Chapter 2

Background

This chapter presents a comprehensive discussion on the concepts, and existing detection methods to defend against the selected Web-based attacks namely Cross-site scripting attacks, HTTP Flooding attacks and Attacks in Critical Infrastructure. Additionally, a generic pipeline for dataset generation is included along with the different types of datasets that one might use for evaluation. Cost-effective methods for attack analysis such as different kinds of feature selection techniques are also elaborated in detail.

2.1 Networks

A computer network comprises of a systematic collection of computers and other network enabled devices such as desktops, laptops, printers, file servers, routers, and switches as shown in Figure 2.1. All the devices, systems and users in a network are interconnected with one another such that communications can be established (either wired or wireless communications), data can be exchanged and resources/services can be shared [18]. Based on the requirements of an organization or individual or community, the size, design, architecture and complexity of a network may vary [19]. For example, a small simple network can be designed to function within a home or office, while complex large networks are designed for data centers, large offices and even the entire Internet. With modern communications being facilitated by computer networks, security issues and challenges also arise. A few major challenges faced by networks include safeguarding the user's data, maintaining privacy, detecting and mitigating threats from intruders and protecting the integrity of the overall network [20].

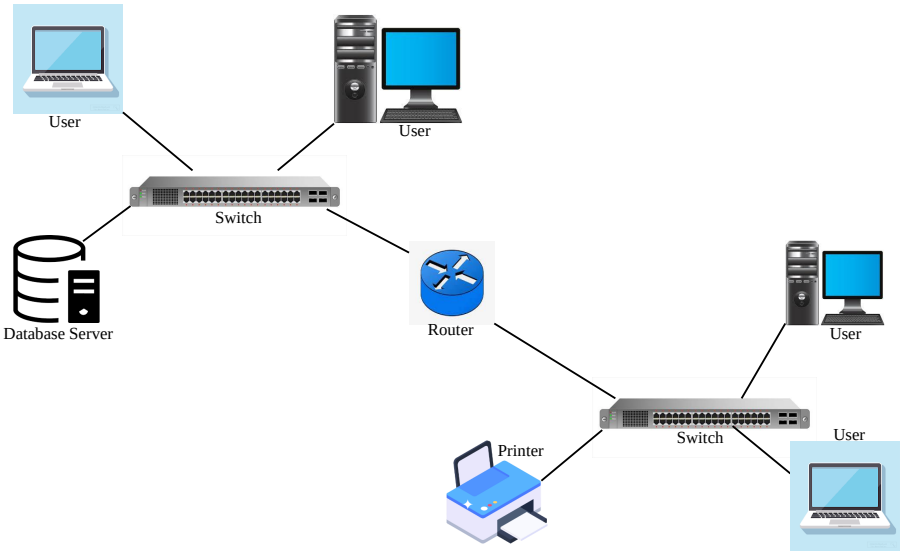


Figure 2.1: A Computer Network

2.2 Network Communications

Fundamentally, network communication refers to the process through which devices and users in a network communicate with one another. For effective communication to take place in a network or inter-network, there must exist three fundamental components[21]. Firstly, two entities, i.e., a sender and a receiver must exist to exchange data or share resources and services. Secondly, for the exchange or sharing to take place, there must be a communication medium or channel between the sender and the receiver. Eventually, for the communication to actually take place, the sender and receiver must agree on a set of rules or protocols. It is also important to note that the network communications can be wired or wireless. For network communication to take place, the following considerations play the key role.

(a) Data: Data to be transferred between the sender and the receiver can be in the form of text, multimedia, files or other forms of information that the parties have agreed to share.

(b) Devices: Network enabled devices such as laptops, desktops, routers, and switches participate in network communication. These devices are responsible for sending, receiving and routing the data to its next hop and to the destination.

(c) Protocol: Network communication protocols facilitate communications among the participating devices. These protocols establish a set of rules and conven-

tions which are necessary to mandate how and in what format data should be transmitted, routed and received.

(d) Addressing: To uniquely identify each device in the network, an address is assigned to the device to ensure data reach the intended recipient. The address may either be an Internet Protocol (IP) address in the network layer of the OSI model or Media Access Control (MAC) address in the data link layer.

(e) Security: Perhaps the most important requirement of all is the security in network communication because data should not be accessed by unauthorized parties or should not be tampered with when transmitted. So, proper encryption, decryption, and access mechanisms should be followed.

2.3 Internet and Its Issues

The Internet is a network of networks spanning the globe. No single entity (individual, group or organization) has full authority over the Internet. Figure 2.2 shows how different devices and entities take part in network communications. Computers and associated networks have been part of human lives since at least the early 1960s. At this time, every aspect of human life involves extensive activities related to computers [22]. Organizations provide their services and the users consume these services by connecting their devices to the Internet[23]. The rise of the Internet over the years has been beneficial and crucial. Almost everything and anything in the world is now just a click away.

It is reasonable to assert that not all users of the Internet are fair and honest, some are malicious with intentions that are not conducive to routine activities [24]. Activities of such individuals may be damaging in a number of ways. They may employ inauthentic ways to break into a system, steal a legitimate user's credentials or valuable information, impersonate a legitimate user, cause an organization or individual financial, socio-political or economic loss, or even obstruct or discontinue health care services for a critical patient. While some such situations may lead to disruption of the normal flow of regular activities or cause financial loss, others could be life threatening. Clearly, an understanding is necessary about how much and what information we are to share over the Internet without allowing any malicious user to harm or intrude our personal and professional spaces. It is essential to note

that all trivial and non-trivial activities performed by an individual on a computer connected over a network are necessarily logged. The amount of traffic and log information to and from a network is vast and consists of vital data which could be mined to extract valuable underlying knowledge. Such knowledge can be used to learn and classify what can be regarded as normal and what as not normal or malicious.

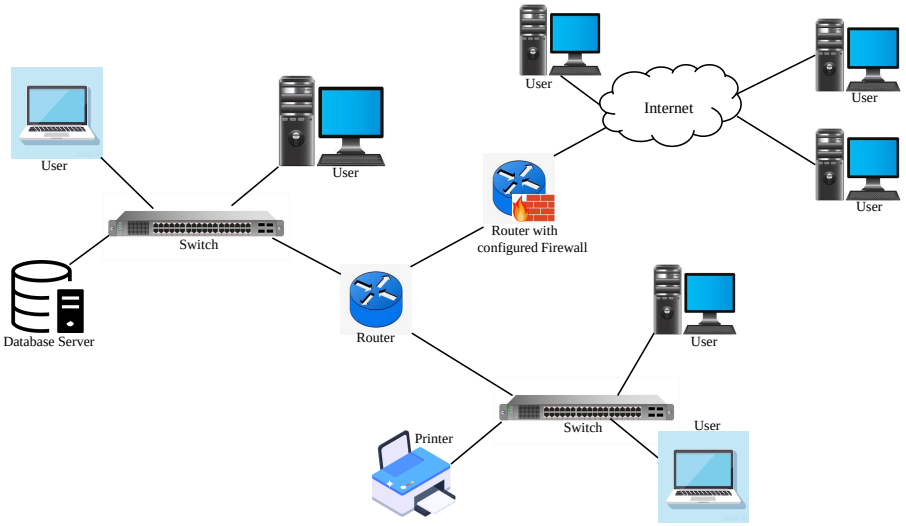


Figure 2.2: Network Communications and the Internet

2.4 Network Attacks

Network attacks are deliberate, unauthorized, and malicious activities by perpetrators that target computer networks, systems, or the data they contain. Key vulnerabilities of the target network are identified and exploited by the adversaries to perform malicious activities[25]. The main focus is to either compromise the confidentiality, integrity of the network hosts or disrupt the normal functioning of network resources. A crucial task in detecting such network attacks is identifying anomalies in the network activities. This can be made possible by characterizing the behavior of normal and abnormal traffic in a network. Challenges arise when sophisticated attackers mimic the trends of normal operations which makes detection of such attacks difficult. Interestingly, the malicious users may have varied motives, including financial gains, espionage, or disruption of services of a particular company.

2.5 Web-based Attacks

As the Internet promises to provide additional and reliable services, the risks to security and maintenance of sensitive information are becoming higher. Many vulnerabilities lurk in Web applications, often without the developer's knowledge. Perpetrators take advantage of these vulnerabilities in Web applications, posing serious threats to the end users. As a result, many users of vulnerable applications fall prey to attackers. The presence of such vulnerabilities in the Web applications serve as a cause for the possibility of attacks by malicious users. Although a large number of Web-based attacks are recognized in the recent times, the three most commonly occurring attack types are considered and their basics are presented next.

2.5.1 Cross-site Scripting Attacks

Cross-site Scripting attack is a code injection attack at the application layer; it is also known as an XSS attack. To start, the attacker needs to find a vulnerability within a Web application and exploit it by injecting malicious code, thereby targeting the end-user of the application. In other words, malicious content is injected into a trusted context. As per the legitimate requests of the user (the victim), a page from the affected application is served to him or her with the malicious injected code. The Web browser on the victim's end unknowingly executes the malicious code as a legitimate part of the Web application. As a result anything can happen, including all user sensitive information ending up at the attacker's server. It is important to note that XSS attacks are successfully executed only when the Web application does not validate its input at all or does not validate its input properly. Particularly, the output generated by the Web application uses raw invalidated input.

An XSS attack may take advantage of ActiveX, HTML, Flash or JavaScript. Out of the lot, the most widely exploited is JavaScript. According to the statistics, 93.6% of the Websites all over the world use JavaScript[1]. A malicious JavaScript snippet exploits the Same Origin Policy. Any content from a Web site may have permission to access a system's resources if the origin site is granted access with those permissions. The client's browser falls victim to the malicious intentions of the attacker as it cannot differentiate between the legitimate and malicious content

[1] `https://heimdalsecurity.com/blog/javascript-malware-explained/`

delivered from the same Web site.

2.5.1.1 XSS Attack Representation and Its Behavior

As stated earlier, XSS attacks can be categorized into three kinds. These attacks differ in nature and are possible only because there exists a vulnerability either on the client side or the server side. For instance, the nature of a DOM-based XSS attack is different from the other two because such an attack is possible due to vulnerabilities on the client side. In contrast, Persistent and Non-persistent XSS attacks are possible due to vulnerabilities on the server side. Figure 2.3 is a representation of an XSS attack showing the steps in a Persistent and a Non-persistent XSS attack.

To gain deeper insights into the functioning of an XSS attack, one must first

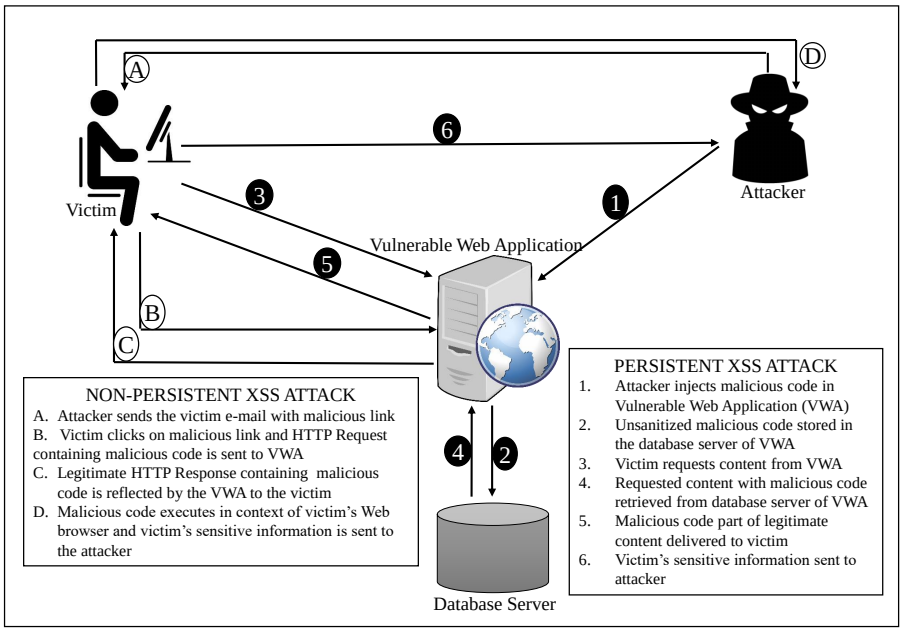


Figure 2.3: Steps in Persistent and Non-persistent XSS Attacks

understand the kind of data used to carry out an XSS attack. Malicious scripts can be injected into the server of a Web application or malicious links can be crafted for the victim to click. The means by which an XSS attack is carried out is called an XSS attack vector. A complete list of XSS attack vectors can be found in [26]. The data associated with an XSS attack can be of two types, raw data or feature level data, both of which are described in detail below.

1. Raw data: Data which have not undergone any processing are called raw data [27]. Script content or the presence of URLs is considered raw data.

Such data usually undergo preprocessing so that important features can be extracted for analysis and detection purposes.

(a) XSS attack script: JavaScript is widely used in a malicious manner to exploit XSS vulnerability in Web applications. A malicious script can either be injected directly into the Web application through input forms or malicious links can be crafted by injecting the malicious script in the URL through parameters. Figure 2.4 gives an example of a malicious JavaScript injected into an application. Similarly, Figure 2.5 gives an example of a crafted malicious link.

(b) XSS attack in execution: Malicious JavaScripts execute in the context of the Web browser. For example, with the help of the script as shown in Figure 2.4, the attacker tries to steal the cookies of the victim. Subsequently, the victim's browser parses the malicious script as part of the HTML document. Social engineering techniques can be used to lure the victims into clicking on crafted links as shown in Figure 2.5. On clicking such malicious links, the parameter *path* is retrieved from the URL and placed at a particular location in the Web page. The value of the parameter *path*, that is the <script>...</script>portion, becomes a part of the legitimate HTML code. The victim's browser now understands no difference between the injected malicious script and the legitimate HTML code in the Web page. On the other hand, the victim is unaware of the execution of the malicious script, which was possible only because of the existence of XSS vulnerabilities in the Web application in the first place.

```
<script>window.location="http://maliciousscript.com/?cookie='document.cookie</script>
```

Figure 2.4: Malicious XSS JavaScript

```
http://commondatastorage.googleapis.com/chromium-browser-
continuous/index.html?path="><script>alert(document.cookie)
</script>
```

Figure 2.5: A Crafted Malicious Link

2. Feature level: Features can be extracted from the collected raw data. Features or characteristics play an important role as they help understand the

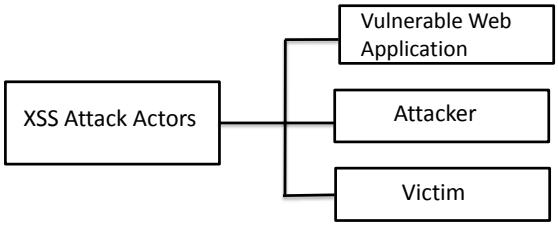


Figure 2.6: XSS Actors

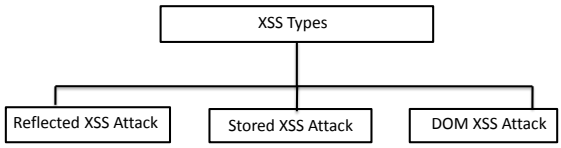


Figure 2.7: XSS Types

complexity, readability, meaning, and functionality of the raw data. The extracted features help gain additional insights into the data, which in turn aid the data analyst for further analysis. Features of the data are usually categorized into the following types.

(a) Script-based features: Script-based features can be extracted from normal or malicious JavaScripts present in Web pages. These features enumerate basic characteristics of scripts such as the number of characters, the number of lines, the number of words, the number of comments, and the number of functions. Such features also assist in quantifying how readable the scripts are.

(b) URL-based features: URL-based features can be extracted from URLs obtained from resources such as [28] and [29]. These features help differentiate malicious URLs from the legitimate ones.

2.5.1.2 XSS Attack Actors

There are three actors in an XSS attack as shown in Figure 2.6. Depending on the way the malicious code is injected into the Web application, XSS attacks are categorized into the three variants as shown in Figure 2.7. Practically, the manner in which the malicious script is delivered to the victim's browser and the way in which it is executed are different for the three categories, and hence the classification. Reflected XSS is the most common XSS attack, although potentially more dangerous is the Stored XSS attack. Reflected and Stored XSS attacks differ from DOM-based XSS attacks because the latter type arises due to flaws in the browser's script interpreter. In contrast, Reflected and Stored XSS attacks are the results of vulnerabilities in the Web application. DOM-based XSS attacks are relatively uncommon.

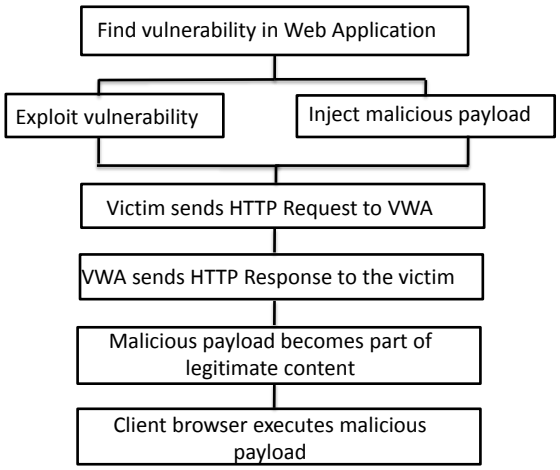


Figure 2.8: Steps in an XSS Attack

2.5.1.3 Steps in an XSS Attack

In an XSS attack, malicious content (example: a malicious script) is introduced into the trusted context of a vulnerable Web application. The victim, on executing the Web application, is served with the malicious content which masquerades as a part of the legitimate code of the application. The victim's browser ends up inadvertently executing the malicious script because of its inability to differentiate between malicious and legitimate content. Figure 2.8 shows the steps involved in successfully launching an XSS attack. An attacker does not directly target the victim, but uses flaws in the vulnerable Web application as tool to deliver the malicious code to the victim's browser.

2.5.1.4 Types of XSS Attacks

XSS attacks can be carried out in different ways depending on the type. The different types are discussed briefly below.

1. *Reflected XSS Attack*: Also popularly called Non-persistent XSS attack or Type-I XSS attack. The attacker smartly crafts a malicious link containing a script and lures the victim into clicking on it. Inevitably, the victim's request to the server also has the malicious string as a part of it. The response from the server incorporates the malicious code snippet (i.e., reflected back) and the victim's browser executes it indifferently. Thus, XSS vulnerability exists if the user input is directly a part of the output generated by the application without any sanitization. The Reflected XSS attack model is as shown in

Figure 2.9.

2. *Stored XSS Attack*: Commonly called Persistent or Type-II XSS attack. The attacker crafts the malicious code, snippet which is permanently stored on the vulnerable server. The malicious code may be injected through message forums or blog posts. It is then stored on a server for the application. Eventually, when the victim navigates to the compromised site, he or she is served with the malicious code snippet as a part of the Web page. Finally, the victim's browser ends up executing the malicious code. Thus, lack of proper validation of user input and sanitization routines results in the existence of XSS vulnerabilities in the server application. As a matter of fact, all users who visit this site are now at the risk of executing the malicious code in their browser unknowingly. Appropriate routines should therefore be in place before storing the data into the database. The Stored XSS attack model is as shown in Figure 2.10.

3. *DOM-based XSS Attack*: Also widely called Type-0 XSS attack. Attacks of this category differ significantly from the above, mainly because DOM-based XSS attacks are possible due to the existence of vulnerability in the script interpreter of the client's browser, whereas other two types of attacks are due to server side vulnerabilities. The DOM structure of the Web page on the client's browser is modified. This is the reason why the attacker is successful in executing the malicious script. It is worth noting that the attacker's crafted malicious payload cannot be found in the response as in the case of the other two types of attacks. It can be found by either scrutinizing the DOM or when the Web page is loaded, i.e., at runtime. The DOM-based XSS attack model is as shown in Figure 2.11.

Out of the three types of XSS attacks, DOM-based XSS attacks requires special attention. This is primarily because of its nature. As mentioned earlier, unlike Reflected and Stored XSS attack, DOM-based XSS attacks arise due to vulnerability on the client side. When carrying out a DOM-based XSS attack, the attacker manipulates the objects in the DOM and inappropriately handles the properties of the HTML page[2]. Such attacks are crucial to detect because the HTML source code and the response of the attack are exact[3]. This means that the attacker payload is

[2]`https://www.acunetix.com/blog/articles/dom-xss-explained/`

[3]`https://www.netsparker.com/blog/web-security/dom-based-cross-site-scripting-vulnerability/`

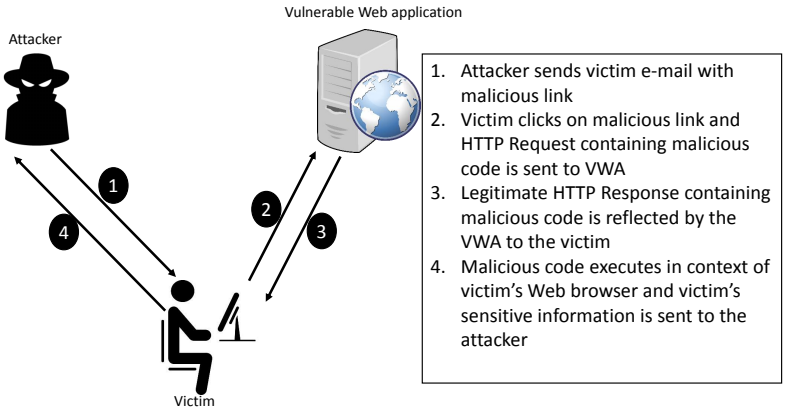


Figure 2.9: Reflected XSS Attack Model

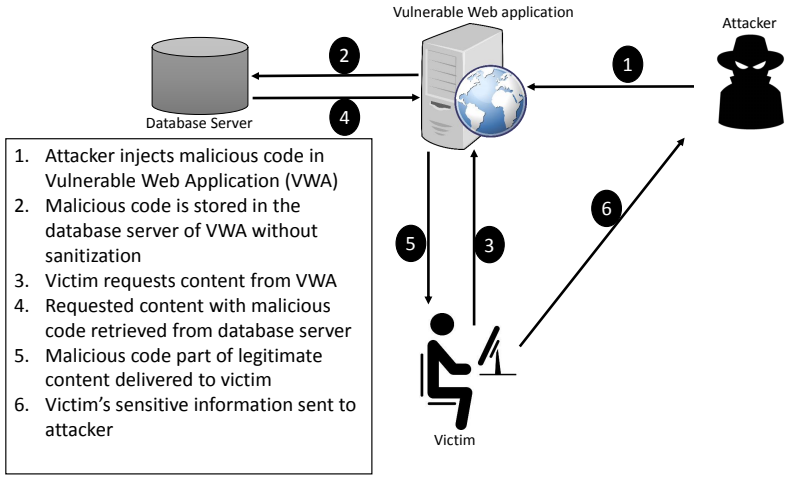


Figure 2.10: Stored XSS Attack Model

not part of the response, but part of the DOM of the HTML page. Objects such as *document.url*, *document.location*, *document.referrer*, *window.location*, *location.href* may be utilized by the attacker. It would be very important to say that DOM-based XSS attacks cannot be detected by server-side defenses effectively. This is true if the '#' character is used, because on encountering the character, the browser identifies it as a fragment and never forwards it further. One such example is as shown in Figure 2.12. As such, a majority of the time the injected malicious code does not reach the server at all. Thus, the defenses or sanitization routines at the server-side do not play any role in detecting DOM-based XSS attacks. Code review for vulnerability detection, sanitization routines or prevention techniques must be clearly implemented at the client-side for detecting such attacks.

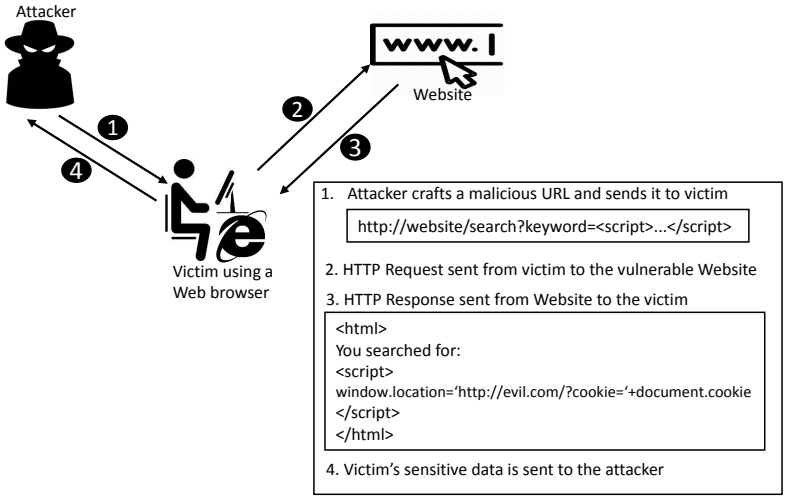


Figure 2.11: DOM-based XSS Attack Model

http://www.example.com/dom-xss-example.html#keyword=<script>...</script>

Figure 2.12: DOM-based XSS Attack Crafted URL

2.5.1.5 A Scenario of a Successful XSS Attack

To better understand XSS attacks, a scenario is illustrated. The attacker sends an e-mail message to the victim, incorporating a link injected with a malicious script. Using social engineering, the victim is lured into clicking on this link. An HTTP Request from the client is generated, meant for the server hosting the vulnerable Web application. The HTTP Request has the malicious script embedded within it. The Web application's server generates an HTTP Response meant for the client. The HTTP Response includes the reflected malicious script from the server. The malicious script is now a part of the legitimate content from the server. However, the victim's browser has no concerns or doubts and executes it. Eventually, the attacker gets hold of much of the vital information from the victim. The attacker can now easily impersonate the victim and access the server on his behalf. Figure 2.13 shows an example of a malicious link sent by the attacker to the victim. The parameter *topic*, after being retrieved from the URL is placed at a specified position on the Web page. Consequently, the $< script >$ element becomes a part of the HTML code. It is then treated by the Web browser in the same manner as any other element in the HTML document. The victim is not knowledgeable of the fact that he or she is executing a script, which is possible only because of the existence of XSS vulnerabilities in the Web application. There are several possible ways by which the attacker may craft the URL in a way that it is rendered unreadable or is not

suspicious. To make the URL unreadable, the attacker may use URL obfuscation techniques, which are readily available as tools. An obfuscated version of the URL is as shown in Figure 2.14. A second way may be shortening the URL to make it look unsuspicious as shown in Figure 2.15. Obfuscation leads to low human readability.

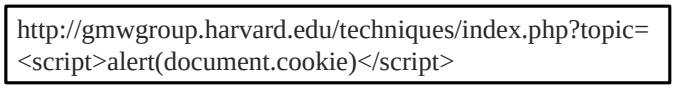


Figure 2.13: Malicious Link

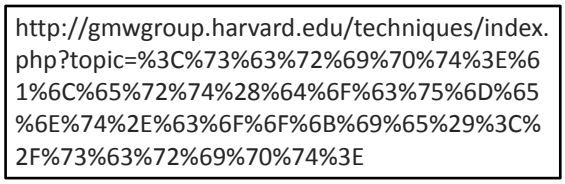


Figure 2.14: Obfuscated Malicious URL

Figure 2.15: Shortened Malicious URL

Such techniques may also be used for legitimate purposes. However, the intentions to obfuscate a malicious script and the intentions to obfuscate legitimate content are totally different. For example, some sophisticated software developers might not want others to understand their code and hence use various obfuscation techniques. On the contrary, ill-intentioned authors of malicious scripts use obfuscation to evade the intrusion detection systems, especially signature based and static analysis based systems.

2.5.1.6 Detection Approaches for XSS Attacks

Detection approaches as available in the literature can be broadly classified into i) Client-side detection approaches, ii) Server-side detection approaches, and iii) Client-Server detection approaches. For detecting XSS attacks on the client-side, detection measures may be embedded in the form of filters in the client browsers, or set up as proxy servers with defined rules. On the other hand, server-side XSS detection mechanisms may be incorporated on the application's servers, or set up as reverse proxy. A detection approach that deploys the defense mechanism on both client-server sides has also gained popularity. It is important to note that detection

of XSS attacks with the use of Machine Learning is also on the rise. The client-side detection approaches and server-side detection approaches may be further based on Static Analysis, Dynamic Analysis or a combination of both. Below, a discussion on the techniques and how these techniques help in detecting vulnerabilities in an application along with attack detection and prevention approaches are presented. Figure 2.16 shows the proposed taxonomy of the detection approaches. The basis

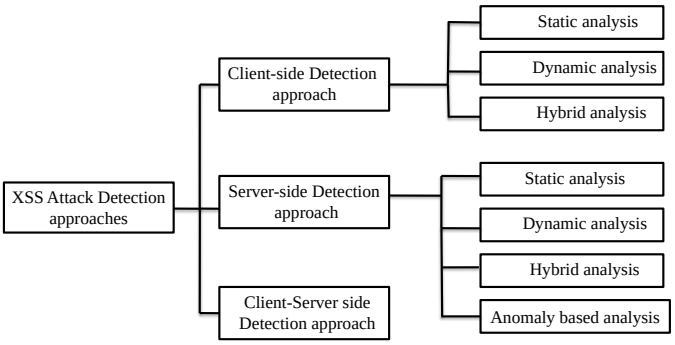


Figure 2.16: Taxonomy of the Detection Approaches

of the presented taxonomy of the detection mechanisms is each mechanism's deployment site. A defense mechanism for XSS attack can be either on the client-side, server-side or client-server side. Again, the functioning of each of the mechanisms depends on the analysis mechanism they employ. So, the detection mechanism under each deployment site is again sub-categorized into Static Analysis, Dynamic Analysis and Hybrid Analysis. Thus, similar mechanisms applying the same analysis technique are categorized accordingly under the specific deployment site. This is the underlying approach of the proposed taxonomy.

1. *Client-side Detection Approaches:* As mentioned earlier client-side detection approaches may perform i) Static Analysis, ii) Dynamic Analysis or iii) Hybrid Analysis. Deployment of the defense mechanism on the client-side can be either on the browser as a filter or plug-in, or on a proxy server. The three analysis variants are discussed below. Table 2.1 gives a summary of some of the XSS filters.

   (a) *Static Analysis:* Static Analysis approaches predominantly focus on the application's source code [30]. The source code is reviewed with the primary purpose of finding the security flaws as shown in Figure 2.17. Static analysis means that there is no execution of the Web application involved.

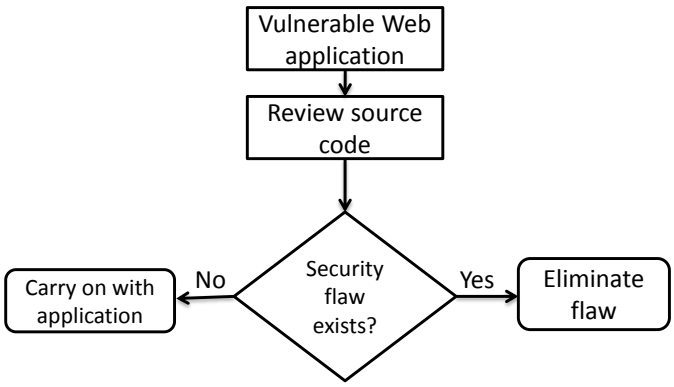


Figure 2.17: Steps in Static Analysis

(b) *Dynamic Analysis:* Dynamic analysis mechanisms concentrate on the runtime behavior of an application as shown in Figure 2.18. Unlike, static analysis mechanisms, they do not go through the source code. The executable code of the application is examined to discover vulnerabilities. Dynamic analysis mechanisms are more accurate in detecting the vulnerabilities and generate lower false positive rates.

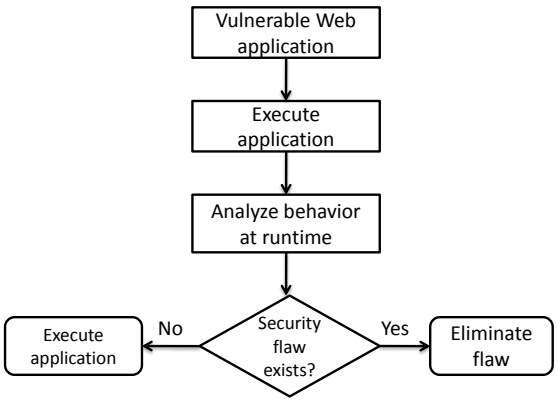


Figure 2.18: Steps in Dynamic Analysis

(c) *Hybrid Analysis:* Hybrid Analysis combines the benefits of both Static Analysis techniques and Dynamic Analysis techniques. It ensures precision and efficiency. Computationally, static analysis techniques are expensive and suffer from the inability to make definite decisions. However, dynamic analysis techniques are precise and relatively effective.

Discussion: A cross-site scripting attack mainly occurs due to the existing vulnerabilities on the server-side. However, with proper mechanisms in place, one can be restricted from visiting a malicious site. A server cannot always be

trusted when it comes to security, as a server may be compromised easily by hackers. The benefits of deploying a defense mechanism on the client-side is that the user has full control on if he or she wants to visit the site. A browser can be embedded with a policy or XSS filters can be used to restrict visiting a malicious site. However, the downside to this is that a layman might not know whether it is safe or not to visit an application.

Table 2.1: Summary of XSS Filters

	XSS Auditor (2010) [31]	XSS Filt (2012)[32]	NoScript [33]	IE8 (2008) [34]	XSS-immune (2016)[35]	XBuster (2016)[36]	Rule Based (2016)[37]
Method used	Exact string matching	Approximate string match-ing	Regular expression based	Regular expression based	Sanitization and string comparison	String matching	Rule patterns and Sequence behavior based
Type of XSS attack detected	Reflected, DOM	Reflected, Stored	Reflected	Reflected	Reflected, Stored, DOM	Reflected, Stored	Reflected
Use of regular expressions	✗	✗	✓	✓	✗	✗	✗
Partial scripting detection	✗	✓	✗	✗	✓	✗	✗
Initial phase prevention	✓	✗	✗	✗	✓	✗	✓
Context aware sanitization	✗	✗	✗	✗	✓	✓	✗
Deployed in browser	Google Chrome	Firefox	Firefox	IE 8,9	Google Chrome	Firefox	Firefox
False positive rate	medium	medium	high	medium	low	medium	medium

2. *Server-side Detection Approaches:* Traditional XSS attacks, whether reflected or stored, occur due to vulnerabilities on the server-side. So, it is only natural to develop a defense mechanism which shields the server from such attacks. Deployment of the defense mechanism on the server-side can be either on the server itself or as a reverse proxy. A discussion on various defense mechanisms as found in the literature is presented below.

   (a) *Static Analysis:* As described in Figure 2.17, static analysis deals with the source code of the application [30]. The main advantage of utilizing such analysis mechanism is that the application need not be executed. The source code of the application is reviewed for potential detection of vulnerability which might be exploited by the attacker at some point of time [38]. Some common techniques under static analysis include taint analysis [39], control flow analysis [40], data flow analysis [41], inter-procedural analysis [42].

   (b) *Dynamic Analysis:* Dynamic analysis involves execution of a Web application [43], as described in Figure 2.18. Such techniques actively analyze the program states to detect vulnerabilities in the application [44]. Program states may include values of variables or even contents of memory locations. Dynamic analysis may be preferred instead of static analysis in case of large scale Web applications. This is because, in such cases statically examining the code may be tedious.

   (c) *Hybrid Analysis:* Hybrid analysis makes use of the advantages of both static analysis mechanism and dynamic analysis mechanism. Static analysis may be used to identify the vulnerable sections in the application's code. Subsequently, such sections may be executed to analyze the behavior, i.e., perform dynamic analysis [45]. However, if the static analysis mechanism fails to detect any vulnerability, then performing dynamic analysis is unlikely to yield any fruitful result. Thus, proper validation of the sanitization mechanism is necessary [46].

   (d) *Anomaly-based Approaches:* Anomalous instances are the instances that do not possess the expected normal behavior or characteristics of a system. These may also be called outliers, anomalies or exceptions [47][48][49][50]. An understanding of what is considered normal is required to classify an instance as anomalous. So, if an instance's characteristics and behavior deviate highly from the profile generated by a

normal model, it is categorized as anomalous [51]. Anomaly scores may be assigned to the instances to signify how anomalous they are.

Discussion: Since an XSS attack arises due to server-side vulnerabilities, it makes full sense if a defense system is located on the server-side. Because, a server is a powerful machine, detection is possible in near real time. But, a server-side defense mechanism is limited to detecting only reflected and stored XSS attacks. Because a DOM-based XSS attack occurs due to vulnerability on the client browser, it is not possible to detect by server-side defenses. Moreover, there may be performance issues in the defense mechanisms, due to which the client may face delay in receiving the output.

3. *Client-Server based Approaches*: An XSS attack poses threats to both the client and the server. The client is at the risk of losing sensitive information to a malicious attacker because the attacker can easily impersonate the victim. The server on the other hand, is at the risk of losing its resources culminating in business and financial risks. A defense mechanism to counter such attacks should therefore rely on both the client and the server, and not just on one of them. A defense mechanism which balances the load between the clients and the server should be preferred. Such a defense mechanism combines the advantages of both the server-side and client-side defense mechanisms. Table 2.2 gives a brief summary of client-side, server-side and client-server side defense approaches.

   A separate summary for a number of machine-learning approaches is also provided in Table 2.3.

2.5.2 Cyber Physical Systems

In recent years, a new generation of control systems have evolved to monitor and control the physical world. Developments in the field of Information Technology (IT) has changed the way how humans interact with the physical world just like how Internet changed the ways of interactions among people. These control systems are called Cyber Physical System (CPS) because of the integration of cyber systems to the physical world [64]. Helen Gill first proposed the term CPS in 2006 at a workshop conducted by US NSF (United States National Science Foundation)[4]. In

[4] `https://www.nsf.gov/awardsearch/showAward?AWD_ID=0647591`

general, a CPS is also termed as a Critical Infrastructure because of its behavior, significance and impact in the real world. A CPS facilitates a wide range of solutions and services that help people in their daily lives. These systems range from small scale infrastructure such as a smart vehicle to large scale infrastructure such as a smart power grid [65].
Overall safe operating conditions are crucially important for any Cyber Physical System because their proper functioning and efficient operation are of prime importance to the day-to-day activities of people in a geographical region [66]. This is the chief reason why such facilities are called Critical Infrastructure (CI). CPSs provide a bridge between the cyber and the physical spaces. These systems offer a sophisticated integration of computational and physical processes of a critical infrastructure, bringing new capabilities to the physical system. Through a communication network, the functioning and operations of the physical resources are monitored and controlled, as necessary.

2.5.2.1 CPS and Its Architecture

A CPS is typically defined by the combination of physical, computational (cyber) and communication processes. More specifically, the objects in the physical world are controlled and monitored from time to time by the computational components in the cyber world over the communication network. The main function of the physical world objects is to sense the data from the physical environment (for example, temperature and pressure) and perform the commands as given by the cyber world elements. The main function of the cyber world objects on the other hand, is to analyze the data received from the physical world and based on the analysis release appropriate commands [67].
According to [68], a CPS architecture consists of three layers, each with its own functionality and the devices required to carry out these functions. The three layers are Application layer, Transmission layer and the Perception layer as shown in Figure 2.19 and are discussed in detail below.

(a) *Perception layer:* This layer corresponds to the physical layer in the CPS architecture. The devices or equipments which can be a part of this layer are the sensors, actuators, intelligent devices, scanners, chemical analyzer indicators, flow indicators, cameras, bar code readers, etc. [69]. The main job of the devices in this layer is to collect real time data from the physical environment and carry out the necessary commands received from the higher layers. Based

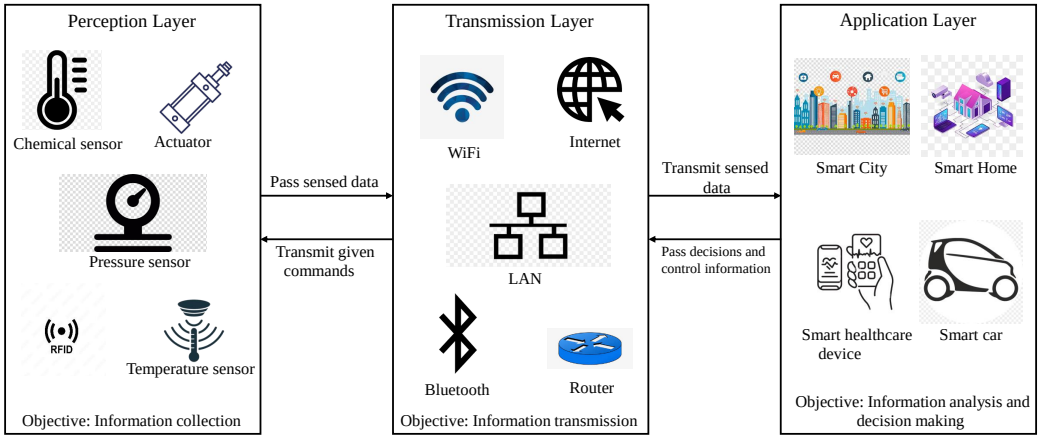


Figure 2.19: Layers of Cyber Physical Systems

on the type of device, readings related to temperature, humidity, pressure, location, chemical concentration, vibration or acceleration may be aggregated [70]. Perception layer is also known as the sensor layer as mentioned by [71].

(b) *Transmission layer:* This layer corresponds to the communication layer in the CPS arcitecture. The main job of this layer is to exchange or forward and process data between the perception layer and the application layer. The data interchange in this layer may take place through communication networks, Internet, local area networks or other technologies such as WiFi, Bluetooth, etc. Of course, according to the technology used, the underlying protocols also differ. Another important functionality of this layer is routing data to appropriate intermediate devices in the network securely [71]. The most important responsibility of this layer is availability, reliability and realization of real time transmission of data [67][69].

(c) *Application layer:* This layer corresponds to the computational layer in the CPS architecture with which the end user interacts. As it receives data transmitted from the transmission layer, it aggregates them and sends neccessary commands back through the transmission layer to the physical layer units like sensors and actuators [72]. The functioning of this layer is relatively complex than the other two because of the participation of complex data mining and decision making algorithms. These algorithms help to aggregate

large amounts of data based on which several corrective and decisive measures might have to be taken. In addition to these, application layer is also responsible for monitoring the overall normal functioning of the systems. In case of any anomalous behavior, corrective measures have to be taken so as to ensure security and optimal behavior as soon as possible. It is also important to note that, past logs of the system are also maintained by the application layer for future references and improvements [68].

2.5.2.2 Aspects of a CPS

Technically, at the core of each critical infrastructure lies a CPS. A CPS integrates engineered cyber and physical elements, the operations and functions of which are supervised, coordinated, and administered with the help of a consolidated communication core [73]. In simple terms, the cyber world and the physical world of a CI are bridged by a communication network. A CPS can span from small scale health care devices like heart monitoring devices to large scale nuclear power plants dedicated to a specific region. Typically, such systems exist and function in a heterogeneous domain as different hardware elements collaborate with software elements for controlling and monitoring purposes. The cyber elements coordinate and communicate with physical elements (like sensors) to collect real time data or to gain intelligence about the real world. The collected data are managed and computed for further decision making. Henceforth, the necessary decision is shared with the physical world through actuators [74]. The computation, communication and control technologies of the cyber physical systems enhance the capabilities of the physical world, optimizing the performance of a system. Cyber physical systems can operate in a centralized or distributed manner, but they must always be connected, robust and positively responsive.

From time to time, security breaches that occur in critical infrastructures like the power grid, nuclear power plants, water distribution plants, give us an indication of the importance of the issues. Critical infrastructure, be it small or large scale, is the new target of the attackers. Modern critical infrastructure functions by creating a bridge between the cyber world and the physical world. Physical entities or the environment (like temperature, pressure, light, heart beat of patients, blood sugar level, etc.) are recorded and monitored with the use of sensors. The sensors pass the information over a communication network to the cyber world, awaiting valuable decisions, if and when necessary. Human analysts or experts governing the cyber

world analyze the incoming data and accordingly make suitable decisions. For example, if the heartbeat of a patient is too high, he or she needs medical attention immediately. The decision given by the analyst is carried over the communication network to the actuators. The actuators ultimately interpret the decision and convey it to the physical world. Figure 2.20 depicts the sectors with which CI may be associated.

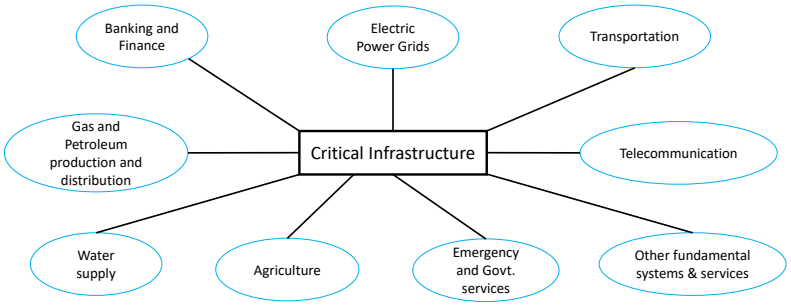


Figure 2.20: Critical Infrastructure Thrust Areas

2.5.2.3 Threats against CPS

The close association between CI and well-being of human lives requires that the infrastructure be free from any inside or outside security threats, vulnerabilities or potential attacks. An attacker can maliciously influence an element either in the physical or cyber space, thus breaching the security of the CI. Upon launching an attack, the attacker successfully disrupts the functioning or ongoing operations, thereby wrecking sudden havoc in the regulations of the day to day activities [75]. In a completely different scenario, the attacker, after gaining illegitimate authorization into a system, instead of causing any disruptions, may just quietly follow the ongoing activities in the system. The acquired critical information then may be used to extensively sabotage the organization later. Such attacks are called Reconnaissance attacks, and are carried out to spy on the target's normal activities. The target can be an individual, or an organization or even a country, in which case, it can lead to a cyber war. Attackers with criminal motives strike the target at an opportune time so that it faces irreversible losses either financially, socially, militarily or politically. The kinds of threats that a cyber physical system may face are described below.

(a) *Criminal threats:* Attackers may remotely hack into an application to take

full control of the host system in an unauthorized manner. Following the loss of control, the smooth running of the system operations may be disrupted. Private sensitive information of the consumers may also be stolen.

(b) *Financially motivated threats:* Vulnerabilities in the physical and cyber spaces of a critical infrastructure facility may be exploited by an attacker for financial gains. For example, a power grid might be injected with false data so that it is misled regarding the actual consumption by a customer. The modified meter readings and bill amounts cause the electricity providing company or the customers to incur financial losses. It may also happen that the attacker, after intruding into the system, demands ransom, non-payment of which may lead to shutdown of the entire power plant or blackout of an entire region. Attackers may gain private information of the customers and sell it to advertisement companies who analyze the user statistics to show advertisements to influence the customers to buy products.

(c) *Politically motivated threats:* Political threats may lead to cyber war or espionage between countries. One country might rage a war against a competing country and target its critical infrastructure so as to harm the common people by disrupting the services.

(d) *Physical threats:* Physical threats correspond to threats that may affect the working of physical devices such as sensors and actuators. This may disrupt the functioning of the whole physical space associated with CPS, and as such no physical information of the surrounding environment may reach the cyber systems and human supervisors. In another scene, the physical devices may be spoofed so that they do not send out the correct readings of the environment. Such a case may be devastating when actually corrective actions are needed, but are not provided due to apparently normal functioning of the involved elements.

(e) *Spying threats:* Spying threats are closely related to reconnaissance attacks where attackers gain access to the system, but do not cause any damage or harm to any entity. However, they skillfully listen to the ongoing communication between two parties so that they can use it later as leverage to launch more damaging attacks.

2.5.2.4 Types of Attack in CPS

Attacks on Cyber Physical Systems could result in severe damage to the physical units, computational resources as well as the users benefiting from the services of these systems. According to Peng et al. [72], CPS systems are more prone to vulnerabilities and threats compared to traditional IT systems because of individual threats to each of the CPS layers. For example, notorious users may target several units in the application layer to gain unauthorized access, or security issues may arise in the transmission layer during data transmission resulting in sensitive data leakage, and even the nodes in the perception layer may be under an attack [67]. Some attacks as described below could target all the three CPS layers [76].

(a) *Man-in-the-Middle (MITM)*: In these kinds of attacks, a malicious user may position himself in the network in such a way that it intercepts all sensitive transmissions between a target and the system. All the while, the user or the system's administrator is unaware that the communications are actually intercepted in between. Such attacks are generally followed by eavesdropping or replay attacks.

(b) *Eavesdropping*: During data transmission, notorious users could eavesdrop sensitive information of the users or the system's monitoring logs. This could in turn hamper a user's privacy or may lead to security breach of the overall system.

(c) *Denial of Service (DoS)*: These kind of attacks exploit protocol vulnerabilities and overwhelm the system with loads of requests in order to make the services provided unavailable to the legitimate users.

(d) Spoofing: In these kind of attacks, a malicious user disguises itself as a legitimate part of the system and tries to perform any activity that a legitimate user is authorized to perform.

(e) Replay: These attacks occur when a malicious user captures an already legitimately transmitted packet from a destination unit and then again retransmits it later. This is done in order to gain the trust of the system, and the system is fooled into thinking that the malicious user is a legitimate one [77].

However, according to authors in [68], it is meaningful to classify the attacks according to the layer they could target as shown in Figure 2.21.

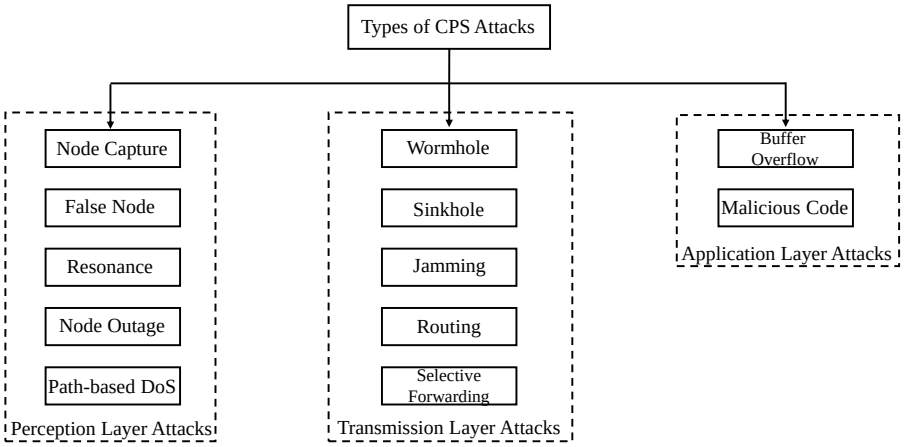


Figure 2.21: Layer-wise Attacks in CPSs

1. *Attacks at the perception layer:* Sensors and actuators are some of the common end devices in this layer which are computationally less resourceful and do not have high capabilities. The main goal of attacks in this layer is either to make the end nodes unavailable or disrupt the services they provide or make the node unreliable for trustworthy operations. Some of the common types of attacks in this layer are discussed below.

   (a) *Node capture:* Malicious entity takes control of an end node and illegitimately reads all the node's data or other encryption keys that could breach security of the whole system.

   (b) *False node:* A node totally under the control of the attacker is added into the network for transmitting malicious data to the users. Such nodes might also be configured in such a way that it consumes the energy of the nodes making them unresponsive to legitimate commands.

   (c) *Resonance:* Makes the compromised sensors and actuators operate at a different frequency than the normal.

   (d) *Node outage:* A node and its operations may be halted by an attacker to disrupt the services it provides. This makes the information emanating from such a node difficult to read.

   (e) *Path-based DoS:* Nodes along a routing path are flooded with packets so as to consume their energy and make them unavailable for further operations.

2. *Attacks at the transmission layer:* The main goal of attacks in this layer is to leak, modify, retransmit or discard legitimate data packets during trans-

mission. Attackers exploit the existing vulnerabilities in the protocols during the transmission of information. Some of the common types of attacks in this layer are discussed below.

(a) *Wormhole:* All the packets deviate from the desired path and are routed through the attacker controlled false paths. Such false paths are intelligently crafted by the attacker by introducing information holes in the network.

(b) *Selective Forwarding:* The attacker compromises a node and makes it to selectively forward some packets, other packets even though legitimate are discarded. Thus the discarded packets never reach the intended destination.

(c) *Jamming:* The communication link between an end node and the base station from where control signals are sent is jammed with same frequency signals. This is done in order to create interference which hinders normal communication.

(d) *Sinkhole:* The attacker smartly crafts a routing path (under its control) which is then announced to the nodes for further packet transmission. This kind of attacks can be used to launch other attacks such as selective forwarding.

(e) *Routing:* The packets take longer time than usual to reach the intended node in the source path. This may be due to delay in transmissions or introducing false nodes in the source paths.

3. *Attacks at the Application layer:* The main goal of attacks in this layer is to damage the accumulated data so as to hamper the decision making process. Unauthorized access, loss of privacy of the users and injecting malicious code into the applications are some of the attacks that can take place in this layer.

(a) *Buffer Overflow:* An attacker may get hold of the memory structure of the application and send its own executable code which due to capacity restrictions overflows the buffer of the application. Such executable code could actually replace the legitimate code of the application resulting in the attacker's total control.

(b) *Malicious Code:* Malicious codes are launched against the application which ultimately affects the normal flow of operations. For example, the

application may be infected with malware which can further replicate and propagate over the network or launch other more damaging attacks like DDoS.

2.5.2.5 Existing CPS Defense Approaches

Many detection mechanisms have been proposed to protect critical infrastructure. Some focus on detecting vulnerabilities in the system, whereas others focus on detecting and preventing attacks. Research presented in [78–80] describe a set of challenges in securing critical infrastructure. The proposed defense mechanisms in the literature concentrate on securing infrastructure systems or elements such as power grids, water distribution systems, and nuclear grids to name a few. Below some important relevant research found in the literature are discussed.

1. Patrascu and Patriciu [81] show how critical infrastructure can be protected with the help of supervised machine learning and game theory decision models. The model has three security layers, of which the first consists of sensors loaded with the responsibility of monitoring and providing training data to the other layers. The second layer has two functions of receiving sensor data and also executing an online algorithm for learning and determining attack patterns. The third layer consists of a rule engine having the role of an intrusion detection system based on game theory heuristics.

2. Rajkumar et al. [73] present a combinatorial algorithm to automatically detect faults and cyber attacks in critical infrastructure. The method's strength lies in its ability to work with limited data.

3. Pan et al. [82] classify cyber attacks and disturbances in power systems using heterogeneous time-synchronized data and propose a sequential pattern approach to do the same.

4. Pasqualetti et al. [83] monitor attacks and identify them in cyber physical systems. Cyber physical systems are inter-dependent on each other. As such, if one of the systems is affected, so are other systems. Inter-dependencies can be in the form of physical interdependency, logical interdependency, geographic interdependency or cyber interdependency [84].

Several other detection mechanisms have been proposed in the literature to identify and defend against cyber attacks in power grids [85–90], water distribution

systems [91–93]. Healthcare systems are regarded as critical infrastructures which require security solutions to safeguard the privacy of patient's details as well as the patient's data itself. In [94], authors present a patient centric machine learning technique to detect cyber attacks in healthcare systems. In general, healthcare devices have limited resources and real time detection of attacks in such devices is of utmost importance. Such facts play a crucial role in designing detection methods for systems in heathcare domain. Highlighting the critical significance, Schneble and Thamilarasu [95] propose a feature selection method based on laplacian scoring techniques to provide ranks to the individual features. Optimal feature subset selection ensures that efficient decisions can be made in real time with less memory consumption in the devices. Intrusion detection using anomaly detection [96], behavior rule specification [97], temporal analysis [98], signature analysis [99], specification analysis [100], and hybrid approaches using signature and traffic analysis [101] have also been proposed in recent years.

2.5.3 HTTP Flooding Attacks

HTTP Flooding attacks are a type of Distributed Denial of Service attack in the application layer of the OSI model, particularly aimed at taking down the services of a Web server. The attackers flood the Web server with HTTP Requests such that the server is overwhelmed and is no more able to serve requests from legitimate users. Thus, denial of service occurs for the actual users of the server. Malicious users may employ a network of compromised hosts called bots to launch a coordinated attack against a targeted victim for a specific purpose. Attackers may compromise the bots with the help of malicious programs or malware under a command and control infrastructure [102]. The main aim of the attacker or the bots, apart from being anonymous, is to exhaust the network bandwidth or block the computational resources (such as CPU cycles, and memory) of the server so that genuine users are denied of its services. Figure 2.22 shows a scenario in which HTTP Flooding attack can take place.

DDoS attacks on the application layer are not new and yet these attacks pose severe security threats to applications and the numerous services they provide [103]. Some of the purposes against launching such attacks are mass consumption of computational resources and bandwidth leading to disruptions in the services provided to legitimate users, defaming the providers by degrading the services provided to customers, targeting the innocent user or customer base, and hijacking the eco-

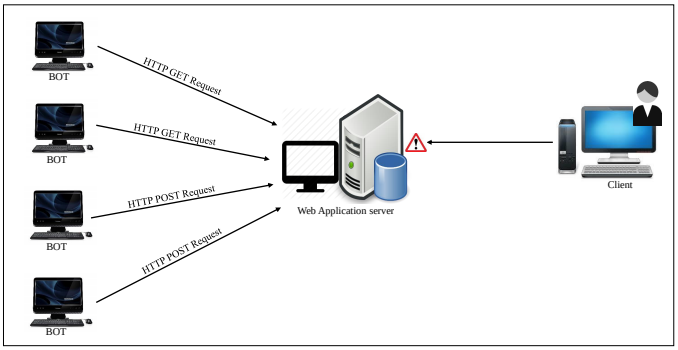


Figure 2.22: HTTP Flood Attack Scenario

nomic benefits of the applications. Such kinds of attacks are hard to detect as the traffic patterns are very similar to normal legitimate traffic patterns. HTTP Flooding attacks can be carried out for a variety of reasons ranging from political, economic to business defamation. Malicious users of the Internet very smartly employ malware to compromise a set of machines as bot. These hosts are then used to send HTTP traffic to the victim server.

2.5.3.1 HTTP Flooding Attack Strategies

The most obvious difference between network level and application level flooding DDoS attacks is that at the network level, network packets are used to flood the network whereas in the application level, HTTP Requests are used for flooding. Application layer DDoS attacks are hard to detect because there is no difference in content between a legitimate HTTP Request and an attack request. The difference lies in the intention to bring down the services of the Web application. Smart attackers first study the behavior of the victim network's traffic, and then intelligently design the attack traffic to have the same pattern as the legitimate one. "Anyone, can encourage others to visit a site, effectively creating a request flood and disrupting a sites stability" [104].

2.5.3.2 Typical HTTP Flood Attack against a Web Server

In an HTTP Flooding attack, Web servers are flooded with HTTP GET or POST Requests. Sending a valid HTTP Request to a server requires establishment of a valid TCP connection in the underlying layers, which in turn requires a valid IP address. Each bot in the botnet is assigned an IP address. The attacker invokes the malicious program in these bots and subsequently HTTP Requests meant for the

server are generated in such a way that detection mechanisms can be evaded, and at the same time attack power is maximized. One of the many ways of maximizing the attack strength is by requesting the download of a large file from the target server. One can only imagine the destruction such tactics can cause to the computational resources of the target like CPU, memory or outbound bandwidth link when such HTTP Requests come simultaneously in bulk.

2.5.3.3 Detection Approaches for HTTP Flooding Attacks

Detection of HTTP Flooding attacks are quite difficult and the reasons are mainly three fold, i) Obscurity, both HTTP attack request and a legitimate HTTP request use valid TCP and UDP connections making it difficult to differentiate between the two, ii) Lethality, irrespective of the hardware capabilities of the server, its resources can quickly be exhausted which results in denial of service, and iii) Efficiency, few established connections are required to successfully launch an attack against the target. In the literature, researchers have categorized defense mechanisms for detecting HTTP flood attacks based on disparate ideas. Zargar et al. [105] categorize the mechanisms based on the deployment site. Destination based mechanisms deploy their defenses at the victim end, i.e., at the Web server end or at the reverse proxy, and Hybrid mechanisms are deployed on both the client and the server. Praseed et al. [106] propose a taxonomy where the detection mechanisms are classified according to request dynamics (traffic estimation, request statistics like entropy based measures), and request semantics (request composition and request sequence). Table 2.4 reports some of the well-known defense mechanisms present in the literature. Many DDoS attack launching tools are available online freely which may be used by individuals with ill-intention for various malicious causes. Because such tools are easily accessible the innocent users are totally at the mercy of the attackers. A comprehensive overview of how these tools can be used for launching HTTP Flooding attacks is described in [116].

2.6 Raw and Feature Data

Data can be in the form of raw data or feature data. Raw data are in the most basic, unfiltered and unprocessed form, collected directly from the source. Such data have not undergone any kind of refinement, modification or formatting and hence may contain missing values, noise, outliers or even irrelevant data points.

Examples of raw data include data collected from sensors (such as temperature, pressure or humidity readings), readings from electronic or electrical instruments or components, field surveys, logs, images or other databases. Figure 2.23 is an example of raw data (in the form of an image ID proof), which contains the information of an employee of an institution. On the other hand, the corresponding feature data consist of specific sets of features or characteristics or attributes that describe a particular data example in a specific context. Feature data make it easy for analysis and for statistical or machine learning models to be built. It is important to note that the features or attributes can hold numeric (or quantitative) values, or categorical (or qualitative) values which carry information. In predictive analysis, to make predictions (decisions or outputs), the features act as inputs. For example, in Figure 2.24, it can be seen that every person (or row, also known as sample) is characterized by three different features (attributes or columns) namely, Name, Age and Gender. Based on the feature values, the task is to decide if a particular person is eligible to vote or not. Needless to say, in different domains, the features characterizing a particular task will be different and relevant to that particular task. Meaningful and conclusive insights for decision making can be drawn from

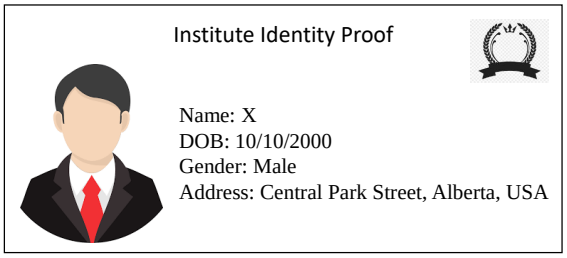


Figure 2.23: Example of Raw Data

Features

Samples

	Name	Age	Gender	Eligible to vote ?
Person 1	X	23	Male	Yes
Person 2	Y	32	Female	Yes
Person 3	Z	17	Male	No

Figure 2.24: Example of Feature Data

raw data only after data have been subjected to processing and analysis phases. The various stages for processing, transforming and analyzing the raw data are discussed in subsection 2.6.1. The final output of these stages in the pipeline is a dataset which can be used in predictive modeling.

2.6.1 A Generic Pipeline of Dataset Generation

As already discussed earlier, for statistical or predictive modeling, raw data cannot be used in its original form. Data have to be subjected to preprocessing techniques so that they can be used in predictive analysis. The dataset generation pipeline presented in this section is generic consisting of stages and steps which may or may

not be utilized by every researcher or practitioner.

2.6.1.1 Data Collection

The very first step in the pipeline is data collection. The input to this step may be a list of verified reliable sources from which data can be collected for further analysis. Some widely used publicly available data sources or platforms are: UCI ML repository, Kaggle, NCBI, Awesome Public Datasets, and Google Database search. During this step, two important questions can be raised, i) *What data are needed for the task at hand ?* and ii) *Do the collected data answer all the needs ?*. The significance of this step lies on the fact that the quality of the collected data will have an effect on the predictions made by a predictive model. So, one must make sure that the collected data are recent, relevant, accurate, consistent, complete, free from redundancies and errors. Although the collected data can be in raw or feature form, to discuss the dataset generation pipeline in details, let us assume that data are collected in raw form.

2.6.1.2 Preprocessing

After data collection, the Preprocessing stage in the pipeline involves a sequence of steps which prepare the data for further analysis. Collected raw or feature data generally, should not be used in original form for predictive analysis. To generate a quality dataset, collected data should first be cleaned, transformed and prepared for effective prediction. It is the most crucial step in the dataset generation pipeline as it involves significant tasks as discussed in the following sections. At this stage, two important questions need to be answered, *i) Are all the preprocessing tasks necessary for the problem that one wants to solve?* and *ii) How does one decide which tasks to perform for a particular problem?*. The answer to the first question is of course *No*. That is, for a particular problem, all the preprocessing tasks may not be necessary. This leads to the answer for the second question, that is, which tasks and when to perform the tasks largely depend on the data at hand.

1. Feature Extraction: Primarily, feature extraction is a pre-processing task which aims to reduce the dimensionality of data. Data, in raw form, may not be suitable for prediction tasks. So features and important characteristics need to be extracted from the raw data. The most widely popular feature extraction algorithm is probably the Principal Component Analysis Algorithm,

which tries to generate a set of principal components or orthogonal features.

2. Missing Value Estimation: Sometimes it may so happen that a data value (of a particular row or column) is not recorded for a given sample. These are the missing values in the dataset, and estimating these values is a major preprocessing task. Due to various technical issues, missing values may arise in the data. Issues such as faults in the measuring equipment, or some drawbacks or errors in the data collection phase. Missing values in the data can have serious implications in the overall results, and hence accurate estimation is an important task. The following are some of the popular missing value estimation techniques.

   (a) Ignore and discard the data instances: Perhaps the simplest way to deal with the problem of missing values is to discard the particular instance that does not have the value. However, this technique is not suitable when missing values occur for a large number of instances in the dataset.

   (b) Parameter estimation: This technique makes use of appropriate statistical and machine learning techniques such as k-nearest neighbors to estimate the missing values in the data.

   (c) Imputation: This technique helps estimate the missing value in the data by taking into consideration the information already present in the data. One popular estimation technique under the imputation method, is replacing the missing value with the mean of the existing values for the attribute.

3. Sampling: Often, it is seen that datasets suffer from the problem of class imbalance. The problem of class imbalance arises when instances of a particular class or category are present in sufficiently higher count compared to instances in other classes or categories. For example, let us consider a dataset with 1000 instances and two classes A and B. For class A, there are apparently 800 instances whereas for class B there are 200 instances only. Thus, the dataset is imbalanced. This problem may result in poor performance of the predictive model and hence should be handled through sampling. There are two kinds of sampling techniques to handle the problem of class imbalance, undersampling and oversampling. Undersampling refers to reducing the number of instances in the class with higher counts of instances by ignoring some samples, whereas oversampling refers to increasing the number of instances

in the class with lower counts of instances by considering some samples more than once. Both techniques aim to balance the number of instances in the two classes.

4. Zero Variance Removal: Sometimes, a dataset may contain features with a constant value, meaning all through the samples of that feature there is zero variance. For example, a feature with all values 0 throughout the dataset. Such features can be removed from the dataset as they do not contain any interesting pattern. Presence of such features only tend to overfit the predictive model.

5. Feature Selection: This step involves selecting or choosing a subset of informative features from the original set of features as shown in Figure 2.25. This subset is supposed to be relevant and adequately describe the problem at hand. Section 2.8.1 discusses on Feature Selection methods and approaches elaborately.

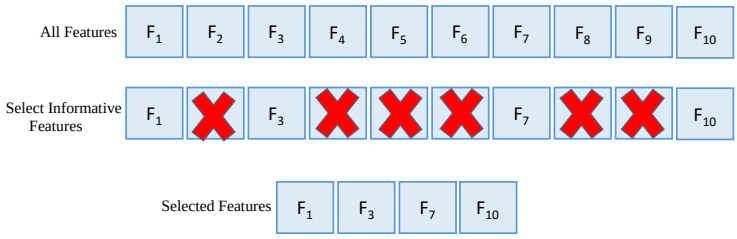


Figure 2.25: Example of Feature Selection

6. Normalization: When data values in a dataset do not belong to a uniform scale, transformations such as normalization may be used. For example, a dataset may have, say, two features f_1 and f_2, where f_1 takes values between 0 to 1000, and f_2 takes value between -1 and 1. Such a dataset will not perform well if proximity measures are used. So, the feature values should be brought into a uniform scale. There exist several established normalization techniques. Some popular ones are discussed below.

   (a) Min-Max normalization: These technique transforms the values in the data examples into a uniform scale of 0 to 1 using Equation 2.1.

$$X_{normalized} = \frac{X - X_{min}}{X_{max} - X_{min}}, \tag{2.1}$$

   where $X_{normalized}$ is the normalized value corresponding to the feature value X, X_{max} and X_{min} are the maximum and minimum value of the

attribute. However, this technique is known to suffer in the presence of outliers.

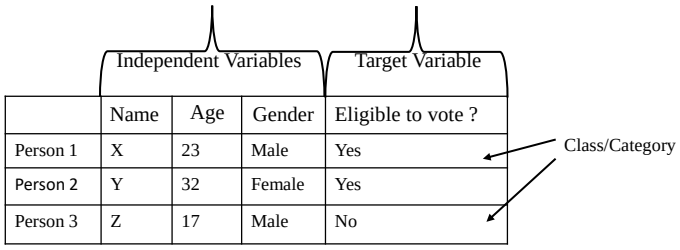
(b) Z-score normalization: This technique makes use of the mean and standard deviation to transform the values of an attribute as shown in Equation 2.2.

$$X_{zscore} = \frac{X - X_{mean}}{Std\_dev(X_{mean})} \tag{2.2}$$

where X_{zscore} is the normalized value of the feature value X, X_{mean} is the average value of the attribute values of X and Std_dev is the standard deviation. This technique is less sensitive to outliers.

(c) Log normalization: This technique is useful in cases of datasets with skewed feature values. The skewness in the data is removed by normalizing the data, bringing it more close to normal distribution.

7. Labeling: Labeling a dataset is the task associated with assigning each instance or sample of the dataset to a particular category or class. Especially, in the case of Supervised Learning (as discussed in Section 2.7.1), the label or target variable is essential for training and testing a learning model. Labeling a dataset generally requires expertise from domain experts. Figure 2.26 is an example of a labeled dataset with two separate categories or labels in the data.



	Name	Age	Gender	Eligible to vote ?
Person 1	X	23	Male	Yes
Person 2	Y	32	Female	Yes
Person 3	Z	17	Male	No

Figure 2.26: Example of a Labeled Dataset

2.6.2 Types of Datasets

To tackle any machine learning or data analysis task, the first and foremost focus should be on the data. The type and quality of the data used plays the crucial role in such a task. Any developed model can be tested and validated only with the help of appropriate datasets. Based on how and for what purpose data are generated, there can be three kinds of datasets as discussed below.

2.6.2.1 Synthetic Data

Sometimes the need may arise to artificially generate a dataset which mimics real life activities and situations. Such datasets are synthetic datasets, which actually do not contain real information, but synthetically generated information. These datasets are important because one can incorporate all the necessary test cases required to exhaustively test the developed method. Such test cases may not often be available in the existing datasets. It is also important to note that the data distribution followed in the synthetic data must be similar to the distribution found in the real world data. One example of such a dataset is a credit card fraud dataset which mimics a company's customers' buying and spending habits. The following are some of the situations where the need for synthetic datasets arise.

1. When existing datasets do not contain updated test cases to validate a recently developed method.

2. When working with sensitive or confidential data. For example, in case of the healthcare datasets, a patient's personal information should remain confidential.

3. When there is no publicly available dataset to work with.

2.6.2.2 Generated Data

Many a times real world data collection is not feasible, or may be limited, or data may be inaccessible. In such cases, data are created through simulation and experiments. Data can also be created by setting up a small scale testbed that mimics a large scale industrial facility. Data produced through such processes are called generated data. Such datasets are useful for research modeling and testing of real world problems and solutions. For example, for climate research on a particular geographical area, the weather conditions for that particular area over a span of time can be simulated with the help of appropriate data generation tools.

2.6.2.3 Benchmark Data

Benchmark datasets are the standard datasets that have been established as reference in the literature. Such datasets are used for performance evaluation, assessment and comparison of newly developed research methods. Newly proposed

datasets can also be compared against the benchmark datasets to assess if they provide the same kind of performance. A few examples of benchmark datasets in network security are: KDDcup99 dataset[5], NSL-KDD dataset[6], DARPA 2000 dataset [7].

2.7 Machine Learning Approaches

Computer programs have provided solutions to a wide range of problems from various fields. While there are problems which can be readily solved by human experts and machines alike, there are some which are too complex and need intensive computations. Such absorbing problems require reliance on computers or machines to supplement human expertise. Especially, predictions or decisions provided by machines can have profound impact in cases where there exists no or minimal human skill, or cases where human experience is unexplainable, or cases where data evolve from time to time [49]. Employing machines to undertake a demanding problem would be of interest to stake hold because a machine may be able to gain deep, accurate and effective insights into the problem beyond what is possible for humans. However, the bigger question here is how to make a machine learn to solve a problem by itself. Machine learning has truly changed the picture how the world views computer programs. It has become indispensable in almost every field of interest to society.

Machines can learn to solve a problem with the help of learning algorithms. These learning algorithms feed on data and grasp the underlying interesting and useful patterns in the data [117]. As it learns from the existing data, it improves its performance via training with the goal to undertake predictive tasks on unseen data [118]. This is the reason why existing data should be of the quality representative of the input to be received. Data are available in the form of datasets. A dataset can be thought of as a matrix. While each row represents an instance, each column depicts a feature or attribute describing a characteristic of each instance, with the exception of the last column. The last column serves as the knowledge (in case of supervised learning only) in solving a specific task and denotes the class label for each instance.

[5] http://www.kdd.ics.uci.edu/databases/kddcup99/

[6] http://www.iscx.ca/NSL-KDD/

[7] http://www.ll.mit.edu/mission/communications/ist/corpora/ideval/data/index.html

Depending on how learning models learn from data, machine learning approaches can be of different types: a) supervised, b) unsupervised, c) semi-supervised, and d) reinforcement learning. In supervised machine learning, the label information corresponding to each data instance in the dataset serves as the knowledge. In unsupervised learning, the label information is absent. In semi-supervised learning there is a bit of both, corresponding to partial knowledge.

Supervised Learning: Let's understand the task of supervised learning with the example as shown in Figure 2.27, where the task is to correctly identify a given shape to be either circle, triangle, rectangle or pentagon. As input, a learning model is provided with a set of objects (in terms of features) along with their respective labels (or responses). Next, let's assume one is given a prediction task where he or she has to predict the shape of an input instance as either *'circle'* or *'not circle'*. Meaning, the given task is a 2-class classification problem. The class *circle* will include all the samples whose shapes are actually a circle and the class *not circle* will include samples with shapes that are rectangle, pentagon and triangle. During the training phase, the model learns how to differentiate a circle from other shapes based on characteristic like the *number of corners and the number of straight lines* (size and color do not play any role in this example). After the model is built, it is ready to be tested with a test set. This test set consists of previously unseen instances whose classes need to be predicted by the model based on the training it received. Let us suppose that the model receives a test instance, as shown in the illustration; then the model would predict its shape to be a circle. This means the test instance belongs to class *circle*. In this way, the test instances are mapped to their respective classes.

2.7.1 Supervised Learning and Its Significance

The type of machine learning that uses prior knowledge (in the form of labels in the data) is called Supervised machine learning [119]. Mathematically, a set of input variables X serves as input to a function f which tries to map X to an output variable y as shown in equation 2.3. Here, X is a set of instances where each $x_i \in X$. Each instance is characterized by a set of attributes or features or characteristics. y is the target variable or response or label, which signifies to which category each instance in X belongs. The end goal of supervised learning is to accurately predict y for a set of previously unseen instances X_{new} (X_{new} that are not present in X).

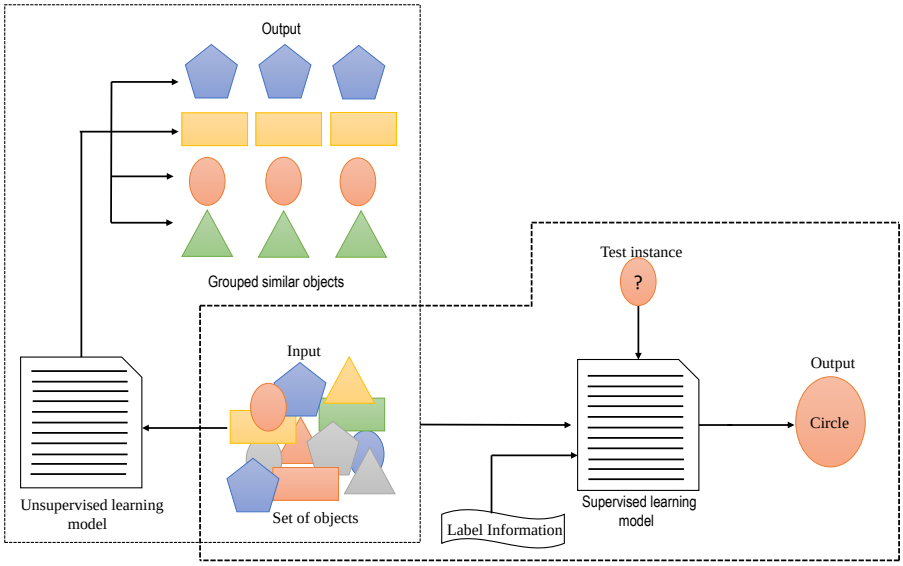


Figure 2.27: Example of Supervised/Unsupervised Learning Technique

In supervised learning, a model goes through two essential phases namely, training and testing. During the training phase, the model learns by training with examples of known instances and their labels, the task of how to differentiate between two specific categories. After training, the model is put to test (testing phase) where it predicts the class labels of previously unseen data instances. Figure 2.28 shows some commonly used supervised learning algorithms.

$$y = f(X) \tag{2.3}$$

Figure 2.29 illustrates the process of how supervised learning can be applied to

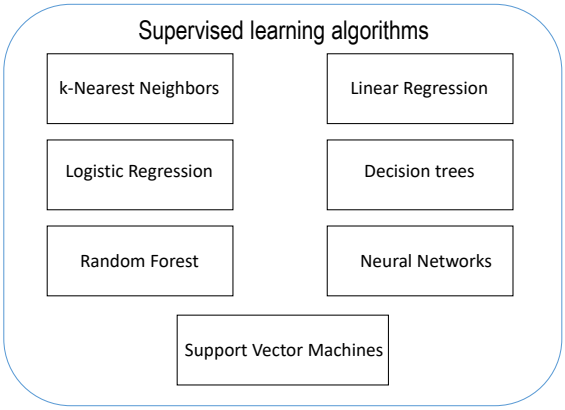


Figure 2.28: Some Supervised Learning Algorithms

problems in the real world. Once the problem statement has been formulated, data need to be collected and prepared. This data collection process needs to be

exhaustive and extensive as the quality of the learning model depends on it. Several publicly available repositories are available like UCI ML library[8], GitHub[9], Kaggle[10], AWS[11], and Google Cloud[12] from where datasets can be collected. Once collected, the data need to be appropriately scrutinized in raw form so that there is no inconsistency, exceptions, or incorrect and skewed information. This is necessary for the data to be of high quality. This is followed by data preprocessing, which may involve a number of activities like finding or removal of missing values, normalization of feature values, feature extraction, and feature selection. The preprocessed data are then split into two parts, namely the train set and the test set. The train set should be an appropriate representative of the unseen input that is to be received later during prediction. The test set is put aside for later use, and the training set is given as input to the selected supervised learning algorithm. In this step, the hyper-parameters of the learning algorithm can be specified; for example the number of trees to grow in a random forest. During the next step, the learning model is built from the training data, and then evaluated on the test set with the help of evaluation metrics. If the performance is not as expected, then there can be several possible reasons as mentioned below. If the performance is as desired, then what is obtained at the end is a predictive model which predicts classes of the newly arriving instances.

1. The collected data may have inconsistencies, or outliers or may be incorrect.

2. The preprocessing step may not be adequate. For example, the feature selection activity may not have yielded all relevant attributes required.

3. Splitting the dataset into train and test set may not have been appropriate, meaning the train set was not inclusive of all the possible cases.

4. The hyper-parameter set during training may not be properly tuned, in which case re-tuning of the model is required.

Next, a few other basic terms are introduced which will be helpful in understanding the concepts well.

[8] `https://archive.ics.uci.edu/ml/index.php`
[9] `https://github.com/awesomedata/awesome-public-datasets`
[10] `https://www.kaggle.com/datasets`
[11] `https://registry.opendata.aws/`
[12] `https://cloud.google.com/bigquery/public-data/`

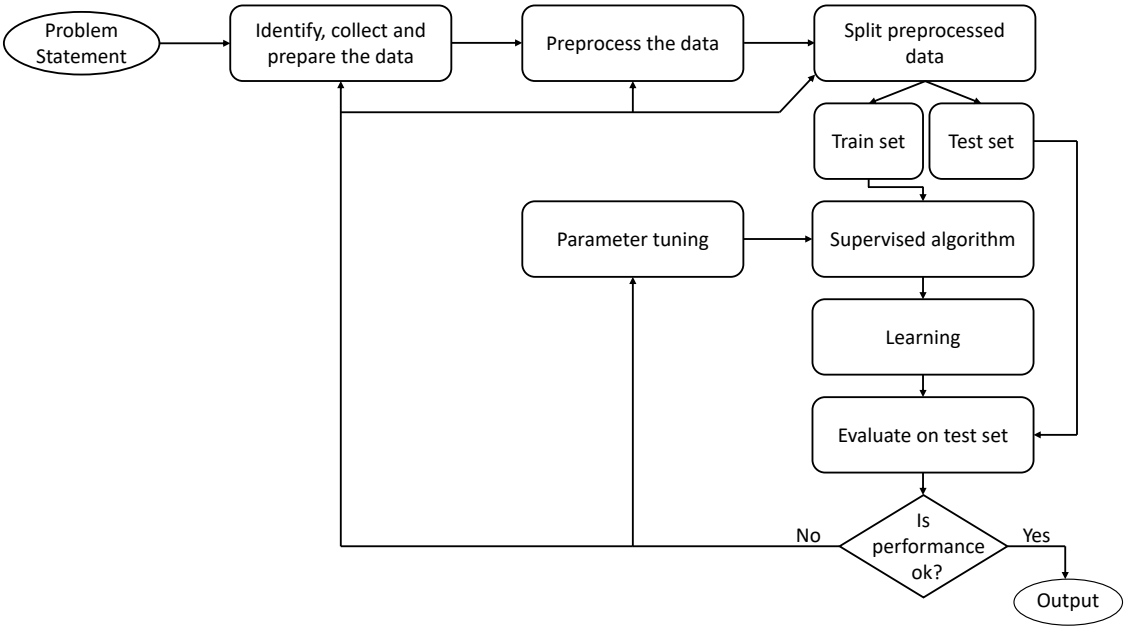


Figure 2.29: Supervised Learning Framework

1. *Generalization:* It describes the capability of a machine learning model to work well with varied unseen data, after it has gained experience by learning through labeled example [118].

2. *Bias:* It is a measure of the average number of times a model predicts a different value for a sample than its actual value. A high value of bias means that the model does not perform well as it is not successful in learning important intrinsic patterns in the data.

3. *Variance:* It is a measure of how varied the predictions are made by the model on same set of observations. High variance leads to over-fitting the training population.

4. *Over-fitting:* It describes the situation when a model learns the training data too well but subsequently under performs on previously unseen data. In other words, the model learns the patterns very well when taught by examples, but fails to perform well when testing on unknown samples.

2.7.2 Unsupervised Learning and Its Significance

Unlike supervised learning where knowledge is provided in the form of labels along with the input data, in unsupervised learning, knowledge is not provided in any form. As shown in Figure 2.30, the input to the learning model is a set of objects, in

terms of features. The problem is to organize similar objects into groups based on basic characteristic, for example, the number of corners and the number of straight lines (not size or color) in the problem being discussed. So the groups in this case may be something like, *shapes with no corners, shapes with three corners, shapes with four corners* and *shapes with five corners*. Accordingly, circles, triangles, rectangles and pentagons may be arranged into different groups.

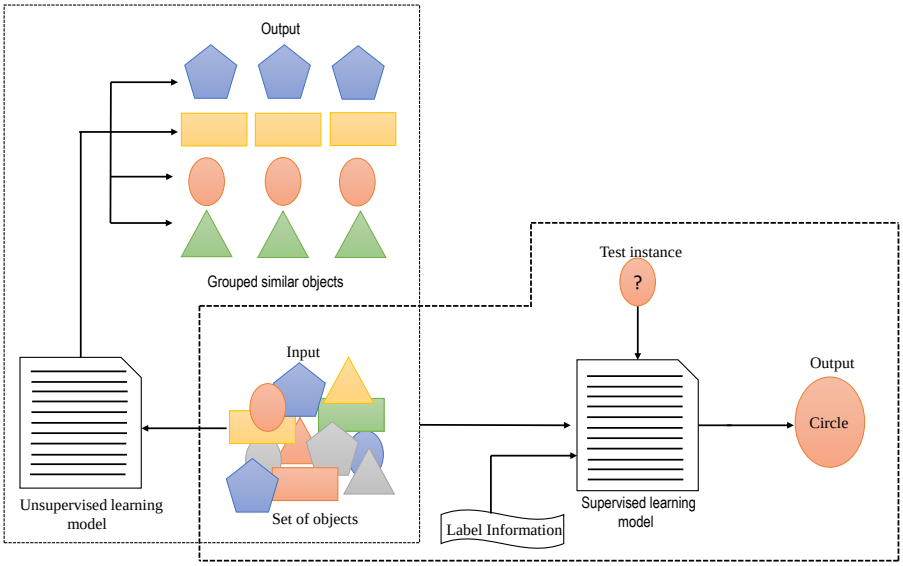


Figure 2.30: Example of Supervised and Unsupervised Learning

2.7.3 Ensemble Learning Approaches

Ensemble learning deals with a diverse set of learners, and combines them in some manner to achieve the best possible overall accuracy [120]. The basic concept behind ensemble learning is that a team's decision is usually far better than the decision of an individual. The ensemble of learners help eclipse the poor performance of an individual learner. The learners taking part in the ensemble are called base learners. A quality ensemble is one in which the base learners are from diverse families and independent of each other [121]. This is highly important because the errors made by base learners should be uncorrelated so that possible misclassification by any of the base learners can be averaged out when correctly classified by other learners. Similar learners, if combined, tend to make similar kinds of errors, which do not contribute to the overall accuracy. The end goal of constructing the ensemble is obtaining an improved and stable model with high precision and accuracy. A combination function is used to aggregate the decisions made by the

individual learners. An example is *majority voting*; if two learners classify an instance to be in class A and three learners classify it to be in class B, then the final output would be the instance classified into class B as majority learners voted for it to be in class B [122].

2.7.3.1 Construction of an Ensemble

An ensemble can be constructed in several ways at different levels. The end goal of an ensemble is to better the decision making process and improve the overall performance of the predictive learners. Below some popular approaches to construct an ensemble [123] are discussed.

1. *Constructing different training sets:* From a given pool of sample training data, several random training sub-sample sets are formed. A learning algorithm is then trained on each of these training sets to obtain different classifiers. Techniques such as resampling [124] and reweighing [125] are used to obtain sets of random training sub-samples.

   (a) Re-sampling: Drawing samples randomly from a dataset in a repetitive manner.

   (b) Re-weighing: Weights of misclassified instances and possibly correctly classified instances also, are adjusted in every iteration so that the focus on misclassified examples is boosted.

2. *Constructing different feature subsets:* Different subsets of features can be selected by choosing features at random or using measures such as mutual information, correlation, or information gain, to name a few. Learning models are then trained on these varying feature subspaces to get distinct classifiers. Since, the feature spaces are likely to be distinct each model may learn some unique characteristics from the data.

3. *Choosing different learning models:* Perhaps the most common technique to construct an ensemble is to choose models belonging to different families of classifiers. For example, an ensemble may contain models based on stable classifiers such as nearest neighbor-based, or unstable classifiers such as tree-based classifiers. Another significant alternative could be varying the parameters of the models to bring about change in the outputs.

4. *Combination schemes:* In this technique, the outputs of different learning models are combined to get the final output using different schemes. Majority voting, weighted majority, mean rule, sum rule, are some of the examples of such schemes [123].

2.8 Cost Effective Methods for Attack Analysis

Cost effective approaches and methods are important to detect attacks in network security because timely detection and response is of utmost importance. The ultimate goal is to minimize the damage caused by attacks, and this is possible only if the attacks are detected in a timely manner and as accurately as possible. In the upcoming subsections, some cost-effective methods are discussed.

2.8.1 Feature Selection Methods

Feature selection is the task of selecting a subset of features from a group of available or candidate features to improve performance and accuracy of a model [126]. The selected subset of features may better define the problem at hand compared to all the original features. In other words, for a given problem, not all available features may be relevant and that is why one needs to choose just the subset which can improve the overall performance of the model. The presence of unnecessary features degrades the performance of the model and may result in overfitting. The subset of features may be chosen based on various criteria such as relevance, correlation, and mutual information, to name a few. The feature subset should be chosen in such a way that there is minimum redundancy but maximum relevance. If two features are perfectly correlated, then it is a case of redundancy, since adding both to the subset does not provide any additional information [127]. In this case, one needs to consider only one of them for inclusion in the subset. Secondly, it may so happen that two features provide useful information only when they are together; individually, they are not useful. In such a case, one needs to include both features in the subset. Four types of feature selection methods are commonly used and are discussed below.

1. Filter Methods: Such methods identify feature subsets without the help of a learning algorithm and rely on the intrinsic characteristics of the data at hand as shown in Figure 2.31. Instead of depending on a predictive model,

statistical tests such as correlation or mutual information are used to rank (or score) the features with regards to the target variable. In other words, a ranking criterion helps rank the features, and appropriate thresholds may be set to remove features which are not relevant and independent. One important aspect of such methods is that they are known to be computationally efficient as they do not need reruns of the learning model over the feature space [128]. However, such approaches may under-perform in providing an optimal feature subset in terms of accuracy.

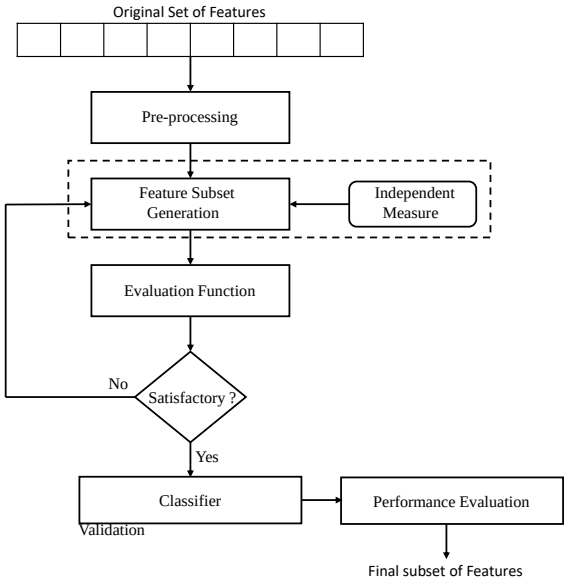


Figure 2.31: Filter Method

2. Wrapper Methods: These methods rely primarily on the learning models for evaluation of the selected feature subset's performance [129]. As a result, these methods are computationally expensive and require more time to reach a conclusion compared to other feature selection methods. Figure 2.32 shows how these methods work. It is more likely to obtain an optimal feature subset with wrapper methods compared to others as they search the feature space (either by adding or removing features) for better accuracy. The evaluation process continues until the desired performance is met or a specified stopping criterion is reached. Popular examples of such methods are the *forward selection* and *backward selection* algorithms.

3. Embedded Methods: These methods refer to learning models which have feature selection methods integrated into them as shown in Figure 2.33. From the original feature set, firstly the learning algorithm generates a feature

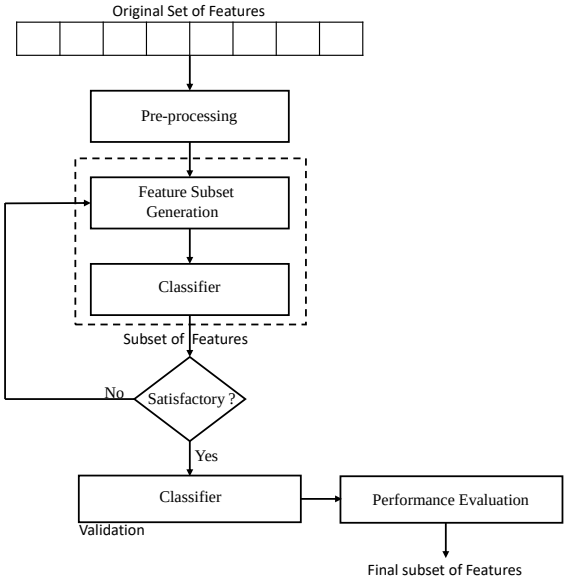


Figure 2.32: Wrapper Method

subset which is then used to assess the learner's performance. Iteratively this process continues until the best subset is obtained.

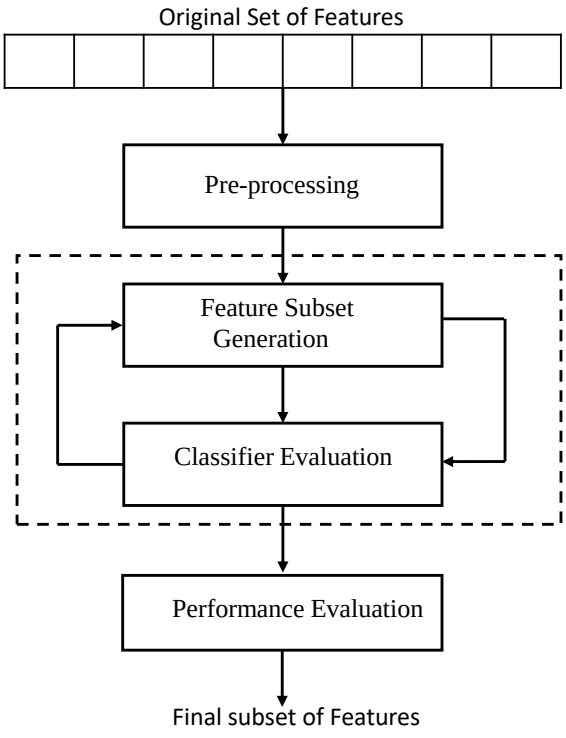


Figure 2.33: Embedded Method

4. Hybrid Methods: These methods strike a balance between the pros and cons of filter and wrapper methods by combining both [130]. A filter method first selects a candidate feature set which is then evaluated by a wrapper method

to assess its predictive performance. Figure 2.34 shows how these methods function.

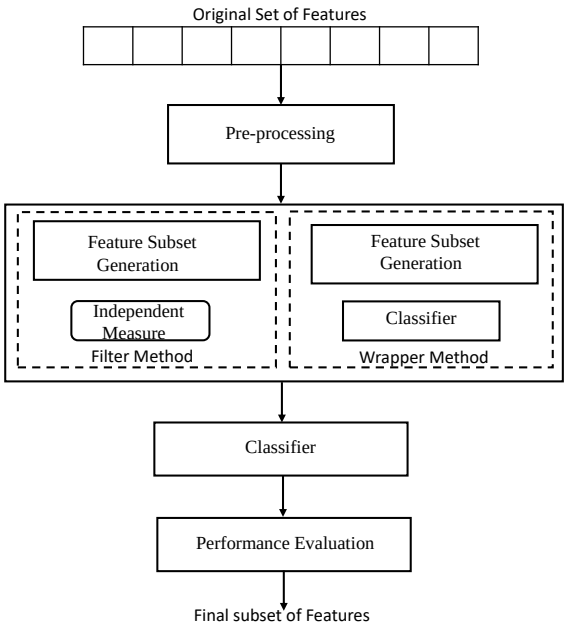


Figure 2.34: Hybrid Method

An important feature selection technique in this context is the Recursive Feature Elimination (RFE) technique which tries to reduce the number of features from a ranked feature list recursively. At each iteration, a validation measure (such as accuracy, or F1-score) may be used to validate the model performance. The ranked list of features are given as input and in each iteration the least ranked feature is eliminated. RFE helps in reducing overfitting, enhancing the model performance, and speeding up the training process by reducing the total number of features.

2.9 Validation Measures

After a prediction method is developed, the next task is to check how good the predictive model is. For this, validation measures, also known as evaluation metrics or performance metrics are used [131]. With the help of these measures one can demonstrate how good a predictive model is in terms of performance. Many validation measures exist. Which measure to use depends on two considerations. First, whether it is a classification or regression task (in case of Supervised Learning). Second, what kind of data is acquired to train and test the model.
For now, let us assume that a defense approach developed for binary classification

of normal and attack instances is to be evaluated using the metrics discussed below. The attack class is assumed to be positive and the normal class is assumed to be negative. With respect to an instance, the defense approach makes a prediction, i.e., whether a previously unseen instance belongs to the normal class or attack class.

1. Accuracy: The accuracy measure indicates how correct a predictive model is. It is widely popular because it not only measures the detection and failure rates, but also the number of false alarms raised by the predictive model. For example, if a prediction model gives 97% accuracy, it means that the model correctly classifies 97 instances to the actual class out of total 100 instances. The accuracy measure is defined as in Equation 2.4. The accuracy metric is most widely used because it is easy to understand and less complex than the others. It can be used with binary, multiclass as well as multilabel classifications.

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{2.4}$$

2. Precision: This measure indicates the number of relevant instances among all the retrieved instances. That is, it tries to answer the question, "of all the positively predicted instances how many were actually positive?". The precision measure is defined as in Equation 2.5.

$$Precision = \frac{TP}{TP + FP} \tag{2.5}$$

3. Recall: It indicates the number of relevant instances that were retrieved. Also known as sensitivity, it tries to answer the question, "of all the positive instances present, how many were predicted as positive?". The recall measure can be defined as in Equation 2.6.

$$Recall = \frac{TP}{TP + FN} \tag{2.6}$$

4. F1-score: F1-score is nothing but the harmonic mean of Precision and Recall measures. When there is an imbalance between the classes in the dataset, the preferred metric is F1-measure over accuracy as in Equation 2.7.

$$F1 - score = \frac{2}{\frac{1}{Recall} + \frac{1}{Precision}} \tag{2.7}$$

5. Misclassification rate: This measure indicates the number of times a predictive model misclassifies an instance. That is, the model predicts an instance to be of a particular class; however, its actual class is a different one. It can be expressed as in Equation 2.8.

$$MisclassificationRate = \frac{FN + FP}{TP + TN + FP + FN} \tag{2.8}$$

   The precision and recall measures are actually inversely proportional to each other.

The next chapter introduces the datasets which are used for evaluating the proposed work in this thesis.

Table 2.2: Existing Approaches for Detection of XSS Attacks

Analysis type	Author name	Way of operation
		Client-side detection approaches
Static	Gupta et al.[35]	XSS-Immune framework relies on context-aware sanitization and JavaScript string comparison between the scripts that are embedded in an HTTP Request and the corresponding HTTP Response. If any potential malicious activity is found, proper sanitization mechanisms are invoked.
Dynamic	Reis et al. [52]	BrowserShield is a framework which functions by rewriting (any malicious) HTML pages. The HTML pages may have embedded scripts on which policies are enforced during runtime before execution by the browser. Henceforth, a safe equivalent of the page is produced. Due to dynamic behavior of a Web page safety is ensured by recursively applying runtime checks on the embedded scripts.
Hybrid	Curtsinger et al. [53]	The authors propose ZOZZLE, a classifier based JavaScript deobfuscator, which can be deployed in a browser to detect and prevent XSS attacks. To differentiate malicious JavaScript code from benign code, an Abstract Syntax Tree (AST) based technique is used. This technique makes use of hierarchical context-sensitive features for detection.
		Server-side detection approaches
Static	Gupta et al. [54]	XSS-Secure is a service provided for the Online Social Networking-based (OSN) multimedia Web applications. The training mode deals with context-sensitive sanitization of untrusted variables of JavaScript. In the detection mode, XSS-Secure carries out the important task of weighing the similarity between the sanitized HTTP response generated at the OSN Web server and the stored response at the repository. Deviations, if detected, help reach the conclusion that XSS worms were injected into the OSN servers.
Dynamic	Guo et al. [55]	It is an XSS attack vulnerability detection technique that uses an attack vector repository which is generated automatically and is built using attack vector patterns such as HTML tag attributes or event attributes, resource repositories and mutation rules. For vulnerability detection, the injection points are identified and the attack vectors are fed as input.
Hybrid	Jaballah and Kheir [56]	The authors propose a grey box approach to detect malicious interactions in Web applications. While behavior graphs are constructed from a set of real world user sessions, attack graphs comprise of sequences of actions taken or the security flaws exploited by an attacker. An intermediate event graph is produced which undergoes sub-graph isomorphism with the behavior graphs. This helps in phasing out the benign interactions of the legitimate users and the malicious behaviors or attacks.
		Client-server detection approaches
Static	Wasserman & Su. [57]	The method aims to detect XSS vulnerabilities arising due to weak input validation. It checks if any untrusted input to a Web application server may result in the invocation of the JavaScript engine in the browser. It is an amalgamation of both string analysis and tainted information flow.
Dynamic	Goswami et al. [58]	This unsupervised method first performs a minimal checking at the client-side using a divergence measure. If a particular threshold is crossed, the request is discarded immediately. If it doesn't, the request is transferred to the proxy. The data collected at the proxy level undergoes preprocessing and feature extraction, followed by a feature selection mechanism, Rank Aggregation. An attribute clustering algorithm is employed at the proxy to detect attacks.

Table 2.3: Summary of Machine Learning Approaches

Analysis Type	Author name	Area of focus	Type of XSS	Deployment
Static Analysis	Likarish et al. (2009)[59]	Attack Detection	Reflected and Stored XSS	Client-Server
	Nunan et al. (2012)[60]	Attack Detection	Not specified	Client-Server
Anomaly based	Kruegel et al. (2003)[61]	Attack Detection	Reflected and Stored XSS	Server-side
	Robertson et al. (2006)[62]	Attack Detection	Not specified	Server-side
	Song et al. (2009)[63]	Attack Detection	Reflected and Stored XSS	Server-side
Dynamic Analysis	Guo et al. (2015)[55]	Vulnerability Detection	Reflected and Stored XSS	Server-side
	Goswami et al. (2017)[58]	Attack Detection	Reflected and Stored XSS	Client-Server

Table 2.4: Existing Approaches for Detection of HTTP Flooding Attacks

Year	Author name	Way of operation
2017	Singh and De [107]	The authors use a multi layer perceptron (MLP) to detect HTTP Flooding attacks with features such as HTTP requests count, number of the IP addresses. For learning and adjusting the weights of the perceptrons genetic algorithm is used. The main advantage of this method is that it provides a high detection accuracy and a low false positive rate.
2018	Zhao et al. [108]	In this approach the authors try to differentiate between flash crowd and application layer DDoS attacks. Two measures based on entropy namely, EUPI (Entropy of URL per IP) and EIPU (Entropy of IP per URL) are used. The main idea behind using such measures is that normal requests vary in size, speed and intent and as such have a high entropy value. Whereas attack packets are more similar to one another and thus have low entropy value.
2011	Liu et al. [109]	The authors present a mechanism to differentiate between the legitimate and malicious user behavior. The features used to differentiate the users through connection are instant traffic volume, session behavior, etc. Depending on the behavior of users, the mechanism provides different services to the users.
2017	Sreeram et al. [110]	The authors propose a bio-inspired anomaly based detection approach to detect HTTP Flooding attacks. Features are extracted from a request stream observed during an absolute time interval, rather than from user sessions and packet patterns. These features help uniquely identify if a request is normal or malicious. Then, a swarm intelligence bat algorithm is used for the classification purpose.
2010	Wen et al. [111]	The authors propose a traffic estimation based defense mechanism which focuses on request dynamics. If the request rate is above an expected threshold at a particular point of time it means something abnormal (either an attack or flash crowd) is going on. Kalman filter is used as a measure for this purpose. Additionally, source IP distribution is used to actually identify an attack flood.
2015	Ndibwile et al. [112]	The authors present an approach which employs three servers for detecting the attacks. First, is a bait server which handles all the request initially. If the traffic is found to be normal then it is forwarded to the second server which is the real server. All the suspicious traffic is sent to the third server, the decoy server. The suspicious traffic undergo classification using decision trees, trained with the help of attack generation tools.
2022	Beitollahi et al. [113]	The authors introduce a detection method based on the combination of Cuckoo search algorithm and Radial Basis Function-Neural Network. The authors emphasize the importance of preprocessing tasks such as data cleaning, normalization and feature selection using Genetic Algorithm (GA). The neural network is trained using Cuckoo Search Algorithm and is successful in acheiving high accuracy.
2014	Choi et al. [114]	The authors present a MapReduce detection method for HTTP-Get flooding attack detection. Interestingly, the proposed method improves upon critical parameters such as availability, precise decision making, and reliability. Moreover, it outmatches the snort detection system because of its low processing time.
2022	Rustam et al. [115]	The authors introduce a detection framework based on machine learning which employs a multi-feature approach. The proposed approach combines features obtained from both Principal Component Analysis (PCA) and Singular Value decomposition (SVD)features to achieve high performance.