

UNIVERSITATEA DIN BUCUREȘTI

FACULTATEA DE MATEMATICĂ ȘI INFORMATICĂ

SECURITY AND APPLIED LOGIC

LUCRARE DE DISERTAȚIE

Python Web Application Scanner

COORDONATOR ȘTIINȚIFIC: ABSOLVENT:

NĂIBOIU TEODOR

BUCUREȘTI, SEPTEMBRIE 2024

**Contents**

[I. Introduction 4](#_Toc173508260)

[I.1. Motivation 4](#_Toc173508261)

[I.2. Personal contribution 5](#_Toc173508262)

[I.3. Structure 5](#_Toc173508263)

[II. Technologies 6](#_Toc173508264)

[II.1. Python 6](#_Toc173508265)

[II.2. Requests 6](#_Toc173508266)

[II.3. Urllib 7](#_Toc173508267)

[II.3.1 Urllib.parse 7](#_Toc173508268)

[II.3.2 Urllib.requests 7](#_Toc173508269)

[II.4. BeautifulSoup 8](#_Toc173508270)

[II.5. Threading 8](#_Toc173508271)

[II.6. OS 8](#_Toc173508272)

[II.7. ConfigParser 8](#_Toc173508273)

[II.8. RE 9](#_Toc173508274)

[II.9. ArgParse 9](#_Toc173508275)

[II.10. HTML 9](#_Toc173508276)

[II.11. JINJA2 10](#_Toc173508277)

[II.12. DVWA 10](#_Toc173508278)

[II.13. Bee-Box 10](#_Toc173508279)

[II.14. Other technologies 11](#_Toc173508280)

**Abstract**

The following project represents a scanner software application written in Python programming language, designed to aid developers, white hat hackers (security researchers, red-teamers, bounty hunters) and other cybersecurity enthusiast in detecting vulnerabilities and assessing the attack surface and risk exposure of a Web Application. This new implementation is a redesigned and enhanced version of my previous Web Application Vulnerability Scanner software presented in my under-graduate thesis [1]. This new implementation is designed for Linux ***O****perating* ***S****ystem* (OS) usage only. Users can interact with the application through the ***C****ommand* ***L****ine* ***I****nterface* (CLI) and by providing the scanning arguments such as the target website, scan type and login details.

The new scanner functionality has been enhanced by refining more accurate detections and by adding new features such as rogue comment identification, hierarchical site map graph, external ***u****niform* ***r****esource* ***l****ocators* (URLs) references, hidden URL paths and ***C****ommon* ***V****ulnerabilities and* ***E****xposures* (CVE) detection. The reporting method has also been changed to a ***H****ypertext* ***M****arkup* ***L****anguage* (HTML) report. The new version of the scanner has been built for scalability and ease of access, as the number of prerequisite configurations has been decreased compared to the previous version. Additionally, the majority of ‘future enhancements’ mentioned in the previous thesis were implemented in the new version along a completely new method of scanner-website interaction logic that increases reliability and decreases the time of execution.

# Introduction

*Cybersecurity is the art of protecting networks, devices, and data from unauthorized access or criminal use* [2]. This formal definition of cybersecurity can be pictured as a border between two armies of people whose whole purpose is to conquer territory in the other’s lands. Over the last years, the need of understanding and educating people on cyber security has been increasing. Nowadays, not only software engineers and security researchers are impacted daily by digital data polluted with malicious additions, but any other person with a smart phone or any other digital device connected to the internet.

*On average every day, approximately 30000 new websites fall to hacking attacks* [3]. As the numbers of websites is increasing due to demand and digitalization, so are the attacks aimed at stealing data by exploiting misconfigurations, vulnerabilities or by simple social engineering attacks facilitated by lack of access restrictions. Currently, more and more solutions are transitioning to web applications designed to provide the users with a fast resolution for everyday activities (for example: buying a bus ticket, ordering a meal from the local restaurant, tracking your parcels or accessing your bank balance). Due to this reason, the need of having a secure website and educated users is more important than ever.

## Motivation

I have decided to continue my previous implementation of the web application vulnerability scanner and to enhance its functionality due to my ongoing passion for web application security and for my personal projects. *A web application is a computer program stored on a remote server and run by its users via a web browser* [4]. *A web browser is a computer program used for accessing sites or information on a network (such as the World Wide Web)* [5]. As the web application is designed to be publicly accessible, attackers have an easy target available to lock on their scanners to and wait for low hanging fruits to be found. Due to this reason, I have decided to create a scanner similar with the tools and methods attackers are using to detect misconfigured websites over the internet.

As information is constantly changing, so are software solutions and their configurations. Due to the ongoing modifications in web application service configurations and their increased complexity, researchers and red-teamers can have difficulties in detecting vulnerabilities in a timely manner. This also causes automated vulnerability detection tools to become deprecated really fast. As my project is an automated detection tools as well, I have decided to further continue its development and update the detection capabilities to keep up the latest security risks while also having scalability in mind. In the following pages, I will present the new features of this tool, as well as the implementation methods I have chosen for ensuring future updates are done in a timely manner.

## Personal contribution

While the previous version was much more focused on the ***O****pen* ***W****orldwide* ***A****pplication* ***S****ecurity* ***P****roject* (OWASP) Top 10 Standards [6], the new version is based on my work obtained experience as a Security Analyst and on ***C****apture* ***T****he* ***F****lag* (CTF) competitions experience and observations, over the last 3 years.

*CTFs are gamified competitive cybersecurity events that are based on different challenges or aspects of information security* [7]. The focus was shifted from a OWASP Top 10 approach to a more subjective means of detections based on my past observations from live models of vulnerable web applications. Almost all of the detection methods and capabilities were implemented based on my best judgement instead of generic approaches. This was done to further increase the accuracy of detections and to decrease the false-negative findings. Due to this approach, there might be a significant number of false-positive findings as well, hence the scanner results should be confirmed by manual replication.

Furthermore, I have decided to bring this scanner up to a competitive level by lowering the number of prerequisite configurations needed and by providing a human readable report with dynamic and intuitive ***U****ser* ***I****nterface* (UI).

## Structure

TBD after capthers

# Technologies

Compared to the previous scanner version which was running Python version 3.2, the new version was written in Python version 3.9 [8]. Python is one of the best programming languages to use for automated sequential or parallel instructions, since it is easy to learn, to read and to understand. Additionally, what makes Python great for automations is the big number of modules and libraries available and easily compatible with numerous software applicability (web and desktop applications, data structures and analysis, scripts, etc.).

For this project, the most important modules and components used are the ones used to communicate with the website, such as *beautifulsoup* [9], *requests* [10], *urllib* [11]. In the following pages I will be describing the modules among other technologies and solutions used to build, run and test the new version of the scanner.

## Python

*Python is an interpreted, object-oriented, high-level programming language with dynamic semantics* [12]. Due to the great, easy to read and write syntax, Python is widely used for complex automated actions among cybersecurity enthusiasts. In projects similar with this one, Python is the go-to choice of any individual interested in ease of use and versatility. Web applications are constantly changing their structure and code, due to this reason, automated scanning tools need to be implemented in easy to maintain solutions that can handle large amount of data and also be compatible with other tools and technologies [13].

## Requests

The *requests* module is used for transferring data from a client (*Web Browser*) to a *Web Server* using the ***H****yper****t****ext* ***T****ransfer* ***P****rotocol* (HTTP) protocol and ***U****niform* ***R****esource* ***L****ocators* (URLs) [10]. There are a total of nine requests type (*GET, HEAD, POST, PUT, DELETE, CONNECT, OPTIONS, TRACE, PATCH*). During the development of this project, I have used four types of requests:

**GET** – This is one of the principal types of requests as it represents the action of requesting data from the server. This request type is used for receiving data according to the provided parameters.

**POST** – Another principal request type method used to send data to the server. This method can be used for creating new data on the server and for updating already existing data.

**PUT** – Method used for creating or changing the representation of a resource according to the data provided in the request payload.

**TRACE** – Method generally used for debugging purposes. This method provides information on the path the requests take to the target destination.

The *requests* module is not only used for exchanging data with the server, but is also used for creating the session to maintain persistent communication with the Web Server during the testing procedures. This module also provides custom request creation functions and HTTP error handling methods.

## Urllib

Urllib module is used generally for URL manipulation [11]. From this module, I have used two functionalities:

### Urllib.parse

The *.parse* module from *Urllib* is used for combining strings back into URL format. In this project, the module has been used for combining actions, extracting certain strings of interest and fragment parts of URLs into variables [14].

### Urllib.requests

The .requests module from Urllib is similar with the *requests* module. The difference between the two is that this module provides some more advanced HTTP actions. In this project, this module was used for opening and reading URLs [15].

## BeautifulSoup

One of the most popular modules used for ***H****yper****t****ext* ***M****arkup* ***L****angua*ge (HTML), ***E****xtensible* ***M****arkup* ***L****anguage* (XML) and ***D****ocument* ***O****bject* ***M****odel* (DOM) of a page. It provides a wide array of functions that help with pulling data from web pages, navigation, modifying and parsing different values available on a website [9].

## Threading

Threading is used for changing the flow of execution from a sequential form to a parallel form by creating a separate flow of execution that can run simultaneously. Threading is useful for applications that handle large number of independent instructions and in which execution time is important. The CPU is used for allocating divisions of the physical core into virtual components or codes, these virtual components are called *threads* [16].

## OS

The ***O****perating* ***S****ystem* (OS) module is used any interaction with the operating system. Actions such as creating, reading, modifying, opening and closing files and directories are performed through the usage of this module [17]. This module can also be used for more advanced interactions with the OS such as accessing environments variables, process information, managing registries and other system related functions.

## ConfigParser

Configuration files are a necessary part of an application that interacts with multiple pieces of data and complex logic. Configuration files are code structures with the *.ini* file extension that can be easily modified by experienced and beginner users without the need of any application architecture knowledge. A configuration file contains custom variables that are usually depended on the execution environment or purpose of the application [18].

The *ConfigParser* module is responsible with the interaction between the configuration file and the application Python code. This module is used to access and import the configuration values in the code and execution logic.

## RE

***R****egular* ***E****xpressions* (RE or Regex) are recognition rules based on symbols that form search patterns to detect any certain characters or string of characters inside other strings based on that searching rule [19]. Once a pattern is defined, a string can be manipulated in any way the user wants, those include edits, deletions, word substitutions, character(s) extractions from any string of characters that contains that pattern.

The RE module is the Python implementation of Regex functionality.

## ArgParse

ArgParse is a module that is used for Python applications that require user interaction through ***C****ommand* ***L****ine* ***I****nterface* (CLI). ArgParse is the link between *sys.argv* (the list of commands passed to the script using the CLI) and the Python code [20]. The advantage of using ArgParse is that it provides a user-friendly CLI while also automatically generating the *–help* menu used for displaying all the available arguments. From a programming point of view, this module also provides an easier method to manipulate the arguments passed via CLI into the code logic.

## HTML

This is a simple module used for manipulating HTML entities. The submodules used from HTML are the modules that *escape* and *unescape* characters. An *escaped* *character* is a character that uses an alternative interpretation of the same character [21]. For Web Application scanning, certain characters need to be escaped when a set of data is sent to the website and others need to be unescaped when data is received.

## JINJA2

The Jinja2 module is responsible for HTML template creation and manipulation through Python code [22]. This module is used for creating the final HTML report file based on an already created template which serves as a starting point for the report output data and format. All scanning results are parsed through Jinja2 to the templating functions that handle the data. The data is then added to the report, through Jinja2 methods, along with the HTML tags required for the HTML page to be correctly interpreted and displayed.

## DVWA

***D****amn* ***V****ulnerable* ***W****eb* ***A****pplication* (DVWA) is a PHP/MySQL web application that is left intentionally vulnerable with the main goal of helping security enthusiasts, web developers, students and teachers to use test their tools, practice their cybersecurity skills and understand web application security better, in a controlled, isolated and legal environment [23].

This application is one of the two web servers used in this project for testing the scanner and simulating real life scenarios. The web server contains numerous different vulnerabilities and misconfigurations which allow a comprehensive coverage of the most frequently found vulnerabilities in the wild. This application was hosted in a controlled virtual environment using the Linux Distribution – Kali Linux.

## Bee-Box

Bee-Box or BWapp is also a PHP/MySQL web application left intentionally vulnerable that follows the same purpose as DVWA. Bee-Box contains over one hundred web vulnerabilities and it covers all the major known web weaknesses and bugs along with the ones from OWASP Top 10 project [24].

This is the second web server used for testing the scanner progress and detection rate in this project. This web server was hosted in a controlled virtual environment using the Linux Distribution – Ubuntu.

## Other technologies

Below are some of the less used but still important modules that were used for implementing the web application scanner.

The *random* module is used for generating pseudo-random numbers or choices for various purposes [25]. In this project, the module was used for choosing a pseudo-random web user agent from a list. A web user agent is a characteristic string of data that helps servers identify the application, operating system, vendor of the requesting source [26].

The *datetime* module is used for accessing the calendar’s date and time. Using this module, I have provided date and time details in the resulting report with the format *YYYY:MM:DD HH:MM:SS* [27]*.*

The *math* module provides access to mathematical operations and functions [28]. In this project, the module was used to perform rounding on list length divisions used to split the data for threading purposes.

The *colorama* module is used to color the output text to terminal [29]. In this project the module was used to color the output text when the error file is not available due to any reason.

The *json* module is used to import **J**ava**S**cript **O**bject **N**otation (JSON) data interpretation. JSON is a standardized format for data that can be easily interpreted and read both by humans and computers [30]. In this project, the *json* module was used for creating and manipulating the Website Map URLs Graph.

The *warning* module provides warning control functions. Warnings are issued in situations in which the program does not raise a condition that terminates the program, but the user should be aware of the event. In this project, the *warning* module was used only to suppress character encoding warnings for HTML received data.

**References**

|  |  |
| --- | --- |
| [1] | T. Naiboiu, Web Application Vulnerability Scanner, Bucharest: University Of Bucharest, 2021. |
| [2] | CISA, 1 February 2021. [Online]. Available: https://www.cisa.gov/news-events/news/what-cybersecurity. [Accessed 1 August 2024]. |
| [3] | BUGTI, "Website Hacking Statistics 2024," BUGTI, 07 05 2024. [Online]. Available: https://www.bugtiwallet.com/website-hacking-statistics-2024/. [Accessed 1 August 2024]. |
| [4] | A. Volle, "Web application," [Online]. Available: https://www.britannica.com/topic/Web-application. [Accessed 1 August 2024]. |
| [5] | W. I. a. W. Browser?. [Online]. Available: https://www.lifewire.com/what-is-a-browser-446234. [Accessed 1 August 2024]. |
| [6] | "OWASP," [Online]. Available: https://owasp.org/www-project-top-ten/. [Accessed 1 August 2024]. |
| [7] | K. Crawley, 10 06 2021. [Online]. Available: https://www.hackthebox.com/blog/what-is-ctf. [Accessed 1 August 2024]. |
| [8] | "What is Python?," [Online]. Available: https://www.qamadness.com/knowledge-base/python-for-automation-why-is-it-great-and-how-to-use-it/. [Accessed 1 August 2024]. |
| [9] | "Beautiful Soup Documentation," [Online]. Available: https://beautiful-soup-4.readthedocs.io/en/latest/. [Accessed 1 August 2024]. |
| [10] | "Requests: HTTP for Humans," [Online]. Available: https://docs.python-requests.org/en/latest/index.html. [Accessed 1 August 2024]. |
| [11] | "URL handling modules," [Online]. Available: https://docs.python.org/3/library/urllib.html. [Accessed 1 August 2024]. |
| [12] | "What is Python?," [Online]. Available: https://www.python.org/doc/essays/blurb/. [Accessed 1 August 2024]. |
| [13] | "Why is Python Important for Cybersecurity?," [Online]. Available: https://www.scaler.com/topics/cyber-security/cybersecurity-importance-for-python/. [Accessed 1 August 2024]. |
| [14] | "urllib.parse — Parse URLs into components," [Online]. Available: https://docs.python.org/3/library/urllib.parse.html#module-urllib.parse. [Accessed 1 August 2024]. |
| [15] | "urllib.request — Extensible library for opening URLs," [Online]. Available: https://docs.python.org/3/library/urllib.request.html#module-urllib.request. [Accessed 1 August 2024]. |
| [16] | "What are Threads in Computer Processor or CPU?," [Online]. Available: https://www.geeksforgeeks.org/what-are-threads-in-computer-processor-or-cpu/. [Accessed 1 August 2024]. |
| [17] | "os - Miscellaneous operating system interfaces," [Online]. Available: https://docs.python.org/3/library/os.html. [Accessed 1 August 2024]. |
| [18] | "configparser - Configuration file parser," [Online]. Available: https://docs.python.org/3/library/configparser.html. [Accessed 1 August 2024]. |
| [19] | "A Practical Guide to Regular Expressions," [Online]. Available: https://www.freecodecamp.org/news/practical-regex-guide-with-real-life-examples/. [Accessed 1 August 2024]. |
| [20] | "argparse — Parser for command-line options, arguments and sub-commands," [Online]. Available: https://docs.python.org/3/library/argparse.html. [Accessed 1 August 2024]. |
| [21] | "Escape sequences," [Online]. Available: https://en.cppreference.com/w/cpp/language/escape. [Accessed 1 August 2024]. |
| [22] | "Jinja," [Online]. Available: https://jinja.palletsprojects.com/en/2.10.x/. [Accessed 1 August 2024]. |
| [23] | "DVWA," [Online]. Available: https://www.kali.org/tools/dvwa/. [Accessed 1 August 2024]. |
| [24] | "BWapp - An extremely buggy web app!," [Online]. Available: http://itsecgames.com/. [Accessed 1 August 2024]. |
| [25] | "random - Generate pseudo-random numbers," [Online]. Available: https://docs.python.org/3/library/random.html. [Accessed 1 August 2024]. |
| [26] | "User-Agent," [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/HTTP/Headers/User-Agent. [Accessed 1 August 2024]. |
| [27] | "datetime — Basic date and time types¶," [Online]. Available: https://docs.python.org/3/library/datetime.html. [Accessed 1 August 2024]. |
| [28] | "math — Mathematical functions," [Online]. Available: https://docs.python.org/3/library/math.html. [Accessed 1 August 2024]. |
| [29] | "colorama," [Online]. Available: https://pypi.org/project/colorama/. [Accessed 1 August 2024]. |
| [30] | "What Is JSON and How Do You Use It?," [Online]. Available: https://www.howtogeek.com/devops/what-is-json-and-how-do-you-use-it/. [Accessed 1 August 2024]. |