CROSS SITE REQUEST FORGERY DETECTION

Musku Revanth Reddy
Student
Department of Information Technology
B.V. Raju Institute of Technology
Affiliated to JNTUH
Vishnupur, Narsapur, Medak,
Telangana State, India
19211a1276@bvrit.ac.in

Meka Nithin Reddy
Student
Department of Information Technology
B.V. Raju Institute of Technology
Affiliated to JNTUH
Vishnupur, Narsapur, Medak,
Telangana State, India
19211a1268@bvrit.ac.in

A Swamy Goud
Assistant professor
Department of Information Technology
B.V. Raju Institute of Technology
Affiliated to JNTUH
Vishnupur, Narsapur, Medak,
Telangana State, India
swamygoud.a@bvrit.ac.in

Abstract

In this project, we propose a methodology to leverage Machine Learning (ML) for the detection of web application vulnerabilities. Web applications are particularly challenging to analyses, due to their diversity and the widespread adoption of custom programming practices. ML is thus very helpful for web application security: it can take advantage of manually labeled data to bring the human understanding of the web application semantics into automated analysis tools. We use our methodology in the design of Mitch, the first ML solution for the black-box detection of Cross Site Request Forgery (CSRF) vulnerabilities. Mitch allowed us to identify 35 new CSRFs on 20 major websites and 3 new CSRFs on production software.

Keywords: *Machine Learning, Random Forest,Cross-site,Black Box*

I.INTRODUCTION

Web applications are the most common interface to security sensitive data and functionality available nowadays. They are routinely used to file tax incomes, access the results of medical screenings, perform financial transactions, and share opinions with our circle of friends, just to mention a few popular use cases. On the downside, this means that web applications are appealing targets to malicious users (attackers) who are

determined to force economic losses, unduly access confidential data or create embarrassment to their victims. Securing web applications is well known to be hard. There are several reasons for this, ranging from the heterogeneity and complexity of the web platform to the adoption of undisciplined scripting languages offering dubious security guarantees and not amenablefor static analysis. In such a setting, black-box vulnerability detection methods are particularly popular. As opposed to white-box techniques which require access to the web application source code, black-box methods operate at the level of HTTP traffic, i.e., HTTP requests and responses. Though this limited perspective might miss important insights, it has the key advantage of offering a language-agnostic vulnerability detection approach, which abstracts from the complexity of scripting languages and offers a uniform interface to the widest possible range of web applications. This sounds appealing, yet previous work showed that such an analysis is far from trivial. One of the main challenges there is how to expose to automated tools a critical ingredient of effective vulnerability detection, i.e., an understanding of the web application semantics. Example: Cross-Site Request Forgery (CSRF) Cross-Site Request Forgery (CSRF) is a well-known web attack that forces a user into submitting unwanted, attacker controlledHTTP requests towards a vulnerable web application in which she is currently authenticated. The key concept of CSRF is that the malicious requests are routed to the web application through the user's browser, hence they might be indistinguishable from intended benign requests which were actually authorized by the user.

II.IMPLEMENTATION

MODELS USED

Quandl: Quandl is a platform that provides users with financial, financial, and alternative datasets.

Sklearn: Provides a variety of efficient tools for machine learning and statistical modelling, including classification, regression, clustering, and dimensionality reduction through Python's integrity interface.

Numpy: Numpy is a Python library used to manipulate arrays. There are functions for working in the fields of linear algebra, Fourier transforms, and matrices

Matplotlib: Matplotlib is a cross-platform data visualization and graphical plot library for Python and its numeric extension numpy.

The UML DIAGRAMS are as follows

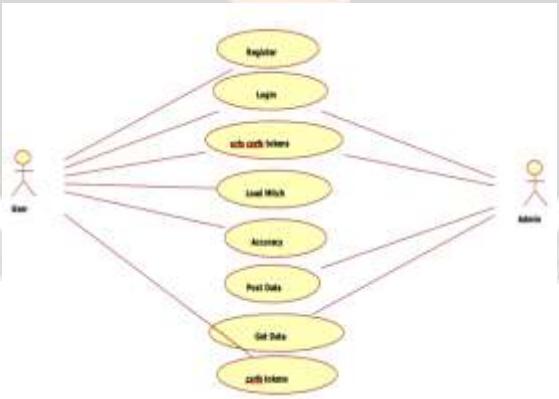


FIG-1:USECASE DIAGRAM

A use case diagram in the Unified Modeling Language (UML) is a type of behavioral diagram defined by and created from a Use-case analysis. Its purpose is to present a graphical overview of the functionality provided by a system in terms of actors, their goals (represented as use cases), and any dependencies between those use cases. Themain purpose of a use case diagram is to show what system functions are performed for which actor. Roles of the actors in the system can be depicted.

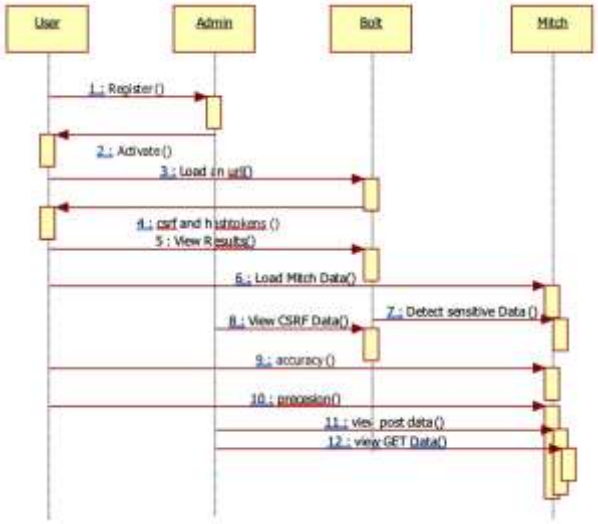


FIG-2:SEQUENCE DIAGRAM

A sequence diagram in Unified Modeling Language (UML) is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of a Message Sequence Chart. Sequencediagrams are sometimes called event diagrams, event scenarios, and timing diagrams.

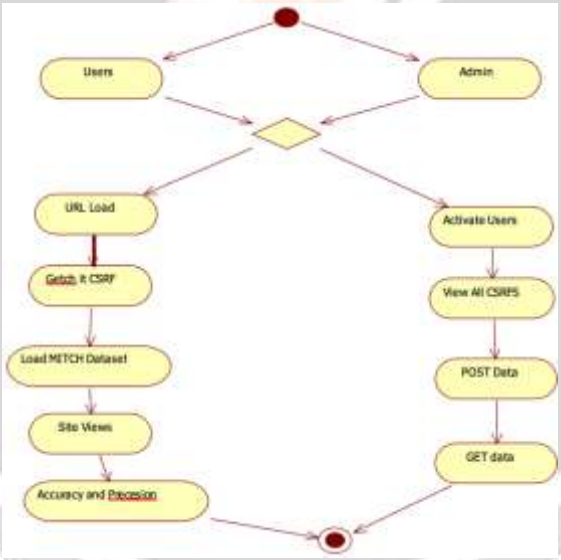


FIG -3 ACTIVITY DIAGRAM

Activity diagrams are graphical representations of workflows of stepwise activities and actions with support for choice, iteration and concurrency. In the Unified Modeling Language, activity diagrams can be used to describe thebusiness and operational step-by- step workflows of components in a system. An activity diagram shows the overallflow of control.

III.RELATED WORK

To create a machine learning model, import libraries like pandas, scikit learn. Then import the dataset to train a model. Use Random Forest classifier to create the ML model. The imported dataset is used to test the ML model created.

An architecture diagram is a graphical representation of a set of concepts, which

An architecture diagram is a visual representation of all the elements that make up part, or all, of a system.

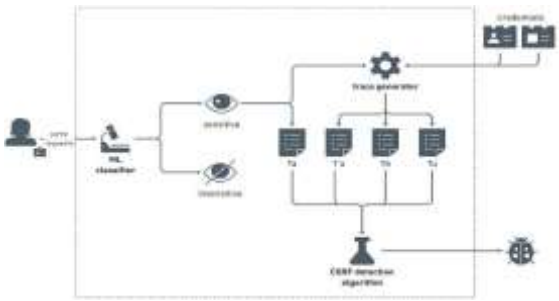


FIG-3:ARCHITECTURE

CODE

USERS\VIEWS.PY

```
from django.shortcuts import render,
HttpResponsefrom django.contrib import
messages
from .forms import UserRegistrationForm
from .models import UserRegistrationModel, UserSearchUrlModel, CSRFResponseimport json

import subprocess
import pandas as
pd
from .UserMachineLearningAlgorithms import MLConcepts# Create your views here.

def UserRegisterActions(request): if request.method == 'POST':

form = UserRegistrationForm(request.POST)if form.is_valid():

print('Data is
V alid')form.save()
messages.success(request, 'You have been successfully

registered')form = UserRegistrationForm()

return render(request, 'UsersRegister.html', {'form': form})else:

messages.success(request, 'Email or Mobile Already

Existed')print("Invalid form") else:

form = UserRegistrationForm()
return render(request, 'UsersRegister.html', {'form':

form})def UserLoginCheck(request): if request.method == "POST":
```

loginid = request.POST.get('loginname')pswd
= request.POST.get('pswd')
print("Login ID = ", loginid, ' Password = ', pswd)try:

check = UserRegistrationModel.objects.get(loginid=loginid, password=pswd) status = check.status
print('Status is = ',
status)if status ==

"activated":
request.session['id'] = check.id request.session['loggeduser'] = check.name request.session['loginid'] = loginid
request.session['email'] = check.email print("User id At", check.id, status)

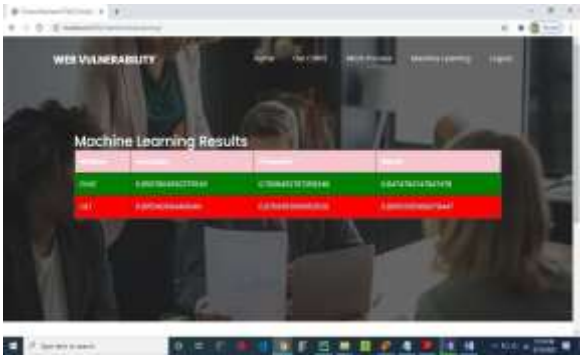
IV. RESULTS

V.CONCLUSION

Web applications are particularly challenging to analyse, due to their diversity and the widespread adoption of custom programming practices. ML is thus very helpful in the web setting, because it can take advantage of manually labeled data to expose the human understanding of the web application semantics to automated analysis tools. We validated this claim by designing Mitch, the first ML solution for the blackbox detection of CSRF vulnerabilities, and by experimentally assessing its effectiveness. We hope other researchers might take advantageof our methodology for the detection of other classes of web application vulnerabilities.

Future Work Enhancements:-

This work provides the most up to date and comprehensive account of the nature of the CSRF attacks and corresponding solution to thwart these attacks. However, as the new knowledge pioneered in this work takes hold, the future extensions of this knowledge are bound to happen. Some areas of further work, in future, are identified asfollows: As a future extension of this work it may be possible to perform the Bayesian estimation by using 1-99 (configurable) threshold probability ratio of suspect CSRF page to safe pages since anything lower than 1% of the threshold can either be random or an unexpected variance] Incorporating routine CSRF scanning in commercial anti-malware • Browser specific and generic anti CSRF solutions.



VI.REFERENCES

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, 2012.

Adam Barth, Collin Jackson, and John C. Mitchell. Robust defenses for cross-site request forgery. InProceedings of the 2008 ACM Conference on Computer and Communications Security, CCS 2008, Alexandria, Virginia, USA, October 27-31, 2008, pages 75–88, 2010.

Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. Foundations of Machine Learning. The MITPress, 2008.

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.

Stefano Calzavara, Riccardo Focardi, Marco Squarcina, and Mauro Tempesta. Surviving the web: Ajourney into web session security. ACM Comput. Surv., 50(1):13:1–13:34, 2019.

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

Stefano Calzavara, Alvise Rabitti, Alessio Ragazzo, and Michele Bugliesi. Testing for integrity flaws in web sessions. In Computer Security - 24rd European Symposium on Research in Computer Security, ESORICS 2017, Luxembourg, Luxembourg, September 23-27, 2017, pages 606–624, 2017.

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