Generative AI-Based Tool for Brute Forcing IoT Devices' Default Credentials

Anas Al Rawi, Nafaa Jabeur and Raja Waseem Anwar

November 10, 2024

Generative AI-Based Tool for Brute Forcing IoT Devices' Default Credentials

Anas Al Rawi[1], Nafaa Jabeur[1], and Raja Waseem Anwar[1]

[1]*Computer Science Department, German University of Technology in Oman, Oman*

***Abstract*—This study beneficially uses the power of generative AI to search for vendor-specific default credentials and uses them to brute force IoT devices logins. IoT devices have a diverse set of open ports used for accessing and configuration. With the increased usage of IoT devices, keeping all devices' ports well-secured is overwhelming and costly, especially for SMEs. Using a variety of methods to approach the problem, this research studied IoT attacks, characteristics, IoT penetration tools, and small to medium size enterprises (SMEs) requirements to produce an automated solution. Findings indicated that a lot of IoT devices are still configured with default credentials making the networks they are connected to vulnerable attacks. The solution presented, is a script that integrates OpenAI GPT to search for default credentials, Nmap to scan for open ports, and Hydra to attack the device. The tool is implemented to assess some specific widely used ports. To detect vulnerable IoT devices and report them to the user, the tool analyses login pages available on ports 80 and 443 to search for the brand and model of the IoT device. The output is used for the default credentials GPT search. Despite its ability to shortlist the dictionary for a brute force list, it should be tested on an experiential environment that includes different IoT simulators with several open ports on changed credentials and default ones. Then verified its functionality on a real IoT network. Further research could explore implementing machine learning to thoroughly analyse IoT device firmware.**

***Index Terms*—Generative AI, IoT Device, Default Credentials, Brute Force, SMEs, Python Script**

I. Introduction

Internet of Things or IoT are devices that communicate with each other and with other systems using the internet. These devices range from simple home appliances such as garage door openers, smart refrigerators, and internet-connected door locks to much more advanced manufacturer devices such as sensors, and pipeline robots. IoT devices are also present in smart cities, traffic control systems, and watering infrastructure. A lot of devices are considered as IoT devices even before the introduction of the term IoT in 1999 [1] such as Wi-Fi routers, access points, and surveillance cameras as they are connected to the network and can be accessed through a client that is connected to the same network as well. Classification of IoT devices is based on the implementation field, to mention a couple, there is the Internet of Medical Things (IoMT), Industrial Internet of Things (IIoT), consumer and commercial IoT devices, and much more. In general, IoT devices are uncountable but in this paper, the focus will be specifically on IoT devices in households and small to medium-sized enterprises (SMEs). A very important characteristic of IoT devices is their simplicity to ensure fast and uninterrupted transaction of data. This approach leads to the main issue of IoT devices which is their security aspects. They introduce vulnerabilities to the networks they are connected to and are considered easy targets to start with for any attack. The most important vulnerability in IoT devices is the use of default credentials to log into and manage them, and it is the problem we are addressing in this report.

Default credentials are username and password combinations that are similar across a brand or manufacturer and can be found in the device's manuals or documentation, out of the box or online. These credentials are used to log in to the device's control panel or access its services. These credentials can easily be found online by an attacker which lets him get into the device control panel and start the network intrusion from there. It is very important to change these credentials after the initial setup of the device for the first time. Some manufacturers force users to change the device credentials upon setup. The very famous Mirai botnet used this vulnerability to create a botnet of IoT devices controlled to do specific tasks [2]. In SMEs, the IoT field being addressed in this project, a lot of IoT devices are being neglected, either because of their huge number or other reasons. Moreover, SMEs need cost-effective security measures that can be done by a single network admin, without the need to hire a full cybersecurity team.

In the available scanning tools, attacks are using a list of credentials to try with, namely a dictionary attack, while others brute force every possible combination. Using these methods increase the number of login attempts, leading to several mitigation techniques used by the IoT manufacturer or the network itself to stop the attack. The IoT device may have a limited number of attempts in which it requires the user to wait for a specific period of time after a number of failed attempts. For example, after 3 failed attempts, the user should wait for a minute before allowing him to try again. This period of time increases for each failed attempt onwards. Another method that maybe implemented in the network firewall is IP blockage. IP address of the user trying to login can be blocked from accessing the network after a number of failed login attempts. The main issue is with the number of attempts used for the attack. Limiting or reducing the number of combinations to use for the attack would solve the case.

In this research, we develop a tool that is easy to be used by the network administrator to scan the network for IoT devices and checks the vulnerability of default credentials. The tool automates the process of checking what ports are open on the IoT device and attempting to log in with multiple combinations of default credentials custom to the device's brand or vendor, limiting the number of trials. Default credentials specific to the device's vendor and used for the login attempts for several ports will be fetched automatically by the tool using the assistance of generative AI. It will robustly help in removing the IP or time blockage issue. The full automation the tool provides lets network administrators assess their networks with ease and gives very clear information about which devices are still on default credentials as well as solves the limitation in the number of trials addressed. The tool alerts the user of any devices that are still configured with default credentials and on what port the attack succeeded. On top of that, the tool shows safe devices that are not set with default credentials.

Generative Artificial Intelligence (AI) is the artificial intelligence technology that generates content after training on a huge dataset [3]. The project uses generative AI to analyse the target IoT device and gives output that helps in narrowing down the list used for the attack procedure. The reason behind using AI for credentials retrieval and not using a dictionary attack or a normal brute force is because of the enormous credentials list used for the attempts these methods use, causing the mentioned blockage problems. AI will search the internet for availably online information about IoT devices such as support pages, digital manuals, and troubleshooting blogs to extract as much information as possible useful for the attack. The main element that AI will be searching for is the set of credentials used by the target IoT device vendor across all of their devices' line or model. To achieve this, a prompt needs to be well engineered to use the full power of generative AI and to get the best results.

II. Literature Review

Wi-Fi routers, access points, and IP cameras are very crucial devices in any office or organisation. Therefore, researchers did excessive security studies on them. Romana et al. created a category for Small Office/Home Office routers called SOHO and discussed the vulnerabilities of these routers. These SOHO routers are vulnerable because the security configuration of these devices is mostly left on the default settings, which leads to the low-security posture of all IoT devices connected to the routers. The researchers statically and dynamically analysed the security of a Netis WF2411 router. The first step of this analysis was to collect information about the device such as firmware version, patches or updates, and publicly reported issues. Moreover, network interfaces of the device are noted such as WAN, LAN, and Wi-Fi. Later the dynamic analysis with a live device starts by finding open ports and performing VAPT on the network interfaces. Dynamic analysis also included network monitoring during several periods. Furthermore, the static analysis is done by retrieving the firmware and unpacking it, then checking its scripts, configuration, and database files. The results showed that the router has the Telnet port open and is still on default credentials and UDP port 53413 is open and vulnerable to remote command execution [4]. Research was also conducted by Rosihan et al. to list and recommend network security measures to be implemented on MikroTik routers, which are routers equipped with a MikroTik operating system that holds numerous network functions and can be used as a router or firewall. The resulting recommendations came from the vulnerability assessment that was done by the researchers on the routers. Assessing the routers included several penetration testing methods such as DDoS, brute forcing, and exploiting. The stages of this research are gathering information about the target as a first step. Then scan the device for open ports using the Nmap tool. Then the device is attacked based on the scan results. Lastly, the analysis is only done if the attacker gains access to the device. Using Routersploit, the device was attacked with an FTP credentials brute force on port 21 and was successful using MikroTik default credentials [5]. Another experiment was done by Perone et al. where a Python script was made to test the security of IP cameras, it was estimated in 2017 that over 10% of IoT devices are still on default credentials. Leading to the creation of repositories that are filled with IoT default credentials, device make, and model. The very famous Mirai malware is conducted by gaining access to a huge number of IoT devices using these credentials repositories and forcing them to form a botnet to eventually start the DDoS attack, 10% of the IoT devices or bots used in the attack were IP cameras. With a main focus on two European IP camera models, a research proposed a Python script that automates the process of identifying IP cameras protected with default credentials. Using GitHub repositories as a credentials pool and Shodan API to get IP cam details such as open ports, the code mainly focuses on HTTP and checks if the response code is 200. The research results show the success of the script on 2 models, a widely used cheap model, and a high-end security camera [6].

Tab. I summarises previous successful attacks that were done on IoT devices. We can see that the attackers used availably online default credentials of the IoT device brand to construct the attack. Also, the range of IoT devices was from home to enterprise level. Adding to that point, the attacks were successful on different ports. The multiple successes in attacking different kinds of IoT devices by utilising the availability of their default credentials reveals how low the security level of IoT devices is. This makes default credentials vulnerability a very critical one to research and try to find mitigating mechanisms.

Numerous types of attacks were also done on IoT devices using several approaches and in different ports with attempts to create mitigation techniques to protect networks that include IoT devices. As many IoT devices use the SSH protocol, the research focuses on the password authentication method used to authorise SSH connections and how widely it is spread. Performing an internet wide SSH scan and attempting to log in with an empty password attribute returned "Authentication (password) failed", which reveals that millions of devices use password authentication policy for SSH protocol. As per the Censys service, there are about 16 million devices that use

TABLE I: Successful IoT Attacks

Authors	Year	Device	Port	Methodology
Romana et al.	2020	SOHO Routers	23	Default credentials of router vendor.
Rosihan et al.	2022	MikroTik Routers	21	MikroTik default credentials (admin/blank).
Perone et al.	2023	IP Cameras	80	Dictionary attack using GitHub repositories containing default credentials of different IP cameras vendors.

password authentication for SSH. The Zmap test results show that more than 65% of devices connected to the internet can be accessed by SSH using a password authentication procedure [7]. Moreover, another research demonstrates brute-forcing Raspberry Pi because it is used by 90% of IoT devices, to remotely log into their systems using SSH. The research demonstration aims to reduce IoT device resource wastage in the case of a brute force attack by detecting it and mitigating it. The brute force was conducted using the Hydra command with 2 wordlists one containing usernames and the other containing passwords. The process of detection and mitigation proceeds by capturing the network traffic during the brute force attack and extracting information about successful and failed login attempts. The detection is then done by spotting the number of unsuccessful SSH connections. Failed SSH connection attempts are spotted by limiting the number of login attempts and by setting a time threshold for guessing the password period (password input step). If a brute force notice is raised, the connection is dropped which leads to successful mitigation of the attack. This procedure was fed into an IDS and after the experiment, researchers found out that resource wastage was reduced by 10%, 25%, and 40% among different resources [8].

Weak and default credentials is a very critical vulnerability and research was done in this aspect. The objective of the research [9] is to present a new authentication mechanism to secure Telnet or SSH connections for IoT devices. The weak credentials pairs that come default from the manufacturer open the door for large-scale DDoS attacks on IoT devices in which they are converted to bots after the capture. Shah et al. proposed a procedure called "login puzzle", a version of the previous client puzzle that forces the user to solve puzzles before reaching the login attempt phase, thus raising the complexity of the login procedure. The difficulty of the puzzle is increased, linearly or exponentially per failed login attempt. They classified the login attempts into 3 states, each differing in the complexity incrementing method of the puzzle. Normal attempt where a user solves the puzzles and then logs in successfully. If the user fails to log in after the successful puzzle solution, the puzzle complexity will increase exponentially. The second state was called the midway give-up attempt where the user stops in the middle of the puzzle-solving phase without reaching the login attempt phase. In this case, the puzzle complexity will increment linearly. Lastly, parallel login attempts state is when the user requests a new login while still in a current login request. Firstly, the new log in request has the same complexity as the still-going previous request. If the previous request fails to log in, the difficulty of the puzzle will be doubled [9].

The bachelor's thesis report aims to examine the existence of default credentials vulnerability in IoT devices used in Sweden such as remote controls, NVRs, and controllers. The research was conducted to check whether these IoT devices can be used to create a botnet. Kim Quach collected details from devices' user manuals and categorised the security mechanism into 4 levels. A total of 273 devices were analysed (excluding routers) and categorised into the levels. Starting from the most secure at level one where the device has other authentication methods than credentials, 127 devices are in this category. Devices with default credentials that force the consumer to change them are at level two, no devices were into this level. 13 devices in level three which focuses on devices that used default credentials with instructions to change them. Level four, the riskiest is where the device is on default username and password without instructions to change them, this category had 7 devices in it [10].

Alladi et al. described common IoT attacks which consumers faced with suggestions for protecting and securing methods. These IoT devices include smart meters, garage door openers, electric vehicle chargers and others. The type of attack in EV chargers is device software failure where the researchers showed how the authentication mechanism can be bypassed in the debug mode. After the bypass procedure, a root user can be created, and the attacker can manipulate the operating system. To countermeasure this vulnerability, the paper suggested changing the string function to a function that accepts limited string length. Node tampering attack was done on the smart meter. The attackers spoofed the Device ID, found stored in the EEPROM, and changed it to another meter's ID. In the research, adding PUFs to the EEPROM was suggested to protect the data and limit read or write access in the EEPROM. A social engineering and brute forcing attack was done on the garage door opener by the researchers. The garage door opener didn't force a difficult password policy making it an easy target. Moreover, the researchers mentioned that packets in transmission were easy to manipulate to open and close doors. The usage of two-factor authentication and a strong password-setting policy are some mitigation techniques suggested by the paper [11].

Another research was done on IoT devices in home networks, Kumar et al. analysed 83 million different IoT devices in homes, these include internet-connected televisions, security cameras, and routers. The main objective of the research was to look at the security postures of these devices, with a focus on their open ports, default credentials, and whether they are vulnerable to famous attacks. Based on the coordination between the research team and Avast Antivirus software, more than 50% of IoT devices in the Middle East still supported FTP with weak credentials. The analysis output was that 88% of IoT devices that still support FTP protocol use admin/admin

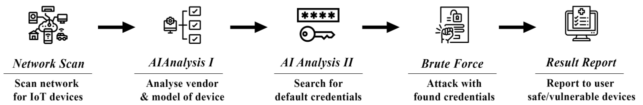

Fig. 1: Proposed Solution Phases

as username and password combination. The same credentials combination was used in 36% of IoT devices that have Telnet service open [12].

To conclude, the different studies presented outline how vulnerable and weak IoT devices are, and that this new technology, despite its widespread, still needs a tremendous amount of improvement in the security aspects. The huge variety of IoT devices and the different functionalities they offer make securing them a serious challenge. Manufacturers, organisations, regulation publishers, and users should coordinate to increase the security of IoT devices. More research efforts like exploring weaknesses, proposing enforcement mechanisms such as the "login puzzle", and developing assessment frameworks need to be done, and this project will hopefully contribute to creating more secure IoT networks.

III. Proposed Solution

The proposed solution is created by developing a script that goes through five phases, scanning, analysing, retrieving then attacking and reporting. The first phase scans through the given network for IoT devices and detects specific open ports. Secondly for each IoT device, the solution will detect the brand, make, or model of the IoT device which will be used for the analysis to output default credentials that the model or brand uses the most. The detection will be done using the help of generative AI that will scan and analyse the available web or login page on ports 80 or 443. A second non-AI scan will be used for devices without web or login pages or have ports 80 or 443 closed. Phase three of the script will use the outputs from the analysis to search for the default credentials. The fourth phase brute forces the scanned devices using the credentials from the third phase. Moreover, the solution will take note of successful and unsuccessful login attempts, this information will be used in the last phase. These steps will compress the number of login attempts and will avoid the blockage occurred when rate limiting is used by IoT devices. The method proposed will reduce the size of the dictionary being used in the attack and thus decrease the detection and mitigation probability. The script will show which combination of username and password returned a successful login attempt and report it to the user.

Fig. 1 gives an overview of the main flow of the solution. The solution starts by scanning the network, analysing twice using AI, brute forcing, and finally, reporting the results to the user. The details of each phase is discussed in this paper.

The flow chart in Fig. 2 shows the steps of the mentioned code phases. The chart illustrates the flow of functions of the tool starting with getting the range of IP addresses for the scan from the user. The rectangle denotes a function whereas the diamond represents an if function with Boolean outputs. The connection between endpoints and the line at the start of the diagram means the looping functionality of the script. The script loops and performs the functions on each device on the network. We can notice that on the right side of the diagram, functions that should be performed on devices with ports 80 and/or 443 are open, such as the web page analysis and the brand search. While on the left side, the diagram illustrates functions to be done only on devices with ports 80 and/or 443 is closed, where web page is not available.

A. Phase 1: Network Scanning

The first step in phase one is initializing the scanner and getting the IP address or IP address range that will be used for the penetration testing from the user. Then the script will start to scan the network for devices that have specific ports open. These ports are 20/21, 22, 23, 80/443.

- Ports 20 or 21: FTP runs on these ports. IoT devices use these ports to transfer files between other IoT devices or connected services.
- Port 22: IoT devices use this port for SSH service, allowing remote connectivity and command execution.
- Port 23: Telnet occupies this port and has similar functionality to SSH but in a less secure and unencrypted method.
- Ports 80 or 443: HTTP works on 80 and the HTTPS works on 443, allowing for web servers to be communicated with to load web pages. This is mostly used if the IoT device has a web-based control panel that can be accessed by requesting the device's IP address.

Since some networks block network scanning, the script uses a different randomised IP address in the scanning functions, overcoming the IP blockage. The scan will result in a two-dimensional list, the main list contains the IP addresses of the devices, and inside of that list, the sub-list contains what ports are open from the specified ports.

B. Phase 2: AI Analysis and Vendor Detection

Phase two of the solution detects the IoT device's vendor and uses it to search for default credentials. The vendor detection is done using two methods. Based on the method used, the usage of generative AI will increase or decrease. The script reasoning behind choosing either method is based on the availability of a web page on ports 80 or 443. The first method, which is an AI-based method, is chosen when ports 80 or 443 are open on the device and have a web page running on one of these ports. The second method, which is a less AI-based method, is chosen when ports 80 and 443 are closed. The detailed working of each method is discussed in the following. In the case where port 80 or 443 is open and has a web page available, the code will input the HTML code of that web page to the AI module with a prompt that asks it to analyse the HTML code and find what brand or model is that IoT device. In the second method where ports 80 and 443 are closed, the script uses a normal header or banner scan to

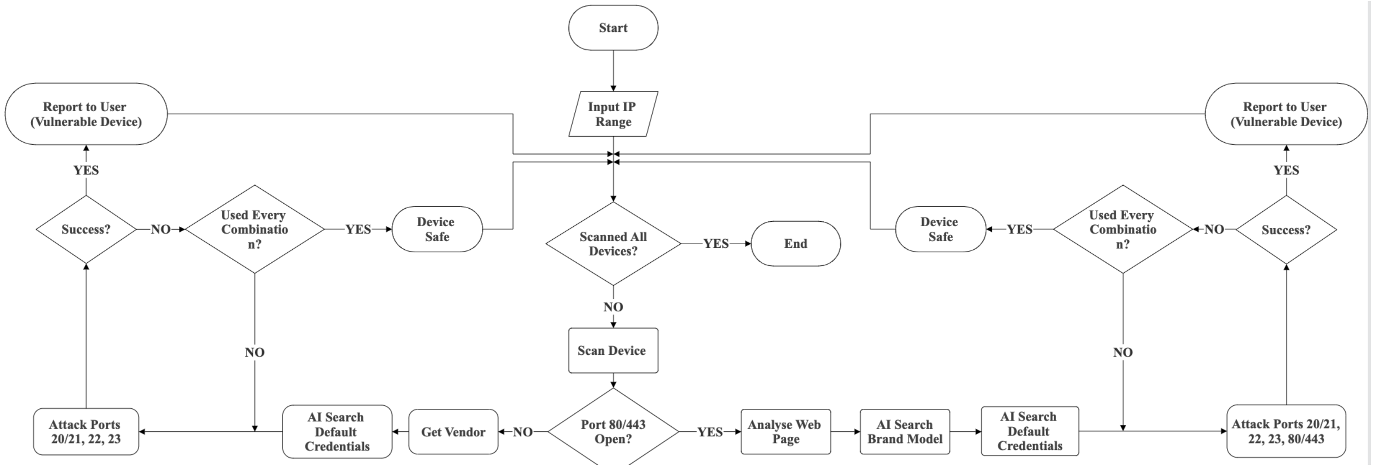

Fig. 2: Script Flow Chart

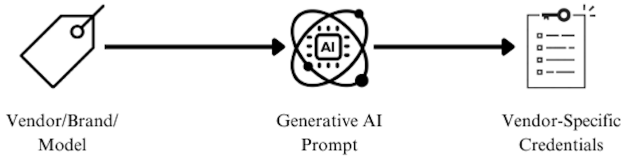

Fig. 3: Phase 3 Flow

get the vendor of the IoT device. This method outputs fewer specific results but uses no AI resources. After the vendor detection using either method, the script parses and arranges the unstructured output of generative AI and creates variables for the results to be used in the next phase which is searching for default credentials.

C. Phase 3: Default Credentials Retrieval

In this phase, the script takes the parsed output from phase 2 as attributes and searches for the default credentials of the detected vendor or brand. Using the attributes from phase two, a prompt is constructed and passed to the generative AI module to output the list of the most used username/password combinations of that vendor. Again, the unstructured data is parsed and prepared for the brute force function in the next phase. Fig. 3 demonstrates the inputs to the generative AI function and the output.

D. Phase 4: Brute Force Attack

The output from phase three will be used to brute force the specified ports. This phase will also be divided into two types. Attacking a login page on ports 80 or 443 is the first one and attacking other ports is the second. It is divided because the login structure or the command used for the login attempt in the script is different between these ports. To illustrate, a login page running on ports 80 or 443 is different from one device to another, with different HTML structures handling the login form. The username login form HTML selector can be named “username”, “Username”, or “user_name” and the password login form HTML selector can be named “password”, “Password”, or “PassWord”. After the correct type is chosen and the brute force command is formulated, the attack begins the brute force, trying to log in with each username/password combination from the list generated using AI.

In this phase, Hydra will be used as the brute forcing tool. Hydra uses a word list for the username field, another word list for the password field, a method, and the options the method needs. In the case of HTTP, Hydra needs the HTTP GET or POST method and the HTML selectors attributes for the username and password field. This to ensure that the parameters entered correctly for the login trial.

E. Phase 5: Reporting

At the end of the script, the tool will present the scan details such as the devices scanned with their IP addresses, what brand or vendor, what ports are open, the login form selectors (if applicable), the top 3 or 4 username/password combination, and most importantly, if the device is still on default credentials or not. Succeeding in logging in using the AI-fetched credentials denotes that the IoT device is still configured with default credentials. The code reports an attention message to the user with the credentials combination that succeeded in logging in. On the contrary, the failure in logging in using all combinations denotes that the device is configured with credentials different from the generated list. Safe devices will also be reported to the user.

F. HTML Selectors

Searching for the correct HTML selectors is crucial for the attack success. Since different login forms use different naming methods for the fields, making the tool general purpose across a wide range of IoT devices is difficult. A mechanism of automatically detecting HTML selectors should be implemented. This can be done using some HTML parsing libraries or tools.

TABLE II: OpenAI API Cost per Scanned Device

Function	Number of Tokens Needed	GPT Model	Amount per 1K Tokens	Total Amount
HTML Content Analysis	1,500	GPT-4 Turbo	$0.03	$0.045
Credentials Search	500	GPT-4 Turbo	$0.03	$0.015
			Total	$0.06

G. False Positives

Despite the wide range of IoT device configurations and the accompanying challenge of different HTML login forms, the script does the HTML content scraping to get the appropriate form field selectors. This function, as mentioned, increases the number of IoT devices the tool can be used for without hardcoding the attack command for each device. It is important to mention that the script may fail to get all required login attributes and thus produce false positives. These false positives are wrong login attempt results, for example, the code may produce multiple successful attempts with different combinations of username and password. This will only happen in the case where the IoT device has a login page.

To put the user into perspective, a detection mechanism is implemented where the script detects that the command is resulting in more than one successful attempt when assessing that device. Therefore, the code notifies the user that the tool is producing false positives for that specific device.

H. Cost Analysis

For the generative AI functionality, the script uses an API from OpenAI. The reasoning model implemented in the tool is GTP-4 Turbo, the newest (as of the date of writing this paper) and the most accurate AI model OpenAI offers. The way it works is by installing the OpenAI library from "pip", which is a Python package manager. Then we need to import the library to our code and add the API key that is generated from OpenAI's account dashboard.

Using OpenAI API and its GPT-4 Turbo model is not free, it is pricing mechanism is based on the number of tokens utilised. A token is counted as a word or a very few words. As of the date of this report, the cost of requesting 1,000 tokens of GPT-4 Turbo is $0.03 [13]. The tool uses the API at most twice, firstly to analyse the HTML code of the web page to detect brand and model, and secondly to search for default credentials. Tab. II shows, for a single scanned device, a rough estimate of the number of AI tokens used per function, and how much they cost each.

This initial experiment was done throught he implementation of OpenAI GPT-4 API. We must mention that the tool can be accompanied with any type of LLM and not exclusive to OpenAI. Any other LLM can be used by implementing its API or even a local LLM can be used for a more cost effictive approach.

IV. Conclusion and Future Works

With the immense increase of IoT usage by small organizations and large corporations, securing them becomes rather requisite. The analysis done on tools and related works shows the severity of the addressed issue. Thus, resulting in a literature gap for a simple and automated tool that simple personnel can use to monitor security posture of IoT devices in hand. The tool proposed simply marks vulnerable IoT devices on the network without kicking the tester out of network or blocking him. The proposed idea ensures classifying vulnerable IoT device connected to the network, making it simple for SMEs and cost-effective. The usage of Generative AI reduces the combinations used for login trials and makes the whole process more efficient by searching for vendor specific credentials. The resulting small list of credentials will then be used to attack the IoT device. Successful attacks means that the device is still on default credentials and this must be reported to the responsible staff immediately. The next step would firstly be implementing the solution then testing it in a real-world network of different organizations and corporates. Based on the test result, the solution can be improved and enhanced. Another area where this tool can be improved in is the integration of machine learning to firstly parse the HTML selectors more accurately and secondly to deeply understand the firmware of the IoT device and its configurations to be used in thoroughly assessing its security.

References

[1] "IoT Technologies Explained: History, Examples, Risks & Future." Accessed: May 04, 2024. [Online]. Available: https://www.visionofhumanity.org/what-is-the-internet-of-things/

[2] G. Kambourakis, C. Kolias, and A. Stavrou, "The Mirai botnet and the IoT Zombie Armies," Proceedings - IEEE Military Communications Conference MILCOM, vol. 2017-October, pp. 267–272, Dec. 2017.

[3] S. Feuerriegel, J. Hartmann, C. Janiesch, and P. Zschech, "Generative AI," Business and Information Systems Engineering, vol. 66, no. 1, pp. 111–126, Feb. 2024.

[4] S. Romana, J. Grandhi, and P. R. L. Eswari, "Security Analysis of SOHO Wi-Fi routers," in Proceedings - 2020 International Conference on Software Security and Assurance, ICSSA 2020, Institute of Electrical and Electronics Engineers Inc., 2020, pp. 72–77.

[5] R. R and Y. Muin, "MikroTik Router Vulnerability Testing for Network Vulnerability Evaluation using Penetration Testing Method," Int J Comput Appl, vol. 183, no. 47, pp. 33–37, Jan. 2022.

[6] S. Perone, L. Faramondi, and R. Setola, "Default Credentials Vulnerability: The Case Study of Exposed IP Cams," in Proceedings of the 2023 IEEE International Conference on Cyber Security and Resilience, CSR 2023, Institute of Electrical and Electronics Engineers Inc., 2023, pp. 406–411.

[7] R. Andrews, D. A. Hahn, and A. G. Bardas, "Measuring the Prevalence of the Password Authentication Vulnerability in SSH," IEEE International Conference on Communications, vol. 2020-June, Jun. 2020.

[8] M. M. Raikar and M. S. M, "SSH brute force attack mitigation in Internet of Things (IoT) network : An edge device security measure," 2021.

[9] T. Shah and S. Venkatesan, "A method to secure IoT devices against botnet attacks," in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics), Springer Verlag, 2019, pp. 28–42.

[10] K. Quach, "DEFAULT USERNAME AND PASSWORD IN INTERNET OF THINGS," 2018.

[11] T. Alladi, V. Chamola, B. Sikdar, and K. K. R. Choo, "Consumer IoT: Security Vulnerability Case Studies and Solutions," IEEE Consumer Electronics Magazine, vol. 9, no. 2, pp. 17–25, Mar. 2020.

[12] D. Kumar et al., “All Things Considered: An Analysis of IoT Devices on Home Networks”, Accessed: Apr. 21, 2024. [Online]. Available: https://www.usenix.org/conference/usenixsecurity19/presentation/kumardeepak

[13] “Pricing — OpenAI.” Accessed: May 11, 2024. [Online]. Available: https://openai.com/api/pricing/