EXECUTIVE SUMMARY

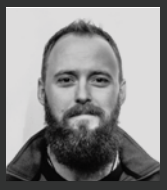
2019 APPLICATION PROTECTION REPORT

The Virtue of Visibility

F5 LABS

Authors

Ray Pompon is a Principal Threat Research Evangelist with F5 Labs. With over 20 years of experience in Internet security, he has worked closely with federal law enforcement in cyber-crime investigations. He was directly involved in several major intrusion cases, including the FBI undercover Flyhook operation and the NW Hospital botnet prosecution. He is the author of *IT Security Risk Control Management: An Audit Preparation Plan* published by Apress books.



Sander Vinberg is a Threat Research Evangelist for F5 Labs. He has worked in information security, geopolitical risk, and linguistic consulting. He holds a master's degree from the University of Washington in Information Management, as well as bachelor's degrees in History and African-American Studies from the University of Chicago.

Contributors

Sara Boddy | *Director, F5 Labs*

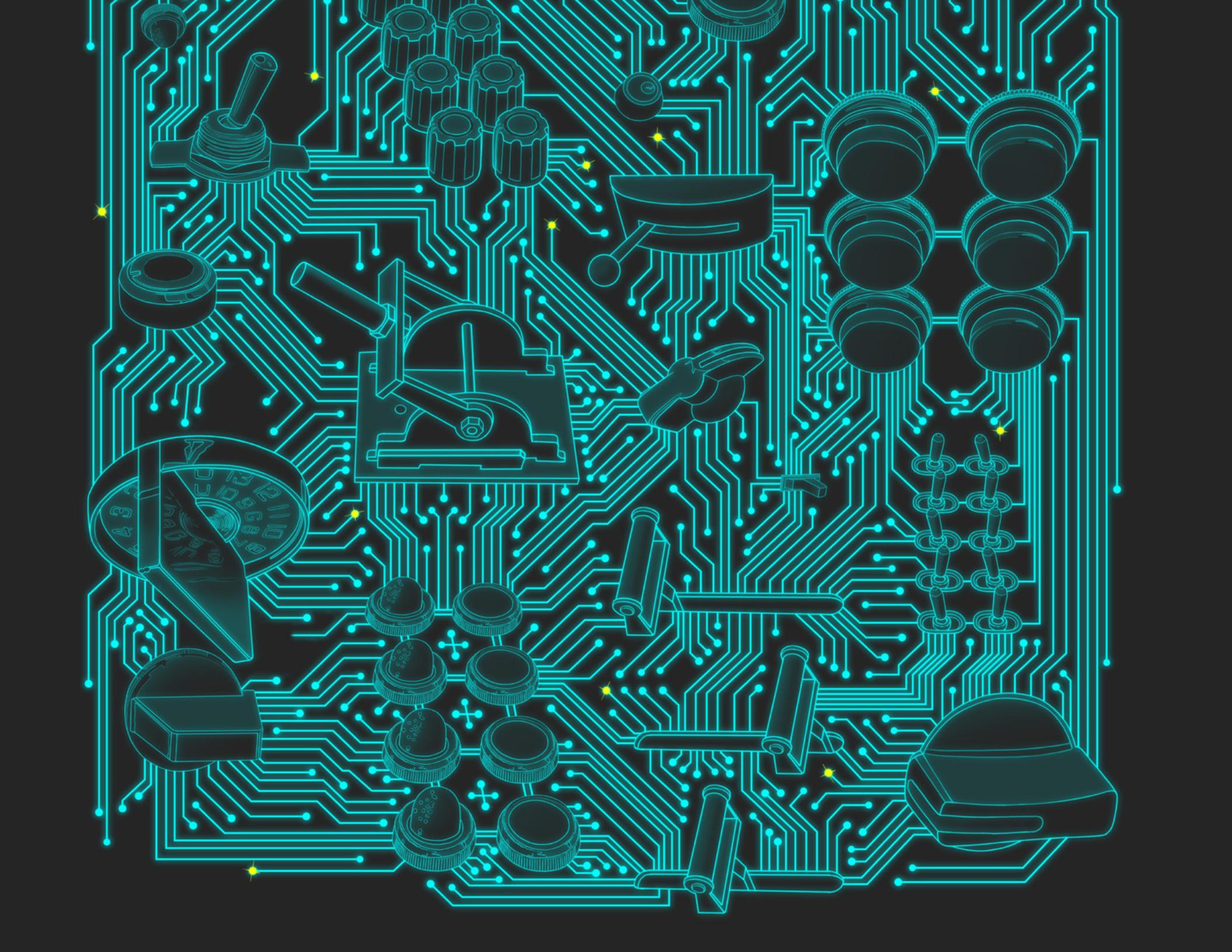
Debbie Walkowski | *Threat Research Evangelist, F5*

David Warburton | *Senior Threat Research Evangelist, F5*

Business and Data Partners

BAFFIN BAY NETWORKS is a team of dedicated researchers monitoring and investigating emerging attacks, advanced persistent threats, and the organizations and individuals responsible for them. They also develop research tools to identify, investigate, and track ongoing attacks and emerging threats. Working with Baffin Bay Networks, we analyzed global intrusion and honeypot data collected from web attacks on 21,010 unique networks over 2017.

CYENTIA INSTITUTE is a cybersecurity research services firm. They deliver high-integrity, high-quality, data-driven research which provides security companies with meaningful marketing content to build mindshare, drive sales, and attain greater visibility in competitive markets. In doing so, it seeks to advance cybersecurity knowledge and practice for the community at large.



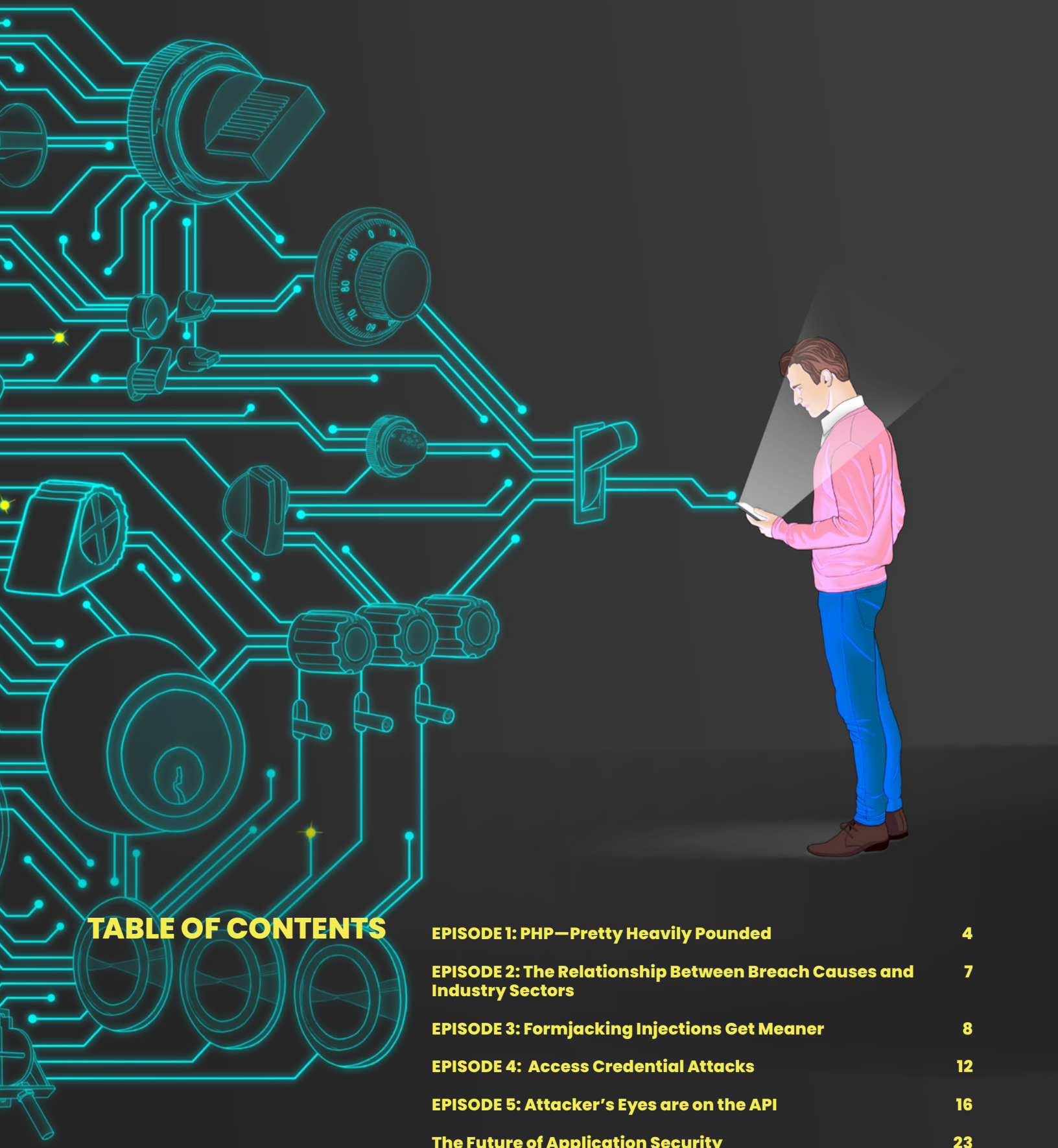


TABLE OF CONTENTS

Figure 1
(opposite page)
APPLICATION STRUCTURE AND THE ATTACKS AT EACH LAYER
To protect your apps, you need to understand how they'e structured and how they work—and the threats that target each layer.

Welcome

to the Executive Summary of the second annual F5 Labs Application Protection Report. This year, we have new and deeper insights from within F5, combined with threat intelligence from Baffin Bay Networks and its global network of over 1,500 sources. We've worked with the Cyentia Institute, the pioneers of security research who created Verizon's Data Breach Investigations Report. We have a lot of data-driven insights that we're excited to share!

As always, we focus on applications because that's what our adversaries do. Applications are the battlefield of information security. As the meeting point of users and networks, they are the defining value proposition for most businesses, and the gateway to that which attackers value most: data.

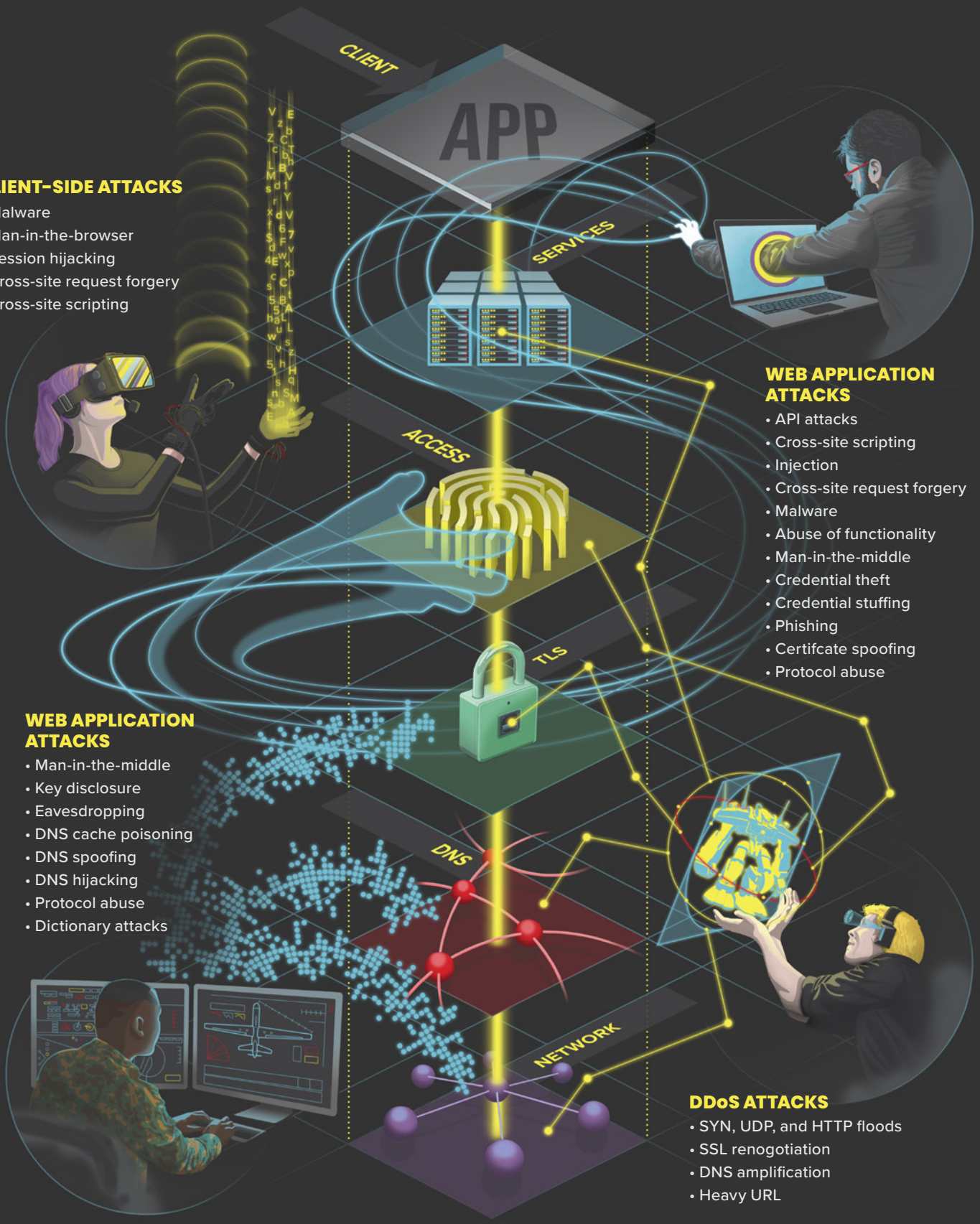
ANY WAY YOU MEASURE IT–BY PORT, BY BREACHES, BY COMPROMISED RECORDS–APPLICATIONS ARE THE NUMBER ONE TARGET ON THE INTERNET

Apps: Even More Like Colony Creatures

Over the course of the development of the 2018 report, F5 Labs created a model for understanding applications that captured the internal complexity and interdependence of modern applications, and illustrated how attack techniques can target completely disparate parts of an application in similar ways. We described the modern web application as a "colony creature," consisting of a multitude of separate, independent components that are glued together over networks.

In the year that followed, this pattern of decentralization has accelerated, leading to changes in how applications are both attacked and defended. Applications are more colony creatures than ever, and nobody understands that better than attackers. We hope that this report will help defenders understand what this trend means for them, as well.

APPLICATION STRUCTURE AND THE ATTACKS AT EACH LAYER



EPISODE 1

PHP—Pretty Heavily Pounded

Figure 2
(following page)

PMA CAMPAIGNS VERSUS DOMAIN-ONLY TRAFFIC

Coordinated campaigns targeting seven phpMyAdmin paths compared with traffic targeting web servers with no specified paths. Note that the data show a gap between March and June 2018 when Loryka's port 80 sensors went dark.

For the past two years, the server-side language PHP has jumped out as a highly targeted attack vector. In 2017, PHP was targeted in 58% of indiscriminate web attacks. In 2018, we saw this rise to 81%. PHP is widespread and powerful, and it's been used continuously in at least 80% of sites on the web since 2013.[1] In 2018, we observed that 42% of sensor traffic was aimed at paths or filenames associated with phpMyAdmin (also known as PMA), a PHP web application used for managing MySQL databases. Most of the PMA targeting looked for older systems from 2011-2013.

Traffic like this provides valuable intelligence because it is not going after any specific targets; it is looking for anything on the Internet that has the right vulnerabilities. This kind of opportunistic scanning tells us what the less sophisticated end of the attacker spectrum is looking for, and how they plan to attack it when they find it.

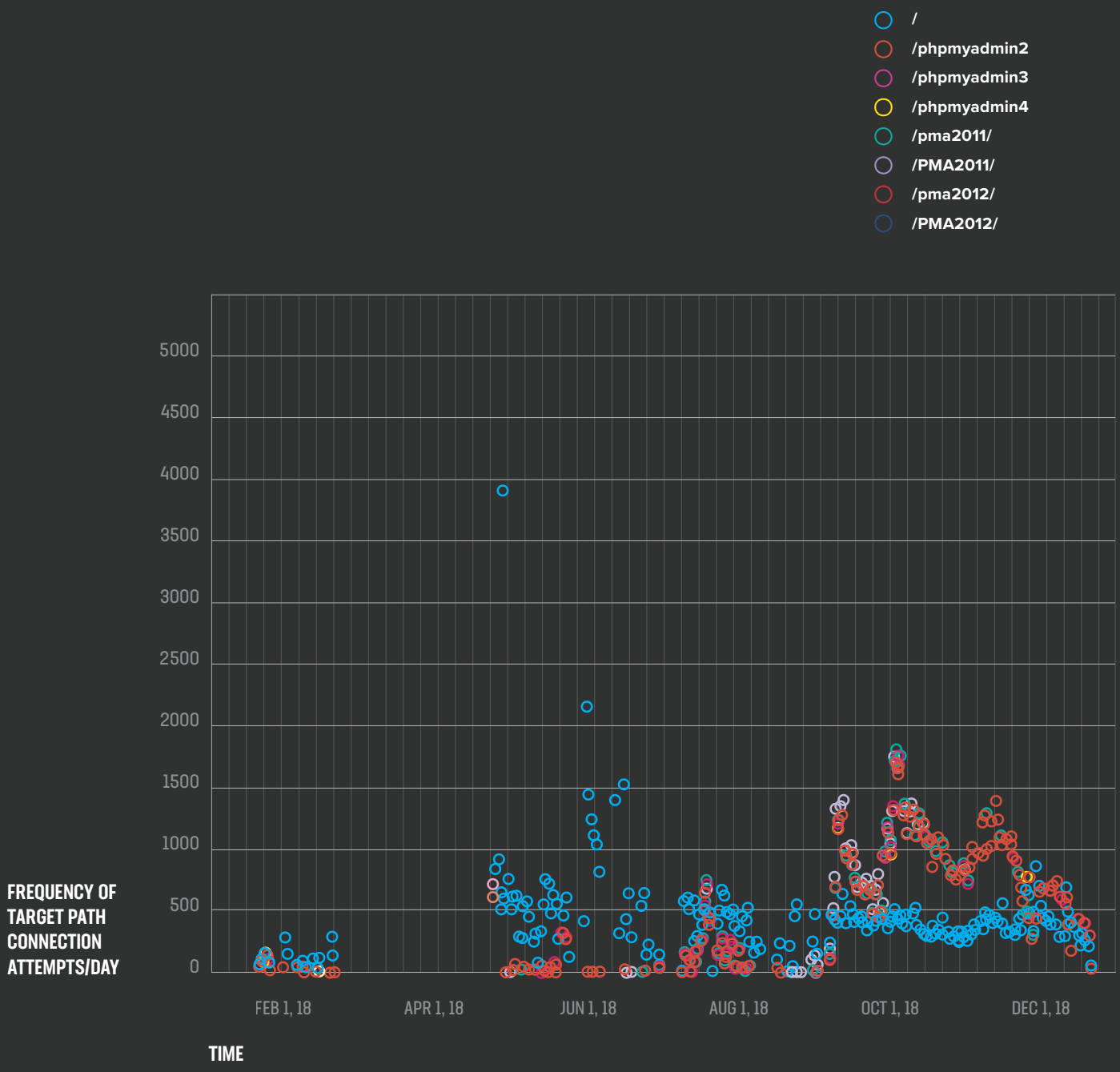
58% IN 2017, PHP WAS TARGETED IN 58% OF INDISCRIMINATE WEB ATTACKS.

When we dug deeper, we found that 87% of the traffic pointed at these common phpMyAdmin paths came from just two IP addresses out of more than 66,000 that hit Baffin Bay Networks' sensors. These two IP addresses, allocated to systems on a North American university campus, represented a huge proportion (37%) of the total attack traffic. Based on our analysis, it is likely that these threat actors were looking for poorly controlled authentication portals on old (and probably neglected) MySQL databases with weak authentication.

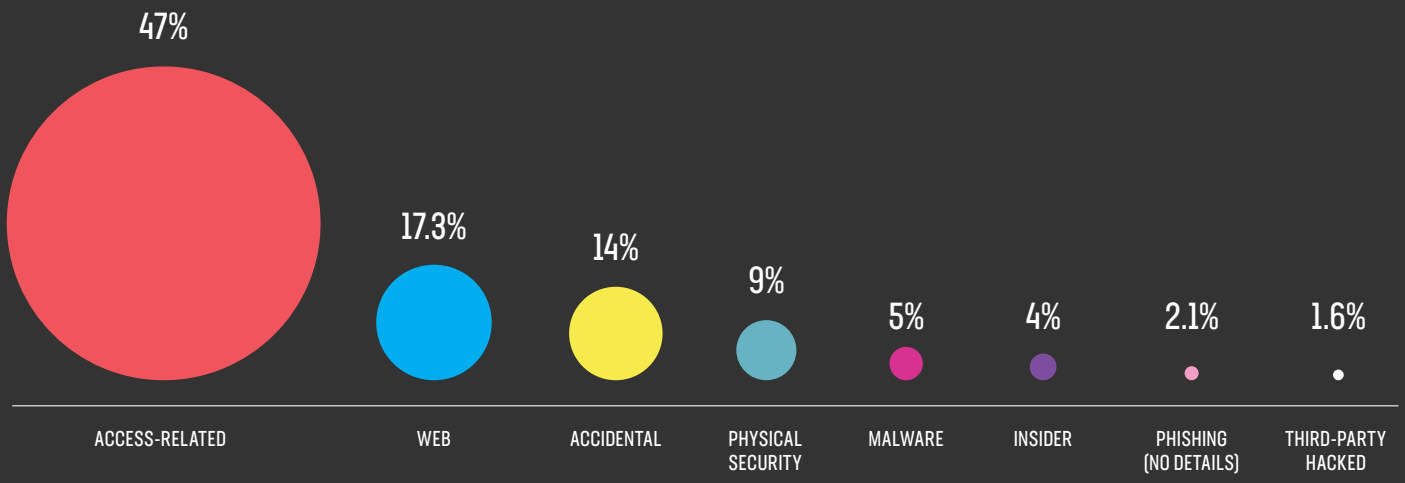
The simple takeaway is that if you're using PHP, you're being scanned for weaknesses. Make sure you're patched up, with a careful eye toward known PHP exploits like CVE-2018-12613 and CVE-2018-20062. And if you've got any PHP-enabled admin interfaces online, you need to lock them down tight.

The higher-level takeaway is that traffic like this is a reminder that old vulnerabilities never quite go away. It is easy to watch the threat landscape change and new critical vulnerabilities come and go. Reconnaissance campaigns that seek out systems with eight-year-old vulnerabilities demonstrate that we are always building on top of our past, and the new threats and vulnerabilities that come on the scene do not erase the old ones.

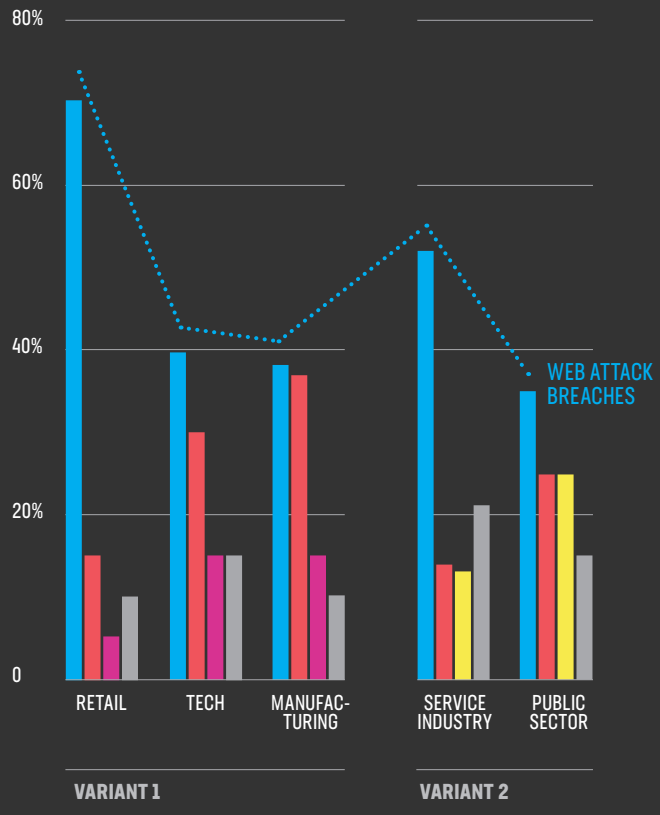
PMA CAMPAIGNS VERSUS DOMAIN-ONLY TRAFFIC



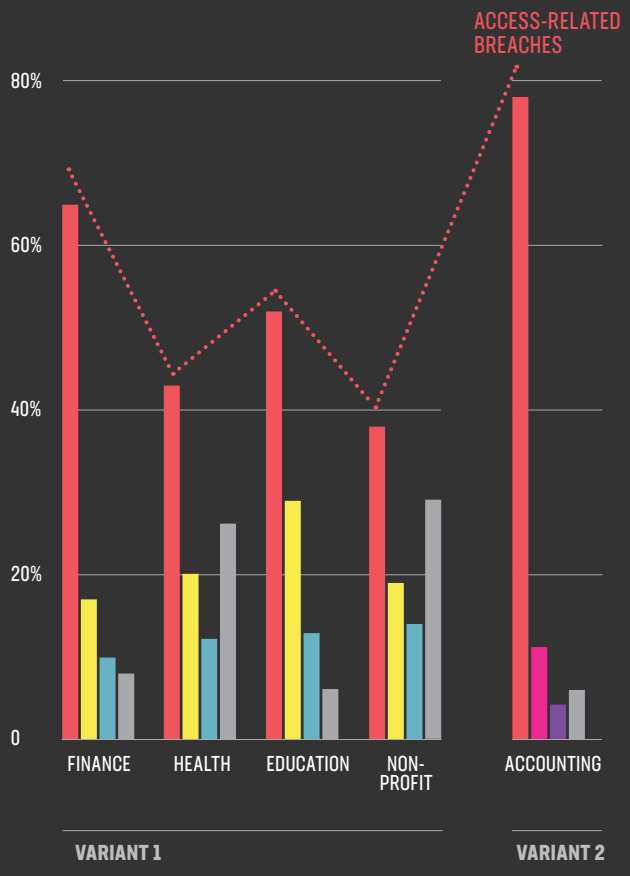
2018 U.S. BREACHES BY CAUSE



BREACH PROFILE: INJECTION



BREACH PROFILE: PHISHING



ACCESS WEB ACCIDENT PHYSICAL MALWARE INSIDER OTHER

EPISODE 2

The Relationship Between Breach Causes and Industry Sectors

Figure 3
(previous page)
2018 BREACHES BY CAUSE: U.S.
Distribution of causes of U.S. breaches in 2018, by breach count. The lack of detail in the breach reports means that there is partial overlap between many of these categories.

Most U.S. states require that victims be notified of data breaches, and some states' attorneys general publish these breach letters on their websites. For this year's report, we examined 761 breaches reported in 2018 across 10 states, representing 21.4% of the U.S. population: California, Washington, Wisconsin, Vermont, New Hampshire, Iowa, Maryland, Oregon, Idaho, and Delaware. Overall, we found that access-related breaches made up the largest proportion of the known breach causes at 47%. In addition, phishing was responsible for 21% of breaches with a known root cause, whereas injection for payment card skimming was responsible for about 12%. Furthermore, we found that certain types of organizations commonly experience the same kinds of breaches. The nature of the breach depends on how these different industry sectors tend to collect and store their valuable data assets.

Figure 4
(previous page)
BREACH PROFILE: INJECTION
Industries that were most likely to experience a data breach through injection, and the subtle variations in the historical likelihood of injection breaches versus other causes per sector.

E-commerce Payment Formjacking Injections

One of the profiles we identified was a pattern of industry sectors with a high rate of compromise through payment form injection. The retail sector, which relies heavily on e-commerce transactions, had a disproportionately high rate of compromise by injection, with 72% of attributable breaches. Similar industries, such as manufacturing and technology, also tended to be breached this way. The public sector also had a high incidence of successful injection attacks, probably due to the prevalence of the Click2gov exploit that haunted local government and utility sites in 2017 and 2018.

Figure 5
(previous page)
BREACH PROFILE: PHISHING
Industries that were most likely to experience a data breach through phishing, and the subtle variations in the historical likelihood of phishing breaches versus other causes.

Phishing and Email Theft

The other profile we identified centers on organizations in the finance, health, education, non-profit, and accounting sectors that were significantly more likely to be compromised through phishing or illicit email access.

In many of these cases, the breach notification letters mention how unauthorized parties (the attackers) were able to find unencrypted personal information within the organization's email caches. Of course, most security policies explicitly prohibit users from storing data of this nature within their email boxes for exactly this reason, but as we are seeing, it happens quite frequently. While it is certainly possible to find valuable information in email (and we saw some breaches happen this way in 2018), data exfiltration from human-structured data such as email is usually laborious and only worth the effort for small, highly valuable data, such as intellectual property or political communications. For large-scale, profit-minded attacks, email is often just the first step in a broader campaign to reap larger stores of valuable information.

EPISODE 3

Formjacking Injections Get Meaner

Formjacking is not a new type of attack, but it has exploded in popularity over the last two years, primarily in the form of Magecart attacks. The name Magecart was originally assigned to the threat actor groups who carried out the initial exploits of a shopping cart vulnerability on the Magento e-commerce platform (Magento + shopping cart = Magecart).[3] The vulnerability itself was a flaw in PHP's *unserialize* function that allows attackers to execute arbitrary PHP code for formjacking.[4] Although formjacking is not limited to PHP systems, PHP is highly targeted by attackers, and formjacking remains one of their preferred tactics.

A formjacking attack injects a command to siphon information that users put into an online form, then delivers that information to a location under the attacker's control. Most of the time the information sought by attackers is login credentials or financial information. We found 83 breaches attributable to formjacking attacks on web payment forms, with 1,396,969 payment cards compromised. In terms of number of breaches, nearly half of these came from the retail industry.

We found 83 breaches attributable to formjacking attacks on web payment forms with 1,396,969 payment cards compromised.

Complexity Widens the Attack Surface

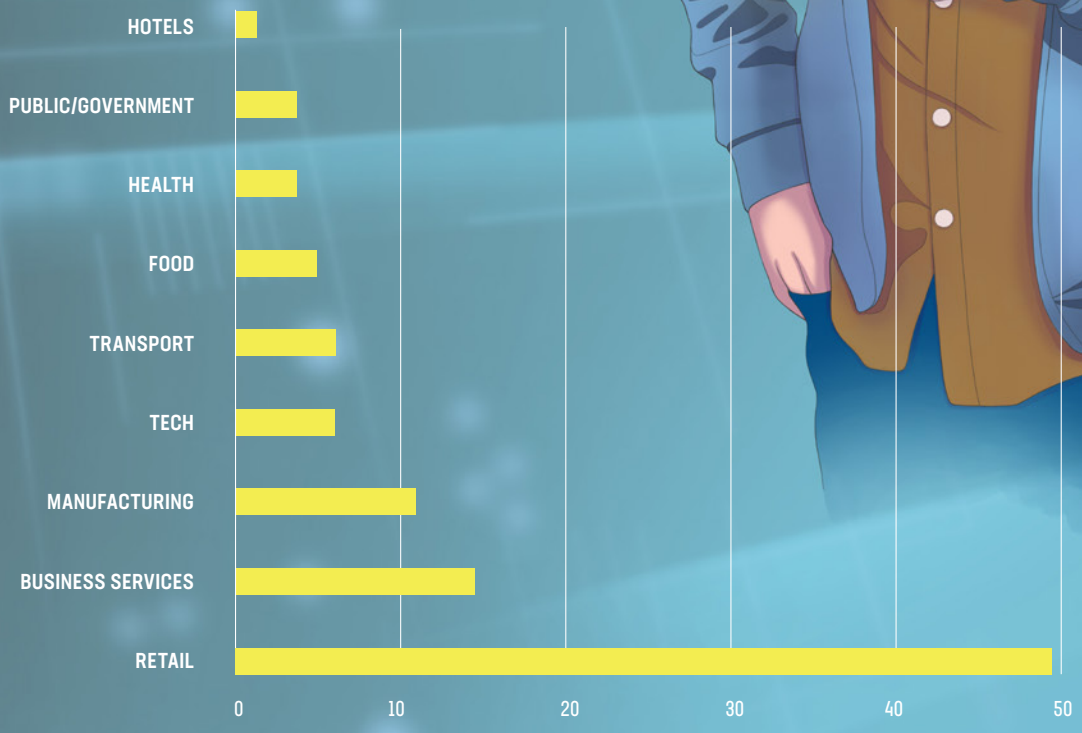
Formjacking has become more sophisticated because of the distributed and decentralized nature of applications. To the user, an application may appear to be a simple program. In reality, most applications on the Internet are swarms of microservices and sub-applications, all converging at the last minute into a coherent user experience.[5] These embedded services can include user analytics, chat features, debugging tools, social media sharing capabilities, and advertising, among others. Increasingly, these microservices are being linked to and run from external third-party sites. In other words, the active code is running on a server that has nothing to do with the "primary" application.

Attackers inject their commands and code into these adjacent services and come sideways to steal form data, the way bank robbers tunnel into a vault from an adjacent storefront. As webpages pull content from increasingly disparate sources, we're seeing more content getting injected in the browser from third-party add-ons.[6] These exploited third-party tools run in the same computing context as the main web application and its sensitive content, like payment input fields.

This raises issues of visibility with respect to traditional controls. Standard web application firewalls (WAFs) protect the primary site by examining traffic between the client and the

ATTACKERS TARGET THIRD-PARTY APP SERVICES TO COME IN SIDEWAYS AND STEAL FORM SUBMISSIONS, THE WAY BANK ROBBERS TUNNEL INTO A VAULT FROM AN ADJACENT STOREFRONT.

FORMJACKING BREACHES BY INDUSTRY



THE LESSON IS CLEAR: FOR ANY ORGANIZATION THAT ACCEPTS PAYMENT CARDS VIA THE WEB, THEIR SHOPPING CART IS A TARGET FOR CYBER-CRIMINALS.

COMMON INJECTION ATTACK PATH

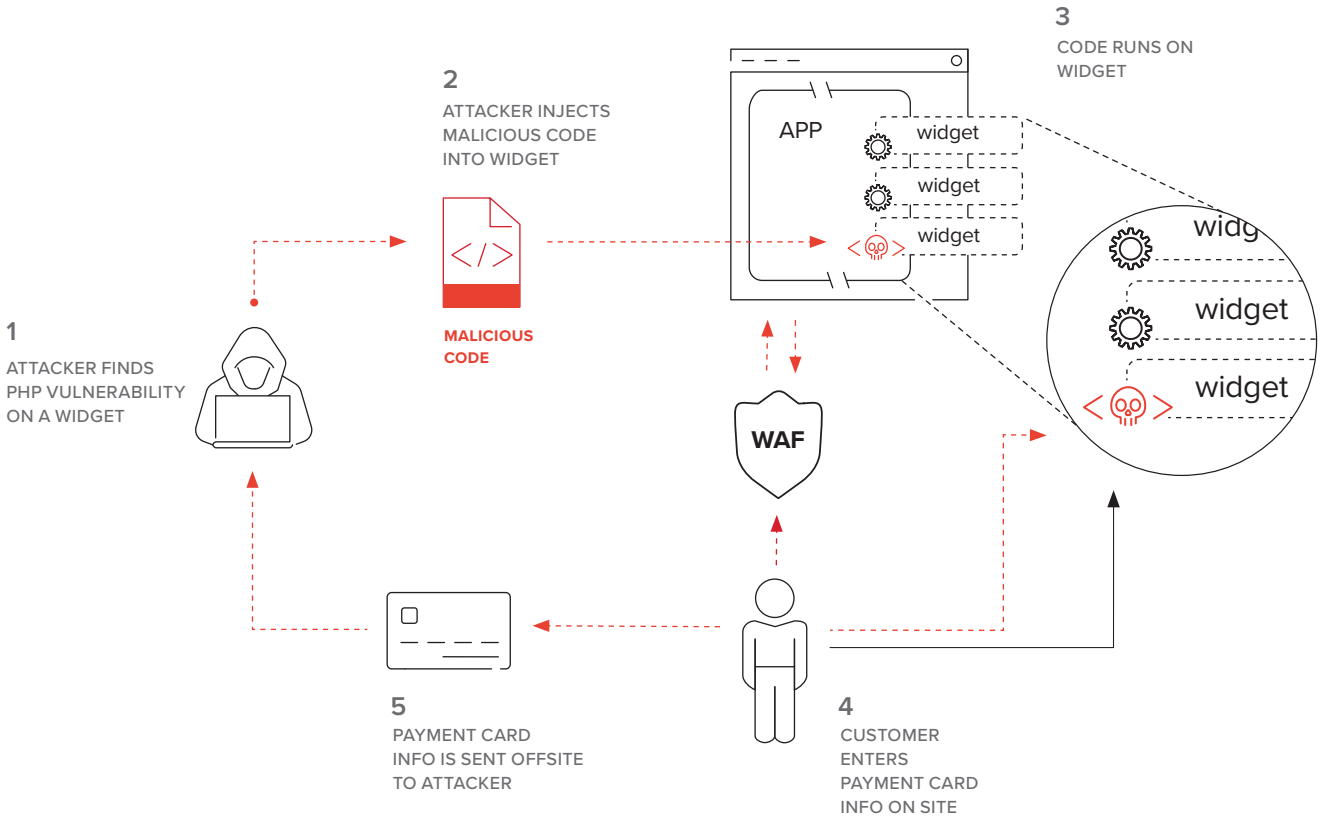


Figure 6
COMMON INJECTION ATTACK PATH
The path that both malicious code and valuable financial information take during an injection attack, now that third-party services are such a common component for web applications.

app server. Third-party scripts, however, are loaded directly by the client browser, bypassing perimeter security. The WAF may see a script such as an advertisement loaded from an ad network, but it does not see the contents of that script. Traditional security tooling views it as completely legitimate. Furthermore, sites that deliver malware or receive skimmed financial information tend to have legitimate encryption certificates on look-alike domains. This is what makes injection such a durable flaw, and why its latest incarnation makes third-party content such a significant problem.

Reject the Inject

The lesson is clear: for any organization that accepts payment cards via the web, their shopping cart is a target for cyber-criminals. If Magecart is in use, it should be immediately patched with the latest version. Because injection flaws can be exploited in any stage of an attack, finding and evaluating their impact depends on context. The risk of these kinds of attacks are magnified when the target web application uses third-party code running offsite. We strongly recommend thorough testing and watching of all third-party components on sites with forms accepting critical information.

Figure 7
(previous page)
FORMJACKING BREACHES BY INDUSTRY
The distribution of formjacking breaches by industry (by breach count, not record count).

EPISODE 4

Access Credential Attacks

Figure 8
(following page)
2018 ACCESS BREACHES BY CAUSE: U.S.
The sample text from breach notification letters for each category shows the overlap between categories and the difficulty doing of root cause analysis from afar.

In 2018, 47% of breaches were access-related, and 20% specifically targeted email access. Think about that: email is directly attributed as a factor in more than one out of five breach reports. A typical breach notification letter goes something like this: “Unauthorized persons used stolen credentials to gain access to emails containing confidential records…”. By accident or design oversight, organizations are still storing unencrypted medical and financial data in weakly protected email boxes. This has been a problem for decades and looks like it will persist for some time.

Brute Force

In addition to email and phishing attacks, we continue to see many brute force attacks. Although they are only successful four times out of a thousand, they’re cheap and easy to attempt. Attackers understand the economic profit of “buying bulk,” so there are a lot of brute force attacks going on.

We typically define brute force attacks as either ten or more successive failed attempts to log in in less than a minute, or 100 or more failed attempts in one 24-hour period. However, attackers realize that these kinds of behaviors are easily monitored and so have begun to alter their behavior. Sophisticated brute force campaigns now employ “low-and-slow” attacks, often using an IP address only a few times before trying from a different one. These kinds of brute force attacks are almost impossible to distinguish from legitimate connection attempts, though the more advanced WAFs have some capability to correlate this traffic and mark it as an attack.

Depending on how robust your monitoring capabilities are, brute force attacks can appear innocuous, like a legitimate login with correct username and password.

One of the best threat intelligence sources we have for brute force attacks comes from our own F5 Security Incident Response Team (SIRT). The F5 SIRT reported that in 2018, brute force attacks against F5 customers were the second most frequent type that they encountered, constituting 19% of addressed incidents. While the SIRT also noted a low success rate, even failed brute force attacks can affect system performance. On six separate occasions, the SIRT found that brute force attacks caused the target’s entire authentication infrastructure to go down. Even when the servers stayed up, authentication for legitimate users locked out or bogged down, resulting in an indirect denial-of-service attack.

2018 ACCESS BREACHES BY CAUSE: U.S.

20% EMAIL

"We have learned that an unauthorized third party appeared to have gained access to employee email accounts and that some of those messages may have contained personal information belonging to some customers."

19% PHISHING TO EMAIL HACK

"We were the victim of an email phishing attack resulting in emails and attachments from employee email accounts may have been potentially accessed by an unauthorized person."

4% STOLEN CREDENTIALS

"We learned that an unauthorized third party gained access to a third party site we use using our employees credentials."

1.8% CREDENTIAL STUFFING

"We've learned that an unauthorized third party obtained usernames and passwords to login to some accounts. We've determined that unauthorized parties may have obtained usernames and passwords from other companies' security breaches."

0.6% PHISHING TO STEAL CREDS

"We have determined that our Office 365 portal was subject to unauthorized access because of a phishing scam and that employees accounts may have been compromised."

0.6% EMAIL SOCIAL ENGINEERING

"Some of your personal information was released as part of an employee's response to a phishing email directed to the company by someone impersonating a senior member of management."

0.4% BRUTE FORCE

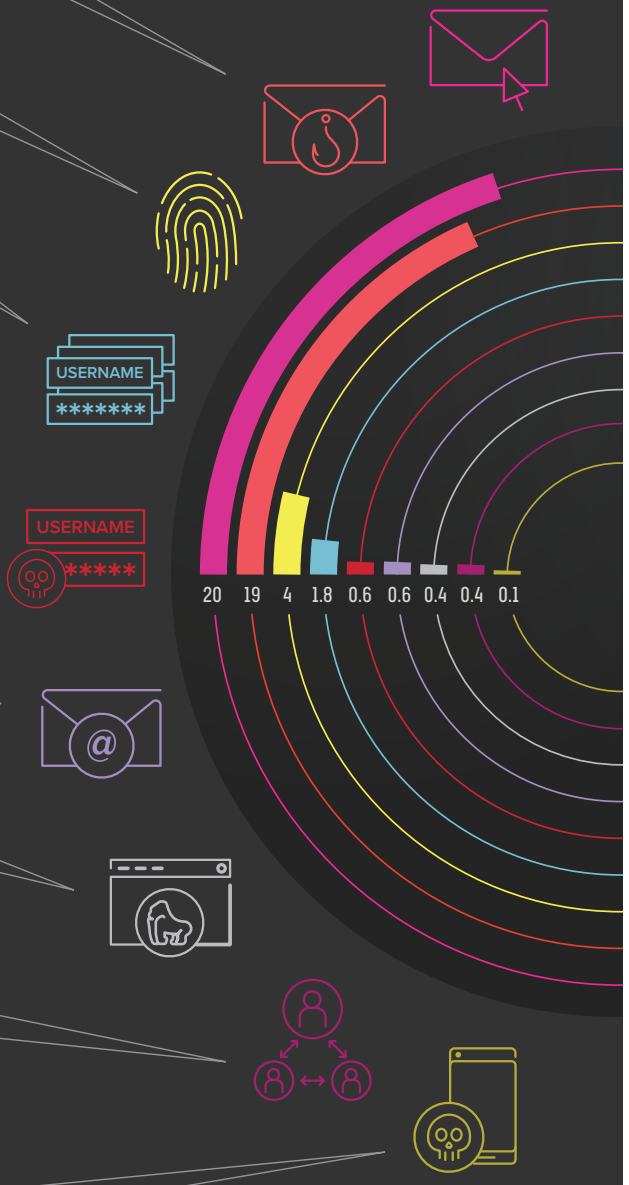
"We became aware of a significant increase in the number of login errors that were the result of numerous attempts to log into our site from foreign countries."

0.4% CREDS STOLEN FROM 3RD PARTY

"We recently learned that a third-party vendor in possession of some of our data experienced a security incident. As a result of this issue, certain personal information contained in the vendor's environment may have been accessed by an unauthorized party."

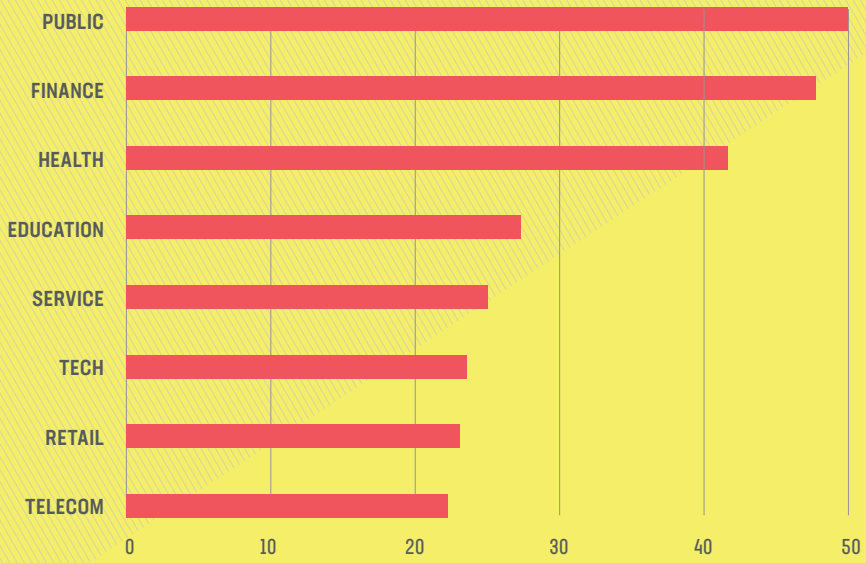
0.1% PHONE SOCIAL ENGINEERING

"An unidentified telephone caller claimed to be one of our customers and was able to answer security questions by providing detailed personal identifying information..."



BRUTE FORCE ATTACKS BY INDUSTRY

AS A PERCENTAGE OF 2018 F5 SIRT INCIDENTS

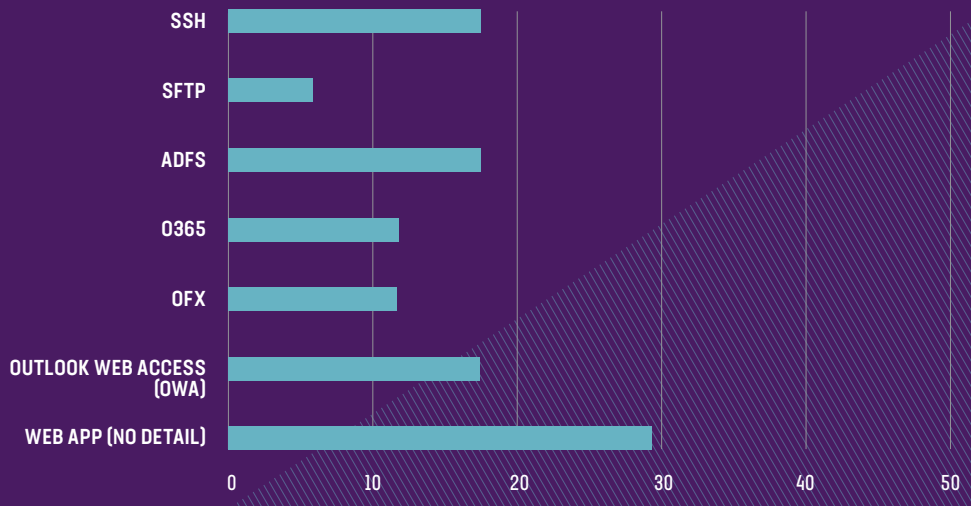


19%

BRUTE FORCE ATTACKS AGAINST F5 CUSTOMERS WERE THE SECOND MOST FREQUENT TYPE, CONSTITUTING 19% OF ADDRESSED INCIDENTS.

BRUTE FORCE ATTACKS BY PROTOCOL/SERVICE

AS A PERCENTAGE OF 2018 F5 SIRT INCIDENTS



29%

AT 29%, ATTACKS AGAINST WEB APPLICATIONS SURPASSED ALL OTHER SERVICES.

Access Denied

Figure 9
(previous page)
BRUTE FORCE ATTACKS BY INDUSTRY
F5 SIRT brute force attacks by industry as a percentage of reported 2018 SIRT incidents.

How do you reduce the risk that access attacks pose? We'd love to say, "just MFA it" and drop the mic, but we realize that multifactor authentication is not always feasible in the timeframes we'd like. The NIST Digital Identity Guidelines (800-63-3B) offer good principles that get away from some well-intentioned but obsolete ideas about access control.[7] An important strategy is to check passwords against a dictionary of default, stolen, and well-known passwords.

Incident response should include a streamlined and guiltless method for users to report suspected phishing. Users should feel no shame in asking about or reporting a phish so you can catch and/or contain them quickly.

As part of an assume breach approach, plan for an attacker to gain access to email, and gear your forensics accordingly.... In the event of an incident, these logs may be your lifeline.

Make sure your system can at least detect brute force attacks. Setting up alarms is a good start, but it's better to slow down the session by throttling or using CAPTCHA, or even blacklisting the IP address. If you're going to lock someone out, make sure you can fail gracefully, and set up reset mechanisms that work for both you and your users and get the legitimate traffic back online as quickly as possible.

Figure 10
(previous page)
BRUTE FORCE ATTACKS BY PROTOCOL/SERVICE
Brute force attacks mitigated by the F5 SIRT, broken down by protocol/service.

In many of these cases, the breach notification letter mentions how unauthorized parties (the attackers) were able to find private, personal information within the organization's email caches. Working to encrypt or eliminate confidential data in email is a strong recommendation.

As part of an "assume breach" approach, plan for an attacker to gain access to email, and gear your forensics accordingly. When setting up logging, check what level of detail your email system provides. Can you recreate an entire email session with log data? Can you tell what settings the attackers might have changed? Can you tell exactly what they downloaded or forwarded? Set the log settings and test them by logging in to see what is logged. In the event of an incident, these logs may be your lifeline.

EPISODE 5

Attackers' Eyes are on the API

In the simplest possible terms, an application programming interface (API) is a user interface for apps instead of users. It creates a connection point through which other app services or mobile apps can push or pull data. An API gateway is software running on an application server that coordinates and manages traffic for the API.[8] APIs allow other applications to use the output of the original service in a different way without having to recreate the original service from scratch.[9]

For example, consider Google Earth. Many of us have used the desktop app, which runs normally, or the mobile app, which pulls data from Google Earth servers through an API. But many more apps use the Google Earth Engine API to achieve more specific goals (such as visualizing and measuring change on the surface of the Earth) than Google Earth itself does.[10] This is what makes APIs such a great way to scale and embed functionality into other apps. They are more than just another piece of infrastructure; they can transform an organization's business model by directly generating revenue. As a result of the growth of APIs, organizations are recognizing new opportunities to generate traffic and revenue, often using existing components of their environments with minimal modification.[11]

APIs are rich targets for attackers because they are often set up with overly broad permissions to access any data within the application environment.

However, APIs are also rich targets for attackers. Because they are not intended for human use, APIs are often set up with overly broad permissions to access any data within the application environment. Permissions are usually set up for the user making the original request, and these permissions are, in turn, passed to the API. That is all well and good until an attack bypasses the user authentication process, going directly to the downstream app. We found that API compromises tended to fall into three patterns of API use that correspond to common breaches: large platforms, mobile apps, and misconfigured "big app" breaches.

APIs ARE AN OBSCURE BUT STARTLINGLY DIRECT PATH TO VALUABLE DATA, LIKE PAYMENT CARD INFORMATION, THAT CRIMINALS CAN RESELL ON THE MARKET.

LARGE PLATFORM API BREACH

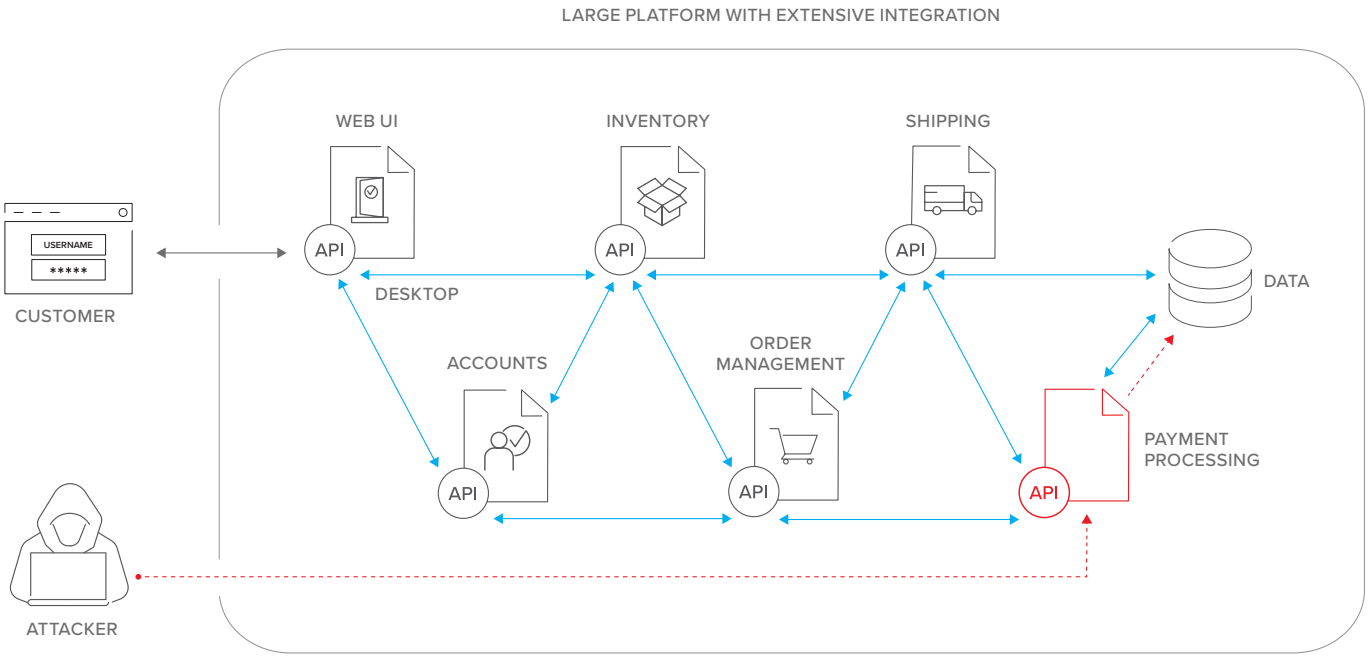


Figure 11
LARGE PLATFORM API BREACH
A large web platform with a theoretical microservices architecture that depends heavily on APIs for communication and integration between functions is more vulnerable to an attack that exploits a vulnerability (or, often, a simple lack of access control) on an API in order to gain access to sensitive data.

API Breaches at Large Platforms

Organizations with high traffic sites offering a wide range of services (such as social media or e-commerce platforms) often feature a large number of third-party integrations. These integrations rely on APIs to collect data from third parties and deliver them to the user in a seamless fashion. The growing decentralization of infrastructure, represented by multi-cloud environments, third-party functions and content, and serverless and containerized architectures means that APIs are essential for modern, high-volume platforms.

Some of these platforms have hundreds of APIs, all of which need to be managed and monitored. These kinds of organizations and business models have tended to figure prominently in the API breach notifications we've seen, and breaches of this type constituted 41% of known API breaches from September 2018 to September 2019.

41% **LARGE PLATFORMS WITH A LARGE NUMBER OF THIRD-PARTY INTEGRATIONS CONSTITUTED 41% OF KNOWN BREACHES IN 2019**

MOBILE APP API BREACH

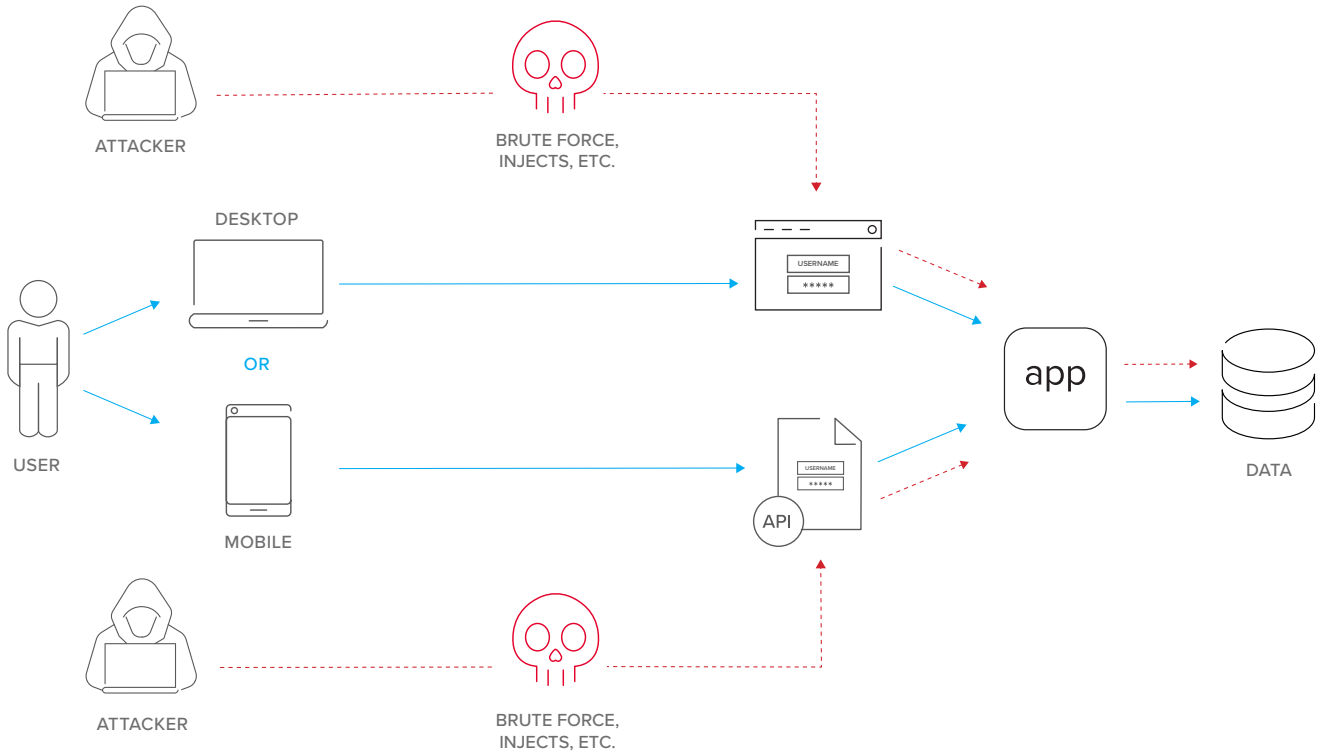


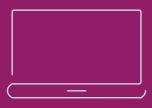
Figure 12
MOBILE APP API BREACH
An app is routing mobile traffic through a mobile-specific API, while desktop users connect to the same backend through a web interface. The same tactics that would work against a traditional web application, such as brute force or injection attacks, are just as likely to work against the mobile API, and defenders are less likely to maintain awareness of the API.

Mobile Apps

Most mobile apps rely on APIs to pull data from servers, which allows the apps to use fewer resources on the devices themselves. Because of some of the inherent challenges with securing mobile applications, there is a vibrant community of attackers who decompile mobile applications looking for vulnerabilities or opportunities, such as hardcoded credentials or weak access control. The API is often a focal point for these efforts. Mobile API breaches represented 31% of all API breaches from September 2018 to September 2019.

31% MOBILE API BREACHES REPRESENTED 31% OF ALL API BREACHES IN 2019.

PLATFORM API BREACHES



MOBILE API BREACHES

2011

- Westfield 09/11 ●

2015

- 03/15 Tinder Mobile ●

2017

- WordPress 02/17 ●
- 08/17 Instagram app ●

2018

- 01/18 Tinder Mobile ●
- 04/18 RSA Conference Mobile ○
- Venmo 07/18 ○
- Salesforce 08/18 ○
- T-Mobile 08/18 ○
- Facebook 09/18 ○
- 09/18 Apple iPhone ○
- Valve 10/18 ●
- GitHub 10/18 ●
- US Postal Service 11/18 ●
- 11/18 City of York, UK ●
- Federation of Industries of Brazilian State of São Paolo 11/18 ○
- Urban Massage 11/18 ○
- Sky Brasil 11/18 ○
- Atrium Health 11/18 ○
- Data&Leads 11/18 ○
- Kubernetes 12/18 ●
- Facebook 12/18 ●
- Twitter 12/18 ○

2019

- Landmark White Ltd 02/19 ○
- 02/19 Uber ○
- Kubernetes 02/19 ●
- 02/19 Padora and Viper (Clifford) Car Alarms ●
- Drupal 02/19 ●
- Portainer Dock Tool 02/19 ●
- 03/19 63red Safe ○
- Nagios XI 04/19 ●
- 04/19 Shopify Exchange app ●
- JustDial Leak 04/19 ○
- 04/19 Tchap Messaging app ○
- Facebook Marketplace 04/19 ○
- GateHub 04/19 ○
- Venmo 04/19 ○
- 06/19 OnePlus Mobile ○

● VULNERABILITY ○ MISCONFIGURATION

Figure 13

API BREACHES TIMELINE

A timeline of API breaches showing both large, decentralized application platforms and mobile APIs. API breaches in 2018 and 2019 have tended to be either vulnerability exploits or misconfigurations, usually involving a lack of access control.

The Misconfigured Big App

These breaches occurred because stakeholders in organizations were not aware of either the existence of an API, or the impact of an insecure one, and so they put no authentication (or weak authentication) in front of it. As silly as it sounds, it is hardly surprising, since it once again points to the fundamental challenge of visibility, both from the standpoint of information systems and large organizations. Misconfigurations in large, multi-tiered applications were responsible for 28% of API-related breaches from September 2018 to September 2019.

Security Researchers: The New Threat Actors?

In terms of new events, every API breach we identified between November 2018 and September 2019 was attributable to misconfigured access controls. In other words, system owners did not realize that their APIs were vulnerable. So far, the principal "threat actors" in these scenarios have not been criminals but security researchers looking to get their names in the headlines. You might say that these system owners got lucky, as did the researchers.

API, I Will Find You, and I Will Lock You

Locating all your APIs is a prerequisite to any defensive scheme, and this is a continuous process as changes to your environment can accidently expose what you thought was hidden. Scanning continuously for configuration anomalies and listening services is always a good idea. As for locking down an API, you should develop a technical security standard (sometimes called an API security policy) that defines who can do what to which services over the API. For example, an API that accepts video should first authenticate the mobile application, then authenticate the user. After that, the API should only allow certain kinds and sizes of uploads, only to that user's storage, via tightly defined methods. An API policy like this makes it easier to apply additional security services to enforce these rules.

EASY TARGETS WILL REMAIN POPULAR. THIS MEANS THAT UNSOPHISTICATED CAMPAIGNS AGAINST OBSOLETE, OBSCURE, OR DIFFICULT-TO-SECURE TARGETS WILL REMAIN PREVALENT.

The Future of Application Security

Understanding what's changed and what hasn't since our 2018 report can help us plot the trajectory of the various forces shaping the threat landscape, and make a few predictions.

Easy Targets Will Remain Popular

Unsophisticated campaigns against obsolete, obscure, or difficult-to-secure targets will remain prevalent. Therefore, PHP will continue to lead in terms of supplying rich, soft targets, and situational awareness remains important in terms of mitigating both vulnerabilities and threats.

Access Attacks Will Shape Human-Computer Interactions

It bears repeating that applications aren't just the code that they execute, but also everything around them that makes them tick: architecture, configuration, other assets to which the application connects, and—not least—the user population. In other words, some things that have little to do with the narrow definition of a web application can have huge effects on that application's security. The prevalence of access attacks such as phishing is a good reminder of this bigger context.

With that in mind, there is much that you can do to mitigate access risk. However, controls such as convoluted authentication, failing closed, and multiple forms of verification might serve the business' security goals, but at the cost of the value of the application to its audience. The underlying questions that access attacks pose are fundamental to the relationship between the digital world and the real one. How this specific part of the arms race evolves will determine much in terms of how we interact with internetworked information systems on the broadest possible level.

Threat Intelligence Contradicts the Concept of Best Practices

One conclusion we can draw from the industry breach profiles is that actual breaches confirm the need for risk-based security programs instead of best practices or checklists. If we know that successful attacks map to where organizations store sensitive assets, it follows that organizations need to tailor controls to reflect the threats they actually face. This supports our long-held assertion that risk assessment needs to be a cornerstone of any security program, and the first step in any risk assessment is a substantive (and ongoing) inventory process.

The Industry Will Metabolize Emerging Risks, as Always

Over time, as new risks emerge from changing technology and the arms race that is information security, the industry will gradually incorporate those risks into our business models. Cloud computing has gradually shifted from a bleeding-edge risk to a cornerstone of modern infrastructure. The risks associated with the cloud have either been mitigated or displaced to contractual risk in the form of service level agreements and audits.

As the business world comes to grips with new trends in service provision, risks gradually morph from purely technical exploits that are managed reactively to facets of a business model that are managed proactively. We predict that the same will happen with the trend of third-party web functions and content. Detecting and mitigating injection flaws in light of these trends depends on adapting our assessments and controls to this new reality, not just fixing code.

[1] https://w3techs.com/technologies/history_overview/programming_language/ms/y

[2] http://fortune.com/2018/12/18/click2gov-local-government-portals-hackers-credit-card-breach/

[3] https://www.riskiq.com/blog/labs/magecart-adverline/

[4] https://www.exploit-db.com/exploits/39838

[5] kumarde.com/papers/tangled_web.pdf
The noted security researcher Dan Geer remarked on this trend as early as 2014 in this talk: https://www.youtube.com/watch?v=hxLJExWk9GE
There is also a useful tool that maps out third-party requests for any domain, and illustrates how decentralized apps have become: http://requestmap.webperf.tools/

[6] https://duo.com/blog/malicious-hackers-take-over-media-sites-via-content-delivery-providers
The Magecart attacks mentioned earlier are also an example of this tactic: https://www.riskiq.com/blog/labs/magecart-adverline/

[7] https://pages.nist.gov/800-63-3/

[8] Not to be confused with AWS' API gateway, which is confusingly named API Gateway. For a non-AWS implementation of API gateways, see https://docs.microsoft.com/en-us/azure/architecture/microservices/design/gateway
For AWS' approach, see https://aws.amazon.com/api-gateway/

[9] For a brief perspective on APIs and microservice architectures, see https://www.kuppingercole.com/blog/balaganski/api-security-in-microservices-architectures

[10] https://earthengine.google.com/case_studies/

[11] https://hbr.org/2015/01/the-strategic-value-of-apis

We Want To Know What You Think

As security practitioners study how the Internet has evolved, the ways we manage new risks will mature. Attacks will also morph in turn, finding new ways to trouble us. In the meantime, we hope that the perspective and practices outlined in this report help you manage the latest incarnations of these older risks.

If you have feedback, data to share, requests for topics, or thoughts about our approach, please let us know. You can reach us on Twitter @f5labs, or email us at F5LabsTeam@f5.com.

APPLICATION THREAT INTELLIGENCE

US Headquarters: 401 Elliott Ave W, Seattle, WA 98119 | 888-882-4447 // Americas: info@f5.com // Asia-Pacific: apacinfo@f5.com // Europe/Middle East/Africa: emeainfo@f5.com // Japan: f5j-info@f5.com