Think like a hacker: Reducing cyber security risk by improving API design and protection

Received (in revised form): 5th March, 2020

Gerhard Giese
Senior Manager, Akamai Technologies, Germany

Gerhard Giese is Industry Strategist at Akamai Technologies. Gerhard started at Akamai in 2010 and is now manager in the Financial Sector, responsible for customer advisory, information sharing and consulting. With more than 20 years' experience in the security field, Gerhard has accumulated in-depth expertise in network security as well as distributed denial of service (DDoS) mitigation and data theft prevention. He continues to interact directly with clients as a trusted security adviser, to identify the most pressing challenges for online businesses. In addition, Gerhard regularly delivers talks at industry conferences and works as an independent consultant for federal state authorities such as the German Ministry of IT Defence. Prior to Akamai, Gerhard was a senior network engineer at McAfee. Gerd holds CISSP and CCSP certifications and is a certified ethical hacker.

Akamai Technologies GmbH, Parkring 20–22, 85748 Garching bei München, Germany
Tel: +49 89 94006-0; E-mail: ggiese@akamai.com

Abstract Application programming interface (API) traffic now dominates the Internet. Unlike traditional web forms, APIs are faster and more powerful, but often do not get the correct protection — expanding the security risk for organisations. APIs connect people, places and things to create seamless integrations, richer experiences and new revenue models. This paper deals with when an API is misused, and stipulates how the exposure to an organisation can be significant. The paper discusses why it is no longer safe to assume APIs will be used as intended or remain hidden to prevent unauthorised access or abuse. To stay ahead of the next cyber security exploit, API developers need to start thinking like a hacker. The paper promotes a proactive approach to identifying, designing, managing and protecting APIs which will minimise the attack surface and prevent damaging data breaches.

KEYWORDS: API, attack surface, apps, Internet of Things (IoT), pen testing, hacking, web security

BUT FIRST, COFFEE

From car batteries inexplicably drained to personal information accessed via a simple phone number lookup, overly broad API access can wreak havoc when exploited. Even seemingly harmless irregular API interactions can pose a threat to business. Such was the case when a coffee chain rolled out an online ordering app for their customers to skip the line.

The coffee order API calls followed typical, well-established security procedures and protocols. In one country, however, the number of orders that were paid for, but not collected, increased. Upon a closer look at the unclaimed orders, the company found that almost every shop in the country received these calls, always more than twice. After ruling out all other technical reasons, the company concluded that automated

Coffee Anyone?

"Only my mobile app will call my API"

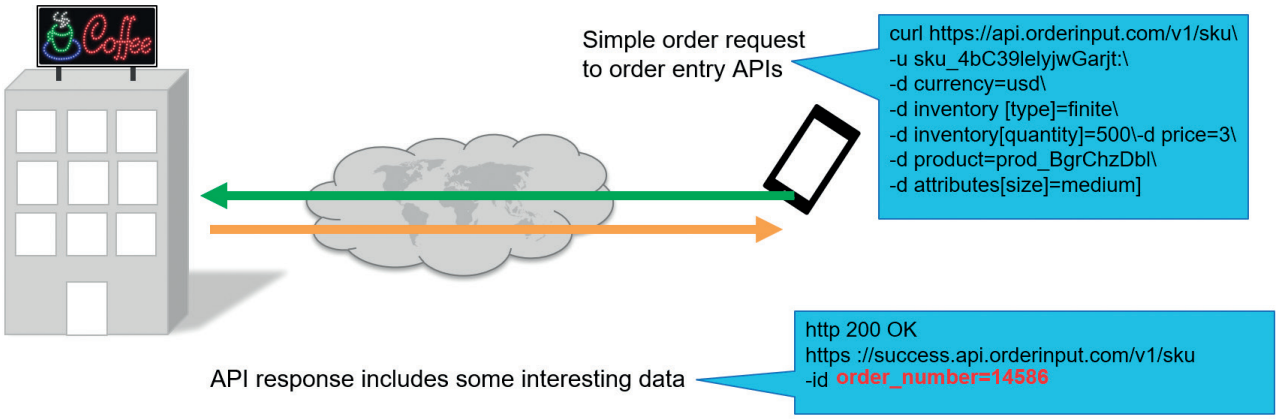

Figure 1: Fake order calls yielded order number to expose store sales
Source: Akamai (2019)

requests for fake orders were being sent to the API to solicit the confirmation response that included a sequential order number — a simple piece of information revealing the sales at each store to the competition or a potential thief.

API GROWTH AND IMPORTANCE

API use has exploded. Today, there are more than 22,000 APIs searchable on ProgrammableWeb.[1] The glue that holds the connected world together, APIs have grown rapidly in number and capabilities. Unlike traditional web forms, they are faster, more powerful and harder to protect, expanding the security risk for organisations.

Gartner predicts that API abuses will be the most frequent attack vector resulting in data breaches for enterprise web applications by 2022.[2] Usually well documented, APIs provide instructions for connecting people, places and things to create seamless integrations, richer experiences and new revenue models. When an API is misused, the exposure can be significant.

Responding to the Cambridge Analytica scandal which affected as many as 50m profiles, Facebook made several API changes to better protect user information.[3] Even as breach after breach is disclosed, companies are still not doing enough to limit API abuse. A computer science student scraped 7m Venmo transactions to show that public activity can still be easily obtained, a year after a privacy researcher downloaded hundreds of millions of transactions.[4]

OVERLOOKED BY SECURITY

Organisations go to great lengths to secure their applications and web pages but leave the backdoor wide open to valuable data with unfettered API access. A false sense of security exists that assumes APIs will be used as intended by only their mobile apps. Just because an API is not directly exposed, however, does not mean it is not vulnerable

TYPE	UA	
Browser	Chrome	13%
	Mobile Safari	8%
	Firefox	2%
	Internet Explorer	2%
	Edge	1%
	Safari	1%
	IE Mobile	0%
Non Browser	Other	66%
	CFNetwork	3%
	Apache HttpClient	2%

Figure 2: API traffic by user agent[6]
Source: Akamai (2019)

to a breach. Security by obscurity is not a threat prevention strategy — a hidden domain name or internet protocol (IP) address is not enough protection.

An API call is not the same as a web page call. Application security controls will not protect APIs, as cyber attackers will bypass them to focus on penetrating unprotected entry points. API abuse is difficult to prevent, especially when no one is looking. Areas with complicated business logic, such as business-to-business connections between databases or business-to-consumer checkout procedures, are the most vulnerable due to their complexity.

When API vulnerabilities are discovered, they can be difficult to resolve. While one-third of calls come from web browsers allowing easier control and fixes, the remaining two-thirds come from non-browsers, such as mobile phones, gaming consoles, smart televisions and others.[5] The software inside many connected devices is not easy to update and maintain. Devices get shipped without protocols for software updates and if vulnerabilities are found, the likelihood that they will get patched is slim to none. How often are users updating firmware on a connected baby monitor or coffee maker?

EVERYBODY LOVES APIS

API proliferation is happening across all verticals, especially media and technology.[7] Now dominating the Internet, API traffic traversing over the Akamai content delivery network, for example, accounts for 83 per cent of all hits, while hypertext markup language (HTML) traffic fell to 17 per cent — with JavaScript Object Notation (JSON) content more than doubling in four years, jumping from 26 per cent to 69 per cent.[8] Leveraged by companies, users and attackers alike, APIs offer a multitude of benefits and challenges to organisations.

In business, APIs accelerate innovation by adapting to user demand more quickly and increasing the stability of application services. Making daily life more convenient,

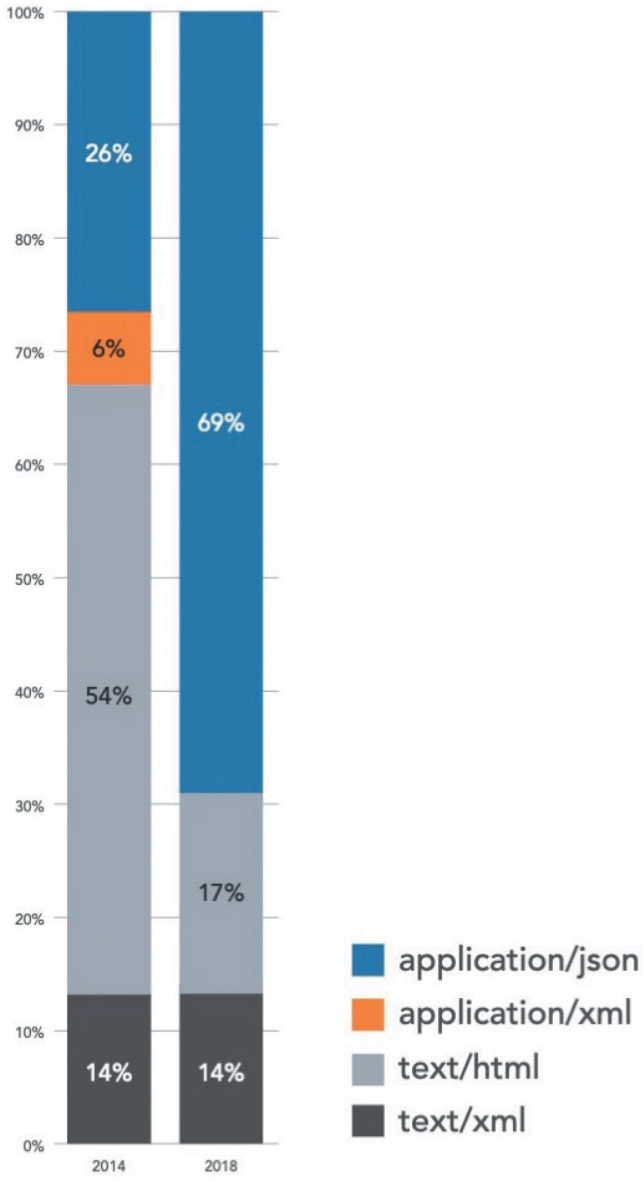

Figure 3: Online traffic by content type on the Akamai content delivery network[9]
Source: Akamai (2019)

APIs help improve customer satisfaction and, in turn, business results. With development, operations and security teams in separate groups, extra work is required to enforce security controls across organisational boundaries.

Although most end-users do not realise it, the core functionalities of web and mobile services have shifted from pure HTML to JSON, separating functionality and backend calls to create lean communication that accelerates the responsiveness and agility of applications. This is especially important for mobile devices to provide the highest speed for immediate actions such as viewing schedules or booking tickets, while facing network challenges such as lossy connections, poor network quality and high latencies.

Mobile apps thrive on APIs, allowing more integrations and unlimited information exchanges to unlock new user experiences — navigation to calendar appointments or order payment with an e-wallet as examples — and drive mobile adoption. New security challenges arise with native mobile apps as opposed to mobile clients, when someone is using a normal browser on a mobile phone. Typically, organisations defend against attacks on high-value login pages with bot detection such as multi-stage logic transparent to humans, but not machines. These deterrents prevent crawlers from stealing data, but will only slow, not stop more sophisticated threats. Since most mobile APIs do not render JavaScript, many common bot defences are unsuitable.

Attackers benefit from API proliferation as well, with neglected APIs providing an easy entry point into the application world. Even when organisations apply security controls to APIs, they often cannot keep up with the rate that development teams are rolling them out or shadow IT that creeps in when a department starts using a new third-party service. Designed to minimise human interaction, APIs allow attacks to quickly scale for data theft and modification of content. This automation also drives down attack costs, making them more frequent.

THREATS ARE FAMILIAR, BUT MORE VIRULENT

There is good news and bad news about the attacks on APIs. The good news is that malicious behaviour looks the same as on the web. Tactics include distributed denial

of service (DDoS) attacks that interrupt availability or overwhelm resources and insider threats that insert intruders into the data flow or inject malware, spyware or ransomware into systems. With rampant data theft, the exploitation of weak or stolen credentials is commonplace. Finally, there is the unexpected use (or misuse) of APIs, such as the coffee chain application example.

Experience fighting the same attacks on the web helps organisations identify and mitigate attacks on APIs. The bad news is that APIs are easier to attack than traditional web forms, with those attacks spreading more quickly as APIs facilitate machine-to-machine communication. Since so many APIs are left totally exposed, they are increasingly becoming a popular target for attackers. Over a period of two months, Akamai collected data on more than 8bn credential stuffing attempts.[10] With four times more credential stuffing attempts occurring on APIs,[11] developers need to start thinking like a hacker to prevent online threats.

PUBLISHING WITH CAUTION

To minimise the attack surface, organisations must exercise extreme caution when sharing or using shared APIs. Developers like to reuse and disseminate both public and private code with software development collaboration tools such as GitHub. This is a common practice to speed up development and leverage knowledge across the developer community. Sensitive information is, however, sometimes uploaded to GitHub by accident — and hackers know it, easily capitalising on careless mistakes with tools such as Gitrob, or shhgit, a live feed of secrets published to Gitrob.

When sharing APIs, it is critical to ensure data cleanliness, only publishing the necessary details. Who in an organisation is responsible for checking correct input values and output sanitation? If an API is shared on GitHub or other collaboration tools, a sanitation review should be conducted to ensure no corporate data such as API keys or domain names remain exposed. Microservices-based architecture communications are generally good, but if code is copied from a shared environment, a security briefing should be required to ensure the proper controls are in place before integration.

IDENTIFYING API EXPOSURE

DevOps teams utilise APIs to automate the operation of their networks across multiple clouds — this is the only way to scale to execute thousands of operations every day. Many of those APIs need to use public IP addresses to be accessible by everyone, introducing security risk. For visibility on the exposure created by these APIs, it is important to understand how they can be discovered and exploited.

There is no sure way to successfully conceal an API connected to the Internet. Only a few tools are required to identify API exposure and potential attack vectors. Several of these tools are freely available for download or part of Linux repositories. Used with good intentions, these tools can help organisations fortify security by checking for vulnerabilities. In the wrong hands, they can also make quick work of harmful online attacks. To identify potential exposures and be vigilant about malicious activity, developers should be familiar with popular discovery tools such as Network Mapper, Fierce, Shodan, Sentry MBA and SNIPR.

Network Mapper

Network Mapper (Nmap) is an open source utility for network discovery and security auditing.[12] It is flexible in supporting dozens of techniques for mapping out networks from port scanning and ping sweeps to operating system (OS) and version detection. This tool is supported by most operating systems and comes in many varieties, even a version with a graphical user interface

(GUI) for the keyboard challenged. The use is simple: with one line of bash code, the tool can quickly scan an entire network to determine if any vulnerabilities exist.

Part of a standard administrator toolbox, Nmap was used inconspicuously until recently. While Nmap is helpful in preventing attacks, it has also been frequently misused — allowing attackers to discover insecure entry points. The abuse has put this valuable tool at the centre of a general debate of the legality of port scanning tools.[13] Given the controversy surrounding port scans, it is prudent to first understand the potential legal ramifications and obtain prior authorisation before using Nmap to avoid any unintended consequences.

Fierce

While Nmap is a helpful solution for identifying APIs, it lacks speed and provides little intelligence. For further reconnaissance, Fierce is a more aggressive intelligence collection tool. Where Nmap stays passive, Fierce actively tries to exploit domain name system (DNS) servers (although no actual exploitation is performed with the tool itself) by using a common misconfiguration: unrestricted zone transfer information.

Zone transfer information contains the complete zone configuration including all registered devices as well as their names and IP addresses. This intelligence is of great value for attackers plotting anything from a simple DDoS to a direct web attack. If the DNS is set up correctly, the tool will begin scanning for typical hostnames such as auth., api. or developer., which results in a list of names and IP addresses. In a second step, the tool executes a reverse lookup by using IPs in the +/- range of the found addresses, which results in a list of new hostnames — attractive targets to attack.

Fierce users are rarely administrative or well-meaning, as this tool is mostly used by aggressors. By understanding the intelligence Fierce will yield about the network, organisations can put the appropriate security measures in place.

Shodan

While most search engines only index the web, Shodan finds, indexes and makes searchable all connected things — from web cams to traffic lights. Paid members even create alerts when new devices are added to their monitored subnets. Organisations can block Shodan from crawling their networks, but attackers will find other ways to exploit vulnerable devices.[14]

Shodan is another helpful tool for finding insecure pathways, especially in an era of shadow IT where companies are not always aware of what has been developed or connected to their networks. It boosts awareness around security risks as more and more things come online — exposing what hackers already know to the rest of the world — displaying the scale of the attack surface to encourage safer practices.

Sentry MBA and SNIPR

Widely available and easy-to-use account checking tools like Sentry MBA and SNIPR enable online threats to launch credential abuse attacks without much technical expertise.

Based on a program originally developed with a disclaimer for users to only run it against their own sites, Sentry MBA is a popular tool in the underground cracking community. Sentry MBA uses hard-coded and outdated HTTP User-Agents, which makes it easier to detect by defenders.[15] But it can still cause significant damage, especially for APIs, where it can take control and automate attacks.

SNIPR is the most advanced toolkit for checking credentials against popular websites. It offers support, tutorial videos and an active community that contributes new public configurations, credential leaks, proxy lists, bug reports and enhancement requests. To minimise exposure, it is imperative that

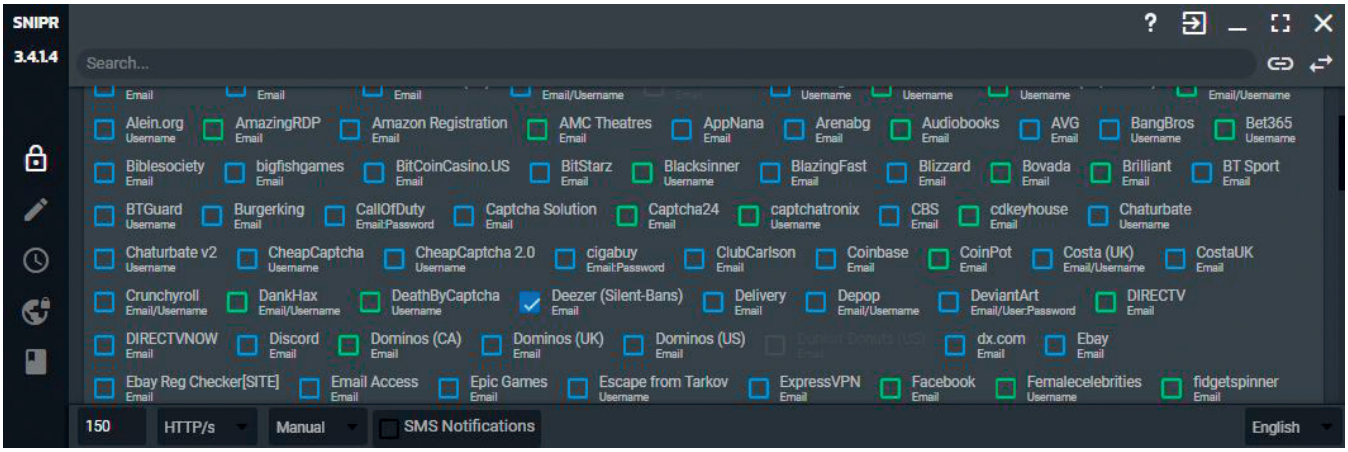

Figure 4: SNIPR configuration list page[17]
Source: Akamai (2018)

organisations check SNIPR configuration repositories for their websites to fend off any further abuse.[16]

DESIGNING SAFER APIS

To design APIs that are less likely to become a conduit for attacks, developers should start considering the usage model, user and operational role of each API. Running mobile apps on web APIs is not a good idea as they are two completely different use cases. The same goes for powering multiple user experiences with one API — user and management interfaces for internal and external users, for example. Sometimes regulations, such as the revised Payment Services Directive (PSD2) and Open Banking, stipulate that use cases be kept separate.

To understand the API usage model, developers need to identify and prioritise users and use cases to more easily spot suspicious activity. For example, a partner placing an order should not have the same access as a university student. Some questions to answer include:

- What are the use cases for the API?;
- Who are the intended users?;
- Who are the actual and current users?;
- Which users are more important?

Once the audience and uses are defined, developers need to create additional data points within the API to help differentiate users. With better user identification, it is easier to track anomalous behaviour that could lead to exploitation. To collect additional data about API use, organisations should require users to register, issue API keys and deploy traffic segregation.

MANAGING API TRAFFIC

Sometimes legitimate API users can cause unpredictable load, knowingly or unknowingly. In an example of API overuse, a company in Asia started receiving an abnormal amount of traffic to one of its web addresses, reaching 875,000 requests per second at one point. An initial assessment showed all the hallmarks of a major DDoS attack; however, the incident was not an attack at all. The spike was actually the result of a warranty tool gone haywire. When security started filtering traffic, the tool kept visiting the destination. As subsequent visits did not alter anything in the headers (such as the User-Agent) to bypass mitigations, the intent was not malicious. The company and tool vendor verified this conclusion and a fix was pushed within hours to the affected systems.[18]

In cases like this, it is important to enforce a quota to keep user requests under control. It is also critical to authenticate and authorise specific access for specific users, as developers cannot presume that users will interact with the API as expected. Deploying an API gateway establishes governance of API access by regulating the number of requests allowed per hour, per consumer. An API gateway can also inspect and validate JSON Web Tokens (JWT) and API keys, rejecting unauthorised traffic early, before it can overwhelm critical infrastructure and impact API functionality.

PROTECTING APIS

To protect APIs, developers must insert security controls in the right places. A layered approach to API security provides the most robust defence. Application security alone will not safeguard an API. Organisations must ensure that mobile and web applications cannot be exploited as well as extend the same level of security consideration to the APIs that connect to them. Attackers will try to access applications through APIs directly, and if successful, will render useless all attempts to secure app data.

Basic encryption between an application and API as well as session-based rate limits will increase security. While an API gateway ensures legitimate users do not misbehave, the addition of a web application firewall (WAF) insulates APIs even further from malicious activity by offering strong protection for known and unknown threats. Like a port firewall, a WAF is meant to secure internal systems against external threats. Instead of just blocking unwanted access on the port level (for example, not allowing web traffic on Port 80), however, a WAF will inspect the web traffic and block potentially malicious requests like SQL injection attempts, cross-site scripting and directory traversal. Mitigating web application attacks before they cause harm is always the best strategy.

The most effective protection against threats is to adopt a positive security model. Detailed API documentation encourages use, which can facilitate exploitation, but also fight threats by allowing security experts to define a positive security model that only accepts approved use. A positive security model allows developers to define users and behaviours — only processing well-formatted, in-specification requests and immediately dropping any deviation — ensuring the API is not misused. A modern WAF solution provides an API to create an ongoing strategy that updates as cybersecurity threats evolve, so the WAF understands 'normal' API usage and denies malicious traffic even without expressly recognising it.

Organisations can proactively guard against online threats by using a content delivery network (CDN) as forward defence to absorb powerful DDoS attacks without performance degradation. In addition, security at the edge of the network keeps threats away from sensitive data and infrastructure. For APIs associated with valuable content, such as account login or transaction pages, specialised bot detection identifies credential stuffing before it leads to fraud.

CREATING A PLAN

Organisations must recognise the significant risk APIs pose if not properly secured. If an organisation does not have the time, talent or resources to develop its own security solutions — often tedious and difficult to maintain — a variety of proven technologies and services are available in the market to protect APIs. A practical plan to start making the shift to a more secure API posture follows.

Next week

- *Assess the API ecosystem*: Find the APIs in the organisation, who owns them and whether there are any rogue systems in the domain (Nmap facilitates this process);

- *Identify potential security risks*: Understand if an API can be accessed with a simple telnet (no encryption), the information retrieved and if the systems used to serve up that information are patched.

Within three months

- *Understand who accesses APIs*: Determine the use case (internal, business-to-business, business-to-consumer) of the API and whether it is serving multiple purposes and audiences;
- *Define appropriate security measures*: Examine encryption and authentication options as well as whether there is an organic infrastructure behind an API that needs to be cleaned up, especially for customer-facing APIs.

Within six months

- Select a security solution that allows proactive API protection tailored to the organisation's needs;
- Set up a proof of concept to confirm functionality and usability;
- Drive a project to protect all APIs, both public and private;
- Establish an annual external penetration test to ensure API security posture is continuously maintained at the highest level.

CONCLUSION

API proliferation benefits enterprises and consumers with faster, value-added services for better user experiences and additional revenue streams. The increasing threat of API exploitation also introduces new security challenges. API developers need to start thinking like a hacker when designing, managing and protecting these valuable and vulnerable integrations. To reduce the attack surface, organisations must first identify the threat vectors with the same visibility as an attacker, starting with an external view from a public network mapping tool. Once the threat landscape is known, the use cases and users for every API must be defined and appropriate security measures put in place. Finally, recognising the dynamic nature of API development and threats, a layered security approach reduces cyber security risk. Like a fresh pot of coffee brewing, keeping good traffic flowing and filtering out bad actors, a secure API will not let another perfectly good cup go to waste.

References

1. ProgrammableWeb, available at https://www.programmableweb.com/apis/directory (accessed 21st February, 2020).
2. Bussa, T., Young, G., Girard, J., Zumerle, D., O'Neill, M., Orans, L., Hils, A., D'Hoinne, J. and Perkins, E. (November 2017), 'Predicts 2018: Infrastructure Protection', Gartner, available at https://www.gartner.com/en/documents/3830086 (accessed 21st February, 2020).
3. Hartmans, A. (March 2018), 'It's impossible to know exactly what data Cambridge Analytica scraped from Facebook — but here's the kind of information apps could access in 2014', Business Insider, available at https://www.businessinsider.com/what-data-did-cambridge-analytica-have-access-to-from-facebook-2018-3?utm_content=buffer069cc&utm_medium=social&utm_source=facebook.com&utm_campaign=buffer-bi (accessed 22nd August, 2019).
4. Whittaker, Z. (June 2019), 'Millions of Venmo transactions scraped in warning over privacy settings', TechCrunch, available at https://techcrunch.com/2019/06/16/millions-venmo-transactions-scraped/ (accessed 22nd August, 2019).
5. McKeay, M., Fakhreddine, A. and Ragan S. (February 2019), '[State of the Internet] / Security: Retail Attacks and API Traffic', Vol. 5, No. 2, p. 16, Akamai, available at https://www.akamai.com/us/en/multimedia/documents/state-of-the-internet/state-of-the-internet-security-retail-attacks-and-api-traffic-report-2019.pdf (accessed 21st February, 2020).
6. *Ibid.*, note 5.
7. *Ibid.*, note 6, p. 15.
8. *Ibid.*, note 5, p. 13.
9. *Ibid.*, note 5, p. 13.
10. McKeay, M. and Fakhreddine, A. (September 2018), '[State of the Internet] / Security: Credential Stuffing Attacks', Vol. 4, No. 4. p. 13, available at https://www.akamai.com/us/en/multimedia/documents/state-of-the-internet/soti-2018-credential-stuffing-attacks-report.pdf (accessed 21st February, 2020).
11. Barnett R. (April 2018), 'The Dark Side of APIs, Part 2', The Akamai Blog, available at https://blogs.

akamai.com/sitr/2018/04/part-2-the-dark-side-of-apis.html (accessed 23rd August, 2019).
12. See NMAP.org, available at https://nmap.org/ (accessed 19th May, 2020).
13. NMAP (September 2019), 'Nmap Network Scanning: Legal Issues', available at https://nmap.org/book/legal-issues.html (accessed 5th September, 2019).
14. Porup, J. M. (November 2019), 'What is Shodan? The search engine for everything on the internet', CSO, available at: https://www.csoonline.com/article/3276660/what-is-shodan-the-search-engine-for-everything-on-the-internet.html (accessed 21st February, 2020).
15. Raga Hines, L. and Wasserman, D. (November 2018), 'Hidden in Plain Sight: The Tools and Resources Used in Credential Abuse Attacks', p. 13, White Paper, Akamai, available at https://www.akamai.com/us/en/multimedia/documents/white-paper/credential-abuse-analysis-white-paper.pdf (accessed 21st February, 2020).
16. *Ibid.*, note 15, p. 20.
17. *Ibid.*, note 15, p. 19.
18. McKeay, M, Fakhreddine, A. and Ragan S. (January 2019), '[State of the Internet] / Security: DDoS and Application Attacks', Vol. 5, No. 1, pp. 11–14, available at https://www.akamai.com/us/en/multimedia/documents/state-of-the-internet/state-of-the-internet-security-ddos-and-application-attacks-2019.pdf (accessed 21st February, 2020).