Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Volume 13, Issue 03 March 2024

Code Guardian Pro

Mr. V Narasimha, Gopu Chitra Bhanu Reddy, Twinkle Sharma, Jakkidi Santhosh,
Assistant Professor, Department of CSE, CMR College of Engineering & Technology, Hyderabad, Telangana
UG Student, Department of CSE, CMR College of Engineering & Technology, Hyderabad, Telangana

***Abstract* – The Code Guardian Pro project is an advanced complex tool find bugs and errors in their code as well as the software built and later improved the overall quality and the level of productivity. The tool employs sophisticated parsing of programming language constructs, static code analysis, and identification of patterns to detect bugs both at a syntax level and logical level. The bug warlock in Code Guardian Pro gives more than standard identify errors but also provides briefly but understandable tips on how to correct the identified errors. In addition to this, the Code Inspection and Security Analysis tools are also able to conduct detailed vulnerability analysis, exposing critical security risks found within the source code. Not only it goes through the source codes, but it can also identify devious patterns in websites, emails and even may detect errors in the images. On the other hand, the Code Guardian Pro naturally expedites the debugging, decreases the development time, and increases the robustness, security, and reliability of software applications regardless of the size of the potential bugs and issues. Consequently, the end product is a better user experience.**

***Keywords* – vulnerability scanning, debugging ,logical error, syntax error, files scanning, urls scanning.**

I. INTRODUCTION

Developing quality software is essential in today's technology-driven world.However, bugs, errors and weaknesses in the source code can be severe participate in software development. Traditional bug detection methods are often time-consuming and may not detect all possible issues, leading to sub-flat software quality and security risks To meet this challenge, an improved tool is needed which just works. The process of identifying and fixing bugs, syntax errors, logical problems, and security vulnerabilities in source code, websites, and emails.

It is multifaceted software development aimed to deal with exceptional bug exception.Correction and code quality upgrade. It uses both static code analysis and pattern recognizing machines.Algorithms to detect syntax and logical problems, that will fail to expose all the runtime flaws.On the other hand, the project is also about delivering a thorough threat analysis to bring to light possible security loopholes. Risks in the codebase. It applies to programming codes, website content, message formats, and picture quality. Reviewing for full security checking, verifications for thoroughness, and testing. The main aim of this tool is to offer.

make the development process shorter by delivering useful directives and simple debugging which, then, brings the time needed for this process to an end and the reforms can ensure the reduction of costs and the rise of quality and security of software development.It stands because information should be a practice of any developer`s kit. Powered by Code Guardian Pro, this automated tool can have positive results but also false positives and false negatives from time to time.

necessitating careful developer scrutiny. The primary emphasis here is on static code analysis, of course with the focus on possible

Noticing the lack of runtime errors and the difficulty in dealing with intricate code including unusual methods.While as it only provides the guidance, human expertise still occupies the central position to make complex decisions. Compatibility

barriers, demand, infringement on privacy and slowness of learning are the aspects to take into account. Consistently focus on reduced costs, and wise use of licenses should not be a problem for the large development teams alongside the small teams therefore a security audit may need a more detailed retesting process.

II. LITERATURE REVIEW

2.1 NMAP

Nmap [1] is a port scanner which is frequently used to make the list of ports. IP address and all information related to it are caught by chit chat in it. If an IP is provided, then that tunes find the host sub-domain to which it has been assigned to, It also finds out what interfaces in this host the services reside, how many ports are being run, number of opened ports, closed ports, software or hardware services that are presented by these ports, which may be using TCP, FSP or any other communication protocol. It is able to forecast the type of operating system player is connected to the network. The particular machine host topology is portrayed in the form of a tree diagram whereby the various gateways that are being utilized for localization of the machine are depicted. The next requisite is to open the ports so the attacker will gain an opportunity to steal the intellectual property. A number of ranges of ports can be scanned with a tool such as Nmap.

2.2 NESSUS

Additionally, deploying Nessus [2] discovers the vulnerabilities on the remote host and returns the list of found errors. The stem cell has two types of scan: internal and external. An Origin Scan is an inner scanning depend

on the router that is present in the host computer that IP. Other thane starition the process identifiees the host out of a circulese t a certain router. Application web tests which can be performed using a scanner constitute another type of test. Here there are three methods which can be employed; the first one being scanning at the initial instance or the facilitation of a template to be able to perform scanning on an attacked host. One particular machine can perform a plurality of scans in parallel. Nessus is able to make analyzation of vulnerability to the security context, and formulate this context in four typologies which are high, medium, low, and information. Results are saved as soon as the host is scanned. Such archival occur either the scanning completed completely for all hosts. Vulnerabilities are disclosed in both ways- vulnerabilities. The report that will be generated can directly go and be used to identify and solve the vulnerability. The former deals with the problems that are listed on the provided form, host compliance inspection and assessment takes care of these accordingly. Results can be exported in any form, e.g. CSV or JSON. Nessus is based on a combination of client-server architecture; it uses HTTP for monitoring, Nessus Responder scanning, and the Nessus Web User Interface. Each session is served by the client, and the server-side validation checks it.

Qabajeh et al. (, 2018) has recently looked at traditional and automated phishing detection by using a very similar methodology. The traditional anti-phishing methods implicate training the individual, holding workshops, and session, the contents of which frequently loop in the legal aspect. In the following passages, Computerized or automated anti-phishing includes list-based and Machine learning-based approaches. The mentioned two technics are compared giving their similarities, and some elements are positive and others are negative from the user's perspective. Based on the studies, machine learning and rule induction, respectively, are the most appropriate tools that can be used to protect against phishing attacks. The limitations of this work are: the review highlights 67 research items that are the founding principle and the research work does not incorporate Deep Learning methods for online phishing website detection.

Zuraiq & Alkasassbeh (2019) conducted an in-depth study of phishing detection methods yielded up to date. The study explicates Anti-Phishing techniques such as Heuristic, Content-Based and Fuzzy rule-based approaches. The finding suggests that phishing websites can be detected more accurately by better methods for identifying them. Research of data was started from 2013 and last till 2018 and that is the base of the project that was carried out. A limitation of this research was the fact that it reviewed only 18 papers; so, more research is required to include Machine Learning, List Based and Deep Learning approaches for phishing website detection.

Benavides et al. (Benavides et al., 2020) conducted a review or an examination of the other researchers' ways to identify phishing (It involved applying the Deep Learning algorithms) in order to enhance the cybersecurity. Finally, the most recognizable rate in the matter of Deep Learning algorithms for fraud email detection is still inadequate. The literature in this work is exclusively based on only 19 studies that were published in 2014 till 2019 journals. Indeed, providing the searched phrase "phishing and Deep Learning" is the way for finding only peer-reviewed articles with the essential topics.

Athulya and Praveen (Athulya, P., 2020) pointed out different phishing attacks that are still a part of the phisher's arsenal, taking into consideration the phisher's most recent phishing tactics as well as some

exploitability by the plugin or exploitability by the host. The detection includes all the vulnerabilities discovered during the scanning and subsequently composed of the CVE lists affected by these

anti-phishing strategies. Alongside that, at the heart of the article is a goal of gaining consciousness in the community regarding phishing attacks and methods which have been employed for phishing detection. The investigation stated in the first sentence of this paragraph; clearly the only prevention is to educate users on different types of phishing attacks. The question of choosing what type of security software tool to find a potential phishing email, browser extensions have been found to be the most effective among users. This research was built off of nine articles. The study does not explore Deep Learning methods for vulnerability detection aimed at phishing websites.

Qiu and al. [3] have given a generalized review of ways that artificial intelligence can be used in both attacking and withdrawing in security attacks, primarily at the stages of training and testing. In their work, they aimed at classifying attacks in the classes of natural language processing, cyberspace security, computer vision, and the physical world such attacks. Likewise, they conceptualized the defense sideways in their research and recommended techniques to counter against certain forms of attack. In the work by Martins et al.[4] similar or more than fifteen sample papers were taken that were based on adversarial machine learning techniques used in intrusion detection and malware detection model. In their work, the authors presented the classification of the most common intrusion and malware recognition methods using adversarial attacks and protecting oneself against them.

Muslihi et al. [5] present a review of more than 14 papers that can detect SQL Injection attacks via some deep learning methods which include Carton neural networks, LSTM, DBN, MLP, and Bi-LSTM. Moreover, the authors have analyzed several of the methods aiming at objective, procedure, feature, and dataset points. Muhammad et al. [6] were focused on arranging and evaluating, by means of an analytical approach, the currently known methods and tools that can be applied to prevent an SQL injection attack. The considered studies total 82. The review outcomes revealed that most of the researchers studied and suggested the SQLIA techniques detection approaches (SQLIAs) and not do their best to test the pragmatic approaches.

III. METHODOLOGIES

A. Finding the Phish tool :

The first variant will be used build a collector unit which will accept a URL as the input. Then in the back end, we will use total wires API which is an online tool to analyze the URL by comparing factors such comparison of parameters to detect malware, viruses, and any other potential threats. When you do input the desired URL, the tool will be able to generate corresponding key terms.

Follows these steps:

1. Parsing the URL: uses the URL the users have given to provide the destination domain and route.

2. Domain analysis: it is responsible for identification the domain's repute and any lists of the banned domain names.

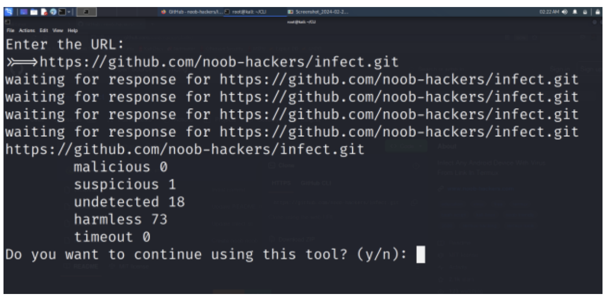
3. URL analysis: thoroughly scrutinizes the URL's form and content in search for any weird traffic tendency.

5. File scanning: makes use of a varying set of antivirus engines to determine if the file may be unsafe by running a check for well-known malware signs.

6. Behavior analysis: The alternate option is dynamic analysis, where the file is executed in a sandbox that watches behavior and traces any malicious activities.

7. Threat detection: is compiling the scan data from different modules and threat behavior analysis functionality, showing everything in one place in a simple manner and confidence levels.

8. Reporting: namely, returns an analysis prompts along with the analysis results which can be downloaded by the user.



```
Enter the URL:
⋙https://github.com/noob-hackers/infect.git
waiting for response for https://github.com/noob-hackers/infect.git
waiting for response for https://github.com/noob-hackers/infect.git
waiting for response for https://github.com/noob-hackers/infect.git
waiting for response for https://github.com/noob-hackers/infect.git
https://github.com/noob-hackers/infect.git
        malicious 0
        suspicious 1
        undetected 18
        harmless 73
        timeout 0
Do you want to continue using this tool? (y/n):
```

Fig. 1 Testing out the github link for the phising or malicious link

B. Scan Vulnerability

The second alternative is about vulnerability scans, it is a tool for penetration testers, bug bounty hunters or security researchers who have authorization to perform security testing, this tool has multiple exploits that can be used in exploitation of a target and determining whether the target is vulnerable or not. We will also asses how critical the vulnerability is.

1.CMS Detection: The tools are CMS detection tools. Thus, it is possible to identify favorite content management systems such us WordPress, Joomla, PrestaShop, Drupal, OpenCart, Magento or Lokomedia management systems. This allows to be done to discover the basics of the target web application's technology.

2.Information Gathering: It provides mechanism for collecting target information, which can range from the underlying information about target website such as the version, and any publicly available details.

3.Subdomain Gathering: A tool may collect and list all the subdomains pertinent to the target domain is its primary function. This allows increasing the scale of the analysis and finding new weak points of a goal in the target system.

4.Multi-threading on Demand: This tool, with multi-threading, which parallel processes multiple tasks concurrently, has it like this. This way can accelerate the duration of evaluation.

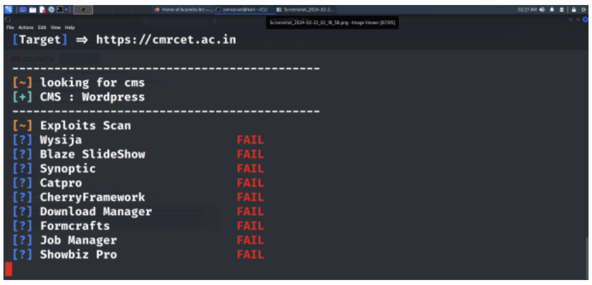
5.Vulnerability Checks: This utility can enhance the process in which target web applications are checked for already known security issues. It is therefore, it is possible to recognize possible failure points that can be used by the attackers as a gateway into an organization's network system.

6.Auto Shell Injector: Andthe tool next to the auto shell injector will detect thevulnerabilities used by attackers to inject malicious code onto various targets' web applications.

7.High-level Port Scan: The tool can use a process of a high-level port scan to locate vulnerabilities in the target web traffic. This can be done to discover open ports which in return, could be used by hackers to steal the data in the targeted infrastructure in an unauthorized way.

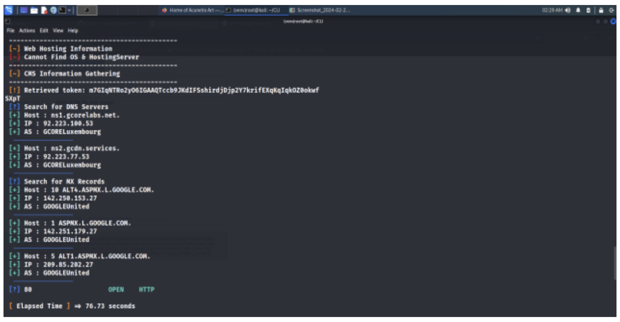
8.DNS Server Dump: This tool performs duplication of DNS servers data which are the ones connected with the particulate target website.

9.Input Multiple Targets: The tool allows you to enter shallow as well as deep web applications for assessment. That enables users to see different locations in one scan pause.



```
[Target] ⇒ https://cmrcet.ac.in

----------------------------------------------
[~] looking for cms
[+] CMS : Wordpress
----------------------------------------------
[~] Exploits Scan
[?] Wysija                    FAIL
[?] Blaze SlideShow           FAIL
[?] Synoptic                  FAIL
[?] Catpro                    FAIL
[?] CherryFramework           FAIL
[?] Download Manager          FAIL
[?] Formcrafts                FAIL
[?] Job Manager               FAIL
[?] Showbiz Pro               FAIL
```

Fig.2 Exploits Scan

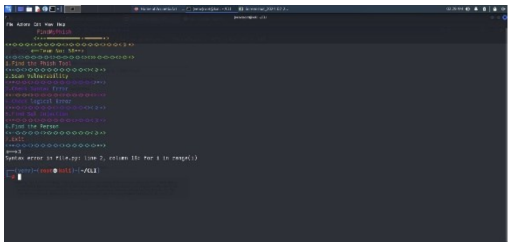


```
----------------------------------------------
[-] Web Hosting Information
[-] Cannot Find OS & HostingServer
----------------------------------------------
[-] CMS Information Gathering
----------------------------------------------
[!] Retrieved token: m7GIqNTRo2yO6IGAAQTccb9JKdIFSshirdjDjp2Y7krifEXqKqIqkOZ0okwf
SXpT
[?] Search for DNS Servers
[+] Host : ns1.gcorelabs.net.
[+] IP : 92.223.100.53
[+] AS : GCORELuxembourg

[+] Host : ns2.gcdn.services.
[+] IP : 92.223.77.53
[+] AS : GCORELuxembourg

[?] Search for MX Records
[+] Host : 10 ALT4.ASPMX.L.GOOGLE.COM.
[+] IP : 142.250.153.27
[+] AS : GOOGLEUnited

[+] Host : 1 ASPMX.L.GOOGLE.COM.
[+] IP : 142.251.179.27
[+] AS : GOOGLEUnited

[+] Host : 5 ALT1.ASPMX.L.GOOGLE.COM.
[+] IP : 209.85.202.27
[+] AS : GOOGLEUnited

[?] 80                  OPEN    HTTP

[ Elapsed Time ] ⇒ 76.73 seconds
```

Fig.3 Result of vulnerability scanning.

C. Check syntax error

Pattern Matching: I have the ability to be trained on a large dataset ranging from the lines of code to vocabulary of natural language which can enable me to identify common patterns and structures associated with valid syntax. Identifiers, which are the parts of the programming code that give instructions to the computer, are a critical element of syntax. An example is a missing punctuation, incorrect keyword, or mismatched bracket may be cues for syntax errors that might need to be corrected.
Rule-Based Checking: Being armed with precise attributes such as rules and constraints of a particular programming language or writing style can be my instrumentality. Compliance with specific regulations, such as in the case of "cheat" or weak grammar structures, are shown as errors.



```
FindMyPhish
<==Team No: 58==>
1.Find the Phish Tool
2.Scan Vulnerability
3.Check Syntax Error
4.Check logical Error
5.Find Sql Injection
6.Find the Person
7.Exit
==>3
Syntax error in file.py: line 2, column 18: for i in range(1)

┌──(venv)-(root㉿kali)-[~/CLI]
└─#
```

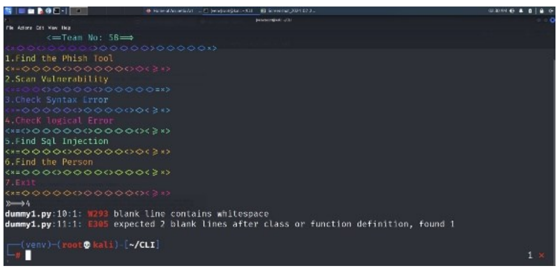
Fig.4 syntax error

D. Check logical error

Data Flow Analysis: This approach observes the data path across the code, thus locating problems as faulty lines like. This includes using the variables before defining them, awarding variables their values, but not referring to them later in the program. Duplication of the calculations or the operations that serve no purpose.

Type Checking: Some aspects may carry out type checking for the purpose of maintaining the data type use to be consistent and quality. This will stop problem of assigning wrong values into variables and using wrong operators in respect to certain data types.

Testing and Debugging: Yet, the issue of testing and debugging is not a step of the detection itself; a tool that runs your code and observes its output can find logical errors there. These tools may offer features like, setting breakpoints to stop at certain points of the run to check up on the value of the different variables. Moving from one code line to another will help you to develop an understanding of the flow of the logic.



```
<==Team No: 58==>
1.Find the Phish Tool
2.Scan Vulnerability
3.Check Syntax Error
4.Check logical Error
5.Find Sql Injection
6.Find the Person
7.Exit
==>4
dummy1.py:10:1: W293 blank line contains whitespace
dummy1.py:11:1: E305 expected 2 blank lines after class or function definition, found 1

┌──(venv)-(root㉿kali)-[~/CLI]
└─#                                                                          1 ×
```

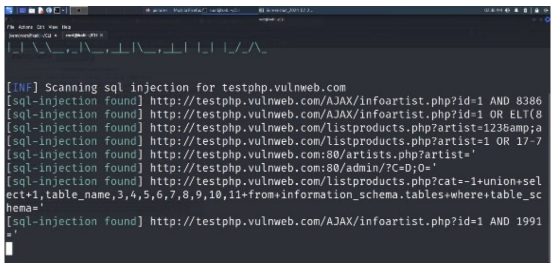
Fig.5 Logical error

E. Find SQL injection

URL Generator: Build URL for downloading Wayback Machine's CDX API resource that the user would provide domain and a subdomain flag.

HTTP Requesting: And extraction functions call the generated URLs to get the responses and the internal URLs are crawled.

SQL Injection Scanning:
Identifies parameters from the webpage address URLs.
Executing sql commands from payloads stored in a file.
Combines URLs and payloads, sending the requests to varying locations.
Looks for SQL injection granting access vulnerabilities in the response.
The script is designed to identify SQL injections against host of URLs received as a parameter while using diverse payloads to execute SQL injections.
If a response contains some keywords (like "SQL", "Sql") which could mean that a website is vulnerable to SQL injection, it is marked as Threat by some tools.



```
[INF] Scanning sql injection for testphp.vulnweb.com
[sql-injection found] http://testphp.vulnweb.com/AJAX/infoartist.php?id=1 AND 8386
[sql-injection found] http://testphp.vulnweb.com/AJAX/infoartist.php?id=1 OR ELT(8
[sql-injection found] http://testphp.vulnweb.com/listproducts.php?artist=123&amp;a
[sql-injection found] http://testphp.vulnweb.com/listproducts.php?artist=1 OR 17-7
[sql-injection found] http://testphp.vulnweb.com:80/artists.php?artist='
[sql-injection found] http://testphp.vulnweb.com:80/admin/?C=D;O='
[sql-injection found] http://testphp.vulnweb.com/listproducts.php?cat=-1+union+sel
ect+1,table_name,3,4,5,6,7,8,9,10,11+from+information_schema.tables+where+table_sc
hema='
[sql-injection found] http://testphp.vulnweb.com/AJAX/infoartist.php?id=1 AND 1991
='
```

Fig.6 SQL injection detection

F. Find the person

i.IP Address
An IP lookup works by querying a database or service that contains information about IP addresses and their associated geolocation data and other metadata. Here's a general overview of how an IP lookup works:

IP Address Input: The user provides an IP address as input to the IP lookup tool or service.

Database Query: The IP lookup tool or service sends a query to its internal database or an external database or service that contains information about IP addresses.

Geolocation Data Retrieval: The database or service

Published by :
http://www.ijert.org

International Journal of Engineering Research & Technology (IJERT)
ISSN: 2278-0181
Volume 13, Issue 03 March 2024

responds to the query by providing the geolocation data associated with the input IP address. This data typically includes information such as the country, region, city, and even the zip code.

Additional Metadata Retrieval: Depending on the capabilities of the IP lookup tool or service, it may also retrieve additional metadata associated with the input IP address. This can include information such as the name of the internet service provider (ISP), the organization or individual associated with the IP address, and other relevant details.

Data Presentation: The IP lookup tool or service presents the retrieved geolocation data and any additional metadata to the user in a user-friendly format, such as a map displaying the location of the IP address or a table containing the various metadata details.

ii. Gmail
API Interface: Interacts with Google APIs (publicly available interfaces to Google services) to store information.

Data Extraction: Based on the targeted email, GHunt extracts data points from relevant services e.g. Google Account : Name, Google ID YouTube: Channel information (if publicly available)

Google Services: Images, Maps and other functional services (depending on the user's privacy settings).
It uses asynchronous processing to process multiple requests simultaneously, improving performance.

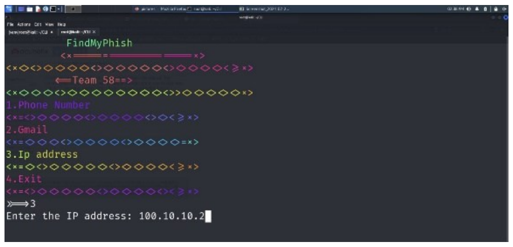
Data Export: Extracted data can be exported in JSON format for further analysis or integration.

iii. Phone number :
Phone number data searches mainly focus on reverse phone numbers and spam score detection. It does not provide the functionality to extract personal information beyond the name associated with the phone number.

Name: The name associated with the phone number (if it exists in the database).
Spam Score: An indicator of the likelihood that a number is spam by a caller.

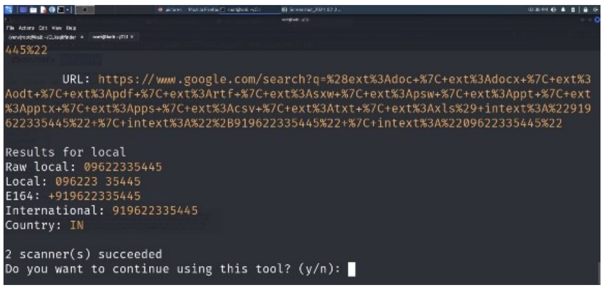
API requests: Developers integrate the API into their applications via authentication keys to make specific requests for phone numbers to be verified.

Data Response: Upon success, the API responds with a structured data structure with name, spam score, and possibly true score (depending on sharing type).



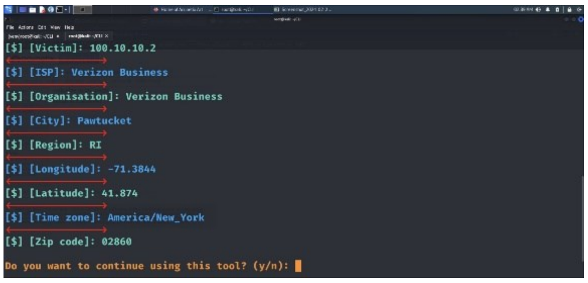
```
            FindMyPhish
          <×═════=═══════════ ═════×>
<×◇<>◇◇◇◇<>◇◇◇◇◇<>◇◇◇◇◇<≥×>
        ⟸Team 58==>
<×◇◇◇<>◇◇◇◇◇◇◇◇<>>◇◇◇◇◇×>
1.Phone Number
<×=<>◇◇◇◇<>◇◇◇◇<>◇<≥×>
2.Gmail
<×=◇◇◇<>◇◇◇◇<>◇◇◇◇=×>
3.Ip address
<×=◇<>◇◇◇◇◇<>◇◇◇◇<≥×>
4.Exit
<×=<>◇◇◇◇◇<>◇◇◇◇<><≥×>
»⟹3
Enter the IP address: 100.10.10.2
```

Fig.7 IP address Lookup



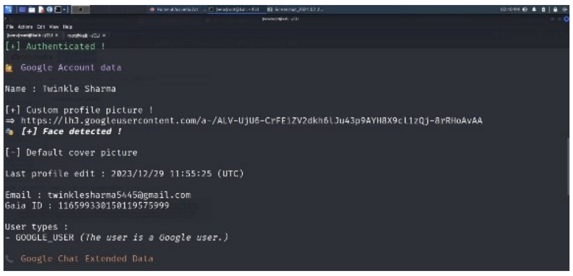
```
445%22

       URL: https://www.google.com/search?q=%28ext%3Adoc+%7C+ext%3Adocx+%7C+ext%3
Aodt+%7C+ext%3Apdf+%7C+ext%3Artf+%7C+ext%3Asxw+%7C+ext%3Apsw+%7C+ext%3Appt+%7C+ext
%3Apptx+%7C+ext%3Apps+%7C+ext%3Acsv+%7C+ext%3Atxt+%7C+ext%3Axls%29+intext%3A%22919
622335445%22+%7C+intext%3A%22%2B919622335445%22+%7C+intext%3A%2209622335445%22

Results for local
Raw local: 09622335445
Local: 096223 35445
E164: +919622335445
International: 919622335445
Country: IN

2 scanner(s) succeeded
Do you want to continue using this tool? (y/n):
```

Fig.8 Finding IP location on Gmaps.



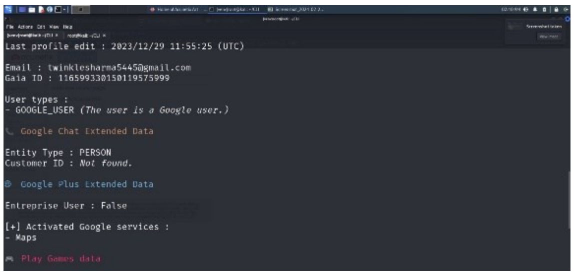
```
[$] [Victim]: 100.10.10.2
<──────────────>
[$] [ISP]: Verizon Business
<──────────────>
[$] [Organisation]: Verizon Business
<──────────────>
[$] [City]: Pawtucket
<──────────────>
[$] [Region]: RI
<──────────────>
[$] [Longitude]: -71.3844
<──────────────>
[$] [Latitude]: 41.874
<──────────────>
[$] [Time zone]: America/New_York
<──────────────>
[$] [Zip code]: 02860

Do you want to continue using this tool? (y/n):
```

Fig.9 Phone number lookup



```
[+] Authenticated !

  Google Account data

Name : Twinkle Sharma

[+] Custom profile picture !
⇒ https://lh3.googleusercontent.com/a-/ALV-UjU6-CrFE1ZV2dkh6lJu43p9AYH8X9cl1zQj-8rRHoAvAA
  [+] Face detected !

[-] Default cover picture

Last profile edit : 2023/12/29 11:55:25 (UTC)

Email : twinklesharma5445@gmail.com
Gaia ID : 116599330150119575999

User types :
- GOOGLE_USER (The user is a Google user.)

  Google Chat Extended Data
```

Fig.10 Google account data.



```
Last profile edit : 2023/12/29 11:55:25 (UTC)

Email : twinklesharma5445@gmail.com
Gaia ID : 116599330150119575999

User types :
- GOOGLE_USER (The user is a Google user.)

  Google Chat Extended Data

Entity Type : PERSON
Customer ID : Not found.

  Google Plus Extended Data

Entreprise User : False

[+] Activated Google services :
- Maps

  Play Games data
```

Fig.11 Gmail account data.

IV. RESULTS AND DISCUSSION

The Code GuardianPro Project is a game changer providing designers and testers with a revolutionary software quality assurance approach. Its extensive capabilities, including code frame extraction technique that uses modern parsing algorithm, static code analysis, and pattern recognition, make it the solution that offers something different than other current tools.

The traditional bug detection instruments are mostly used to identify syntax errors. But, Code Guardian Pro is different. It digs into every possibility of a complexity which is even beyond syntax level and identify logical errors as well. This methodology is based on a comprehensive analysis since it involves the full codebase, and the results are more faultless and high-quality.
Code GuardianPro is a standout feature of the debugging system because it is based on the proactive approach of error resolution, which provides programmers with the technical knowledge of how to fix the errors detected. This enables it to outpace the majority of other currently available tools that necessitate extra efforts from developers: from figuring out what has failed, to proposing the solution for the identified issue. Being a tool for discovery rather than dust usage, the project cuts down debugging time. Therefore, it boosts development efficiency. What is more, the tool not only detects and locates devious bugs but also security risks at the deep layer of code. Its versatility in the determination of the nature of various digital entities, including websites and emails, opens the developers sights to a fuller picture of the risks involved. Moreover, it empowers them to act swiftly on the identified vulnerabilities prevalent not only in one but several mediums. Code Guardian Pro offers help in achieving software reliability, security, and efficiency in one package.

V. CONCLUSION

In conclusion, the Code GuardianPro understands the programming as a huge development step in the software developing process. It concentrates on bug-detection and code-quality improvement by using novel approaches of syntax analysis, static code analyzer, and pattern finding. The unique nature of microfinance is that it suits to the local needs successfully perform source code, website, email and images scanning, making it robust and covers all possibilities. This adaptability drives development by simplifying decision-making processes for developers, giving them intelligence-based evidence. While processing the digitized sets in this age of growing digital vulnerabilities it make sure the software reliability by detecting syntax, logical errors, and security issues. Code GuardianPro paves the way for a new trend, which is a from repairing to innovation and then to achieving stable and secure software which performs exceptionally and enables great user experience. As a result, this project translates the age of software development when security is both possible and pure satisfaction.

VI. REFERENCES

[1] Mansour Alsaleh, Noura Alomar, Monirah Alshreef, Abdulrahman Alari and AbdulMalik Al-Salman, “Performance-Based Comparative Assessment of Open Source Web Vulnerability Scanners”, (2017) .
[2] Sheetal Bairwa, Bhawna Mewara and Jyoti Gajrani, “Vulnerability Scanners: A Proactive Approach to Assess WebApplication Security”, (2014)
[3] Qiu, S.; Liu, Q.; Zhou, S.; Wu, C. Review of artificial intelligence adversarial attack and defense technologies. Appl. Sci.**2019**, 9, 909. [Google Scholar] [CrossRef]
[4] Mart ins, N.; Cruz, J.M.; Cruz, T.; Abreu, P.H. Adversarial Machine Learning Applied to Intrusion and Malware Scenarios: A Systematic Review. IEEE Access**2020**, 8, 35403–35419. [Google Scholar] [CrossRef]
[5] Muslihi, M.T.; Alghazzawi, D. Detecting SQL Injection on Web Application Using Deep Learning Techniques: A Systematic Literature Review. In Proceedings of the 2020 Third International Conference on Vocational Education and Electrical Engineering (ICVEE), Surabaya, Indonesia, 3–4 October 2020. [Google Scholar] [CrossRef]
[6]Aliero, M.S.; Qureshi, K.N.; Pasha, M.F.; Ghani, I.; Yauri, R.A. Systematic Review Analysis with SQLIA Detection and Prevention Approaches. Wirel. Pers. Commun.**2020**, 112, 2297–2333. [Google Scholar] [CrossRef]
[7] Anti-Phishing Working Group (APWG), “Phishing activity trends report— second half 2011,” http://apwg.org/reports/apwg trends report h22011.pdf, 2011, accessed July 2012.
[8] B. Schneier, “Details of the RSA hack,” http://www.schneier.com/blog/archives/2011/08/details of the.html, 2011, accessed December 2011.
[9] P. Kumaraguru, Y. Rhee, A. Acquisti, L. F. Cranor, J. Hong, andE. Nunge, “Protecting people from phishing: the design and evaluationof an embedded training email system,” in Proceedings of the SIGCHIconference on Human factors in computing systems, ser. CHI ’07. NewYork, NY, USA: ACM, 2007, pp. 905–914.
[10] A. Alnajim and M. Munro, “An anti-phishing approach that uses trainingintervention for phishing websites detection,” in Proceedings of the2009 Sixth International Conference on Information Technology: NewGenerations. Washington, DC, USA: IEEE Computer Society, 2009,pp. 405–410.
[11] S. Gorling, “The Myth of User Education,” Proceedings of the 16thVirus Bulletin International Conference, 2006.
[12] G. Gaffney, “The myth of the stupid user,” http://www.infodesign.com.au/articles/themythofthestupiduser, accessed March 2011
.[13] A. Stone, “Natural-language processing for intrusion detection,” Com-puter, vol. 40, no. 12, pp. 103 –105, dec. 2007.
[14] S. Abu-Nimeh, D. Nappa, X. Wang, and S. Nair, “A comparison ofmachine learning techniques for phishing detection,” in Proceedingsof the anti-phishing working groups 2nd annual eCrime researcherssummit, ser. eCrime ’07. New York, NY, USA: ACM, 2007, pp. 60–69.
[15] C. Yue and H. Wang, “Anti-phishing in offense and defense,” in Com-puter Security Applications Conference, 2008. ACSAC 2008. Annual,8-12 2008, pp. 345 –354.
[16] P. Knickerbocker, D. Yu, and J. Li, “Humboldt: A distributed phishingdisruption system,” in eCrime Researchers Summit, 2009, pp. 1–12.
[17] L. James, Phishing Exposed. Syngress Publishing, 2005.
[18] J. S. Downs, M. Holbrook, and L. F. Cranor, “Behavioral response tophishing risk,” in Proceedings of the anti-phishing working groups 2ndannual eCrime researchers summit, ser. eCrime ’07. New York, NY,USA: ACM, 2007, pp. 37–44.
[19] H. Huang, J. Tan, and L. Liu, “Countermeasure techniques for deceptivephishing attack,” in International Conference on New Trends in Infor-mation and Service Science, 2009. NISS ’09, 2009, pp. 636 – 641.
[20] T. Moore and R. Clayton, “Examining the impact of website take-downon phishing,” in eCrime ’07: Proceedings of the anti-phishing workinggroups 2nd annual eCrime researchers summit. New York, NY, USA:ACM, 2007, pp. 1–13.
[21]S. Egelman, L. F. Cranor, and J. Hong, “You’ve been warned: anempirical study of the effectiveness of web browser phishing warnings,”in Proceeding of the twenty-sixth annual SIGCHI conference on Humanfactors in computing systems, ser. CHI ’08. New York, NY, USA:ACM, 2008, pp. 1065–1074.
[22] M. Wu, R. C. Miller, and S. L. Garfinkel, “Do security toolbars actuallyprevent phishing attacks?” in Proceedings of the SIGCHI conference onHuman Factors in computing systems, ser. CHI ’06, New York, NY,USA, 2006, pp. 601–610.
[23] S. Sheng, B. Wardman, G. Warner, L. F. Cranor, J. Hong, and C. Zhang,“An empirical analysis of phishing blacklists,” in Proceedings of the 6thConference in Email and Anti-Spam, ser. CEAS’09, Mountain view, CA,July 2009.
[24] Google, “Google safe browsing API,” http://code.google.com/apis/safebrowsing/, accessed Oct 2011.
[25] Google, “Protocolv2Spec,” http://code.google.com/p/google-safe- browsing/wiki/Protocolv2Spec, accessed Oct 2011.

[26] P. Prakash, M. Kumar, R. R. Kompella, and M. Gupta, “Phishnet:predictive blacklisting to detect phishing attacks,” in INFOCOM’10:Proceedings of the 29th conference on Information communications.Piscataway, NJ, USA: IEEE Press, 2010, pp. 346–350.
[27] Y. Cao, W. Han, and Y. Le, “Anti-phishing based on automated individ-ual white-list,” in DIM ’08: Proceedings of the 4th ACM workshop onDigital identity management. New York, NY, USA: ACM, 2008, pp.51–60.
[28] N. Chou, R. Ledesma, Y. Teraguchi, and J. C. Mitchell, “Client-sidedefense against web-based identity theft,” in NDSS. The InternetSociety, 2004.
[29] T. Holz, C. Gorecki, K. Rieck, and F. C. Freiling, “Measuring andDetecting Fast-Flux Service Networks,” in Proceedings of the Networkand Distributed System Security Symposium (NDSS), 2008.
[30] P. Likarish, D. Dunbar, and T. E. Hansen, “Phishguard: A browserplug-in for protection from phishing,” in 2nd International Conferenceon Internet Multimedia Services Architecture and Applications, 2008.IMSAA 2008, 2008, pp. 1 – 6.
[31] D. L. Cook, V. K. Gurbani, and M. Daniluk, “Phishwish: A statelessphishing filter using minimal rules,” in Financial Cryptography and DataSecurity, G. Tsudik, Ed. Berlin, Heidelberg: Springer-Verlag, 2008, pp.182–186.
[32] Y. Zhang, J. I. Hong, and L. F. Cranor, “Cantina: a content-basedapproach to detecting phishing web sites,” in Proceedings of the 16thinternational conference on World Wide Web, ser. WWW ’07. NewYork, NY, USA: ACM, 2007, pp. 639–648.