Web Application Firewalls and Ways of Seeing Imperfect Tools

Andrew Zitek
New York University
New York, USA
e-mail: alz236@nyu.edu

Aspen Olmsted
Fisher College
Boston, USA
e-mail: aolmsted@fisher.edu

Abstract— **Common wisdom on how to evaluate preventative goods is weak, and as a result cybersecurity suppliers provide tools without hard evidence or guarantees. While it may be naive to expect any one tool to act as a silver bullet, information asymmetry is a problem that can and should be addressed. We argue that well-informed consumers are essential to responding to the security, privacy, and usability challenges associated with developing web applications hosted in the cloud. Accordingly, we study Web Application Firewalls to draw attention to the status quo, and provide questions that allow the public to readily identify information asymmetry in the goods they consider.**

Keywords- application firewalls; secure coding;

I. INTRODUCTION

A number of market studies indicate the demand for Web Application Firewalls (WAFs) is increasing rapidly [1]-[4]. At the same time, the InfoSec community readily offers concrete examples of how to carry out attacks on systems protected by WAFs [5]-[8]. We are confused by these two observations. Do consumers understand the extent of the limitations of their tooling? Are better options not available? Are they obligated the purchase by compliance requirements? The modern cybersecurity consumer faces many challenges. We argue that a broad survey of the WAF landscape will serve as a means to identify the paradigms with which researchers should equip consumers, so they make prudent and informed decisions.

A quick internet search will show that much published research on WAFs focuses on measuring and improving specific aspects of attack detection via involved techniques like machine learning [9]-[12]. Although none of the authors say so directly, the papers offer the impression that researchers are well aware that WAFs are flawed and that energies are focused narrowly on making these flaws smaller. While we agree that novel techniques may in the end improve these tools, we find it implausible that WAFs will ever provide the same protection as bug-free code. We'll support this theory and explain why you should care in later sections. First though, we'll step back and ask the natural question, what problems are WAFs actually intended to solve?

A good challenge for readers would perhaps include exploring a few vendor sites and, using only the information there, explain the purpose of WAFs. We found this task somewhat onerous, but in good faith we'll offer the following non-comprehensive list of uses: (1) protect applications, (2) detect attacks, (3) provide reporting and (4) meet compliance [13]-[16]. Upon compiling this list of uses, we found something to admire in each—they represent genuine concerns that consumers need to address and for which they seek out solutions. On closer inspection, however, we wondered how one could quibble with such broad objectives? Were they so broad as to be rendered meaningless? We find that savvy consumers are left still wondering a number of questions. First, how do WAFs accomplish their intended purpose? Second, to what extent do WAFs actually solve the problems that vendors claim they solve? Third, are WAFs in particular better suited to address these problems than other tools or processes?

Some cybersecurity specialists have argued that Payment Card Industry (PCI) requirement 6.6 explains the proliferation of WAFs without necessarily answering these questions. Requirement 6.6 states that organizations must either (1) use an application firewall *or* (2) implement a process for code reviews [17]. Wicket offers the somewhat critical conclusion that, given the unappealing nature of the second option, most organizations read this as a WAF mandate [18]. His argument is that organizations don't install WAFs for their security value, but instead out of a desire to pass their mandatory PCI certification. While we agree that PCI probably does drive some demand for WAFs, we disagree that this alone could explain such high demand for WAFs. This is simply due to the fact that a vast number of organizations don't actually pursue PCI certification. We considered the possibility that organizations look to PCI as a defacto standard, essentially "if it's good enough for banks it's good enough for us." We would be more inclined to expand on that theory, however, provided more evidence. Our debate of PCI is, in fact, addressing a larger matter—that some in the cybersecurity community believe it is safe to forgo the proactive process of removing bugs from code as long as one installs some type of reactive tool like a WAF. This is at best misleading and at worst wrong.

Other popular channels of information, like Wikipedia, are more realistic in their description of WAFs, but in our opinion, are not without problems. Although Wikipedia does give some matter of fact information such as, "By inspecting HTTP traffic, WAFs can prevent attacks stemming from web application security flaws, such as SQL injection, cross-site scripting, file inclusion, and security misconfigurations," it also makes hard-to-support claims such as, "The Open Web Application Security Project (OWASP) produces a list of the top ten web application security flaws. All commercial WAF offerings cover these ten flaws at a minimum [19]." We argue that less savvy readers may be misled into feeling a false sense of security due to the fact that the meaning of the word *coverage* is unclear. We have to ask, is this just a poorly worded sentence, or is it evidence of bona fide embellishment?

In their recent work, Muegge and Craigen have offered the conclusion that cybersecurity specialists manipulate cognitive limitations to over dramatize and oversimplify risks [20]. Essentially, Muegge and Craigen maintain that, because there is a lack of reliable information around cybersecurity, processes should be anchored around what they call "evidence-based design principles." We agree that it's not easy to find reliable data, or data that's not oversimplified, in cybersecurity because our experience researching WAFs confirms it. Muegge and Craigen's theory on the absence of quality information is extremely useful because it sheds light on the problem of how difficult it is for consumers to make well-informed decisions without sufficient evidence.

At this point we would like to raise some objections inspired by our own internal skepticism. We feel that we may have been ignoring the fact that eliminating risk entirely is considered impossible. "Tools will never be perfect", we say, "we should reduce harm in any ways we can afford." Cybersecurity specialists in particular will note that the goal is less about perfection and more about reducing risk. Our point is not that we should cast aside tools simply because they're not perfect. Our point is that if suppliers are not offering a guarantee for their claims about the quality of services provided, consumers should be given information that lets the cold sting of these limitations sink in.

We are not the first to make the connection between cybersecurity tools and Akerlof's Market for Lemons [21]-[23]. Putting to use the example of used car sales, Akerlof famously put forth that quality will degrade in markets where it is not possible for consumers to validate the quality of goods being offered [24]. He maintains that these markets lead to weary consumers, willing to pay only lower prices for specific classes of goods no matter the quality [25]. Still more interesting, others have made the claim that information asymmetry has been solved in the market of used cars by guarantees like pre-certified used car programs and reputable third-party quality information sources like Carfax [26]. Arguments like this make us optimistic about the future, and we would like to see efforts toward analogous solutions for the problems of information asymmetry in markets for cybersecurity goods.

During the course of the COVID-19 pandemic, firms capable of working in the cloud have benefited, and those yet to shift to the cloud are accelerating plans to do so [27]. As the cloud continues to prove itself essential, the selection processes consumers use for tools to secure applications run in the cloud grows proportionally. We encourage researchers to acknowledge these trends and focus on addressing security, privacy, and usability challenges with solutions that lead to well informed consumers.

The organization of the paper is as follows: Section II reviews work related to assessing WAFs, and we provide a motivating example along with explanations of our empirical evidence. Section III provides discussion of our solution—a mental paradigm for savvy consumers. We conclude and describe future work in Section IV.

II. Measuring the Extent To Which WAFs Solve Problems

Many assume that the capability of WAFs to analyze and filter requests at the application level is new technology. In fact, application-level access control systems that embody the firewall design have existed since at least 1998 [28]. In these systems, depicted as a flow diagram in Figure 1, just like in traditional network firewalls a special intermediate server establishes a barrier between a trusted internal domain and an untrusted external domain. These self-contained, generally configurable firewalls provide a chokepoint from which a policy of security rules may be enforced with the intent of denying suspicious traffic while allowing other credible seeming traffic. Toward this goal, a negative or positive security model can be used as a basis for access decisions. We focus only on the negative security model, as we have found this to be more popular by far, likely due to the fact that it requires little manual configuration by administrators when compared with the positive security model. We construct a basic threat model for this generic system using the STRIDE methodology in Table I.

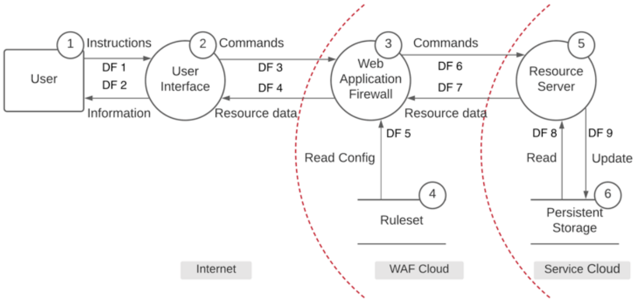

Figure 1. Data Flow Diagram

TABLE I. STRIDE Threat Analysis of Figure I

Data Flow Diagram Element	*S*	*T*	*R*	*I*	*D*	*E*
1) User	✓		✓			
2) User Interface	✓	✓	✓			
3) Web Application Firewall	✓	✓	✓	✓	✓	
4) Ruleset					✓	
5) Resource Server	✓	✓	✓	✓	✓	✓
6) Persistent Storage	✓	✓	✓	✓	✓	✓
7) DFs 3-4, 6-9		✓		✓	✓	

There are numerous closed- and open-source initiatives attempting to provide tools for measuring the performance of WAFs, most with an emphasis on regression testing [29]-[31]. Azaria and Shulman, affiliated with Imperva, presented a methodology for assessing the performance of WAFs with a focus on two qualities: legitimate traffic that is blocked, and malicious traffic that is not blocked [31]. In their benchmark analysis, it was demonstrated that in the set of sample requests shown in Table II, there existed no instance of legitimate traffic that was blocked and there existed many instances of malicious traffic that was not blocked. This presentation is instructive because it sheds light on the fact that the complexities of binary classification systems are central to the issues that WAF developers face. Using this information, we can speculate that False Negatives are preferred over False Positives, probably because they do not cause service interruptions for clients using the WAF.

TABLE II. BENCHMARK OF CLASSIFICATION OF ATTACKS BY WAF

Attack Type	*Total Attacks*	*Misclassified*	*%*
False Negative	67	67	100
False Positive	148	0	0

a. False Negative Attacks are malicious requests that should be blocked

b. False Positive Attacks are legitimate requests that should not be blocked

We may expand on this speculation with a theoretical example. Consider the situation given in Figure 2 when 2 percent of all traffic received by a web server is malicious. We then integrate a WAF that returns a positive classification result 95 percent of the time for requests that are actually malicious. If a request is not malicious, the WAF returns a negative classification result 99 percent of the time.

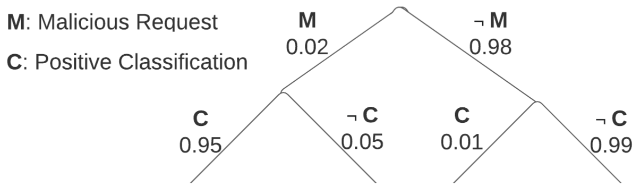

Figure 2. Tree Diagram of Request Classification

If the WAF returns a positive classification for a request, the probability the request is actually malicious is given by (2),

$$\begin{aligned} P(C) &= (0.02)(0.95) + (0.98)(0.01) \\ &= 0.029 \end{aligned} \tag{1}$$

$$\begin{aligned} P(M|C) &= P(M \cap C) / P(C) \\ &= (0.02)(0.95) / 0.029 \\ &= 0.655 \end{aligned} \tag{2}$$

If the WAF returns a negative classification for a request, the probability the request is actually not malicious is given by (4),

$$\begin{aligned} P(\neg C) &= (0.02)(0.05) + (0.98)(0.99) \\ &= 0.971 \end{aligned} \tag{3}$$

$$\begin{aligned} P(\neg M|\neg C) &= P(\neg M \cap \neg C) / P(\neg C) \\ &= (0.98)(0.99) / 0.971 \\ &= 0.999 \end{aligned} \tag{4}$$

We remark that the sensitivity we assigned to our WAF in this example is high and would drastically reduce the *total number* of malicious requests received by the webserver. For a webserver responding to 1 million requests daily this would result in a 95 percent decrease from 20K to 1K malicious requests. While this improvement, if reflective of real-world scenarios, seems encouraging, we are left with the thought that a webservice absent of its own well-designed security mechanisms processing 1K malicious requests on a daily basis seems far from secure. Basically, it seems that no matter how advanced the sensitivity of our WAF, the reality appears that our webservice will always have the responsibility to respond appropriately to a nonzero number of malicious requests. This, in a nutshell would imply that the benefits of WAFs are strictly supplementary and not substitutionary.

Bonneau, Herley, Oorschot, and Stajano describe the use of passwords for authentication purposes as an, at first, seemingly analogous situation where organizations settle for more lax security policies involving binary classification systems, due to usability challenges [32]. They offer the fairly sympathetic argument that organizations do not expect to achieve ironclad invulnerability, so they instead seek only to reduce harm at acceptable cost. We agree that this is the status quo and we believe, in the case of authentication strategies, that this compromise seems justified because it is likely to impact only individual users rather than an organization as a whole. We find this reminder helpful because it sheds light on the fact that consumers are in fact very accustomed to making compromises in their security strategies.

If we had an imaginary dial for the sensitivity of a system of authentication signals, we could imagine, at the highest setting, many users would have a hard time logging in, but fewer accounts would be hijacked. As we lowered this imaginary setting, we could foresee these numbers shifting until, at the lowest setting, few users would have problems authenticating, but most accounts could be easily hijacked. Despite having sensitivity configurations, WAFs simply do not provide an analogous type of tradeoff because application-specific implementation bugs can lead to all-or-nothing types of attacks, for example database dumps instead of attacks that only impact specific users. We find that making this distinction is essential. There is no blanket one-size-fits-all policy suggesting what compromises to security strategies are favorable.

In the remainder of this section we will refute the idea that WAFs provide coverage of vulnerabilities created by application-specific implementation bugs. We have elected to test injection attacks, because they represent the number one web application security risk per the OWASP Top Ten 2017 list [33]. Additionally, we document things of interest that arise in the process of integrating the WAF with our basic webservice.

A. Technology Stack

We use the following tools to conduct this experiment.

- IBM Public Virtual Server C1
- Ubuntu 18.04-64
- Node.js 12.14.1
- MySQL 5.7.28
- Cloudflare Web Application Firewall

LISTING I. VULNERABLE CODE SNIPPET

```
const userInput = req.query.itemID;
const statement = `
  SELECT
    ItemID,
5     ItemName,
6     ItemDescription
7   FROM Items
8   WHERE ItemID = ${userInput};
9  `;
10
11 connection.query(
12   statement,
13   function callback(err, rows) { … }
14 );
```

In Listing 1, representing a snippet of the vulnerable code, on line 8 an unsanitized user input is interpolated into the string representing the SQL statement. This bug represents the source of the vulnerability we will use to test the WAF.

B. *Attacks*

We start with three basic SQL-injection (SQLi) attacks [34] and enhance each version by applying a technique called obfuscation [5].

i. Basic Tautology – the goal of tautology is to inject SQL tokens that cause the conditional statement of a query to evaluate true, like

```
GET /items?itemID=1 or 1=1
```

ii. Basic Union Query – the goal of a union query is to manipulate the where clause of a query so that multiple sub-queries can be made in addition to the one the programmer intended, like

```
GET /items?itemID=1 UNION SELECT UserID,
UserName, UserPassword FROM Users
```

iii. Basic Piggyback Query – the goal of a piggyback query is to exploit a misconfiguration where it is sometimes possible to append a query to another query, like

```
GET /items?itemID=1; DROP TABLE Users
```

iv. Obfuscated Tautology – the goal of this obfuscation is to use quotation marks to trick the WAF into thinking the attack is legitimate traffic, like

```
GET /items?itemID=1 OR 1#"OR"’OR’’=’"="’OR’’=’
```

v. Obfuscated Union Query – the goal of this obfuscation is to use different encodings to trick the WAF into thinking the attack is legitimate traffic, like

```
GET /items?itemID=1
union%23foo*%2F*bar%0D%0Aselect%23foo%0D%
0A UserID,UserName, UserPassword+FROM+Users
```

vi. Obfuscated Piggyback Query – we can use similar techniques for piggyback queries, like

```
GET /items?itemID=1; +DROP%20TABLE%20Users
```

In further consideration of the STRIDE classification model, the tautology and union query attacks represent information disclosure threats, while the piggyback query represents a tampering threat. In the DREAD threat rating methodology, SQLi attacks are given the highest possible score of ten out of ten [35]. These styles of attacks are prolific, decades-old and have impacted significant players like the World Health Organization and the Wall Street Journal [36].

C. *Integration*

We find some snafus encountered during the integration process noteworthy. At first, the process of activating the WAF appeared to involve updating our DNS provider and clicking a button next to our CNAME entry to turn an icon from grey to color. We were unsure what to think when, at first, all of our attacks succeeded. We reviewed the configuration settings in the provided dashboard several times and, after a few days, contacted customer support. Customer support explained that the service tier we were using would protect against only DDoS attacks, not OWASP sourced attacks like the ones we were testing.

Later, we upgraded our service tier and ran our tests a second time. Again, all of our attacks succeeded. We returned to our configuration settings and discovered that upon upgrading plans, new options had become available and, by default, were not active. After toggling these to active, we, at last, observed our first blocked attack. Still, we later uncovered more configurations for the sensitivity of the WAF. All tests in the next section were performed with the sensitivity set to the highest possible setting. These snafus may represent human-usability issues and demonstrate how a pivotal ingredient to usable cybersecurity is informative feedback, especially visibility of the system state [37]. Basically, integrating a WAF adds a nontrivial level of operational complexity to a system, and this is a drawback because it can sometimes make it difficult to measure the security integrity of a system.

D. *Results*

The results of the experiment, provided in Table III, concluded that the WAF is unable to guarantee protection from the risk of injection attacks caused by application-specific bugs. A trivial level of obfuscation makes it possible for an adversary to succeed at all three flavors of the attacks tested. This result makes us doubt the significance of the calculations made in Section II. At first, the possibility that under certain conditions we could reduce the total number of malicious requests received by a webservice seemed promising, but in retrospect, when there still exists in reality a nonzero number of *known* attacks that the WAF does not correctly classify, it is not straightforward to describe what benefit this would provide, if any.

TABLE III. CLASSIFICATION OF ATTACKS ON VULNERABLE WEBSERVICE BY WAF

Attack Class	*True-negative*	*False-negative*	*Misclassified*
B. Tautology	yes	no	no
B. Union	yes	no	no
B. Piggyback	no	yes	yes
O. Tautology	no	yes	yes
O. Union	no	yes	yes
O. Piggyback	no	yes	yes

a. For each of the six attack classes we send one request in order to observe the result. Because each of the six instances represent a malicious request, each should result in a True-negative outcome. We label all requests with different outcomes as Misclassified.

We contacted Cloudflare customer support and provided the obfuscated versions of each example attack along with links to a live server for demonstration purposes. A customer support representative communicated that the keywords we provided were, "not a combination we have connected to an active software vulnerability we are ware [sic] of currently." The representative suggested that we create a custom ruleset to block these exact requests from our system using the web interface. The same representative later added that, "for our global rulesets we need to balance coverage and avoiding false positive(s) from over agrresive [sic] rules in our network." We find that this commentary further supports the hypothesis made in this paper regarding inherent weaknesses of systems involving binary-classification. In the end, a different customer support representative in the same conversation wrote, "our WAF Engineering team will add the first two examples to our WAF engine so this will be picked up by Cloudflare WAF rules. I am afraid we are not yet on a position to provide you with a [sic] ETA but it will be taken care of soon." Another representative later reiterated that they were unable to share further details regarding how or when these changes would take effect.

E. Guaranteeing Protection

We will briefly demonstrate the effort involved in patching the application bug using secure coding. We know where the bug resides in our source code because we designed it intentionally. We are aware of course, that the writers of applications do not always know about the bugs in their code.

The patch will involve changing two lines of code, lines 8 and 13, to leverage a technique called parameterized queries, or prepared statements. Parameterized queries guarantee protection from SQLi attacks by ensuring that the SQL engine parses and compiles the query separately from the variables. The variables are escaped and inserted into the query later, so that no matter their content, they will be interpreted as ordinary strings [38].

LISTING 2. PATCHED CODE SNIPPET

```
1  const userInput = req.query.itemID;
2  const statement = `
3    SELECT
4      ItemID,
5      ItemName,
6      ItemDescription
7    FROM Items
8    WHERE ItemID = ?;
9  `;
10
11 connection.query(
12   statement,
13   [userInput],
14   function callback(err, rows) { … }
15 );
```

In Listing 2, representing a snippet of the patched code, a placeholder is put in line 8 indicating that the second argument to the query function on line 13 will contain the variable that should be escaped and inserted into the query after it has been parsed and compiled.

After our modifications, the attacks are unsuccessful at tampering with the integrity of the database and disclosing information additional to what the author intended. This solution is low effort and highly effective but depends on knowledge.

III. A MENTAL PARADIGM FOR THE SAVVY CONSUMER

To paraphrase John Berger on art, it isn't so much the WAFs we want to consider, but the ways we see them [39]. Essentially, our point is not to convince consumers to reject tools like WAFs because they are imperfect. Our point is to convince consumers that they must resist the potential peace of mind and assurance that comes with preventative goods like WAFs. These delusions may become reasons to not carry out other prudent behaviors.

Although much of this paper may make this idea seem obvious, we argue that, in fact, it's difficult for consumers to recognize the extent to which the position they hold in the market for cybersecurity tools lacks quality information. As we have discussed, while organizations are desperate for meaningful solutions, suppliers offer tools without guarantees and it is difficult to research credible information on the quality of tools offered. In situations like these, we wish to provide a paradigm allowing consumers to readily identify information asymmetry in the goods they consider. Due to the nature of cybersecurity tools we will focus specifically on preventative goods that aim to forestall negative outcomes.

We are aware that in economics, goods are often given labels when they exhibit particular qualities that make them special. In the case of luxury, or Veblen goods for instance, demand can appear to increase as price increases contradicting the law of demand [40]. In this close study of tools like WAFs, it is possible to make the argument that many cybersecurity products embody their own unique set of characteristics, and we have yet to discover an economic term for this type of good. These unique properties are:

1. You pay for it hoping to stop something undesirable
2. If you observe nothing, you might assume it worked
3. If you observe anything, you will know it did not

Standing alone, we think these observations may not seem striking, so, in an attempt to promote sticky mental associations between the domain of our problem and the solution, we surveyed a few students and colleagues, asking them to name a familiar product that has these characteristics. The list below represents the responses. The entries do not necessarily reflect our own opinions.

- Flu Vaccinations
- Vaccinations (Other)
- Vitamin C Supplements
- Supplements (Other)
- Surgical Masks
- Mosquito Repellent
- Pest Extermination Service
- Antivirus Software
- Anti-Aging Treatments (Beauty Industry)
- Contraceptives
- Light Therapy Lamps
- "Paying off the mob"
- "A rock that keeps tigers away"

To make it clear, this paper has no interest in making arguments for nor against any of these goods. The observation that many of these goods are controversial however, is interesting because it sheds light on the fact that goods with the particular qualities highlighted above may present special challenges for consumers. Basically, we argue that thinking about a few preventative goods that consumers are already familiar with may enable us to more quickly grasp the challenges present in markets for cybersecurity tooling. Complexity of subject matter, lack of data, supplier reputation, industry regulations and social pressure appear to be key factors that these markets share in common.

In the end, we cannot provide a blanket prescription regarding whether or not organizations should use preventative tools like WAFs to protect their cloud hosted web applications. What we can do is ask the consumer an analogous question like, do you think you should take a vitamin C supplement to prevent illness? To what extent does the supplement prevent you from getting sick? How will you know? Specifically, how will you measure whether the claims the supplement supplier makes are true using valid data? If you cannot obtain the data needed to make this analysis, will the supplier provide you a guarantee? Ultimately, if you have a few extra dollars, and taking a supplement would give you peace of mind, the negative impacts of doing so, on the surface, seem low, but that's no excuse to not wash your hands in the first place.

IV. Conclusion

In this paper, we address the problem of how to assess preventative goods. We argue that consumers are left to trust suppliers who provide imperfect technology for cybersecurity without guarantees. In this paper, we evaluate problems with WAFs and how they can be compared and contrasted. We utilize the STRIDE threat model in an applied experiment on a WAF analyzing a SQLi attack. Our conclusion is that small changes in configuration can lead to very different results with the tooling and implementation knowledge is currently the most important ingredient in the equation. Our future work will calculate a measure of dependency on outside knowledge that is required for individual cybersecurity tools.

References

[1] CISOMAG. Web application firewall market worth $5.48 Billion by 2022. [Online]. Available from: https://www.cisomag.com/web-application-firewall-market-worth-5-48-billion-2022/ 2020.02.27

[2] MarketWatch. Web Application Firewall Market Research Reports 2019. [Online]. Available from: https://www.marketwatch.com/press-release/web-application-firewall-market-research-reports-2019-global-industry-size-in-depth-qualitative-insights-explosive-growth-opportunity-regional-analysis-by-market-reports-world-2019-06-14 2020.03.16

[3] C. Rodriguez. Web Application Firewall (WAF) Global Market Analysis New Technologies and Threats Collide to Create Expanded Opportunities. [Online]. Available from: https://www.akamai.com/us/en/multimedia/documents/content/frost-sullivan-web-application-firewall-global-market-analysis-research-excerpt-report.pdf 2020.03.16

[4] KBV Research. Web Application Firewall Market Size. [Online]. Available from: https://www.kbvresearch.com/web-application-firewall-market/ 2020.03.18

[5] R. Salgado. SQL Injection Optimization and Obfuscation Techniques. [Online]. Available from: https://media.blackhat.com/us-13/US-13-Salgado-SQLi-Optimization-and-Obfuscation-Techniques-WP.pdf 2020.03.20

[6] Z. Allen. WAFs FTW: A Modern DevOps Approach to Security Testing your WAF. [Online]. Availale from: https://www.youtube.com/watch?v=05Uy0R7UdFw 2020.08.20

[7] V. Ivanov. Web Application Firewalls: Analysis of Detection Logic. [Online]. Available from: https://www.youtube.com/watch?v=dMFJLicdaC0 2020.08.20

[8] I. Schmitt and S. Schinzel. WAFFLe: Fingerprinting Filter Rules of Web Application Firewalls. [Online]. Available from: https://www.usenix.org/conference/woot12/workshop-program/presentation/schmitt 2020.08.20

[9] A. Moosa, “Artificial Neural Network based Web Application Firewall for SQL Injection” World Academy of Science, Engineering and Technology International Journal of Computer and Information, 2010, pp. 12-21, ISSN: 2010-3778

[10] K. Demertzis and L. Iliadis, “Cognitive Web Application Firewall to Critical Infrastructures Protection from Phishing Attacks” Journal of Computations & Modelling, 2019, pp. 1-26, ISSN: 1792-8850

[11] D. Appelt, C.D. Nguyen and L. Briand, "Behind an application firewall, are we safe from SQL injection attacks?" IEEE 8th International Conference on Software Testing, Verification and Validation, May 2015, 10.1109/ICST.2015.7102581

[12] A. Makiou, Y. Begriche and A. Serhrouchni, "Improving Web Application Firewalls to Detect Advanced SQL Injection Attacks" International Conference on Information Assurance and Security (lAS), Mar. 2014, pp. 35-40, 10.1109/ISIAS.2014.7064617

[13] F5 Security Products. Advanced Web Application Firewall (WAF). [Online]. Available from: https://www.f5.com/products/security/advanced-waf 2020.03.2020

[14] Cloudflare. Web Application Firewall. [Online]. Available from: https://www.cloudflare.com/waf/ 2020.03.18

[15] Imperva. Web Application Firewall (WAF). [Online]. Available from: https://www.imperva.com/products/web-application-firewall-waf/ 2020.03.18

[16] Trustwave Managed Security. Managed Web Application Firewall. [Online]. Available from: https://www.trustwave.com/en-us/services/managed-security/managed-web-application-firewall/ 2020.03.18

[17] Security Standards Council. Payment Card Industry Data Security Standard (PCI DSS) Requirement 6.6 Code Reviews and Application Firewalls. [Online]. Available from: https://www.pcisecuritystandards.org/pdfs/infosupp_6_6_applicationfirewalls_codereviews.pdf 2020.08.20

[18] J. Wickett. Three Ways Legacy WAFs Fail, Signal Sciences Blog. [Online]. Available from: https://www.signalsciences.com/blog/three-ways-wafs-fail/ 2020.03.05

[19] Internet Archive WayBack Machine. [Online]. Available from: https://web.archive.org/web/20161104030043/https://en.wikipedia.org/wiki/Web_application_firewall 2020.08.20

[20] S. Muegge and D. Craigen, "A Design Science Approach to Constructing Critical Infrastructure and Communicating Cybersecurity Risks" Technology Innovation Management Review, Jun. 2015, vol. 5, pp. 6-16, ISSN 19270321

[21] M. Lesk, "Cybersecurity and economics" IEEE Security & Privacy, Nov. 2011, vol. 9, pp.76-79, 10.1109/MSP.2011.160

[22] T. Moore, "The economics of cybersecurity: Principles and policy options" International Journal of Critical Infrastructure Protection, Dec. 2010, vol. 3, pp. 103-117, 10.1016/j.ijcip.2010.10.002

[23] C. Dacus and P. Yannakogeorgos, "Designing Cybersecurity into Defense Systems: An Information Economics Approach" IEEE Security & Privacy, May. 2016, vol. 14, pp. 44-51, 10.1109/MSP.2016.49

[24] G. Akerlof. "The Market for 'Lemons': Quality Uncertainty and the Market Mechanism" The Quarterly Journal of Economics, Aug. 1970, vol. 84, pp. 488-500, ISSN 1531-4650

[25] Wikipedia. The Market for Lemons. [Online]. Available from: https://en.wikipedia.org/wiki/The_Market_for_Lemons#Criticism 2020.02.29

[26] D. Maimon. An Evidence Based Cybersecurity Approach to Risk Management: Risk Management and "Market for Lemons". [Online]. Available from: https://scholarworks.gsu.edu/cgi/viewcontent.cgi?article=1000&context=ebcs_presentations 2020.08.20

[27] J. Rubenstone. Cloud Infrastructure Keeps Firms Afloat During Coronavirus Pandemic. [Online]. Available from: https://www.enr.com/articles/48975-cloud-infrastructure-keeps-firms-afloat-during-coronavirus-pandemic 2020.08.20

[28] R. Hunt, "Internet/Intranet firewall security—policy, architecture and transaction services" Computer Communications (Elsevier), Sep. 1998, vol. 21, pp. 1107-1123, ISSN 0140-3664

[29] Microsoft. WAFBench. [Online]. Available from: https://github.com/microsoft/WAFBench 2020.08.20

[30] Fastly. FTW. [Online]. Available from: https://github.com/fastly/ftw 2020.08.20

[31] Y. Azaria and A. Shulman. WTF - WAF Testing Framework. [Online]. Available from: https://www.youtube.com/watch?v=ixb-L5JWJgI 2020.08.20

[32] J. Bonneau, C. Herley, P. Van Oorschot and F. Stajano, "Passwords and the evolution of imperfect authentication" Communications of the ACM, Jul. 2015, vol. 58, pp. 78-87, 10.1145/2699390

[33] OWASP Foundation. OWASP Top Ten. [Online]. Available from: https://owasp.org/www-project-top-ten/ 2020.03.26

[34] S. Shanmughhaneethi, S. Shyni and S. Swamynathan, "SBSQLID: Securing Web Applications with Service Based SQL Injection Detection" International Conference on Advances in Computing, Control, and Telecommunication Technologies, Dec. 2009, pp. 702-704, 10.1109/ACT.2009.178

[35] P. Carter, "Threat Analysis and Compliance," Securing SQL Server: DBAs Defending the Database, 2018, Apress, pp. 12–16

[36] J. Cox. The History of SQL Injection, the Hack That Will Never Go Away. [Online]. Available from: https://www.vice.com/en_us/article/aekzez/the-history-of-sql-injection-the-hack-that-will-never-go-away 2020.02.22

[37] J. Nurse, S. Creese, M. Goldsmith and K. Lamberts, "Guidelines for usable cybersecurity: Past and present" Third International Workshop on Cyberspace Safety and Security (CSS), Sep. 2011, pp. 21-26, 10.1109/CSS.2011.6058566

[38] A. Sadeghian, M. Zamani and S. Ibrahim, "SQL Injection is Still Alive:A Study on SQL Injection Signature Evasion Techniques" International Conference on Informatics and Creative Multimedia (ICICM), Sep. 2013, pp. 265-268, 10.1109/ICICM.2013.52

[39] J. Berger. Ways of Seeing Episode 1. [Online]. Available from: https://www.youtube.com/watch?v=0pDE4VX_9Kk 2020.08.20

[40] Wikipedia. Veblen Good. [Online]. Available from: https://en.wikipedia.org/wiki/Veblen_good 2020.08.20