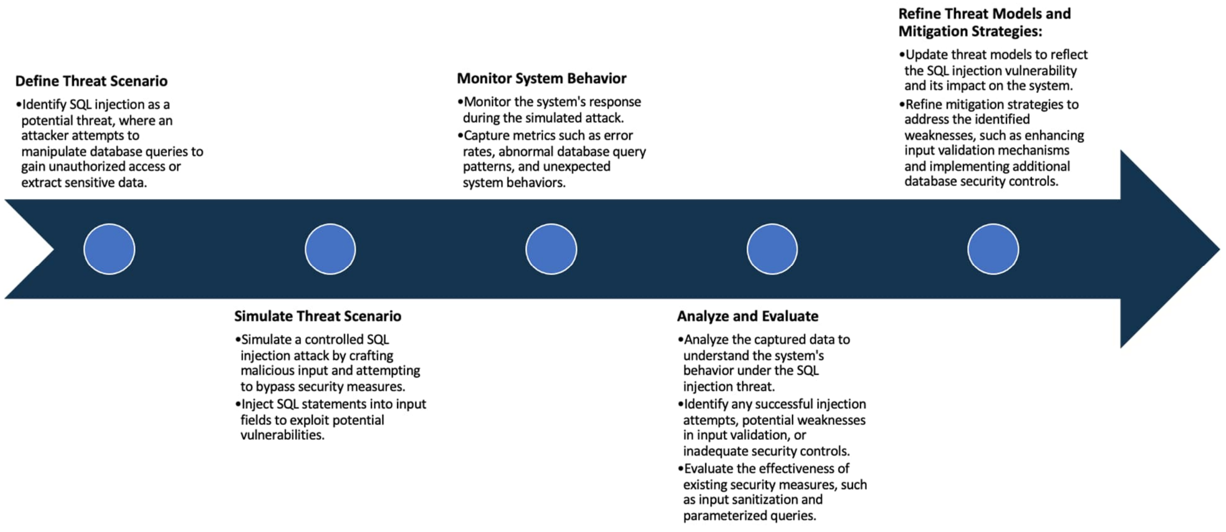
Chaos Engineering – Hypothesis

Name:	GOWRAV KUMR BAITHARU ARIPAKA
Lab User ID:	23SEK3324_U02
Date:	14-01-2024
Application Name:	Data Dog – Vulnerable Java Application

Follow the below guidelines:

Define Hypotheses and Scenarios:	Inject Controlled Failure:	Measure System Behavior:	Learn and Iterate:
• Similar to Chaos Engineering, start by defining hypotheses and scenarios related to potential threats.	• Introduce controlled failure scenarios that mimic potential attack vectors or vulnerabilities identified in Threat Modeling. • Simulate failure conditions, such as network disruptions, component failures, or data breaches, to observe the system's response.	• Capture and measure relevant system behavior metrics during the chaos experiments. • Monitor the system's response to the injected failures, including performance metrics, error rates, and security-related indicators. • Analyze and compare the observed behavior against the expected outcomes defined in the Threat Modeling process.	• Learn from the results of the chaos experiments and iterate on the Threat Modeling process. • Analyze the observations and insights gained from the chaos experiments to refine the Threat Models. Update threat scenarios, adjust mitigation strategies, and improve security controls based on the lessons learned.



Define Threat Scenario

•Identify SQL injection as a potential threat, where an attacker attempts to manipulate database queries to gain unauthorized access or extract sensitive data.

Simulate Threat Scenario

•Simulate a controlled SQL injection attack by crafting malicious input and attempting to bypass security measures.
•Inject SQL statements into input fields to exploit potential vulnerabilities.

Monitor System Behavior

•Monitor the system's response during the simulated attack.
•Capture metrics such as error rates, abnormal database query patterns, and unexpected system behaviors.

Analyze and Evaluate

•Analyze the captured data to understand the system's behavior under the SQL injection threat.
•Identify any successful injection attempts, potential weaknesses in input validation, or inadequate security controls.
•Evaluate the effectiveness of existing security measures, such as input sanitization and parameterized queries.

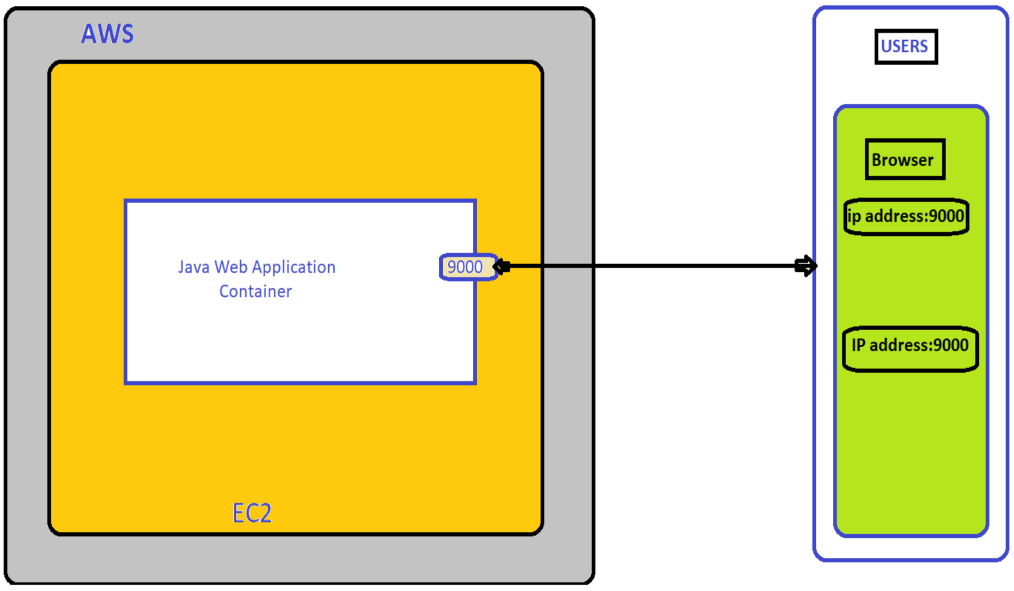
Refine Threat Models and Mitigation Strategies:

•Update threat models to reflect the SQL injection vulnerability and its impact on the system.
•Refine mitigation strategies to address the identified weaknesses, such as enhancing input validation mechanisms and implementing additional database security controls.

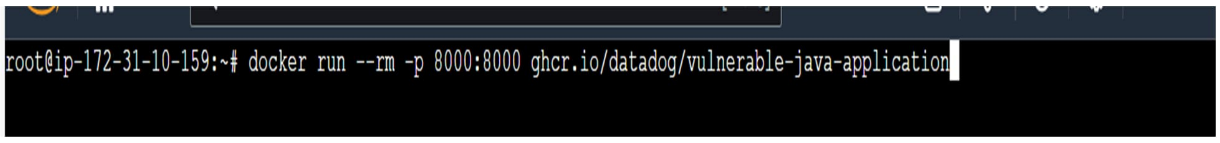
Chaos Engineering – Hypothesis

System Architecture:

(Understand the system and document the physical and logical architecture of the system, use the shapes and icons to capture the system architecture)



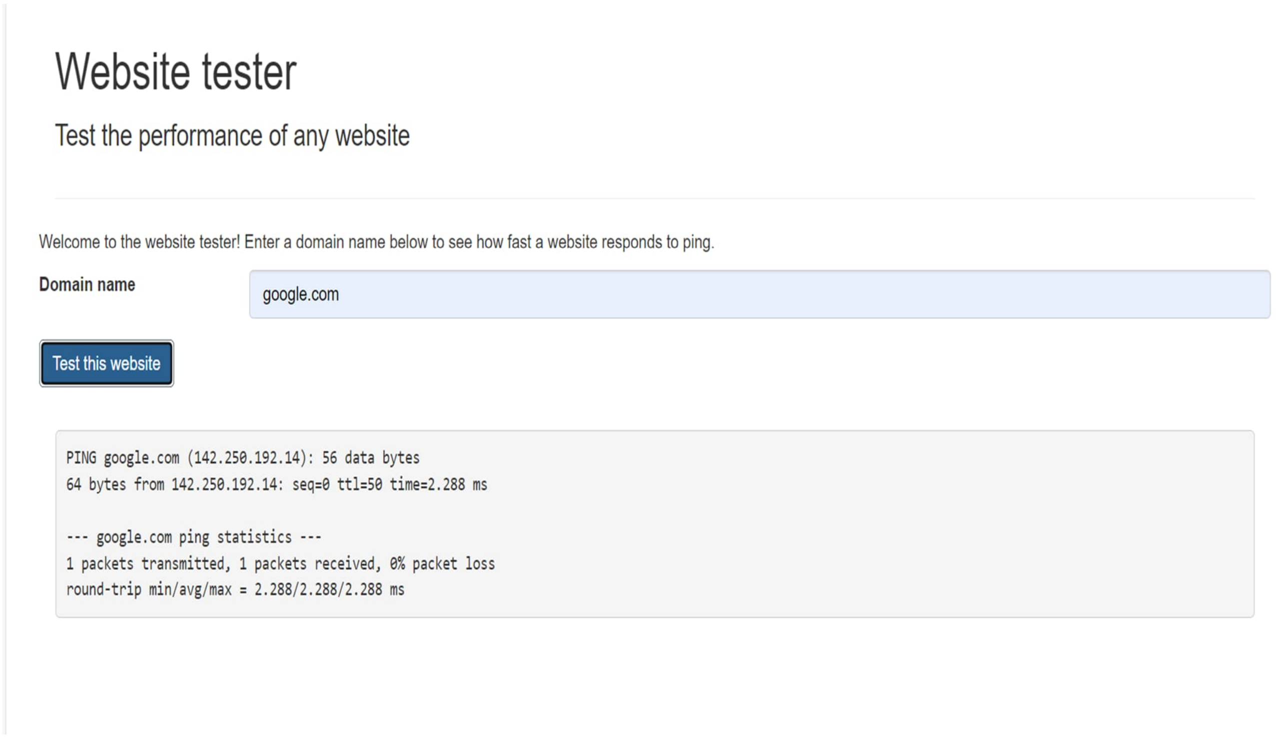
Command:



```
root@ip-172-31-10-159:~# docker run --rm -p 8000:8000 ghcr.io/datadog/vulnerable-java-application
```

After performing this command we can access the application in the wesite. Use public IP address and open port 8000. < IP address:8000 >

Chaos Engineering – Hypothesis


Website tester
Test the performance of any website
Welcome to the website tester! Enter a domain name below to see how fast a website responds to ping.
Domain name
google.com
Test this website

```
PING google.com (142.250.192.14): 56 data bytes
64 bytes from 142.250.192.14: seq=0 ttl=50 time=2.288 ms

--- google.com ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max = 2.288/2.288/2.288 ms
```

The Java application you have allows users to perform live website tests through a user-friendly interface. The main feature involves a domain name search bar where users can input the URL of a live website. Upon entering a domain and clicking the "test the website" button, the application initiates a ping to the specified domain.

Ping is a network utility that sends a message to the target domain and measures the time it takes for a response, providing valuable information about the website's connectivity and responsiveness.

For example, if you input "google.com" and initiate the test, the application sends a ping to Google's servers and displays the ping result. This information is crucial for assessing the health and performance of a live application. The application's user-friendly design simplifies the testing process, making it accessible even for users with limited technical knowledge.

This functionality aids in troubleshooting network issues, identifying potential latency problems, and ensuring that the live application is responsive and accessible. Overall, the Java application serves as a valuable tool for website administrators and developers to assess the status of live applications.

Chaos Engineering – Hypothesis

Define system’s normal behavior:

(Define the steady state of the system is defined, thereby defining some measurable outputs which can indicate the system’s normal behavior)

The web server starts, listening on defined ports like ip address: 8000.

A user accesses a website hosted on this server via a web browser.

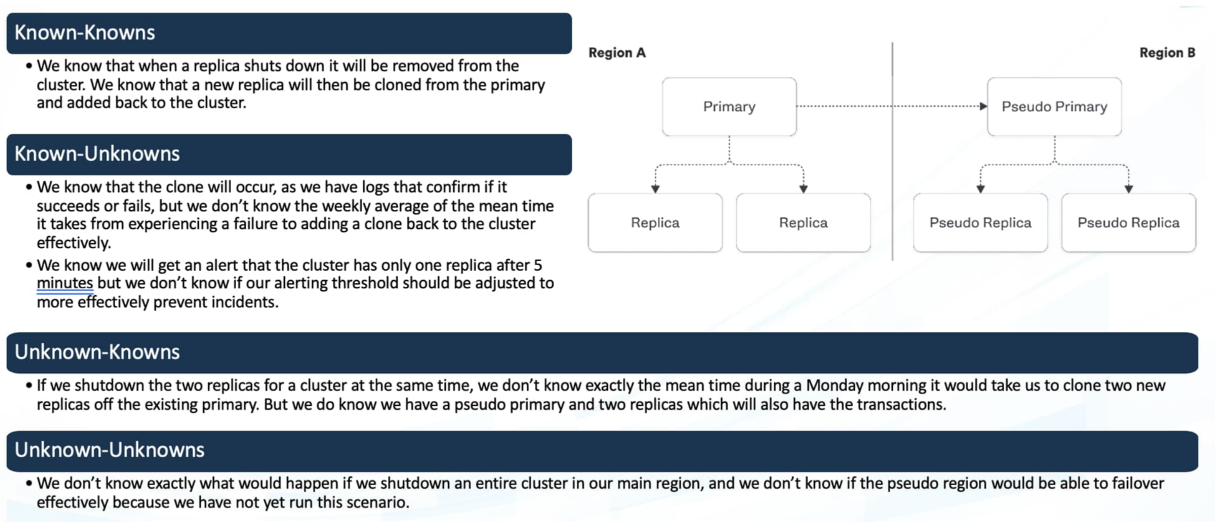
The website is divided into 3 parts. header division, domain search division and the ping result.

If the user gives the correct domain name, then the website start pinging that domain name.

At the ping in division, it will show all the results getting from that website

Hypothesis:

(During an experiment, we need a hypothesis for comparing to a stable control group, and the same applies here too. If there is a reasonable expectation for a particular action according to which we will change the steady state of a system, then the first thing to do is to fix the system so that we accommodate for the action that will potentially have that effect on the system. For eg: "If one of our database servers fails, our service will automatically switch to a backup server, and users will not experience any downtime or data loss.")


Region A
Region B
Primary
Pseudo Primary
Replica
Replica
Pseudo Replica
Pseudo Replica

Known-Knowns

- We know that when a replica shuts down it will be removed from the cluster. We know that a new replica will then be cloned from the primary and added back to the cluster.

Known-Unknowns

- We know that the clone will occur, as we have logs that confirm if it succeeds or fails, but we don’t know the weekly average of the mean time it takes from experiencing a failure to adding a clone back to the cluster effectively.
- We know we will get an alert that the cluster has only one replica after 5 minutes but we don’t know if our alerting threshold should be adjusted to more effectively prevent incidents.

Unknown-Knowns

- If we shutdown the two replicas for a cluster at the same time, we don’t know exactly the mean time during a Monday morning it would take us to clone two new replicas off the existing primary. But we do know we have a pseudo primary and two replicas which will also have the transactions.

Unknown-Unknowns

- We don’t know exactly what would happen if we shutdown an entire cluster in our main region, and we don’t know if the pseudo region would be able to failover effectively because we have not yet run this scenario.

Chaos Engineering – Hypothesis

known	Engineers intentionally shut down a database server during non-peak hours to observe how the system reacts. They expect service degradation or failures in specific functionalities that rely on the database	Engineers test the system's capacity by gradually increasing user load until it reaches and surpasses the expected operational limits. While they know overloading the system might cause issues, the specific breaking point or the precise failure mode might be unknown
unknown	Through load testing, engineers discover that a particular microservice experiences significant latency only when multiple users concurrently access a specific feature.	Engineers randomly throttle network bandwidth or introduce intermittent latency into different parts of the system to simulate unpredictable real-world conditions.
	known	unknown

Chaos Engineering – Hypothesis

Experiment:

(Document your Preparation, Implementation, Observation and Analysis)

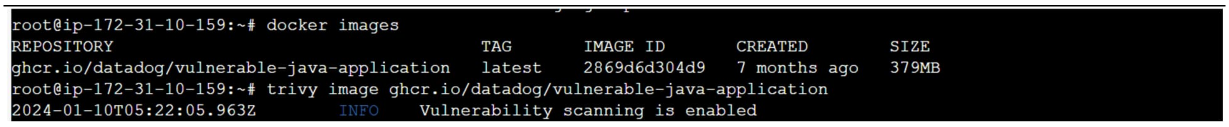
Using some tools for security analysis on this Java Web Testing application.

1. TRIVY
2. NUCLEI
3. OWASAP ZAPs

Observation:

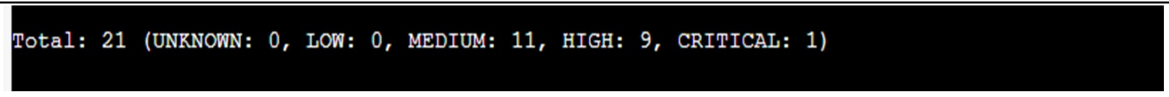
TRIVY: Performing TRIVY

Command:

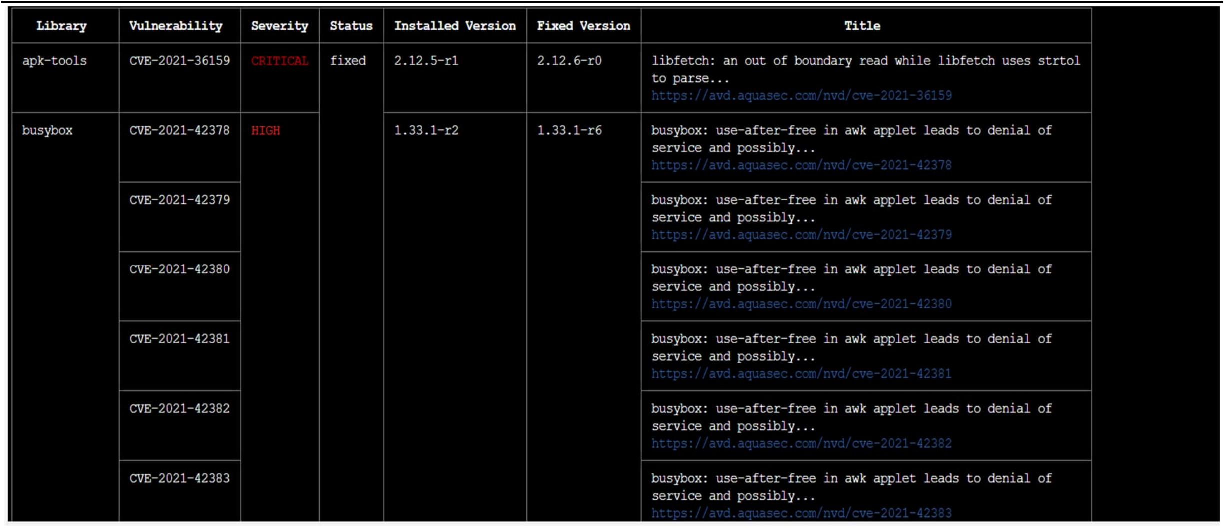


```
root@ip-172-31-10-159:~# docker images
REPOSITORY                                 TAG       IMAGE ID       CREATED        SIZE
ghcr.io/datadog/vulnerable-java-application   latest    2869d6d304d9   7 months ago   379MB
root@ip-172-31-10-159:~# trivy image ghcr.io/datadog/vulnerable-java-application
2024-01-10T05:22:05.963Z        INFO    Vulnerability scanning is enabled
```

Output:



```
Total: 21 (UNKNOWN: 0, LOW: 0, MEDIUM: 11, HIGH: 9, CRITICAL: 1)
```



Library	Vulnerability	Severity	Status	Installed Version	Fixed Version	Title
apk-tools	CVE-2021-36159	CRITICAL	fixed	2.12.5-r1	2.12.6-r0	libfetch: an out of boundary read while libfetch uses strtol to parse... https://avd.aquasec.com/nvd/cve-2021-36159
busybox	CVE-2021-42378	HIGH		1.33.1-r2	1.33.1-r6	busybox: use-after-free in awk applet leads to denial of service and possibly... https://avd.aquasec.com/nvd/cve-2021-42378
	CVE-2021-42379					busybox: use-after-free in awk applet leads to denial of service and possibly... https://avd.aquasec.com/nvd/cve-2021-42379
	CVE-2021-42380					busybox: use-after-free in awk applet leads to denial of service and possibly... https://avd.aquasec.com/nvd/cve-2021-42380
	CVE-2021-42381					busybox: use-after-free in awk applet leads to denial of service and possibly... https://avd.aquasec.com/nvd/cve-2021-42381
	CVE-2021-42382					busybox: use-after-free in awk applet leads to denial of service and possibly... https://avd.aquasec.com/nvd/cve-2021-42382
	CVE-2021-42383					busybox: use-after-free in awk applet leads to denial of service and possibly... https://avd.aquasec.com/nvd/cve-2021-42383

By performing this TRIVY, I have got several security issues.

Explaining some vulnerabilities with solution:

Before July 26, 2021, **libfetch, as utilized in apk-tools, xbps,** and similar products, had a vulnerability. Numeric strings mishandling in FTP and HTTP protocols led to an out-of-bounds read due to insufficient checks in the FTP passive mode implementation, where strtol was used to parse numbers. Premature line ending could occur, causing a for-loop to check for the 0 terminator one byte too late.

Solution: Address the libfetch vulnerability, update the library to a version released after July 26, 2021. This ensures that numeric string parsing in FTP and HTTP protocols is handled securely, preventing potential out-of-bounds reads and addressing the premature line ending issue. Regularly check for and apply software updates to maintain system security.

A **use-after-free vulnerability in Busybox's awk applet allows a crafted awk pattern** to trigger a denial-of-service condition and potentially execute arbitrary code. Exploiting the flaw in the getvar_i function can lead to unintended consequences, compromising the stability and security of the system running Busybox.

Solution: To mitigate the use-after-free vulnerability in Busybox's awk applet, users should promptly update their software to the latest version provided by the vendor. This helps to address the identified issue, incorporating security patches and preventing potential denial-of-service attacks and code execution risks. Regularly monitoring and applying software updates is crucial for maintaining a secure and resilient system.

Pivotal Spring Framework, up to version 5.3.16, is susceptible to a potential remote code execution risk when used for Java deserialization of untrusted data. The vendor asserts that untrusted data usage is not intended, and the product behavior won't change as some users depend on deserialization of trusted data. Authentication may be necessary, and the impact depends on the specific product's implementation of the library.

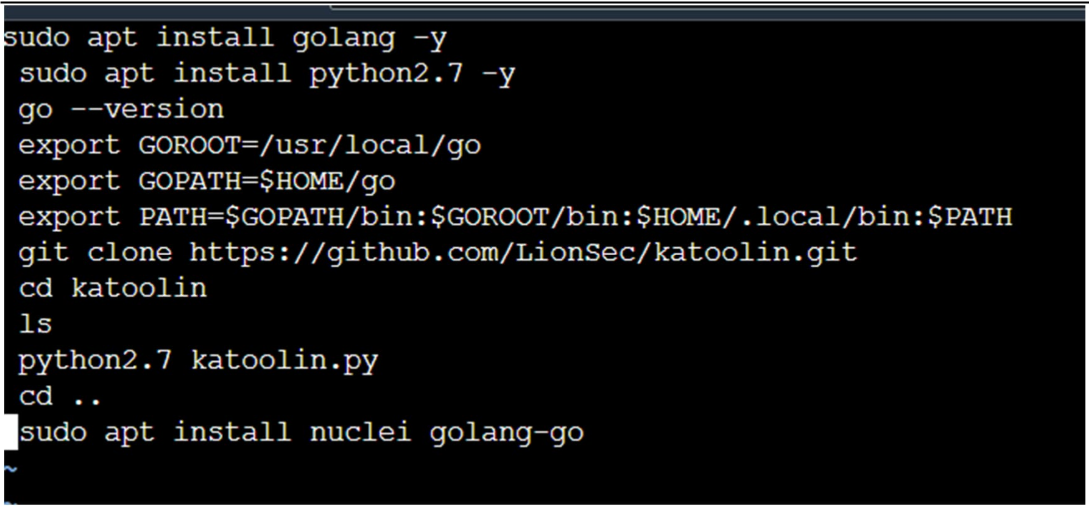
Solution: Pivotal Spring Framework, up to version 5.3.16, is susceptible to a potential remote code execution risk when used for Java deserialization of untrusted data. The vendor asserts that untrusted data usage is not intended, and the product behavior won't change as some users depend on deserialization of trusted data. Authentication may be necessary, and the impact depends on the specific product's implementation of the library.

SnakeYAML's vulnerability in the Constructor() class allows deserialization of arbitrary types, posing a remote code execution risk when processing attacker-supplied YAML content. To mitigate this, users are advised to employ SnakeYAML's SafeConstructor for parsing untrusted content, restricting deserialization. Upgrading to version 2.0 or later is strongly recommended to ensure the security of YAML processing in applications.

Solution: To address the vulnerability in SnakeYAML, users should adopt SnakeYAML's SafeConstructor when parsing untrusted content. This constrains deserialization, mitigating the risk of remote code execution. Additionally, upgrading to SnakeYAML version 2.0 or later is strongly recommended to ensure the implementation of the latest security enhancements and patches. These measures collectively enhance the security posture of applications utilizing SnakeYAML for YAML processing.

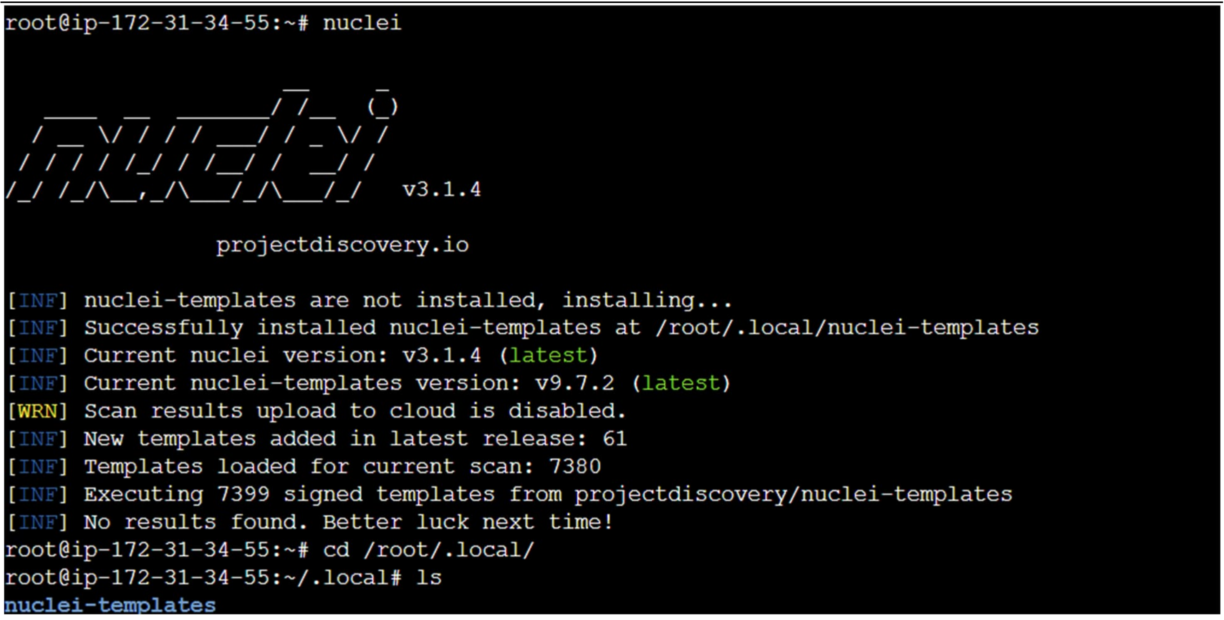
NUCLEI: Performing NUCLEI

Steup the nuclei in your virtual machine using the following commands



```
sudo apt install golang -y
 sudo apt install python2.7 -y
 go --version
 export GOROOT=/usr/local/go
 export GOPATH=$HOME/go
 export PATH=$GOPATH/bin:$GOROOT/bin:$HOME/.local/bin:$PATH
 git clone https://github.com/LionSec/katoolin.git
 cd katoolin
 ls
 python2.7 katoolin.py
 cd ..
 sudo apt install nuclei golang-go
~
```

After setting up check nuclei is installed or not



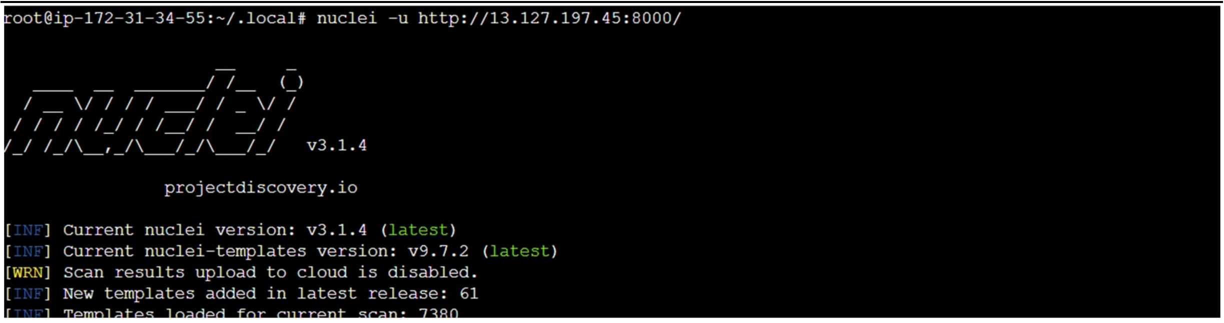
```
root@ip-172-31-34-55:~# nuclei

                     __     _
   ____  __  _______/ /__  (_)
  / __ \/ / / / ___/ / _ \/ /
 / / / / /_/ / /__/ /  __/ /
/_/ /_/\__,_/\___/_/\___/_/   v3.1.4

                projectdiscovery.io

[INF] nuclei-templates are not installed, installing...
[INF] Successfully installed nuclei-templates at /root/.local/nuclei-templates
[INF] Current nuclei version: v3.1.4 (latest)
[INF] Current nuclei-templates version: v9.7.2 (latest)
[WRN] Scan results upload to cloud is disabled.
[INF] New templates added in latest release: 61
[INF] Templates loaded for current scan: 7380
[INF] Executing 7399 signed templates from projectdiscovery/nuclei-templates
[INF] No results found. Better luck next time!
root@ip-172-31-34-55:~# cd /root/.local/
root@ip-172-31-34-55:~/.local# ls
nuclei-templates
```

Using nuclei command perform the test against the live application

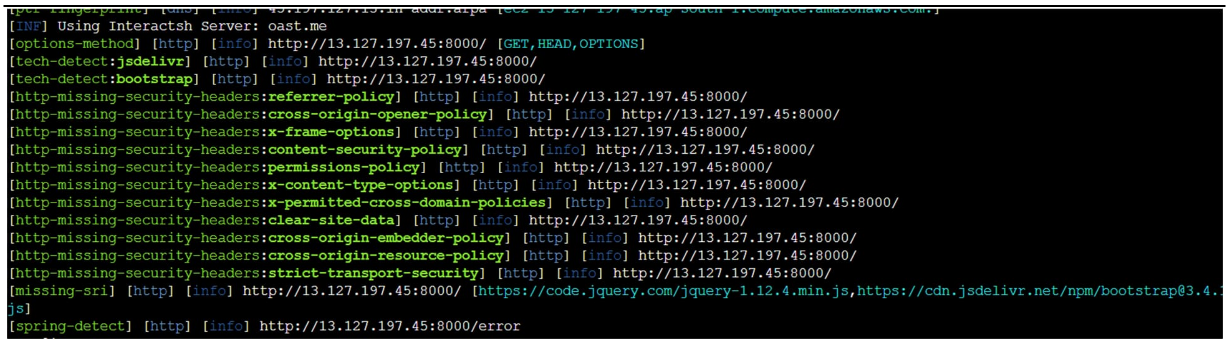


```
root@ip-172-31-34-55:~/.local# nuclei -u http://13.127.197.45:8000/

                     __     _
   ____  __  _______/ /__  (_)
  / __ \/ / / / ___/ / _ \/ /
 / / / / /_/ / /__/ /  __/ /
/_/ /_/\__,_/\___/_/\___/_/   v3.1.4

                projectdiscovery.io

[INF] Current nuclei version: v3.1.4 (latest)
[INF] Current nuclei-templates version: v9.7.2 (latest)
[WRN] Scan results upload to cloud is disabled.
[INF] New templates added in latest release: 61
[INF] Templates loaded for current scan: 7380
```



```
[INF] Using Interactsh Server: oast.me
[options-method] [http] [info] http://13.127.197.45:8000/ [GET,HEAD,OPTIONS]
[tech-detect:jsdelivr] [http] [info] http://13.127.197.45:8000/
[tech-detect:bootstrap] [http] [info] http://13.127.197.45:8000/
[http-missing-security-headers:referrer-policy] [http] [info] http://13.127.197.45:8000/
[http-missing-security-headers:cross-origin-opener-policy] [http] [info] http://13.127.197.45:8000/
[http-missing-security-headers:x-frame-options] [http] [info] http://13.127.197.45:8000/
[http-missing-security-headers:content-security-policy] [http] [info] http://13.127.197.45:8000/
[http-missing-security-headers:permissions-policy] [http] [info] http://13.127.197.45:8000/
[http-missing-security-headers:x-content-type-options] [http] [info] http://13.127.197.45:8000/
[http-missing-security-headers:x-permitted-cross-domain-policies] [http] [info] http://13.127.197.45:8000/
[http-missing-security-headers:clear-site-data] [http] [info] http://13.127.197.45:8000/
[http-missing-security-headers:cross-origin-embedder-policy] [http] [info] http://13.127.197.45:8000/
[http-missing-security-headers:cross-origin-resource-policy] [http] [info] http://13.127.197.45:8000/
[http-missing-security-headers:strict-transport-security] [http] [info] http://13.127.197.45:8000/
[missing-sri] [http] [info] http://13.127.197.45:8000/ [https://code.jquery.com/jquery-1.12.4.min.js,https://cdn.jsdelivr.net/npm/bootstrap@3.4.
js]
[spring-detect] [http] [info] http://13.127.197.45:8000/error
```

The following are the some high vulnerabilities founded using nuclei

Strict-Transport-Security: The HTTP Strict-Transport-Security response header (often abbreviated as HSTS) informs browsers that the site should only be accessed using HTTPS, and that any future attempts to access it using HTTP should automatically be converted to HTTPS.

solution: The HTTP Strict Transport Security header informs the browser that it should never load a site using HTTP and should automatically convert all attempts to access the site using HTTP to HTTPS requests instead.

Cross-Origin Resource Policy (CORP): Cross-Origin Resource Policy is a policy set by the Cross-Origin-Resource-Policy HTTP header that lets websites and applications opt in to protection against certain requests from other origins (such as those issued with elements like <script> and <img>), to mitigate speculative side-channel attacks, like Specter, as well as Cross-Site Script Inclusion attacks.

solution: The Cross-Origin-Embedder-Policy HTTP response header, when used upon a document, can be used to require sub resources to either be same-origin with the document, or come with a Cross-Origin-Resource-Policy HTTP response header to indicate they are okay with being embedded. This is why the cross-origin value exists.

Clear-Site-Data: The Clear-Site-Data header clears browsing data (cookies, storage, cache) associated with the requesting website. It allows web developers to have more control over the data stored by a client browser for their origins.

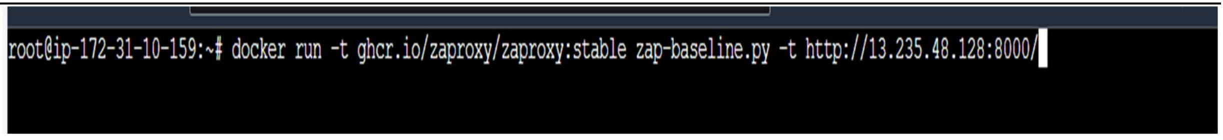
Solution: If a user signs out of your website or service, you might want to remove locally stored data. To do this, add the Clear-Site-Data header to the page that confirms the logging out from the site has been accomplished successfully.

X-Content-Type-Options: The X-Content-Type-Options response HTTP header is a marker used by the server to indicate that the MIME types advertised in the Content-Type headers should be followed and not be changed. The header allows you to avoid MIME type sniffing by saying that the MIME types are deliberately configured.

Solution: Blocks a request if the request destination is of type style and the MIME type is not text/css, or of type script and the MIME type is not a JavaScript MIME type.

OWASP ZAP: Analyzing the application by OWASP ZAP.

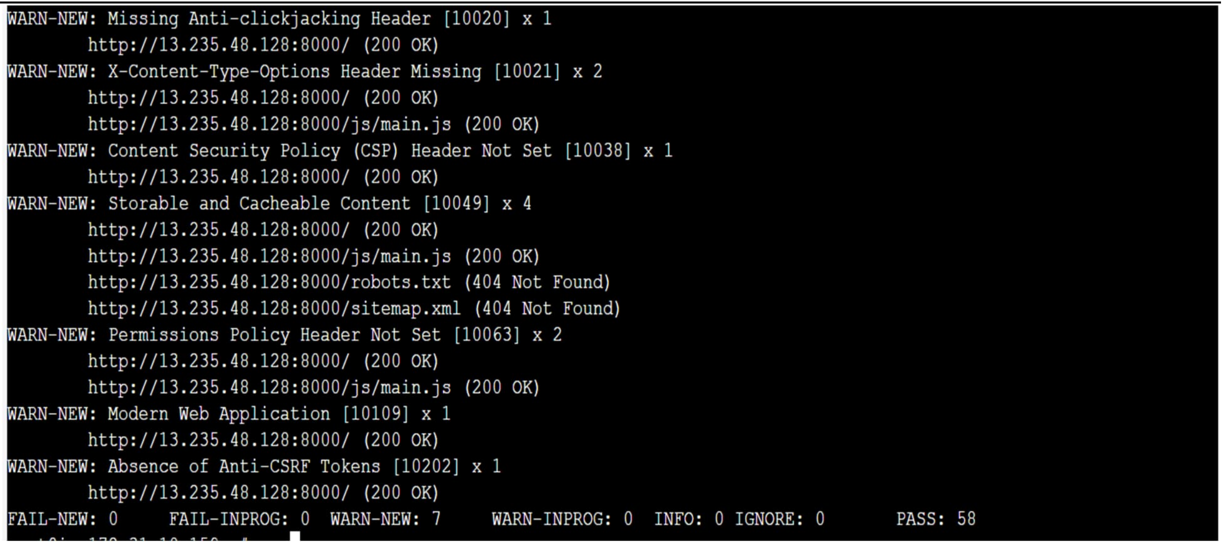
Command:



```
root@ip-172-31-10-159:~# docker run -t ghcr.io/zaproxy/zaproxy:stable zap-baseline.py -t http://13.235.48.128:8000/
```

By using this command, we are performing zap base line command on our live application.

Analysis and result:



```
WARN-NEW: Missing Anti-clickjacking Header [10020] x 1
        http://13.235.48.128:8000/ (200 OK)
WARN-NEW: X-Content-Type-Options Header Missing [10021] x 2
        http://13.235.48.128:8000/ (200 OK)
        http://13.235.48.128:8000/js/main.js (200 OK)
WARN-NEW: Content Security Policy (CSP) Header Not Set [10038] x 1
        http://13.235.48.128:8000/ (200 OK)
WARN-NEW: Storable and Cacheable Content [10049] x 4
        http://13.235.48.128:8000/ (200 OK)
        http://13.235.48.128:8000/js/main.js (200 OK)
        http://13.235.48.128:8000/robots.txt (404 Not Found)
        http://13.235.48.128:8000/sitemap.xml (404 Not Found)
WARN-NEW: Permissions Policy Header Not Set [10063] x 2
        http://13.235.48.128:8000/ (200 OK)
        http://13.235.48.128:8000/js/main.js (200 OK)
WARN-NEW: Modern Web Application [10109] x 1
        http://13.235.48.128:8000/ (200 OK)
WARN-NEW: Absence of Anti-CSRF Tokens [10202] x 1
        http://13.235.48.128:8000/ (200 OK)
FAIL-NEW: 0     FAIL-INPROG: 0  WARN-NEW: 7     WARN-INPROG: 0  INFO: 0 IGNORE: 0       PASS: 58
```

The "**Missing Anti-clickjacking Header" issue (ID: 10020) in**dicates the absence of a proper anti-clickjacking header in the HTTP response. Clickjacking involves tricking users into clicking on malicious elements unknowingly. Implementing headers like X-Frame-Options with appropriate settings, such as "DENY" or "SAMEORIGIN," helps prevent unauthorized embedding of a web page within an iframe, mitigating the risk of clickjacking attacks.

Solution: To address the "Missing Anti-clickjacking Header" issue, implement an appropriate anti-clickjacking header in the HTTP response, such as X-Frame-Options with values like "DENY" or "SAMEORIGIN." This helps prevent unauthorized embedding of the web page within iframes, enhancing security and mitigating the risk of clickjacking attacks. Regularly update and audit security headers for robust protection.

The **"X-Content-Type-Options Header Missing" issue (ID: 10021**) indicates that the HTTP response lacks the X-Content-Type-Options header. This header prevents browsers from interpreting files as a different MIME type than declared by the server, reducing the risk of MIME-sniffing attacks. Implementing this header with the value "nosniff" ensures browsers adhere strictly to the declared content type, enhancing security.

Solution: To address the "X-Content-Type-Options Header Missing" issue (ID: 10021), include the X-Content-Type-Options header in the HTTP response with the value "nosniff." This ensures browsers strictly adhere to the declared content type, preventing MIME-sniffing attacks. Regularly audit and update security headers to maintain a robust defense against potential security vulnerabilities.

The "Content Security Policy (CSP) Header Not Set" issue (ID: 10038) indicates the absence of a Content Security Policy header in the HTTP response. CSP defines rules for browser interactions, mitigating various types of attacks, including cross-site scripting. Implementing CSP headers helps restrict which resources the browser can load, enhancing web application security by preventing unauthorized script execution and minimizing the impact of potential security threats.

Solution: To address the "Content Security Policy (CSP) Header Not Set" issue (ID: 10038), include a Content Security Policy header in the HTTP response. Define and enforce policies specifying trusted sources for scripts, styles, and other resources. Regularly review and update the CSP directives to adapt to evolving security requirements, ensuring a robust defense against cross-site scripting and related threats in web applications.

The "Storable and Cacheable Content" issue (ID: 10049) signifies that certain content is both storable and cacheable. While caching enhances performance, it might expose sensitive information if improperly stored. To resolve, carefully assess the sensitivity of data, implement proper cache-control headers, and use encryption where necessary, ensuring a balance between performance optimization and data security.

Solution: To mitigate the "Storable and Cacheable Content" issue (ID: 10049), carefully assess data sensitivity. Implement appropriate cache-control headers, ensuring that sensitive information is not stored or cached inadvertently. Utilize encryption for sensitive content. Regularly review and update caching policies to align with security requirements, achieving a balance between performance optimization and safeguarding sensitive data.