as sensitive data like customer information, financial records, or intellectual property. They can also use SSRF to exfiltrate data from internal systems to an external location under their control, potentially stealing sensitive information or launching further attacks. Moreover, SSRF can enable attackers to perform unauthorized actions on behalf of the server application, including creating or deleting files, modifying data, or even executing arbitrary code. SSRF attacks can be launched from both inside and outside of an organization's network, targeting vulnerabilities in public-facing web applications, internal applications, and even cloud-based services.

To protect against SSRF attacks, organizations should implement input validation on all requests to external resources, including validating the URL, query parameters, and request body. Using a web application firewall (WAF) can help block malicious requests. Restricting access to internal resources to only the necessary users and applications is also crucial. Finally, organizations should monitor for suspicious activity, such as requests to unusual URLs or requests that are made from unexpected locations.

By understanding SSRF risks and taking proactive measures to mitigate them, organizations can strengthen their API security and protect their systems from potential attacks.

Apigee: Native security features for mitigation

Feature	Description
Input validation	Apigee policies offer a wide range input validation capabilities, safeguarding APIs from malicious input injection attempts. They prevent attackers from exploiting vulnerabilities by validating and sanitizing user input. This is especially crucial for APIs that accept user-provided input, like search queries or registration forms. Apigee policies can perform various input validation checks, including: By implementing input validation, Apigee effectively thwarts attackers' attempts to exploit API weaknesses, contributing to a more secure API environment.

Additional mitigation features in Advanced API Security

Feature	Description
Risk assessment	Apigee Advanced API Security continuously evaluates API proxies and traffic, identifying potential SSRF vulnerabilities that might raise due to policies not configured as part of the API proxies, such as various Input Validation policies . You can also define security policies for your APIs that specify the allowed operations, resources, and users. Apigee will enforce these policies and block any requests that violate them. Advanced API Security will continuously monitor these policies to validate they are configured correctly at runtime. The risk assessment process is automated and runs continuously, so you can be confident that your API proxies are always being monitored for vulnerabilities. Actionable recommendations are provided to address identified vulnerabilities, such as updating API proxy configurations or implementing additional security measures.
Threat protection	Advanced API Security uses anomaly detection to identify unusual requests that may indicate SSRF attacks. It analyzes API traffic patterns and identifies requests that deviate from the normal pattern. For example, it can look for requests coming from unexpected sources, requests that access multiple resources in a short period of time. These requests can then be investigated further to determine if they are malicious. Upon users setting Actions, Advanced API Security can block SSRF attacks in real-time, so you can be confident that your API proxies are protected from these attacks.
SIEM and WAF integrations	Advanced API Security can integrate with other security tools, such as security information and event management (SIEM) systems, to provide additional visibility into SSRF attacks. It also integrates with Cloud Armor, Google Cloud's Web Application Firewall (WAF), to

	help block malicious requests, including blocking traffic from certain IPs, geos, and custom parameters.

8 - Security Misconfiguration

Background

Security Misconfiguration is a broad category encompassing vulnerabilities that stem from improper security setups or configurations within an API ecosystem. This can include issues like insecure default configurations, where APIs are shipped with settings that may not prioritize security, such as allowing access from any IP address. Incomplete configurations are another common problem, where administrators may unintentionally overlook crucial security settings. For example, an API might have HTTPS enabled but lack Secure Sockets Layer (SSL) certificate validation, leaving it susceptible to man-in-the-middle attacks. APIs may also include unnecessary features, like debug modes, that are not essential for most users and can serve as potential attack vectors if not properly disabled. Misconfigured HTTP headers can also lead to vulnerabilities, potentially enabling cross-site scripting (XSS) or clickjacking attacks. Similarly, overly permissive Cross-Origin Resource Sharing (CORS) configurations can facilitate cross-origin attacks, allowing unauthorized access to API resources. Finally, verbose error messages can inadvertently expose sensitive details about the API or its infrastructure, potentially aiding attackers in identifying and exploiting vulnerabilities.

To mitigate the risks associated with security misconfiguration, organizations should establish comprehensive security configuration practices, including thorough reviews of default settings, rigorous testing of configurations, and disabling or removing any unnecessary features. Regular security audits and vulnerability scans can help identify and address misconfigurations proactively. Additionally, organizations should prioritize secure coding practices and implement comprehensive security training programs to raise awareness among developers and administrators about the importance of proper security configuration.

Through the integration of Apigee's core security measures and the advanced functionalities offered by Advanced API Security, organizations can drastically minimize the likelihood of security misconfigurations and safeguard their APIs from potential attacks. This can help organizations to improve their overall security posture and reduce the risk of data breaches.

Apigee: Native security features for mitigation

Feature	Description
Security policies	Apigee offers a comprehensive set of policies that can be leveraged to enforce security best practices across APIs. These

	policies allow organizations to define and apply security measures. Rate limiting policies help mitigate the risk of denial-of-service attacks by limiting the number of requests that can be made to an API within a specified timeframe. Implementing strict CORS policies is another important step. CORS (Cross-Origin Resource Sharing) is a security mechanism that allows browsers to restrict access to resources from different origins. By implementing strict CORS policies, organizations can prevent unauthorized access to their APIs. Having these policies in place does not mean that they can't be misconfigured, however. See below for how Advanced API Security can help with this.
Access control	Apigee enforces strong authentication to verify user identities and prevent unauthorized API access. This protects sensitive data by ensuring that only legitimate users can interact with APIs. Furthermore, Apigee implements granular authorization policies, including API key verification, to control what actions authenticated users are permitted to perform. This restricts users to only the specific API functions necessary for their role, further enhancing security.
Monitoring and logging	Comprehensive monitoring and logging capabilities are essential for identifying potential security misconfigurations. Apigee provides detailed monitoring and logging features that allow organizations to track API activity and identify any suspicious or anomalous behavior, including real-time monitoring dashboards, historical logs, and alerting mechanisms. These capabilities enable security teams to detect and respond to security incidents promptly, minimizing the impact of potential breaches.

Advanced API Security: Additional features for mitigation

Feature	Description

Security misconfiguration evaluation	Advanced API Security can help identify security misconfigurations in your API proxies and traffic by continuously evaluating them against industry best practices and security standards. Some specific examples of misconfigurations that AAS can detect include: • **Incorrect or missing authentication and authorization settings:** Advanced API Security can identify API proxies that are not properly configured for authentication and authorization, such as those that allow public access to sensitive data. • **Insecure API endpoints:** Advanced API Security can identify API endpoints that are exposed to the internet without proper protection, such as those that do not use SSL/Transport Layer Security (TLS) encryption. • **Insufficient logging and monitoring:** Advanced API Security can identify API proxies that do not have adequate logging and monitoring in place to detect and respond to security incidents.
Security insights	Advanced API Security provides detailed insights into API security posture, helping identify potential areas of weakness. For example, AAS can provide insights into the number of APIs that are missing authentication or authorization checks, or the number of APIs that are using outdated security libraries.
Real-time protection	Advanced API Security provides real-time protection against attacks, helping to prevent exploits of security misconfigurations. For example, it can alert on attacks that attempt to exploit APIs and security response teams can use Security Actions to block bad clients.

9 - Improper Inventory Management

Background

Improper inventory management of APIs has become a critical issue affecting many organizations today. This lack of visibility and control over APIs within an organization's environment can lead to several challenges and potential risks.

One significant aspect of improper inventory management is the presence of undocumented or "shadow" APIs. These APIs exist within an organization but are not officially documented or sanctioned. They are often created by developers without following proper protocols or standards. Shadow APIs pose significant security risks as they are not subject to the same level of scrutiny and oversight as documented APIs. Attackers can exploit these undocumented APIs to gain unauthorized access to sensitive data or launch malicious attacks.

Outdated versions of APIs are another concern resulting from improper inventory management. When APIs are not updated regularly, they become vulnerable to security exploits and stability issues. These outdated APIs may also be incompatible with newer versions of software or libraries, leading to operational problems and potential disruptions.

APIs with unknown dependencies also contribute to the challenges of improper inventory management. When APIs have dependencies on other components that are not known or documented, it becomes difficult to troubleshoot issues and manage the API effectively. These unknown dependencies can introduce security vulnerabilities if the dependent components are themselves vulnerable to attack.

The consequences of improper inventory management can be severe and far-reaching. Undocumented and outdated APIs can provide attackers with entry points into an organization's systems, leading to security breaches, data theft, and denial-of-service attacks. Organizations subject to regulatory compliance requirements, such as General Data Protection Regulation (GDPR) or Health Insurance Portability and Accountability Act of 1996 (HIPAA), may be at risk of violating these regulations if they do not have proper inventory management of their APIs.

Moreover, improper inventory management can result in operational inefficiencies and bottlenecks. Managing and troubleshooting APIs that are not properly documented or understood can be time-consuming and resource-intensive. This can hinder an organization's ability to innovate and respond quickly to changing business needs.

To address the challenges of improper inventory management, organizations must implement effective API governance and management practices. This includes establishing a centralized inventory of all APIs, including both documented and shadow APIs. Regular audits should be conducted to identify outdated versions, unknown dependencies, and other potential issues. Additionally, organizations should ensure that all APIs are properly documented and adhere to standard protocols and best practices.

Apigee: Native security features for mitigation

Feature	Description
Versioning and lifecycle management	Apigee supports revisions of API proxies as a way to manage updates to an API proxy configuration as you iterate. With proxy revisions, you can deploy an API proxy into a production environment, while continuing to create new revisions of that API proxy in a test environment and promoting it to prod when you're ready. Apigee also offers lifecycle management capabilities, enabling organizations to define and enforce policies for each stage of the API lifecycle, such as development, testing, and production. By managing API versions and lifecycles effectively, organizations can maintain API stability and minimize the risk of API downtime.
Centralized inventory and management	Apigee offers a centralized platform called API Hub for managing the entire API lifecycle, from design and development to deployment and retirement. API Hub provides a single point of control for managing all APIs, helping organizations maintain an accurate inventory of their APIs. It streamlines the API development process, enabling teams to collaborate and track the progress of APIs throughout their lifecycle. API Hub also facilitates reusability, making it easier for developers to reuse existing APIs.

Advanced API Security: Additional features for mitigation

Feature	Description
Shadow API discovery	Advanced API Security's shadow API discovery capability is crucial for organizations seeking a comprehensive view of their API landscape. By analyzing API traffic flowing through Google Cloud load balancers, Advanced API Security can identify undocumented,

	unauthorized, or forgotten APIs that might be vulnerable to exploitation. It continuously compares the current API landscape with previous snapshots to identify any discrepancies, ensuring that organizations are always aware of the latest changes to their API ecosystem. This visibility is critical for ensuring a secure API environment and proper management and protection of all APIs.
Risk assessment	Advanced API Security's risk assessment capabilities assist organizations in identifying potential vulnerabilities of their APIs. Through continuous evaluation of API proxies, Advanced API Security can detect issues such as APIs with known security vulnerabilities. This information is essential for organizations aiming to enhance their security posture and mitigate the risk of API-based attacks.
Alerting and incident response	Advanced API Security provides alerting to promptly notify organizations of potential API security threats. When suspicious activity or known attack patterns are detected, Advanced API Security triggers real-time alerts, allowing organizations to respond swiftly and effectively. This proactive approach helps minimize the impact of API-based attacks and ensures rapid containment of security incidents.
Continuous monitoring	Advanced API Security's continuous monitoring capabilities are essential for maintaining an up-to-date inventory of an organization's APIs. By monitoring API traffic and configurations, Advanced API Security can detect changes in the API landscape, such as the introduction of new APIs or modifications to existing ones. This real-time visibility allows organizations to proactively address potential security risks and ensure that their APIs are constantly protected.

10 - Unsafe Consumption of APIs

Background

Unsafe Consumption of APIs highlights the risks associated with integrating and interacting with external or third-party APIs without sufficient security measures. This vulnerability arises when your API consumes data or services from another API that might be compromised, poorly designed, or even malicious. This can lead to various attacks, including data breaches where attackers exploit vulnerabilities in the third-party API to access sensitive data within your system, such as customer information, financial data, or intellectual property. This data can be used for identity theft, fraud, or other malicious purposes. Injection attacks are another risk, where malicious data injected through the third-party API can compromise your API and backend systems, potentially allowing attackers to execute arbitrary code, gain unauthorized access to sensitive data, or disrupt your API's functionality.

Additionally, if the third-party API becomes unavailable or overloaded, it can lead to Denial of Service (DoS), preventing legitimate users from accessing your API and potentially resulting in lost revenue, reputational damage, and customer dissatisfaction. In severe cases, exploitation of vulnerabilities in the third-party API can even lead to a system takeover, giving attackers unauthorized access and control of your systems, enabling them to steal data, disrupt operations, or launch further attacks on your infrastructure.

To mitigate these risks, organizations should conduct a thorough security review of the third-party API before integrating it, assessing its security features, development practices, and track record. Validating all input data received from the third-party API is crucial to prevent injection attacks, including checks for malicious code, SQL injection, and other types of attacks. Implementing rate limiting mechanisms can prevent the third-party API from overloading your system with requests, mitigating the impact of potential DoS attacks. Encrypting all data transmitted between your API and the third-party API helps prevent eavesdropping and man-in-the-middle attacks. Organizations should also implement comprehensive monitoring and logging mechanisms to detect and investigate any suspicious activity related to the third-party API, enabling prompt identification and response to breaches or attacks. Finally, regularly updating your API and the third-party API ensures that you are using the latest security patches and fixes.

By following these best practices, organizations can significantly reduce the risks associated with unsafe consumption of APIs and protect their systems from various attacks.

Apigee: Native security features for mitigation

Feature	Description
API gateway	The gateway acts as a central hub for managing and securing all interactions with external APIs. It provides a single point of entry and control for API traffic, streamlining API management and reducing complexity. Apigee enforces security policies, handles authentication and authorization, and applies rate limiting to

	prevent abuse. It ensures that only authorized users with valid credentials can access the APIs and prevents unauthorized access, data breaches, and other threats.
Access control policies	Apigee offers a comprehensive set of policies to control access and usage of third-party APIs. These policies include OAuth 2.0, API key verification, and quota management. The OAuth 2.0 policy enables secure authentication and authorization by allowing users to access APIs using access tokens issued by an authorization server. The API key verification policy ensures that only authorized clients with valid API keys can access the APIs. The quota management policy helps prevent API overuse by setting limits on the number of requests a client can make within a specified time period. These policies work together to ensure that APIs are consumed securely and responsibly.
Validation	Message validation helps reduce the risk of injection attacks by validating incoming data against predefined schemas. Apigee supports message validation for XML and SOAP payloads as well as validation of restful API requests against an OpenAPI Specification. Apigee's threat protection policies can also validate data against predefined data format rules and limits. This is supported for both XML and JSON payloads.
Monitoring and logging	Apigee offers comprehensive monitoring and logging capabilities to track interactions with external APIs. It provides real-time visibility into API traffic, response times, errors, and other key metrics. Monitoring and logging help identify suspicious activities, detect anomalies, and ensure compliance with security standards. All API requests and responses are logged, enabling forensic analysis and troubleshooting in case of issues. This information is crucial for maintaining the security and integrity of API consumption. These components collectively create a secure API consumption environment, protecting both internal systems and external APIs from unauthorized access, data breaches, and other threats. Apigee's foundation for secure API consumption empowers organizations to confidently integrate with external APIs, drive innovation, and unlock the full potential of API-driven architectures.

Advanced API Security: Additional features for mitigation

Feature	Description
Threat detection	Advanced API Security employs advanced machine learning algorithms to identify malicious traffic originating from third-party APIs. This proactive approach ensures protection against even unknown threats, providing a critical layer of defense.
Real-time visibility and threat response	Advanced API Security provides visibility into API consumption and risk. AAS integrates seamlessly with Google Cloud Monitoring, pushing both real-time threat detection data *and* risk assessment information for comprehensive insight. This means you not only see potential abuse as it happens, but also have continuous awareness of underlying vulnerabilities in your APIs and those you consume from third parties. This deep visibility, coupled with real-time alerts for suspicious activity and known attack patterns, empowers organizations to respond swiftly and effectively to potential API security threats. This proactive approach helps minimize the impact of API-based attacks, ensures rapid containment of security incidents, and facilitates informed decision-making about API usage.
Integration with 3P security tools	Advanced API Security can integrate with other security tools, such as security information and event management (SIEM) systems and threat intelligence platforms, to enhance threat detection and response capabilities. This integration allows organizations to leverage existing security investments and gain a holistic view of their API security posture.

Appendix

Additional Resources

Resource	Description
OWASP resources	
OWASP API Security Top 10 - 2023	The official OWASP page detailing the Top 10 API Security Risks for 2023. This is a primary source for understanding the risks in detail.
OWASP Cheat Sheet: REST Security	A concise overview of how to protect REST APIs.
Apigee resources	
Apigee overview	General overview of Apigee's capabilities
Apigee's security administration features	Details on Apigee's built-in features for administering security controls.
Advanced API Security *(add-on for Apigee)*	Details on the capabilities of Advanced API Security for Apigee.
Apigee documentation	Technical documentation for Apigee X and Apigee Hybrid.
Mandiant resources	
Mandiant overview	Overview of Mandiant's cybersecurity consulting capabilities, including API incident response services.
Mandiant Academy	Courses and certification programs for a range of cybersecurity topics.