NAME – NANCY
ISSUED BY CODEWAY TECHNOLOGY

SECTION A

Q1. Identify public S3 bucket vulnerability

📌 **Link**: [Pentester Land S3 Bucket Enum]

Amazon S3 bucket are storage container in AWS If a bucket is set to public (due to misconfiguration) it may leak sensitive file such as backup password or personal data Ethical hacker test for such misconfigurations to prevent data breache

Tools and Methods:

- Use tool like s3scanner, Bucket Finder, or AWS CLI.
- Common bucket name include company name, service, or variant like `company-dev`, `media-assets`, etc
- A method called DNS-style brute-forcing help enumerate possible bucket name

Example Command:

```
bash
CopyEdit
s3scanner --bucket my-bucket-names.txt
```

Real-World Example:
In 2017, Accenture left four S3 buckets publicly exposed containing credentials and APIs used in client project

Mitigation:

- Restrict public access in bucket setting
- Use IAM policies for tighter access control.
- Monitor buckets using AWS Config Rules or Trusted Advisor

```
$ python3 cloud_enum.py -k "company"

##########################
        cloud_enum
   github.com/initstring
##########################


Keywords:    calculator
Mutations:   /opt/cloud_enum/enum_tools/fuzz.txt
Brute-list:  /opt/cloud_enum/enum_tools/fuzz.txt

[+] Mutations list imported: 306 items
[+] Mutated results: 1837 items

++++++++++++++++++++++++++
      amazon checks
++++++++++++++++++++++++++

[+] Checking for S3 buckets
  Protected S3 Bucket: http://company-secured.s3.amazonaws.com/
  Protected S3 Bucket: http://company2-secured.s3.amazonaws.com/
  OPEN S3 BUCKET: http://company.s3.amazonaws.com/
      FILES:
      ->http://company.s3.amazonaws.com/index.html
      ->http://company.s3.amazonaws.com/downloads/
      ->http://company.s3.amazonaws.com/archive.zip
...
```

Q2. Scan cloud misconfigures with ScoutSuite

📌 **Link**: [ScoutSuite GitHub]

ScoutSuite is a powerful Python-based tool used to scan AWS Azure, and GCP environment for misconfiguration. It is widely used by security teams to detect weak configuration and policy gap.

Key Features:

- Scans identity, compute, storage, and networking setting
- Generates an HTML report showing vulnerabilitie
- Supports multi cloud auditing

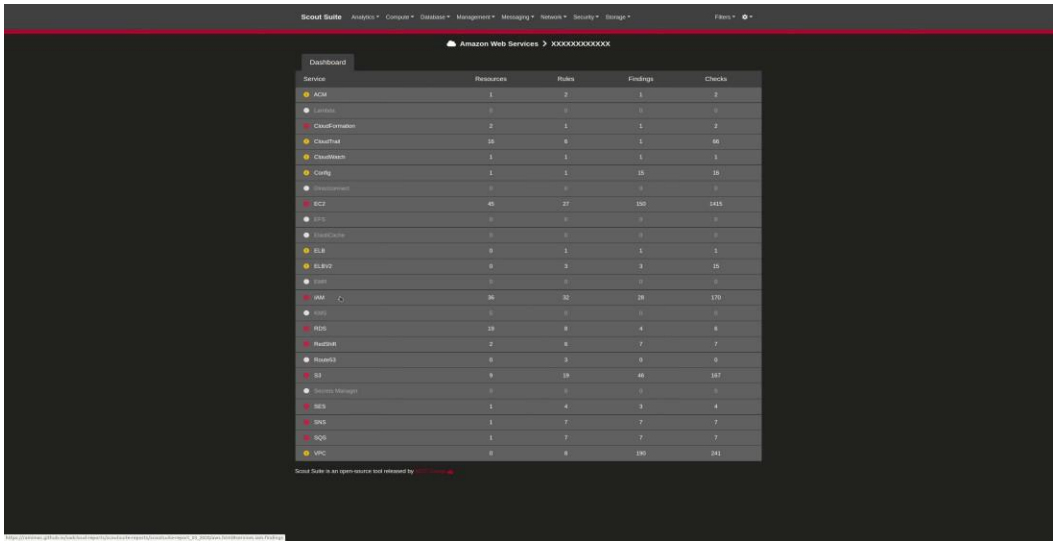
How to Use:

```
bash
CopyEdit
git clone https://github.com/nccgroup/ScoutSuite
cd ScoutSuite
python3 scout.py aws
```

Real-Life Use:
A company used ScoutSuite and found that their AWS security group allowed open access on port 22 (SSH) exposing them to brute force attack

Benefits:

- Detect weak IAM roles and public S3 bucket.
- Visual summary of cloud security posture
- Helps companies prepare for compliance audit



Q3. Use kube-hunter for Kubernetes vulnerability scan

📌 **Link**: [Kube-Hunter GitHub]

kube-hunter is a security scanning tool designed to find vulnerabilitie in Kubernete cluster. It is developed by Aqua Security and can be run in two mode: remote scanning (from outside the cluster) and internal scanning (from within the cluster)

How it works:

- Scan for common misconfiguration like open dashboard, insecure API server, exposed kubelet port, etc.
- Generate a vulnerability report

Example:
A misconfigured Kubernetes cluster exposed its dashboard publicly without login. kube-hunter flagged it, helping the team to restrict access

Benefits:

- Helps DevOp team secure Kubernete environment
- Easy to use in CI/CD pipeline
- Detect real-world exploitable issue

```
- kubectl logs pod/kube-hunter-65bpk
022-06-20 02:02:47,350 INFO kube_hunter.modules.report.collector Started hunting
022-06-20 02:02:47,354 INFO kube_hunter.modules.report.collector Discovering Open Kubernetes Services
022-06-20 02:02:47,356 INFO kube_hunter.modules.report.collector Found vulnerability "Read access to pod's service account token" in L
cal to Pod (kube-hunter-65bpk)
022-06-20 02:02:47,356 INFO kube_hunter.modules.report.collector Found vulnerability "CAP_NET_RAW Enabled" in Local to Pod (kube-hunte
-65bpk)
022-06-20 02:02:47,358 INFO kube_hunter.modules.report.collector Found vulnerability "Access to pod's secrets" in Local to Pod (kube-h
nter-65bpk)
022-06-20 02:02:56,638 INFO kube_hunter.modules.report.collector Found open service "Kubelet API" at 10.244.1.1:10250
022-06-20 02:03:04,137 INFO kube_hunter.modules.report.collector Found open service "API Server" at 10.96.0.1:443
022-06-20 02:03:04,171 INFO kube_hunter.modules.report.collector Found vulnerability "K8s Version Disclosure" in 10.96.0.1:443
022-06-20 02:03:04,171 INFO kube_hunter.modules.report.collector Found vulnerability "Access to API using service account token" in 10
96.0.1:443

odes
-------------+------------+
 TYPE        | LOCATION   |
-------------+------------+
 Node/Master | 10.244.1.1 |
-------------+------------+
 Node/Master | 10.96.0.1  |
-------------+------------+

etected Services
-------------+------------------+----------------------+
 SERVICE     | LOCATION         | DESCRIPTION          |
-------------+------------------+----------------------+
 Kubelet API | 10.244.1.1:10250 | The Kubelet is the   |
             |                  | main component in    |
             |                  | every Node, all pod  |
             |                  | operations goes      |
             |                  | through the kubelet  |
-------------+------------------+----------------------+
 API Server  | 10.96.0.1:443    | The API server is in |
             |                  | charge of all        |
             |                  | operations on the    |
```

Q4. Configure secure Docker image

📌 **Link**: [Snyk's 10 Docker Security Best Practices]

Docker image can carry vulnerabilities if not built securely Snyk outline key practice for creating safer Docker image

Best Practices:

- Use minimal base image like Alpine
- Pin specific image version to avoid auto-update
- Don not run container as root
- Clean up unnecessary file and secret
- Regularly scan image using tool like Trivy or Snyk CLI

Example:

```
Dockerfile
CopyEdit
FROM node:16-alpine
RUN addgroup app && adduser -S -G app appuser
USER appuser
```

Impact:
These practice reduce attack surface and prevent privilege escalation

```
➜  buildkit-image DOCKER_BUILDKIT=1 docker build --no-cache -t secret:buildkit --secret id=mysecret,src=mysecret.txt .
[+] Building 1.8s (8/8) FINISHED
 => [internal] load build definition from Dockerfile                                                    0.0s
 => => transferring dockerfile: 37B                                                                     0.0s
 => [internal] load .dockerignore                                                                       0.0s
 => => transferring context: 2B                                                                         0.0s
 => resolve image config for docker.io/docker/dockerfile:1.2                                            0.6s
 => CACHED docker-image://docker.io/docker/dockerfile:1.2@sha256:e2a8561e419ab1ba6b2fe6cbd              0.0s
 => => resolve docker.io/docker/dockerfile:1.2@sha256:e2a8561e419ab1ba6b2fe6cbdf49fd92b959              0.0s
 => [internal] load metadata for docker.io/library/alpine:latest                                        0.7s
 => CACHED [1/2] FROM docker.io/library/alpine@sha256:234cb88d3020898631af0ccbbcca9a66ae73              0.0s
 => [2/2] RUN --mount=type=secret,id=mysecret cat /run/secrets/mysecret                                 0.2s
 => exporting to image                                                                                  0.0s
 => => exporting layers                                                                                 0.0s
 => => writing image sha256:a7681c77cd3187dee05c7ebf9efc32d4e4bfef77f8f685ce6001a4dac3be71              0.0s
 => => naming to docker.io/library/secret:buildkit                                                      0.0s
```

z

Q5. Run Trivy to scan Docker images

📌 **Link**: [Trivy GitHub]

Trivy is an open source vulnerability scanner for Docker image file systemand Git repositorie

Feature:

- Detect OS package and language specific vulnerabilitie
- Scans container image before deployment
- Fast and easy CLI interface

Command:

```
bash
CopyEdit
trivy image nginx:latest
```

Real Example:
Trivy detected critical `openssl` and `glibc` vulnerabilities in a base Ubuntu image used by a company helping them update it before production.

Why Trivy?

- Lightweight and developer friendly.
- Easily integrate with CI/CD tools like GitHub Action or Jenkin

```
tristan@big-brother-II:~$ trivy
NAME:
   trivy - A simple and comprehensive vulnerability scanner for containers

USAGE:
   trivy command [command options] target

COMMANDS:
   image, i          scan an image
   filesystem, fs    scan local filesystem for language-specific dependencies and config files
   rootfs            scan rootfs
   repository, repo  scan remote repository
   client, c         client mode
   server, s         server mode
   config, conf      scan config files
   plugin, p         manage plugins
   help, h           Shows a list of commands or help for one command

OPTIONS:
   --quiet, -q        suppress progress bar and log output (default: false) [$TRIVY_QUIET]
   --debug, -d        debug mode (default: false) [$TRIVY_DEBUG]
   --cache-dir value  cache directory (default: "/home/tristan/.cache/trivy") [$TRIVY_CACHE_DIR]
   --help, -h         show help (default: false)
   --version, -v      print the version (default: false)
```

Q6. Monitor AWS account using CloudTrail

📌 **Link**: [AWS CloudTrail]

AWS CloudTrail is a monitoring tool that logs all actions taken by user, role, and AWS service. It is essential for incident response and audit tracking

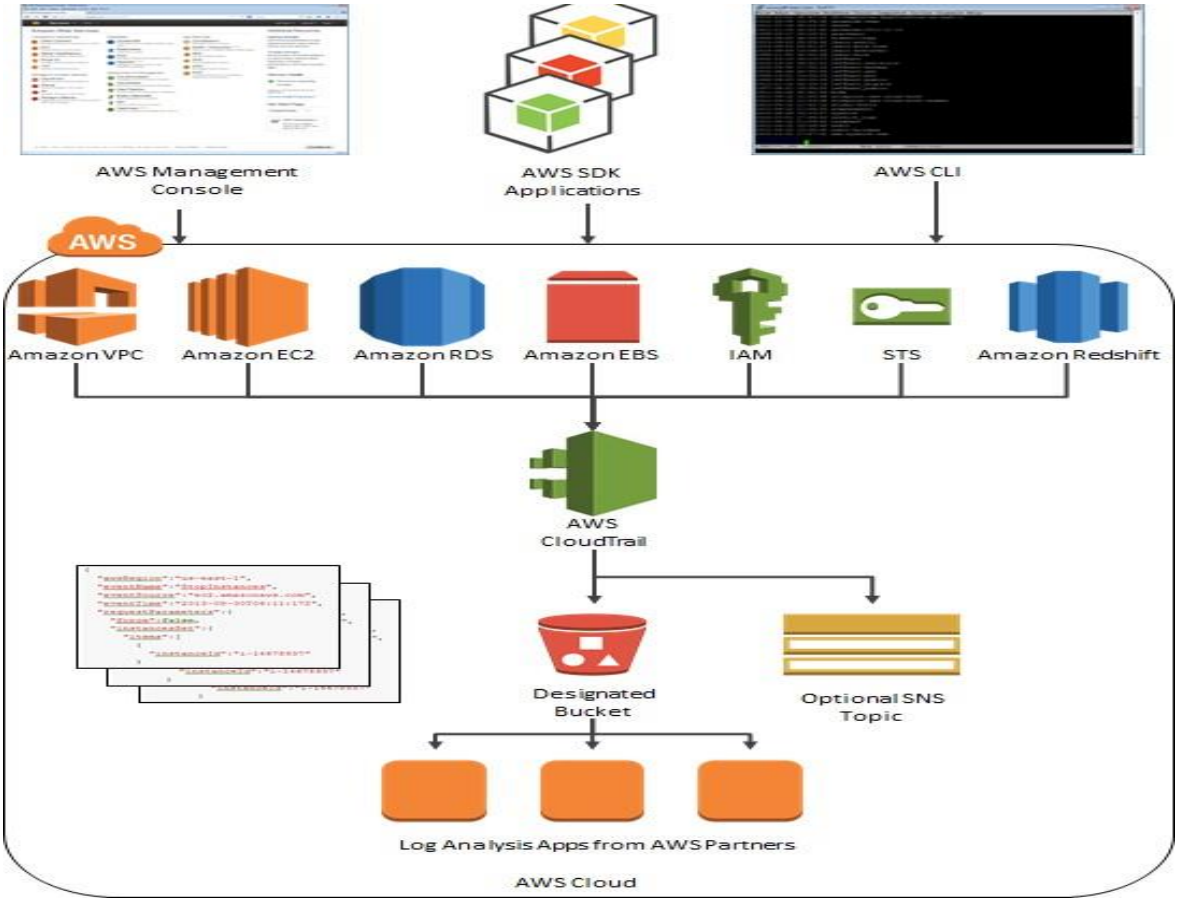
Use Case:

- Detect unauthorized logins or API call
- Monitor change in IAM or EC2
- Meet compliance like PCI-DSS and GDPR.

How to Enable:

1. Go to CloudTrail in AWS Console
2. Create a new trail
3. Enable logging for all region
4. Store log in a secure S3 bucket.

Example:
A CloudTrail log helped a company detect an unauthorized user deleting EC2 instance after stealing access key




```
~  trivy k8s --report summary
179 / 179 [----------------------------------------------------------------------------------------------------] 100.00% 12 p/s

Summary Report for k3d-first-cluster
```

Namespace	Resource	Vulnerabilities					Misconfigurations					Secrets				
		C	H	M	L	U	C	H	M	L	U	C	H	M	L	U
kube-system	Deployment/local-path-provisioner	2	5	2		1			8	11						
kube-system	Deployment/metrics-server		2	1		1			6	8						
kube-system	Deployment/traefik	3	5	1		3			7	7						
kube-system	DaemonSet/svclb-traefik	2	21	2				4	16	20						
kube-system	DaemonSet/svclb-traefik	2	21	2				4	16	20						
kube-system	Job/helm-install-traefik	10	54	20	1	14			8	11						
kube-system	Job/helm-install-traefik-crd	10	54	20	1	14			8	11						
kube-system	Deployment/coredns		1			1			8	5						
kube-system	Service/kube-dns								2	2						
kube-system	Service/metrics-server								2	2						
kube-system	Service/traefik								2	2						
default	Service/mysql								1	2						
default	Service/mysql-headless								1	2						
default	StatefulSet/mysql	12	36	26	113				7	12						
default	Pod/thisisfine	43	217	196	514	2			9	11						
default	Pod/nginx	6	18	24	92				9	11						
default	Service/kubernetes								1	2						

```
Severities: C=CRITICAL H=HIGH M=MEDIUM L=LOW U=UNKNOWN
```

Q7. Use AWS Inspector for vulnerability scanning

📌 **Link**: [AWS Inspector]

AWS Inspector is a security assessment service that scan EC2 instance for vulnerabilitie

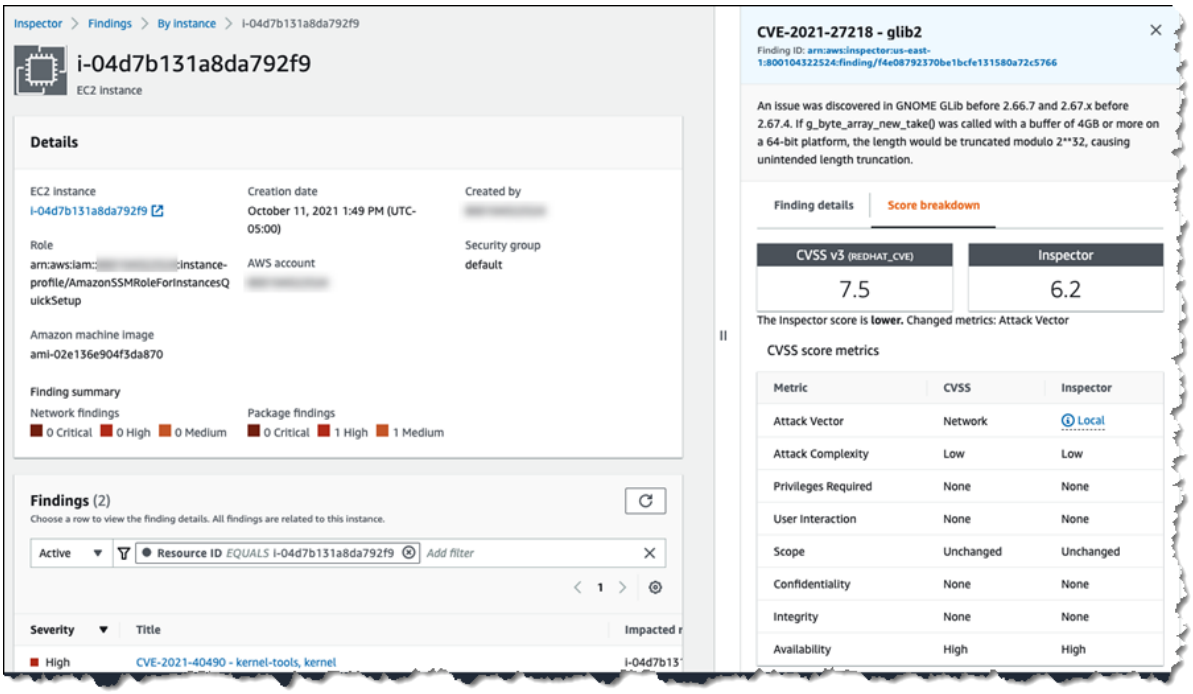
Features:

- Analyze OS, installed software, and network accessibility.
- Scores vulnerabilitie using CVSS
- Integrate with AWS Security Hub.

How to Use:

1. Open AWS Inspector in console
2. Create an assessment target and template
3. Run the assessment and review finding

Real-Life Use:
A company found outdated Apache and PHP version on EC2 via Inspector, which they patched immediately



```
- on ☁ kloudle-test (us-east-2) on ☁ ayush@kloudle.com(us-central1)
❯ aws inspector2 list-findings
{
    "findings": [
        {
            "awsAccountId": "914397138150",
            "description": "OpenSSL 1.0.2 supports SSLv2. If a client attempts to negotiate SSLv2 with a server that is configured to support both SSLv2 and more recent SSL and TLS versions then a check is made for a version rollback attack when unpadding an RSA
signature. Clients that support SSL or TLS versions greater than SSLv2 are supposed to use a special form of padding. A server that supports greater than SSLv2 is supposed to reject connection attempts from a client where this special form of padding is present
, because this indicates that a version rollback has occurred (i.e. both client and server support greater than SSLv2, and yet this is the version that is being requested). The implementation of this padding check inverted the logic so that the connection attemp
t is accepted if the padding is present, and rejected if it is absent. This means that such as server will accept a connection if a version rollback attack has occurred. Further the server will erroneously reject a connection if a normal SSLv2 connection attempt
is ",
            "findingArn": "arn:aws:inspector2:us-east-2:914397138150:finding/007400ea3753d374267589fafb5b9425",
            "firstObservedAt": 1638965591.334,
            "inspectorScore": 3.7,
            "inspectorScoreDetails": {
                "adjustedCvss": {
                    "adjustments": [],
                    "cvssSource": "NVD",
                    "score": 3.7,
                    "scoreSource": "NVD",
                    "scoringVector": "CVSS:3.1/AV:N/AC:H/PR:N/UI:N/S:U/C:N/I:L/A:N",
                    "version": "3.1"
                }
            },
            "lastObservedAt": 1638965591.334,
            "packageVulnerabilityDetails": {
                "cvss": [
                    {
                        "baseScore": 4.3,
                        "scoringVector": "AV:N/AC:M/Au:N/C:N/I:P/A:N",
                        "source": "NVD",
                        "version": "2.0"
                    },
                    {
                        "baseScore": 3.7,
                        "scoringVector": "CVSS:3.1/AV:N/AC:H/PR:N/UI:N/S:U/C:N/I:L/A:N",
                        "source": "NVD",
                        "version": "3.1"
                    }
                ],
                "referenceUrls": [
                    "https://www.oracle.com/security-alerts/cpuApr2021.html",
                    "https://www.openssl.org/news/secadv/20210216.txt",
                    "https://www.oracle.com//security-alerts/cpujul2021.html",
                    "https://git.openssl.org/gitweb/",
                    "https://www.oracle.com/security-alerts/cpuoct2021.html"
                ],
                "relatedVulnerabilities": [],
                "source": "NVD",
                "sourceUrl": "https://nvd.nist.gov/vuln/detail/CVE-2021-23839",
                "vendorCreatedAt": 1613405700.0,
                "vendorSeverity": "LOW",
                "vulnerabilityId": "CVE-2021-23839",
                "vulnerablePackages": [
                    {
                        "arch": "X86_64",
                        "epoch": 0,
                        "name": "openssl",
                        "packageManager": "OS",
                        "release": "r0",
                        "sourceLayerHash": "sha256:cbdbe7a5bc2a134ca8ec91be58565ec07d037386d1f1d0385412d224deafca08",
                        "version": "1.1.1g"
                    }
                ]
```

Q8. Enable MFA in AWS IAM

📌 **Link**: [AWS MFA Guide]

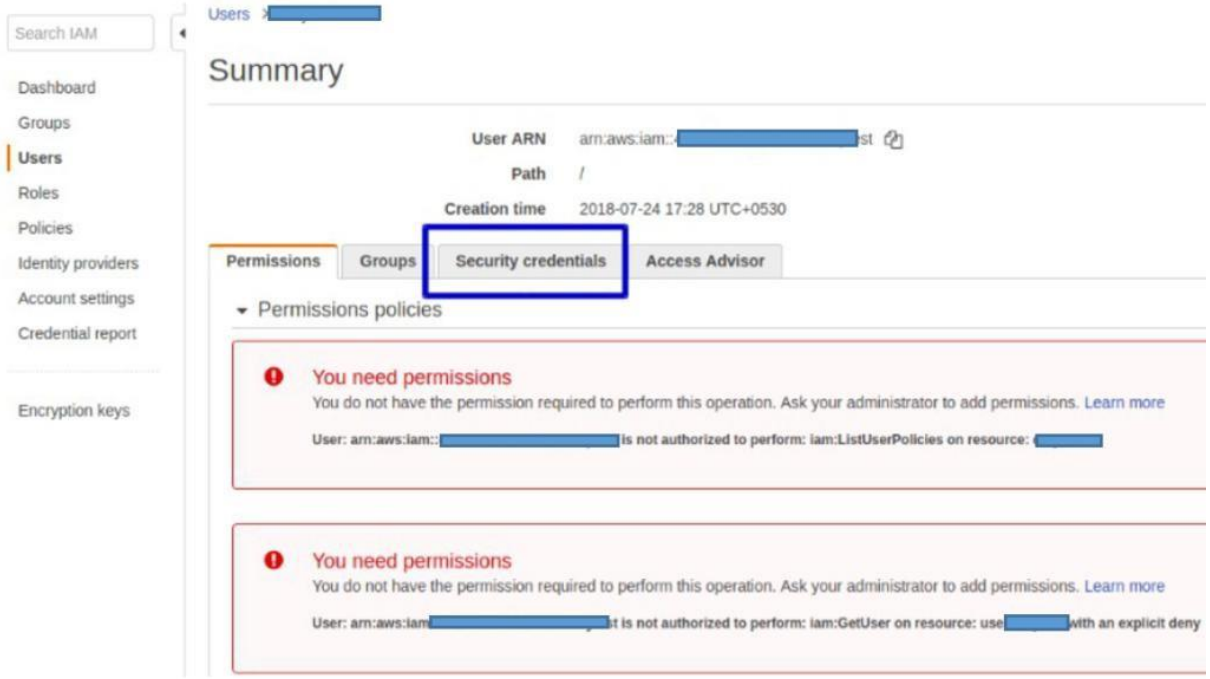
Multi Factor Authentication (MFA) adds an extra layer of security for AWS user. It prevent unauthorized access even if credential are stolen

Steps:

1. Go to IAM → Users → Security Credential
2. Click Assign MFA device
3. Use virtual (like Google Authenticator) or hardware MFA
4. Scan QR code and verify with two OTP

Why Important?

- Prevent unauthorized root acces
- Mandatory for high privilege role
- Support compliance need



Q9. Test access control in GCP

📌 **Link**: [GCP IAM Documentation]

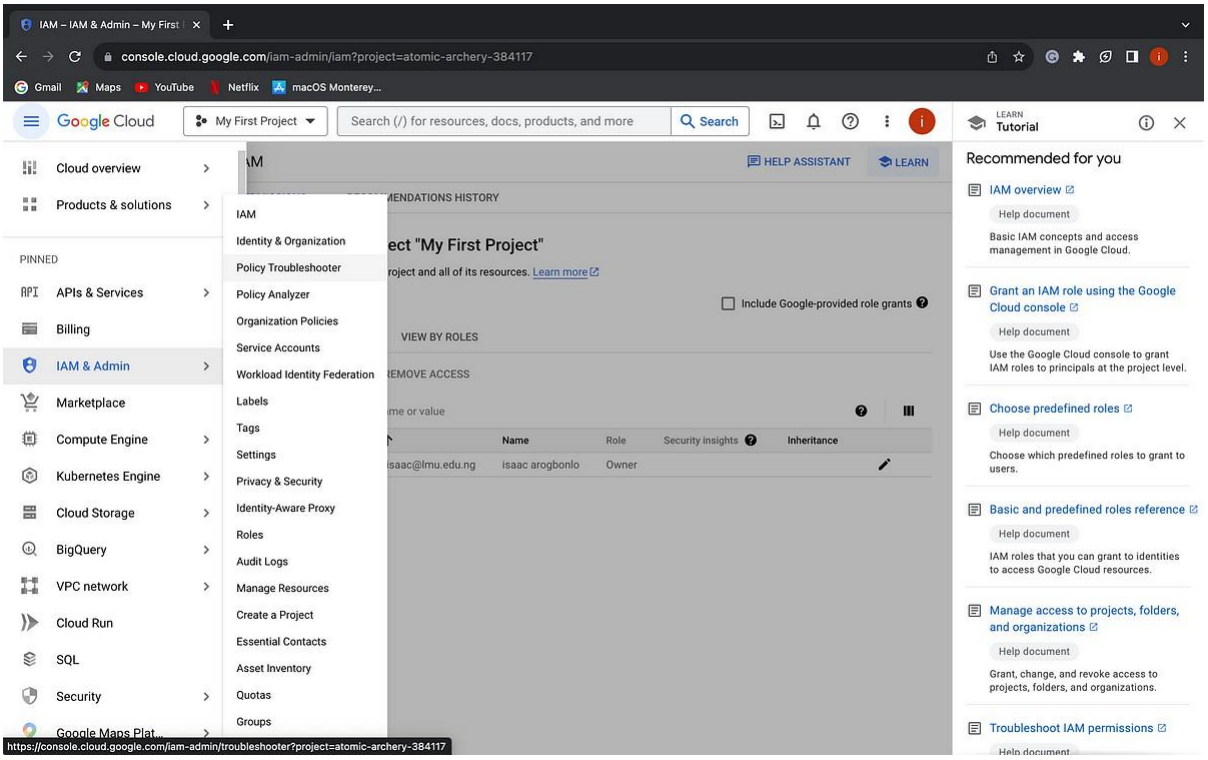
Google Cloud IAM allows fine grained access control by assigning role to users or group

Key Concepts:

- **Principal**: user, group, or service.
- **Roles**: Viewer, Editor, Owner, or custom.
- **Policies**: Bind role to principal on resource.

Test Scenario:
Create a custom role with limited permission read-only access to GCS and test if the user can edit or delete file

Real Case:
A company mistakenly gave Editor role to an intern who deleted important VMs. Least privilege principle would have prevented it.



Q10. Scan Azure cloud using AzScanner

📌 **Link**: [AzScanner GitHub]

AzScanner Azure Tenant Security Scanner is a Microsoft tool to evaluate security settings across Azure tenant

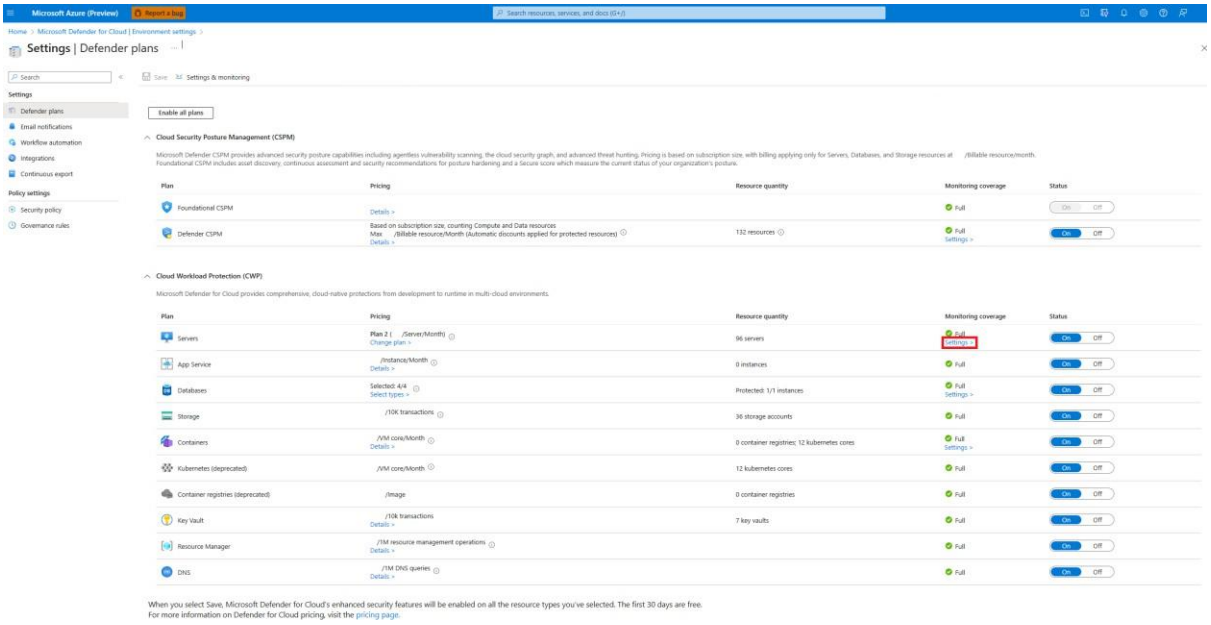
Functionality:

- Check compliance with Azure Security Benchmark
- Detects weak password, missing MFA, open RDP/SSH port, and more
- Support PowerShell based automation.

How to Run:

- Clone repo and install pre reqs like PowerShell module
- Execute Start-AzTSAssessment.

Example Finding:
Identified storage account with public blob access enabled which were later locked down.



Q11. Detect CSRF vulnerability

📌 **Link**: [PortSwigger CSRF Guide]

CSRF (Cross-Site Request Forgery) is an attack where an attacker trick a logged-in user into submitting maliciou request unknowingly.

Example:
If a banking site does not verify the user identity using CSRF token an attacker can craft a form like:

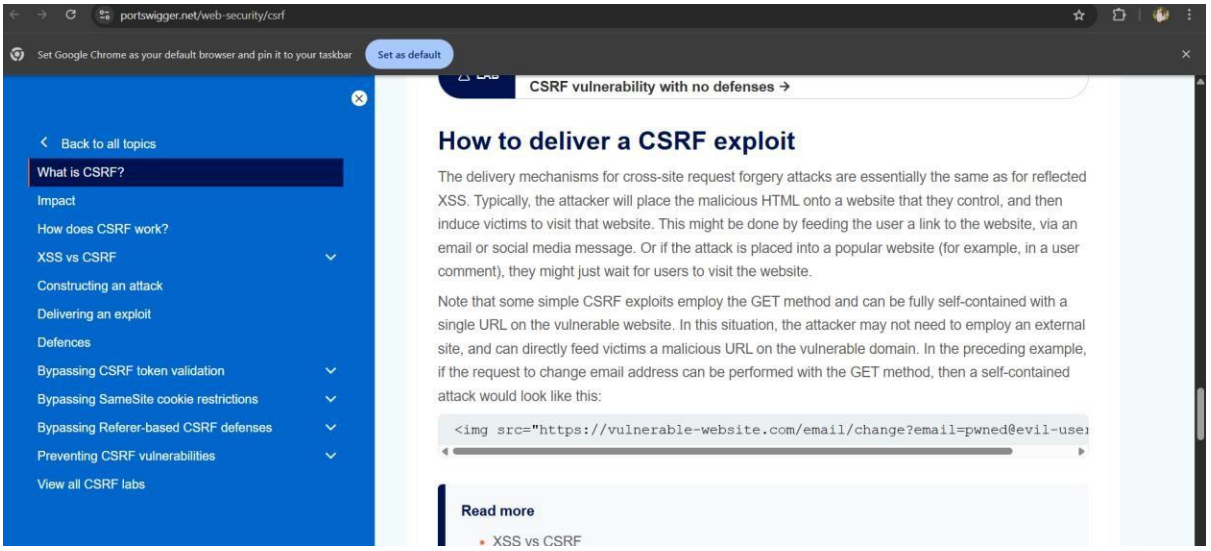
```
html
CopyEdit
<form action="https://bank.com/transfer" method="POST">
  <input type="hidden" name="amount" value="1000">
  <input type="hidden" name="to" value="attacker">
</form>
```

When the victim open this page while logged in the request is executed.

Mitigation:

- Use anti-CSRF token
- Validate origin or referer header
- Use SameSite cookie

Tool: Burp Suite can help identify CSRF by modifying parameter and replaying request.



Q12. Run an IDOR exploit lab

📌 **Link**: [PortSwigger IDOR Lab]

IDOR (Insecure Direct Object Reference) allows attacker to access unauthorized resource by changing identifier in the URL or request

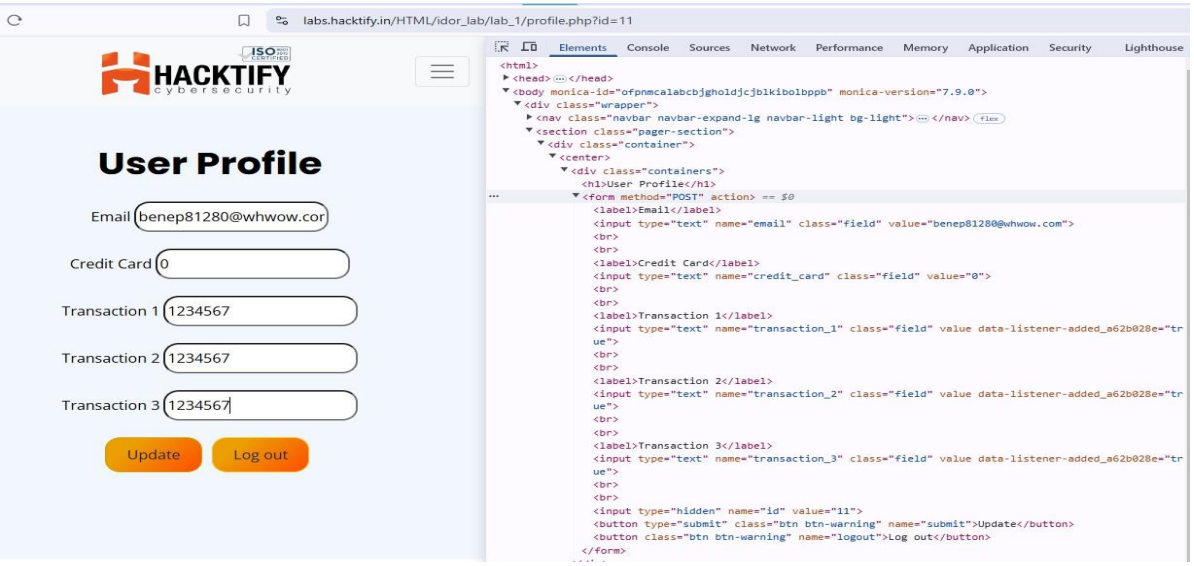
Example:

```
url
CopyEdit
GET /invoice/123.pdf → shows your file
GET /invoice/124.pdf → shows another user's file
```

Real-World Breach:
Facebook had an IDOR issue where attacker accessed private photo by manipulating image ID

Mitigation:

- Use acces control check on the server
- Avoid exposing sequential ID
- Use UUID and authorization check



Q13. Test insecure direct object references

📌 **Link**: [TryHackMe OWASP10 Room]

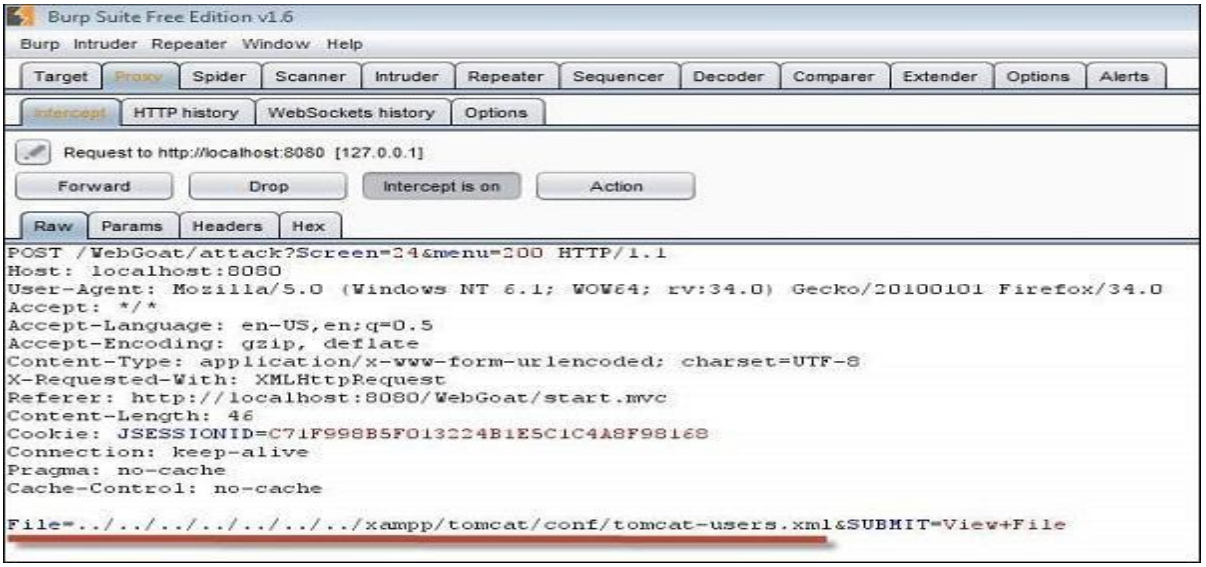
This room contain practical lab for OWASP Top 10 including IDOR

Step to test:

1. Access a URL with numeric ID (e.g., `/profile?id=2`).
2. Change ID to check for unauthorized acces
3. Observe if sensitive data leak

TryHackMe teaches:

- How to identify broken access control
- How IDOR are combined with privilege escalation



Q14. Inspect cookies and session IDs

📌 **Link**: [MDN Cookies Guide]

Cookie store session and preference data. If session ID are exposed, attacker can hijack user account

How to Inspect:

- Use browser DevTools → Application → Cookies.
- Look for session token, authentication data.

Security Flags:

- `HttpOnly`: Prevent JavaScript access
- `Secure`: Sends cookie only over HTTPS.
- `SameSite`: Blocks cross origin request.

Example:
Poor cookie setting allowed attacker to steal session via XSS on a forum site.

Go Cancel < > Target: http://192.168.1.102:81

Request

Raw Params Headers Hex

```
POST /dvwa/vulnerabilities/weak_id/ HTTP/1.1
Host: 192.168.1.102:81
User-Agent: Mozilla/5.0 (X11; Linux x86_64;
rv:45.0) Gecko/20100101 Firefox/45.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0
.9,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Referer:
http://192.168.1.102:81/dvwa/vulnerabilities/weak_i
d/
Cookie: security=low; security_level=0;
PHPSESSID=p38kq30vi6arr0b32lp2uv86k0
Connection: close
Content-Type: application/x-www-form-urlencoded
Content-Length: 0
```

Response

Raw Headers Hex HTML Render

```
HTTP/1.1 200 OK
Date: Thu, 06 Jul 2017 14:33:16 GMT
Server: Apache/2.4.25 (Win32) OpenSSL/1.0.2j
PHP/5.6.30
X-Powered-By: PHP/5.6.30
Expires: Tue, 23 Jun 2009 12:00:00 GMT
Cache-Control: no-cache, must-revalidate
Pragma: no-cache
Set-Cookie: dvwaSession=6
Content-Length: 4310
Connection: close
Content-Type: text/html;charset=utf-8


<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0
Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd
">
<html xmlns="http://www.w3.org/1999/xhtml">

        <head>
                <meta http-equiv="Content-Type"
content="text/html; charset=UTF-8" />

                <title>Vulnerability: Weak
Session IDs :: Damn Vulnerable Web Application
(DVWA) v1.10 *Development*</title>
```

Q15. Test HTTP headers for security misconfig

📌 **Link**: SecurityHeaders.com

HTTP security header protect website from attack like clickjacking, XSS, and code injection.

Important Header:

- `Content-Security-Policy`
- `Strict-Transport-Security`
- `X-Frame-Options`
- `X-Content-Type-Options`

How to Test:
Enter a domain on securityheaders.com and review the score and recommendation

Use Case:
A site missing `CSP` was vulnerable to JavaScript injection via third-party widget

```
Testing HTTP header response @ "/"

HTTP Status Code             200 OK
HTTP clock skew              +195057 sec from localtime
Strict Transport Security    182 days=15768000 s, includeSubDomains, preload
Public Key Pinning           --
Server banner                Apache
Application banner           --
Cookie(s)                    (none issued at "/")
Security headers             X-Frame-Options: SAMEORIGIN
                             X-XSS-Protection: 1; mode=block
                             X-Content-Type-Options: nosniff
                             Content-Security-Policy: default-src 'self' *
Reverse Proxy banner         --
```

Q16. Use ZAP proxy for scanning

📌 **Link**: [OWASP ZAP]

ZAP (Zed Attack Proxy) is a free and powerful tool used to find web application vulnerabilitie

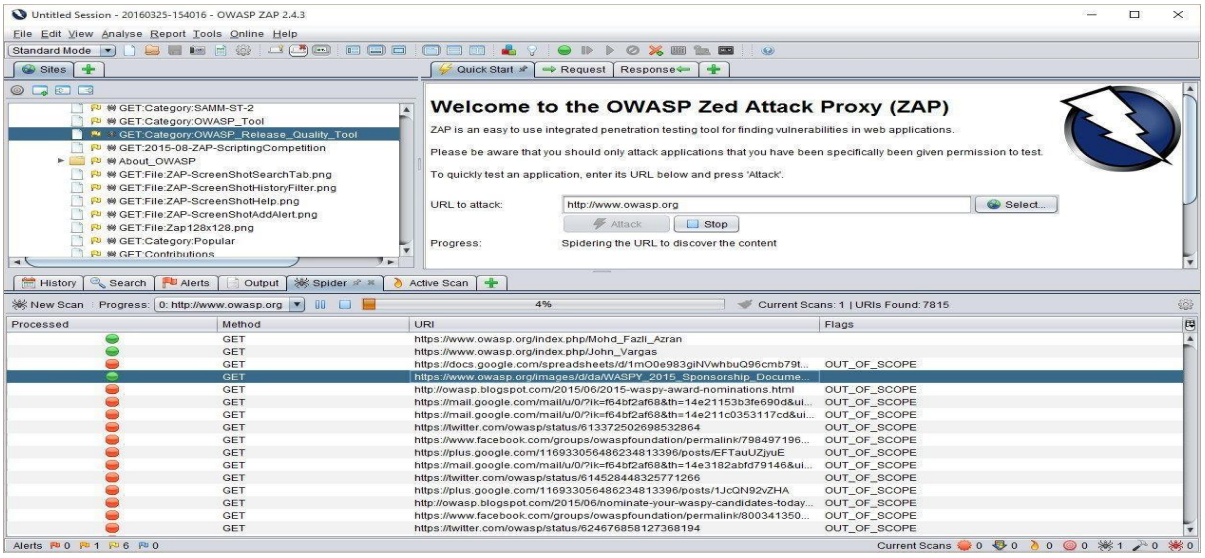
Features:

- Spidering and scanning
- Passive and active scanning
- Intercepting proxy for request

How to Use:

1. Run ZAP.
2. Set browser proxy to 127.0.0.1:8080.
3. Browse your app to ZAP logs and analyze traffic.

Finding Example:
ZAP flagged a site for missing input validation leading to a reflected XSS issue



Q17. Run broken access control test

📌 **Link**: [PortSwigger Access Control]

Broken access control let unauthorized user perform restricted action (like editing another users profile

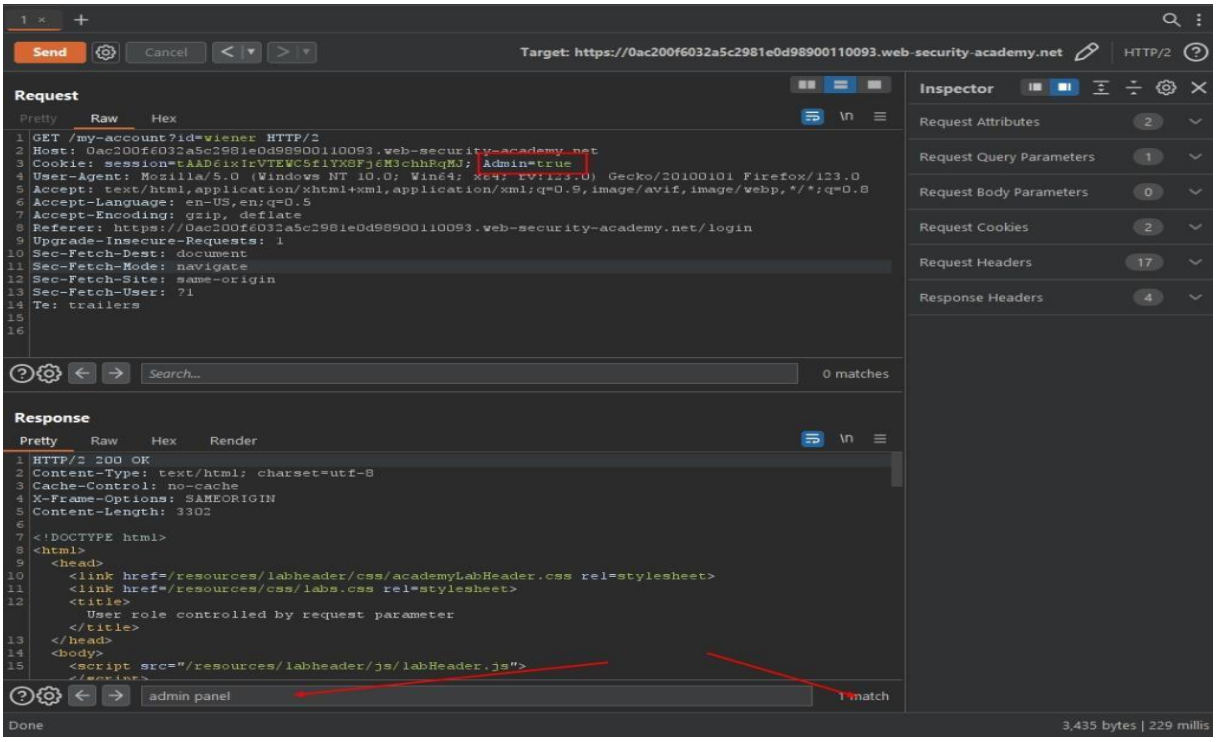
How to Test:

- Try using lower privilege account to perform admin task
- Modify API request (e.g., change user ID, role).

Example:
Changing `user_role=viewer` to `admin` in the request let an attacker change site settings.

Prevention:

- Use server-side authorization.
- Apply role-based access consistently.



Q18. Simulate Local File Inclusion (LFI)

📌 **Link**: [TryHackMe LFI Room]

LFI is a vulnerability where attacker include local file from the server (e.g., `/etc/passwd`).

Real Risk:
Can lead to:

- Disclosure of sensitive file.
- Code execution if log or upload are included.

Mitigation:

- Validate and sanitize file path
- Use whitelisting and path restriction.

```
root@esp01:/proc/16601/fd# ls -al
total 0
dr-x------ 2 root     root      0 Sep 20 11:11 .
dr-xr-xr-x 9 www-data www-data  0 Sep 20 11:11 ..
lr-x------ 1 root     root     64 Sep 20 11:11 0 -> /dev/null
l-wx------ 1 root     root     64 Sep 20 11:11 1 -> /dev/null
lrwx------ 1 root     root     64 Sep 20 11:11 10 -> 'anon_inode:[eventpoll]'
l-wx------ 1 root     root     64 Sep 20 11:11 2 -> /var/log/apache2/error.log
lrwx------ 1 root     root     64 Sep 20 11:11 3 -> 'socket:[215759]'
lrwx------ 1 root     root     64 Sep 20 11:11 4 -> 'socket:[215760]'
lr-x------ 1 root     root     64 Sep 20 11:11 5 -> 'pipe:[215797]'
l-wx------ 1 root     root     64 Sep 20 11:11 6 -> 'pipe:[215797]'
l-wx------ 1 root     root     64 Sep 20 11:11 7 -> /var/log/apache2/other_vhosts_access.log
l-wx------ 1 root     root     64 Sep 20 11:11 8 -> /var/log/apache2/access.log
lrwx------ 1 root     root     64 Sep 20 11:11 9 -> '/tmp/.ZendSem.MpWLtS (deleted)'
```

Q19. Test clickjacking attack

📌 **Link**: [OWASP Clickjacking Page]

Clickjacking trick user into clicking hidden element, like a Buy button behind a image.

How it Works:

- Attacker embed the target page in an iframe.
- Uses CSS to make it transparent.
- Victim click unknowingly.

Defense:

- Use `X-Frame-Options: DENY` or `SAMEORIGIN` header.
- Content Security Policy (CSP) to disallow framing.

Example:
Facebook previously suffered from clickjacking that forced user to like page without knowing

Q20. Practice HTTP Request Smuggling

📌 **Link**: [PortSwigger Request Smuggling]

Request Smuggling manipulate how HTTP request are parsed between frontend and backend server

Vulnerability:

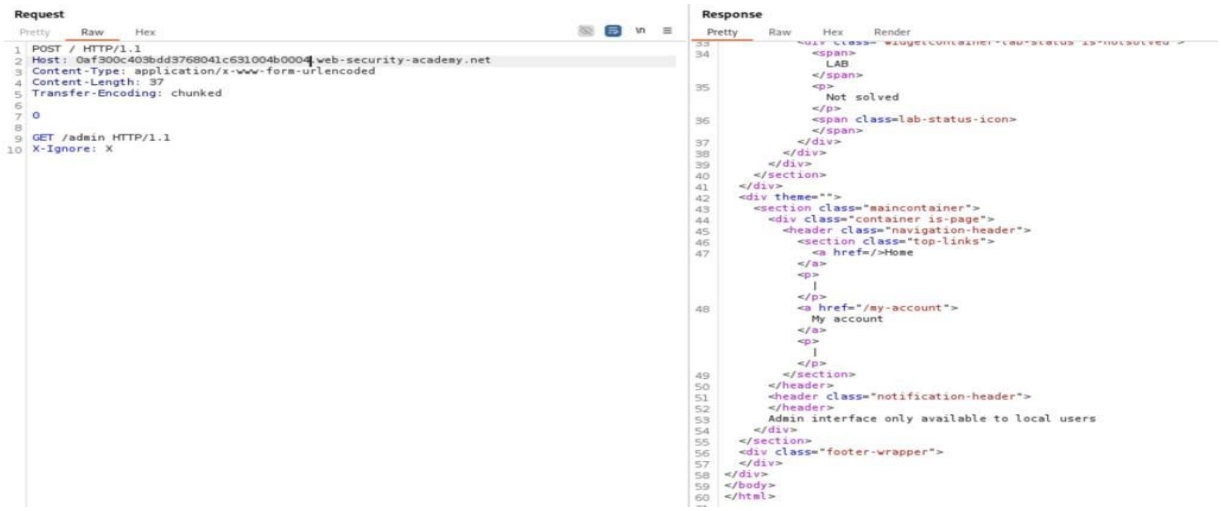
- Occur due to difference in interpreting `ContentLength` and `TransferEncoding` header

Impact:

- Bypass security control.
- Deliver malicious requests hidden inside other.

Prevention:

- Disable `Transfer-Encoding` unless necessary.
- Normalize header parsing on all servers.



SECTION B

Q21. Submit file to VirusTotal

📌 **Link**: [VirusTotal]

VirusTotal is a free online platform that scans suspicious files, URL, and IPs using 70+ antivirus engine

How it works:

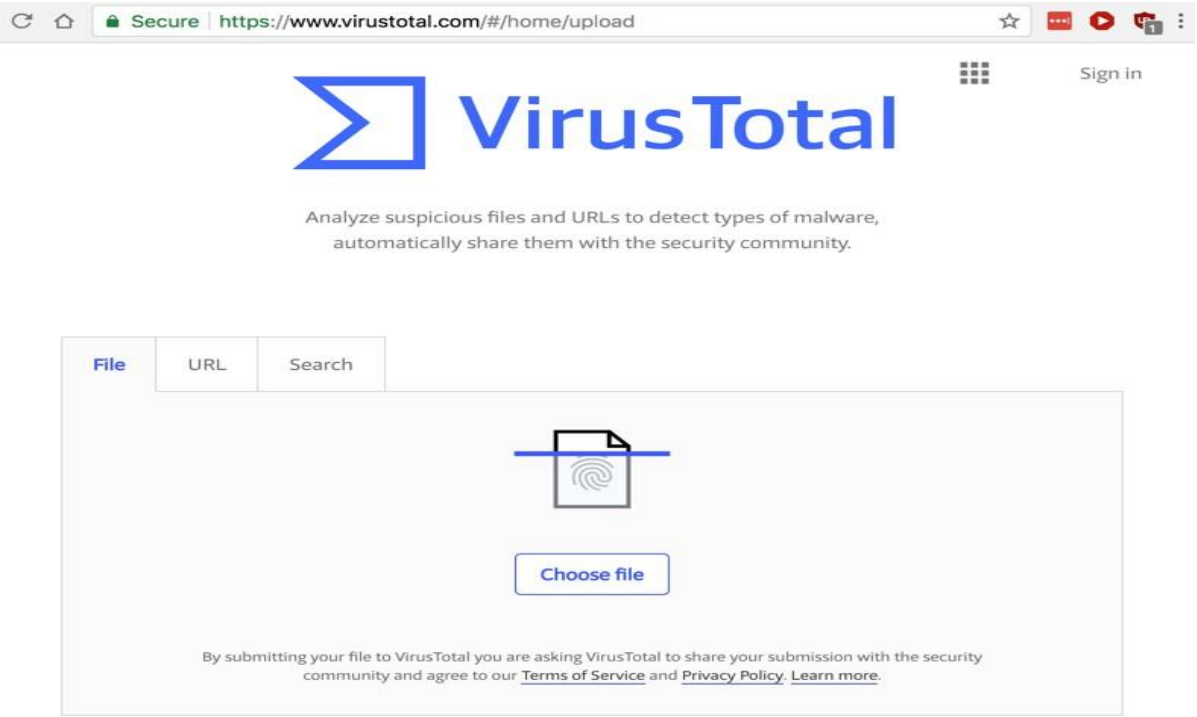
- Upload a file or enter a URL.
- VirusTotal runs it through multiple scanner (tlike Kaspersky, Avast, Bitdefender).
- Show detection rate, file behavior, community commen

Use Case:

- A user received a suspiciou PDF through email.
- Uploaded to VirusTotal to detected as malware by 25 engine.
- Helped in identifying a phishing attack.

Tool Use:

1. Go to virustotal.com
2. Click Choose file → Upload
3. Review analysi and threat label



Q22. Analyze phishing email

📌 **Link**: [TryHackMe Phishing Lab]

Phishing email trick users into clicking maliciou link or giving away credential

How to Analyze:

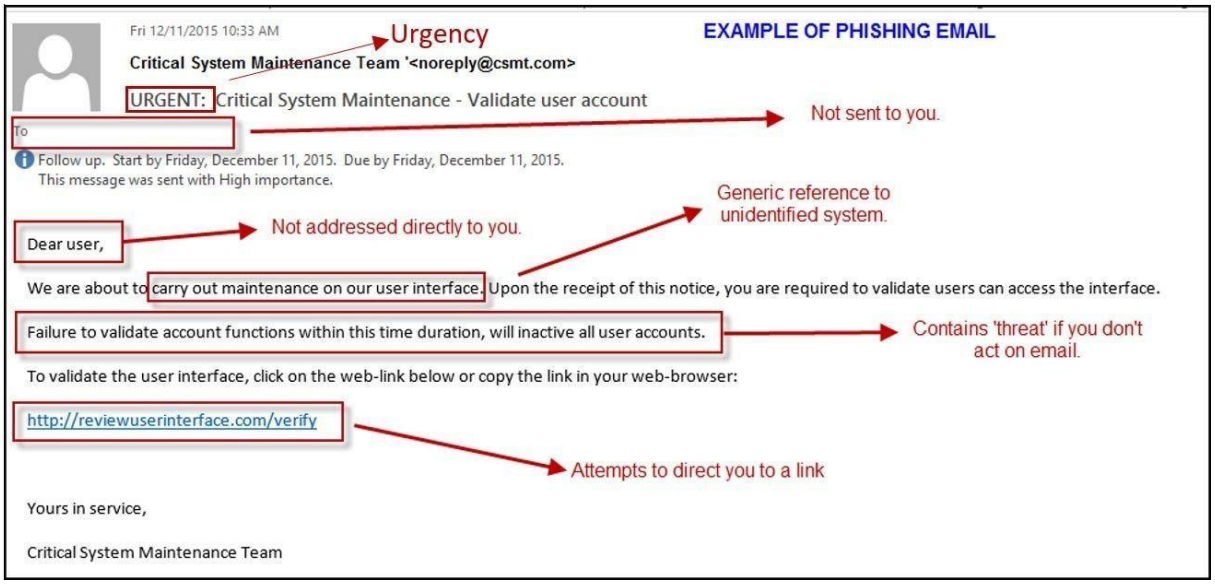
- Check sender email is it spoofed??
- Hover on links do they match the displayed URL.
- Look for urgeny, grammar mistake, and fake logo

Example:
Email claims your PayPal account is suspended and asks to log in. But the link points to `paypal-login.com`.

TryHackMe Lab:

- Simulate real phishing inbox.
- Train to detect fake, suspiciou header, and attachment

Tip: Always check full email headers and don not download random file



Q23. Perform static analysis on malware sample

📌 **Link**: [Hybrid Analysis]

Static analysi inspects code or files **without running them** useful for malware detection.

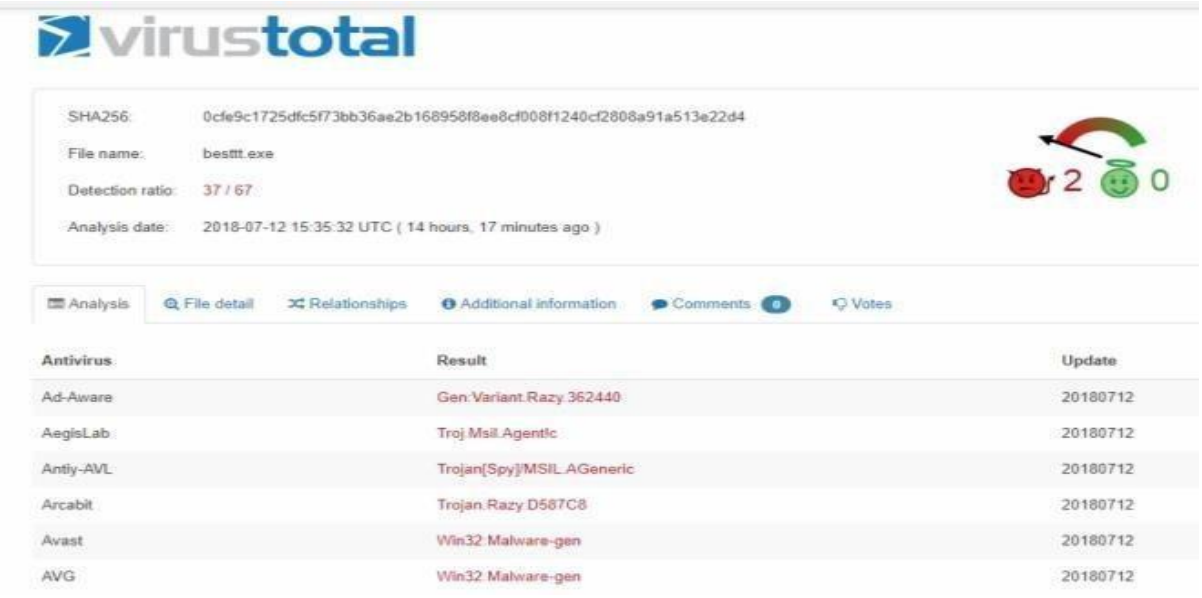
Hybrid Analysis:

- Free service by CrowdStrike
- Upload file to It analyze API, string, behavior.
- Shows if it is ransomware keylogger, or trojan.

How to Use:

1. Go to hybrid analysis.com
2. Upload suspicious file.
3. View score, classification, and system calls.

Example:
An .exe file showed suspicious Window API calls like `GetProcAddress`, often used by malware loader



Q24. Practice ransomware detection lab

📌 **Link**: [CyberTalents Challenges]

Ransomware lock your data and demand payment. Detecting early sign is crucial

Signs of Infection:

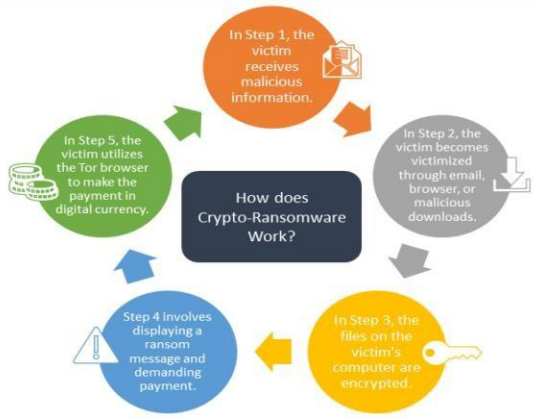
- Files suddenly encrypted (.locked, .abc extensions).
- Ransom note on desktop.
- System slowdown or suspicious processe (e.g., `wannacry.exe`).

CyberTalents Lab:

- Teach how to find ransomware indicator
- Helps identify file change registry entry and network behavior.

Defense:

- Keep backup
- Use behaviour based antiviru
- Disable macros in Office file



Crypto-Ransomware Facts

1. It doesn't steal but rather renders it impossible for users to access information.
2. It spreads through targeted email-based phishing campaigns.
3. Detection is not the solution as it won't restore lost data.
4. It is the favored attack tool for hackers because it is easy to produce and there are a number of well-documented cryptographic libraries available.

A screenshot shows a WannaCry ransomware executable.

Q25. Use PEStudio for malware PE file

📌 **Link**: [PEStudio]

PEStudio analyse Window executable file (PE format) without running them.

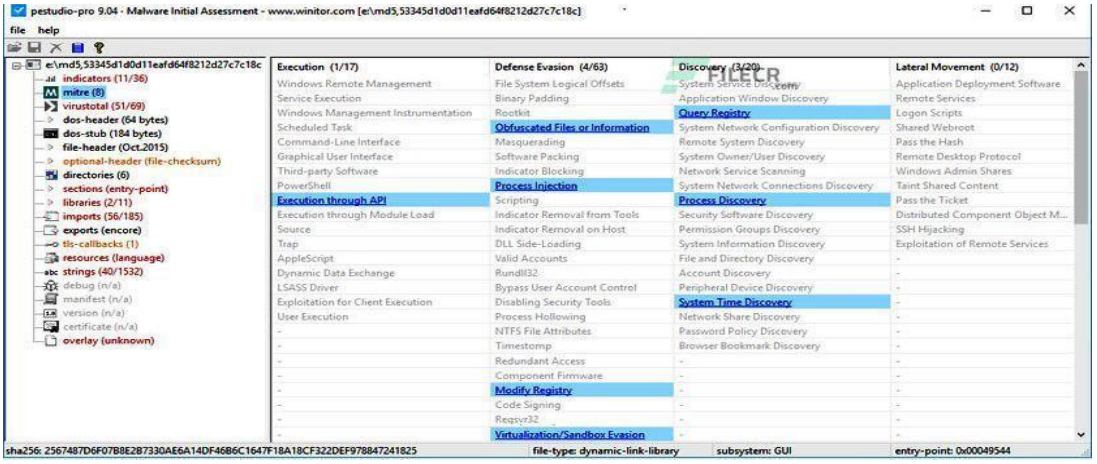
Features:

- Extract metadata, API call, import, entropy level
- Flag suspiciou element like `CreateRemoteThread`, `VirtualAlloc`.

Usage:

1. Download and run PEStudio
2. Load a `.exe` or `.dll` file
3. Review suspiciou indicator and virus score

Example:
A fake Chrome installer had network related import and high entropy → flaggd as malware



Q26. Simulate keylogger detection

📌 **Link**: [Keylogger GitHub]

Keylogger capture keystroke to steal password and message

Detection Methods:

- Monitor unknown running process
- Use Task Manager or tool like Process Explore
- Look for strange file (e.g., `log.txt`, hidden .py script).

GitHub Demo:

- Contain simple Python keylogger.
- Can be tested safely in lab environment to understand behavior.

Prevention:

- Use antiviru with keylogger detection
- Avoid downloading software from unknown source

```
outLog > keyLogger-report-2022-03-04-220316-2022-03-04-220326.log
1    [shift {key down}]
2    [T {key down}]
3    [shift {key up}]
4    [t {key up}]
5    [h {key down}]
6    [h {key up}]
7    [i {key down}]
8    [i {key up}]
9    [s {key down}]
10   [s {key up}]
11   [space {key down}]
12   [space {key up}]
13   [i {key down}]
14   [i {key up}]
15   [s {key down}]
16   [s {key up}]
17   [space {key down}]
18   [space {key up}]
19   [a {key down}]
20   [a {key up}]
21   [space {key down}]
22   [space {key up}]
23   [t {key down}]
24   [e {key down}]
25   [t {key up}]
26   [e {key up}]
27   [s {key down}]
28   [s {key up}]
29   [t {key down}]
30   [t {key up}]
31
```

Q27. View system process using Sysinternals

📌 **Link**: Sysinternals

Sysinternals Suite is a set of advanced tools for Window ystem monitoring.

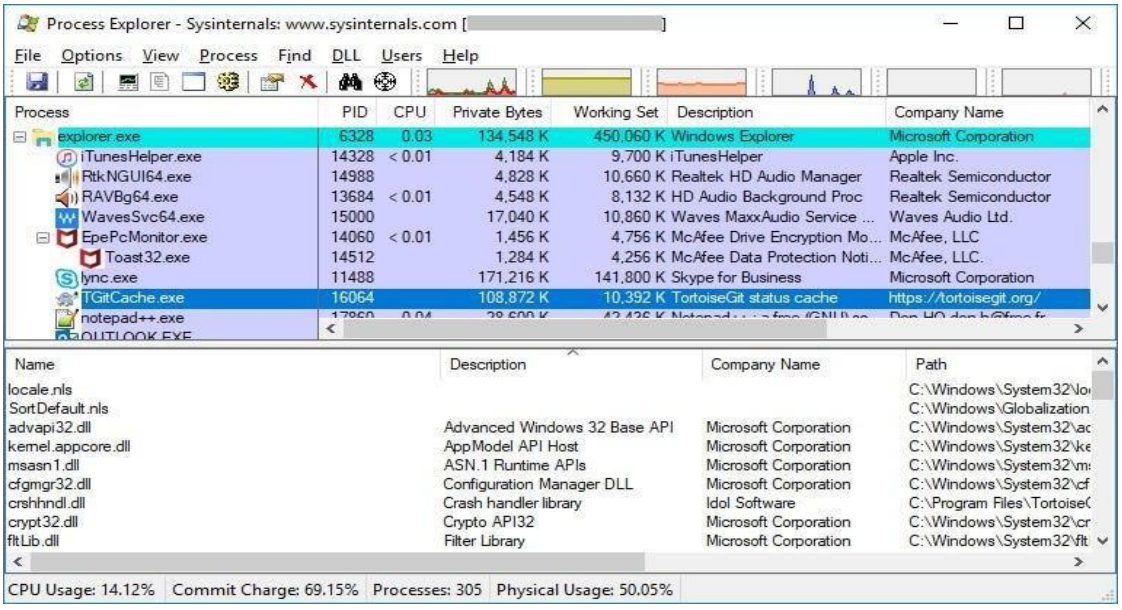
Popular Tools:

- **Process Explorer**: Like Task Manager, but deeper.
- **Autoruns**: Shows program that auto start.
- **TCPView**: Monitor network connection

Use Case:
Use to detect hidden crypto miner and unauthorized admin tools running in the background.

How to Use:

- Download suite → Run `procexp.exe`
- Inspect CPU/memory usage and verify signed executable.



Q28. Analyze malicious script

📌 **Link**: [MalwareJS]

MalwareJS is a platform to analyze **malicious JavaScript** often found in phishing page or ad.

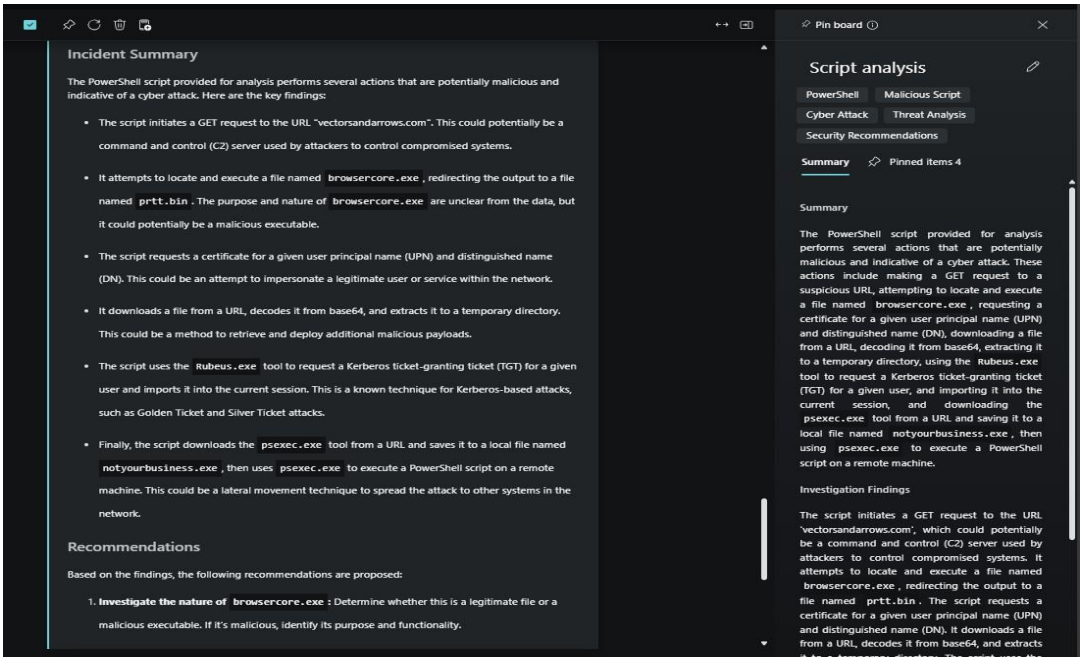
Steps:

- Paste suspicious JS code.
- Review what it does:-redirection, keylogging, data exfiltration.

What to Look For:

- Obfuscated code like `eval(atob(...)`
- Unusual network requests (XHR or fetch)
- DOM manipulation to hide malicious behavior

Example:
A phishing site used JS to capture login form input and send it to a third party server.



Incident Summary

The PowerShell script provided for analysis performs several actions that are potentially malicious and indicative of a cyber attack. Here are the key findings:

- The script initiates a GET request to the URL "vectorsandarrows.com". This could potentially be a command and control (C2) server used by attackers to control compromised systems.
- It attempts to locate and execute a file named `browsercore.exe`, redirecting the output to a file named `prtt.bin`. The purpose and nature of `browsercore.exe` are unclear from the data, but it could potentially be a malicious executable.
- The script requests a certificate for a given user principal name (UPN) and distinguished name (DN). This could be an attempt to impersonate a legitimate user or service within the network.
- It downloads a file from a URL, decodes it from base64, and extracts it to a temporary directory. This could be a method to retrieve and deploy additional malicious payloads.
- The script uses the `Rubeus.exe` tool to request a Kerberos ticket-granting ticket (TGT) for a given user and imports it into the current session. This is a known technique for Kerberos-based attacks, such as Golden Ticket and Silver Ticket attacks.
- Finally, the script downloads the `psexec.exe` tool from a URL and saves it to a local file named `notyourbusiness.exe`, then uses `psexec.exe` to execute a PowerShell script on a remote machine. This could be a lateral movement technique to spread the attack to other systems in the network.

Recommendations

Based on the findings, the following recommendations are proposed:

1. **Investigate the nature of `browsercore.exe`:** Determine whether this is a legitimate file or a malicious executable. If it's malicious, identify its purpose and functionality.

Pin board

Script analysis

PowerShell | Malicious Script | Cyber Attack | Threat Analysis | Security Recommendations

Summary | Pinned items 4

Summary

The PowerShell script provided for analysis performs several actions that are potentially malicious and indicative of a cyber attack. These actions include making a GET request to a suspicious URL, attempting to locate and execute a file named `browsercore.exe`, requesting a certificate for a given user principal name (UPN) and distinguished name (DN), downloading a file from a URL, decoding it from base64, extracting it to a temporary directory, using the `Rubeus.exe` tool to request a Kerberos ticket-granting ticket (TGT) for a given user, and importing it into the current session, and downloading the `psexec.exe` tool from a URL and saving it to a local file named `notyourbusiness.exe`, then using `psexec.exe` to execute a PowerShell script on a remote machine.

Investigation Findings

The script initiates a GET request to the URL 'vectorsandarrows.com', which could potentially be a command and control (C2) server used by attackers to control compromised systems. It attempts to locate and execute a file named `browsercore.exe`, redirecting the output to a file named `prtt.bin`. The script requests a certificate for a given user principal name (UPN) and distinguished name (DN). It downloads a file from a URL, decodes it from base64, and extracts

---

Q29. Detect rootkits with rkhunter

📌 **Link**: [rkhunter GitHub]

Rootkit hide malware by modifying the OS kernel.`rkhunter` helps detect them on Linux

Installation & Scan:

```
bash
CopyEdit
sudo apt install rkhunter
sudo rkhunter --check
```

Checks Done:

- File permission
- Suspiciou binarie
- Hidden process and port

Output Example:
Warn if `/usr/bin/ls` is modified or if hidden login user exist.

Best Use:
Combine with log monitoring and integrity checker (like `aide`).

```
tree@house:~$ sudo rkhunter --check
[sudo] password for tree:
[ Rootkit Hunter version 1.4.2 ]

Checking system commands...

  Performing 'strings' command checks
    Checking 'strings' command                          [ OK ]

  Performing 'shared libraries' checks
    Checking for preloading variables                   [ None found ]
    Checking for preloaded libraries                    [ None found ]
    Checking LD_LIBRARY_PATH variable                   [ Not found ]

  Performing file properties checks
    Checking for prerequisites                          [ Warning ]
    /usr/sbin/adduser                                   [ OK ]
    /usr/sbin/chroot                                    [ OK ]
    /usr/sbin/cron                                      [ OK ]
    /usr/sbin/groupadd                                  [ OK ]
    /usr/sbin/groupdel                                  [ OK ]
    /usr/sbin/groupmod                                  [ OK ]
    /usr/sbin/grpck                                     [ OK ]
    /usr/sbin/nologin                                   [ OK ]
    /usr/sbin/pwck                                      [ OK ]
    /usr/sbin/rsyslogd                                  [ OK ]
    /usr/sbin/useradd                                   [ OK ]
    /usr/sbin/userdel                                   [ OK ]
```

Q30. Use ClamAV to scan for malware

📌 **Link**: ClamAV

ClamAV is a free open source antiviru engine for detecting viruse, trojan, and malware in Linux system.

Installation & Usage:

```
bash
CopyEdit
sudo apt install clamav
sudo freshclam          # Update definitions
sudo clamscan -r /home
```

Features:

- Scan file, email, and web traffic
- Can be scheduled through cron
- Work with mail server for attachment scanning

Example:
Used to clean up a compromised web server infected with PHP based backdoor

```
Software settings
-----------------
Version: 0.98.6
Optional features supported: MEMPOOL IPv6 AUTOIT_EA06 BZIP2 RAR JIT

Database information
--------------------
Database directory: /var/lib/clamav
daily.cvd: version 20071, sigs: 1327903, built on Tue Feb 17 21:48:25 2015
bytecode.cld: version 246, sigs: 42, built on Thu Feb 12 22:13:36 2015
main.cvd: version 55, sigs: 2424225, built on Tue Sep 17 16:57:28 2013
Total number of signatures: 3752170

Platform information
--------------------
uname: Linux 3.18.2-2-ARCH #1 SMP PREEMPT Fri Jan 9 07:37:51 CET 2015 x86_64
OS: linux-gnu, ARCH: x86_64, CPU: x86_64
zlib version: 1.2.8 (1.2.8), compile flags: a9
Triple: x86_64-unknown-linux-gnu
CPU: i686, Little-endian
platform id: 0x0a214f4f0804090201040902
```