Chapter 1

Introduction

The Web is an essential part of our individual everyday lives as well as societal activities as a whole. With advancements in technology, even the most complex applications are being increasingly delivered over the Web. However, with the proliferation of services being provided, there arise crucial questions. How secure is the Web? How secure are we when we access a resource on the Web? Answers to such questions have a single pointed focus - Security at all levels of interaction on the Web [1].
With monumental advances in Cloud Computing data security is one of the main aspects to maintain the privacy, confidentiality, integrity and authority of the users and the Web Application they use[2]. Security plays a significant role in all Web applications. Applications coupled with respective Web servers provide a wide range of useful services to the users. However, there exists a particular sub-population of advanced users who incessantly exploit the potential vulnerabilities in Web applications and servers, which are literally hubs of personal communication and information for numerous users. Typically, the first step in launching a damaging attack is to discover security flaws in an application and later leverage the flaws to gain sensitive information.

1.1 The Web and Its Users

The web has introduced a fundamental world-wide revolution with an escalation of websites and applications delivered for use. Usage of the Web has seen tremendous increase over the years and the most contributing factor is the increase of individ-

uals and organizations willing to share their data in a global space[3]. Deluge of information created, shared and consumed everyday play a vital role in making decision in the daily lives. This is because websites and applications provide a variety of indispensable services to its user base[4]. To use those services the users need to share their valuable personal information so as to customize the services provided according to their needs. The interactions over the Web however, may not always be secure. This is because not all users of the Web are like minded and moral. According to Garfinkel and Spafford[5], the three main facets in Web security are:

1. Security relating to the Web server and its data.
2. Security relating to the interactions between the Web server and the end-user.
3. Security relating to the user's systems used to access the Web.

Security of the Web largely depends on the design structure of applications and sites delivered over the Web[5]. From the user's perspective, the design structure is crucial because shared sensitive information of the user base depends on it. On the other hand, from the application's perspective the design structure is important so that uninterrupted and reliable services can be provided to its users. The technology stack used to develop web applications and sites are continuously evolving making it difficult to document a particular set of rigid rules to be followed during the design phase. As a result some unintended logical flaws and exposures[1] may exist in the applications and sites. Nefarious users exploit these flaws to gain unauthorized access to the systems to carry out different kinds of *Web-based attacks* as discussed in the next section.

1.2 Web-based Attacks

As the modern Internet era promises to provide more and more services to the society, the risks to security and maintenance of sensitive information are becoming higher and higher. Weaknesses or flaws existing in the system are technically termed as Vulnerabilities[2]. Many vulnerabilities lurk in Web applications, often without the developer's knowledge[6]. User inputs which does not go through any input validation mechanism is a very common example of vulnerability. The Open

[1] `https://www.balbix.com/insights/what-is-a-cve/`

[2] `https://devoxsoftware.com/blog/10-common-web-application-vulnerabilities/`

Worldwide Application Security Project (OWASP)[3] maintains a list of vulnerabilities typically faced by a Web application[4]. On the other hand, the Common Vulnerabilities and Exposures (CVE) system[5] maintained by The MITRE Corporation [6] manages a database of publicly known security vulnerabilities in released software packages with unique identifier for each vulnerability and exposure. Malicious perpetrators take advantage of these vulnerabilities in Web applications, posing serious threats to the end users of the applications. As a result, the users of vulnerable applications fall prey to the attackers. It is important here to note that, the goal of the attacker may either be to steal sensitive information from the user base of the application, or disrupt the services provided by the application or gain unauthorized access to the system or even execute malicious software.
Figure 1.1 gives a depiction of the Vulnerability count of different types of vulnerabilities from the year 2014 to 2nd January, 2024 [7]. As it can be seen, Cross-site scripting (XSS) vulnerability has the highest count with a total count of 21,821 during the specified time, which makes approximately 26.4% of the total vulnerabilities reported.

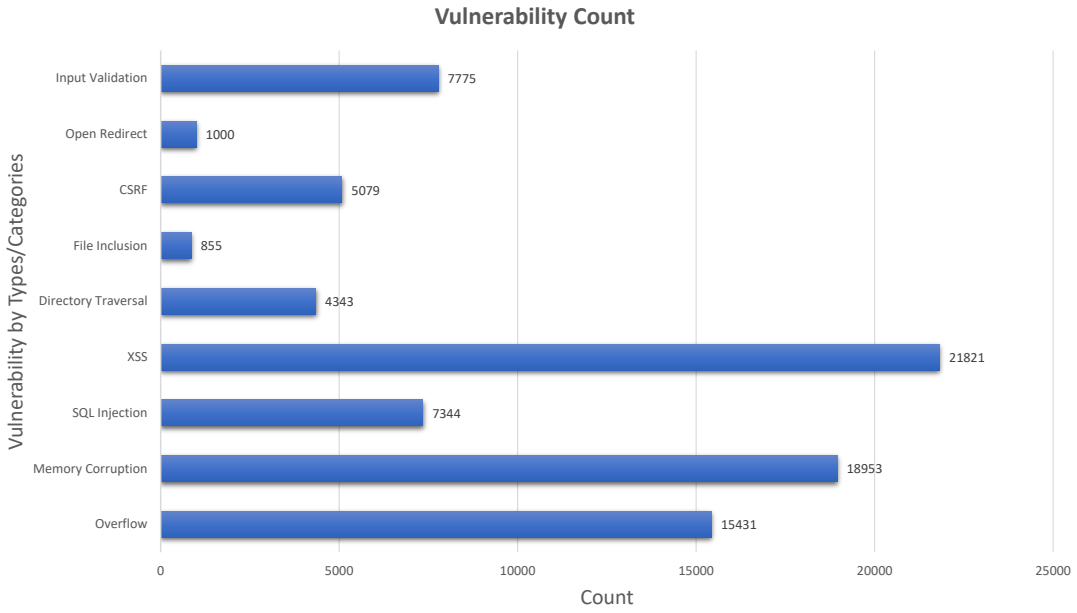

Figure 1.1: Vulnerability Count of Different Vulnerabilities

[3] https://owasp.org/
[4] https://owasp.org/www-community/vulnerabilities/
[5] https://www.cve.org/About/Process
[6] https://www.mitre.org/
[7] https://www.cvedetails.com/vulnerabilities-by-types.php

1.2.1 Steps in Launching a Web-based Attack

To launch an attack against a Web application, an attacker typically follows a series of steps, which are illustrated in Figure 1.2. Each of these steps are discussed in details below.

S1. Target Identification: The very first step is to identify the target to launch

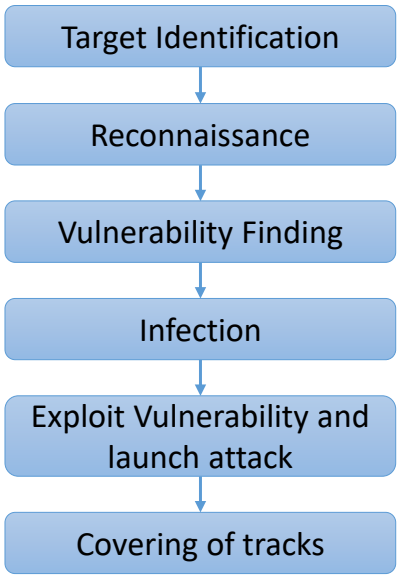

Figure 1.2: Typical Steps in a Web Attack

the attack. When choosing a target, the attacker may have different motives (as discussed in Section 1.5). Often, a high stake target is chosen so as to make higher gains.

S2. Reconnaissance: After the target is identified, the attacker tries to garner information about the target system which can be exploited later. In doing so, the attacker may or may not directly interact with the web application. If information is collected by directly interacting with the application it is called Active Reconnaissance, otherwise it is called Passive Reconnaissance. In passive reconnaissance, information may be gathered through social media, or other public platforms. Conversely, in Active Reconnaissance the attacker queries (or probes) the application. Technologies used, design structure of the application, software versions, potential miconfigurations (if any), IP Addresses, and domain names are some of the information that may be collected.

S3. Vulnerability Finding: Information gathered in the previous step can be used to find vulnerabilities in a web application. Scanning tools may be used to Crawl an application and map its structure, or any unlinked pages. Additionally, Static

Analysis and Dynamic Analysis can be useful to scan the web application for potential vulnerabilities, which can be exploited later to launch attacks. A few widely used tools available for scanning an application are: Zed Attack Proxy (ZAP)[8], Burp Suite[9], Web Application Attack and Audit Framework[10] (w3af), and Metasploit[11].

S4. Infection: After successfully finding the vulnerability in a system, the next step from the attacker's perspective is to infect the target. The attacker crafts an infection vector to get unauthorized access into the system. compromise the integrity of the system.

S5. Exploit Vulnerability and Launch Attack: Once the vulnerabilities are found and infection vector is crafted, next task is to exploit the vulnerability and launch the attack. Attacks may be launched in many ways such as social engineering techniques to entice the user or by crafting some clickable link.

S6. Covering of Tracks: After the attack is launched on the application, next task is to evade any detection mechanism in place. To do so, logs may have to be manipulated (so that unauthorized access cannot be detected), or other evidences (if any) may have to be erased so as to prolong the duration of the unauthorized access. Ultimately, the goal in this step is to conceal the attack history so that attacker cannot be traced back.

1.3 Types of Web-based Attacks

This section focuses on the types of attacks that can be carried out against a website or an Web application. Most of these attacks exploit some vulnerability to launch damaging attacks. Three Web-based attacks are chosen and the reason why they are chosen is three fold as discussed below.

1. *Attacks exploiting vulnerabilities in Web application:* A vulnerability in the web application is exploited by an attacker to execute his/her malicious payload in the end user's browser (client-side). Important here are the Code injection attacks that make use of untrusted code which propagates into and are utilized as code in the context of a Web application[7]. When executed,

[8] https://www.zaproxy.org/
[9] https://portswigger.net/burp
[10] http://www.w3af.org/
[11] https://www.metasploit.com/

such code potentially changes the intended behavior of an application. Cross-site scripting attacks are one example of such attacks in the Application layer of the Open Systems Interconnection (OSI) model which are introduced in Section 1.3.1 and discussed in detail in Chapter 2, section 2.5.1.

2. *Attacks exploiting vulnerability in application layer protocols:* Applications over the Web primarily rely on the HyperText Transfer Protocol (HTTP) protocol to transmit web pages to and from systems. A variety of application layer protocols exist, but among all HTTP is the most targeted because of its versatility and ease of integration with online services [8]. The attackers leverage the fact that no detection system by default blocks any HTTP traffic. Thus by exploiting such characteristics of the protocol attacks are launch against web applications. HTTP Flooding attacks are one such attacks in the application layer of the OSI model which are introduced in Section 1.3.2 and discussed in detail in Chapter 2, section 2.5.3.

3. *Disruption of critical services*: Web applications may provide critical services to a geographical region (for example: Water treatment facilities). Such applications when targeted by nefarious users cause havoc to the individuals or organisations of that region. From attackers perspective, the aim is to make a bigger impact by disrupting the services. This may be done by executing or gaining unauthorized access to the critical systems. Attacks on such Critical Infrastructure (CI) are introduced in Section 1.3.3 and discussed in detail in Chapter 2, section 2.5.2

Additionally, some real life attack examples and attack trends are also presented in Section 1.3.1.1, 1.3.2.1 and 1.3.3.1.

1.3.1 Cross-site Scripting Attacks

A special type of application layer attacks called Cross-Site Scripting attacks (XSS) have become frightening over the past couple of decades. Traditionally, these attacks were used to steal personal information, leading to possible impersonation of a victim. However, recently with the evolution of technology, these attacks are being used with social engineering techniques to create and launch other punishing attacks. A cross-site scripting attack can best be described as an application layer code injection attack on the client-side where an attacker injects malicious scripts

into a vulnerable Web application, paving the way for the successful execution of the malicious code in an innocent user's browser [9][10].

A Web application is said to have an XSS vulnerability if there is a lack of proper input validation, and there are no proper sanitization routines. XSS vulnerabilities were discovered as early as the 1990s, but they became publicly known only in the early 2000s. The severity of such attacks has considerably increased over the years. XSS attacks can now be used to manipulate the Web content, spread malware, launch DDoS attacks, and hijack user sessions in addition to transferring personal sensitive information to attacker's servers. To gain more insight into the statistics of XSS attacks over the years, Figure 1.3 shows how the number of XSS vulnerabilities have increased over the last 15 years.

Perhaps the greatest danger lurking around the XSS vulnerabilities is its potential

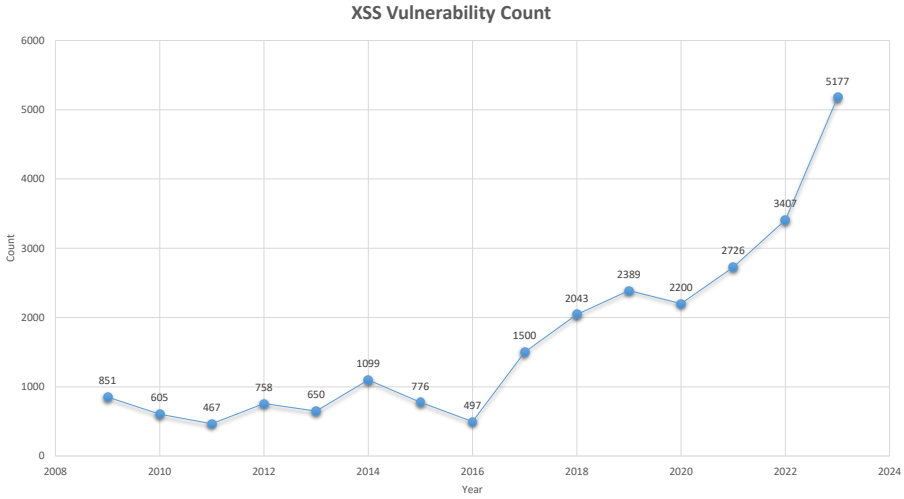

Figure 1.3: XSS Vulnerability Count Over the Years

possibility of propagating from user to user of an application, until either the vulnerability is patched or the whole system gets infected. This is the reason why it is called XSS Worms [11]. XSS worms may be of two types. Server-side XSS worms and Client-side XSS worms. The rarely occurring Server-side XSS worms store the malicious payload in the Web application itself. But the Client-side XSS worms store the payload in external files or parameter values. The Client-side XSS worms can be activated by simply viewing the infected Web pages. The Server-side XSS worms require more actions from the user like installing a fake malicious software. The greatest weapon of an XSS worm is that it works on the client-side and hence it exploits the user's circle of trust[12]. XSS worms can propagate at a much faster rate

[12] `www.gnucitizen.org/blog/the-generic-XSS-worm/`

than the traditional OS based worm. Similar to the functioning of self-propagating malware, XSS worms also have two important tasks [12] as mentioned below.

1. Propagate: This task is important because once the worm is launched, it has to spread itself to new uninfected areas so as to reach and sabotage large audiences. Samy worm[13], was the first self-propagating JavaScript worm.

2. Execute payload: After propagating, the worm will not sit idle without causing any damage. The worm may execute itself to do several tasks like steal the user's cookies, delete the files in the user's computer, or post arbitrary messages from the user's account in Online Social Networks.

The behavior of the worms vary according to the platforms and propagation vector. Different types of worms can propagate in different ways, either through attachments or through malicious links or exploit a security hole in the mail service.

1.3.1.1 Real World XSS Attacks

As discussed in Section 1.3.1, depending on the propagation vector and the platform different worms behave in different ways. One such example is the Yamanner XSS worm[14]. It contacted users whose email address would end with @yahoo.com or @yahoogroups.com. Some of the real world XSS are discussed in Table 1.1. The table gives details of the real world XSS worms over the years. As it can be seen for propagation most of the XSS worms used XMLHttpRequest (XHR) method. This method is used to send asynchronous request in the background and hence much of the malicious activity goes unnoticed. This is the method's advantage over form-submission method. The form-submission method denotes HTTP Requests which might result due to submitting a form or links being clicked. Some of the worms like Yamanner were OS specific while some were not. Many of them spread through social networking sites or Webmail services and some through gaming applications. Some worms were designed as part of an experiment to see how many users could be affected at the most or to find the vulnerability or exploits in an application. Intention was not to take control of any account. But it obviously paved the way to launch more damaging attacks. An important detection mechanism here would be Spectator [13] which performs automatic detection of XSS worms. It also works for the containment of the worm.

[13] https://betanews.com/2005/10/13/cross-site-scripting-worm-hits-myspace/

[14] https://www.theregister.co.uk/2006/06/12/javscript_worm_targets_yahoo/

Table 1.1: Real World XSS Worms Over the Years

Year	XSS Worm	Propagation Method	Infected Platform	Nature	Description
2005	Samy	XHR	MySpace	Self-replicating and persistent	It's a self replicating XSS worm and took less than a day to infect more than 1 million MySpace users. It added the infected users to the creator's MySpace account.
2006	JS Yamanner	XHR	Yahoo	Self-replicating and URL redirection	It first sends a mail to the users of Yahoo! mail. When the user opens the mail, it spreads to the user's contacts. As a matter of fact, it exploits a vulnerability in Yahoo! Mail, which allow the scripts in a HTML document to execute in a user's browser, when actually it should have been blocked.
2007	Orkut XSS	XHR	Orkut	Persistent and self-propagating	An example of a classic XSS attack. It would exploit the security flaw in Orkut's URL parsing mechanism. This security flaw allowed an attacker to load and execute arbitrary code within the Orkut environment. As many as 400,000 Orkut users were infected.
2007	Hi5 XSS	Form submission	Hi5	Persistent	Lack of proper input validation led to the existence of XSS vulnerabilities in the application. One could easily write HTML or JavaScript code in the input fields of comments or in the mail messages.
2010	Bom Sabado		Orkut	Persistent	It is a cookie stealing script. The infection caused the users to receive scraps which displayed the words "Bom Sabado", meaning Happy Saturday in portuguese. Also, the users receiving this script were forcefully a part of fake communities.
2010	Boonana Java		Facebook	Malicious URL activity, Persistent, replicating	The worm is in reality a trojan malware, trojan.osx.boonana.a . It used Facebook messages as a propagation vector and masqueraded itself as a video link with the subject "Is this you in this video?". On clicking the malicious link, it directs the user to an external malicious website.
2010	Rainbow XSS	XHR	Twitter	Persistent, replicating, URL redirection	This worm particularly exploited the onmouseover event in JavaScript. The infected user's tweet would contain strange messages with giant letters, or say Hello followed by blacked out strips of lines. The followers of the infected account automatically recevied the malicious string, thus infecting other non-suspicious users. As many as 1000 users were infected every 10 seconds.
2011	Facebook XSS		Facebook	Persistent	The worm put to good use the XSS vulnerabilities present in the mobile API version of Facebook. JavaScript written code were not properly filtered. It granted any Website the permission to include a malicious iFrame. The code witihin the malicious iFrame redirected the browser to a prepared URL containing JavaScript. On execution, it successfully and automatically posted messages to other people's walls.

1.3.2 HTTP Flooding Attacks

HTTP Flooding attacks on the application layer of the OSI model are a kind of DDoS attack where the services offered by a Web server are brought down by an attacker. These kind of attacks consume lower bandwidth as the attacker floods the server with legitimate HTTP requests. After a point of time upon flooding, the server is overwhelmed and cannot respond to the legitimate requests of its user base. The attacker customizes the requests according to the target web application and the ultimate aim is to exhaust the server's limited resources. It is important to note here that, an attacker may employ a botnet army to launch a co-ordinated attack to bring down the services. Comparatively, there is very less difference between a legitimate HTTP request and an attack request, which is why these attacks are hard to detect. This is because the underlying Transmission Control Protocol (TCP) and User Datagram Protocol (UDP) connections for both an attack request and a normal HTTP request are the same. Only difference between the two is the intent with which a request is made. Additionally, as seen from real instances of application layer DDoS attacks[15] it is difficult to perceive the starting and ending point of an attack. Depending on the scale of the attack, the disruptions caused by the attack often lives longer than the attack itself and this is the reason why it is hard to estimate the actual length of an attack.
The Cisco Annual report[16] outlined how DDoS attacks have increased over the years (2018-2023). Figure 1.4 depicts the number of DDoS attacks over the same period. It is seen that in the year 2023 alone a total of 15.4 million DDoS attacks were recorded.

1.3.2.1 Real World HTTP Flooding Attacks

According to Kaspersky's press release[17], the gaming and gambling industry faced the highest number of DDoS threats between 2022-2023. This is because nearly 40% of the world's population forms the gaming community which is why it becomes an enticing target for the attackers. To make things more interesting, it is found that Fridays recorded the busiest day of attacks with approximately 15.36%, while

[15] https://www.enisa.europa.eu/publications/enisa-threat-landscape-for-dos-attacks

[16] https://www.cisco.com/c/en/us/solutions/collateral/executive-perspectives/annual-internet-report/white-paper-c11-741490.html

[17] https://www.kaspersky.com/about/press-releases/2023_kaspersky-reports-growth-in-gamer-cyberattacks-in-2023

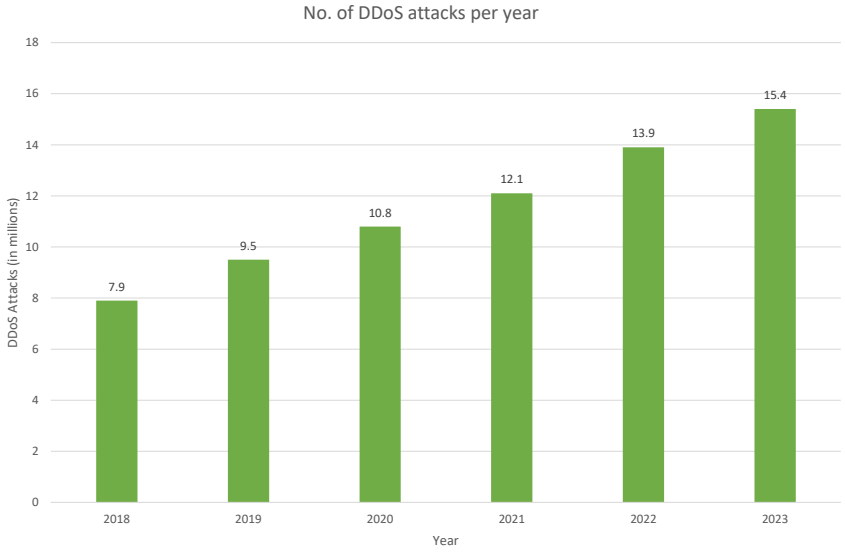

Figure 1.4: DDoS Attacks Over the Years

the lowest are recorded on Thursdays with 12.99% [18]. Table 1.2 presents some real world HTTP Flooding attacks in the recent times.

[18] https://www.getastra.com/blog/security-audit/ddos-attack-statistics/

Table 1.2: Real World HTTP Flooding Attacks

Year	Targeted system	Goal of the attacker	Service	Disruption length	Botnet used?	Description
25th Aug, 2023	Polish railway system	Disruption of service	Railway infrastructure	Total disruption of service for days	No	Due to vulnerability in the system, the attackers could send stop signal to the trains and thus halting their operations during the Russia Ukraine war
7th June, 2023	Microsoft Azure infrastructure	Financial gain with extortion	web infrastructure, network bandwidth exhausted	Partial disruption of service for days due to intermittent nature of attack	Yes	Services provided by Microsoft Azure were disrupted for days, after which the attackers tried to extort money in order to halt the attacks.
30th January, 2023	Hospitals in the USA	Revenge or retaliation against government	Web infrastructure, network bandwidth exhausted	Unknown	Yes	The web infrastructure of a dozen hospitals was targeted by a russia-affiliated group as part of retaliation against the government for support of Ukraine
April 2022	Cloudfare	Disruption of service	Web infrastructure	unknown	Yes (6000 bots used across 112 countries)	A cloudflare customer on a professional plan was the victim
March 2022	Israel government ministries (Health, Defence, Interior, Justice) and Prime minister's office	Disruption of service (services provided by the government website of Israel[19])	Web infrastructure	1 day	Yes	One of the largest HTTP DDoS attack targeted the Israel government's website making services unavailable for a day to the citizens.
March 2022	Ukraine ministry of Defence's webmail server	Disruption of webmail service	webmail server's services targeted	Unknown	Yes	The webmail server was attacked using DanaBot[20], a platform for Malware-as-a-service
August 2021	Cloudflare customer base	Disruption of service for the customers	Cloudflare service	Unknown	Yes (20,000 bots used across 125 countries)	The attack traffic was generated using Meris botnet[21] and it peaked at 17.2 million requests per second.

[19] gov.il
[20] https://www.malwarebytes.com/blog/detections/trojan-danabot
[21] https://blog.cloudflare.com/meris-botnet

1.3.3 Attacks in Critical Infrastructure

Development and innovation in the field of information technology has brought about a significant revolution in the day-to-day lives of human beings. Industrial sectors such as electricity, transportation, banking, gas and petroleum, water distribution, and agriculture are all influenced and governed with technology, primarily because technology has made everything so easy to use. Everything and anything today is just a click away. Traditionally, such industries were not prone to security issues and hence were not well protected. However, in recent times, safe operations of these industries have been jeopardized by damaging attacks. Attacks on the Ukranian power grid[22] and David-Besse Nuclear plant[23] are some instances which illustrate the seriousness of the security issues faced by such facilities. Because these industrial facilities and their operations directly impact the smooth functioning of basic societal infrastructure, including the safety of human lives in a particular geographical region or area or organization, they are termed Critical Infrastructure. One security breach, and a lot of people get affected drastically.

Cyber attacks are carried out on critical infrastructure to disrupt their normal operations and functioning. This in turn negatively impacts almost every sector of everyday life. The implementation of these attacks may be motivated by a wide variety of reasons ranging from financial gains, criminal intentions, political rivalry or to carry out a cyber war or espionage from one state/country against another. On one hand the attacker might have malicious intentions to sabotage the operations, and on the other hand he/she may just illegitimately gain access into the network and quietly observe the ongoing activities in the system. Such attacks are called Reconnaissance attacks, which are generally used to find out the vulnerabilities in the system. The goal of such attacks may be to incapacitate the organization at a much larger scale economically or business-wise. The ultimate reason why critical infrastructure security is important is because of the impact that it can inflict to an entire geographic region, state or even a country in case of a security breach. Failure to prevent such attacks may lead to catastrophic consequences. Code Injection and Data Injection attacks on the critical infrastructure are quite common and mainly aim to manipulate data integrity. Erroneous code generating invalid data or false illegitimate data are injected into the applications. Such attacks disrupt the normal flow of operations in critical infrastructure, thereby affecting day-to-day

[22] https://ics.sans.org/media/E-ISAC_SANS_Ukraine_DUC_5.pdf
[23] http://large.stanford.edu/courses/2015/ph241/holloway2

activities of human life.

1.3.3.1 Real World CPS Attacks

Over the years there has been several incidences of cyber attacks against Cyber Physical Systems, some of which are discussed below in Table 1.3. While a few attacks initially were for intelligence gathering and espionage, some others were purely to cause devastation for a particular geographic area.

Table 1.3: Real World Attacks on Critical Infrastructure

Year	Attack/Malware name	Affected Facility	Element affected	Consequence
2000	Maroochy	Maroochy water services, Queensland, Australia	Pumps in sewerage pumping stations	Raw sewage totalling 800,000 litres spilled into neighbouring areas of the plant destroying marine life,water bodies and residences.
		Source: https://www.mitre.org/sites/default/files/pdf/08_1145.pdf		
2003	SQL Slammer worm	David-Besse Nuclear plant	The safety monitoring system disabled for nearly 5 hours	Employees were not able to access the Safety Parameter display system, responsible for displaying the safety indicators at the plant. As a result, crucial safety hazard were left unmonitored for a long period of time.
		Source: http://large.stanford.edu/courses/2015/ph241/holloway2/		
2008	Radio attacks on medical devices	Cardio Patients	A medical device	Patients do not undergo the expected therapy. Patient safety and privacy compromised.

Table 1.3: Real World Attacks on Critical Infrastructure

Year	Attack/Malware name	Affected Facility	Element affected	Consequence
		Source: Halperin et al.		
2010	Stuxnet	Iranian Nuclear facility	Associated centrifuges of the PLCs	Centrifuges span too quickly or for too long, thus sabotaging the equipment.
		Source: https://www.csoonline.com/article/562691/stuxnet-explained-the-first-known-cyberweapon.html		
2014	German Steel Mill cyber attack	German steel mill	Individual Control System components	Security settings of blast furnaces were not triggered in time, leading to serious damage to the physical components of the infrastructure
		Source: https://ics.sans.org/media/ICS-CPPE-case-Study-2-German-Steelworks_Facility.pdf		
2014	Havex (aka Dragonfly)	European companies developing industrial machines		Intelligence gathered were passed to the attacker servers for possible future cyber attacks on the Control Systems
		Source: https://www.scmagazine.com/havex-malware-strikes-industrial-sector-via-watering-hole-attacks/article/538721/		

Table 1.3: Real World Attacks on Critical Infrastructure

Year	Attack/Malware name	Affected Facility	Element affected	Consequence
2015	Black Energy attack	Ukranian power grid	IT infrastructure components	Three energy distribution companies were affected losing control over the grid system and power substations were switched off remotely. More than 225,000 customers were in the dark for 1 to 6 hours.

Source: https://ics.sans.org/media/E-ISAC_SANS_Ukraine_DUC_5.pdf

1.4 Sophistication of Attacks

Fortifying every security aspect of an application is next to impossible because of their inherent complexities. With the growth of an application's complexity the chance of overlooking potential vulnerabilities and exposures may also rise. Smart attackers introduce sophistication in their techniques by customizing the attack vectors according to the design flaws of the application. Such engineered attack vectors increase the effectiveness of the attacks. In recent times, attackers are successful in not only finding the vulnerabilities in complex web applications but also deceiving the security mechanism put in place to secure the system. One of the primary goals of an attacker is to execute an attack with minimum costs but at the same time create damaging effects of same or even higher intensity. Using sophisticated techniques, an attacker may not only disguise itself as a legitimate user of the system but also gain unauthorized access to the system. Once victorious in gaining unauthorized access into the system the innocent users are totally at the mercy of the attacker. It is important here to note that, an attacker may also employ an army of zombies to act on his/her behalf or in some cases co-ordinate with other like minded entities spanning across different countries to threaten a high value target. Social engineering techniques are also garnering tremendous attention, which outwit an unsuspecting user into clicking on malicious links. Subsequently, such trickery may lead to more harmful consequences such as demanding of ransom from the user in lieu of his/her stolen information.

1.5 Motivation of Attackers

A very important question to ponder upon when discussing web attacks is what is the reason or decision behind the attack. The key reason or the decision behind the attack is what motivates the attacker to carry out the attack in the first place. The motivation behind a web attack can be many fold. There may be a wide range of goals and incentives for the attackers to attack an individual, or group of individuals or organization, or a particular geographical area. The victims and their behaviors are carefully studied over a period of time so as to make a bigger impact when the attack takes place. It is important here to note that, the attacks may be carried out by an individual, group of individuals, organization or attacks may be government sponsored. Depending on the target, the attackers may employ varied

strategies to bypass a security mechanism in place. For example, the strategy to attack an individual will be much easier to orchestrate than the strategy to attack an organization. Some of the motivations behind a web attack are listed below.

1. *Financial motives:* Perhaps the most common motivation of an attacker is to make financial gains from their victims. This can be made possible by stealing sensitive information of the victim such as: bank login details, and credit/debit card information. Another scheme may be to make the target suffer financially. More recently, ransomware attacks are also on the rise where the users lose access to their systems and data unless a ransom is paid to the attackers.

2. *Competitive motives:* Business organizations or individuals may orchestrate an attack to secure an edge in the market over its competitors. This can be made possible by stealing information for soon to be launched products/prototypes, intellectual property, or sensitive trade information, etc.

3. *Political motives:* Attackers may be politically motivated to orchestrate an attack against a victim who is not aligned with their own political beliefs. Such motives may be sponsored and are more common during the election times.

4. *Espionage and gathering intelligence:* Nations, governments and intelligence agencies may sponsor attacks against other nations and governments to gather crucial information (for example nuclear codes), or gain strategic upper hand in times of war.

5. *Revengeful motives:* Sometimes attackers may have revengeful motives against its victims because they believe they were wronged in the past, or because the attacker's ideology may not be aligned with the victims.

1.6 Defense Approaches

From a security perspective, the last several years have been very eventful with several high profile data breaches, threats and attacks against web applications. Web application attacks are possible because of existing vulnerabilities which puts both the end users and the businesses at risk[14]. An effective defense approach is essential to timely detect and secure a web application. Typically, a defense

system may consists of four modules- i) Detection module which tries to analyze data for specific attack occurrence, ii) A prevention module which tries to prevent to attack from occurring (may be at source-end or the victim-end) , iii) A mitigation module which tries to employ mitigation techniques (such as blocking some IPs) after attack occurrence, and iv) A tolerance module which tries to provide services to the legitimate users even when an attack is occurring. Additionally, a defense approach may follow a centralized approach or even a distributed approach. When developing a defense approach for detection of attacks in web applications the following important points should be noted:

1. Most web applications are in themselves heavily complicated because of the technologies used, hence the design of the defense approach should be such that it does not further add to the application's complexity.

2. The operations of legitimate users should not be harmed because of the deployment of a defense approach.

3. A defense approach should be scalable and robust.

1.6.1 Machine Learning for Web-based Attack Defense

Machine learning is used to solve complex problems whose solutions are in high demand without expert intervention. Most data intensive problems are complex, requiring intense and exhaustive computations, beyond the ability of human expertise. Demanding problems necessitate the use of machine learning techniques to extract hidden insights inherent in underlying patterns, which are likely to be overlooked by humans. This is when experts turn to machines. The general notion is just like in the human world, observe examples to learn which aids in constructing a sufficiently precise model of the problem. The built model is then put to test against data and if needed necessary improvements are made to the model.
For quite a long time now, machine learning is used in diverse set of fields, ranging right from engineering to biological domains. Machine learning can also be expressed technically from a data perspective. Typically, machine learning is more capable of handling large and complex datasets effectively than traditional methods like manual computations, analysis or experiments[15]. This is because machine learning models are iterative, more reliable and adaptive in nature to new data. Technically in other words, machines are empowered by machine learning algo-

rithms, and depending on the nature of the algorithm pre-existing knowledge may or may not be used. These algorithms work on given data to find intrinsic interesting patterns to fulfill an objective; learning of such patterns in the data can be thought of as gaining experience [16] (also known as training). The learning model generalizes this experience to work on previously unseen samples (known as test data used during testing phase). The better the generalization capability of a learning model, the better it works with previously unseen data.
A significant step before using any learning technique on the original data is preprocessing. There are numerous preprocessing techniques such as normalization, missing value estimation, discretization, feature selection or other necessary data transformations that can be used on the data for better characterization of the problem. The performance of a learning technique largely depends on the preprocessing techniques used. For example, one can use a variety of feature selection techniques based on information theory, or proximity or correlation measures. The type of technique employed depends largely on the type of data at hand. In the case of Web-based attack detection, the primary idea is to choose a subset of features which are independent of each other, and are relevant to the target class of the sample [17]. Choosing relevant features and reducing the redundant ones can positively influence the performance of a learning model. Effectiveness of the selected subset can be subsequently evaluated using a learning model. It is imperative to consider the amount of processing time that a traditional feature selection technique may consume as it considers all the features at once to select a feature subset. Such a technique recomputes from scratch for any added-in feature in the data and hence may not always be suitable. This is when an incremental feature selection method can play a vital role as they can dynamically select features to discriminate attack and normal instances.

1.7 Motivation and Objectives

The Web is a powerhouse of resources as almost all industry, and organization are banking on it to create, grow and maintain their customer base. Ranging from financial, education, entertainment, gaming and gambling almost all services are provided over the Web. Primarily, all Web applications exchange information to and from its users by relying on the HTTP protocol. Since, the past two decades

Web-based attacks serve as major security problem as attackers exploit vulnerabilities in the applications and protocols. The sophistication of these attacks are troublesome not only for the users but also for the security experts and researchers. Defending against the attacks and shielding the users and services of these applications is a challenging task but not impossible. Over the years, numerous detection mechanisms have been developed but with these developments come another challenge - 'evolving of new attack vectors'. Smart attackers employ these vectors wisely to generate attack payload which has very similar characteristics with normal operations. In this regard, real time detection of these attacks with minimum false alarm is a major research issue.

Identifying distinguishing features of an attack play a crucial role for a detection mechanism. An effective Web-based attack detection mechanism should be able to provide its decision using low computational resources and in near real time. This is possible only if small number of distinguishing features are considered. However, a Web-based attack may be characterized by a large number of features. So, preprocessing of these features to select only the most relevant ones is imperative. It is important here to note that features may change dynamically for an attack. Handling of such added in data necessitates the use of incremental feature selection. Sometimes, relying on only one technique may not be a good idea if the detection mechanism is to handle attacks in a critical infrastructure facility. For such detection mechanisms, ensemble feature selection should be employed because decision given by a group of selectors is better than the decision given an individual selector. Hence, the main motivation of this thesis is to choose feature subset which can accurately characterize attacks. The subset of selected features are subsequently used for cost-effective detection of Web-based attacks.

The principal objective of this thesis is to develop efficient detection mechanisms to counter selected Web-based attacks. Particularly, this work focuses on Cross-site scripting attacks, HTTP Flooding attack and attacks on critical infrastructure. Following are the objectives of the research work.

1. **To explore different detection mechanisms for defending against selected Web-based attacks**

   Justification: Many detection mechanisms have been proposed in recent years to defend against widely popular Web-based attacks such as XSS attacks, HTTP Flooding attacks and attacks on critical infrastructure. So, an extensive literature survey to study the detection methods of these attacks along

with analyzing their pros and cons is unavoidable.

2. **To study and explore the applicability of machine learning techniques or ensemble approaches to defend against Web-based attacks**
   Justification: From the literature study it is seen that, machine learning techniques have been applied to detect and mitigate Web-based attacks. Such techniques relying on individual decisions have their own pros and cons. To mitigate the pitfalls of these individual approaches, ensemble learning models may be used, where the individual decisions given by a group is combined in some manner. There is a silver lining to using an ensemble of experts, which is governed by the intuition that a combined opinion of several experts is likely to be more precise and balanced than an individual expert. With a goal to gain deep insights into ensemble approaches, it is intended to study the topic extensively and analyze how they can be applied effectively and in a timely manner to detect Web-based attacks.

3. **To explore and gather existing or synthetic benchmark datasets for XSS attacks, HTTP Flooding attacks and attacks in critical infrastructure. Also, to create a real life dataset for XSS attacks**
   Justification: The effectiveness and efficiency of a detection system can be well scrutinized with the help of a dataset. It is aimed to explore the existing repositories for gathering datasets concerning XSS attacks, HTTP Flooding attacks, and attacks in critical infrastructure. A significant task here is to identify features which can help to effectively distinguish between normal and malicious instances. However, in the case of XSS attacks, a standard XSS feature dataset is not publicly available on the Internet. So, the creation of such a dataset with updated information is important and necessary for the timely, efficient and accurate detection of XSS attacks. The quality and effectiveness of the dataset is ensured by factors as mentioned below.

   - Labels in the data: Can be *normal* or *malicious*.
   - Adequate number of instances: Required for sufficient training and for unbiased evaluation.
   - Adequate number of features: Meaningful and non-redundant features help define characteristics of a class.
   - Balance between the classes: Imbalance in the class instances may result in inappropriate evaluation of a test method.

- Recency: Outdated data will not follow the newer and evolving attack trends.
- Lack of inconsistency: Consistent data will result in development of reliable defense method.
- Relevance: Irrelevant data will bring down the performance of the detection model.
- Completeness: Incomplete data may lead to inadequate training of the learning model, performance will be degraded.

4. **To develop a cost-effective solution for detection of XSS attack.**
   Justification: Once state-of-the-art methods are studied, next step is to focus on developing a robust detection method for solving the problem of Cross-site scripting attacks. The main goal of such a detection method is to identify highly relevant and irredundant subset of features which are necessary for the timely, effective and assured detection of XSS attacks. This subset needs to be optimal, meaning addition of new features to the subset should not make the performance any better and at the same time reduction of any feature should bring down the model's performance.

5. **To develop a cost-effective incremental detection mechanism to mitigate HTTP Flooding attacks in the application layer.**
   Justification: The selection of appropriate characteristics or features plays a major role in identifying actions that may lead to an attack. All features may not be equally informative; some may be redundant or irrelevant and thus play no role in the detection process. As a matter of fact, a subset of highly informative features needs to be selected for good performance. On the other hand, in case of a dynamic real world applications all the data may not be available at one time. Such cases necessitates the use of incremental learning, so as to avoid learning from scratch each time data is available. Thus, a defense mechanism that incrementally selects relevant and irredundant features will definitely aid in solving the problem of HTTP Flooding attacks in the application layer.

6. **To develop an ensemble approach to defend against attacks in critical infrastructure facility.**
   Justification: Once the existing state-of-art methods to tackle attacks on critical infrastructure (such as power grids, and water treatment plants) are

studied, next task is to develop a defense method to solve the problem. It is of the notion that a single feature selector sometimes may be biased and as a result may not be appropriate to differentiate attack and normal instances in a critical facility. In such cases, an ensemble of experts should be employed because decision given by a group is better than the decision of an individual selector. As a matter of fact, the decisions given by the group should be combined in some manner to preserve the importance of each feature in the aggregation process. Thus, a defense mechanism that relies on the foundations of ensemble learning will definitely help in the identification of attacks in a critical infrastructure facility.

1.8 Contributions

The contributions in this thesis are presented briefly below and discussed in details in the upcoming chapters.

- *Contribution 1:* Design and development of an XSS dataset generation pipeline. The proposed pipeline comprises of several stages and modules for generating the dataset named XSSD. Additionally, a feature extraction algorithm to extract relevant features from collected scripts towards creation of XSSD is also proposed.

- *Contribution 2:* Design and development of a data-centric supervised ensemble framework to comprehend the effects of ensemble learning methods in security data. The ensemble learning methods considered are Bagging, Boosting, Bagging-Boosting and Stacking.

- *Contribution 3:* A mutual information and correlation based feature subset selection method called MICC-UD for identifying a subset of highly relevant and independent feature subset so as to detect XSS attacks with best possible classification performance. The proposed method is able to handle multi-class data as well.

- *Contribution 4:* A mutual information and correlation based incremental feature subset selection method called INFS-MICC for identifying a subset of highly relevant and independent feature subset so as to detect HTTP Flooding attacks with best possible classification performance in near-real time.

- *Contribution 5:* An ensemble feature selection method called FSRA for the identification of an optimal subset of relevant features to help detect attacks in critical infrastructure with minimum cost.

1.9 Thesis Organization

The organization of the thesis is given below.

- Chapter 2 discusses in detail the related background for comprehending the selected Web-based attacks namely XSS attacks, HTTP Flooding attacks and Attacks in Critical Infrastructure. Detection methods to counter the mentioned attacks are also presented. It also discusses a generic pipeline for dataset generation and the types of datasets that one might use for evaluation of a detection mechanism. Additionally, concepts related to machine learning approaches, cost-effective methods for attack analysis and several validation measures are also discussed in details.

- Chapter 3 presents the desired characteristics that a dataset should possess. It also introduces several benchmark security datasets used for evaluation of the proposed methods in later chapters. In addition to the benchmark datasets, a dataset generation pipeline consisting of several stages and modules is proposed. Finally, this chapter reports the characteristics of the generated dataset along with its performance evaluation.

- Chapter 4 introduces a data centric ensemble framework for understanding the effects of ensemble learning methods namely Bagging, Boosting, Bagging-Boosting and Stacking in security data.

- Chapter 5 describes a traditional feature selection method named MICC-UD to detect XSS attacks. MICC-UD selects an optimal number of features using mutual information to find feature relevance and correlation to find feature redundancy. Comparison of with other benchmark feature selection methods are also included.

- Chapter 6 reports an incremental feature selection method for the detection of HTTP Flooding attacks named INFS-MICC. It identifies a subset of features incrementally from HTTP traffic samples.

- Chapter 7 introduces an ensemble feature selection method called FSRA for detection of attacks in critical infrastructure. Comparative analysis of the ensemble method with existing critical infrastructure defense methods is also included.
- Lastly, Chapter 8 winds up with the concluding remarks and future work.