A Practical Guide to *API Security*

argano

Table of Contents

Application Programming Interfaces, or APIs, are fast becoming the de facto tool for connecting businesses, eliminating silos within the enterprise, safely exposing data, and breaking down monolithic applications into maintainable and reusable services.

As the use of APIs continues to increase, so does the potential for vulnerabilities in your system if there are no security processes put in place. In the rush to realize the benefits of an API, it is tempting to sideline or disregard crucial processes. However, API security is an area that cannot be ignored.

Companies need a comprehensive strategy that establishes everything from defining ownership and roles to implementing appropriate tools and adhering to best practices. Creating secure APIs involves a multi-disciplinary approach, balancing the necessary measures to protect an organization's systems and data with leveraging the advantages APIs offer.

Ultimately, API security must be a top priority, which may seem challenging, but there are straightforward steps you can take to begin securing your APIs.

Chapter 1: The Rise of APIs and Security Risks

APIs are the connectivity and functionality mechanisms with which enterprises can enable digital transformation. Digital transformation refers to the process of using digital technologies to fundamentally change and improve the way businesses operate, deliver value to customers, and interact with stakeholders. The stark growth in the number of APIs indicates how much enterprises and developers value the technology as part of their transformation strategy.

As enterprises continue to add APIs into the mix, breaking down monolithic applications, accelerating communications with partners, and offering clients new and innovative services, they become increasingly dependent on a growing number of applications. Their size makes them no less important to the business than their larger, complex predecessors.

However, as API deployment continues to increase, proper API security is not as widely practiced as it should be.

Obstacles to Effective API Security:

- **Lack of awareness:** Some developers and organizations may not fully understand the importance of API security, or the potential risks associated with insecure APIs.
- **Time and resource constraints:** In today's fast-paced development environments, organizations may prioritize delivering features and functionality over security. This can lead to rushed development cycles that neglect thorough security practices.
- **Legacy systems:** Older systems may have APIs that were developed without considering modern security standards. Updating or securing these legacy APIs can be challenging and may require significant resources.

- **Complexity:** APIs can be complex and securing them requires a good understanding of various security measures. Some organizations may find it challenging to implement robust security measures due to the complexity involved.
- **Misconfigurations:** Improperly configured APIs can pose security risks. Organizations may overlook the importance of correctly configuring APIs, leading to vulnerabilities that attackers can exploit.

The lack of API security awareness is concerning, as the rise in APIs means there is a corresponding rise in security risks for enterprises.

APIs, acting as the data-rich links between different applications, expose multiple vulnerabilities that hackers can target and extend the attack surface of an enterprise.

Approaching API security as a priority

What is API security? Generally, API security refers to the "practices and solutions designed to prevent malicious attacks on, or misuse of, application program interfaces (API). It constitutes an integral part of API management and governance."

While APIs leverage web technology for application integration, it is erroneous to assume that safeguarding APIs can be achieved through the same practices and techniques applied to securing the Web. The risk profile of APIs is distinct and requires a unique security approach. Developers who neglect to prioritize security in the creation or utilization of APIs jeopardize both the data and the applications.

Enterprises must adopt a proactive approach towards API security and be prepared for worst-case scenarios. Ensuring the security of an API should be a top consideration in its architectural development. Security testing should commence early in the process, well before deployment, and persist throughout subsequent development stages.

Decisions regarding how specific requests are managed should also be made during API development. While security measures may initially be broad, they should progressively be narrowed down based on the specific needs of the enterprise.

How is API security managed?

There are many ways hackers are using APIs to gain access to enterprise systems. Some of the most common attack paths include parameter attacks (often via an SQL injection), identity attacks, and man-in-the-middle attacks.

To combat these attacks, the most widely used API security models employ identification and authorization measures implemented in balance with the use of tokens, API gateways, quotas, throttling, encryption, and signatures.

Some enterprises are opting to use in-house solutions for API security and are finding their efforts becoming mere reactionary measures to attacks rather than actively preventative ones. There may be no consensus between an enterprise's IT security team and the API development team on who is responsible for API security, which can cause the task to be delegated downward.

More enterprises are using API-security firms that have routinely updated threat databases and that offer a complete arsenal of identity and management tools. However, even with these tools, the protection level can fail to detect the most sophisticated attacks. While the security measures are robust, additional steps are needed to address the resulting security gaps that arise when APIs are deployed. In present day, machine learning-backed API security is already being deployed, that is to be further discussed.

Key Takeaway:

API security has not kept pace with the broad adoption of the technology. It is crucial that businesses take API security seriously. Regardless of the level of security measures required or how they are employed, technology leaders and teams must understand the risks APIs can pose, along with the benefits, and be prepared to lay down a foundation for secure API development and use.

Chapter 2: Common API Security Mistakes and Measures to Prevent Them

In the rush to adopt new technology that promised — and delivered — greater flexibility and agility to businesses, companies began building APIs. Only later did some discover that, without process and planning, API implementations can get quickly out of hand. Others are still learning this. Unfortunately, security often is one of the design process elements that can get sidelined in a company's early days of realizing the power APIs provide.

There is a distinct difference between merely building an API and designing and developing an API with the security features that allow it to address the needs of users reliably. Here are some common API security mistakes to be aware of and helpful best practices for avoiding them.

1. *No plan for API security*

Not prioritizing API security during development can be attributed to apprehension about tackling API security, lack of expertise on the subject, or reluctance to be held responsible for any security complications. Also, many times, organizations want to develop quickly, and API security is not seen as a priority to move a product to production. However, not having an action plan in place for worst-case scenarios could leave an enterprise severely compromised in the event of an attack.

Security should be a primary focus during the development of an API, not an afterthought. There are many different development tools available that can be used to ensure an API has the proper security features it needs.

2. *Leaving APIs exposed*

Implementing an API with no mechanisms in place for verifying who is trying to gain access and whether they have the appropriate authority exposes both the API and any connected digital assets to hackers and bots searching for vulnerabilities and will gain access to exposed APIs and mine for sensitive data.

Implement API security — even if it is minimal. This can include incorporating authorization mechanisms so that only specific IP addresses are allowed. Basic authentication can be used at the front-end, requiring user- names and passwords. You can take security a step further by instituting multi-factor authentication at the front end followed by some form of authorization check so that the appropriate users have access to certain types of information. There also should be a checkpoint at the back end of an API.

3. *Homegrown API security*

The risk with keeping API security in-house is that errors can occur, even if security was a priority during development. There can be loopholes in the code-API code that is poorly written, which could be easy to hack and is a security risk. Another factor to consider is the effort and time required by an enterprise's developers to ensure that security standards are updated frequently and are effective. This is likely to occur at the expense of other projects and result in technical debt.

Use third-party API security providers, such as Ping Identity or Okta, that keep up to date with the latest in API security for you, so you do not have to. The security standards used by their platforms are routinely updated and enhanced to handle new threats. Working with a reputable technology partner can help guide your API strategy.

4. *Not having an organizational governance policy*

API governance can be considered a form of API maintenance and is an essential part of API management. The governance policy should encompass documentation. If there is no documentation when APIs are initially set up, it can be challenging to go back to properly inventory them, particularly if an enterprise has many APIs. A governance policy is also necessary to track which APIs can be accessed by which parties, who owns the APIs, and who is responsible for maintaining them. Without a clear picture of its API environment, an enterprise can suffer from a lack of agility as it tries to function without knowing precisely what APIs they have, which ones are secured (if any) and who should have access to them.

Establish early on (preferably during the design and development processes) who owns the APIs and is responsible for their security. Documenting is a key piece of proactive protection and reactive mitigation of security risks, especially as an API evolves. Documenting the vast network of APIs in a complex organization may be a daunting task. Consider using a third-party API security and risk management provider, such as Traceable Defense AI, that provides continuously updating, automatic API documentation and tracking of API definition and usage.

Key Takeaway:

Require a minimal level of API security. The minimum level of API security would include these three basic security mechanisms. First, basic authentication can be used at the front-end, requiring usernames and passwords. This mechanism has, over time, become less and less recognized as a responsible and vigorous response. Second, incorporating Internet Protocol (IP) address authorization so only specific IP addresses may access the API. Third, authorization processes can segregate the originating request's data and information needs and allow appropriate access based on level access definitions.

API security should not be an afterthought. Avoiding breaches requires that API security be an integral part of planning and design. The few mitigation actions in this chapter are easy to implement, but best practices for API security should be an ingrained element of the end-to-end process.

Chapter 3: Best Practices and Benefits of API Security

The rush to realize the benefits of API-led integration frequently leads to API security being overlooked or shoved to the side. While organizations have the best intentions to revisit their API applications' security, setting this vital aspect of development aside for later can lead to severe consequences, especially for public-facing APIs.

API security does not need to be overly complicated, but it can seem overwhelming to many, given the advanced threats that occur in the industry.

Following best practices, API security will change from a daunting task to one that most organizations can accomplish.

1. *Develop a plan for API ownership*

Many times companies implement APIs without the critical step of assigning an owner. An API owner does more than simply maintain the application. An owner is responsible, among other things, for ensuring API security is considered and managed. Because an owner must take charge when there is a security incident that involves the API, it is important to make sure securing the application is part of its development.

When planning or designing your API, you need to ask:

- Who or what group owns the API(s)?
- Who is responsible for maintaining it or them, including updating documentation?
- Who reacts to an API security incident?

Ownership of an API is at a higher level than accountability or responsibility. An owner might even be a CTO. If it is a group, then it is typically the group leader or architect at the tip of the spear when there is an incident.

One of the benefits of identifying an API owner is that it provides clarity on who should be directing action during an incident and who is in charge of ensuring security is considered and implemented.

Without ownership, an incident can result in many different people and groups pointing fingers in many different directions. It is more likely that an API will be protected to the fullest from the start with ownership.

2. *Include API security in your planning or design phases*

API security can fall victim to the desire to implement a solution quickly. At other times, API security ends up an afterthought. In both cases, you can get seriously burned by leaving your APIs unprotected.

Although often thought of as technical requirements, API security requirements are business requirements. Since security is a business requirement, it is not optional. APIs are an essential part of your business results, so protecting your APIs is the same as a secure lock on a bank vault.

The surest way to prevent security from being sidelined is to incorporate it into the API's design and planning process. It can take some time to include security, but this is time well spent. Plus, including it in the design means you can choose the type of API security most appropriate to your application right from the start.

Depending on your environment and the application's purpose and availability, you may choose to implement only OAuth 2.0 and IP authorization. However, you may discover that your API needs greater protection and decide to include WAF, OAuth 2.0, IP authorization, and even Traceable or PingIntelligence for a public-facing, high-risk API. By including security thinking early, you can integrate the right security model into the development process instead of retrofitting it later.

By raising the issue of API security from the beginning, it will increase stakeholder awareness of the need and keep it on their radar. Project milestones can be set with the development of the security components included. Stories and epics can be created alongside the other app features. Plus, the entire team will be brought into security as part of the API's creation.

3. *Build-in API monitoring*

As much as possible, your security model should allow you to be proactive in protecting your API, not reactive. Building monitoring and alerting into the app promotes a preemptive approach. During the design phase, you should ask:

- What kind of metrics do we want to see?
- Do we have the visibility needed for our use cases?
- Do we have the ability to detect a breach?

As you consider your metrics, start with the ones that will give you the alerts needed to address issues quickly. As a baseline, your metrics should include:

- *Latency*
- *Request size*
- *Response size*
- *Geographic location*

Alerting on these metrics will keep you aware of performance and outliers without requiring someone to look at a screen all day. These metrics can also help identify problems when the API performs poorly, which can be like trying to find a needle in a haystack.

If you must detect more advanced attacks, you may need a more refined approach like machine learning or an AI engine with logging, tracing, and auditing support. A product like PingIntelligence or Traceable can also help find API security-related attacks by learning your API's behavior. Using an AI/ML software can help protect your APIs also allows you to track metrics and usage patterns.

Understanding typical performance and metric numbers will help you improve your API user experience. It also creates a viewport to detect breaches where you may not have had one before. Setting metric limits allows for alerting and can even kick off automated actions.

Metrics and alerting also free up time for your team. With well-defined metrics and the right tools, no one needs to spend time actively monitoring your APIs every day — the system can monitor itself and let you know when a problem or anomaly occurs. Tools like Traceable provide a great interface for monitoring and tracing the traffic on your APIs.

4. *Protect your API with something, rather than nothing*

This may not be necessarily considered a best practice, but when the alternative is to leave your APIs unprotected, something is better than nothing. That is because bots can and do scan the Web for open APIs, and they will find yours if it is left unsecured.

Even the most basic security actions will prevent more automated and brute-force attacks.

At a minimum, you should put API gateway security on your app, including:

- IP authorization with targeted use
- Basic authentication
- OAuth 2.0

Depending on your environment, whether your app is public or private, and your API's purpose, you may choose to go all the way, including OAuth 2.0, IP authorization, and operational tools such as PingIntelligence, Traceable Defense AI, and/or WAF. Do as much as is warranted by the API, its intended use, and its risk association.

By doing even the minimum, you will protect your APIs from common attacks, including bots probing the internet for low-hanging fruit. With an even more secure API posture, you can eliminate many sophisticated and known attacks.

5. *Maximize your API security with a layered approach*

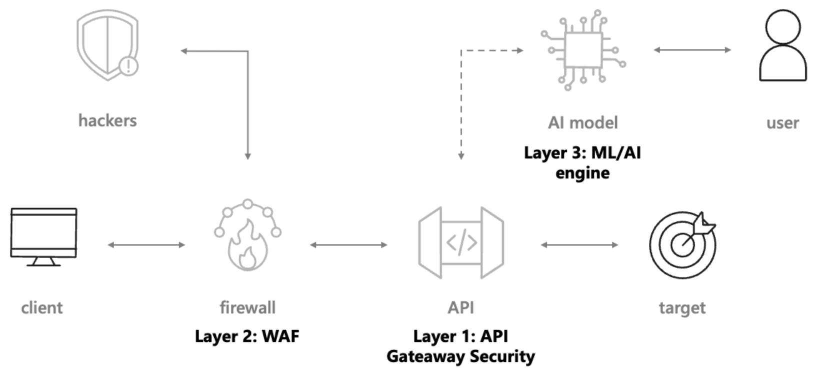
For peace of mind, using a layered API security approach is your best bet for protecting your APIs.

Having secure APIs requires a multi-layered approach. Knowing how these layers interact with one another to provide protection is essential for adequate API security.

The layers of API security include:

- **API Gateway Security:** Great for rate limiting and access control
- **Web Application Firewall (WAF):** Ideal for OWASP top 10 protection
- **Machine Language / Artificial Intelligence (ML/AI) engine:** Engines like PingIntelligence and Traceable monitor your APIs' behavior and protect against advanced, authenticated attacks that can fly under the radar.

This simplified illustration gives you an idea of what layered API security looks like, conceptually:



Employing a layered approach gives you the best protection with the most flexibility to address attacks. Using an ML/AI engine is like fighting fire with fire — hackers are using automated, AI-driven attacks, so your protection should too. API security practitioners must use API gateway security to protect against standard attacks, WAFs to protect against OWASP top ten attacks, and ML/AI engines to protect against advanced attacks using sophisticated methods.

Using ML/AI, you can gain deep visibility into your API's usage, operation, and performance. An ML/AI model will learn your API's normal operation and alert you if anything deviates from those norms. That data can also be mined for actionable insights on how to improve your application. ML/AI models offer a greater level of automated threat protection. There is no need to update security protocols or algorithms when using artificial intelligence and machine learning.

Key Takeaway:

API security should be an integrated part of your API planning, design, and development process. If left as an afterthought, you are gambling on not if, but when, bad actors will find your API and exploit it. The benefits of securing your API are substantial and far outweigh the cost of developing with API security in mind. Following these best practices puts you in a far better position.

Chapter 4: Must-Have Skills for Effective API Security

APIs are among the most frequent targets hackers use to compromise data. API security, which should always be at the forefront for enterprises, requires a particular set of skills — skills that may not be fully available in your current teams. In fact, some of the skills that should be found within your API groups might be surprising. However, these skills are necessary for developing and implementing solutions and strategies that can adequately address the vulnerabilities and security risks unique to APIs. There are technical and non-technical skills your team should have to run an effective API security strategy.

Technical skills

Technical awareness

Having expertise in certain products or brands is not as important as having a thorough technical awareness. Technical awareness gives you a clear picture of the product and technology landscape and is more valuable for understanding and mitigating API security issues.

Designing for API-led connectivity

With API-led connectivity, you can methodically connect data to applications by employing reusable and purposeful APIs. These APIs must be designed and developed to properly execute the specific roles assigned to them and correctly fit into the security framework.

API design best practices

There are well-crafted design principles that can help make APIs more convenient for developers to consume.

These best practices, which can include prioritizing documentation, improve the developer experience, and reduce incorrect code likelihood.

Networking and network security

Areas of vulnerabilities increase as networks, and network infrastructure capabilities continue to drive both application and API development. Knowing what is necessary for maintaining network security is critical.

API security best practices

Adhering to best practices for API security can help ensure that API deployments do not result in security issues. This can entail being mindful of the risks of APIs and carefully monitoring add-on software.

OSAWP API Security Top 10:

1. Broken Object Level Authorization
2. Broken User Authentication
3. Excessive Data Exposure
4. Lack of Resources & Rate Limiting
5. Broken Function Level Authorization
6. Mass Assignment
7. Security Misconfiguration
8. Injection
9. Improper Assets Management
10. Insufficient Logging & Monitoring

General authentication mechanisms

Being serious about API security means having a solid understanding of effective authentication methods, such as Oauth2.0, SAML, SSO, and basic authentication.

General authentication and authorization flow

Securing authentication and authorization requires a careful understanding of API security strategies. Implementing access control using a modern API security strategy can be compared to the airport model, as explained here, where multiple techniques work in unison.

Cyber security threat awareness

To stop hackers, you must know how they implement their attacks. This entails having a working knowledge of:

- OWASP top ten attacks
- Basic attacks
- Advanced attacks
- Attacks at different network layers
- OWASP API Top 10

Non-Technical / Soft Skills

Effective communication

API development does not occur in a vacuum. Effective security requires communication with InfoSec, enterprise architects, developers, CIOs, and CISOs. Security professionals will have to bring team members together and get the most out of each of their skills to help secure the API environment. You need to be able to relay the purpose of the API and what data is exposed and articulate the data in a manner that is easy to understand for those who have to use and apply the information.

Effective communication is also necessary for the gathering of API requirements. Post-development, API developers need good documentation and must communicate design and how to consume APIs. Visual communications, such as diagrams of different layers of security, system interactions, and data flow, can help people understand the API. Visual tools can also expose API definitions or communicate how to consume APIs, assisting consumers in testing the API's end-user version.

Empathy

Empathy helps to provide insight. Not understanding the consumer's perspective can contribute to conflict between the technical teams and users because the consumers want flexibility and greater access. You also need to understand APIs from the viewpoint of hackers to know how to stop them.

Empathy is gained through experience. Being curious and seeking knowledge by asking pertinent questions can also help you understand another point of view. How does the user interact with the API? How would a hacker interface with this API? What kind of information can a hacker get from the API? are examples of the type of thinking empathy allows.

Foresight

You also have to keep in mind the future issues associated with APIs and what contingencies, plans, or solutions need to be in place for what will happen to the API in the future. Long-range issues can pertain to management, maintenance, the impact of personnel turnover, how to execute version control, and how to remain up to date with the latest in API security.

Key Takeaway:

The skills needed for successful development and security of your APIs reach across IT disciplines and roles. Of course, these distinct roles all approach a challenge or project from different perspectives. That is precisely what you want, but at the same time requires a balancing act to achieve the business's goals while developing and maintaining secure APIs. In chapter five, we look at this natural tension that must exist for API security.

Chapter 5: The Natural Tension that Exists in the World of API Security

Secure development of APIs comes from bringing together the right tools, teams, and skills, within a well-considered structure and process. Getting that all to come together well on any project requires a few things. One is that everyone understands and is committed to a common end goal. The other is a balance created between the collaborative but sometimes opposing forces required for success.

There are several areas of tension in the API security world. Exposing data and processes needs to be balanced with permissions and credentials. The speed of development might be challenged with consistent implementation and standards.

For your enterprise's API security to be effective, it is important to plan who must be involved and what responsibilities each party plays. That clarity will help focus on the natural tension of API security into productive areas that improve the end product while avoiding the more harmful elements that a lack of direction can create.

Roles

The three principal entities that lead to top-notch development of API security are:

- Information Security Group
- API Developers
- Enterprise Architects

The operational side of monitoring and managing your APIs should also be mentioned right along with securing your APIs. The API operations team that governs your organization's APIs are a key part of helping you keep APIs up to date, managing APIs, and working with other teams if an attack or breach were to occur.

Ensure that your API operations team is trained on managing your APIs and alerts notify this team so they can react to anomalies and escalate when necessary.

The CIO should define the relationships between the three entities in most cases. The primary benefit of having assigned and defined roles and having players understand their responsibilities is that cohesive relationships can be developed. These relationships will ensure the elements for a comprehensive API security environment are in place.

Typically, API developers will report to the CTO, the InfoSec group will report to the CISO, and enterprise architects are likely to report to either the CIO or CTO.

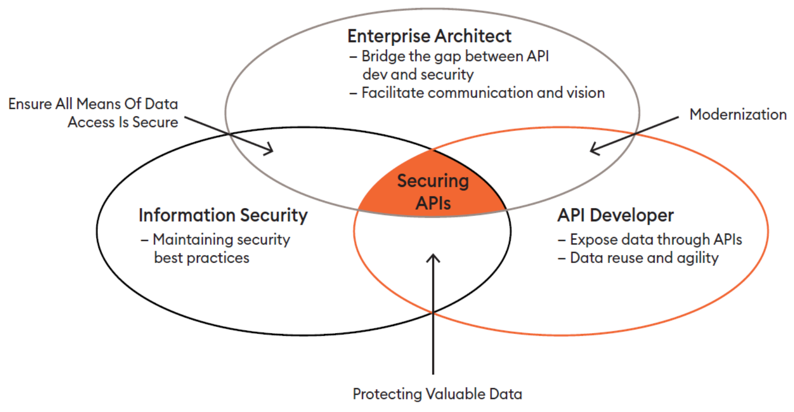
If any of the roles are absent or the players fail to understand or adequately execute their duties, there are likely gaps in your API security.

Roles and responsibilities of the information security group

The information security team is tasked with securing and protecting the enterprise's data and systems. Its objectives, policies, and processes center on locking down the system and protecting valuable data.

Roles and responsibilities of API developers in API security

There tends to be a natural tension between API developers and information security teams because their goals and processes for handling data exist on opposite ends of the spectrum. API developers are focused on exposing data through APIs for reuse and utilizing existing APIs. The developers leverage enterprise data via an API to produce digital value. There is also usually a significant role to play for API developers within API governance and security.


Enterprise Architect
– Bridge the gap between API dev and security
– Facilitate communication and vision
Ensure All Means Of Data Access Is Secure
Modernization
Securing APIs
Information Security
– Maintaining security best practices
API Developer
– Expose data through APIs
– Data reuse and agility
Protecting Valuable Data

Roles and responsibilities of enterprise architects in API security

Enterprise architects bridge the gap between the information security team and API developers. They facilitate communication between both groups to gather sufficient information and insight to create API specifications and governance rules.

Architects have to determine how API security fits into the enterprise's overall security protocols and translate the requirements into the appropriate API architecture, ensuring that it is established, secure, but exposed. Additional responsibilities for Enterprise Architects include updating and modernizing APIs and developing API security rules as the enterprise matures.

The risk of not defining API security roles and responsibilities

An enterprise lacking clearly defined roles and ownership of APIs is prone to gaps in API security management. API developers, primarily focused on functionality and agility, may attempt to shoulder security tasks, leading to guesswork due to the absence of proper expertise. This results in inconsistent management and inadequate security measures.

In the absence of enterprise architects, a natural conflict may arise between information security and API development teams as they grapple with defining their roles in the broader landscape of API development. These two teams have divergent objectives — information security safeguards data, while API developers expose it. Moreover, information security personnel often lack critical insights shared by enterprise architects that are essential for adequately protecting APIs. Conversely, security has not yet ascended to a top priority in API development for many enterprises. The absence of enterprise architecture contributes to a lack of unified communication and effective collaboration between these two essential entities.

This can have a cascading negative impact, resulting in inadequately designed APIs, a lack of API ownership, and an increase in the rework and maintenance costs needed to fix gaps in APIs. Without the proper definition of these roles, this ultimately leads to developing APIs prone to attack by hackers.

Key Takeaway:

For optimal API security within an enterprise, all three entities should be fulfilling their clearly defined roles and responsibilities. The absence of any one party, mainly the information security team or enterprise architects (API developers are always naturally necessary for developing APIs), results in the other party or parties assuming responsibilities they are ill-equipped to execute. If one or more roles are not present, organizations must take a serious look at filling roles or training a team to fulfill all these vital roles in API development.

Chapter 6: Getting Started with API Security

The time to make API security a priority is now, but where to get started? Make it part of your overall IT strategy with a comprehensive approach that involves understanding the risks and vulnerabilities associated with APIs, defining ownership and roles within the organization, implementing appropriate security measures, and adhering to best practices. By prioritizing API security from the beginning and incorporating it into the planning and design phases of API development, you can ensure that your organization's APIs are secure and protected. Below is a high-level list of action items for managing a successful API security strategy.

1. Understand the risks and vulnerabilities associated with APIs.
2. Define ownership and roles within the organization for API security.
3. Document and track your APIs as part of an organizational governance policy.
4. Establish clear communication and collaboration channels between the information security group, API developers, and enterprise architects.
5. Train and educate your team on API security best practices and ensure they have the necessary technical skills and awareness.
6. Implement appropriate security measures, such as API gateways, authentication mechanisms, authorization measures, and encryption.
7. Regularly review and update your API security measures to address new threats and vulnerabilities.

Taking on all these tasks on your own can be overwhelming. To effectively monitor, manage, and secure APIs, leverage the tools and technologies provided by enterprise integration services.

Consider using a third-party provider, like Argano, that brings extensive integration and API security experience and best practices. By doing so, your business benefits from the expertise and resources provided by these services, ensuring the effective implementation of API security measures. This includes utilizing the technical skills and knowledge of the integration service provider as well as tools and technologies for monitoring, managing, and securing APIs. By prioritizing API security and leveraging the expertise and resources of enterprise integration services, you can ensure the security, protection, and effective integration of your APIs within your systems and processes.

Enterprise Integration Quick Start delivered by Argano

As your business grows, so do the complexities of your enterprise applications and infrastructure. Let Argano help you develop a comprehensive IT strategy that optimizes your technology environment for greater agility and scalability. With enterprise integration services from Argano, you get more out of your solutions by leveraging our solution expertise and industry best practices to help maximize your technology investment, minimize risk, and achieve faster, better outcomes.

Visit www.argano.com for more insights or contact us at info@argano.com to find out how Argano can integrate your enterprise to ensure secure applications that drive efficiency and agility.

About Argano

Argano is the world's largest global digital consultancy exclusively connecting design and delivery for the transformation of high-performance business operations, extending our clients' commercial agility, profitability, customer experience, and growth. Learn more at www.argano.com.