SANS

Analyst Program

Survey

2023 SANS Survey on API Security

Written by **John Pescatore**

July 2023

Akamai

Introduction

From its beginning, computing has involved continual movement from monolithic to distributed and layered systems. Computers went from mainframes to departmental to client/server to virtual machines to cloud computing. Networks went from point-to-point connections to layered physical networks to internet communications. Applications went from monolithic blocks of code to layered to distributed applications. See Figure 1.

This migration led to increases in performance and flexibility, but as the old saying goes, "There is no such thing as a free lunch." Those advantages came at the expense of additional complexity and, as the other old saying goes, "Complexity is the enemy of security." Distributed applications invariably increase both the attack surface available to malicious actors and the likelihood of vulnerabilities being built into production code.

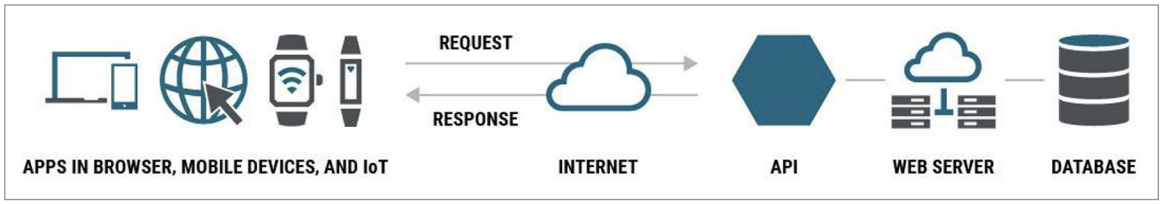

Figure 1. Evolution of Computing Migration (Source: Axway)

Modern applications use application programming interfaces (APIs) to define rules for how different elements should communicate with each other. In a distributor's catalog, for example, rather than having to continually modify one gigantic application every time a supplier is added or deleted, or their listing is changed, the distributor publishes APIs that define data flows for vendors to join, leave, update, and so on. These APIs essentially capture the business processes and break them into the lower-level communications required to efficiently enable business partners and customers to work with the business. A 2022 survey by 451 Group Research reported the average enterprise has more than 15,000 APIs in use.[1]

Like software developers, API writers are highly skilled at capturing legitimate business requirements and defining how legitimate business needs can be met efficiently. Modern APIs also must support a variety of computing platforms and user devices, which means that APIs are a threat surface that malicious actors may try to subvert, corrupt, or disrupt in unexpected ways. Most APIs get updated many times as attackers find vulnerabilities that will then need to be mitigated.

The most used standards for implementing APIs are Simple Object Access Protocol (SOAP) and Representational State Transfer (REST). SOAP is XML-based and incorporates WS-Security for encryption, digital signing, and authentication services. REST is HTML-based and uses HTTPS and JSON standards.

[1] S&P Global Market Intelligence, "The 2022 API Security Trends Report," https://nonamesecurity.com/resources/api-security-trends-report/

The bottom line is that API security, like application security, starts with:

- Inventory of APIs in use and processes that use those APIs
- Vulnerability assessment of APIs in use
- Threat assessment of active attacks exploiting those vulnerabilities
- Risk-based mitigation of critical API vulnerabilities

Although those security activities are well known, there are often gaps in knowledge, skills, and management prioritization in applying them to API security issues. The SANS API security survey was conducted to determine enterprise awareness, readiness, and future plans for dealing with API security risks.

Survey Results

In most publicly reported security incidents, the top three exploited vulnerabilities are generally:

1. Reusable privileged credentials obtained via phishing
2. Attackers exploiting misconfigured servers or cloud services
3. Exploitation of missing patches on servers and PCs

These same issues (weak authentication, misconfigured settings/positions, and failure to use latest versions) are vulnerabilities that are also exploited in attacks focusing on APIs.

Perceived Risks

Survey respondents ranked phishing and missing patches as the top two API security risks. See Figure 2. Of note, misconfigured servers/services were rated last in the weighted rankings, below exploiting vulnerable apps/APIs with zero-day (no patch available) attacks.

The weighted results from this same question show the following ranking:

1. Phishing to obtain reusable credentials
2. Attackers exploiting missing patches
3. Attackers exploiting vulnerable applications/APIs
4. Accidental disclosure of sensitive/covered information by users
5. Denial of service
6. Misconfiguration of servers/services by system administrators

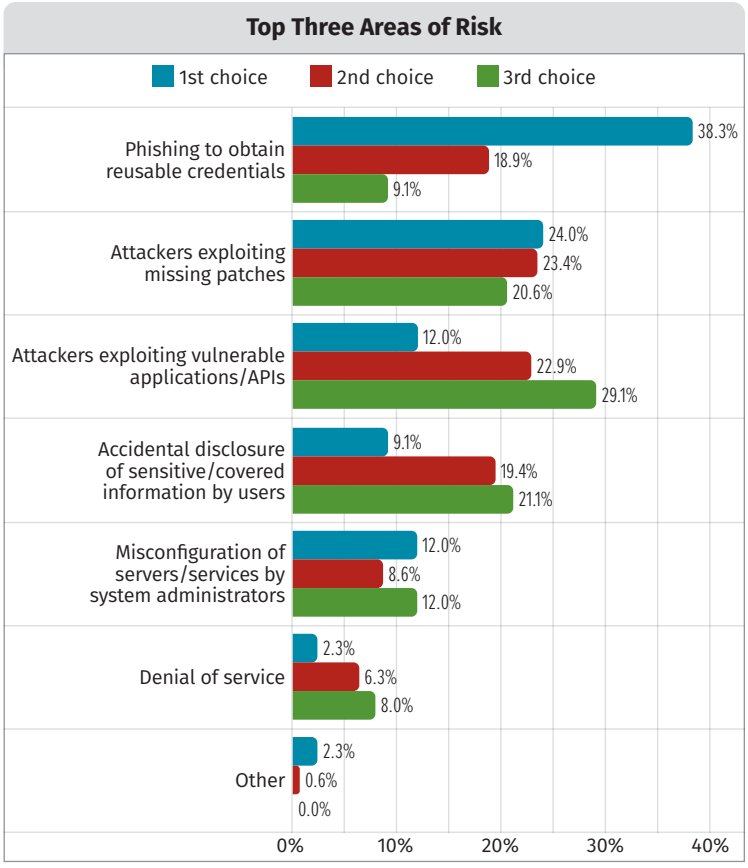

Figure 2. Top Three Areas of Risk

Takeaways

In the ranked weightings, when thinking about API security risks, respondents seemed to be underweighting the risk of misconfigured applications and overestimating zero-day risks. However, the same number of respondents chose misconfigured applications as their top risk as chose zero-day risks, indicating awareness of misconfiguration risks. Security managers should prioritize assuring that an accurate inventory of APIs is maintained, that updated versions of APIs are in use, and that configurations and options emphasize security.

Frameworks in Use

Cybersecurity frameworks provide a common language and reference model for determining the completeness of a security program, exposing gaps, and assessing risks. Mature security programs generally use full-coverage frameworks such as the Center for Internet Security Critical Security Controls or the NIST Cybersecurity framework.

More than half of respondents cited the Open Worldwide Application Security Project (OWASP)[2] Application Security and API Top Ten lists (Figure 3), and the MITRE ATT&CK Framework[3] as the basis for defining application and API risk. See Figure 4.

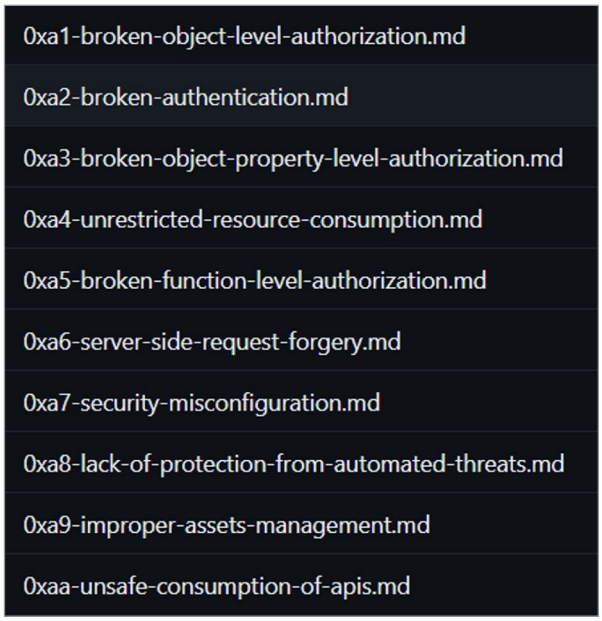

Figure 3. OWASP API Security Top 10 Vulnerabilities

Takeaway

The OWASP API Top 10 vulnerabilities and the MITRE ATT&CK model are powerful community-driven starting points for vulnerability assessment of APIs in use, assessing protection gaps and prioritizing action steps to mitigate API risks.

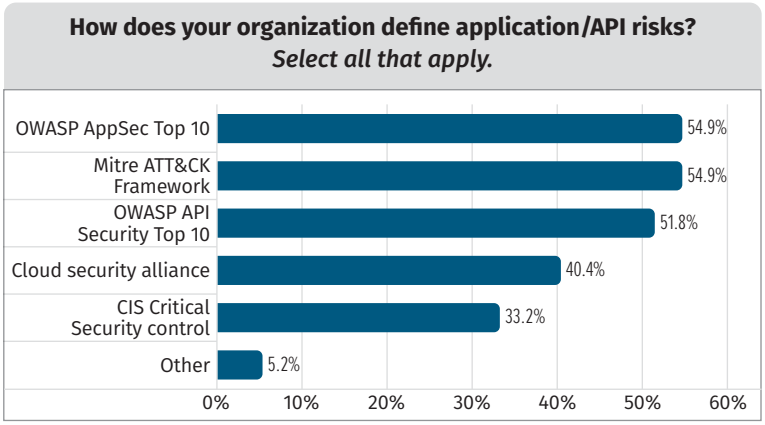

Figure 4. Frameworks Used to Define Application and API Risk

Tools/Controls in Use

Vulnerability assessment and management is a core component of every successful cybersecurity program. It requires a well-defined set of processes, including:

- **Discovery/inventory—**Knowing what systems, networks, resources, and applications are relied on for business operation
- **Vulnerability assessment and prioritization—**Determining if assets have vulnerabilities and their level of exposure and criticality
- **Remediation/mitigation—**Applying patches to or replacing vulnerable assets or shielding those that cannot be remediated

[2] OWASP is a nonprofit organization that has been leading community efforts to improve the security of applications and the accuracy and effectiveness of application security tools since 2001.

[3] MITRE, a nonprofit company that operates US federally funded research labs, started ATT&CK in 2013 to document the tactics, techniques, and procedures (TTPs) actively being used to compromise enterprise networks, systems, applications, and data. The MITRE ATT&CK framework is a widely used model for defining API threat models and assessing current and needed security posture against API threats.

General-purpose vulnerability discovery and assessment tools often do not provide visibility into API use and issues. Similarly, standard mitigation approaches such as denial of service, web security gateway, and application-level firewall products and services may not address API-level risks, though some web application firewalls can provide API-level protection.

Of those respondents that are using at least one tool, 23% are using tools or technologies across three of the areas shown in Figure 5.

Almost two-thirds of respondents are using web application firewalls (WAFs) as part of API risk mitigation, and more than half cited use of dynamic or static application security testing tools. Those three controls were also the most mature—more than 42% have been using WAFs for more than three years, while 38% and 35% have been using static and dynamic application testing tools, respectively. See Figure 6 for the full breakdown.

API discovery tools are used by 29% of respondents, but only 18% reported more than three years of experience using those tools; another 23% cited mature use of application-level discovery tools. API security features in DDoS and content delivery network/load balancing services were in use by less than one-third of respondents.

A wide variety of commercial products and open source software libraries are available today from larger application security vendors and smaller API security vendors. The ones currently in use by respondents are shown in Table 1 (on the next page) with four functional groupings as follows:

- **Discovery/inventory—**Inclusive of application inventory and application/API discovery
- **Security testing—**Inclusive of application vulnerability testing and API security testing
- **Firewall/gateway—**Including web application and application-level firewalls and web security gateways
- **Cloud-based services—**Including app/API in content delivery and cloud-based DoS

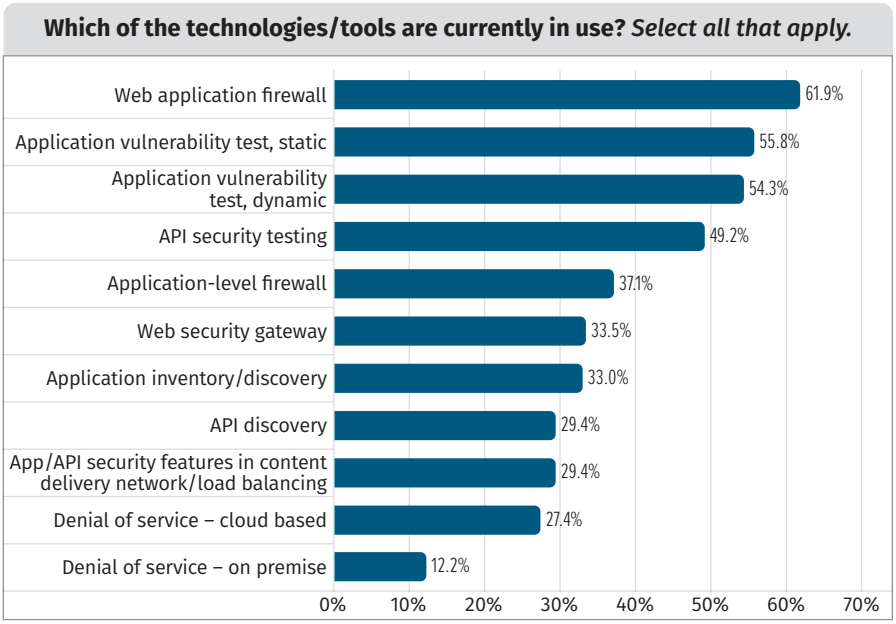

Figure 5. Technologies/Tools Currently in Use

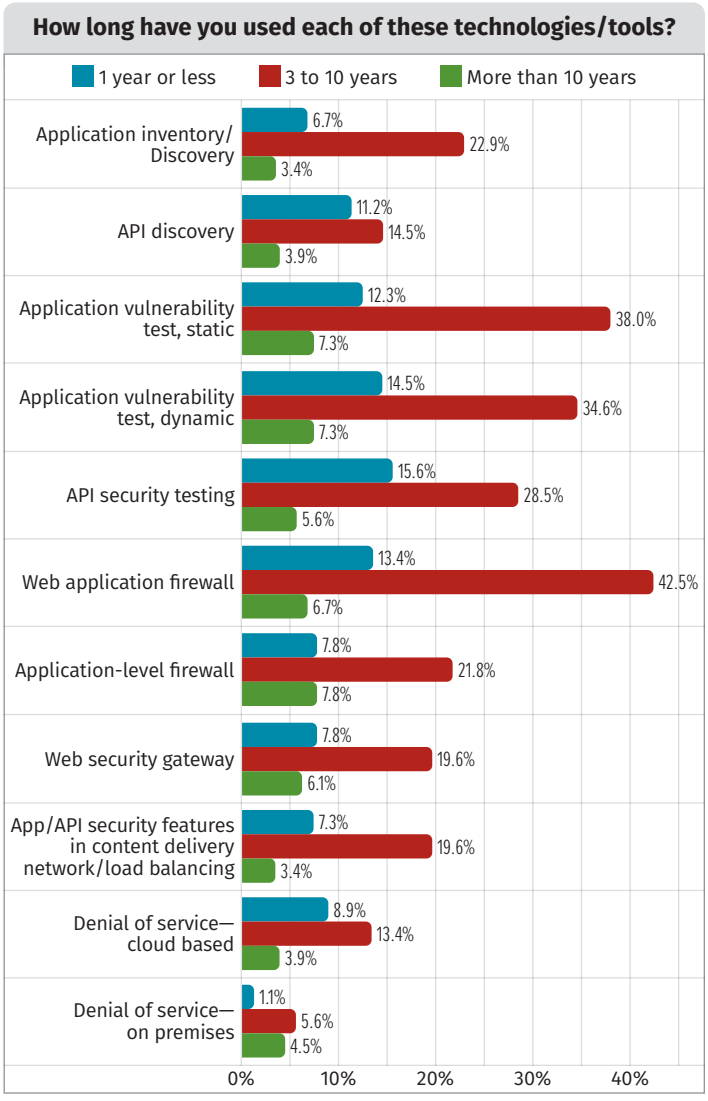

Figure 6. Length of Time Each Technology/Tool Has Been in Use

Takeaways

Discovering, assessing, and mitigating API risks is a complex problem that requires multiple security controls and processes to provide complete coverage. Application-level tools already in widespread use may provide partial coverage—API security-specific controls that provide full coverage are not yet in widespread use. An underutilized area is taking advantage of API security controls that are included in DDoS and load balancing services. Any use of tools from multiple vendors requires investing in training of analysts and integration of results from disparate tools.

Table 1. Functional Group/Products Used

Functional Group	Products Used
Discovery/Inventory	ADS, Akamai, APIsec, Checkmarx One, Cloud Asset Inventory, Internal Tool, Istio/GKE, ModSecurity, Postman, Streamline, Swagger, Tenable
Security testing	APIsec, AppScan, Burp Suite, Enterprise, Checkmarx One, CodeQL, Coverity, DAST, ESAPI WAF, FreeWAF, Gartner, HiHTTPS, ModSecurity, Nessus, NGAF, Nikto, Owasp Dependency- Check, Paros, Postman, Qualys, SAST, SmartBear ReadyAPI, Snyk, SoapUI, SonarQube, Snyk, Synopsys API Scanner, Tenable, Veracode, Wulian, Zap
Firewall/Gateway	Acunetix, Akamai, API, AWS Shield, AWS WAFv2, Azure WAF, Checkpoint, Cisco, Citrix ADC, Cloud Armor, Cloudflare, F5, Forcepoint, GKE Network Policy, Microsoft Azure, ModSecurity, Naxsi, NGAF, Norman Personal Firewall, pfSense, Secure Web Gateway, TCP-Wrappers, UniWSG, URL, Word Fence
Cloud-based services	Akamai, Amazon Lightsail, AWS, AWS Shield, Azure DoS protection (against excessive traffic, but not DOS caused from within the app), Cisco, Cloud Armor, Cloud CDN, Cloudflare, LINUX: RUDY (r-u-dead-yet), Uptime Robot

Visibility/Inventory Accuracy

One of the oldest sayings in cybersecurity is, "You can't protect it if you don't know it's there." An accurate inventory of networks, computers, and applications has long been considered the starting point for essential security hygiene, but enterprises have struggled for years to reach even 80% accuracy on asset inventories. Inventories of APIs face additional challenges; the average application uses three APIs and cloud-native apps (often more).[4]

Most (57.1%) respondents reported API inventory accuracy of between 25% and 75%, with 20% reporting under 25% and 23% reporting over 75%. See Figure 7 for specifics.

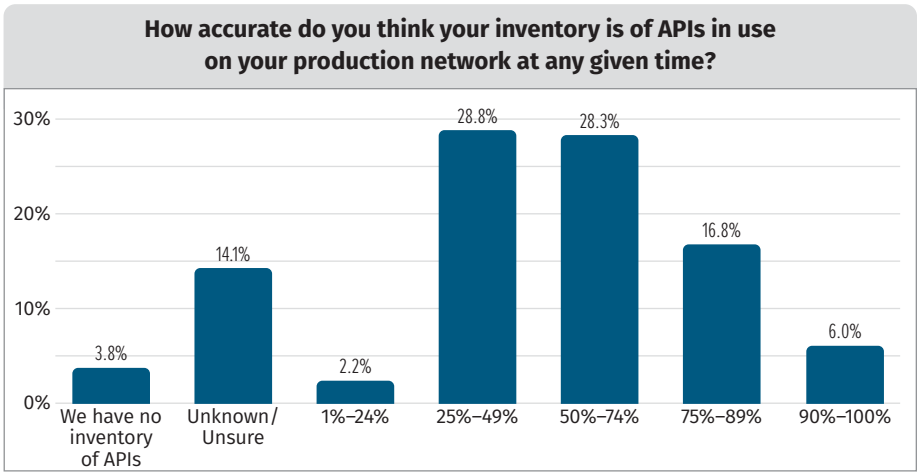

Figure 7. API Inventory Accuracy

Takeaway

API discovery/inventory accuracy should be at least as high as overall asset inventory accuracy. Because vulnerable APIs are becoming the most common access point for attacks, API inventory accuracy should increase and discovery should be performed more often.

[4] Nordic APIS, "APIs Have Taken Over Software Development," October 27, 2020, https://nordicapis.com/apis-have-taken-over-software-development/

Plans for the Future

As seen in Figure 8, companies are planning to close API security gaps in the future with the following four technologies and tools:

- Web security gateways (WSGs) (14%)
- API security features in content delivery network/load balancing (13%)
- Web application firewalls (13%)
- Dynamic application security testing (13%)

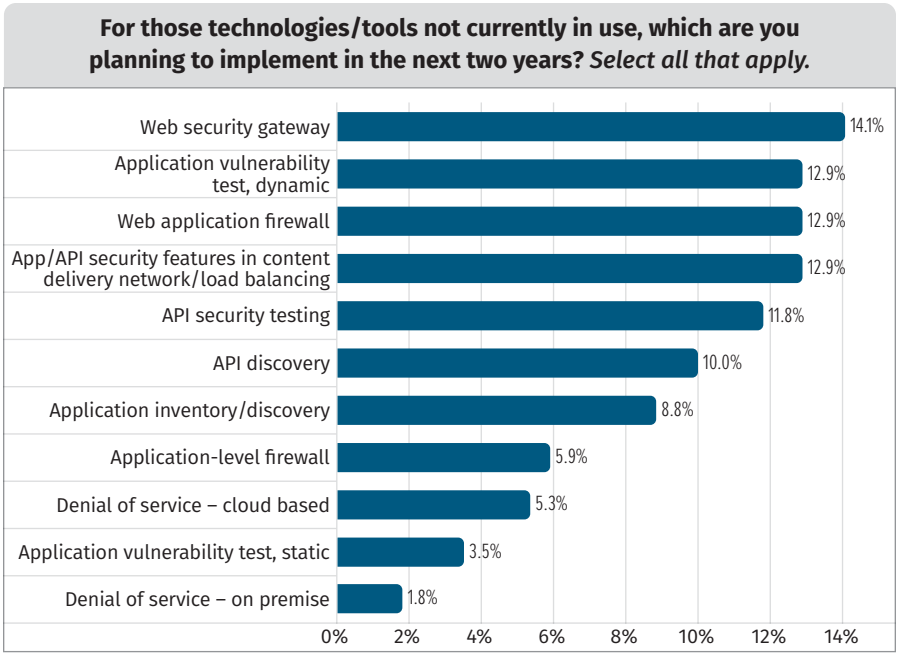

Figure 8. Tools to be Implemented

All these tools can be effectively used to increase API security, depending on where and how they are used in the overall security architecture. In particular, WSGs are often placed between user traffic and the internet and do not provide protection for server/cloud-based applications. Similarly, WAFs may only be in the path between the internet and server/cloud-based applications and not provide protection to user PCs against API attacks. Content delivery networks (CDNs) and load balancers often span both areas. Choosing a cloud- or CDN-agnostic solution for API discovery can help close many of these gaps.

Takeaway

Providing effective API security will require multiple tools from multiple vendors. To increase both effectiveness and efficiency, common vendors chosen from across multiple functions and services in existing services (such as CDNs/load balancers) should be looked at first.

Staff Training

The most effective approach to overall application security is to write and/or buy secure applications. The same is true for APIs. The best way to begin is to educate developers on secure coding practices and overall application security. More than 75% of respondents reported training development staff on application security. See Figure 9.

The bad news is that even trained developers make mistakes—writing and procuring applications and cloud services with vulnerable APIs. Security staff also need to be trained on application and API security issues. Training is not required for every security analyst or incident responder, but enough trained personnel must be available to support all functions and shifts.

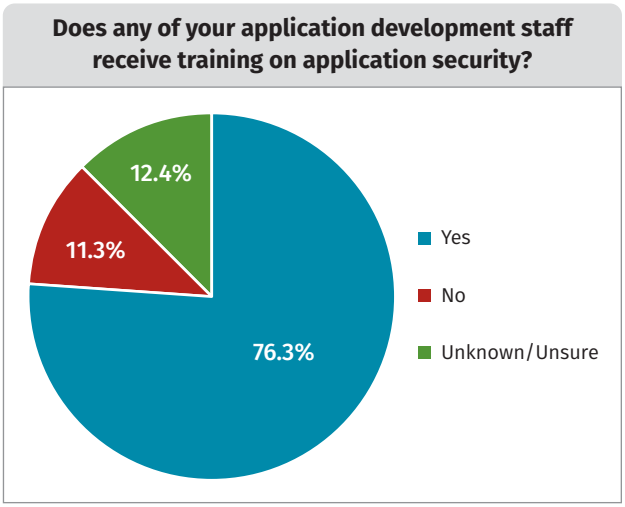

Figure 9. Application Security Training for Application Development Staff

The good news is that more than 70% of respondents (73%) reported training at least 25% of their security team on application security (see Figure 10). The overall IT and IT security architecture and technology choices will drive what is right for your organization.

Takeaway

Building security in is always the best approach. Convince management to invest in educating developers on API security. Some portion of your security staff should also receive application/API security training. (Between 25% and 75% of survey respondents received such training.)

Figure 10. Percentage of Staff Trained on Application Security

Demographics

Most of the 231 survey respondents conduct operations from or are headquartered in the United States, as shown in Figure 11.

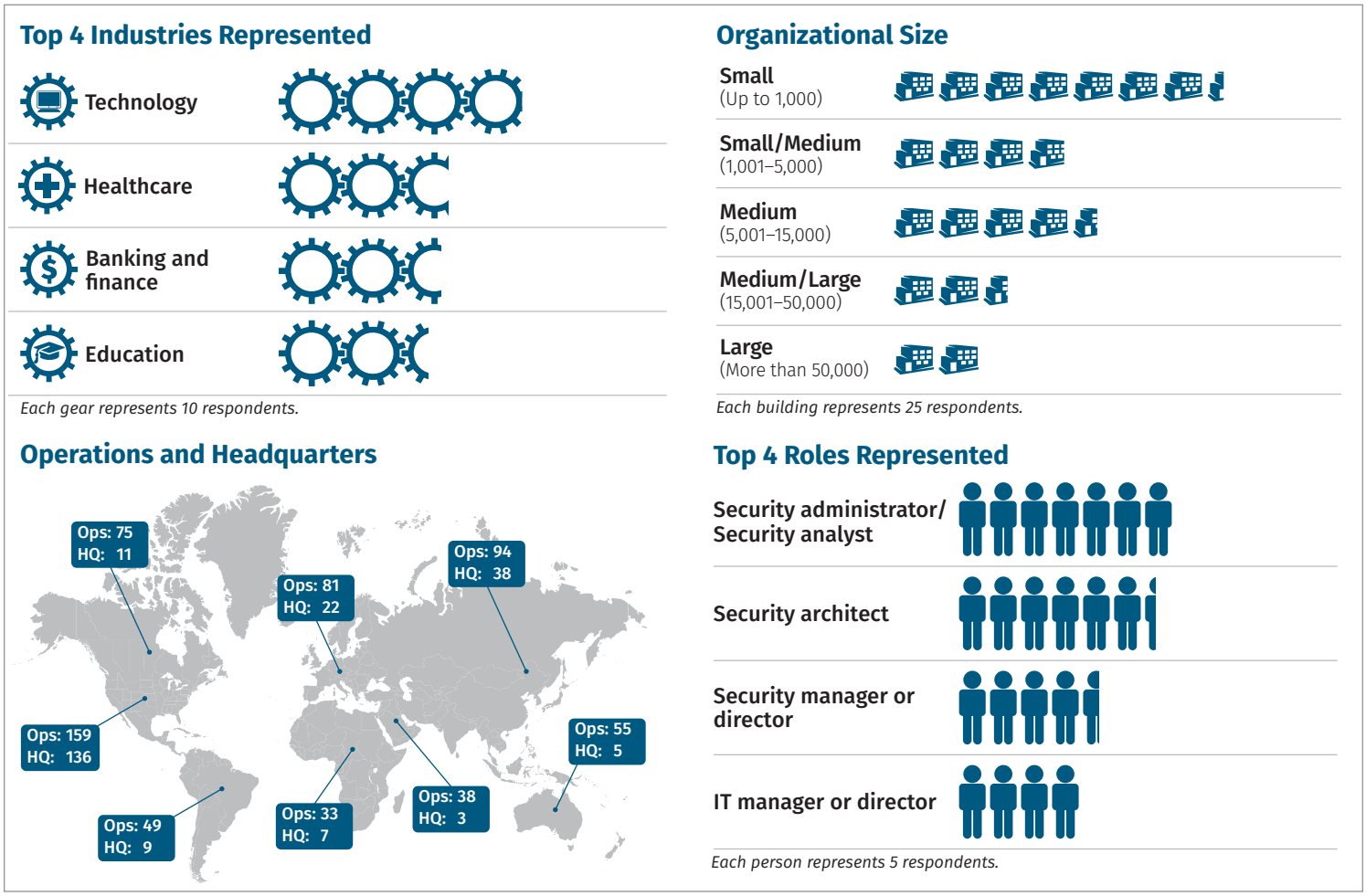

Figure 11. Demographics of Survey Respondents

Seventy-eight percent of respondents currently play a role in application security, with another 15% looking toward future involvement. The majority of these respondents (23%) are involved with testing applications for vulnerabilities before the application is placed into production, quickly followed by those (21%) who scan applications for vulnerabilities once they have been place in production. See Figure 12.

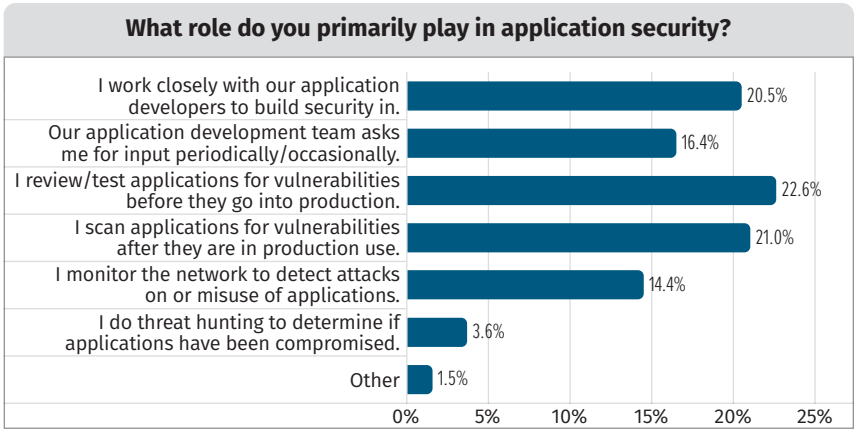

Figure 12. Roles of Respondents in Application Security

Results/Conclusions

API security is a complex area, requiring security leaders to upgrade/enhance many security processes at a time when budgets and staffing are under pressure. These factors put a premium on architectures and solutions that are both effective and efficient. Just adding spending on more layers of products and more staff is not feasible.

The key findings from this survey are:

- **Discovery and vulnerability assessment of APIs in use needs to be top priority—**Many APIs are already in use, and that number is increasing along with constant updates to the existing inventory. Techniques for discovery and classification of APIs need to advance analytics beyond whitelists and other static approaches.
- **Protection/mitigation solutions are needed—**Low levels of accuracy of API inventory mean vulnerable APIs may be attackable for long periods of time. Segmentation, shielding, and mitigation are needed until API development and deployment security practices are more mature.
- **To be both effective and efficient, API security controls need to span both user-to-application and application-to-application traffic—**Risks exist in both areas, and attackers will always find the low-hanging fruit. Applications may be authenticated and approved but should only be allowed for a subset of applications or servers.

Summary

Any new technology that can increase customer satisfaction, and ultimately revenue and profit, will be rapidly adopted by businesses—and quickly probed for weaknesses by criminals to enable malicious exploits. To be successful, businesses first have to be movers not only in adopting such technology, but also in adapting, extending, and improving security architectures and controls to mitigate new risk.

The application programming interfaces relied on by modern distributed applications are the latest example of this. The essential security hygiene controls of strong authentication, asset inventory, vulnerability management, and change control need to address API security issues. Prevention and detection need to be upgraded to deal with API-centric attacks, and infrastructure services (such as content delivery networks and denial of service filtering) need to be put to work, as well.

Sponsor

SANS would like to thank this paper's sponsor: