THE CASE OF CROSS-SITE REQUEST FORGERY AND MACHINE LEARNING FOR WEB VULNERABILITY DETECTION

J.V.Anil Kumar[1], P. Lakshmi Usha Sri[2], K. Bewala[3], G. Kavya[4], SK. Abdul Munaf[5], M. Amith[6]

[1] Professor, Krishna Chaitanya Institute of Technology & Sciences , Markapur, A.P, India
[2,3,4,5] Scholar, Krishna Chaitanya Institute of Technology &Sciences , Markapur, India

ABSTRACT:

In this paper, we present a method in this research for using machine learning (ML) to identify weaknesses in web applications. Web applications provide a particularly challenging set of analytical issues because of their variety and frequent usage of custom programming approaches. Because it may employ manually labelled data to incorporate automatic analytic tools with a human's understanding of the semantics of online applications, machine learning is therefore very advantageous for web application security. We used our methods to develop Mitch, the first machine learning (ML) tool for detecting Cross-Site Request Forgery (CSRF) vulnerabilities. Mitch assisted us in discovering 35 new CSRFs on 20 important websites and 3 new CSRFs in production applications.

Keywords-Machine Learning, Mitch, vulnerability, detection techniques.

[1] INTRODUCTION

A web application is the most widely used interface for today's security-sensitive data and services. They are regularly used to file tax returns, check the results of medical examinations, carry out financial transactions, and discuss thoughts with our social circle, to mention a few popular functions. On the other hand, malicious users (attackers) might find web applications to be appealing targets if they wanted to inflict financial losses, get unauthorised access to personal information, or defame their victims. Security for web applications is notoriously difficult.

The online platform's variety and complexity, as well as the usage of shoddy scripting languages with uncertain security guarantees and unsuitable for static analysis, are all contributing reasons. In this context, black-box vulnerability detection approaches are extremely popular. Instead of requiring access to the web application source code, white-box techniques operate at the level of HTTP traffic, i.e., HTTP requests and responses. Black-box approaches operate at this level. The main benefit of this confined approach, even though it may miss important insights, is a language-agnostic vulnerability detection technique that abstracts from the complexity of scripting languages and gives a common interface to the widest range of web applications.Despite how appealing this looks, previous research has demonstrated that such an evaluation is everything but simple.

One of the main problems is how to develop automated algorithms capable of comprehending the semantics of the web application, which is a crucial part of effective vulnerability discovery. Example: Cross-Site Requests Made in error (CSRF) A well-known online attack called Cross-Site Request Forgery (CSRF) induces a user to submit unauthorised, attacker-controlled HTTP requests to a vulnerable web application in which she is now signed in. The basic tenet of CSRF is that because malicious requests are transmitted to web apps through the user's browser, it may be challenging to distinguish them from legal, innocuous ones that the user has really authorised.
A typical CSRF attack works as follows:
1) For instance, Alice logs onto her preferred social network, an accessible yet reliable online service. The malicious advertisement uses HTML or JavaScript to send a cross-site request to the social network, for example, asking users to "like" a particluar page;
2) Alice presses on a different tab and traverses to a wholly unrelated website, like an online newspaper, which returns a page with a malicious advertisement;
3) The suspicious advertising uses HTML or JavaScript to send a cross-site request to the social network, for example, asking users to "like" a particluar page;
Because the request contains Alice's cookies, it is processed in the context of her social network authentication. The deceptive advertising may affect the outcomes of online polls by pressuring Alice to "like" the relevant political party.
It should be emphasised that CSRF merely demands that the Stopping CSRF attacker must not intercept or change the user's requests or responses.

To avoid CSRF, web developers must utilise particular security measures. If requiring extra user interaction does not severely impair usability, it is possible to impose re-authentication or employ one-time passwords/CAPTCHAs to prevent cross-site searches from coming through unnoticed. The recently available Same Site cookie capability may be used to avoid cookie attachment on cross-site requests. It is highly recommended for new web applications to use this attribute because it tackles the CSRF issue at its core. However, automated prevention is typically preferred. Unfortunately, the majority of online services now in use block out cross-site requests using one of the following techniques:

Verifying the existence of specific HTTP request headers, such as X-Requested-With, which cannot be set from a cross-site position
1) examining the value of common HTTP request headers like Referrer and Origin, which identify the page from which the request originated
2) verifying the existence of erratic anti-CSRF tokens, which the server inserts into vulnerable forms
3) verifying the existence of cross-site incompatible HTTP request headers.

In a recent study, the advantages and disadvantages of these diverse tactics were discussed. The disadvantage shared by all three options is that security checks must be strategically positioned. For instance, tokens should be attached to all and only the HTTP requests that are security-sensitive in order to provide complete protection without affecting the user experience.

A token on the homepage of a social network is undesirable since it could lead to the denial of valid cross-site requests, including those from clicks on search engine results that include the social network. However, the above-mentioned attack may be avoided by utilising a token to secure a "like" button. For web developers, determining the "perfect" placement for anti-CSRF measures eventually proves to be a difficult problem. Although modern web application development frameworks allow this automatically, CSRF vulnerabilities are commonly found even on highly rated websites. This increases the need for trustworthy CSRF detection systems. But how can we use automated techniques to enable CSRF detection if we can't figure out which HTTP requests are actually security-sensitive? Votedon – Nosplits.

The most recent and comprehensive description of the little-known internet vulnerability known as CSRF is provided in this book, along with specific techniques to find and avoid CSRF vulnerabilities (Cross-Site Request Forgery). The immediate benefits of this effort include a tangible and useable application framework that individuals, organisations, and developers may use to either consume or offer web services.

In contrast to the traditional anti-virus and anti-spyware strategies, this research specifically addresses the challenges of keeping up with the quickly evolving cyber technologies and vulnerabilities that expose businesses to attacks like privacy and identity theft.The rapid development of Cloud-based technologies, HTML5, Semantic Web, and various new security frameworks created out of "Big Data's" embryonic leftover need for incredibly powerful protection measures underline the necessity for robust detection and preventive methods against deadly CSRF assaults.

A systematic approach is employed to investigate CSRF dangers by putting out a novel, distinctive set of algorithms that employ intelligent assumptions to recognise and thwart CSRF. This paper elaborates on the design details of a CSRF Detection Model (CDM), its implementation, and experimentation outcomes in order to recognise, predict, and provide countermeasures for CSRF attacks on contemporary Web Applications and Web Services environments. Additionally, suggestions based on the CDM are given for customers and vendors of cyber security products and services.During an authorised browser session, a web application may be vulnerable to a Cross-Site Request Forgery (CSRF) attack without the user's knowledge. CSRF attacks specifically target requests that change the status of the system, such as those that transfer money or change email addresses, etc. If the victim is an administrator account, CSRF may compromise the entire web application. The Sleeping Giant, often known as CSRF, was one of the top five web vulnerabilities just four years ago. However, at least 270 CSRF attack instances had been recorded as of 2016. There hasn't been much development in terms of new CSRF solutions since the problem initially became apparent in 2010; nevertheless, this is changing. The vulnerabilities known as Cross-Site Scripting (XSS) and Cross-Site Reference Forgery (CSRF) have lately received a lot of attention.An XSS attack, which occurs when an attacker puts malicious code (often JavaScript), including a CSRF attack code, into a website with the goal of assaulting users of the website, such as websites that accept posting comments, is one of the top 3 current cyber security issues. The Open Online Application Security Project (OWASP), an open web community focused to tackling cyber security issues, lists this vulnerability as one of the top eight worldwide. Amazingly, CSRF attacks are simple to build and exploit but difficult to spot and protect against.

Cross Site Scripting, which is not the same as CSRF, turned up 117 articles in the ACM Digital Library, whereas CSRF turned up just four papers.When "XSS" was searched for on Safari Publications Online, a collection of more than 5000 technological publications, it turned up in 96 volumes as opposed to just 13 for "CSRF OR XSRF," which featured in more than 5000 books. Few CSRF solutions are being developed and deployed. Even Nevertheless, all the elements for extensive, very successful CSRF assaults are now in place , despite the fact that the current solutions are not yet widely used. The primary impetus for this study is the interaction between the ongoing CSRF assaults that are now occurring and the lack of proper security.

[2] LITERATURE SURVEY

The Web serves as the primary entry point for online data and applications. In order to deliver the greatest user experience, it includes a significant amount of dynamic contents made by other parties, making it extremely complex and varied. Because of this heterogeneity, it is very challenging to enforce security in a way that is effective because putting in place new security measures frequently prevents existing websites from operating as intended or has a negative impact on the user experience, both of which are generally viewed as unacceptable given the huge number of Web users. However, the relentless pursuit of usability and backward compatibility had a subtly detrimental effect on web security research: developers of new defensive mechanisms have been very cautious and the vast majority of their proposals consist of very local patches against very specific vulnerabilities.The growth of multiple threat models, some of which have rather divergent underlying assumptions against which many suggestions have been evaluated, is proof that this piecemeal evolution hindered a thorough understanding of many nuanced vulnerabilities and problems. It is easy to become confused amid the number of solutions that have been proposed and practically difficult to appreciate the relative benefits and drawbacks of each unique suggestion without a thorough comprehension of the current literature. In this research, we take on the challenging goal of undertaking a thorough description of a massive collection of widespread assaults aimed against the current Web and the corresponding security solutions thus far presented.We focus on attacks that target web sessions, i.e., assaults that target law-abiding web browser users who are connected to a reliable online service using authentication. By interfering with elements like dynamic content, client-side storage, or cross-domain connections to disrupt browser activity and/or network connectivity, these attacks take use of the Web's inherent complexity. We came to this conclusion because attacks on web sessions constitute a sizable subgroup of serious online security incidents and because multiple viable remedies, operating at different levels, have been proposed to stop these attacks.

We investigate typical online session assaults and categorise them according to I the attacker model and (ii) the security properties they break. This initial classification aids in identifying the planned security precautions of an online session can be violated and how by a particular attack. Then, we look at the defences and security mechanisms in place to prevent or decrease the different dangers, and we evaluate each plan based on the security guarantees it provides. When security can only be guaranteed under particular circumstances, we make these assumptions transparent. We evaluate each security system's compatibility, usability, ease of setup, and efficacy.These are essential criteria to use when assessing a solution's viability, and they allow us to establish if a certain solution is likely to be widely accepted on the Web as it stands. Additionally, we provide a summary of the many theories put out in the literature that are intended to provide solid defences against a variety of assaults.

Cross-Site Request Forgery (CSRF) attacks are a severe threat to internet applications. In this study, we focus on CSRF attacks that aim at websites' identity management and authentication systems. All of these will be known as Authentication CSRF (Auth-CSRF in short). Auth-CSRF attacks published in the literature were compiled, their underlying strategies were examined, and seven security testing methodologies that may help a human tester detect Auth-CSRF vulnerabilities were identified. In order to assess the effectiveness of our testing procedures and ascertain the incidence of Auth-CSRF, we conducted an experimental analysis employing 300 websites from three different rank ranges of the Alexa global top 1500.The results of our testing raise alarms: Out of the 300 websites we analysed, 90 of the 133 suitable for conducting them had at least one flaw that permitted Auth-CSRF (i.e. 68 percent). We developed our testing methods further, made them better with the knowledge we learned from our research, and released them (namely, CSRF-checker) as an extension to the free penetration testing application OWASP ZAP. Using CSRFchecker, we assessed 132 more websites (again, from the Alexa global top 1500) and discovered 95 that were vulnerable (i.e. 72 percent). We found serious issues with the websites of eBay, Microsoft, Google, and other businesses. Finally, we properly disclosed our findings to the impacted suppliers.

Online applications are inspected for security issues by automated tools referred to as black-box web application vulnerability scanners. To evaluate the state of the art, we had access to eight premier instruments and researched the following subjects: I consider the type of vulnerabilities these scanners test, II their potency against the target vulnerabilities, and III the relationship between the target flaws and vulnerabilities found in the wild. In order to conduct our analysis, we used both obsolete versions of well-known online applications with known faults and a custom web application that was susceptible to known and anticipated issues. Our results show both the strengths and promise of automated technology as a whole, as well as certain limitations.Particularly, there aren't many tools available right now that can find "stored" XSS and SQL Injection (SQLI) vulnerabilities. We don't provide comparative data or make any recommendations for the purchase of certain instruments because our goal is to assess the potential for new research, not to evaluate specific vendors.

Black-box web vulnerability scanners are a class of tools that may be used to detect security holes in online applications. These tools are commonly marketed as "point-and-click pentesting" tools that evaluate the security of web applications automatically and with little to no human intervention. Due to the fact that they access online applications in the same way that users do, these tools have the advantage of being independent of the exact technology used to develop the web application. These tools must be able to access the program's multiple components, even if they are usually hidden behind forms, JavaScript-generated links, and Flash programmes. This article evaluates eleven open-source and commercial black-box internet vulnerability scanners.The examination contains a wide range of vulnerabilities that pose different crawling challenges for the tools. these tests' incorporation into a genuine web application. The results of the examination show that several vulnerability classes are entirely disregarded by these tools, demanding more research to improve automated detection of these problems. As important and challenging to the total capacity to identify the weaknesses as the weakness detection technique itself, crawling is a chore.

Cross-Site Request Forgery (CSRF), one of the first and simplest Web attacks, is still effective on many websites and can have major consequences, including money losses and account takeovers. Unfortunately, to find CSRF vulnerabilities, the recommended methods and tools either need manual review by human experts or assume that the web application's source code exists. In this paper, we provide Mitch, the first machine learning method for identifying black-box CSRF vulnerabilities. Mitch's core functionality consists of an automated analyser of HTTP requests that are security-sensitive or that require CSRF protection.We trained the detector using supervised learning methods on a dataset of 5,828 HTTP requests collected from well-known websites. We provide other security researchers with access to this dataset. With the help of our method, which performs better than the detection criteria currently suggested in the literature, we were able to discover 35 new CSRF vulnerabilities on 20 significant websites. On production software that had already been examined by a cutting-edge technique, we also discovered 3 previously unidentified CSRF vulnerabilities.

3. ARCHITECTURE

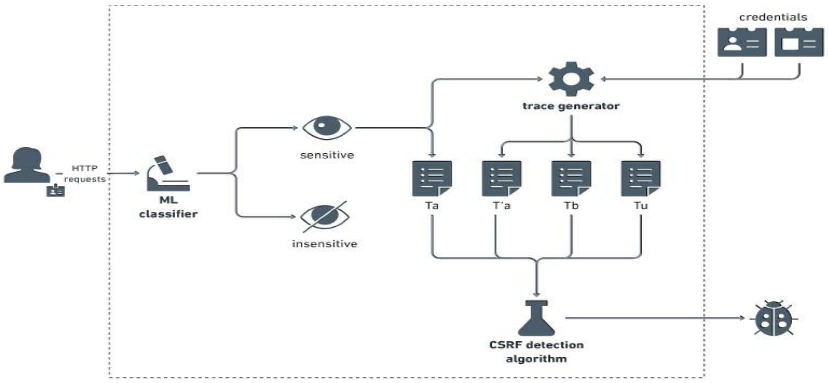


Fig. 1 Architecture

[4] IMPLEMENTATION

4.1 Modules Description

i) User: The user can initially register. For further chats, he required a functional user email and phone following enrollment. The administrator can activate the user after registration. Once the admin has activated the user, they may log onto our system. Data preprocessing is possible. Running website name must be first. By using that webpage, the user may test the csrfs. With the help of the bolt tool, the user is able to extract the related names for all developed algorithms and csrfs. The results will be saved as JSON files. Later, the user could receive the Mitch dataset results. The Mitchell dataset was also used to assess the POST and GET methods. The browser will display the results.

ii) Admin: Admin can sign in using his login information. After signing in, he may make the users active. The only programmes that let the enabled user log in are ours. The administrator can define the training and testing data for the Mitch Dataset project. The user looks through all associated urls that the csrf token administrator may view. The administrator may also look at the GET method-related data and POST method-accomplished data for the dataset.

iii) False Positives and False Negatives: It produces a false positive when Mitch presents a suspected CSRF that cannot really be exploited. Although manual testing is a difficult and time-consuming task, it is a very simple technique to identify this. It is typically hard to accurately verify whether Mitch provides a false negative since doing so would need knowing about every CSRF vulnerability available on the examined websites. We may estimate this critical component by keeping track of all the sensitive requests that Mitch's ML classifier returned and focusing our manual testing on those circumstances. This is a sensible choice to make the study tractable because we already showed that the classifier functions effectively using conventional validity measures.

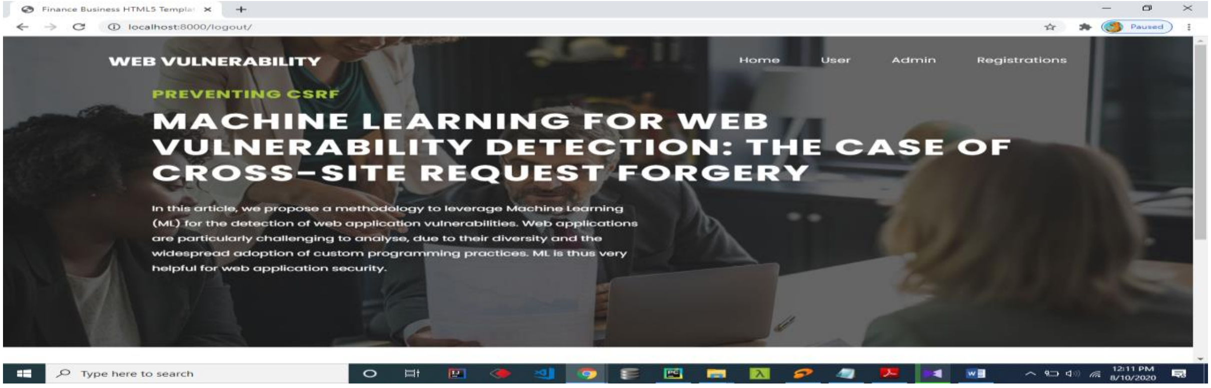
iv) Machine Learning Classifier: A dataset of roughly 6000 HTTP requests from live websites that was obtained and analysed by two human experts was used to create the machine learning classifier that Mitch used. The feature space X of the classifier has 49 dimensions, each of which captures a different aspect of HTTP requests. These fall under the following categories.

the following group of numerical attributes: The total number of parameters; the number of request parameters tied to boolean values; the number of request parameters linked to identifiers, such as hexadecimal strings, whose usage was experimentally discovered to be frequent in our dataset;

and the number of request parameters bound to identifiers; Number of Blobs (num of Blobs): Any string that is not an identifier that is associated to a Blob;

Requirement Length, or reqLen, is the total character of characters in the request, including parameter names and values.

4.2 Screenshots

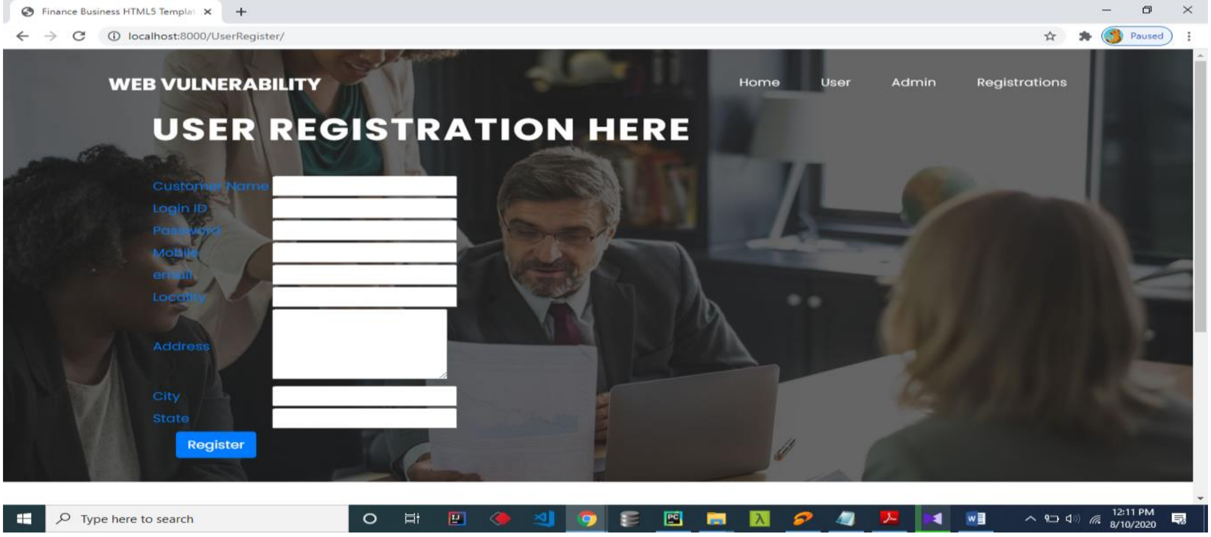


PREVENTING CSRF

MACHINE LEARNING FOR WEB VULNERABILITY DETECTION: THE CASE OF CROSS-SITE REQUEST FORGERY

In this article, we propose a methodology to leverage Machine Learning (ML) for the detection of web application vulnerabilities. Web applications are particularly challenging to analyse, due to their diversity and the widespread adoption of custom programming practices. ML is thus very helpful for web application security.

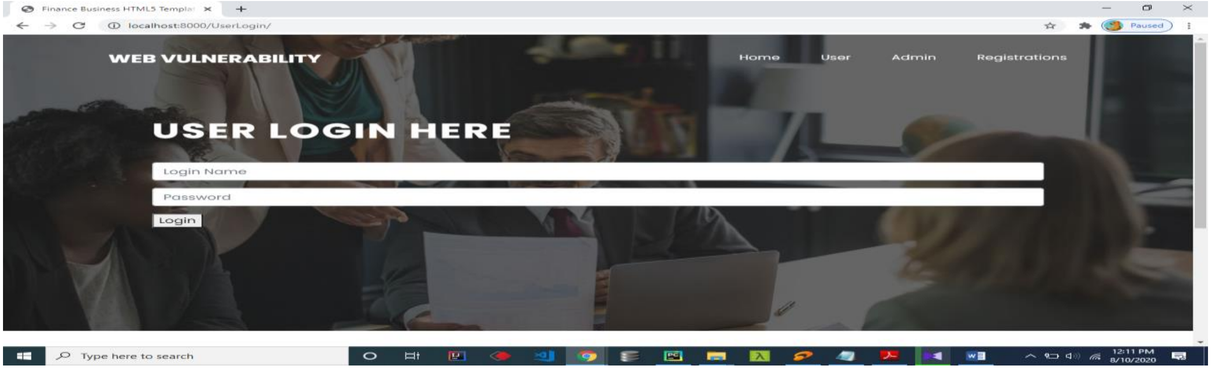
Fig. 2 Home page



USER REGISTRATION HERE

Customer Name
Login ID
Password
Mobile
email
Locality
Address
City
State

Fig. 3 User Registration Form

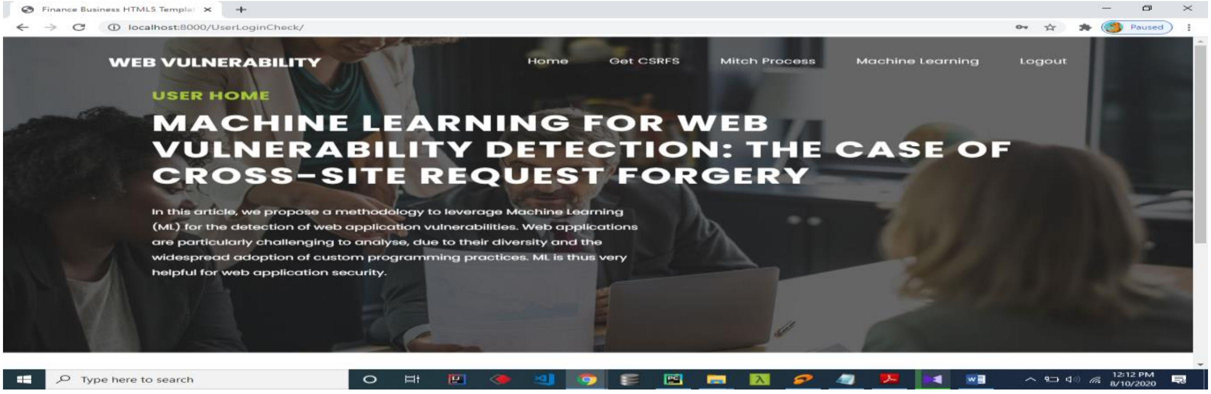


USER LOGIN HERE

Login Name
Password

Fig. 4 User Login Form

:

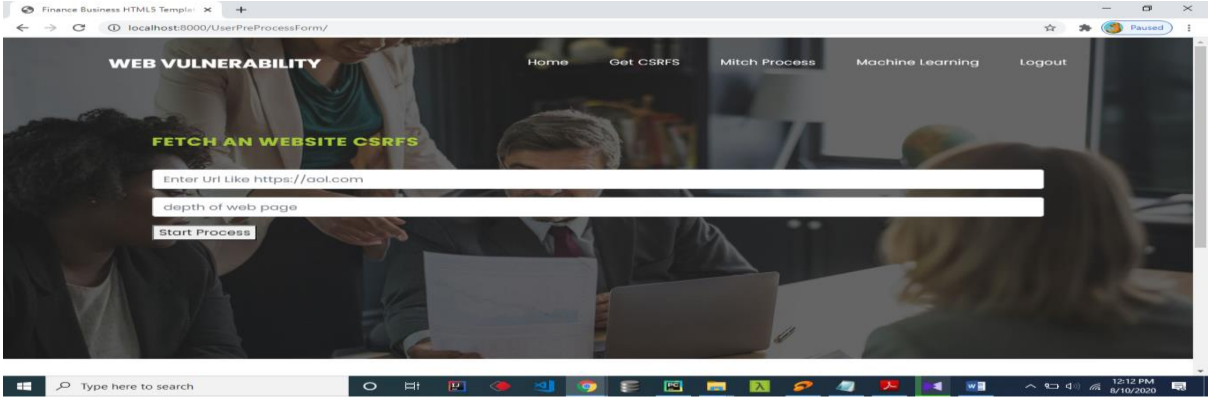


USER HOME

MACHINE LEARNING FOR WEB VULNERABILITY DETECTION: THE CASE OF CROSS-SITE REQUEST FORGERY

In this article, we propose a methodology to leverage Machine Learning (ML) for the detection of web application vulnerabilities. Web applications are particularly challenging to analyse, due to their diversity and the widespread adoption of custom programming practices. ML is thus very helpful for web application security.

Fig. 5 User Home

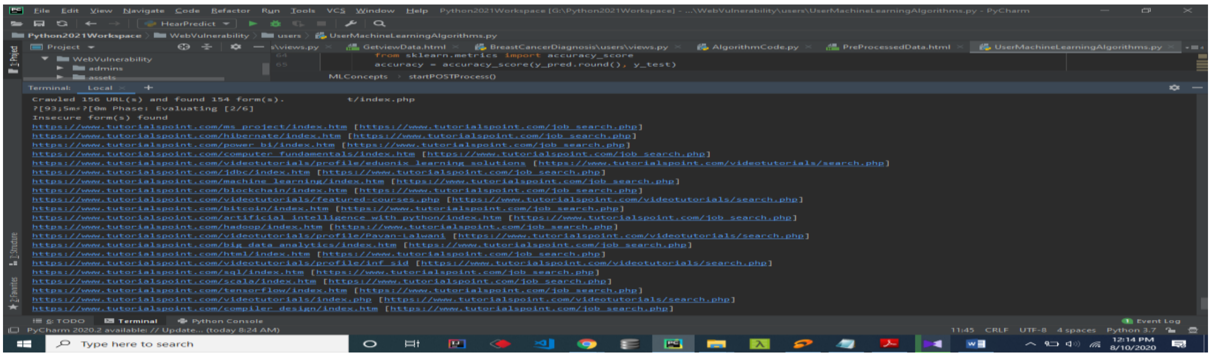


FETCH AN WEBSITE CSRFS

Enter Url Like https://aol.com

depth of web page

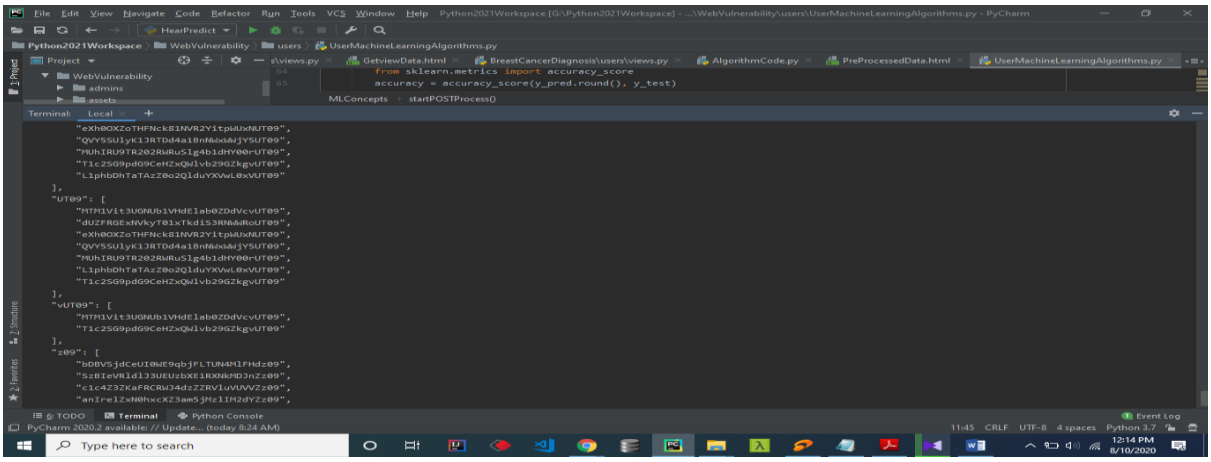
Fig. 6 Getting website CSRFS



```
        from sklearn.metrics import accuracy_score
        accuracy = accuracy_score(y_pred.round(), y_test)
```

```
Crawled 156 URL(s) and found 154 form(s).          t/index.php
?[93;5m•?[0m Phase: Evaluating [2/6]
Insecure form(s) found
https://www.tutorialspoint.com/ms_project/index.htm [https://www.tutorialspoint.com/job_search.php]
https://www.tutorialspoint.com/hibernate/index.htm [https://www.tutorialspoint.com/job_search.php]
https://www.tutorialspoint.com/power_bi/index.htm [https://www.tutorialspoint.com/job_search.php]
https://www.tutorialspoint.com/computer_fundamentals/index.htm [https://www.tutorialspoint.com/job_search.php]
https://www.tutorialspoint.com/videotutorials/profile/eduonix_learning_solutions [https://www.tutorialspoint.com/videotutorials/search.php]
https://www.tutorialspoint.com/jdbc/index.htm [https://www.tutorialspoint.com/job_search.php]
https://www.tutorialspoint.com/machine_learning/index.htm [https://www.tutorialspoint.com/job_search.php]
https://www.tutorialspoint.com/blockchain/index.htm [https://www.tutorialspoint.com/job_search.php]
https://www.tutorialspoint.com/videotutorials/featured-courses.php [https://www.tutorialspoint.com/videotutorials/search.php]
https://www.tutorialspoint.com/bitcoin/index.htm [https://www.tutorialspoint.com/job_search.php]
https://www.tutorialspoint.com/artificial_intelligence_with_python/index.htm [https://www.tutorialspoint.com/job_search.php]
https://www.tutorialspoint.com/hadoop/index.htm [https://www.tutorialspoint.com/job_search.php]
https://www.tutorialspoint.com/videotutorials/profile/Pavan-Lalwani [https://www.tutorialspoint.com/videotutorials/search.php]
https://www.tutorialspoint.com/big_data_analytics/index.htm [https://www.tutorialspoint.com/job_search.php]
https://www.tutorialspoint.com/html/index.htm [https://www.tutorialspoint.com/job_search.php]
https://www.tutorialspoint.com/videotutorials/profile/inf_sid [https://www.tutorialspoint.com/videotutorials/search.php]
https://www.tutorialspoint.com/sql/index.htm [https://www.tutorialspoint.com/job_search.php]
https://www.tutorialspoint.com/scala/index.htm [https://www.tutorialspoint.com/job_search.php]
https://www.tutorialspoint.com/tensorflow/index.htm [https://www.tutorialspoint.com/job_search.php]
https://www.tutorialspoint.com/videotutorials/index.php [https://www.tutorialspoint.com/videotutorials/search.php]
https://www.tutorialspoint.com/compiler_design/index.htm [https://www.tutorialspoint.com/job_search.php]
```

Fig. 7 Scanning URLS

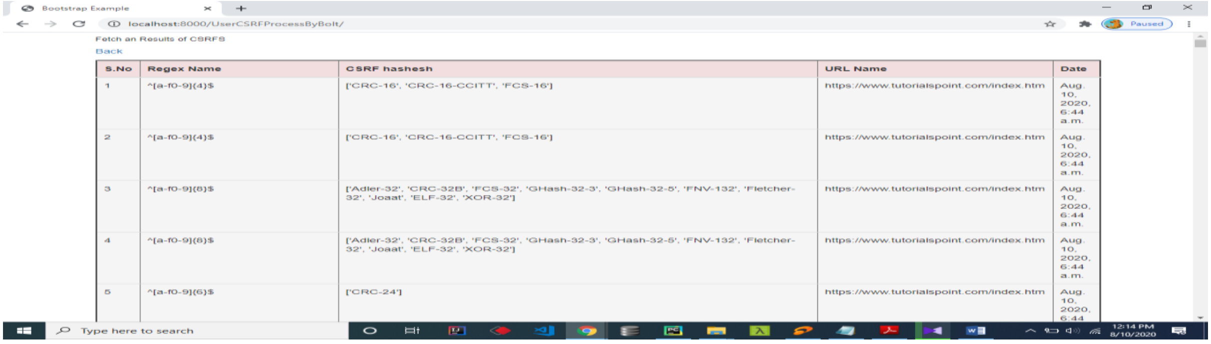


File Edit View Navigate Code Refactor Run Tools VCS Window Help Python2021Workspace [G:\Python2021Workspace] - ...\WebVulnerability\users\UserMachineLearningAlgorithms.py - PyCharm
HearPredict
Python2021Workspace › WebVulnerability › users › UserMachineLearningAlgorithms.py
Project
WebVulnerability
admins
assets
s\views.py
GetviewData.html
BreastCancerDiagnosis\users\views.py
AlgorithmCode.py
PreProcessedData.html
UserMachineLearningAlgorithms.py
MLConcepts › startPOSTProcess()
Terminal: Local
6: TODO
Terminal
Python Console
Event Log
PyCharm 2020.2 available: // Update... (today 8:24 AM)
11:45 CRLF UTF-8 4 spaces Python 3.7
Type here to search
12:14 PM 8/10/2020

```
from sklearn.metrics import accuracy_score
accuracy = accuracy_score(y_pred.round(), y_test)
```

```
        "eXh0OXZoTHFNck81NVR2YitpWUxNUT09",
        "QVY5SUlyK1JRTDd4a1BnNWxWWjY5UT09",
        "MUhIRU9TR202RWRuSlg4b1dHY00rUT09",
        "T1c2SG9pdG9CeHZxQWlvb29GZkgvUT09",
        "L1phbDhTaTAzZ0o2QlduYXVwL0xVUT09"
    ],
    "UT09": [
        "MTM1Vit3UGNUb1VHdElab0ZDdVcvUT09",
        "dUZFRGExNVkyT01xTkdiS3RNWWRoUT09",
        "eXh0OXZoTHFNck81NVR2YitpWUxNUT09",
        "QVY5SUlyK1JRTDd4a1BnNWxWWjY5UT09",
        "MUhIRU9TR202RWRuSlg4b1dHY00rUT09",
        "L1phbDhTaTAzZ0o2QlduYXVwL0xVUT09",
        "T1c2SG9pdG9CeHZxQWlvb29GZkgvUT09"
    ],
    "vUT09": [
        "MTM1Vit3UGNUb1VHdElab0ZDdVcvUT09",
        "T1c2SG9pdG9CeHZxQWlvb29GZkgvUT09"
    ],
    "z09": [
        "bDBVSjdCeUI0WE9qbjFLTUN4MlFHdz09",
        "SzBIeVRldlJ3UEUzbXE1RXNkMDJnZz09",
        "c1c4Z3ZKaFRCRWJ4dzZZRVluVUVVZz09",
        "anIrelZxN0hxcXZ3am5jMzlIM2dYZz09",
```

Fig. 8 CSRF Token

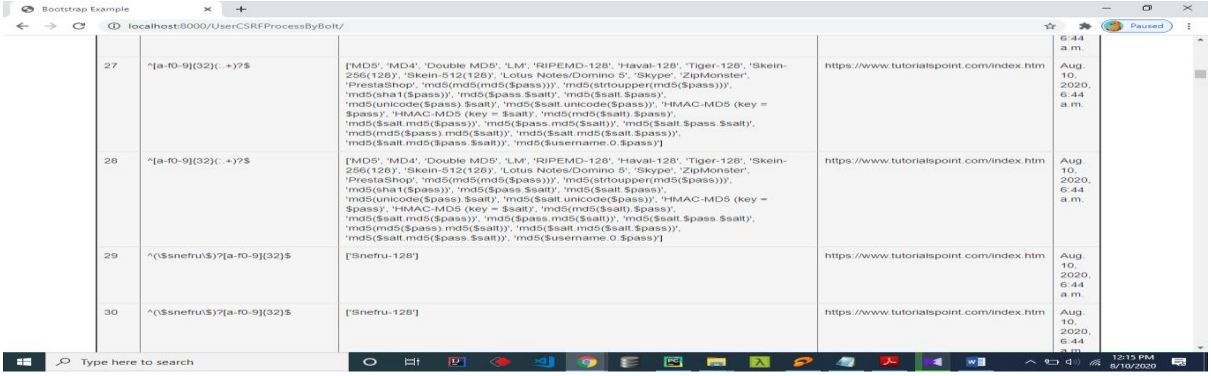


Fetch an Results of CSRFS

Back

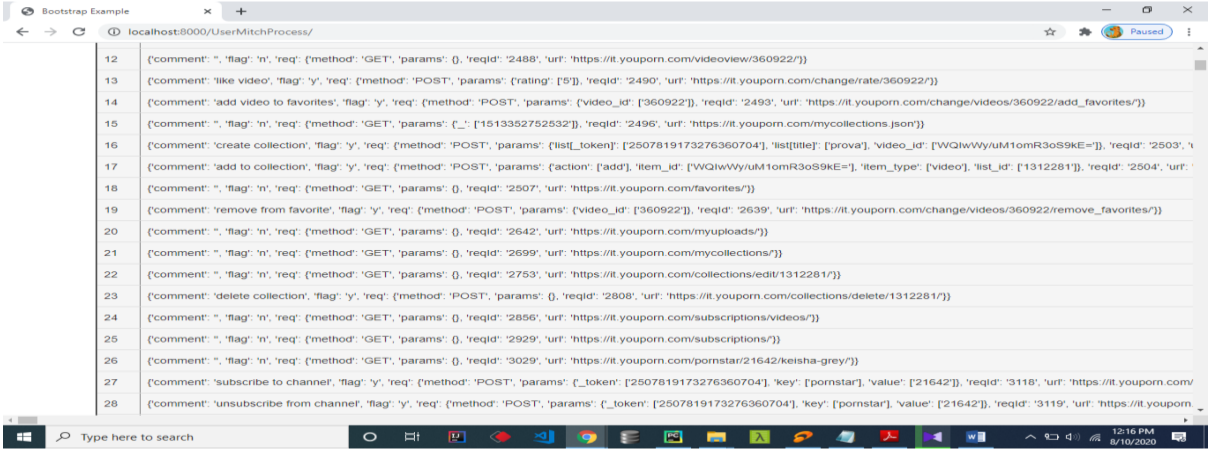
S.No	Regex Name	CSRF hashesh	URL Name	Date
1	^[a-f0-9]{4}$	['CRC-16', 'CRC-16-CCITT', 'FCS-16']	https://www.tutorialspoint.com/index.htm	Aug. 10, 2020, 6:44 a.m.
2	^[a-f0-9]{4}$	['CRC-16', 'CRC-16-CCITT', 'FCS-16']	https://www.tutorialspoint.com/index.htm	Aug. 10, 2020, 6:44 a.m.
3	^[a-f0-9]{8}$	['Adler-32', 'CRC-32B', 'FCS-32', 'GHash-32-3', 'GHash-32-5', 'FNV-132', 'Fletcher-32', 'Joaat', 'ELF-32', 'XOR-32']	https://www.tutorialspoint.com/index.htm	Aug. 10, 2020, 6:44 a.m.
4	^[a-f0-9]{8}$	['Adler-32', 'CRC-32B', 'FCS-32', 'GHash-32-3', 'GHash-32-5', 'FNV-132', 'Fletcher-32', 'Joaat', 'ELF-32', 'XOR-32']	https://www.tutorialspoint.com/index.htm	Aug. 10, 2020, 6:44 a.m.
5	^[a-f0-9]{6}$	['CRC-24']	https://www.tutorialspoint.com/index.htm	Aug. 10, 2020, 6:44

Fig. 9 Given Website CSRF Results



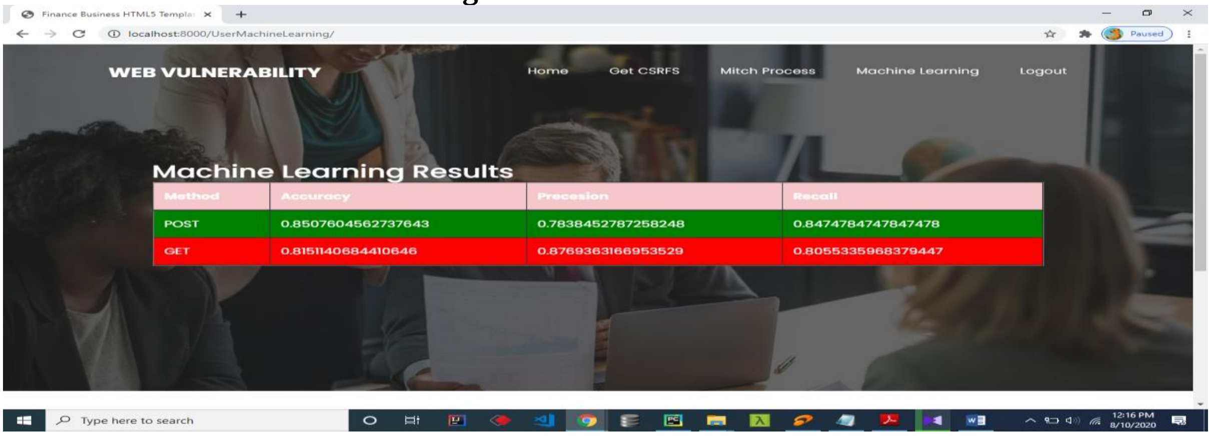
				6:44 a.m.
27	^[a-f0-9]{32}(:.+)?$	['MD5', 'MD4', 'Double MD5', 'LM', 'RIPEMD-128', 'Haval-128', 'Tiger-128', 'Skein-256(128)', 'Skein-512(128)', 'Lotus Notes/Domino 5', 'Skype', 'ZipMonster', 'PrestaShop', 'md5(md5(md5($pass)))', 'md5(strtoupper(md5($pass)))', 'md5(sha1($pass))', 'md5($pass.$salt)', 'md5($salt.$pass)', 'md5(unicode($pass).$salt)', 'md5($salt.unicode($pass))', 'HMAC-MD5 (key = $pass)', 'HMAC-MD5 (key = $salt)', 'md5(md5($salt).$pass)', 'md5($salt.md5($pass))', 'md5($pass.md5($salt))', 'md5($salt.$pass.$salt)', 'md5(md5($pass).md5($salt))', 'md5($salt.md5($salt.$pass))', 'md5($salt.md5($pass.$salt))', 'md5($username.0.$pass)']	https://www.tutorialspoint.com/index.htm	Aug. 10, 2020, 6:44 a.m.
28	^[a-f0-9]{32}(:.+)?$	['MD5', 'MD4', 'Double MD5', 'LM', 'RIPEMD-128', 'Haval-128', 'Tiger-128', 'Skein-256(128)', 'Skein-512(128)', 'Lotus Notes/Domino 5', 'Skype', 'ZipMonster', 'PrestaShop', 'md5(md5(md5($pass)))', 'md5(strtoupper(md5($pass)))', 'md5(sha1($pass))', 'md5($pass.$salt)', 'md5($salt.$pass)', 'md5(unicode($pass).$salt)', 'md5($salt.unicode($pass))', 'HMAC-MD5 (key = $pass)', 'HMAC-MD5 (key = $salt)', 'md5(md5($salt).$pass)', 'md5($salt.md5($pass))', 'md5($pass.md5($salt))', 'md5($salt.$pass.$salt)', 'md5(md5($pass).md5($salt))', 'md5($salt.md5($salt.$pass))', 'md5($salt.md5($pass.$salt))', 'md5($username.0.$pass)']	https://www.tutorialspoint.com/index.htm	Aug. 10, 2020, 6:44 a.m.
29	^(\$snefru\$)?[a-f0-9]{32}$	['Snefru-128']	https://www.tutorialspoint.com/index.htm	Aug. 10, 2020, 6:44 a.m.
30	^(\$snefru\$)?[a-f0-9]{32}$	['Snefru-128']	https://www.tutorialspoint.com/index.htm	Aug. 10, 2020, 6:44

Fig. 10 MD5 Token



```
12  {'comment': '', 'flag': 'n', 'req': {'method': 'GET', 'params': {}, 'reqId': '2488', 'url': 'https://it.youporn.com/videoview/360922/'}}
13  {'comment': 'like video', 'flag': 'y', 'req': {'method': 'POST', 'params': {'rating': ['5']}, 'reqId': '2490', 'url': 'https://it.youporn.com/change/rate/360922/'}}
14  {'comment': 'add video to favorites', 'flag': 'y', 'req': {'method': 'POST', 'params': {'video_id': ['360922']}, 'reqId': '2493', 'url': 'https://it.youporn.com/change/videos/360922/add_favorites/'}}
15  {'comment': '', 'flag': 'n', 'req': {'method': 'GET', 'params': {'_': ['1513352752532']}, 'reqId': '2496', 'url': 'https://it.youporn.com/mycollections.json'}}
16  {'comment': 'create collection', 'flag': 'y', 'req': {'method': 'POST', 'params': {'list[_token]': ['2507819173276360704'], 'list[title]': ['prova'], 'video_id': ['WQIwWy/uM1omR3oS9kE=']}, 'reqId': '2503', 'u
17  {'comment': 'add to collection', 'flag': 'y', 'req': {'method': 'POST', 'params': {'action': ['add'], 'item_id': ['WQIwWy/uM1omR3oS9kE='], 'item_type': ['video'], 'list_id': ['1312281']}, 'reqId': '2504', 'url': '
18  {'comment': '', 'flag': 'n', 'req': {'method': 'GET', 'params': {}, 'reqId': '2507', 'url': 'https://it.youporn.com/favorites/'}}
19  {'comment': 'remove from favorite', 'flag': 'y', 'req': {'method': 'POST', 'params': {'video_id': ['360922']}, 'reqId': '2639', 'url': 'https://it.youporn.com/change/videos/360922/remove_favorites/'}}
20  {'comment': '', 'flag': 'n', 'req': {'method': 'GET', 'params': {}, 'reqId': '2642', 'url': 'https://it.youporn.com/myuploads/'}}
21  {'comment': '', 'flag': 'n', 'req': {'method': 'GET', 'params': {}, 'reqId': '2699', 'url': 'https://it.youporn.com/mycollections/'}}
22  {'comment': '', 'flag': 'n', 'req': {'method': 'GET', 'params': {}, 'reqId': '2753', 'url': 'https://it.youporn.com/collections/edit/1312281/'}}
23  {'comment': 'delete collection', 'flag': 'y', 'req': {'method': 'POST', 'params': {}, 'reqId': '2808', 'url': 'https://it.youporn.com/collections/delete/1312281/'}}
24  {'comment': '', 'flag': 'n', 'req': {'method': 'GET', 'params': {}, 'reqId': '2856', 'url': 'https://it.youporn.com/subscriptions/videos/'}}
25  {'comment': '', 'flag': 'n', 'req': {'method': 'GET', 'params': {}, 'reqId': '2929', 'url': 'https://it.youporn.com/subscriptions/'}}
26  {'comment': '', 'flag': 'n', 'req': {'method': 'GET', 'params': {}, 'reqId': '3029', 'url': 'https://it.youporn.com/pornstar/21642/keisha-grey/'}}
27  {'comment': 'subscribe to channel', 'flag': 'y', 'req': {'method': 'POST', 'params': {'_token': ['2507819173276360704'], 'key': ['pornstar'], 'value': ['21642']}, 'reqId': '3118', 'url': 'https://it.youporn.com/
28  {'comment': 'unsubscribe from channel', 'flag': 'y', 'req': {'method': 'POST', 'params': {'_token': ['2507819173276360704'], 'key': ['pornstar'], 'value': ['21642']}, 'reqId': '3119', 'url': 'https://it.youporn.
```

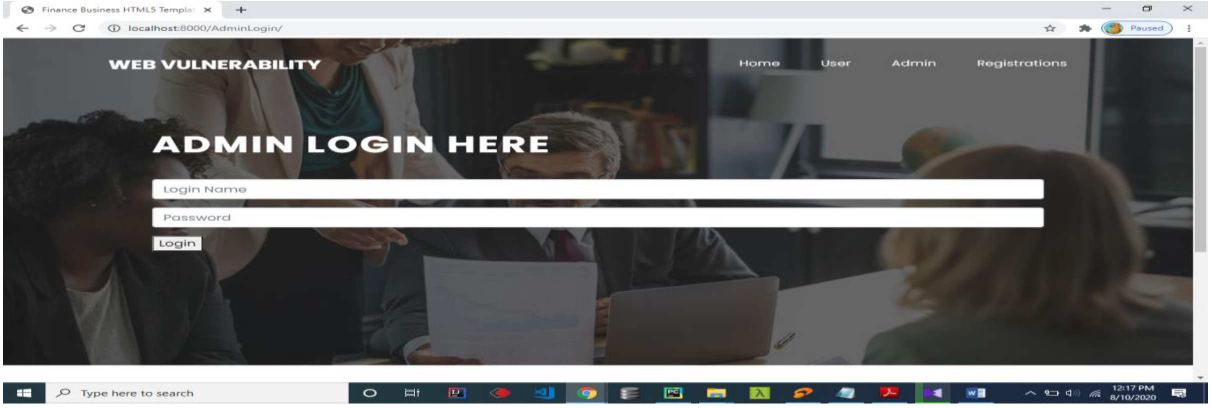
Fig. 11 Mitch Detected sites



Machine Learning Results

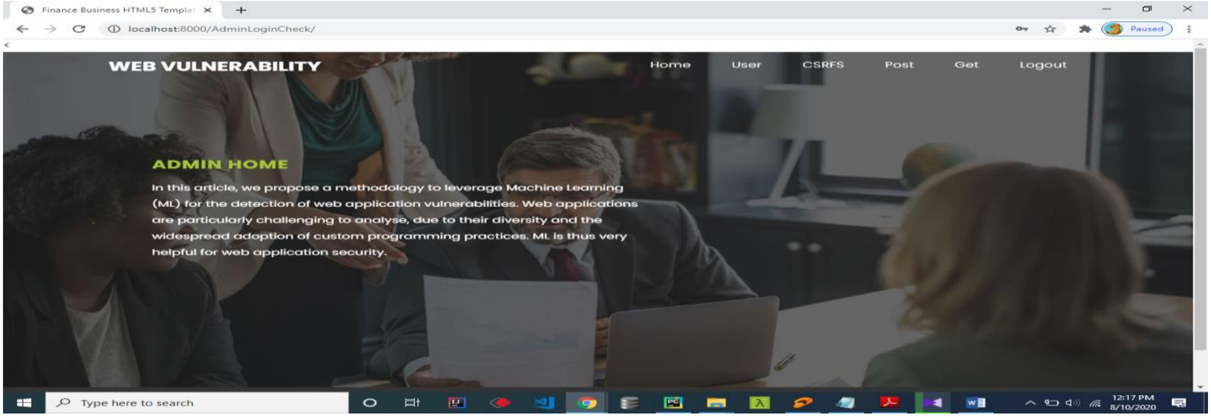
Method	Accuracy	Precesion	Recall
POST	0.8507604562737643	0.7838452787258248	0.8474784747847478
GET	0.8151140684410646	0.8769363166953529	0.8055335968379447

Fig. 12 Machine Learning Results



ADMIN LOGIN HERE

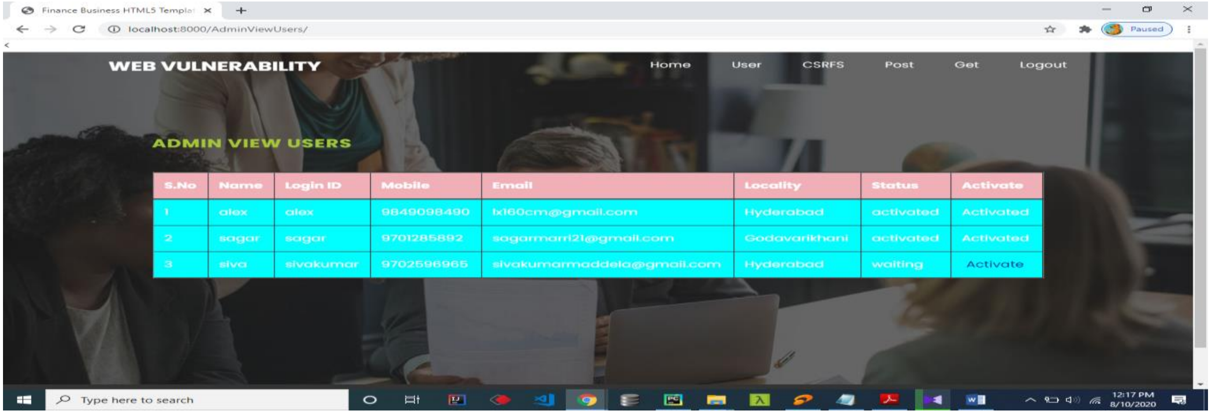
Fig. 13 Admin Login



ADMIN HOME

In this article, we propose a methodology to leverage Machine Learning (ML) for the detection of web application vulnerabilities. Web applications are particularly challenging to analyse, due to their diversity and the widespread adoption of custom programming practices. ML is thus very helpful for web application security.

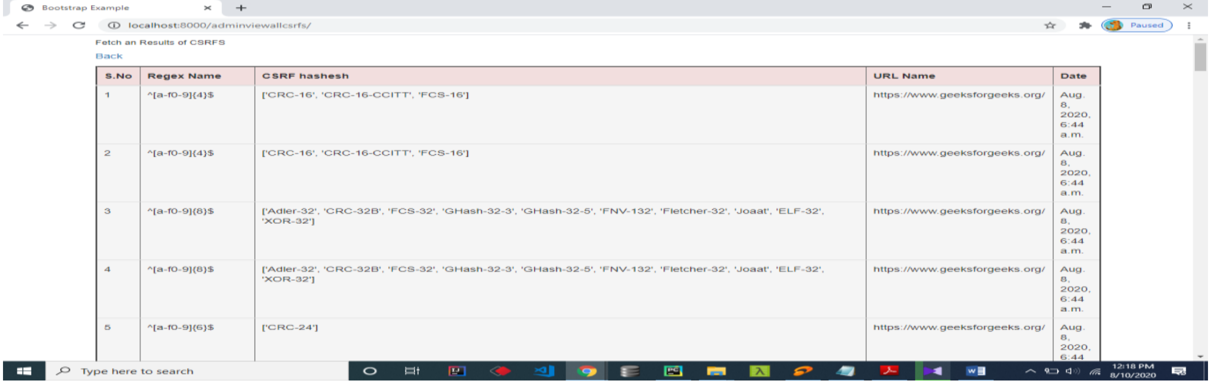
Fig. 14 Admin Home Page



ADMIN VIEW USERS

S.No	Name	Login ID	Mobile	Email	Locality	Status	Activate
1	alex	alex	9849098490	lx160cm@gmail.com	Hyderabad	activated	Activated
2	sagar	sagar	9701285892	sagarmarri21@gmail.com	Godavarikhani	activated	Activated
3	siva	sivakumar	9702596965	sivakumarmaddela@gmail.com	Hyderabad	waiting	Activate

Fig. 15 View Registered Users

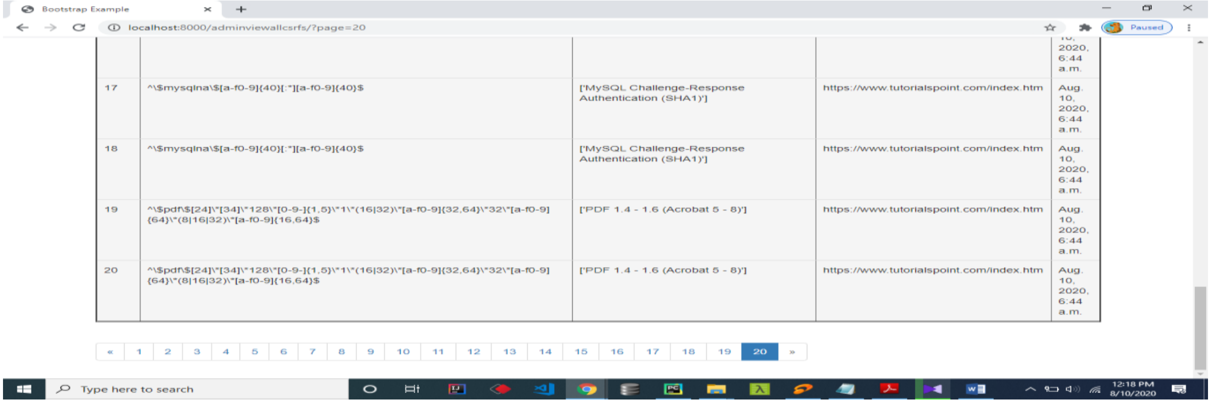


Fetch an Results of CSRFS

Back

S.No	Regex Name	CSRF hashesh	URL Name	Date
1	^[a-f0-9]{4}$	['CRC-16', 'CRC-16-CCITT', 'FCS-16']	https://www.geeksforgeeks.org/	Aug. 8, 2020, 6:44 a.m.
2	^[a-f0-9]{4}$	['CRC-16', 'CRC-16-CCITT', 'FCS-16']	https://www.geeksforgeeks.org/	Aug. 8, 2020, 6:44 a.m.
3	^[a-f0-9]{8}$	['Adler-32', 'CRC-32B', 'FCS-32', 'GHash-32-3', 'GHash-32-5', 'FNV-132', 'Fletcher-32', 'Joaat', 'ELF-32', 'XOR-32']	https://www.geeksforgeeks.org/	Aug. 8, 2020, 6:44 a.m.
4	^[a-f0-9]{8}$	['Adler-32', 'CRC-32B', 'FCS-32', 'GHash-32-3', 'GHash-32-5', 'FNV-132', 'Fletcher-32', 'Joaat', 'ELF-32', 'XOR-32']	https://www.geeksforgeeks.org/	Aug. 8, 2020, 6:44 a.m.
5	^[a-f0-9]{6}$	['CRC-24']	https://www.geeksforgeeks.org/	Aug. 8, 2020, 6:44

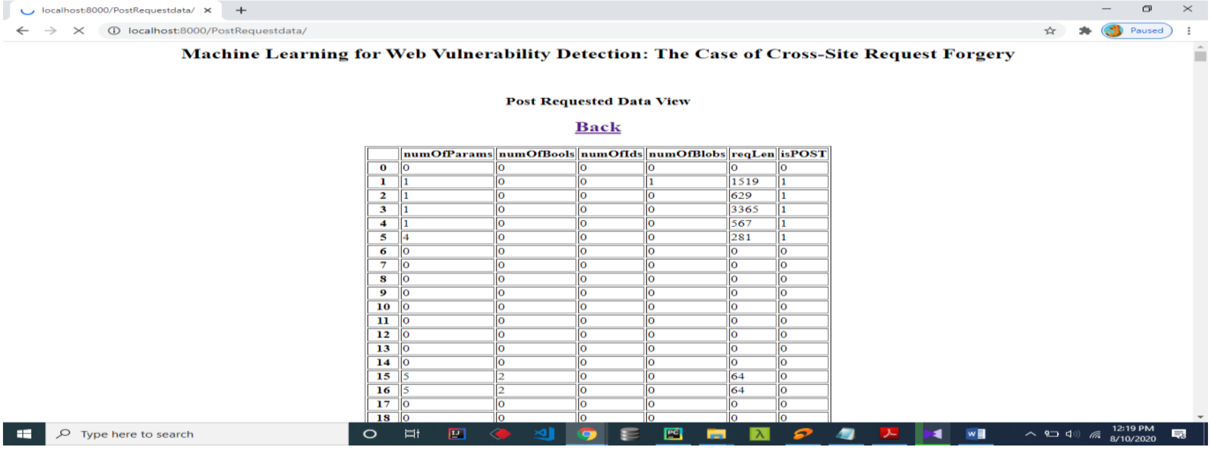
Fig. 16 Admin View All CSRFS



				10, 2020, 6:44 a.m.
17	^\$mysqlna\$[a-f0-9]{40}[:"][a-f0-9]{40}$	['MySQL Challenge-Response Authentication (SHA1)']	https://www.tutorialspoint.com/index.htm	Aug. 10, 2020, 6:44 a.m.
18	^\$mysqlna\$[a-f0-9]{40}[:"][a-f0-9]{40}$	['MySQL Challenge-Response Authentication (SHA1)']	https://www.tutorialspoint.com/index.htm	Aug. 10, 2020, 6:44 a.m.
19	^\$pdf\$[24]\"[34]\"128\"[0-9-]{1,5}\"1\"(16|32)\"[a-f0-9]{32,64}\"32\"[a-f0-9]{64}\"(8|16|32)\"[a-f0-9]{16,64}$	['PDF 1.4 - 1.6 (Acrobat 5 - 8)']	https://www.tutorialspoint.com/index.htm	Aug. 10, 2020, 6:44 a.m.
20	^\$pdf\$[24]\"[34]\"128\"[0-9-]{1,5}\"1\"(16|32)\"[a-f0-9]{32,64}\"32\"[a-f0-9]{64}\"(8|16|32)\"[a-f0-9]{16,64}$	['PDF 1.4 - 1.6 (Acrobat 5 - 8)']	https://www.tutorialspoint.com/index.htm	Aug. 10, 2020, 6:44 a.m.

« 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 »

Fig. 17 CSRFS


localhost:8000/PostRequestdata/
localhost:8000/PostRequestdata/
Paused
Type here to search
12:19 PM
8/10/2020

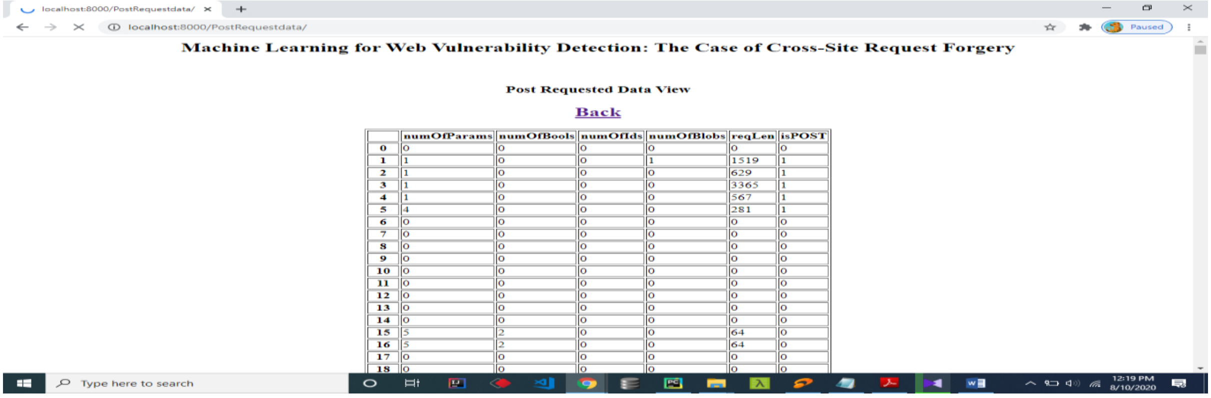
Machine Learning for Web Vulnerability Detection: The Case of Cross-Site Request Forgery

Post Requested Data View

Back

	numOfParams	numOfBools	numOfIds	numOfBlobs	reqLen	isPOST
0	0	0	0	0	0	0
1	1	0	0	1	1519	1
2	1	0	0	0	629	1
3	1	0	0	0	3365	1
4	1	0	0	0	567	1
5	4	0	0	0	281	1
6	0	0	0	0	0	0
7	0	0	0	0	0	0
8	0	0	0	0	0	0
9	0	0	0	0	0	0
10	0	0	0	0	0	0
11	0	0	0	0	0	0
12	0	0	0	0	0	0
13	0	0	0	0	0	0
14	0	0	0	0	0	0
15	5	2	0	0	64	0
16	5	2	0	0	64	0
17	0	0	0	0	0	0
18	0	0	0	0	0	0

Fig. 18 Post Data View


localhost:8000/PostRequestdata/
localhost:8000/PostRequestdata/
Paused
Type here to search
12:19 PM
8/10/2020

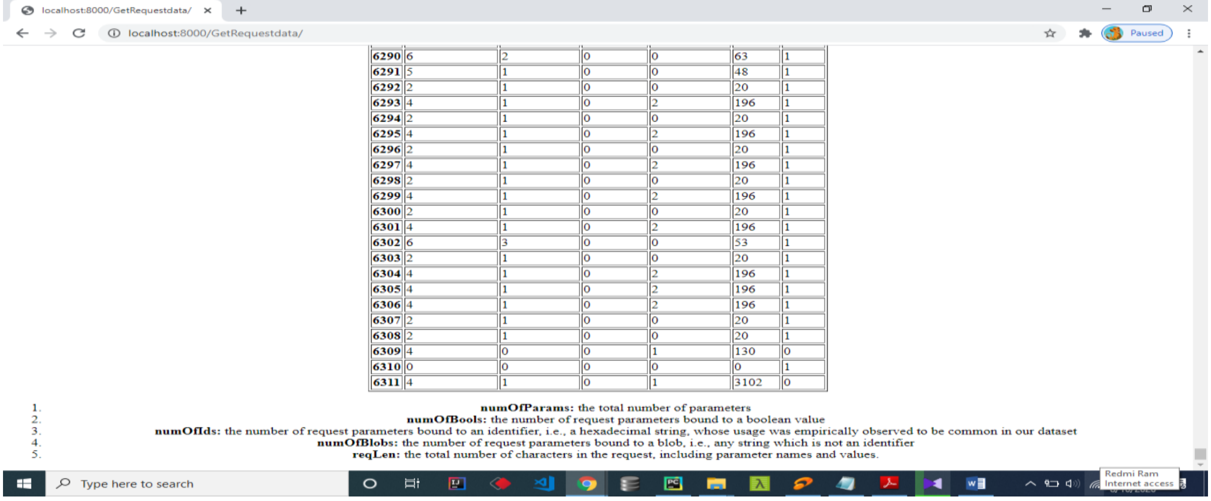
Machine Learning for Web Vulnerability Detection: The Case of Cross-Site Request Forgery

Post Requested Data View

Back

	numOfParams	numOfBools	numOfIds	numOfBlobs	reqLen	isPOST
0	0	0	0	0	0	0
1	1	0	0	1	1519	1
2	1	0	0	0	629	1
3	1	0	0	0	3365	1
4	1	0	0	0	567	1
5	4	0	0	0	281	1
6	0	0	0	0	0	0
7	0	0	0	0	0	0
8	0	0	0	0	0	0
9	0	0	0	0	0	0
10	0	0	0	0	0	0
11	0	0	0	0	0	0
12	0	0	0	0	0	0
13	0	0	0	0	0	0
14	0	0	0	0	0	0
15	5	2	0	0	64	0
16	5	2	0	0	64	0
17	0	0	0	0	0	0
18	0	0	0	0	0	0

Fig. 19 Get Data


localhost:8000/GetRequestdata/
localhost:8000/GetRequestdata/
Paused
Type here to search
Redmi Ram
Internet access

6290	6	2	0	0	63	1
6291	5	1	0	0	48	1
6292	2	1	0	0	20	1
6293	4	1	0	2	196	1
6294	2	1	0	0	20	1
6295	4	1	0	2	196	1
6296	2	1	0	0	20	1
6297	4	1	0	2	196	1
6298	2	1	0	0	20	1
6299	4	1	0	2	196	1
6300	2	1	0	0	20	1
6301	4	1	0	2	196	1
6302	6	3	0	0	53	1
6303	2	1	0	0	20	1
6304	4	1	0	2	196	1
6305	4	1	0	2	196	1
6306	4	1	0	2	196	1
6307	2	1	0	0	20	1
6308	2	1	0	0	20	1
6309	4	0	0	1	130	0
6310	0	0	0	0	0	1
6311	4	1	0	1	3102	0

1. **numOfParams:** the total number of parameters
2. **numOfBools:** the number of request parameters bound to a boolean value
3. **numOfIds:** the number of request parameters bound to an identifier, i.e., a hexadecimal string, whose usage was empirically observed to be common in our dataset
4. **numOfBlobs:** the number of request parameters bound to a blob, i.e., any string which is not an identifier
5. **reqLen:** the total number of characters in the request, including parameter names and values.

Fig. 20 Attribute Descriptions

[5] CONCLUSION

Web applications are particularly challenging to analyse due to their variety and substantial usage of custom development approaches. Because it may use manually labelled data to expose automatic analytical tools to human interpretation of the semantics of online applications, machine learning is therefore very helpful in a web setting. We validated this assertion by developing Mitch, the first machine learning (ML) solution for the blackbox detection of CSRF vulnerabilities, and by experimentally assessing its performance. We anticipate that other researchers will employ our techniques to identify various web application vulnerabilities. The most up-to-date and comprehensive explanation of CSRF attacks and a corresponding defence against them may be found in this paper. However, as the new knowledge gains wider acceptance, it will inevitably expand in the future. The following are a few subjects that may need more investigation in the future: It may be feasible to implement the Bayesian estimation by using the 1-99 (adjustable) threshold probability ratio of suspect CSRF page to safe pages as a future development of this work because anything below 1% of the threshold can either be random or have unexpected volatility. Browser-specific and generic CSRF protection programmes. Routine CSRF scanning in commercial anti-malware programmes.

REFERENCES

[1] Stefano Calzavara, Riccardo Focardi, Marco Squarcina, and Mauro Tempesta.Surviving the web: A journey into web session security.ACM Comput.Surv., 50(1):13:1–13:34, 2017.

[2] AvinashSudhodanan, Roberto Carbone, Luca Compagna, Nicolas Dolgin, Alessandro Armando, and Umberto Morelli. Large-scale analysis & detection of authentication cross-site request forgeries. In 2017 IEEE European Symposium on Security and Privacy, EuroS&P 2017, Paris, France, April 26-28, 2017, pages 350–365, 2017.

[3] Stefano Calzavara, AlviseRabitti, AlessioRagazzo, and Michele Bugliesi.Testing for integrity flaws in web sessions. In Computer Security - 24rd European Symposium on Research in Computer Security, ESORICS 2019, Luxembourg, Luxembourg, September 23-27, 2019, pages 606–624, 2019.

[4] OWASP. OWASP Testing Guide. https://www.owasp.org/index.php/ OWASP Testing Guide v4 Table of Contents, 2016.

[5] Jason Bau, ElieBursztein, Divij Gupta, and John C. Mitchell. State of the art: Automated black-box web application vulnerability testing. In 31st IEEE Symposium on Security and Privacy, S&P 2010, 16-19 May 2010, Berkeley/Oakland, California, USA, pages 332–345, 2010.

[6] Adam Doup´e, Marco Cova, and Giovanni Vigna. Why johnny can't pentest: An analysis of black-box web vulnerability scanners. In Detection of Intrusions and Malware, and Vulnerability Assessment, 7th International Conference, DIMVA 2010, Bonn, Germany, July 8-9, 2010.Proceedings, pages 111–131, 2010.

[7] Adam Barth, Collin Jackson, and John C. Mitchell. Robust defenses for cross-site request forgery. In Proceedings of the 2008 ACM Conference on Computer and Communications Security, CCS 2008, Alexandria, Virginia, USA, October 27-31, 2008, pages 75–88, 2008.

[8] MehryarMohri, AfshinRostamizadeh, and Ameet Talwalkar.Foundations of Machine Learning.The MIT Press, 2012.

[9] Michael W. Kattan, Dennis A. Adams, and Michael S. Parks.A comparison of machine learning with human judgment. Journal of Management Information Systems, 9(4):37–57, March 1993.

[10] D. A. Ferrucci. Introduction to "This is Watson". IBM Journal of Research and Development, 56(3):235–249, May 2012.

[11] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, IoannisAntonoglou, Veda Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, NalKalchbrenner, Ilya Sutskever, Timothy Lillicrap, Madeleine Leach, KorayKavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of Go with deep neural networks and tree search. Nature, 529(7587):484–489, Jan 2016.

[12] Michele Bugliesi, Stefano Calzavara, Riccardo Focardi, and Wilayat Khan.Cookiext: Patching the browser against session hijacking attacks. Journal of Computer Security, 23(4):509–537, 2015.

[13] GNR Prasad, SK Althaf Hussain Basha, Mallikharjuna Rao K M GnanaVardhan "A Review of Predictive And Descriptive Data Mining Techniques In Higher Education Domain, International Journal of Computer Engineering and Applications(IJCEA),Volume 13, Issue 6, January. 21, ISSN2321-3469.

[14] B Sasidhar, Sk Althaf Hussain Basha , " A Comparative Study of Educational Data Mining Methods Used to Forecast Student Success and Failures", International Journal Computer Science Information and Engineering Technologies (IJCSIET), International Conference 2014,ISSN:2277-4408,2014.

[15] Ch. Prakash, Sk Althaf Hussain Basha, D. Mounika, G. Maheetha, "An Approach for Multi Instance Clustering of Student Academic Performance in Education Domain", IIJDWM Journal, Volume 3,Issue 1,pp.1-9,Feb.2013,ISSN: 2249-7161

[16] Sd.Muneer , Sk Althaf Hussain Basha, A.Govardhan, V.Uday Kumar " Generate Eligible Students using Decision Trees-A Frame work for Employee Ability" International Journal of Advanced Computing(IJAC), Volume 4,Issue 2,2012,pp.68-76, ISSN:0975-7686.

[17] Mohd. Zaheer Ahmed , Sk Althaf Hussain Basha, A.Govardhan ,Y.R.Ramesh Kumar , "Predicting Student Academic Performance Using Temporal Association Mining" International Journal of Information Systems and Education (IJISE), Vol.2, No.1(2012),pp.21-41,ISSN: 2231- 1262.

[18] A.Govardhan , SK Althaf Hussain Basha, Y.R.Ramesh Kumar , Mohd. Zaheer Ahmed, "Study of Education Patterns Using Association Mining" International Journal Data Warehousing (IJDW), Vol.3 ,No.2,2011, pp. 53-64, ISSN: 0975-6124.

[19] SK Althaf Hussain Basha, A.Govardhan, "MICR: Multiple Instance Cluster Regression for Student Academic

Performance in Higher Education", International Journal of Computer Applications(IJCA), Volume 14– No.4,2011,pp.23-29, ISSN: 0975-8887 (Impact Factor : 0.8

[20] SK Althaf Hussain Basha, A. Govardhan "A Comparative Analysis of Prediction Techniques for Predicting Graduate Rate of University", European Journal of Scientific Research (EJSR) ,Vol.46 No.2,2010, pp.186-193, ISSN No:1450-216X . (Impact Factor0.783, Citations: 12)

[21] Sk. Althaf Hussain Basha, A.Govardhan "Rank Analysis Through Polyanalyst using Linear Regression" , International Journal of Computer Science and Network Security(IJCSNS), VOL.9 No.9,2009, pp. 290-293, ISSN: 1738-7906. (Impact Factor:2.512, Citations:7)

[22] T Naveen Kumar, SK Althaf Hussain Basha, V. Anand , DonapatiSrikanth, "Categorization of Academic Student Performance using Hybrid Techniques" International Conference on Advanced Computing Methodologies (ICACM-2013), Hyderabad, pp.325- 330,2013.

[23] Y. Vijayalata, Sk Althaf Hussain Basha, V. Anand ,Donapati Srikanth, " Study of Education patterns using Rare Association Mining-A case Study " , IEEE International Conference on Engineering for Humanity (ICEH-2013), Hyderabad, pp. 53-61,2013,ISSN: 978-93-82880- 53-0.

[24] Y Ramesh Kumar, Sk Althaf Hussain Basha, Y Vijayalata, " Predicting Student Academic Performance using Temporal Association Mining-A case Study on Educational Data " , IEEE International Conference on Engineering for Humanity (ICEH-2013), Hyderabad, pp. 21- 27,2013, ISSN:978-93-82880-53-0.

[25] B Sashidhar, SK Althaf Hussain Basha, Y R Ramesh Kumar , A Govardhan, "A Case Study: Data Mining and Data Modelling Techniques Applied to Student Enrollment", National Conference on Data Modeling, Image Analysis Pattern Recognition (DMIAPR) 2011 at GITAM Institute of Technology, GITAM University, Vizag.

[26] N Kartiek, Sk Althaf Hussain Basha, "Forecasting the Academic Results of Students using Artificial Neural Networks", National Conference in Modern. Trends in Computer Science and Technology (NCMTCSCT2013), ECET, Hyderabad,2013,ISSN:978-162776537-4.

[27] Y R Ramesh Kumar, SK Althaf Hussain Basha, A Govardhan, B.Sasidhar, " Mining Educational Data to Analyze Academic Student's Performance", National Conference in Modern.TrendsinComputerScienceandTechnology(NCMTCSCT2013),ECET,Hyderabad,201 3,ISSN:978-162776537-4.

[28] D.Mounika, Sk Althaf Hussain Basha, Y.R. Ramesh Kumar, Y Vijayalatha, A. Govardhan, " Study of Education Patterns using Rare Association Mining", National Conference on Emerging Trends of Computing Technologies, (NCECT2013), GRIET,Hyderabad,pp.93- 103,2013.

[29] Stefano Calzavara, Gabriele Tolomei, Andrea Casini, Michele Bugliesi, and Salvatore Orlando.A supervised learning approach to protect client authentication on the web. TWEB, 9(3):15:1–15:30, 2015.

[30] Stefano Calzavara, Mauro Conti, Riccardo Focardi, AlviseRabitti, and Gabriele Tolomei. Mitch: A machine learning approach to the blackbox detection of CSRF vulnerabilities. In IEEE European Symposium on Security and Privacy, EuroS&P 2019, Stockholm, Sweden, June 17-19, 2019, pages 528–543, 2019.

[31] Giancarlo Pellegrino, Martin Johns, Simon Koch, Michael Backes, and Christian Rossow.Deemon: Detecting CSRF with dynamic analysis and property graphs. In Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, CCS 2017, Dallas, TX, USA, October 30 - November 03, 2017, pages 1757–1771,2017.