STRENGTHEN YOUR API SECURITY

Improve Cyber Resilience and Compliance with Identity-aware APIs, Deep API Visibility and AI-powered Threat Detection

WHITE PAPER

TABLE OF CONTENTS

INTRODUCTION

Whether you provide application-based services to consumers, partners, employees or others, APIs have become the most common way of exchanging information between users and your services. Providing the foundation for enterprise modernization initiatives, APIs facilitate:

- Creation of digital revenue streams
- Development of successful partnerships
- Efficiency of internal operations

Given the critical role they play in digital transformation—and the access to internal sensitive data and systems they provide—APIs warrant a dedicated approach to security and compliance. Already an attractive target for bad actors, APIs are predicted to soon become the top attack vector. And while APIs may hold the key to compliance with privacy regulations like the General Data Protection Regulation (GDPR), they also expose potential vulnerabilities.

According to GDPR, personal data is the property of the individual, not the application or service providers. But even in Europe, where GDPR is having the most severe impact, only 60% of organizations reported having systems in place and operational to fully comply days before the deadline. This highlights several areas of concern around API security, including access control, data governance and consent management, visibility into the activity of your applications, auditing and forensics.

In addition to privacy regulations, regulations like the EU's Revised Payment Service Directive (PSD2) are requiring banks and other financial institutions to share data more freely than ever and offer their services via APIs. In the UK, the definition of standard API specifications and security profiles is provided by Open Banking. Other regions are borrowing heavily from these efforts, including Australia. Their Consumer Data Right (CDR) regulation extends beyond the banking sector but also requires financial institutions to share data more freely via APIs.

These new APIs are creating increased security risk by opening up a new attack vector for hackers. Beyond just the potential leak of information, regulations like PSD2 and those impacting the financial sector introduce even more complications to include fraud, identity theft, terrorism financing and money laundering.

Banks affected by these regulations need to enforce that access to data is authorized only when necessary for providing the agreed services and only with the parties designated by users for their financial services. Meanwhile, this dynamically changing matrix of consent and permissions across third parties needs to be implemented across multiple services.

While the financial service industry has felt the brunt of these regulations, no industry can escape the need for better privacy and more openness whether through regulation or competitive market pressures. Addressing this increasing demand is a core security challenge and the objective of many digital transformation strategies. And you achieve it by increasing your cyber resilience at the API layer.

As APIs attract increased attention across the spectrum from bad actors to data privacy advocates, forward-thinking enterprise security professionals are making API security a top priority. Read on to gain a deeper understanding of API security threats and vulnerabilities and how to build a solid defense.

25%	of companies have over 1,000 APIs, while 35 percent report having between 400–1,000.	**45%**	of security and IT professionals aren't confident in their security team's ability to detect API threats.
51%	aren't sure if their security team knows about all the APIs in their enterprise.	**30%**	do not know if their enterprise has experienced an API-related breach, leak, or security event.

The world is finally coming to the realization that the explosive growth of the API economy has ushered in a whole range of new security implications.

- KuppingerCole

API BREACHES: WHO WILL BE NEXT?

API breaches are increasing, not just in volume but in visibility. Several household names have fallen prey to API vulnerabilities, demonstrating that even those with large budgets and big security teams aren't immune.

In mid-2018, news broke that Venmo unintentionally leaked hundreds of thousands of transaction details. Shortly thereafter, T-Mobile reported that they'd leaked the sensitive data of 2.3 million customers. Then, Facebook had more than 30 million user accounts compromised, while the USPS breach exposed data on some 60 million accounts.

In 2019, the largest property valuation firm in Australia, LandMark White, had an API breach that resulted in leaking property valuation details and customer information. In this case, an API originally intended for internal use only eventually became accessible from outside their domain. In the wake of the breach, the firm's high-profile reputation was damaged, customers and banking partners scrambled to look for alternatives, and their CEO resigned.

Others who've made headlines include Verizon/LocationSmart, Snapchat, Instagram, Panera Bread and PF Chang's. The vulnerabilities resulted in breaches of varying severity, including account takeovers, theft of private information and photos, even the extraction of credit card numbers.

All of these breaches took weeks or months to detect. In the Facebook case, it took over a year for the breach to come to light. And the grim reality is that there are many API attacks in progress right now that are as yet undetected. To understand how to protect against these threats, you must first understand the underlying vulnerabilities that APIs present.

UNDERSTANDING API VULNERABILITIES

Many enterprises—both large and small—are currently relying on inadequate security measures to protect their APIs. While API management tools provide an important set of security features—including authentication and rate limiting—these practices often stop short at stopping attacks that are built specifically to breach APIs and the data and systems to which they provide access.

> **As the growing number of advanced attacks and data breaches caused by insufficient API cybersecurity indicates, even the largest companies with big security budgets are not safe from these kinds of attacks.**
>
> **- KuppingerCole**

Incomplete or Missing Validations

Missing entitlement checks are a recurring vulnerability pattern in a number of recent API breaches. These missing entitlement checks are leaving vulnerabilities exposed in production APIs. In some cases, a complete lack of access control has left APIs wide open, making it possible for bad actors with basic skills to gain access for malicious purposes.

In the cases of Facebook, USPS and Verizon/LocationSmart, the hacker used a valid account to reverse engineer the API behavior to identify at least one vulnerability that would provide access to data from another account without the proper credential check—all while looking like a normal user. This technique has the potential to deliver access to a large number of accounts and has been used successfully to breach banks and insurance companies.

Such vulnerabilities are not exposed when using the application calling the API. The malicious access is gained by skipping the client-side app (a web app, for example) and calling the API directly to observe data and control flows. Client-side apps greatly limit the way that an API can be used through user interface restrictions. Relying on the app can create security blind spots, particularly when testing is not performed outside of the app, at the API layer.

In addition to access control, API security must also include the implementation of content validation. This lack of security maturity is evident in the case of the Kubernetes' API server which was discovered (and patched) in early 2019. Users that were authorized to make patch requests to the Kubernetes API server could also send a specially crafted patch which consumed excessive resources during processing, incidentally (or purposefully) causing a denial of service (DoS) attack on the API server.

This type of vulnerability is leveraged to dramatically disrupt a service. The Kubernetes API server fix includes returning 413-type errors if the incoming JSON patch contains more than 10,000 operations. This type of content validation is easily configured in API gateways, yet it's often missed because doing so requires going beyond auto-generated JSON schemas that only define simple rule types and require human intervention to recognize the need for specific validations and ensure proper configuration and testing.

Proliferation & Lack of Visibility

The proliferation of APIs only adds to their vulnerability. APIs are being deployed faster than ever and by many different teams. In some cases, the ongoing pressure to innovate, to reduce friction and to create new revenue streams has the unintended effect of opening cracks through which APIs can fall.

It's not surprising then that many stakeholders report a lack of visibility across all APIs deployed by their organization. The fact that many API-related breaches have gone undetected for months—and sometimes years—further illustrates this lack of visibility on overall API traffic.

Furthermore, some APIs are not meant to be public and may be considered little more than implementation details of an overarching project. This keeps them hidden from the perspective of security practitioners and, in turn, leads to a lack of specific security considerations. In other cases, APIs emerging from different parts of an organization may leverage heterogeneous platforms and inconsistent security policies.

Regardless, these APIs constitute as much of an attack vector as their public counterparts because they are just as susceptible to being reverse-engineered by hackers to expose vulnerabilities. To close these gaps, you need a clear understanding of what needs to be secured. Deep insights into your API traffic provides the starting point for improved cyber resilience.

CLOSING THE GAPS IN API SECURITY

Organizations concerned with API security can learn from those who've come before them. The vulnerabilities exposed by high-profile data breaches can also help you identify security deficiencies in your own API infrastructures, many of which are rooted in misconfiguring API access management and in vulnerabilities left during the design and implementation of the APIs that remain hidden.

To adequately protect against increasingly sophisticated cyber attacks, you need a layered approach to API security that goes beyond the basics of your current web access security protocol.

Unifying Web and API Access Management

Many organizations implemented web access management (WAM) solutions long before adopting an API-first strategy. This led to separate but overlapping areas of responsibilities between web and API-based access management.

Even when provisioning web and API access control solutions from a single vendor, the resulting architecture is often redundant and inefficient. A typical framework might include:

- A fragmented solution with access control policies that must be duplicated between systems.
- Multiple components that act as authorization servers, but in uncoordinated ways.
- A duplication of OAuth client information between the IAM and an API developer portal.
- Disparate authentication requirements for the same users across different channels, which hurts the user experience.

Moving forward, it doesn't make sense to separate web and API access control. A modern identity solution can play the role of authorization server by issuing tokens and cookies in different formats and converting between standards like JWT and SAML on demand.

Today's web apps implement single-page applications in which APIs are called from the browser, native mobile apps call APIs directly, and devices and sensors feed IoT APIs. When all channels increasingly flow through the API, access control applied at that level becomes a unified security layer which maximizes enterprise-wide governance while minimizing friction for a better end user experience.

Going Beyond Token Validation

Validating a token and authenticating a requesting user's identity is often not sufficient for API infrastructure protection. The ability to apply granular access controls at the API layer is not only logical, it will become the norm as APIs become the most common channel for accessing the data you need to govern.

Furthermore, rules that define what data should be allowed by API requesters are not only defined by the API provider but also by the end users of your service. During OAuth handshakes, for example, we ask end users to make consent decisions. These decisions allow users to limit the ways in which applications act on their behalf when it comes to data they own. The management of user consent is therefore tightly connected with core API security concepts.

At the heart of foundational API security preparedness is the definition of review and governance processes which brings all of these concepts together. New or updated APIs must go through a review which starts with identifying answers to questions including:

- What permissions are required to access the API?
- Who are the expected requesters?
- What databases and data will this service leverage to read and write?
- What other services will this API interact with?
- What do input and output parameters look like and how should they be restricted?

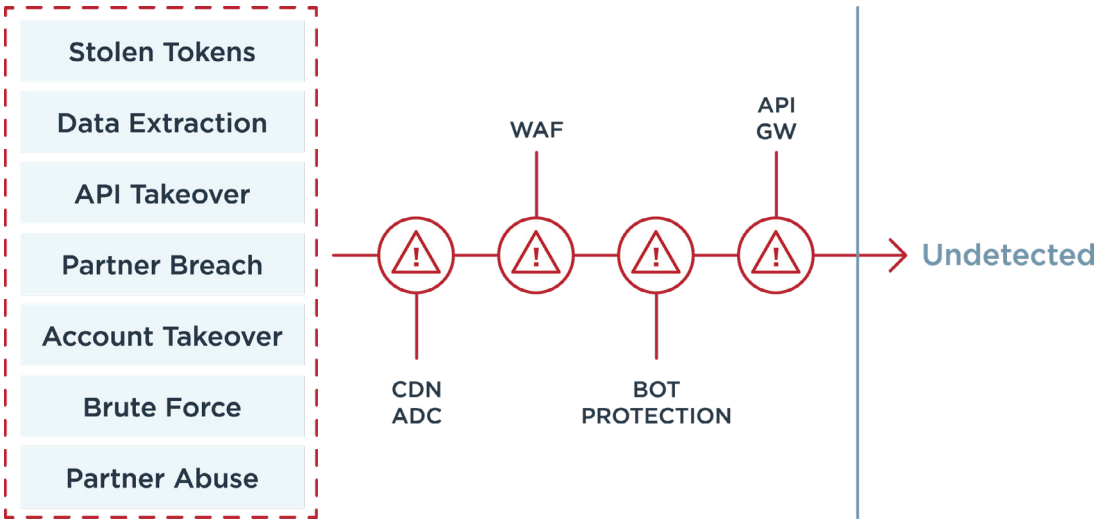
Answers to these questions will inform which security policies should apply to the new API being published.

EXTENDING BEYOND FOUNDATIONAL API SECURITY

Even with the right foundation in place, your security is only as strong as its configuration. But when configuration is subject to human error, the risk of an API vulnerability making it past testing and into production persists as the complexity of the API layer increases. Even if you've done everything right to prevent a breach and meticulously tested at the API layer, you can still be at risk of vulnerabilities exposed on a different vector beside the API itself.

For example, hackers often steal tokens through phishing attacks that allow them to pose as legitimate applications. Additionally, the client applications that call APIs on behalf of users also have weaknesses and are notoriously bad at keeping secrets. This can be as simple as API keys that are reversed-engineered from looking at a single-page application's JavaScript code or looking at API traffic through an HTTPS proxy. Case in point, a study from North Carolina State University revealed that GitHub is a gold mine of application secrets used to call APIs. The study concluded that over 100,000 repositories had leaked API tokens and cryptographic keys. GitHub is acting on this finding by implementing a new security feature to scan tokens from code submitted to its platform.

Attackers also use remote access trojans to steal credentials using tools like Mimikatz. They then use the stolen credentials to impersonate users and obtain tokens. Finally, another common vulnerability is credential stuffing on an authorization server leveraging collections of credentials mined from previous breaches.



Finely tuned attacks on APIs are bypassing traditional security measures provided by CDNs, WAFs and API gateways to steal data.

In each of these scenarios, an attacker obtains a token by exploiting vulnerabilities on users, client-side applications or authorization servers—not the API itself. This presents a number of issues as these tokens appear to be legitimate, and the hackers look like valid users, setting the stage for user data "leaking through the API." Even with proper access control policies and runtime enforcement, these attacks—plus those involving vulnerabilities with APIs themselves—will go undetected by traditional API security tools, often for months and sometimes years.

You could invest a tremendous amount of energy in educating users and API developers about client-app security and identity infrastructure, but you'll still be at risk of vulnerabilities in production APIs and the threat of compromised credentials. API providers must account for this in their security measures whether the APIs are internally or externally facing. You can do this by applying artificial intelligence (AI) to both accelerate attack detection and block attacks automatically.

APPLYING AI TO API SECURITY

Leveraging API traffic, you can train a machine learning engine on the client and user behavior normally exhibited, as well as on the behavior of the API itself. And by using AI, you can identify good and bad traffic, and recognize attempted attacks and ongoing breaches—without manual intervention.

Detecting Atypical Behavior with AI

Metadata for each API call, as well as access tokens or cookies and the timing and sequence of certain actions can all feed into the AI model. At runtime, this machine learning engine leverages the behavioral model to identify threats, recognize whether or not a particular token or cookie is being used outside of a legitimate application, whether data is being leaked or changed, find DDoS attacks on APIs that were undetected by traditional tools, and more. In other words, API traffic from a malicious party stands out from the normal API behavior baseline.

Leveraging AI-based detection of abnormal use accelerates threat visibility dramatically, shifting attack discovery from months to minutes or seconds. Once the suspicious API activity is detected, the access token or cookie used to obtain access to the API can be blacklisted or revoked, instantly stopping access from any party using that token across all API endpoints. If the same user identity is behind these attacks, then the user identity itself must be blacklisted and flagged accordingly. This type of monitoring, attack detection and blocking is realized without a human operator having to define custom rules.

Using API Decoys

The use of API decoys, or honeypots, can also accelerate the detection of API hacking. This involves adding fake API resources that return seemingly valid responses to a requester. Decoys leverage a hacker's tendency to poke around in ways that legitimate applications do not, effectively turning the tables to take advantage of knowledge surrounding typical hacking behavior.

When hackers fall into these traps, they are instantly recognized. At the same time, their associated IP addresses and access tokens, if they have them at the time, are automatically deemed compromised and blacklisted. The decoy listener is able to recognize that these requests cannot possibly be incoming from genuine applications since those API resources don't legitimately exist. These security measures need no custom rules or extensive configurations.

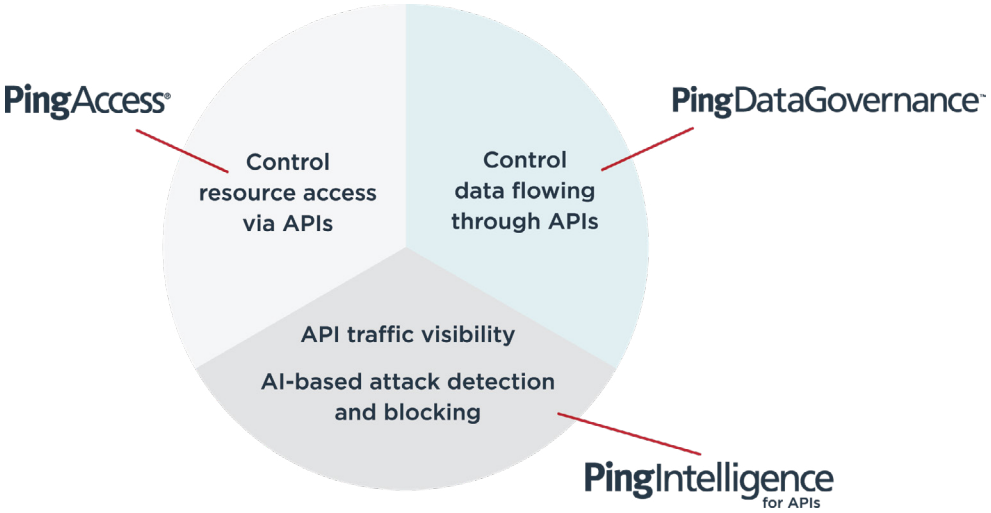
Beyond Detection: Blocking & Repairing

Of course, detection is only part of the solution. How the system reacts is equally important. Once the revelation that a client identifier (e.g., user or token) has been compromised occurs, it must promptly be revoked and/or blacklisted. This information needs to propagate widely across all API enforcement points, API gateways and agents. Additionally, these events must be recorded for subsequent auditing, and the Security Information and Event Management (SIEM) system must be notified to initiate any standard response processes.

Details surrounding the methods and resources invoked prior to the detection and blocking of the compromised credentials should be logged. Such information in the form of forensic reports allows an API provider, a DevOps team or a security team to take required actions to repair or reverse damage resulting from the attack. Forensics are also leveraged to help comply with regulations such as GDPR, PSD2 and Open Banking.

SECURING YOUR APIS WITH PING IDENTITY

The Ping Intelligent Identity Platform provides a suite of solutions to address foundational API security and compliance enforcement specifically designed for API traffic.



PingAccess

PingAccess controls access to your enterprise's internal and public-facing APIs. It is deployed at the perimeter of a protected network as a reverse-proxy, or as an agent running alongside your API endpoints. PingAccess supports granular API access control, limiting users to API transactions permitted by the authorization scopes contained in their access tokens.

PingDataGovernance

PingDataGovernance enforces fine-grained authorization to APIs, and the data flowing out through your APIs, by enabling policies that evaluate the body of API request and response. On API requests, fine-grained authorization policies can limit what a user can do in an otherwise authorized API call (e.g., if they're authorizing a payment that exceeds a certain limit, modifying a high volume of data entries, etc.). On API responses, PingDataGovernance examines the outbound data against policy and consent records for unauthorized, unintended, sensitive or restricted data that should be dynamically modified or removed from the response prior to release to the client.

PingIntelligence for APIs

PingIntelligence for APIs is an award-winning software solution that provides deep visibility into API traffic, discovers APIs automatically, provides AI-powered automated API attack detection and blocking, and uses deception/honeypot environments to identify hacking in real time. Its artificial intelligence engine brings cyberattack protection and deep insight into API activity to PingAccess, API gateways and API management platforms, as well as APIs implemented directly on App Servers, such as Node.JS, WebLogic, Tomcat or WebSphere. To learn more, download the technical solution guide.

PingIntelligence for APIs	• API traffic visibility • AI-based attack detection and blocking
PingAccess®	• Control resource access via APIs
PingDataGovernance™	• Control data flowing through APIs
PingFederate®	• OAuth authorization server • Token mediation
PingID	• Strong authentication • MFA
PingDirectory™	• Identity store • User profiles, consent

PingFederate

API flows are secured with tokens that are issued to the applications calling your APIs. The OAuth authorization server is the API security architecture component responsible for this. PingFederate is the OAuth authorization server which orchestrates the issuing of tokens between applications, your users and your identity stores. PingFederate doubles down as a federation server that supports web single sign-on and the translation of tokens between formats. Known as the swiss army knife of identity, PingFederate lets you support the token workflow you need, leveraging the identity legacy that you have. Just as for tokens issued by PingOne, tokens issued by PingFederate are standards-based and can be consumed across your API endpoints.

PingID

As part of token issuing workflows, a user identity is authenticated. PingID allows for the strong authentication of users and works in conjunction with PingFederate to authenticate using adaptive policies and multiple factors like TouchID or device biometrics that ensure a secure, frictionless user experience. PingID lets you choose between different authentication methods such as push notifications, email, SMS and voice. PingID is also available as an SDK which allows the embedding of this superior user experience and enhanced security in the same mobile applications that are calling your APIs.

PingDirectory

PingDirectory stores user identities, profile data and consent records, arguably the most valuable information you have and a key target for attackers. PingDirectory allows all applications to easily access user data when they need it and encrypts it at every state—at rest, in motion and during replication. Providing dynamic consent controls, it safeguards against insider attacks by limiting admin access to user records and sending active and passive alerts to notify you of suspicious activity. PingDirectory works alongside PingDataGovernance to allow the recording and the enforcement of these privacy requirements for API providers.

PingIntelligence for APIs	• API traffic visibility • AI-based attack detection and blocking
PingAccess®	• Control resource access via APIs
PingDataGovernance™	• Control data flowing through APIs
PingOne for Customers	• OAuth authorization server • Strong authentication • MFA • Identity store • User profiles, consent

PingOne for Customers

PingOne for Customers is an all-inclusive SaaS solution for managing identity and access to applications. When you build modern applications for your customers, you shouldn't have to build identity services like registration, account recovery and MFA. Doing this yourself can take countless hours and potentially put the security of your application at risk. Whether you choose REST APIs or pre-built, customizable UIs, you can rapidly embed secure identity services into your app that allow your users to conveniently register, login and manage their data. PingOne includes an OAuth Authorization Service that issues tokens to your API-calling applications. Tokens issued by PingOne are based on JSON Web Token (JWT) and as such can easily be validated by any API endpoint such as API gateways, a service mesh or in your own application using standard libraries.

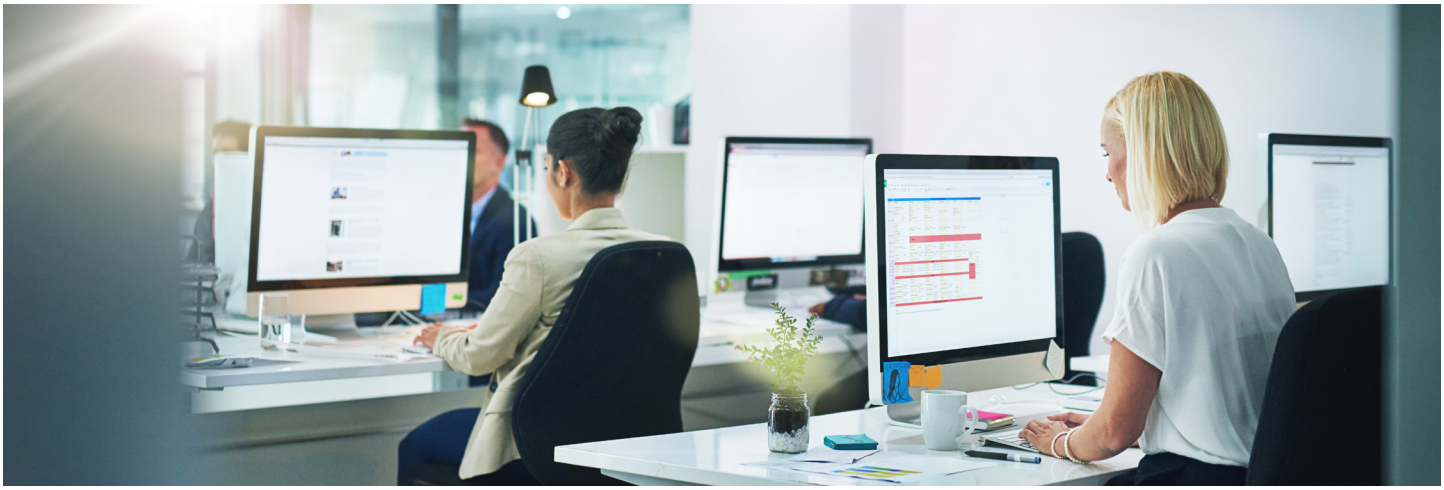
CONCLUSION

API security is a critical component of digital transformation initiatives and can improve cyber resiliency across channels.

In an era of Zero Trust, adopting an identity-centric API security infrastructure is core to a modern digital transformation strategy. To ensure the security of your APIs, you need both deep visibility into API activity and AI-powered breach detection. And both need to be layered over basic API access control to catch attacks that stem from API vulnerabilities and to remediate the risk of human error in the configuration of traditional API security systems.

To learn more about securing your APIs with PingIntelligence for APIs, read the technical solution guide.

Want to see PingIntelligence for APIs in action? Sign up for a free, no-obligation trial.



REFERENCES

Balaganski, Alexei, "The Dark Side of the API Economy," KuppingerCole, May 2019.

Global Forensic Data Analytics Survey 2018, EY.

Pearce, Rohan, "LandMark White Blames Exposed API for Data Breach," ComputerWorld, Feb 13, 2019.

Singh, Nitish, "Vulnerability Found in Venmo Public API Causing Massive Data Leak," TechNadu, July 18, 2018.

Cimpanu, Catalin, "Over 100,000 GitHub Repos Have Leaked API or Cryptographic Keys," ZDNet, Mar 21, 2019.

Hernandez, Rudy, and Brozek, Mark, "Security though Simplicity," Forrester, Dec 2018.