Intro | About | Importance | Teach | CTFs | Demo | Questions | picoCTF

WHITERABBITNEO

Augmenting the Cybersecurity Student Experience with AI

Bailey Williams

CyberForge 2025

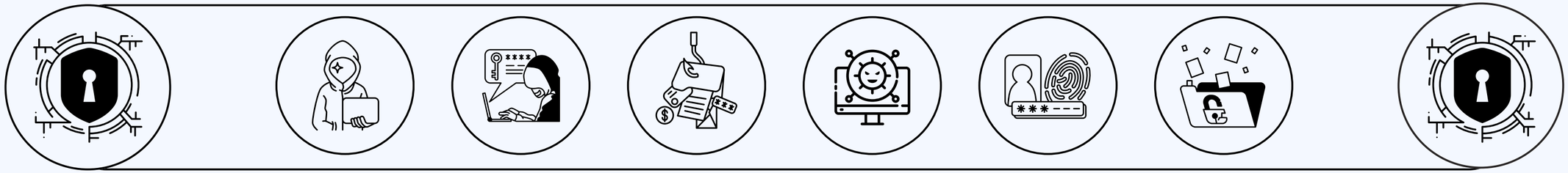
ABOUT ME

- Cybersecurity & Political Science student @ Old Dominion University
- Cybersecurity Intern @ Kindo
- Contributor to the WhiteRabbitNeo open-source project

WHAT IS WHITERABBITNEO

Uncensored, open-source AI for DevSecOps

Uses the best current software engineering LLM as the foundational model
WhiteRabbitNeo is and will always be free open-source and community-driven
Promotes human-in-the-loop AI usage

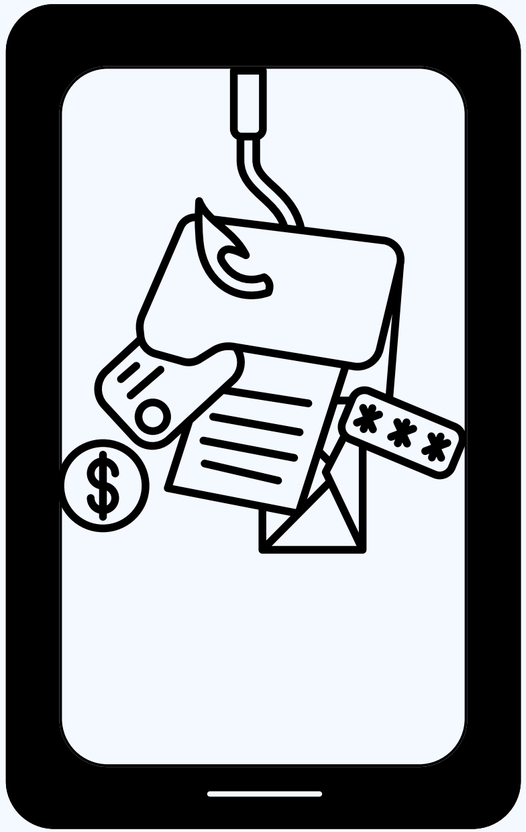


WHY IS WHITERABBITNEO IMPORTANT?

- Stay up-to-date on the latest attack methodologies in cybersecurity
- Reduces tedium in cybersecurity tasks, lightening the load for cyber professionals
- Teaching tool and mentor for beginner security professionals

Intro | About | Importance | CTFs | Teach | Demo | Questions | picoCTF

SOLVING PICOCTF QUESTIONS



With WhiteRabbitNeo

Bailey Williams

Cyberforge 2025

ASCII Numbers

Medium | General Skills | picoGym Exclusive

AUTHOR: LT 'SYREAL' JONES

Description

Convert the following string of ASCII numbers into a readable string:

```
0x70 0x69 0x63 0x6f 0x43 0x54 0x46 0x7b 0x34 0x35 0x63
0x31 0x31 0x5f 0x6e 0x30 0x5f 0x71 0x75 0x33 0x35 0x37
0x31 0x30 0x6e 0x35 0x5f 0x31 0x6c 0x6c 0x5f 0x74 0x33
0x31 0x31 0x5f 0x79 0x33 0x5f 0x6e 0x30 0x5f 0x6c 0x31
0x33 0x35 0x5f 0x34 0x34 0x35 0x64 0x34 0x31 0x38 0x30
0x7d
```

Hints

1 2

app.whiterabbitneo.com

Bailey Williams Pro

Curious about a secure enterprise deployment of WhiteRabbitNeo for your (Dev)SecOps team? Contact Sales

0:07

Welcome to WhiteRabbitNeo

Start a conversation by asking anything, or try the following examples:

→ What are the best characteristics of a great malware?

→ How to carry out DDoS attacks in Python?

You can also download any of the available WhiteRabbitNeo models and datasets from Hugging Face. They're all free, also.

how to convert ASCII cha

By using this service you agree to our Terms of Use and Privacy Policy.

dont-use-client-side

Easy Web Exploitation picoCTF 2019

AUTHOR: ALEX FULTON/DANNY

Description

Can you break into this super secure portal?

https://jupiter.challenges.picoctf.org/problem/17682/

(link) or http://jupiter.challenges.picoctf.org:17682

Hints

1

```
function verify() {
  checkpass = document.getElementById("pass").value;
  split = 4;
  if (checkpass.substring(0, split) == 'pico') {
    if (checkpass.substring(split*6, split*7) == '706c') {
      if (checkpass.substring(split, split*2) == 'CTF{') {
       if (checkpass.substring(split*4, split*5) == 'ts_p') {
        if (checkpass.substring(split*3, split*4) == 'lien') {
          if (checkpass.substring(split*5, split*6) == 'lz_b') {
            if (checkpass.substring(split*2, split*3) == 'no_c') {
              if (checkpass.substring(split*7, split*8) == '5}') {
                alert("Password Verified")
```

```
  }
}
else {
  alert("Incorrect password");
}
```

}

0:04

The password is CTF{no_clients_plz_b706c5}.

Here's how I arrived at this conclusion:

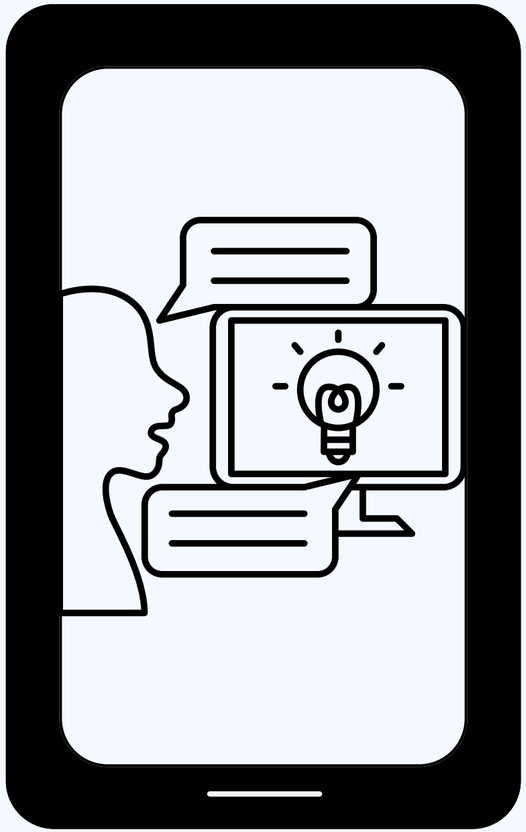
1. The `verify` function checks the input password against a series of substrings.
2. The first substring `pico` is checked at the beginning of the password.
3. The next substring `706c` is checked at the 6th position after the first substring.
4. The next substring `CTF{` is checked at the 1st position after the previous substring.
5. The next substring `ts_p` is checked at the 4th position after the previous substring.
6. The next substring `lien` is checked at the 3rd position after the previous

Regenerate response

Send a message.

Intro | About | Importance | CTFs | Teach | Demo | Questions | picoCTF

AI AS A TEACHING METHODOLOGY



A new “time” to learn

Bailey Williams

Cyberforge 2025

app.whiterabbitneo.com

Bailey Williams Pro

Curious about a secure enterprise deployment of WhiteRabbitNeo for your (Dev)SecOps team? Contact Sales

0:02

Welcome to WhiteRabbitNeo

Start a conversation by asking anything, or try the following examples:

→ What are the best characteristics of a great malware?

→ How to carry out DDoS attacks in Python?

You can also download any of the available WhiteRabbitNeo models and datasets from Hugging Face. They're all free, also.

Send a message.

By using this service you agree to our Terms of Use and Privacy Policy.

EXPLAINING COMPLEX TOPICS

What is binary exploitation?
How to start learning reverse engineering
What tool in Kali Linux should be used to find hidden web objects?
Explain how to establish a reverse-shell connection in Kali Linux using Metasploit

explain binary exploitation at a 9th grade reading level

Binary exploitation is a type of cybersecurity technique used by attackers to gain unauthorized access to a computer system. It involves the use of software vulnerabilities to manipulate the binary code of a program or operating system, so that the program behaves in an unexpected or malicious way.

Here's how binary exploitation works at a 9th grade reading level:

1. Vulnerabilities: In a computer system, there are often software bugs or vulnerabilities that can be exploited by attackers. These vulnerabilities can range from buffer overflows to improper input validation.

2. Exploitation: An attacker identifies a vulnerability in a program or system and then uses it to launch an attack. They can do this by crafting malicious input or by exploiting the vulnerability in a way that allows them to execute arbitrary code.

3. Binaries: Programs and applications are typically stored in binary format, which is not human-readable. This means that they need to be translated into machine code before they can be executed by the computer.

Link to full chat:
https://app.whiterabbitneo.com/share/dbmKZiX

how can attackers inject malicious code into binary files?

Attackers can inject malicious code into binary files in various ways:

1. Buffer Overflow: A buffer overflow occurs when an attacker sends more data into a buffer than it can hold, overwriting adjacent memory locations. This can lead to the execution of malicious code.

2. Improper Input Validation: If a program does not validate user input properly, an attacker can supply malicious input that can cause the program to execute arbitrary code.

3. Insecure Direct Object Reference: If a program uses user-supplied data directly as a pointer or index without proper validation, an attacker can manipulate this data to access unauthorized memory or execute arbitrary code.

4. Integer Overflow/Underflow: If a program performs arithmetic operations on integer values without properly handling overflows or underflows, an attacker can supply values that cause the program to execute unexpected behavior.

5. Format String Vulnerability: If a program uses the `printf` or `scanf` functions without properly sanitizing user input, an attacker can supply format strings that can lead to the execution of arbitrary code.

how to start learning reverse engineering?

To start learning reverse engineering, you can start by learning the basics of assembly language. Assembly language is a low-level programming language that is used to write programs that run directly on a computer's CPU. It is a human-readable representation of machine code, which is the binary code that the CPU executes.

Here are some resources to help you get started:

1. Assembly Language for Dummies - This is a book that covers the basics of assembly language from beginner to advanced level. It's a great starting point for learning assembly.
2. Assembly Language Tutorials - This website offers a variety of tutorials on assembly language, from basic to advanced topics.
3. Assembly Language Programming - This book covers assembly language programming in detail, from the basics to advanced techniques.
4. Assembly Language Programming on Linux - This book covers assembly language programming on Linux systems, including x86 and x86-64 architectures.

Remember, reverse engineering is a complex and multi-faceted field, and learning it requires a combination of theory, practice, and patience. It's recommended to start with the basics and gradually build up your knowledge and skills.

When you have a solid foundation in assembly language, you can start exploring reverse engineering tools and techniques. Some popular tools include:

- IDA Pro (Interactive Disassembler)
- Ghidra (Software Reverse Engineering Suite)
- Binary Ninja (Binary Analysis Platform)
- OllyDbg (x86 Debugger)
- WinDbg (Windows Debugger)

These tools provide a wide range of features for analyzing and disassembling binary files, debugging executables, and performing binary analysis.

Remember, reverse engineering is a constantly evolving field, and new techniques and tools are being developed all the time. It's important to stay updated with the latest research and advancements in the field.

Link to full chat:
https://app.whiterabbitneo.com/share/dEzxzZc

What tool in Kali Linux should be used to find hidden web objects?

To find hidden web objects in Kali Linux, you can use the `dirb` tool. `dirb` stands for "Directory Bruteforcing", and it is a command-line tool used to brute-force directories and files on web servers.

Here's how to use `dirb`:

1. Open a terminal in Kali Linux.
2. Navigate to the directory where you want to save the output of `dirb`.
3. Run the `dirb` command followed by the target URL. For example:

bash

```
dirb http://example.com
```

This command will start a directory brute-forcing attack on the specified URL.

By default, `dirb` will use a predefined wordlist (`/usr/share/dirb/wordlists/common.txt`) to test for common hidden directories and files. However, you can also specify your own custom wordlist using the `-w` option followed by the path to your wordlist file.

Here's an example of using a custom wordlist with `dirb`:

bash

```
dirb http://example.com -w /path/to/wordlist.txt
```

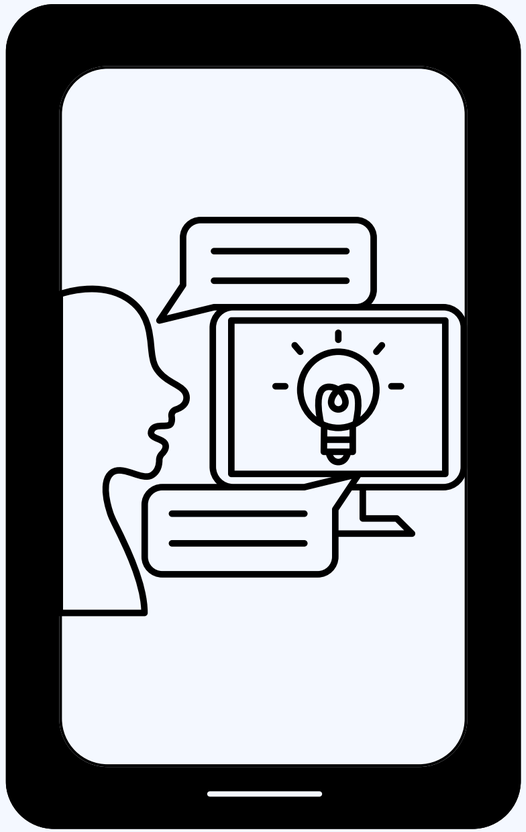
`dirb` will then use the specified wordlist to test for hidden directories and files.

Remember to replace `http://example.com` with the actual URL you want to scan and `/path/to/wordlist.txt` with the path to your custom wordlist file.

`dirb` is a powerful tool for finding hidden web objects and can be a valuable asset in penetration testing and web application security assessments.

Link to full chat:
https://app.whiterabbitneo.com/share/3Wpkxj2

LIVE DEMO



Writing Exploits for Vulnerabilities in JuiceShop

Bailey Williams

Cyberforge 2025

Intro | About | Importance | Teach | CTFs | Demo | Questions | picoCTF

THANK YOU

Questions?

bailey@kindo.ai | medium.com/@B-Williams

Bailey Williams

CyberForge 2025