The Evolving API Security Landscape

by Kin Lane

About Ping Identity

Ping Identity is pioneering Intelligent Identity. We help enterprises achieve Zero Trust identity-defined security and more personalized, streamlined user experiences. The Ping Intelligent IdentityTM platform provides customers, employees, partners and, increasingly, IoT, with access to cloud, mobile, SaaS and on-premises applications and APIs, while also managing identity and profile data at scale.

About TIBCO

TIBCO fuels digital business by enabling better decisions and faster, smarter actions through the TIBCO Connected Intelligence Cloud. From APIs and systems to devices and people, we interconnect everything, capture data in real time wherever it is, and augment the intelligence of your business through analytical insights. Thousands of customers around the globe rely on us to build compelling experiences, energize operations, and propel innovation. Learn how TIBCO makes digital smarter at www.tibco.com.

Over the last couple of years, the API security landscape has significantly shifted and expanded due to growing threats and the evolving lifecycle for deploying, managing, testing and operating APIs. During this period, much of the API security conversation has focused on authentication, authorization, rate limiting and other foundational features provided by the API management sector. And while API management solutions provide an important set of foundational security tools, the conversation should not end here. This whitepaper looks at how we got here, where things are going, and how the landscape is beginning to rapidly shift and evolve.

The State of API Security

Much of what we've learned about securing web applications can apply to securing APIs. Yet, our security practices need to evolve and consider the unique needs of API usage across mobile, voice, and other emerging applications. Rather than focus on how web security practices help API security, we should analyze how these practices differ from the needs of API security. For example, API clients are often mobile devices and applications, whereas website clients are primarily browsers and search engine bots. API security is about striking a balance between making valuable data, content, media, algorithms, and other digital assets accessible to authorized users, while preventing unauthorized users from gaining access. APIs go beyond simple web publishing and provide deeper engagement with data and applications, requiring a new security paradigm.

Over the last decade, API management has become synonymous with API security. Authentication and rate limiting

are considered core API management security features and ensure resources are securely accessible by internal groups, partners, and third-party developers. In an evolving threat landscape, we must build on the established base of healthy API management and security practices by expanding our toolbox according to the unique needs of APIs.

API security isn't just a handful of stops across the API life cycle. As systems are decoupled and further distributed with continuous deployment and integration practices, the threat landscape is expanding. We are going from a handful of APIs and web services to thousands of microservices and event-based solutions that potentially operate across many infrastructure providers and regions around the globe. API security must move beyond an API management state of mind and widen its view across all API operations.

Today's Threats

APIs face a range of threats today, but hackers are discovering that API-driven digital transformation efforts are an attractive target to gain access to corporate data, applications, and systems. Since APIs are often included in web development efforts, IT administrators tend to consider these deployments secure because they are managed as part of regular operations. However, exposing APIs creates a new, attractive path for hackers to gain access to sensitive data. Let's explore some common API attacks, and why existing security practices are deficient in stopping them.

Login Attacks

In this type of attack, a hacker typically begins by probing the environment and then executing attacks to bypass or defeat login systems. The API surface area for popular mobile applications is often mapped out using off-the-shelf proxy software and published to GitHub as a machine-readable blueprint for hackers to exploit an application's infrastructure. With this blueprint, a hacker can probe API paths and look for potential access points (e.g. login services) to the API. An area of interest for hackers is the authentication system, which can support one or more access technologies including basic login, OAuth tokens, API keys, or a combination (e.g. API key + token). After discovering the authentication scheme, a hacker will develop attacks to compromise the service. Once login services have been compromised, attackers will continue to probe APIs for vulnerabilities. This underscores the need to better understand the nature of API attacks pre and post login.

Example Login Attacks

The IRS announced in February 2016 that almost 400,000 taxpayer accounts were compromised by an attack on the "Get Transcript" API. The attackers used a brute force attack in conjunction with personal information separately gathered to extract tax returns. In addition to social security numbers, hackers were able to access wage and other sensitive information.

Hackers also compromised a mobile device manufacturer's cloud API by allegedly using a brute force attack to compromise the passwords of celebrity accounts. The hackers downloaded personal information from the accounts and published nude photos, which were later traced back to the attack.

Login services are a common API attack surface. Many recent breaches have exposed account data including usernames and passwords, which were likely reused on corporate accounts. A recent HBR article estimates a 2% success rate for password guessing attacks with stolen passwords. It may not sound like a big threat, but 2% of a million is 20,000 successful guesses. Hackers can rent botnets that programmatically send API calls to test usernames and passwords on API login services. Although API management systems will reject invalid login attempts, these systems have inadequate mechanisms to stop clients from continuously trying new combinations. Many hackers keep request rates below rate limits and periodically change IP addresses to make control very difficult, and successful attempts often go undetected.

Hackers can also steal API keys or tokens used for client authentication through man in the middle attacks, tricking users into connecting to a compromised system that then captures the user's token or key. The hacker then presents the stolen credential to gain access to API services. Since proper credentials are presented by the client, an API management system cannot detect this attack.

API DDoS Attacks

Unlike volumetric DDoS attacks designed to overrun an organization's defenses, API DDoS attacks are often executed by multiple clients sending traffic to overload an API service. Since each hacker sends normal traffic volumes, these attacks are difficult to detect without analyzing the aggregate traffic rate on each unique API service. Sophisticated hackers can even detect rate limiting controls and adapt traffic rates to stay beneath the throttling limits to avoid detection. Although API management systems use rate limiting to control individual client activity, these systems typically cannot view aggregate traffic rates among multiple clients to stop distributed DDoS attacks.

Example API DDoS Attacks

Since DDoS attacks disrupt access without breaching data, organizations typically do not report DDoS attacks. For example, the IRS initially thought the attack that compromised taxpayer accounts was a DDoS attack and later determined it was a much more serious data breach.

A video streaming service recently published a paper on an internal attack that took advantage of their microservices architecture to compromise their backend systems. The inbound request generated a cascading set of requests, which overran their servers.

Examples of DDoS targets include login services, session management and many other services critical to an application's reliability. A group of hackers could simultaneously send login requests and make it difficult for legitimate users to log in to the API. A similar attack could be performed on cookie management or token servers. Hackers could also target a shopping cart or other important API service with bogus requests designed to impact service availability.

In addition to disrupting applications, DDoS attacks can generate high computing costs, particularly in cloud environments with consumption-based billing practices. Many organizations have enabled elastic computing services, which kick in during peak demand. Without the proper controls, a DDoS attack can impact customer access to important services while also driving up computing costs.

Application and Data Attacks

With the right credentials, insiders and hackers can access any system or data they'd like without further efforts to compromise security. Today, open business and digital transformation initiatives often include giving partners access to applications and data through APIs using their own corporate credentials. But without the immediate knowledge of when employees leave one of potentially hundreds of partner organizations, how can an enterprise discern between valid and and rogue API traffic? Furthermore, the increasing prevalence of phishing kits and keyloggers should drive organizations to operate under the assumption that credentials aren't sufficient to protect their most vital assets. Since attackers with compromised credentials look like valid clients, API management systems have no way to recognize when a compromised user is accessing applications via the APIs. There is also a presumption in the enterprise that APIs built for internal use are somehow less prone to attack from an outside source. The prevalence of compromised credentials combined with a common lack of centralized API governance often exposes APIs meant for internal use only to the outside world.

Example Application and Data Attacks

A popular social media site announced in September 2017 that information from more than 6 million user accounts was exposed by a flaw in their API. Although password data was not compromised, hackers were able to access contact information of some of their high-profile customers.

Application and Data Attack Types

Application and data vulnerabilities depend on the exposed API functionality. For example, an API with exclusively read-only functionality exposed would likely not be susceptible to an injection attack. However, APIs commonly expose a range of functionality, and examples of attacks using these functions include:

- Data extraction or theft: Instead of looking up a single account, a hacker could program an attack to gather information from many accounts.
- Data deletion or manipulation: A disgruntled employee could delete information to sabotage systems, or a hacker could change data to compromise information.
- Data injected into an application service: A hacker could load large data files to overrun system memory or inject excessive data to overload an API service.
- Malicious code injection:A hacker may inject malicious code, such as a key logger, which could compromise other users accessing the service.
- Extreme application activity: A hacker can generate calls that require unusually high system resources and affect server response time.

These examples display just a few API vulnerabilities exposed by compromised credentials. Organizations should review API management system deployments and identify the API services that could expose sensitive information. At a minimum, these API services should be secured beyond the foundational access control protection provided by API management systems. The following sections will discuss considerations for building this security layer.

API Security Moving Forward

The next wave of API security must focus on making intelligent API security an integral part of API operations. The IT and security operations teams must be agile and responsive, and work with the latest information, leveraging both internally and externally collected data. Let's discuss how we can get there.

Behavioral Analysis

Before API management systems, organizations barely understood API consumer behavior and resource utilization. Now, organizations have a robust understanding of which consumers access their resources and which behaviors to incentivize from a technical and a business standpoint. API authentication and service composition help enforce behavioral models for API operations. These policies determine which groups can access what resources and at what cost. API Management can throttle and incentivize positive behavior, but how is this translated into security terms? API monitoring reports what we know about API consumption but provides minimal reporting on the behavior of bad actors. We don't have tools to identify unknown behavioral models, let alone the tools to automate and scale our response.

API management and monitoring should leverage behavioral security models because policy based access control models for APIs can be difficult to scale. Organizations must understand real-time behavior using existing information sources, including platform logging and external threat information from third-party threat sharing networks, to defend APIs from threats that target API platform providers.

API platform behavior must be developed in house and put immediately into production to respond to threats. Our models must be automated with humans engaged at the right points. API security isn't about locking everything down tight. It is about putting everything on the web in a well-defined way with proper identity and access management controls, understanding who has access to what, encouraging positive behavior, and quickly shutting down negative behavior.

API Analysis

Analysis has been baked into API operations as part of the API management suite. Moving forward, we need analysis to exist across all stops along the API lifecycle, beginning with the definition and design process, following through deployment and management, as well as monitoring, testing and every other aspect of API operations in production. We need API analysis to expand into a more robust set of capabilities so it can work in tandem with other API security efforts.

Analysis needs to work in concert with API security operations. It needs to provide analysis to API security operations, as well as lend a hand in analyzing API security practices, models, and execution. API analysis in the coming years should be providing insight into threats, helping us understand security, as well as the business side of API operations. We need to realize that API security is never a done deal, even with encryption, authentication, and rate limiting in place, and a dashboard telling us which users and applications are actively using our APIs.

Testing

API service providers give us an external lens to look through when monitoring APIs. The monitoring and testing of each API uses a common set of assertions regarding what each API should and should not be doing. Testing the availability, integrity, and performance of each API can just as easily surface a security hole as it can an error in API code. Security attack vectors inject extra or malformed information outside the intended behavior of an API's operation. Bad actors often exploit errors, making API testing a critical tool in our API security toolboxes.

Modern approaches to API testing allow for structured, repeatable tests conducted on regularly scheduled and event-based patterns. These tests can be used as part of security agreements, ensuring that APIs are delivering as expected within the framework of security guidelines established by IT and security operations. API monitoring and security testing go hand in hand. API security testing must analyze the performance and integrity of your API calls. API performance can often be an early warning sign of a security incident, and testing the integrity of API requests and responses can identify a breach before it spreads.

Modeling

Machine learning provides tools for developing models to allow positive behavior and block negative behavior. But the proper tools or services must be in place to make these models a reality. These models can be developed from existing logging practices that are part of API operations and integrate external sources of information to include additional threat patterns not captured internally. The contract between API providers and consumers that details the intended uses for a given API makes the application of artificial intelligence and machine learning techniques especially useful for modeling normal behavior and easily detecting anomalies.

Emerging API security providers train their models across multiple API providers and threat information databases. This creates an important role for API security service providers to step up and compete based upon the most mature, evolved machine learning models. The challenge for API providers will be doing their homework to understand which security providers have the best models to complement their internal security platforms.

Reporting

Many organizations struggle to manually review the massive amount of log data generated on API activity. Attackers often leave clues that could be discovered in the log data, but operations teams often don't have the bandwidth or

skills to detect these threats in time to thwart an attack. Machine learning systems built to analyze API logs can detect anomalous activity, which could indicate an attack. These systems can synthesize suspicious activity into forensic reports, which can be reviewed by security analysts for attacks. In addition, these reports could provide in-depth insight into API activity, which can be used as input for compliance reporting for organizations in heavily regulated industries.

Compliance

Corporate security compliance doesn't have to be part of regulatory compliance. It could be an organization-wide set of security guidance with flexible enforcement mechanisms. Your API security team should look at industry recommendations and National Institute of Standards and Technology (NIST) policies for best practices. When it comes to API security, corporate and regulatory compliance can be a lofty goal, but adhering to best practices can make a significant difference in the impact of an API security breach.

Conclusion

Next-generation API security providers are emerging with services and tooling to help automate, scale, and augment API management, monitoring, and other operational security practices. And the latest wave of API security providers is bringing new processes and technology to augment existing API deployments. A significant trend is the use of machine learning to analyze the vast amounts of log data and to connect with API management solutions. This will help security analysts discover and stop threats before they become incidents. Organizations also need to invest in the internal capacity to engage with API service providers, leverage their knowledge, and feed them best practices in API security to prevent future incidents.

I am also optimistic about the API lifecycle focus that API service providers are using for API delivery. These providers use API definitions to map the landscape, support API deployment in any infrastructure, and deliver continuous deployment workflows. Many elements are being containerized and decoupled with security considerations managed across the major tools and services.

It is clear that the winners in 2020 will be the companies that are proactive about API security. It is not the time to be ignoring threats. The market for data is lucrative right now, making the incentive for hacking attractive—something you don't want to ignore.

Global Headquarters
3307 Hillview Avenue
Palo Alto, CA 94304
+1 650-846-1000 TEL
+1 800-420-8450
+1 650-846-1005 FAX
www.tibco.com

16Sep2019