Securing the Next Generation of Digital Infrastructure: The Importance of Protecting Modern APIs

Dhaka Timsina

Spri2023-S10-CAPS795.30782

Davenport University

Professor: Lonnie Decker

July 8, 2023

Table of Contents

Abstract 3

Introduction 4

Thesis Statement 4

Problem Statement 4

Research Questions 5

Literature Review 5

Research Methods 10

Results 12

Conclusion and Suggestions for Future Research 13

References 16

Abstract

In today's interconnected digital world, Application Programming Interfaces (APIs) serve as the backbone of modern software infrastructure. APIs enable seamless communication between various information technology systems to allow applications, websites, and other services to interact and share data continuously among the interconnected systems. The rapid growth of APIs in various industries has brought about a pressing need to address the security risks associated with their usage. This research paper examines the importance of implementing robust security controls for APIs and highlights the inadequacy of traditional security approaches in safeguarding modern APIs. The paper discusses the 2023 list of OWASP API vulnerabilities, which includes new threats identified through the analysis of recent data breaches. The impact of API vulnerabilities is illustrated through notable cases, such as the T-Mobile and Twitter data breaches. The limitations of traditional security tools, like web application firewalls (WAFs), in protecting APIs are highlighted, emphasizing the need for specialized API protection systems. The research also explores the emerging trend of incorporating API security in DevOps pipelines and the availability and effectiveness of API security tools from leading security providers. Based on the findings, the paper recommends essential features for API security tools, including discovery, reconnaissance, posture management, security testing, and run-time protection. These features aim to provide visibility, proactive risk management, and real-time threat detection to mitigate the evolving API security landscape. Overall, the paper underscores the criticality of robust API security controls and the need for organizations to adopt comprehensive approaches to protect their APIs against evolving threats.

Introduction

An application Programming Interface (API) is a programming system that facilitates the communication between two software components using specific rules and protocols. The use of APIs in various industries has increased significantly in the last decade. According to a survey by Rapid, a leading API hub, 63% of the application developers reported utilizing more APIs in 2022 compared to 2021, and 75% of developers stated they are prioritizing API developments in their organizations (Khoury, 2023). These statistics indicate a strong potential for rapid API developments in the future. However, the growing reliance on APIs in organizations also raises concerns about their security risks. With the development and prioritization of API in almost every industry, there's an urgent need to understand the security risk posed by the growing number of APIs. Organizations that heavily rely on APIs for various data migrations inside and outside the company are particularly concerned about API security. Investigations and analyses by various security firms reveal that not having proper API security is a significant concern today. US companies lost approximately $12 to $23 billion in 2022 alone from data breach associated to API vulnerabilities (Lemos, 2022). This number is likely to increase if no steps are taken to secure APIs. As we embark on the next generation of digital infrastructure, securing APIs is critical to any organization.

Thesis Statement

The lack of emphasis on API security has led to significant data breaches resulting in major financial and reputational damage to organizations.

Problem Statement

There is an urgent need to analyze the current state of API security and implement effective security controls to mitigate the risk associated with API vulnerabilities. The number of data breaches involving vulnerable APIs has increased significantly in the last few years.

Research Questions

1. What are the potential consequences of not adequately protecting APIs?
2. What are organizations doing to protect modern APIs? Are these protection strategies enough to prevent data breaches?
3. Why traditional web application protection systems may not protect modern APIs?
4. How can we secure modern APIs?

Literature Review

The Open Worldwide Application Security Project (OWASP), a non-profit organization dedicated to improving the security of software, has been releasing its top 10 web security vulnerabilities every 2-3 years since 2003. In response to the increased vulnerabilities associated with APIs, the Foundation began releasing a list of the top 10 API vulnerabilities starting in 2019. The separation of the top 10 vulnerabilities for web applications and API by OWASP Foundation shows that the modern APIs are different from traditional web applications and a different approach is necessary to secure them. The official list of the top 10 API vulnerabilities for 2023 came out a month ago. According to the OWASP Foundation (2023) below is the list of the OWASP 2023 top 10 API vulnerabilities:

- API1:2023 - Broken Object Level Authorization (BOLA)
- API2:2023 - Broken Authentication
- API3:2023 - Broken Object Property Level Authorization
- API4:2023 - Unrestricted Resource Consumption
- API5:2023 - Broken Function Level Authorization (BFLA)
- API6:2023 - Unrestricted Access to Sensitive Business Flows
- API7:2023 - Server Side Request Forgery

- API8:2023 - Security Misconfiguration
- API9:2023 – Improper Inventory Management
- API10:2023 - Unsafe Consumption of APIs

The 2023 list has new vulnerabilities that are different from the official list released in 2019 as new API threats emerge daily, and more data breaches involving API vulnerabilities have been studied and analyzed by the OWASP community. Some recent data breaches involving APIs have highlighted the fact that APIs have become prime targets for malicious actors. Let's review some of the latest data breaches associated with APIs.

The first case involves a massive data breach that garnered significant media attention in early 2023. The T-Mobile data breach, which occurred in late November 2022 but was publicized in January 2023, served as a wake-up call for many organizations that have overlooked the security posture of their APIs. The OWASP list of top 10 API vulnerabilities identifies Broken Object Level Authorization (BOLA) as the number one vulnerability. The T-Mobile breach, which led to the theft of approximately 37 million customer records, exploited a vulnerable API (Stupp, 2023). In its Security and Exchange (SEC) filing following the attack, T-Mobile stated that unauthorized access was gained through one of its APIs resulting in the data breach. Several security firms confirmed that the T-Mobile breach was because of an unsecured API. T-Mobile's official statement, along with the findings of multiple security firms, indicates the data breach involved exploitation of the BOLA vulnerability.

Large organizations like T-Mobile usually have robust network security controls to protect their assets. However, attacking APIs does not require sophisticated tools and techniques typically used to breach a secure network. Since externally facing APIs are publicly accessible on the Internet and are not secured by layer 3 firewalls, threat actors can use any publicly

available tools and techniques to exploit inherent vulnerabilities of the API. One of the primary reasons APIs are attractive targets is that security teams often lack the processes to identify and maintain an inventory of APIs within their organization, let alone remediate existing vulnerabilities (Keary, 2023). In addition to maintaining a secure internal network, if T-Mobile had implemented a secure solution to track all their APIs and performed routine security audits, this attack could have been prevented or at least significantly mitigated.

The second recent data breach case involving an API is the ransomware attack on Twitter at the end of 2022, during which details of approximately 200 million Twitter users were posted on a hacker's forum after ransom negotiations failed. The attack on Twitter also exploited a vulnerability in one of its APIs. Public-facing APIs that handle sensitive information usually require users to authenticate with API keys, but this was not the case with Twitter (Suciu, 2023). This API had a broken authentication vulnerability, listed as number two in the 2023 OWASP top 10 API vulnerabilities. A bug in Twitter's API between June 2021 and January 2022 allowed attackers to submit information, such as an email address, and receive a corresponding Twitter account in return (Newman, 2023). This API vulnerability was extensively exploited to extract large amounts of data from Twitter until it was patched. A vulnerability that persisted for over six months poses a significant challenge for a social media company like Twitter. With effective API security controls in place, this vulnerability would have been detected early and either patched immediately or mitigating measures applied to prevent further damage. Additionally, implementing robust security practices such as stringent access controls, anomaly detection, and comprehensive auditing could have proactively detected and mitigated the issues preventing unauthorized access from the bad actors.

In addition to the data beaches described above, there have been numerous API attacks resulting in massive data breaches in 2022 alone. For example, in January 2022, the Texas Department of Insurance experienced a breach where the personal information of 1.8 million Texans was stolen by exploiting a web application connected to an API. A flaw in the web service application, which can be categorized as BFLA vulnerability unintentionally made protected areas of the application accessible (McCormick, 2023). BFLA vulnerability is ranked number five on the 2023 OWASP API vulnerabilities list. Similarly, FlexBooker, a scheduling platform, was targeted by exploiting an API vulnerability that mistakenly allowed users to access and retrieve the personally identifiable information (PII) of other users. The harvested data was subsequently sold on the dark web. This vulnerability was categorized as BOLA attack by security experts, the top-ranked vulnerability. Numerous other examples exist where APIs have been exploited to gain unauthorized access and steal sensitive information, including PII. All these examples above suggest that despite knowing the vulnerabilities, there's little to nothing done by companies to protect APIs from being exploited.

When it comes to API security, many organizations rely on traditional security protections such as Web Application Firewalls (WAF), Identity and Access Management (IAM) systems, application gateways, etc. However, such controls are not sufficient to thwart attacks targeting API vulnerabilities. I have been working as an application security specialist for an organization in the US Midwest region for the past four years. During this time, I have gained firsthand experience in configuring and managing application security architectures. For large corporations like the one I am employed with, multiple security controls are implemented to prevent attackers from accessing the corporate network. These controls can be broadly categorized into three main types: physical, administrative, and technical. Physical controls

include measures such as locks, security cameras, and security guards to safeguard physical access to the organization's premises. Administrative controls encompass the development and enforcement of policies, standards, and security training programs to promote a culture of security within the organization. Technical controls involve the use of various tools to monitor and control user access, such as firewalls, access control lists, anti-virus software, and intrusion detection and prevention systems (IDPS), among others.

However, analysis indicates the recent API attacks have been primarily focused on exploiting OSI Layer 7 security controls rather than the Layer 2 and Layer 3 controls which often require advance skills. Layer 7 attacks directly target application vulnerabilities, making secure coding practices essential for protecting against such attacks. While large corporations often employ web application firewalls (WAFs) to defend against application vulnerabilities, there are limitations when it comes to protecting APIs using WAFs. WAFs primarily protect against known vulnerabilities by matching signatures with a database. They also provide significant protection against Denial-of-Service (DoS and DDoS) attacks, automated bot attacks, and the OWASP Top 10 vulnerabilities. It's important to note that the OWASP Top 10 vulnerabilities differ from the OWASP API Top 10 vulnerabilities, although there is some overlap between the two lists. WAFs were not originally designed to protect APIs. Although most WAFs can parse JSON and REST, they cannot often parse other API data protocols (Hiremath, 2023). The inability to parse API protocols means that WAFs will not be able to inspect the packets and block malicious traffic. Additional security tools such as gateways and IAM systems, alongside WAFs, play a crucial role in protecting web applications and some APIs. However, these tools and systems alone are insufficient to fully safeguard against API vulnerabilities. They rely on proxies, which introduce overhead to API communication, and they can only protect against

known signatures, leaving organizations vulnerable to zero-day vulnerabilities. Moreover, these tools often lack specialized protection against business logic attacks and API abuse or misconfiguration. While they work in tandem with API protection, having a dedicated API protection system is essential for gaining visibility within the API environment and applying specialized defenses against API threats.

Many organizations now recognize the importance of securing their APIs and have started incorporating API security into their DevOps pipelines. The shift-left approach to application security has become a prominent trend in the developer's world. However, only a handful of companies are currently considering a secure DevOps environment for APIs. MuleSoft, a major integration software provider, is at the forefront of this movement, having introduced a secure API management suite called Anypoint. This suite enables developers to securely create, manage, and coordinate both on-premises and cloud-based deployments of their APIs (Ghoshal, 2021). This trend is likely to be followed by other firms, including MuleSoft's competitors such as F5, IBM, Microsoft, Oracle, Boomi, Informatica, and Jitterbit, among others. Therefore, security is no longer optional when it comes to creating and deploying APIs. Organizations must prioritize API protection and invest in specialized tools and systems to address the unique challenges posed by API vulnerabilities. By implementing robust API security measures, organizations can mitigate risks, protect sensitive data, and maintain the trust of their customers and partners.

Research Methods

Recent breach analyses indicate a clear correlation between API vulnerabilities and the percentage of API attacks. While traditional web application attacks still make headlines, API attacks have gained prominence. A study conducted by SALT (2022), a leading API security provider, reveals a 400% increase in API attacks in December 2022 compared to previous

months. This surge in API attacks is expected to continue in 2023. Many of the major attacks in 2022 focused on exploiting the top two API vulnerabilities identified by the OWASP community. These findings suggest a lack of effective API security controls despite the vulnerabilities published by the OWASP foundation and other security communities. Many API security tools can address the top 10 API vulnerabilities identified by OWASP Foundation. Furthermore, many of these tools also possess the ability to detect and mitigate zero-day attacks. Leading Content Delivery Networks (CDNs) and API protection providers like Imperva and Akamai provide the ability to create custom rules to block zero-day attacks.

The increasing demand for API security is evident across organizations worldwide, leading to significant revenue growth for top API security tools. SALT, for example, experienced a 160% increase in customer count and a 400% revenue increase in 2020 alone, attributed to a 380% surge in API traffic on their network (Lunden, 2021). Similarly, other API security providers have reported substantial growth in their customer base. According to Noname (2023), their API security product is utilized by 25% of Fortune 500 companies, protecting over 10 million critical APIs and securing nearly 3 trillion dollars in annual revenue streams. These statistics provide strong evidence supporting the necessity of API protection, as traditional security measures and firewalls have proven insufficient in safeguarding modern APIs.

It is no longer optional for companies, regardless of their size, to rely solely on traditional security measures to protect their APIs. The need for API security is widely recognized, and organizations must adopt specialized API security tools and practices to effectively mitigate risks and protect APIs from evolving threats.

Results

The case analysis above highlights the increasing importance of protecting APIs due to the emergence of new vulnerabilities and the rising number of data breaches associated with API vulnerabilities. The OWASP Foundation's release of the top 10 API vulnerabilities for 2023 further emphasizes the need for organizations to address these specific threats. The analysis of recent data breaches involving APIs, such as the T-Mobile and Twitter incidents, illustrates the real-world consequences of failing to secure APIs effectively.

The T-Mobile data breach demonstrated that even large organizations with robust network security controls can be vulnerable to API attacks. The lack of proper inventory management and routine security audits contributed to the breach. The Twitter incident showcased the impact of broken authentication vulnerabilities in APIs and the potential for unauthorized access to sensitive user information.

Additionally, the importance of protecting APIs is further reinforced by the numerous other data breaches mentioned in the literature review, including those affecting the Texas Department of Insurance and FlexBooker. These examples demonstrate that companies often overlook the vulnerabilities associated with APIs, allowing malicious actors to exploit them and gain unauthorized access to valuable data.

Traditional security measures, such as Web Application Firewalls (WAFs) and Identity and Access Management (IAM) systems, are insufficient for API protection. WAFs, in particular, were not originally designed to protect APIs and have limitations in terms of parsing API protocols and inspecting packets. The attack for API is different from that of the web application. Authentication, authorization, excessive data exposure, and misconfigurations vulnerabilities are

not identified by WAFs. Additional security tools like gateways and IAM systems play a role but lack specialized protection against API-specific threats.

The shift towards a secure DevOps environment for APIs is a promising trend, with companies like MuleSoft leading the way in providing dedicated API management suites. This approach recognizes the unique challenges posed by API vulnerabilities and emphasizes the need to incorporate API security into the development and deployment processes.

Lately, APIs have become prime targets for attackers, and failing to secure them properly can lead to significant data breaches and loss of customer trust. By implementing robust API security measures, organizations can mitigate risks, protect sensitive data, and maintain the integrity of their systems and the trust of their stakeholders.

Conclusion and Suggestions for Future Research

There are several security tools available in the market specifically designed to protect APIs. According to Gartner Reports (2023), leading API security providers include SALT, Noname, Imperva, Cequence, and Synopsys. These tools offer specialized API protection, and organizations can choose the one that best fits their API architecture. While some tools may have higher costs, they also provide additional capabilities. Therefore, organizations should select an API security tool based on their affordability and the tool's ability to meet their specific requirements. The chosen tool should encompass essential API security features that are not typically provided by traditional WAFs. These important features include discovery, reconnaissance, posture management, API security testing, code reviews, and runtime protection.

The API security tool should possess the capability to discover and inventory all APIs within the corporate environment. After all, you cannot protect what you cannot see. Shadow

APIs and zombie APIs that are not previously managed or inventoried could pose serious security risk to the organizations. As a matter of fact, Salt in its state of API security, identifies zombie APIs as the number one API concern in the last four years (Rago, 2022). By maintaining an accurate and up-to-date list of both external and internal APIs, the security team can effectively manage and safely decommission outdated APIs. Similarly, reconnaissance is essential to identify all APIs exposed to the public that may have not been known to the organizations. Recon removes blind spots on the external API attack surface by identifying the attacks paths accessible to hackers (Verloy & Rowe, 2023). Posture management should be incorporated into the security DevOps cycle, allowing the security team to have visibility into the API development, deployment, and operation lifecycles. Regular testing and code reviews of API applications are critical to ensure that vulnerable APIs are not deployed in production. This approach can be integrated into the early stages of the software development life cycle, following the widely adopted "shift left" approach by software developers. Identifying and mitigating security flaws in the early stages reduces risks. Therefore, the API security tool should seamlessly integrate with the DevOps cycle. Lastly, runtime protection is a crucial security feature. Every API should have mechanisms in place to identify active API attacks and automatically apply mitigation controls or alert the security team to quickly isolate incidents and implement mitigation measures on-the-fly. This real-time detection and response to security anomalies contribute to maintaining the overall security of the system.

API security tools can help mitigate attacks targeted at API vulnerabilities, but they cannot eliminate all threats. It is crucial to consider additional security measures beyond those provided by API security tools. Controls must be in place to address the OWASP API top 10 vulnerabilities.

Authentication and authorization can be implemented using API keys, JSON Web Tokens (JWT), OAuth, or certificates. Role-based access with the least privilege enables administrators to monitor and audit access and reduce the impact of compromised credentials.

Applying security protection is essential for API data at rest, in transit, and in use. Implementing disk encryption safeguards data at rest, while utilizing HTTPS and TLS ensures secure communication channels. Data Loss Prevention (DLP) tools can be employed to protect data in motion. Similarly, safeguarding data in use can be achieved through access and identity management tools, single sign-on, and multi-factor authentication.

Regular patching and updating of APIs is crucial to protect against known vulnerabilities. Additionally, input validation and rate limiting, similar to other web applications, are necessary for APIs.

Logging and monitoring play a vital role in tracking API activities and enabling prompt action in response to anomalies. Finally, periodic security awareness training is critical to educate users about API usage and promote adherence to best security practices. Humans are often the weakest link, but they can also be the best defense by raising awareness and reporting potential vulnerabilities and risks to the security teams.

References

API attacks are on the rise. (2022). *SALT*. https://salt.security/api-security-trends?&utm_medium=nurture&utm_source=email&utm_campaign=general

API protection tools reviews and ratings. (2023). *Gartner*. https://www.gartner.com/reviews/market/api-protection-tools

Ghoshal, A. (2021, December 13). Mulesoft updates Anypoint to streamline API management, support devops. *InfoWorld.com*, NA. https://link-gale-com.proxy.davenport.edu/apps/doc/A686608925/GBIB?u=lom_davenportc&sid=summon&xid=65187524

Hiremath, O. (2023, January 16). Why WAFs are not enough. *Software Secured*. https://www.softwaresecured.com/why-wafs-are-not-enough/#:~:text=WAFs%20have%20a%20limitation%20on,request%20to%20bypass%20WAF%20security.

Industry-Leading Impact. (2023). *Noname Security*. https://nonamesecurity.com/

Keary, T. (2023, January 20). T-Mobile data breach shows API security can't be ignored. *Venture Beat*. https://venturebeat.com/security/t-mobile-data-breach-shows-api-security-cant-be-ignored/

Khoury, J. (2023, January 14). State of APIs: growth and more growth on tap for 2023. *Rapid*. https://rapidapi.com/blog/state-of-apis-growth-and-more-growth-on-tap-for-2023/

Lemos, R. (2022, June 30). API Security Losses Total Billions, But It's Complicated. *Dark Reading*. https://www.darkreading.com/application-security/api-security-losses-billions-complicated

Lunden, I. (2021, May 26). Salt Security lands $70M for tech to protect APIs from malicious abuse. *Tech Crunch*. https://techcrunch.com/2021/05/26/salt-security-lands-70m-for-tech-to-protect-apis-from-malicious-abuse/?guccounter=1&guce_referrer=aHR0cHM6Ly93d3cuZ29vZ2xlLmNvbS8&guce_referrer_sig=AQAAAFrMWIN7ljVE6

McCormick, A. (2023, January 11). The top 5 API security breaches in 2022, and how to avoid them in 2023. *Cisco Tech Blog*. https://techblog.cisco.com/blog/top-5-api-security-breaches-in-2022#:~:text=In%20July%202022%2C%20a%20major,users%2C%20and%20mistakenly%20revealed%20PII.

Newman, L. H. (2023, January 5). What Twitter's 200 Million-User Email Leak Actually Means. *Wired*. https://www.wired.com/story/twitter-leak-200-million-user-email-addresses/

OWASP API Security Project. (2023). *OWASP*. https://owasp.org/www-project-api-security/#

Rago, N. (2022, October 30). Why Are Zombie APIs and Shadow APIs So Scary?. *DarkReading*. https://www.darkreading.com/edge-ask-the-experts/why-are-zombie-apis-and-shadow-apis-so-scary-

Stupp, C. (2023, Jan 24). T-Mobile Breach Highlights Security Achilles' Heel of API. Wall Street Journal http://search.proquest.com.proxy.davenport.edu/newspapers/t-mobile-breach-highlights-security-achilles-heel/docview/2768625595/se-2

Suciu, P. (2023, January 5). Cybersecurity Experts Warn Twitter Breach Will Have Lasting Ramifications. *Forbes*. https://www.forbes.com/sites/petersuciu/2023/01/05/cybersecurity-experts-warn-twitter-breach-will-have-lasting-ramifications/?sh=4d072aa32032

Verloy, F., & Rowe, G. (2023, June 9). API Security Reconnaissance As A Service using Noname Recon. *Security Boulevard*. https://securityboulevard.com/2023/06/api-security-reconnaissance-as-a-service-using-noname-recon/