AIG Cybersecurity Virtual Internship

Reference: https://www.theforage.com/simulations/aig/cybersecurity-ku1i

Tomás Villaseca C.

Tomas.villaseca.c@gmail.com

linkedin.com/in/tomasvc93/

Table of Contents

Task 1 - Responding to Zero-Day Vulnerability

Task Overview:

CISA has just released an alerts on a new zero-day vulnerability for ApacheLog4j. Research the vulnerability and publish an advisory to affected teams to alert and prevent exploitation.

Will learn:

- How to address a vulnerability that may affect Product Development Staging Environment Infrastructure.

Will do:

- Review some recent publications from the Cybersecurity & Infrastructure Security Agency (CISA).
- Research the reported vulnerability.
- Draft an email to affected teams to alert them of the vulnerability, and explain how to remediate.

Context

You are an **Information Security Analyst** in the Cyber & Information Security Team.

A common task and responsibility of information security analysts is to stay on top of emerging vulnerabilities to make sure that the company can remediate them before an attacker can exploit them.

In this task, you will be asked to review some recent publications from the Cybersecurity & Infrastructure Security Agency (CISA).

- **CISA** = Agency that has the goal of reducing the nation’s exposure to cyber security threats and risks.

After reviewing the publications, you will then need to draft an email to inform the relevant infrastructure owner at AIG of the seriousness of the vulnerability that has been reported.

Instructions

CISA has recently published the following two advisories:

- The first advisory (Log4j), outlines a serious vulnerability in one of the world's most popular logging software.
- The second advisory explores how ransomware has been increasing and is becoming professionalized.

Your task is to respond to the Apache Log4j zero-day vulnerability that was released to the public by advising affected teams of the vulnerability.

1. Conduct your research on the vulnerability using the "CISA Advisory" resources provided above as a starting point.

2. Analyze the "Infrastructure List" below to find out which infrastructure may be affected by the vulnerability, and which team has ownership.

Product Team	**Product Name**	**Team Lead**	**Services Installed**
IT	Workstation Management System	Jane Doe (tech@email.com)	OpenSSH dnsmasq lighttpd
Product Development	Product Development Staging Environment	John Doe (product@email.com)	Dovecot pop3d Apache httpd Log4j Dovecot imapd MiniServ
Marketing	Marketing Analytics Server	Joe Schmoe (marketing@email.com)	Microsoft ftpd Indy httpd Microsoft Windows RPC Microsoft Windows netbios-ssn Microsoft Windows Server 2008 R2 - 2012 microsoft ds
HR	Human Resource Information System	Joe Bloggs (hr@email.com)	OpenSSH Apache httpd rpcbind2-4

Task 1.1 - Research Vulnerability

Conduct your research on the vulnerability using the “CISA Advisory" provided resources as a starting point (remember that the vulnerability in this simulation is considered zero-day).

<u>**First Advisory:**</u>

Apache Log4j Software Library Vulnerabilities:

Apache Log4j = Open-source logging framework for Java applications that is widely used in enterprise software.

- **Log4j** → Allows developers to control log output destinations and formats easily.

CVE-2021-44228 (Log4Shell): Vulnerability that allows an attacker to execute arbitrary code on the target system by exploiting the JNDI (Java naming and directory interface) lookup feature in Log4j.

- Critical Severity → CVSS 10.0
- Affects Log4j versions 2.0-beta9 to 2.14.1
- Allows Remote Code Execution (RCE).
- Exploited by sending a specially crafted string that gets logged, triggering JNDI lookup, potentially loading and executing malicious code from an attacker-controlled server.

CVE-2021-45046: Vulnerability that was discovered after the initial patch for CVE-2021-44228. The first fix was incomplete, allowing attackers to craft malicious input data using a JNDI Lookup pattern, potentially leading to a DoS attack or (in some non-default configurations) remote code execution.

- Critical Severity → CVSS 9.0
- Affects Log4j versions 2.0-beta9 to 2.15.0
- Can lead to Remote Code Execution (RCE) in some non-default configurations.

CVE-2021-45105: Vulnerability that causes uncontrolled recursion in Log4j string substitution evaluation. When processing certain log messages, it can lead to a stack overflow, resulting in DoS.

- High Severity → CVSS 7.5
- Affects Log4j versions 2.0-alpha1 to 2.16.0
- Causes DoS through uncontrolled recursion (infinite loop of evaluations).

Sophisticated cyber threat actors are actively scanning networks to potentially exploit Log4Shell, CVE-2021-45046, and CVE-2021-45105 in vulnerable systems.

- According to public reporting, Log4Shell and CVE-2021-45046 are being actively exploited.

CISA Mitigation guidance of Apache Log4j Vulnerability:

1. **Identify vulnerable assets in your environment**

   a) **Inventory all assets that make use of the Log4j Java Library.**

   - Assume all versions of Java and Log4j are vulnerable and include them in your inventory.
   - Ensure the inventory includes all assets and make sure they include the following information:
     - ➢ Software versions.
     - ➢ Timestamps of when last updated and by whom.
     - ➢ User accounts on the asset with their privilege level.
     - ➢ Location of the asset in your enterprise topology.

   b) **Identify the inventoried assets that are likely vulnerable.**

   - Use different scanning and detection tools and technologies (refer to CISA resource).

2. **Mitigate known and suspected vulnerable assets in your environment.**

   a) **Treat known and suspected vulnerable assets as compromised.**

   - These assets should be isolated until they are mitigated and verified.
   - Method of isolation depends on the criticality of the asset.

   b) **Patch Log4j and other affected products to the latest version.**

   - If unable to patch, apply appropriate compensating mitigations (should not be considered permanent fixes).
   - Mitigation alternatives:
     - ➢ Remove Jndilookup.class from the class path.
     - ➢ Delete or rename Jndilookup.class (will cause JndiContextSelector and JMSAppender to no longer function).
     - ➢ Apply a hot patch.

c) Keep an inventory of known and suspected vulnerable assets and what is done with them throughout this process.

- Organizations should keep a meticulous record of vulnerable assets they have patched to identify whether a threat actor may have patched an asset.

d) Verify the mitigation worked.

- Scan the patched/mitigated asset with the tools and methods listed in the CISA resource.
- Monitor the asset closely.
- Remain alert to changes from vendors for the software on the asset.

3. Initiate threat hunting and incident response procedures.

a) Hunt for signs of exploitation and compromise.

- Treat assets that use Log4j as suspect and conduct vigorous forensic investigation of those assets.
- Inspect and monitor accounts across your enterprise that exist on or connect to assets that use Log4j.
- Inspect changes to configurations made since December 1, 2021, and verify they were intended (especially on assets that use Log4j).
- Use CISA's GitHub page to detect possible exploitation or compromise (additional resources to detect possible exploitation or compromise are referred in the CISA resource).

b) If compromised is detected, organization should:

- Initiate incident response procedures.
- Consider reporting compromises immediately to applicable cybersecurity authorities.

4. Evaluate and apply other mitigations

a) Remain alerts to changes from vendors for the software on the asset, and immediately apply updates to assets.
b) Continue to monitor Log4j assets closely.
c) Continue to monitor Apache Log4j Security Vulnerabilities webpage for new updates.
d) Block specific outbound TCP and UDP network traffic.

- Outbound LDAP, Remote Method Invocation (RMI), and Outbound DNS.

<u>Second Advisory:</u>

Ransomware Threat Trends in 2021:

Ransomware = Malware designed to block access to a computer system or data until a sum of money (ransom) is paid.

- **Method** → Encrypts files on the device & Lock the device itself

Ransomware Top Trends:

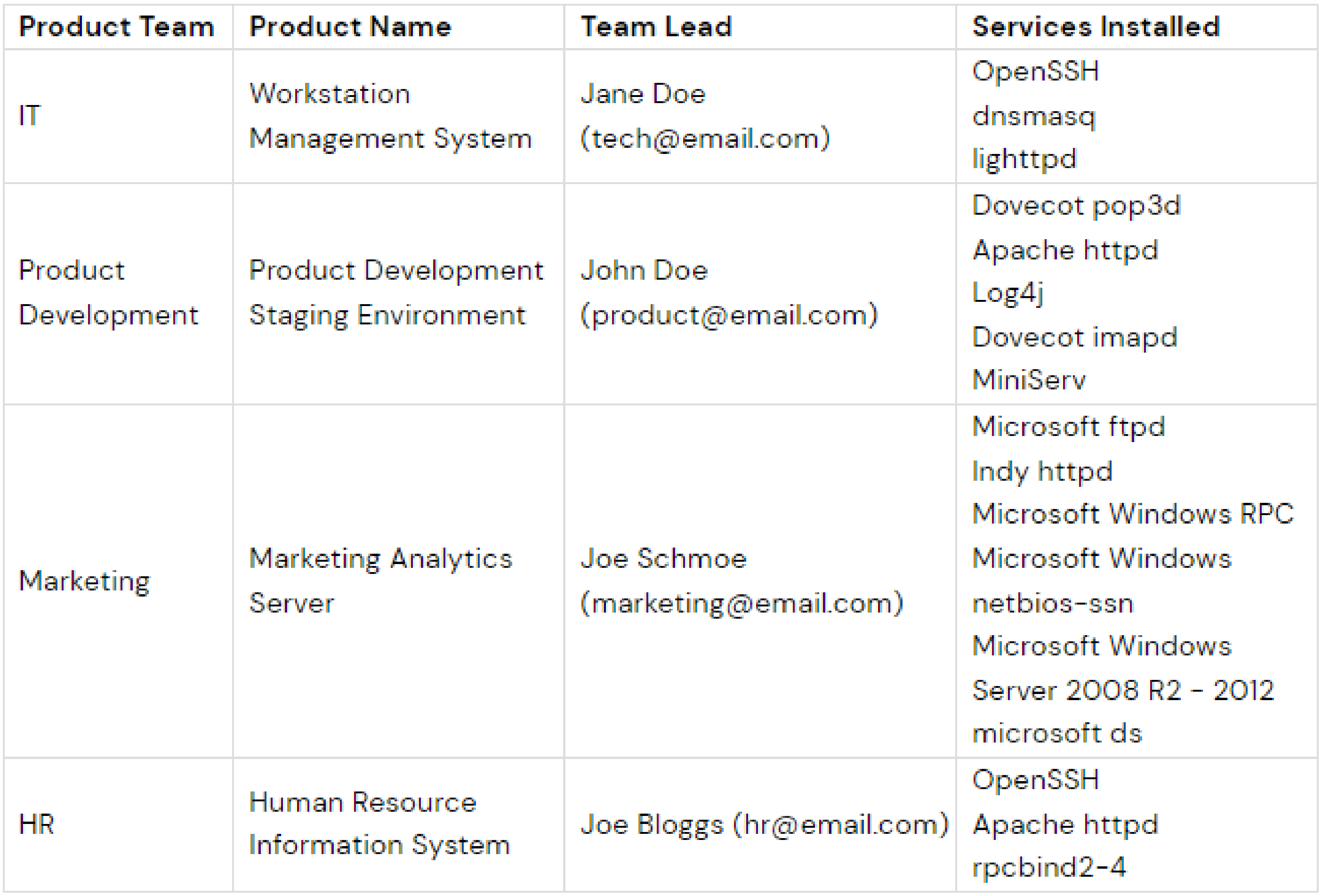
- Cybercriminals are increasingly gaining access to networks via phishing, stolen Remote Desktop Protocols (RDP) credentials or brute force, and exploiting software vulnerabilities.
- The market for ransomware became increasingly "professional" and there has been an increase in cybercriminal services-for-hire.
- More and more, ransomware groups are sharing victim information with each other, including access to victims' networks.
- Cybercriminals are diversifying their approaches extorting money.
- Ransomware groups are having an increasing impact thanks to approaches targeting the cloud, managed service providers, industrial processes and the software supply chain.
- Ransomware groups are increasingly targeting organizations on holidays and weekends.

Ransomware Mitigations:

- Keep all operating systems and software up to date.
- If you use RDP or other potentially risky service, secure and monitor them closely.
- Implement a user training program and phishing exercises.
- Require MFA.
- Require all accounts with password logins to have strong & unique passwords.
- If using Linux, use a Linux security module for defense in depth (SELinux, AppArmor, or SecComp).
- Protect cloud storage by backing up to multiple locations, requiring MFA access, and encrypting data in the cloud.

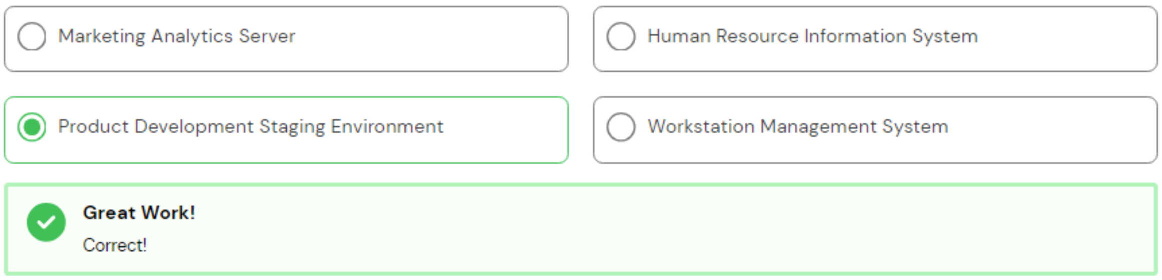
Task 1.2 - Analyze Infrastructure List

Analyze the “Infrastructure List” to find out which infrastructure may be affected by the vulnerability, and which team has ownership.



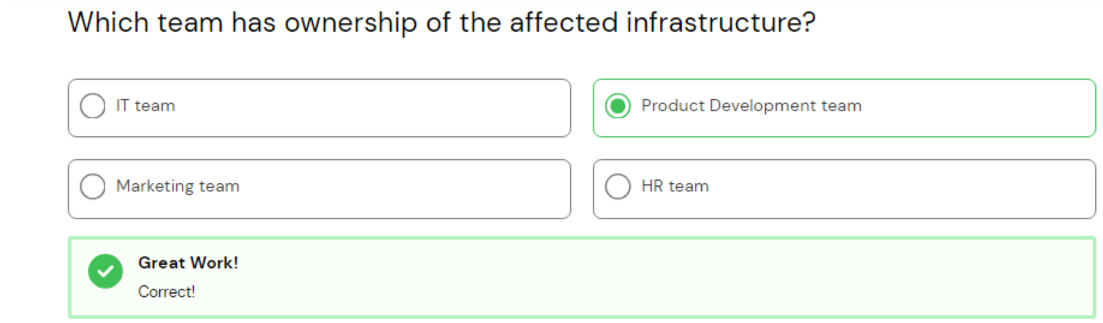
Product Team	Product Name	Team Lead	Services Installed
IT	Workstation Management System	Jane Doe (tech@email.com)	OpenSSH dnsmasq lighttpd
Product Development	Product Development Staging Environment	John Doe (product@email.com)	Dovecot pop3d Apache httpd Log4j Dovecot imapd MiniServ
Marketing	Marketing Analytics Server	Joe Schmoe (marketing@email.com)	Microsoft ftpd Indy httpd Microsoft Windows RPC Microsoft Windows netbios-ssn Microsoft Windows Server 2008 R2 - 2012 microsoft ds
HR	Human Resource Information System	Joe Bloggs (hr@email.com)	OpenSSH Apache httpd rpcbind2-4

The Product Development infrastructure is the only one that explicitly lists Log4j in its installed services, so it would be affected by the Apache Log4j vulnerability.



- Marketing Analytics Server
- Human Resource Information System
- [✓ Product Development Staging Environment]
- Workstation Management System

Great Work!
Correct!



Which team has ownership of the affected infrastructure?

- IT team
- [✓ Product Development team]
- Marketing team
- HR team

Great Work!
Correct!

Task 1.3 - Draft Advisory Email

Draft an advisory email to alert the infrastructure owner of the seriousness of this vulnerability.

Tips for your email:

- Make it direct and straight to the point.
- You can assume the infrastructure owner is technical.
- Explain the risk/impact, method of exploitation, and remediation steps.

Student Attempt:

From: AIG Cyber & Information Security Team
To: Product Development Team
Subject: Security Advisory concerning Product Development Staging Environment Apache Log4j
—
Body:
Hello John Doe,

AIG Cyber & Information Security Team would like to inform you that a recent Log4j vulnerability has been discovered in the security community that may affect the Product Development Staging Environment.

Multiple critical vulnerabilities (CVE-2021-44228, CVE-2021-45046, CVE-2021-45105) have been identified in Apache Log4j versions 2.0-beta9 to 2.16.0. These vulnerabilities allow for remote code execution and denial of service attacks.

Exploitation could lead to unauthorized access, data theft, or complete system compromise. Given that this affects the staging environment, there's a risk of lateral movement to production systems if exploited.

Attackers can trigger the vulnerability by sending specially crafted log messages that exploit JNDI lookups or cause uncontrolled recursion in string substitution evaluation.

Remediation Steps:

1. Immediately update Apache Log4j to version 2.17.0 or later.

2. If immediate updating is not possible, set the system property "log4j2.formatMsgNoLookups" to "true" or remove the JndiLookup class from the classpath as a temporary mitigation.
3. Review and monitor all systems and applications using Log4j for any signs of exploitation.
4. Implement network segmentation to isolate the staging environment if not already in place.

Please confirm once these steps have been implemented. Our team is available to assist with any questions or issues during the remediation process.

For any questions or issues, don't hesitate to reach out to us.

Kind regards,
AIG Cyber & Information Security Team

AIG Advisory Email Example:

From: AIG Cyber & Information Security Team
To: Product Development Team (product@email.com)
Subject: Security Advisory concerning Product Development Staging Environment | Log4j
—
Body:
Hello John Doe,

AIG Cyber & Information Security Team would like to inform you that a recent Log4j vulnerability has been discovered in the security community that may affect the Product Development Staging Environment infrastructure.

Vulnerability Overview
Log4j is a common open-source tool used for application logging and monitoring across the web. Recently, a vulnerability has been identified in versions Log4j2 2.0-beta9 through 2.15.0 that would allow an unauthenticated attacker to perform remote code execution on affected infrastructure, making this a critical vulnerability. You can learn more in the NIST disclosures: NVD - CVE-2021-44228 and NVD - CVE-2021-45046.

Affected products
Log4j2 2.0-beta9 through 2.15.0

Risk & Impact
Critical - remote code execution (RCE). An attacker will be able to remotely access the Product Development Staging Environment infrastructure to exfiltrate data or execute malicious actions.

Remediation
● Identify any assets or infrastructure running the affected Log4j version
● Update to the following versions: Log4j 2.16.0 (Java 8) and 2.12.2 (Java 7)
● Be on the lookout for any signs of exploitation

If you identified any signs of exploitation, please immediately reach out. After you have remediated this vulnerability, please confirm with the security team by replying to this email.

For any questions or issues, don't hesitate to reach out to us.

Kind regards,
AIG Cyber & Information Security Team

Task 2 - Bypassing Ransomware

Task Overview:

One of our servers has been exploited by the Apache Log4j vulnerability, and the attacker just tried to load some Ransomware. Write a bruteforcer to break into the ransomware-encrypted files, so we don't have to pay the ransom.

Will learn:

- What 'bruteforcing' involves.
- How to respond to a ransomware virus using Python.

Will do:

- Write a python script to bruteforce the decryption key of the encrypted file, to avoid paying a ransom.

Context

Your advisory email in the last task was great. It provided context to the affected teams on what the vulnerability was, and how to remediate it.

Unfortunately, an attacker was able to exploit the vulnerability on the affected server and began installing a ransomware virus. Luckily, the Incident Detection & Response team was able to prevent the ransomware virus from completely installing, so it only managed to encrypt one zip file.

Internally, the Chief Information Security Officer (CISO) does not want to pay the ransom, because there isn’t any guarantee that the decryption key will be provided or that the attackers won’t strike again in the future.

Instead, we would like you to bruteforce the decryption key. Based on the attacker’s sloppiness, we don’t expect this to be a complicated encryption key, because they used copy-pasted payloads and immediately tried to use ransomware instead of moving around laterally on the network.

Instructions

In this task, you will write a Python script to bruteforce the decryption key of the encrypted file.

Bruteforcing = Act of repeatedly trying different combinations to break the password encryption.

- **Brute Force Attack** → Attempts every possible combination to crack password-protected archives.
- **Dictionary Attack** → Uses a list of potential passwords (dictionary) from a file and tries each one.

RockYou = Widely known password wordlist that contains thousands of common passwords in one wordlist.

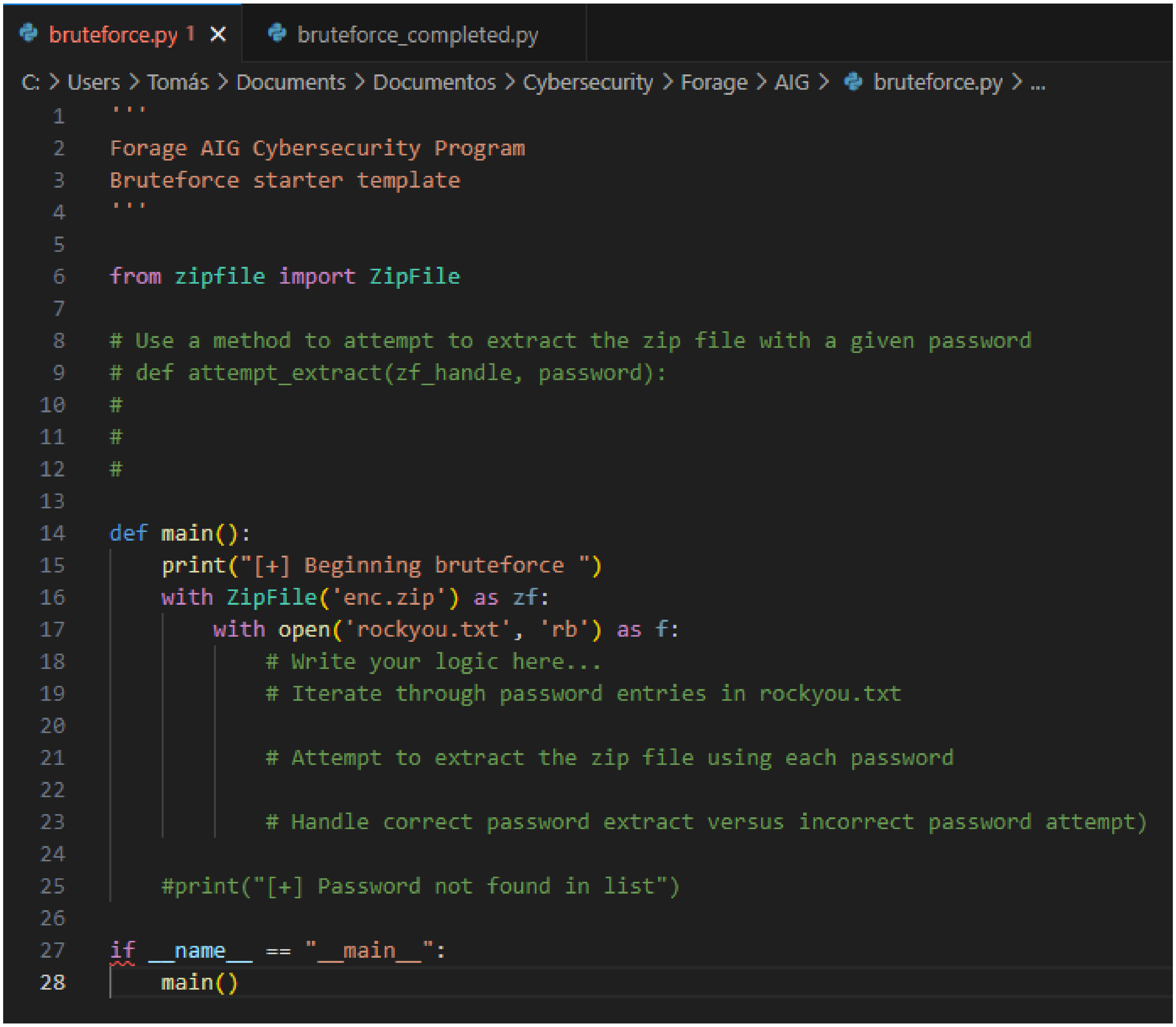
Ransomware will often encrypt all files on a device, and sometimes give the decryption key after the ransom has been paid (but this is not always the case).

In this task, we would like you to break the encryption without paying the ransom.

A foundational Python 3+ template has also been provided for you in the resource below. One potential implementation is described in the code comments.

Task 2.1 - Write a Python Script

Python script bruteforce template:



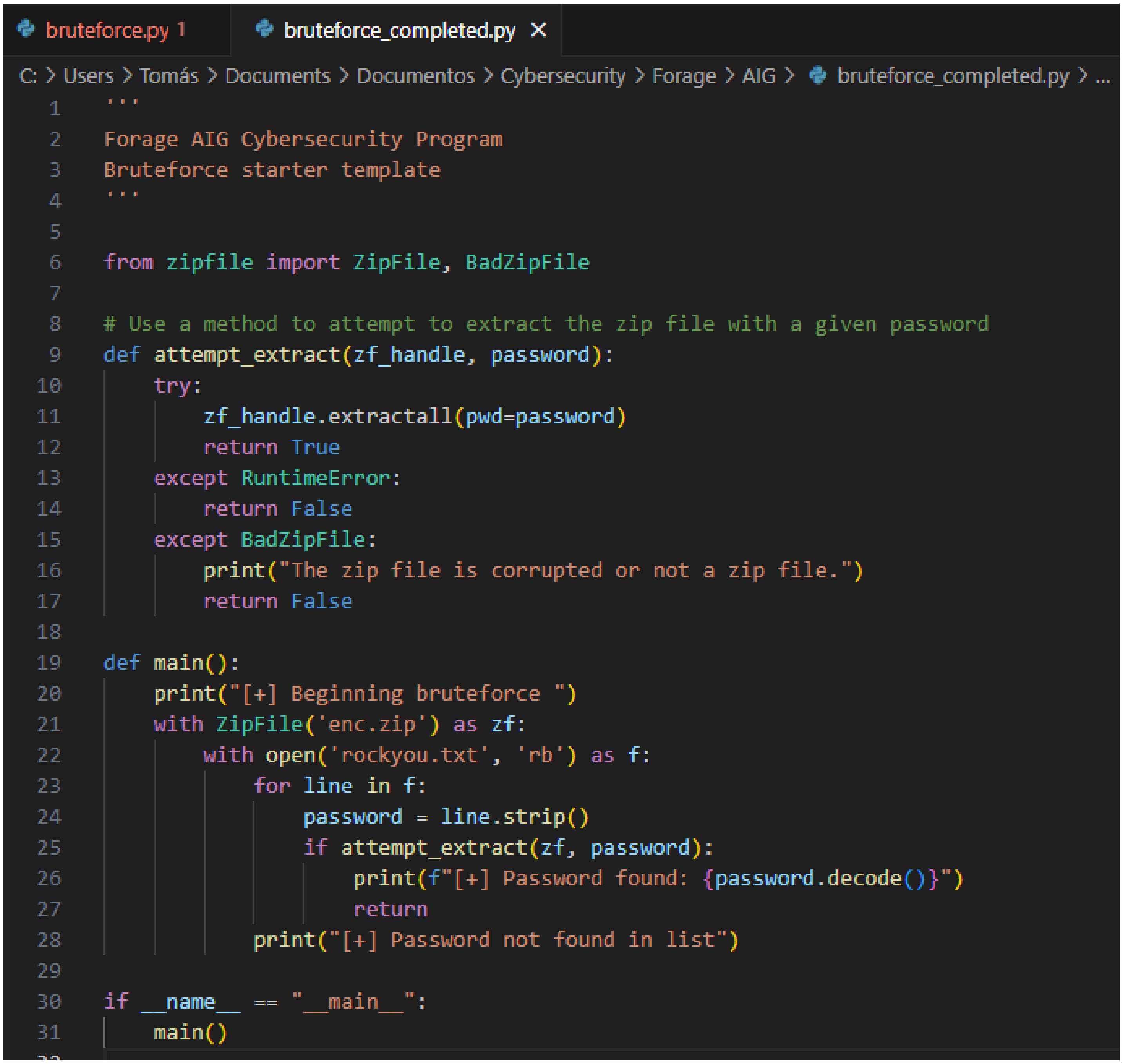
bruteforce.py 1 ×   bruteforce_completed.py

C: > Users > Tomás > Documents > Documentos > Cybersecurity > Forage > AIG > bruteforce.py > ...

```python
'''
Forage AIG Cybersecurity Program
Bruteforce starter template
'''

from zipfile import ZipFile

# Use a method to attempt to extract the zip file with a given password
# def attempt_extract(zf_handle, password):
#
#
#

def main():
    print("[+] Beginning bruteforce ")
    with ZipFile('enc.zip') as zf:
        with open('rockyou.txt', 'rb') as f:
            # Write your logic here...
            # Iterate through password entries in rockyou.txt

            # Attempt to extract the zip file using each password

            # Handle correct password extract versus incorrect password attempt)

    #print("[+] Password not found in list")

if __name__ == "__main__":
    main()
```

Stundent Attempt:

Python Script bruteforce completed:

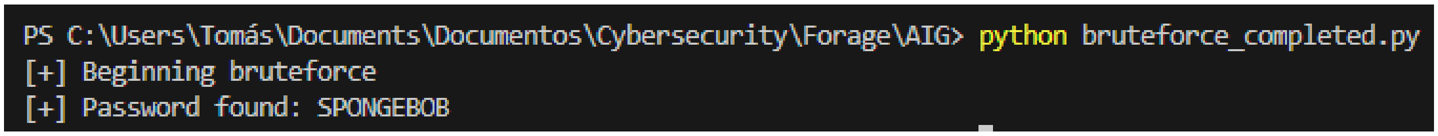


bruteforce.py 1 | bruteforce_completed.py

C: > Users > Tomás > Documents > Documentos > Cybersecurity > Forage > AIG > bruteforce_completed.py > ...

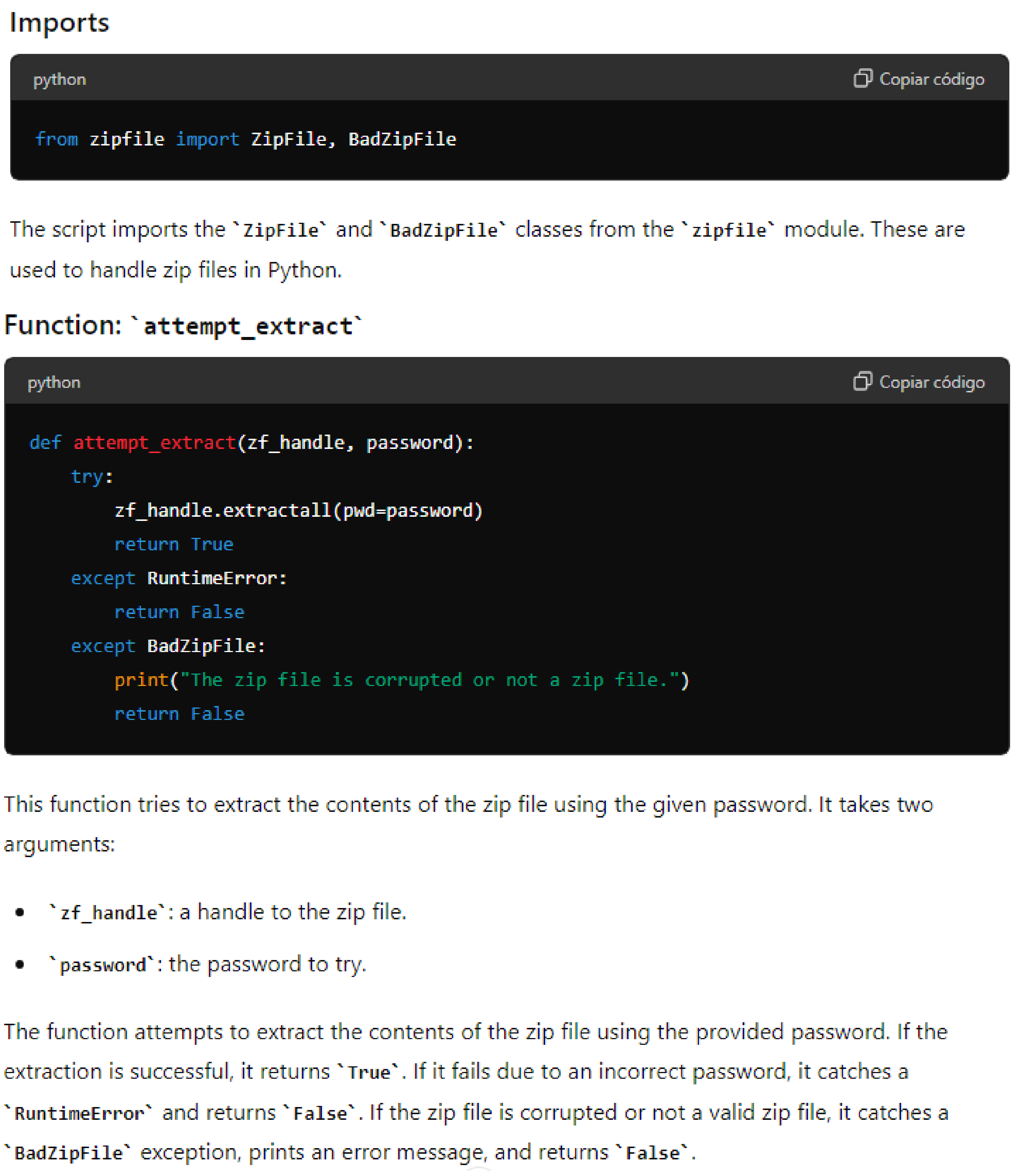
```python
'''
Forage AIG Cybersecurity Program
Bruteforce starter template
'''

from zipfile import ZipFile, BadZipFile

# Use a method to attempt to extract the zip file with a given password
def attempt_extract(zf_handle, password):
    try:
        zf_handle.extractall(pwd=password)
        return True
    except RuntimeError:
        return False
    except BadZipFile:
        print("The zip file is corrupted or not a zip file.")
        return False

def main():
    print("[+] Beginning bruteforce ")
    with ZipFile('enc.zip') as zf:
        with open('rockyou.txt', 'rb') as f:
            for line in f:
                password = line.strip()
                if attempt_extract(zf, password):
                    print(f"[+] Password found: {password.decode()}")
                    return
            print("[+] Password not found in list")

if __name__ == "__main__":
    main()
```

After running the script:



```
PS C:\Users\Tomás\Documents\Documentos\Cybersecurity\Forage\AIG> python bruteforce_completed.py
[+] Beginning bruteforce
[+] Password found: SPONGEBOB
```

Python Script Bruteforce completed explanation:



Imports

python Copiar código

```python
from zipfile import ZipFile, BadZipFile
```

The script imports the `ZipFile` and `BadZipFile` classes from the `zipfile` module. These are used to handle zip files in Python.

Function: `attempt_extract`

python Copiar código

```python
def attempt_extract(zf_handle, password):
    try:
        zf_handle.extractall(pwd=password)
        return True
    except RuntimeError:
        return False
    except BadZipFile:
        print("The zip file is corrupted or not a zip file.")
        return False
```

This function tries to extract the contents of the zip file using the given password. It takes two arguments:

- `zf_handle`: a handle to the zip file.
- `password`: the password to try.

The function attempts to extract the contents of the zip file using the provided password. If the extraction is successful, it returns `True`. If it fails due to an incorrect password, it catches a `RuntimeError` and returns `False`. If the zip file is corrupted or not a valid zip file, it catches a `BadZipFile` exception, prints an error message, and returns `False`.

Function: `main`

python Copiar código

```python
def main():
    print("[+] Beginning bruteforce ")
    with ZipFile('enc.zip') as zf:
        with open('rockyou.txt', 'rb') as f:
            for line in f:
                password = line.strip()
                if attempt_extract(zf, password):
                    print(f"[+] Password found: {password.decode()}")
                    return
            print("[+] Password not found in list")
```

This is the main function where the bruteforce process takes place. It:

1. Prints a message indicating the start of the bruteforce attack.
2. Opens the zip file `enc.zip` using a context manager (`with` statement).
3. Opens the password list file `rockyou.txt` (a commonly used password list in security testing) in binary read mode (`'rb'`).
4. Iterates over each line in the password file.
   - Each line is read as a password candidate, stripped of whitespace characters.
   - The `attempt_extract` function is called with the current password.
   - If the extraction is successful (`attempt_extract` returns `True`), it prints the found password (decoded from bytes to a string) and exits the function.
5. If no password is found after all candidates are tried, it prints a message indicating that the password was not found in the list.

Script Execution

python Copiar código

```python
if __name__ == "__main__":
    main()
```

This ensures that the `main` function is called only when the script is executed directly, not when it is imported as a module in another script.

AIG Python Script to Bruteforce Example:

```
from zipfile import ZipFile


def attempt_extract(zf_handle, password):
    try:
        zf_handle.extractall(pwd=password)
        return True
    except:
        return False


def main():
    print("[+] Beginning bruteforce ")
    with ZipFile('enc.zip') as zf:
        with open('rockyou.txt', 'rb') as f:
            for p in f:
                password = p.strip()
                if attempt_extract(zf, password):
                    print("[+] Correct password: %s" % password)
                    exit(0)
                else:
                    print("[-] Incorrect password: %s" % password)

    print("[+] Password not found in list")


if __name__ == "__main__":
    main()
```