ec ring the Ne t Gener tion of Digit nfr str ct re he port nce of Protecting Modern AP s

Dhaka Timsina

Spri 0 -S 0-CAPS . 0

Davenport University

Professor Lonnie Decker

July , 0

e of Contents

A str ct

ntrod ction

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, # % of the application developers reported utilizing more APIs in 0 compared to 0 , and % of developers stated they are prioritizing API developments in their organizations (Khoury, 0). 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 $ to $ billion in 0 alone from data breach associated to API vulnerabilities (Lemos, 0). 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.

hesis t te ent

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

Pro e t te ent

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.

Rese rch Q estions

.   What are the potential consequences of not adequately protecting APIs

.   What are organizations doing to protect modern APIs   Are these protection strategies

- API .0 - Security Misconfiguration
- API .0 Improper Inventory Management
- API 0 .0 - Unsafe Consumption of APIs

The 0 list has new vulnerabilities that are different from the official list released in 0 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 0 . The T-Mobile data breach, which occurred in late November 0 but was publicized in January 0 , served as a wake-up call for many organizations that have overlooked the security posture of their APIs. The OWASP list of top 0 API vulnerabilities identifies Broken Object Level Authorization (BOLA) as the number one vulnerability. The T-Mobile breach, which led to the theft of approximately million customer records, exploited a vulnerable API (Stupp, 0). 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 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

In addition to the data beaches described above, there have been numerous API attacks resulting in massive data breaches in 0 alone. For example, in January 0 , the Texas Department of Insurance experienced a breach where the personal information of . 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, 0). BFLA vulnerability is ranked 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   security controls rather than the Layer   and Layer   controls which often require advance skills. Layer   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 0 vulnerabilities. It s important to note that the OWASP Top 0 vulnerabilities differ from the OWASP API Top 0 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, 0). 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 known signatures, leaving organizations vulnerable

Res ts

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 0 API vulnerabilities for 0 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 s

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, 0). 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, 0). 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

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.

Lunden, I. (0 , May #). Salt Security lands $ 0M for tech to protect APIs from malicious abuse. ***Tech Crunch***. https //techcrunch.com/ 0 /0 / #/salt-security-lands- 0m-for-tech-to-protect-apis-from-malicious-abuse/ guccounter &guce referrer aHR0cHM#Ly d cuZ vZ xlLmNvbS &guce referrer sig AQAAAFrMWIN ljVE#

McCormick, A. (0 , January). The top API security breaches in 0 , and how to avoid them in 0 . ***Cisco Tech Blog***. https //techblog.cisco.com/blog/top- -api-security-breaches-in- 0 # text In% 0July% 0 0 % C% 0a% 0major,users% C% 0and% 0mistakenly% 0revealed% 0PII.

Suciu, P. (0 , January). Cybersecurity Experts