



Security Tests Automation using Robot Framework

Riddhi Shree

30-SEP-2021

Table of contents

1. SecQAtion - Security Tests Automation using Robot Framework	3
2. Standard Security Testing Approach	9
3. Exercise	13
4. OWASP Top 10 2021	14
5. Inconveniences/Gaps	18
6. Introduction to Robot Framework	20
7. Configuring PyCharm	24
8. Recommended Folder Structure	27
9. Basic Elements of Robot Framework	30
10. Exercise: Structured Reconnaissance in <10 min	36
11. mitmproxy: Intercepting API Requests & Responses	48
12. Pabot: Parallel Processing	60
13. Understanding Docker and Docker Compose	67
14. Dockerizing Selenium Test Execution Environment	75
15. Solution: Headless Browser	76
16. CI/CD Build Pipeline	80
17. Serving test report via S3 and CloudFront	98
18. Implementing authentication via Lambda Edge	115
19. Leveraging HTTPolice	121
20. Basics of Python programming	129
21. Example: Custom Keywords Library	132
22. When and How to Use Burp Suite	135
23. DVWA: Get Your Hands Dirty	136

1. SecQAtion - Security Tests Automation using Robot Framework

1.1 What are we solving?

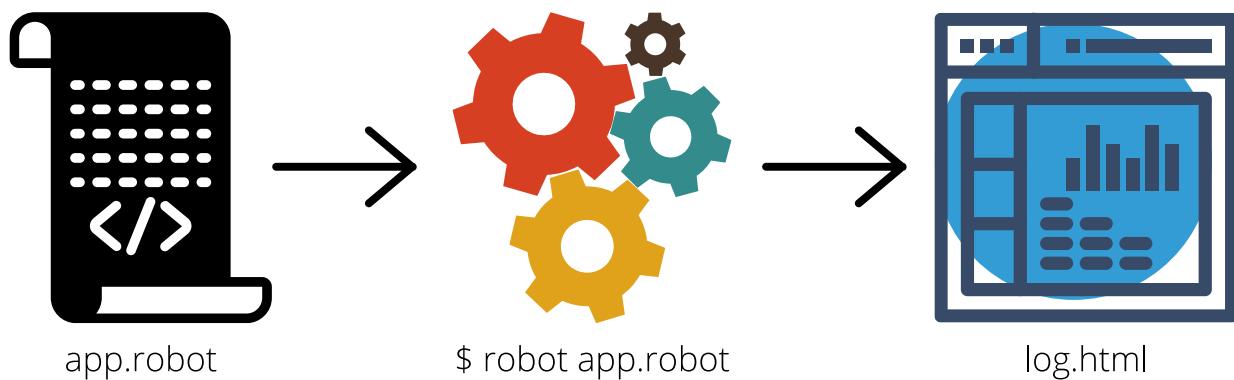
While doing a security assessment:

- * Get rid of boring manual tasks
- * Make your job fun and productive
- * Ensure consistency and repeatability
- * Generate business friendly test reports
- * Let developers incorporate your tests into their build pipeline
- * Adapt and evolve, quickly and easily

1.2 Scenario 1:

BASIC FLOW

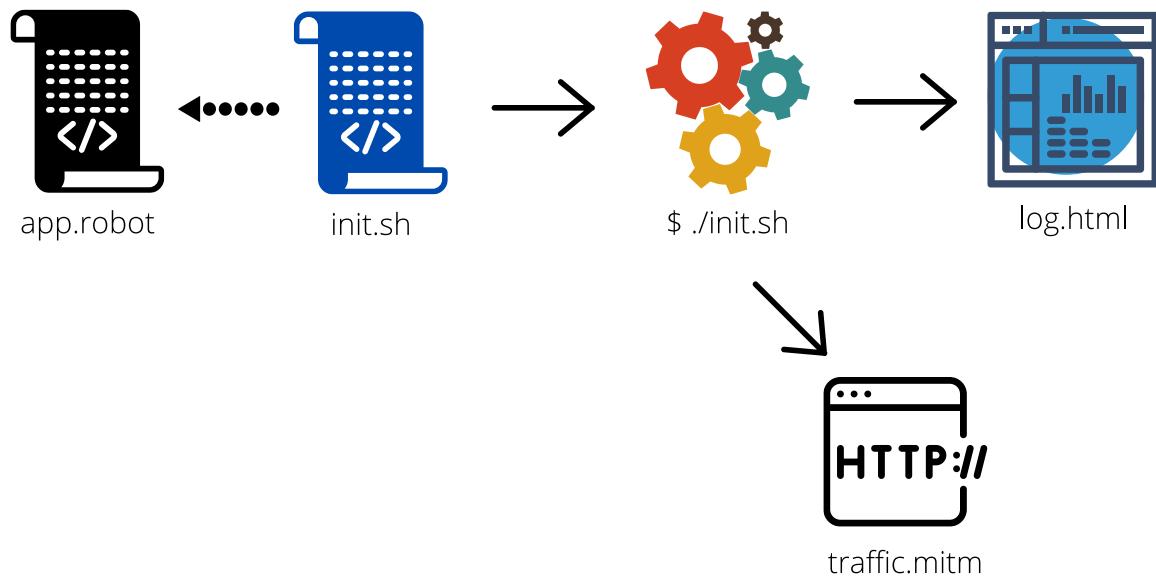
ALL CONFIGURATIONS ARE LOCAL



1.3 Scenario 2:

INTRODUCING MITMProxy

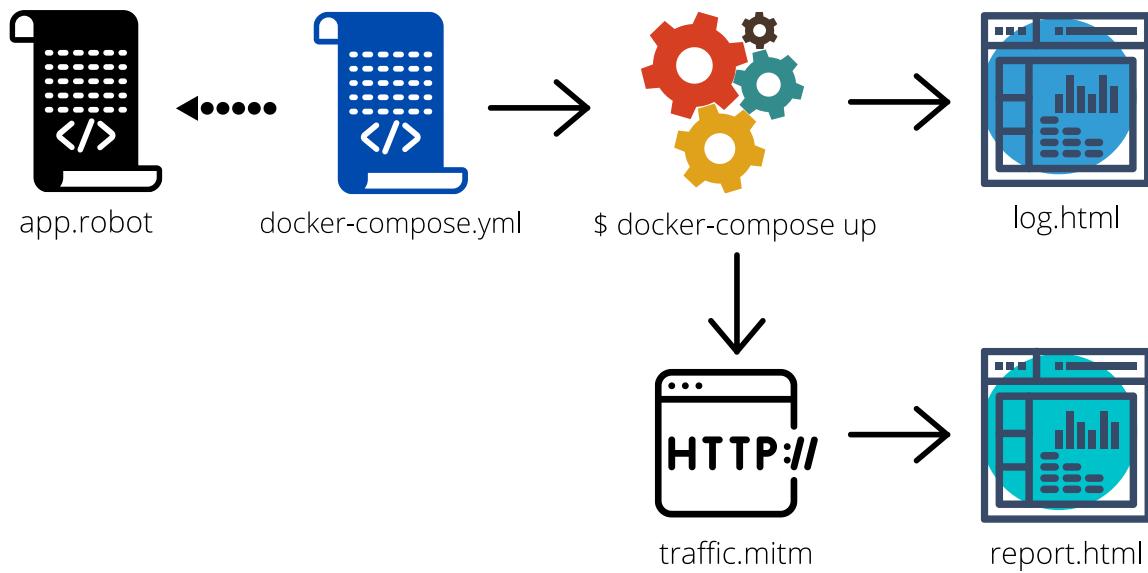
INTERCEPT RAW SERVER REQUESTS AND RESPONSES



1.4 Scenario 3:

INTRODUCING DOCKER

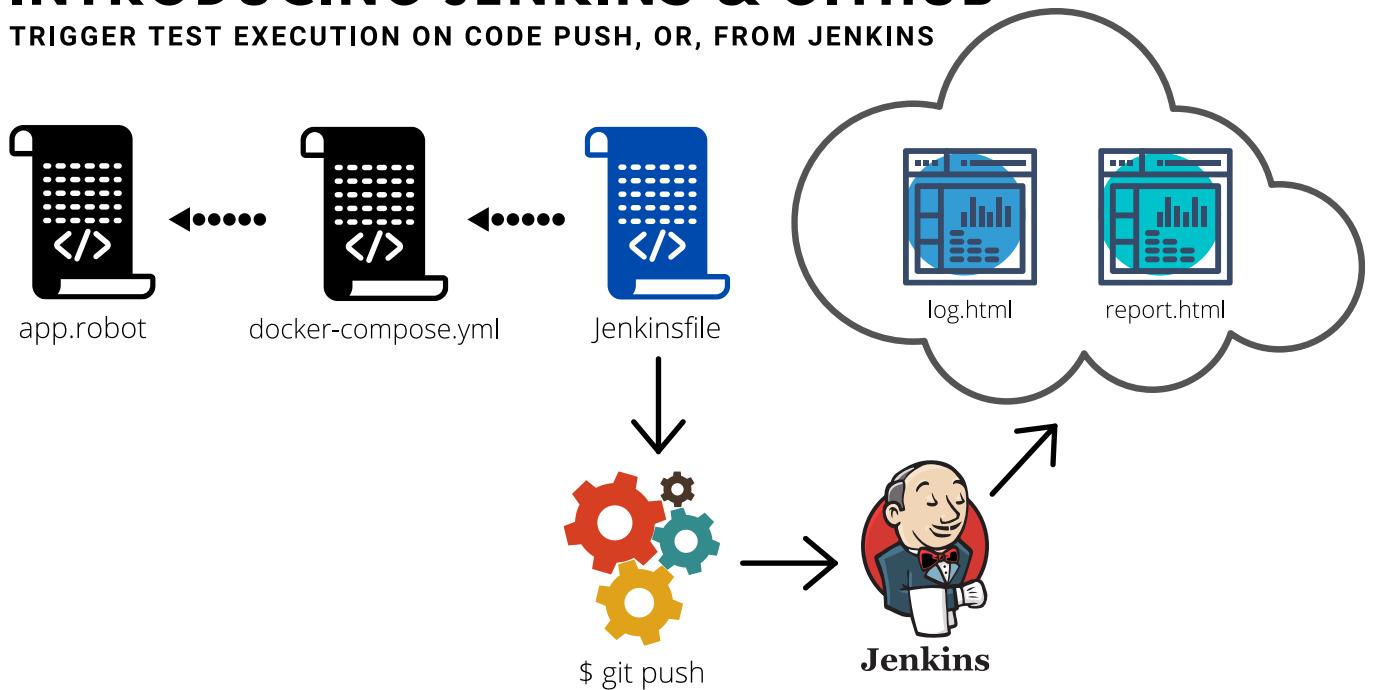
NO NEED FOR LOCAL INSTALLATIONS & SETUP



1.5 Scenario 4:

INTRODUCING JENKINS & GITHUB

TRIGGER TEST EXECUTION ON CODE PUSH, OR, FROM JENKINS



1.6 Table of Contents

1. General Understanding of Security Testing Approach
2. OWASP Top 10
3. Inconveniences/Gaps
4. Introduction to Robot Framework
5. Configuring PyCharm
6. Recommended Folder Structure
7. Basic Elements of Robot Framework
8. Exercise: Structured Reconnaissance in <10 min
9. mitmproxy: Intercepting API Requests & Responses
10. Pabot: Parallel Processing
11. Understanding Docker and Docker Compose
12. Dockerizing Selenium Test Execution Environment
13. CI/CD Build Pipeline
14. Serving test report via S3 and CloudFront
15. Implementing authentication via Lambda Edge
16. Leveraging HTTPPolice
17. Basics of Python programming

- 18. Example: Custom Keywords Library
- 19. When and How to Use Burp Suite
- 20. DVWA: Get Your Hands Dirty

1.7 Trainer



NULLCON
INTERNATIONAL SECURITY TRAINING
SEPTEMBER EDITION
2021

SecQAtion: Tools and Techniques for Security Tests

Riddhi Shree

Online Training:
27th to 30th September 2021

Zoom + Discord

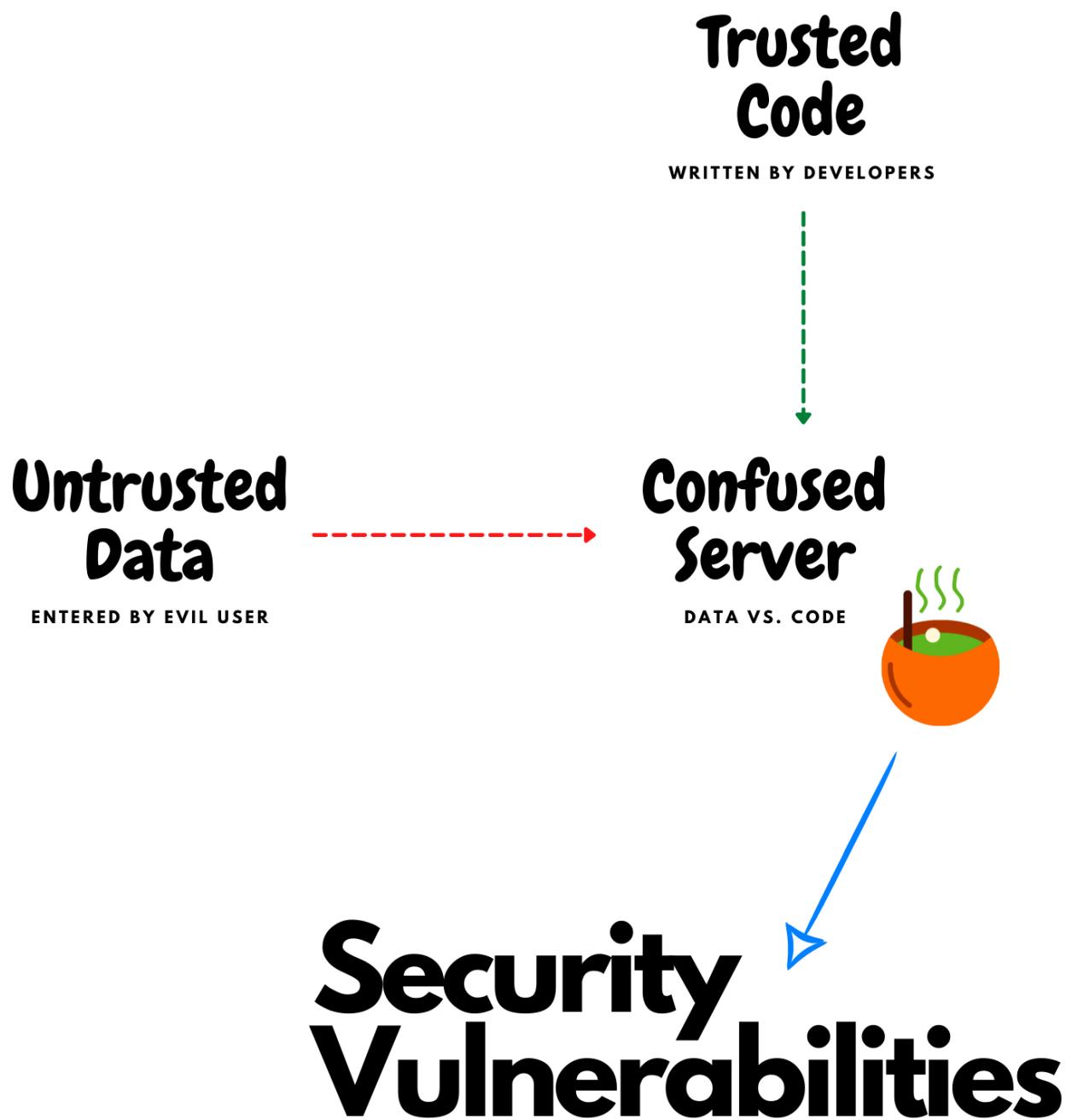
www.nullcon.net | info@nullcon.net

Security Analyst Consultant @Arogya.ai

2. Standard Security Testing Approach

1. Browse
2. Analyze
3. Prepare
4. Attack
5. Confirm
6. Report





2.1 1. Browse

Browse the target application in its entirety, using valid data, so that no feature is left untouched.

But, why?

Let's call this process "reconnaissance".

What information are we collecting?

Everything we possibly can. Most importantly, the raw server requests and responses.

- API endpoints
- Request headers
- Response headers
- Request body
- Response body

2.2 2. Analyze

Identify all entry and exit points.

What are they?

An entry point could be any place that accepts user input. An exit point could be any place in the application that either reflects user input directly, or returns a processed result based upon user-supplied input.

How does it help?

Probably one of these entrypoints is not secure enough. Probably, user can enter arbitrary data that allows confusion at the server. What if due to lack of sufficient user input validation on the server-side, the server confuses user entered data for code that needs to be executed.

The situation could worsen if system manages to dump the unprocessed server responses insecurely in a webpage.

What information are we collecting?

Identify target operating system and version, supported programming language, supported server-side technologies, running software versions, etc.

2.3 3. Prepare

Identify the metacharacter, or, a set of characters that can successfully elicit a "useful" (error) response from the server.

What are metacharacters?

According to Wikipedia,

A **metacharacter** is a character that has a special meaning to a computer program, such as a shell interpreter or a regular expression (regex) engine.

In POSIX extended regular expressions, there are 14 metacharacters that must be *escaped* (preceded by a backslash (\)) in order to drop their special meaning and be treated literally inside an expression: opening and closing square brackets ([] and []); backslash (\); caret (^); dollar sign (\$); period/full stop/dot (.); vertical bar/pipe symbol (|); question mark (?); asterisk (*); plus and minus signs (+ and -); opening and closing curly brackets/braces ({ } and { }); and opening and closing parentheses (() and ()).

For example, to match the arithmetic expression `(1+1)*3=6` with a regex, the correct regex is `\(1\+1\)*3=6`; otherwise, the parentheses, plus sign, and asterisk will have special meanings.

What are we looking for?

We are looking for an error response from the server. A server error would indicate a possibility to control the code running on the server via malicious inputs controlled by a low privileged user.

What shall we prepare?

Prepare a set of payloads to launch targeted attacks on the server.

2.4 4. Attack

With entry points identified, vulnerability confirmed via server error response, and payloads prepared, we are ready to launch an attack on the server.

Take help of cheatsheets to possibly bypass any filters or firewall protection mechanisms.

Research, refine and relaunch the attack until you are satisfied with your findings.

2.5 5. Confirm

If you think your attack was successful, repeat the steps and confirm its reproducibility.

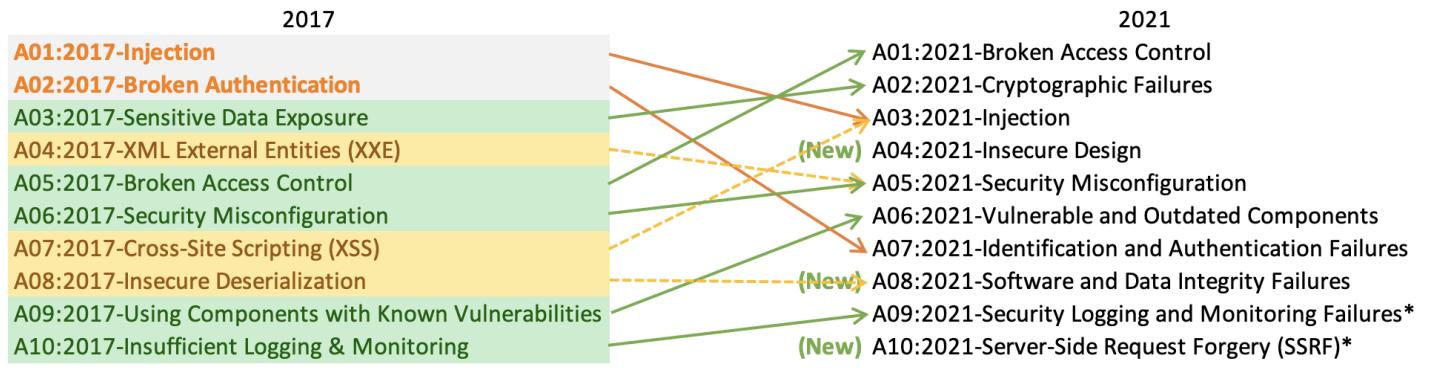
2.6 6. Report

Once confirmed, document your findings and share the security report with the concerned team.

3. Exercise

1. Go to defendtheweb.net
2. Follow the approach explained above (Allotted time: **5 minutes**)
3. Share your findings with us. Explain your approach, any interesting observation, difficulties you faced, issues you found, etc.

4. OWASP Top 10 2021



* From the Survey

1. A01:2021-Broken Access Control
2. A02:2021-Cryptographic Failures
3. A03:2021-Injection
4. A04:2021-Insecure Design
5. A05:2021-Security Misconfiguration
6. A06:2021-Vulnerable and Outdated Components
7. A07:2021-Identification and Authentication Failures
8. A08:2021-Software and Data Integrity Failures
9. A09:2021-Security Logging and Monitoring Failures
10. A10:2021-Server-Side Request Forgery

4.1 Let Me Explain

1. **A01:2021-Broken Access Control:** Parameter tampering, URL tampering, CORS misconfiguration
 1. [Broken Authentication and Authorization Scenarios](#)
 2. [Example: Generate valid JWT token by modifying the vulnerable "email" input parameter](#)
 3. [JWT Toolkit: Attacking JWT Tokens](#)
2. **A02:2021-Cryptographic Failures:** This includes security failures when data is in transit or at rest, such as the implementation of weak cryptographic algorithms, poor or lax key generation, a failure to implement encryption or to verify certificates, and the transmission of data in cleartext.
 1. [Example: Decrypting RSA](#)
 2. [Example: Hashes do NOT ensure safety on their own!](#)

3. A03:2021-Injection:

1. [Various types of Injection attacks](#)
2. [Cross-Site Scripting](#)

4. A04:2021-Insecure Design: Not following secure design patterns and principles

1. Follow best practices and secure design principles, wherever possible
2. Build things with security-centric mindset
3. Conduct code reviews to avoid any bad code going into the production
4. Conduct regular code audits and pentests

5. A05:2021-Security Misconfiguration: The simplest example would be misconfigurations in the cloud service called IAM (Identity and Access Management). If the permissions assigned to an entity is more than it needs to do its job effectively, this could open room for exploitation, if the credentials for that entity got compromised or misused by trusted parties.

1. [Security Misconfigurations](#)
2. [Example: Amazon Cognito misconfiguration](#)
3. [Example: Cognito misconfiguration detection and exploitation technique](#)
4. [Exploiting XML External Entity](#)

6. A06:2021-Vulnerable and Outdated Components: Example: Zero-day in Sign in with Apple

1. Maintain an inventory of components you are using and ensure that they are kept up-to-date
2. Remove unused dependencies and components
3. Install the components via trusted channels and make sure to validate their integrity. Also, it's better to use signed packages (if available)
4. Be on the lookout for any security patches for the components you are relying on. If the packages you use are not maintained, then either make sure you apply patches yourself or use some alternate component that is well maintained.

7. A07:2021-Identification and Authentication Failures: Password bruteforcing, credential stuffing, flawed password reset functionality, mishandled/weak session identifiers

1. [Example: Improper API rate limiting](#)
2. [Example: Security Shepherd — Broken Auth and Session Management Challenge 5](#)

8. A08:2021-Software and Data Integrity Failures: How sure are you about the integrity of the apps or data that you are consuming? Ensure that a software supply chain security tool, such as **OWASP Dependency Check** or **OWASP CycloneDX**, is used to verify that components do not contain known vulnerabilities.

1. [Example: Supply chain attack trojanized SolarWinds Orion business software updates in order to distribute malware](#)

9. A09:2021-Security Logging and Monitoring Failures: When a security-critical event occurs, the software either does not record the event or omits important details about the event when logging it. When security-critical events are not logged properly, such as a failed login attempt, this can make malicious behavior more difficult to detect and may hinder forensic analysis after an attack succeeds. If security critical information is not recorded, there will be no trail for forensic analysis and discovering the cause of problems or the source of attacks may become more difficult or impossible.

10. A10:2021-Server-Side Request Forgery:

1. [Journey of a security bug — From a naive-looking PDF Download to SSRF via HTML Injection in AWS](#)
2. [A10:2021 - Server-Side Request Forgery \(SSRF\)](#)

4.2 Getting Serious?

Pre-requisites: Create a free account on [portswigger.net](#)

Try out these labs:

1. [SQL Injection](#)
2. [XSS](#)
3. [CSRF](#)
4. [Clickjacking](#)
5. [XXE](#)
6. [Many More](#)

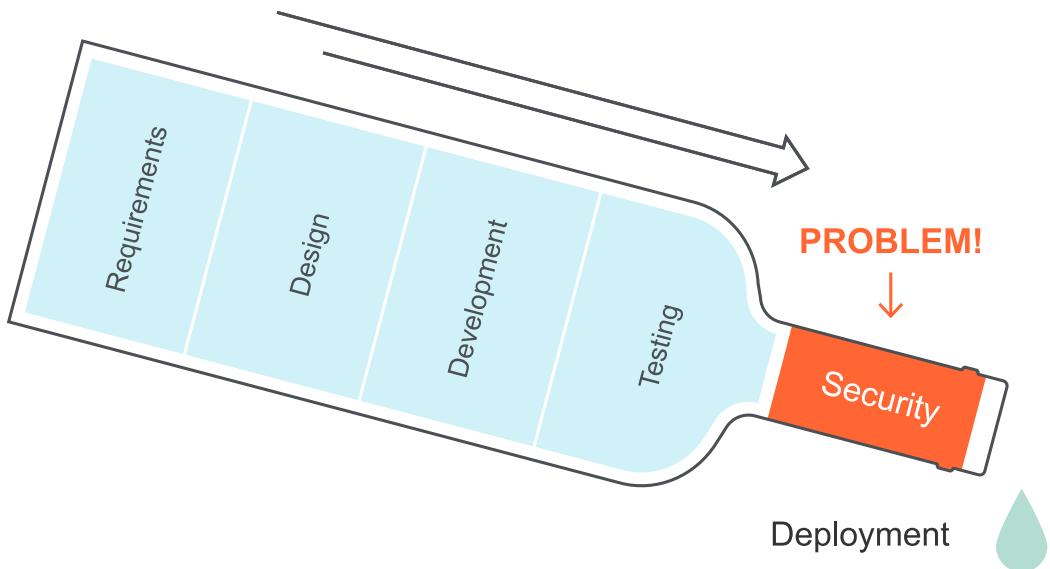
4.3 Further Reading

- <https://owasp.org/Top10/>
- https://owasp.org/Top10/A01_2021-Broken_Access_Control/
- <https://portswigger.net/daily-swig/owasp-shakes-up-web-app-threat-categories-with-release-of-draft-top-10>
- https://owasp.org/Top10/A02_2021-Cryptographic_Failures/
- https://medium.com/@shivam_bathla/a05-2021-security-misconfiguration-fe6d321d71d7
- https://medium.com/@shivam_bathla/a06-2021-vulnerable-and-outdated-components-a5d96017049c
- https://medium.com/@shivam_bathla/a07-2021-identification-and-authentication-failures-b585c856eb24
- <https://www.sans.org/brochure/course/log-management-in-depth/6>
- <https://owasp-top-10-proactive-controls-2018.readthedocs.io/en/latest/c9-implement-security-logging-monitoring.html>
- <https://cwe.mitre.org/data/definitions/778.html>
- <https://docs.microsoft.com/en-us/azure/architecture/best-practices/monitoring>

5. Inconveniences/Gaps

5.1 Developer's Perspective

In traditional software development, security is a bottleneck



5.2 DevSecOps: Is there a scope for improvement?

How does penetration testing work in DevSecOps?

DevSecOps means software gets released with a basic level of security built in. But detection of certain vulnerabilities can still require penetration testing. This more manual step will generally happen shortly before or after development - and is crucial for effective DevSecOps.

While penetration testing can reveal advanced vulnerabilities, it's not a quick process. A worldwide skills shortage also makes it difficult to carry out at scale. Conversely, vulnerability scanning is fast and gives broad coverage, but can lack depth compared to manual testing. Each has benefits and drawbacks - and DevSecOps security best practice demands both.

This approach is of great benefit to organizations with many applications to secure. While blanket penetration testing at this scale may be impossible, DevSecOps allows for an acceptable level of security to be achieved before release. Manual testing can then be carried out on a priority-based approach.

- Do we really have to choose between manual testing and vulnerability scanning?
- Manual pentesting is slow and tedious, but highly effective
- Vulnerability scans are fast and effortless, but less reliable
- Is there a way to achieve much more than a plain vulnerability scan? Is there a way to make manual testing less tedious?

5.3 Test Automation: Is it too tedious for pentesters?

Not really.

5.4 References:

- <https://portswigger.net/solutions/devsecops/guide-to-devsecops>

6. Introduction to Robot Framework

6.1 Overview

Robot Framework is a generic open source automation framework. It can be used for **test automation** and **robotic process automation** (RPA). **SeleniumLibrary** is one of the many test libraries that can be used with Robot Framework.

6.2 Advantage

- Easy syntax
- Human-readable keywords
- Operating system and application independent
- Supports both Python 2.7 and Python 3.5+
- Has a rich ecosystem consisting of pre-existing libraries and tools
- Capabilities can be extended by libraries implemented with Python or Java

6.3 Installation

```
pip install robotframework pip install --upgrade robotframework-seleniumlibrary sudo pip install webdrivermanager sudo webdrivermanager firefox chrome --allow-insecure-downloads
```

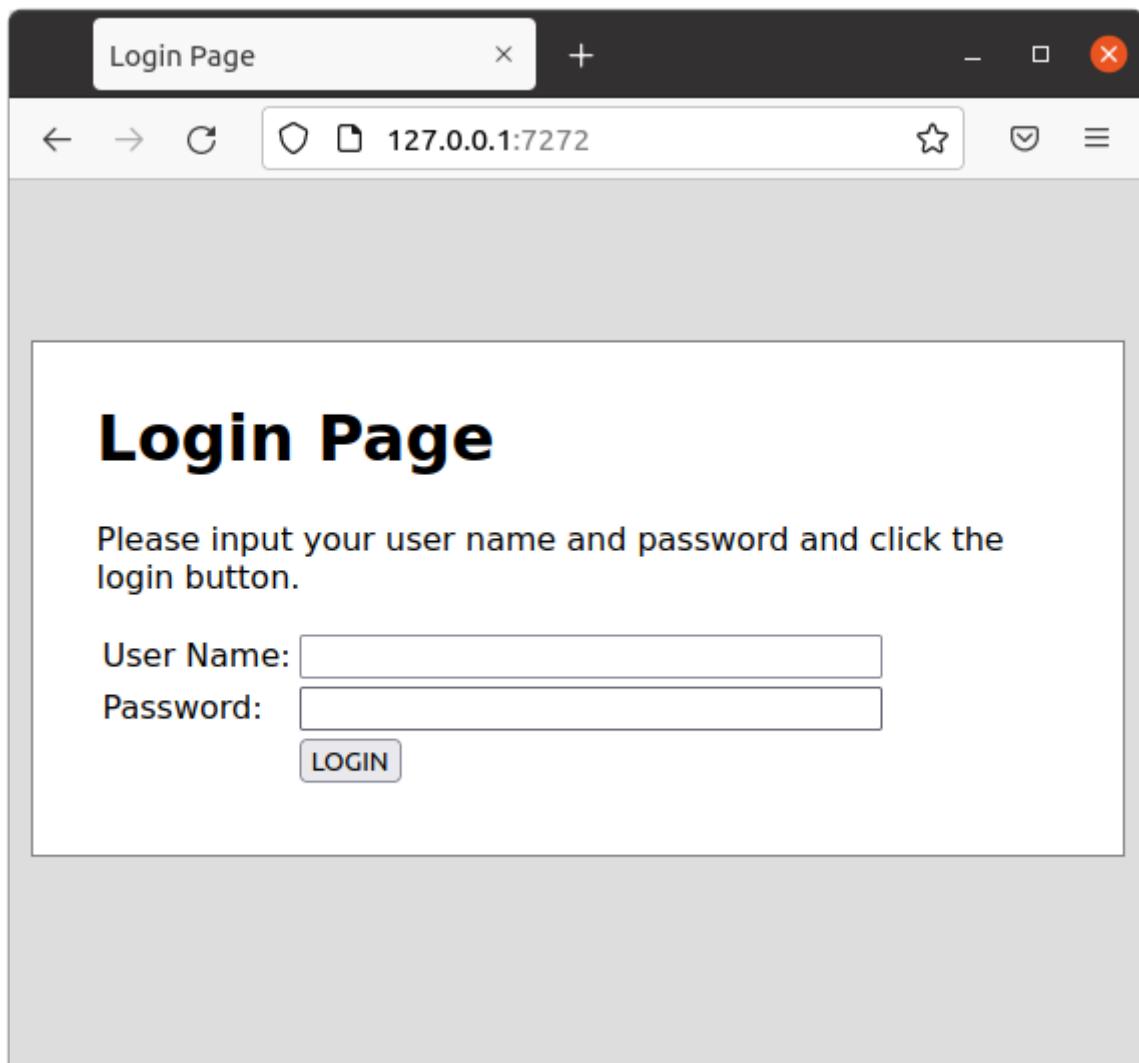
More details [here](#) and [here](#).

6.4 Demo: Standard Usage

1. Clone following GitHub repository `git clone https://github.com/robotframework/WebDemo.git`
2. Start the demo application `cd WebDemo/ python demoapp/server.py`

```
secqation@mirage:~/Desktop/WebDemo$ ls -1
BUILD.rst
demoapp
demoapp.png
docs
login_tests
README.rst
requirements.txt
tasks.py
secqation@mirage:~/Desktop/WebDemo$ python demoapp/server.py
Demo server starting on port 7272.
127.0.0.1 - - [08/Aug/2021 21:51:19] "GET / HTTP/1.1" 200 -
127.0.0.1 - - [08/Aug/2021 21:51:19] "GET /demo.css HTTP/1.1" 200 -
127.0.0.1 - - [08/Aug/2021 21:51:19] code 404, message File not found
127.0.0.1 - - [08/Aug/2021 21:51:19] "GET /favicon.ico HTTP/1.1" 404 -
```

3. Access the demo application by navigating to `127.0.0.1:7272`



4. Trigger automated test execution

```
robot login_tests/
Login Tests.Invalid Login :: A test suite containing tests related to inval...
=====
| Invalid Username | PASS |
| Invalid Password | PASS |
| Invalid Username And Password | PASS |
| Empty Username | PASS |
| Empty Password | PASS |
| Empty Username And Password | PASS |
=====
| Login Tests.Invalid Login :: A test suite containing tests related... | PASS |
| 6 tests, 6 passed, 0 failed |
=====
Login Tests.Valid Login :: A test suite with a single test for valid login.
=====
| Valid Login | PASS |
=====
| Login Tests.Valid Login :: A test suite with a single test for val... | PASS |
| 1 test, 1 passed, 0 failed |
=====
| Login Tests | PASS |
| 8 tests, 8 passed, 0 failed |
=====
Output: /home/secqation/Desktop/WebDemo/output.xml
Log:   /home/secqation/Desktop/WebDemo/log.html
Report: /home/secqation/Desktop/WebDemo/report.html
```

5. Access test execution report

firefox log.html

Statistics by Suite	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
Login Tests	8	8	0	0	00:00:25	<div style="width: 100%; background-color: #6aa84f;"></div>
Login Tests.Gherkin Login	1	1	0	0	00:00:08	<div style="width: 100%; background-color: #6aa84f;"></div>
Login Tests.Invalid Login	6	6	0	0	00:00:10	<div style="width: 100%; background-color: #6aa84f;"></div>
Login Tests.Valid Login	1	1	0	0	00:00:07	<div style="width: 100%; background-color: #6aa84f;"></div>

Test Execution Log

- **SUITE** Login Tests

Full Name: Login Tests
 Source: /home/secqation/Desktop/WebDemo/login_tests
 Start / End / Elapsed: 20210808 22:07:28.920 / 20210808 22:07:53.464 / 00:00:24.544
 Status: 8 tests total, 8 passed, 0 failed, 0 skipped

- **SUITE** Gherkin Login

Full Name: Login Tests.Gherkin Login
 Documentation: A test suite with a single Gherkin style test.
 This test is functionally identical to the example in valid_login.robot file.
 Source: /home/secqation/Desktop/WebDemo/login_tests/gherkin_login.robot
 Start / End / Elapsed: 20210808 22:07:28.952 / 20210808 22:07:36.574 / 00:00:07.622
 Status: 1 test total, 1 passed, 0 failed, 0 skipped

- **TEST** Valid Login

Full Name: Login Tests.Gherkin Login.Valid Login
 Start / End / Elapsed: 20210808 22:07:29.216 / 20210808 22:07:36.572 / 00:00:07.356
 Status: **PASS**

- + **KEYWORD** Given browser is opened to login page
- + **KEYWORD** When user "demo" logs in with password "mode"
- + **KEYWORD** resource. Then welcome page should be open
- + **TEARDOWN** SeleniumLibrary.Close Browser

+ **SUITE** Invalid Login

+ **SUITE** Valid Login

6.5 References

- Installation instructions
- SeleniumLibrary
- Web testing with Robot Framework and SeleniumLibrary
- Running Demo

7. Configuring PyCharm

1. Download [PyCharm](#)
2. Install and run PyCharm

Installation Instructions

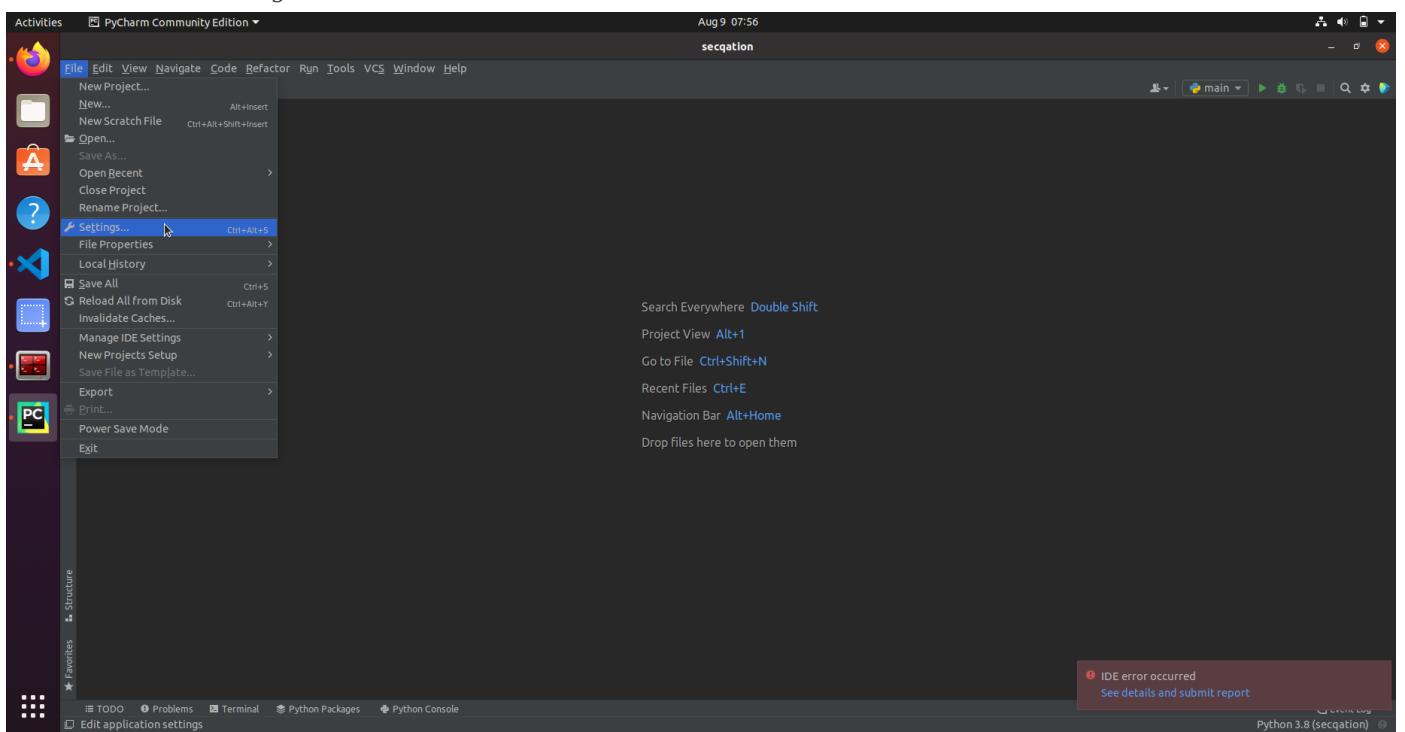


1. Copy the pycharm-2021.2.tar.gz to the desired installation location
(make sure you have rw permissions for that directory)
2. Unpack the pycharm-2021.2.tar.gz file to an empty directory using the following command: tar -xzf pycharm- 2021.2.tar.gz

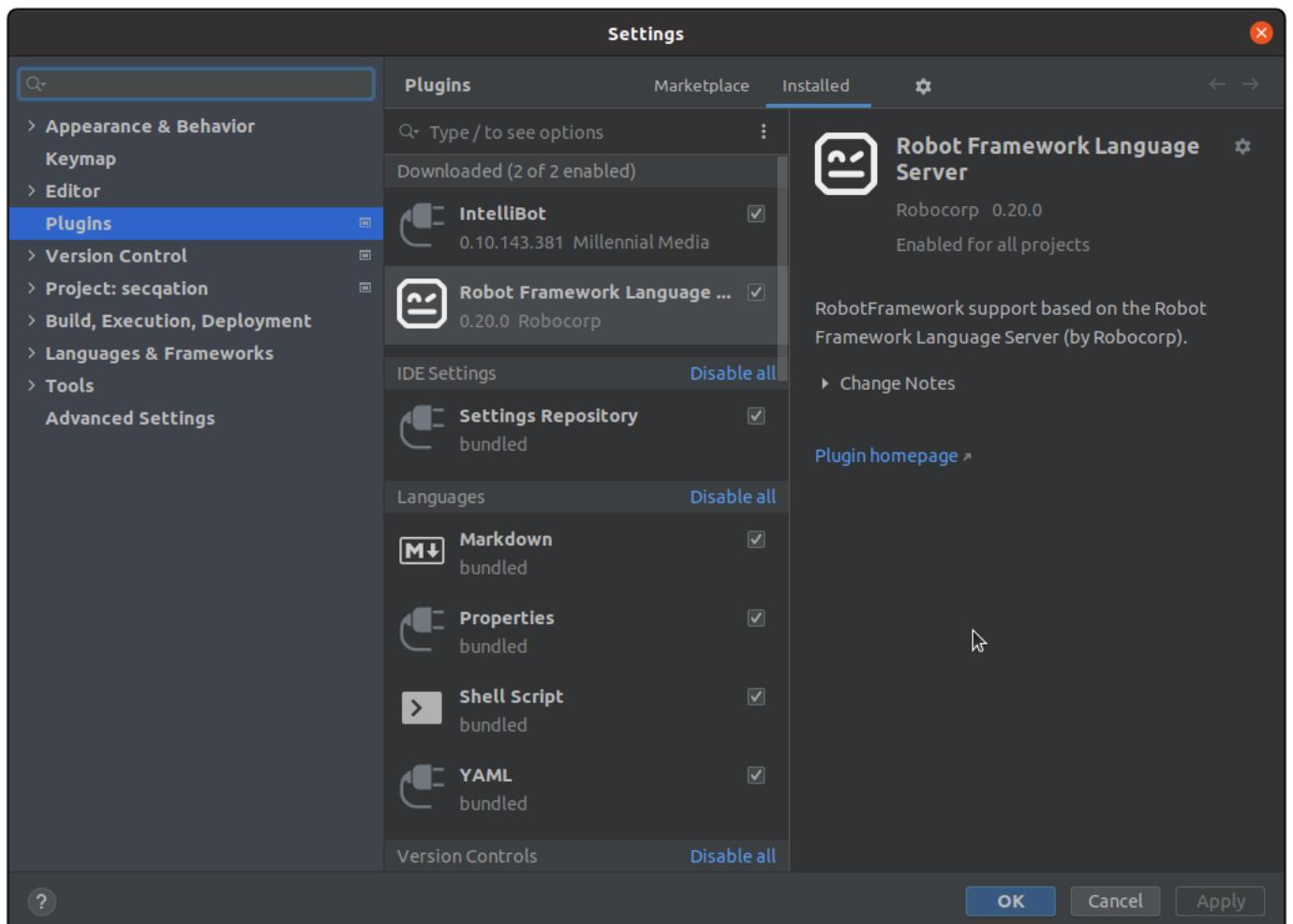
Note: A new instance **MUST NOT** be extracted over an existing one. The target folder must be empty

3. Run pycharm.sh from the bin subdirectory

3. Go to "File" > "Settings"

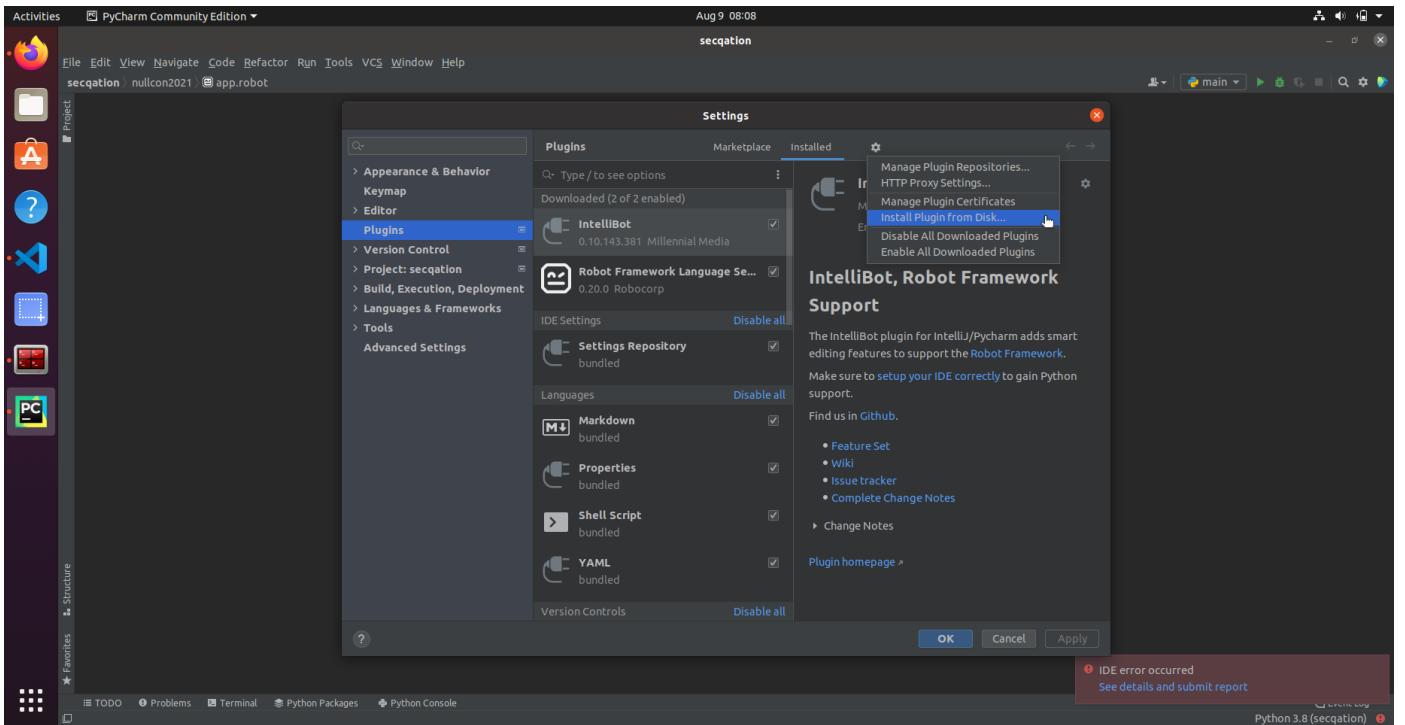


4. Install the "Robot Framework Language Server" plugin



5. Download [intellibot.jar](#)

6. Install `intellibot.jar` plugin to your IDE by using the 'Install plugin from disk...' option



7. Re-start PyCharm

7.1 References

- Download and install PyCharm
- IntelliJ/PyCharm Plugin for Robot Automation Framework
- `intellibot.jar`

8. Recommended Folder Structure

8.1 Short on time? Create single robot file.

```
1      *** Settings ***
2          Library      SeleniumLibrary
3
4      *** Variables ***
5      ${BASE_URL} =  https://www.winja.site
6
7      *** Keywords ***
8      Custom keyword
9          Log      This is a custom keyword
10
11     *** Test Cases ***
12     Test Case 1
13         Set Selenium Speed    0
14         Open Browser  ${BASE_URL}
15         Custom keyword
16         Close All Browsers
17     |
```

8.2 For something more serious, get organized?



1. Be in control of your assets
2. Know where to find what
3. Avoid unmanageable pile of files, as your project grows bigger
4. Scale easily
5. Easy maintenance
6. etc.

In short, don't waste your time cleaning the garbage off your desk, when you could very well use that time for doing something more productive. Get organized now, to save time and effort later.

8.3 What kind of structure are we talking about?



8.4 Practice Exercise

1. Open PyCharm IDE
2. Create a new project
3. Create a folder structure as described below:

- Create 6 folders: **Tests**, **Resources**, **Data**, **Results**, **CustomLibraries**, **BashScripts**
- Create `App.robot` file inside **Tests** folder
- Inside **Resources** folder
 - Create **PageObjects** sub-folder
 - Create `Common.robot` and `App.robot` files

9. Basic Elements of Robot Framework

1. Keyword
2. Library
3. Template
4. Test Case
5. Report and Log

9.1 Keyword

Robot Framework has easy syntax, utilizing human-readable keywords.

***** Keywords *****

Open Browser To Login Page

```
Open Browser    ${LOGIN URL}    ${BROWSER}
Maximize Browser Window
Set Selenium Speed    ${DELAY}
Login Page Should Be Open
```

Login Page Should Be Open

```
Title Should Be    Login Page
```

Go To Login Page

```
Go To    ${LOGIN URL}
Login Page Should Be Open
```

Input Username

```
[Arguments]    ${username}
Input Text    username_field    ${username}
```

Input Password

```
[Arguments]    ${password}
Input Text    password_field    ${password}
```

Submit Credentials

```
Click Button    login_button
```

Welcome Page Should Be Open

```
Location Should Be    ${WELCOME URL}
Title Should Be    Welcome Page
```

9.2 Library

Custom keywords can be created by defining methods in a Python file. Capabilities can be extended by libraries implemented with Python or Java.

The screenshot shows the PyCharm IDE interface. On the left is the Project tool window displaying a file tree for a project named 'secqat'. The 'CustomLibraries' folder contains a file named 'reconnaissance.py'. The right side shows the code editor for 'reconnaissance.py'.

```

1 import requests
2 from bs4 import BeautifulSoup
3 from tldextract import extract
4
5
6 def grab_hyperlinks(url):
7     reqs = requests.get(url)
8     soup = BeautifulSoup(reqs.text, 'html.parser')
9
10    urls = []
11    for link in soup.find_all('a'):
12        print(link.get('href'))
13
14
15    def grab_forms(url):
16        reqs = requests.get(url)
17        soup = BeautifulSoup(reqs.text, 'html.parser')
18
19        urls = []
20        for form in soup.find_all('form'):
21            print(form.get('id'))
22            print('-----')
23            print(form)

```

9.3 Template

For scenarios where test data varies, but test steps remain the same, use templates.

```

*** Settings ***
Test Template      Calculate
Library           CalculatorLibrary

*** Test Cases ***
Additions          Expression    Expected
                        12 + 2 + 2    16
                        2 + -3        -1

Subtractions        Expression   Expected
                        12 - 2 - 2    8
                        2 - -3        5

Multiplication     Expression  Expected
                        12 * 2 * 2    48
                        2 * -3        -6

Division            Expression  Expected
                        12 / 2 / 2    3
                        2 / -3        -1

Calculation error   [Template]  Calculation should fail
                        kekkonen    Invalid button 'k'.
                        ${EMPTY}    Invalid expression.
                        1 / 0       Division by zero.

*** Keywords ***
Calculate
    [Arguments]    ${expression}    ${expected}
    Push buttons   C${expression}=
    Result should be ${expected}

Calculation should fail
    [Arguments]    ${expression}    ${expected}
    ${error} =      Should fail   C${expression}=
    Should be equal ${expected}    ${error}

```

9.4 Test Case

It's an abstraction layer, under which a series of automated test steps and output validations take place.

***** Settings *****

Documentation A test suite with a single test
...
...
...
Resource This test has a workflow that is
the imported resource file.
resource.txt

***** Test Cases *****Valid Login

Open Browser To Login Page
Input Username demo
Input Password mode
Submit Credentials
Welcome Page Should Be Open
[Teardown] Close Browser

9.5 Report and Log

One-stop place to view the test outputs and to analyze the overall application health.

← → ⌂ file:///home/secqation/PycharmProjects/secqation/nullcon2021/Results/log.html#s1-t1

App Log

Generated
20210811 00:55:16 UTC+05:30
2 days 10 hours ago

Test Statistics

Total Statistics		Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
All Tests		1	1	0	0	00:01:08	<div style="width: 100%; background-color: #6aa84f;"></div>

Statistics by Tag		Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
No Tags							<div style="width: 0%; background-color: #e0e0e0;"></div>

Statistics by Suite		Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
App		1	1	0	0	00:01:22	<div style="width: 100%; background-color: #6aa84f;"></div>

Test Execution Log

- **SUITE** App
 - Full Name: App
 - Source: /home/secqation/PycharmProjects/secqation/nullcon2021/Tests/App.robot
 - Start / End / Elapsed: 20210811 00:53:54.525 / 20210811 00:55:16.498 / 00:01:21.973
 - Status: 1 test total, 1 passed, 0 failed, 0 skipped
- + **SETUP** Common.Begin Web Assessment
- + **TEARDOWN** Common.End Web Assessment
- **TEST** Information Gathering
 - Full Name: App.Information Gathering
 - Start / End / Elapsed: 20210811 00:54:06.935 / 20210811 00:55:14.971 / 00:01:08.036
 - Status: **PASS**
 - **KEYWORD** App.Crawl <https://defendtheweb.net/>
 - Start / End / Elapsed: 20210811 00:54:06.938 / 20210811 00:54:56.433 / 00:00:49.495
 - + **KEYWORD** SeleniumLibrary.Go To \${Target_URL}
 - + **KEYWORD** Reconnaissance.Read Source Code \${Target_URL}
 - + **KEYWORD** reconnaissance.Grab Hyperlinks \${Target_URL}
 - + **KEYWORD** reconnaissance.Grab Forms \${Target_URL}
 - + **KEYWORD** \${domain} = reconnaissance.Extract Domain \${Target_URL}
 - + **KEYWORD** Reconnaissance.Google search \${domain}
 - + **KEYWORD** Reconnaissance.Check Wayback Machine \${domain}
 - + **KEYWORD** App.Crawl <http://demo.testfire.net/>

10. Exercise: Structured Reconnaissance in <10 min

Target Application: <https://defendtheweb.net/>

10.1 Task #1

Allotted Time: 10 minutes

1. Open a browser.
2. Navigate to <https://defendtheweb.net>.
3. Explore the target application and share what you've discovered.

Questions:

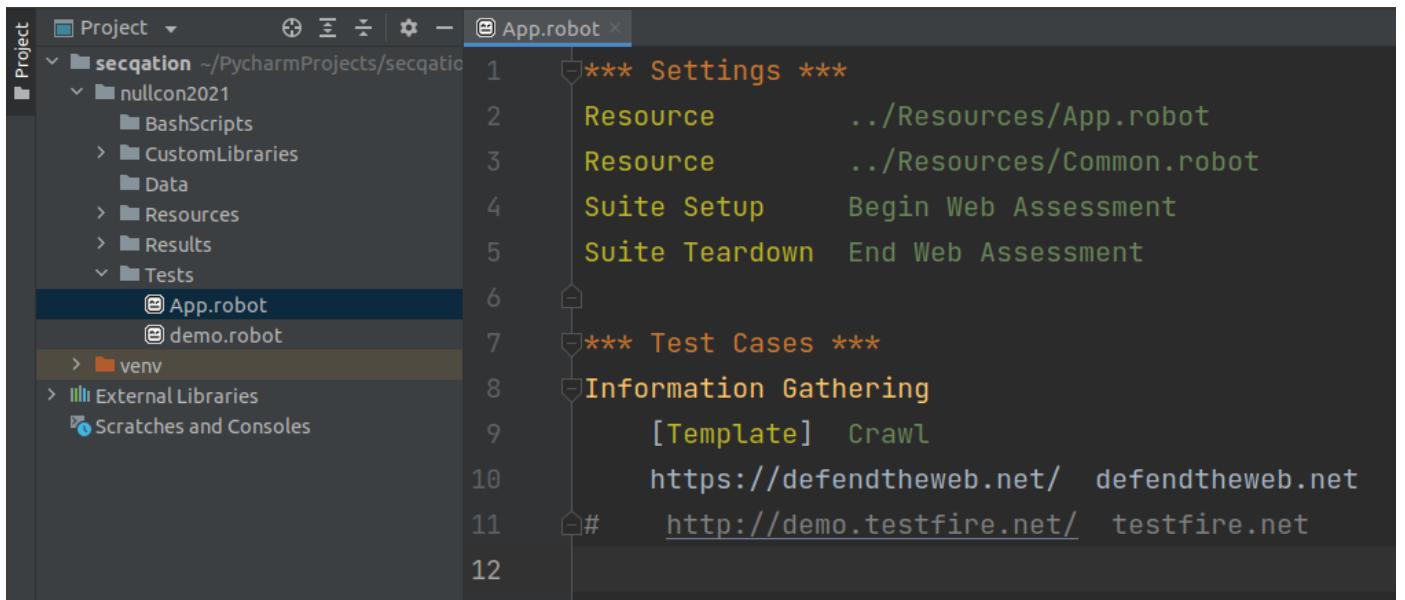
- How much reconnaissance were you able to perform manually?
- Is it enough, or do you feel you need to spend more time to explore the target application?

10.2 Task #2

Allotted Time: 10 minutes

Resource Folder Location: /home/secqation/Desktop/NullconTraining2021/2-example/

1. Create a new project in PyCharm
2. Copy `App.robot` file from `2-example/Tests/App.robot` into **Tests** folder

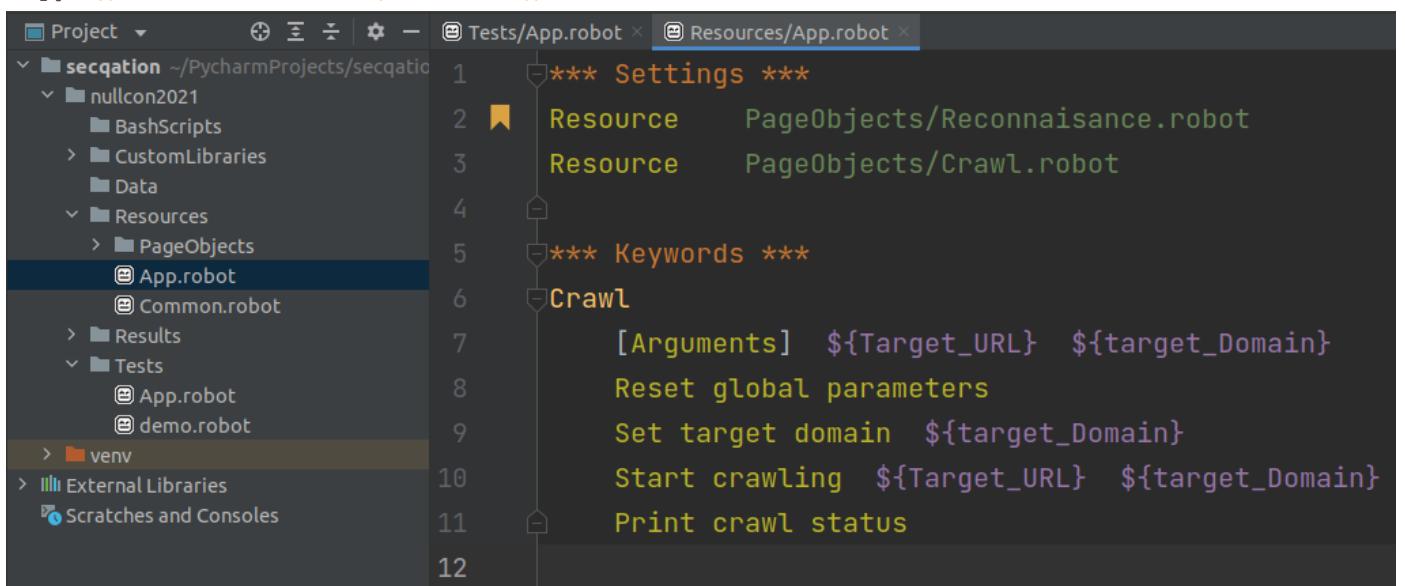


```

1  *** Settings ***
2  Resource      ./Resources/App.robot
3  Resource      ./Resources/Common.robot
4  Suite Setup   Begin Web Assessment
5  Suite Teardown End Web Assessment
6
7  *** Test Cases ***
8  Information Gathering
9    [Template] Crawl
10   https://defendtheweb.net/ defendtheweb.net
11   # http://demo.testfire.net/ testfire.net
12

```

3. Copy `App.robot` file from `2-example/Resources/App.robot` into **Resources** folder



```

1  *** Settings ***
2  Resource      PageObjects/Reconnaissance.robot
3  Resource      PageObjects/Crawl.robot
4
5  *** Keywords ***
6  Crawl
7    [Arguments]  ${Target_URL}  ${target_Domain}
8    Reset global parameters
9    Set target domain  ${target_Domain}
10   Start crawling  ${Target_URL}  ${target_Domain}
11   Print crawl status
12

```

4. Copy `Common.robot` file from `2-example/Resources/Common.robot` into **Resources** folder

The screenshot shows the PyCharm interface with the project tree on the left and the code editor on the right. The project tree shows a directory structure under 'secqation' with 'Resources' containing 'PageObjects' and 'Common.robot'. The code editor displays the content of 'Common.robot'.

```

1  *** Settings ***
2  Library      SeleniumLibrary
3
4  *** Variables ***
5  ${FIREFOX_PROXY_PROFILE} = #BLANK
6
7  *** Keywords ***
8  Begin Web Assessment
9    Open Browser  about:blank  ff_profile_dir=${FIREFOX_PROXY_PROFILE}
10   Maximize Browser Window
11   Set Selenium Speed  0
12   Set Selenium Implicit Wait  8 seconds
13   Set Selenium Timeout  10 seconds
14   Delete All Cookies
15
16  End Web Assessment
17  Capture Page Screenshot
18  Close All Browsers
19

```

5. Copy `Crawl.robot` file from `2-example/Resources/PageObjects/Crawl.robot` into **Resources/PageObjects** folder

The screenshot shows the PyCharm interface with the project tree on the left and the code editor on the right. The project tree shows a directory structure under 'secqation' with 'Resources' containing 'PageObjects' and 'Crawl.robot'. The code editor displays the content of 'Crawl.robot'.

```

1  *** Settings ***
2  Library      SeleniumLibrary
3  Library      ../../CustomLibraries/reconnaissance.py
4  Library      Collections
5  Library      XML
6
7  *** Variables ***
8  @{ALL_DISCOVERED_LINKS} = #BLANK
9  @{ALL_IN_SCOPE_LINKS} = #BLANK
10  @{OUT_OF_SCOPE_LINKS} = #BLANK
11  @{VISITED_LINKS} = #BLANK
12  @{DISCOVERED_BUT_NOT_VISITED_LINKS} = #BLANK
13
14  @{ALL_DISCOVERED_FORMS} = #BLANK
15  @{ALL_FORMS_ATTRIBUTES_AND_INPUT_ELEMENTS} = #BLANK
16
17  ${IN_SCOPE_DOMAIN} = ""
18  ${CURRENT_CRAWL_DEPTH} = 0
19  ${MAX_CRAWL_DEPTH} = 2
20
21  *** Keywords ***
22  Set target domain
23    [Arguments]  ${target_Domain}
24    ${IN_SCOPE_DOMAIN} = Set Variable  ${target_Domain}

```

6. Copy `reconnaissance.py` file from `2-example/CustomLibraries/reconnaissance.py` into **CustomLibraries** folder

The screenshot shows the PyCharm IDE interface. The left sidebar displays the project structure under the 'Project' tab. The 'CustomLibraries' folder contains a single file named 'reconnaissance.py'. The right pane shows the code content of this file.

```

1 import requests
2 from bs4 import BeautifulSoup
3 from tldextract import extract
4 import re
5 import base64
6
7 try:
8     from urlparse import urljoin, urlparse # Python2
9 except ImportError:
10    from urllib.parse import urljoin, urlparse # Python3
11
12 def grab_hyperlinks(url):
13     reqs = requests.get(url)
14     soup = BeautifulSoup(reqs.text, 'html.parser')
15     set_of_links = set()
16
17     for link in soup.find_all('a'):
18         href = link.get('href')
19         if not bool(urlparse(href).netloc):
20             href = urljoin(url, href)
21         set_of_links.add(href)
22
23     unique_links = list(set_of_links)
24
25     return unique_links

```

7. Trigger test execution:

```
robot -d Results/ Tests/App.robot
```

```
secqation@mirage:~/PycharmProjects/secqation/nullcon2021$ ll
total 40
drwxr-xr-x 8 secqation seqation 4096 Aug  9 19:39 .
drwxr-xr-x 5 secqation seqation 4096 Aug  4 21:03 ..
drwxr-xr-x 2 secqation seqation 4096 Aug  9 19:39 BashScripts/
drwxr-xr-x 3 secqation seqation 4096 Aug 16 18:03 CustomLibraries/
drwxr-xr-x 2 secqation seqation 4096 Aug  9 19:38 Data/
drwxr-xr-x 3 secqation seqation 4096 Aug 16 17:43 Resources/
drwxr-xr-x 2 secqation seqation 12288 Aug 15 23:20 Results/
secqation@mirage:~/PycharmProjects/secqation/nullcon2021$ robot -d Results/ Tests/App.robot
```

8. Take a break until test execution is done

```
secqation@mirage:~/PycharmProjects/secqation/nullcon2021$ robot -d Results/ Tests/App.robot
=====
App
=====
Information Gathering | PASS |
-----
App | PASS |
1 test, 1 passed, 0 failed
=====
Output: /home/secqation/PycharmProjects/secqation/nullcon2021/Results/output.xml
Log:   /home/secqation/PycharmProjects/secqation/nullcon2021/Results/log.html
Report: /home/secqation/PycharmProjects/secqation/nullcon2021/Results/report.html
secqation@mirage:~/PycharmProjects/secqation/nullcon2021$ █
```

9. Open `log.html` in your browser window, to view the test results 10. Expand "**Print crawl status**" keyword

<file:///home/secqation/PycharmProjects/secqation/nullcon2021/Results/log.html>

Test Statistics

Total Statistics	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
All Tests	1	1	0	0	00:01:57	<div style="width: 100%; background-color: green;"></div>

Statistics by Tag	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
No Tags						<div style="width: 0%; background-color: grey;"></div>

Statistics by Suite	Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
App	1	1	0	0	00:02:09	<div style="width: 100%; background-color: green;"></div>

Test Execution Log

- SUITE App	00:02:08.880
Full Name: App	
Source: /home/secqation/PycharmProjects/secqation/nullcon2021/Tests/App.robot	
Start / End / Elapsed: 20210816 18:10:26.080 / 20210816 18:12:34.960 / 00:02:08.880	
Status: 1 test total, 1 passed, 0 failed, 0 skipped	
+ SETUP Common.Begin Web Assessment	00:00:09.607
+ TEARDOWN Common.End Web Assessment	00:00:00.822
- TEST Information Gathering	00:01:57.411
Full Name: App.Information Gathering	
Start / End / Elapsed: 20210816 18:10:36.726 / 20210816 18:12:34.137 / 00:01:57.411	
Status: PASS	
- KEYWORD App.Crawl https://defendtheweb.net/, defendtheweb.net	00:01:57.409
Start / End / Elapsed: 20210816 18:10:36.728 / 20210816 18:12:34.137 / 00:01:57.409	
+ KEYWORD Crawl.Reset global parameters	00:00:00.008
+ KEYWORD Crawl.Set target domain \${target_Domain}	00:00:00.002
+ KEYWORD Crawl.Start crawling \${Target_URL}, \${target_Domain}	00:01:56.852
- KEYWORD Crawl.Print crawl status	00:00:00.546
Start / End / Elapsed: 20210816 18:12:33.591 / 20210816 18:12:34.137 / 00:00:00.546	
+ KEYWORD Crawl.Collect final statistics	00:00:00.536
+ KEYWORD Collections.Log List \${VISITED_LINKS}	00:00:00.005
+ KEYWORD Collections.Log List \${DISCOVERED_BUT_NOT_VISITED_LINKS}	00:00:00.001
+ KEYWORD Collections.Log List \${OUT_OF_SCOPE_LINKS}	00:00:00.000
+ KEYWORD Collections.Log List \${ALL_IN_SCOPE_LINKS}	00:00:00.001
+ KEYWORD Collections.Log List \${ALL_DISCOVERED_FORMS}	00:00:00.002
+ KEYWORD Collections.Log List \${ALL_FORMS_ATTRIBUTES_AND_INPUT_ELEMENTS}	00:00:00.001

11. Expand desired keywords to view captured details

<input type="checkbox"/>	KEYWORD	Crawl.Print crawl status	00:
Start / End / Elapsed:	20210816 18:12:33.591 / 20210816 18:12:34.137 / 00:00:00.546		
<input checked="" type="checkbox"/>	KEYWORD	Crawl.Collect final statistics	00:
<input type="checkbox"/>	KEYWORD	Collections.Log List \${VISITED_LINKS}	00:
	Documentation:	Logs the length and contents of the <code>list</code> using given <code>level</code> .	
Start / End / Elapsed:	20210816 18:12:34.127 / 20210816 18:12:34.132 / 00:00:00.005		
18:12:34.131	INFO	List length is 26 and it contains following items: 0: https://defendtheweb.net/ 1: https://defendtheweb.net/article/4-lesser-known-attack-types 2: https://defendtheweb.net/article/beyond-root-custom-firmware-for-embedded-mobile-chips 3: https://defendtheweb.net/article/buffer-overflow-to-run-root-shell-full-tutorial 4: https://defendtheweb.net/article/build-a-virtual-private-network-with-wireguard-and-fedora 5: https://defendtheweb.net/article/have-you-been-pwned 6: https://defendtheweb.net/articles 7: https://defendtheweb.net/articles/coding 8: https://defendtheweb.net/articles/hacking 9: https://defendtheweb.net/articles/privacy 10: https://defendtheweb.net/articles/talks-videos 11: https://defendtheweb.net/auth 12: https://defendtheweb.net/auth/signup 13: https://defendtheweb.net/dashboard 14: https://defendtheweb.net/discussions 15: https://defendtheweb.net/donations 16: https://defendtheweb.net/help 17: https://defendtheweb.net/help/conduct/code-of-conduct 18: https://defendtheweb.net/help/contact 19: https://defendtheweb.net/legal/privacy 20: https://defendtheweb.net/legal/terms 21: https://defendtheweb.net/playground 22: https://defendtheweb.net/profile/hackingguy 23: https://defendtheweb.net/profile/keeper 24: https://defendtheweb.net/shop 25: https://defendtheweb.net/statistics	
<input checked="" type="checkbox"/>	KEYWORD	Collections.Log List \${DISCOVERED_BUT_NOT_VISITED_LINKS}	00:

-	KEYWORD	Collections.Log List \${DISCOVERED_BUT_NOT_VISITED_LINKS}	00:00:00
Documentation:	Logs the length and contents of the list using given level.		
Start / End / Elapsed:	20210816 18:12:34.132 / 20210816 18:12:34.133 / 00:00:00.001		
18:12:34.132	INFO	List length is 311 and it contains following items: 0: http://defendtheweb.net 1: https://defendtheweb.net/article/about-sql-injections-with-ms-sql-server 2: https://defendtheweb.net/article/all-your-devices-can-be-hacked 3: https://defendtheweb.net/article/blind-sql-injection 4: https://defendtheweb.net/article/buffer-overflows-and-ids-basics 5: https://defendtheweb.net/article/bypassing-msi-installer-checks 6: https://defendtheweb.net/article/c-basics 7: https://defendtheweb.net/article/common-php-attacks-directory-traversal 8: https://defendtheweb.net/article/common-php-attacks-poison-null-byte 9: https://defendtheweb.net/article/confessions-of-a-cyber-spy-hunter 10: https://defendtheweb.net/article/cookie-based-sql-injection 11: https://defendtheweb.net/article/cracking-zip-files-known-plaintext-attack 12: https://defendtheweb.net/article/cross-site-request-forgery 13: https://defendtheweb.net/article/cross-site-scripting-xss-by-example 14: https://defendtheweb.net/article/defcon-21-the-secret-life-of-sim-cards 15: https://defendtheweb.net/article/defcon-23-looping-surveillance-cameras 16: https://defendtheweb.net/article/heartbleed-the-door-to-secret-communications 17: https://defendtheweb.net/article/how-to-spoof-magnetic-stripes-credit-card-hacking 18: https://defendtheweb.net/article/how-to-use-gpg 19: https://defendtheweb.net/article/hybrid-multiple-layer-encryption 20: https://defendtheweb.net/article/introduction-to-java 21: https://defendtheweb.net/article/introduction-to-jquery-basic-jquery-animations 22: https://defendtheweb.net/article/password-generation-and-safe-offline-storage 23: https://defendtheweb.net/article/php-oop-part-1-abstract-classes-interfaces 24: https://defendtheweb.net/article/php-oop-part-2-serialization-of-objects-variable-variables 25: https://defendtheweb.net/article/php-oop-part-3-magic-methods 26: https://defendtheweb.net/article/php-oop-part-4-exception-handling-namespaces-overview 27: https://defendtheweb.net/article/php-oop-part-5-property-hiding-type-hinting 28: https://defendtheweb.net/article/php-oop-part-6-object-types-validation-documenting 29: https://defendtheweb.net/article/practical-applications-of-cross-site-scripting-xss 30: https://defendtheweb.net/article/practical-applications-of-directory-traversals 31: https://defendtheweb.net/article/second-order-sql-injections 32: https://defendtheweb.net/article/security-of-virtual-hosts-traffic-cessation 33: https://defendtheweb.net/article/serious-vulnerability-in-excel-sheets-vba-bruteforce 34: https://defendtheweb.net/article/shell-via-lfi-and-proc-self-environ 35: https://defendtheweb.net/article/sqlmap-tutorial-to-your-first-sql-injection-tool 36: https://defendtheweb.net/article/ted-cracking-stuxnet-a-21st-century-cyber-weapon 37: https://defendtheweb.net/article/ted-gps-spoofing 38: https://defendtheweb.net/article/ted-hackers-the-internet-s-immune-system 39: https://defendtheweb.net/article/ted-how-the-nsa-betrayed-the-world-s-trust 40: https://defendtheweb.net/article/ted-nsa-is-spying-on-us-but-there-are-others 41: https://defendtheweb.net/article/understanding-source-code 42: https://defendtheweb.net/article/warriors-of-the-net 43: https://defendtheweb.net/article/what-is-cross-site-scripting-xss 44: https://defendtheweb.net/article/xslt-inclusion-vulnerabilities 45: https://defendtheweb.net/articles/encryption-steganography 46: https://defendtheweb.net/articles/lock-picking 47: https://defendtheweb.net/articles/misc 48: https://defendtheweb.net/articles/network-security 49: https://defendtheweb.net/articles/phreaking 50: https://defendtheweb.net/articles/submit 51: https://defendtheweb.net/auth/facebook 52: https://defendtheweb.net/auth/forgot 53: https://defendtheweb.net/auth/github 54: https://defendtheweb.net/auth/google 55: https://defendtheweb.net/auth/twitter 56: https://defendtheweb.net/discussion/119-level-4	

REPORT 00:00.00.000

[-] KEYWORD Collections.Log List \${OUT_OF_SCOPE_LINKS}

Documentation: Logs the length and contents of the `list` using given `level`.

Start / End / Elapsed: 20210816 18:12:34.133 / 20210816 18:12:34.133 / 00:00:00.000

18:12:34.133 INFO List length is 69 and it contains following items:
 0: <http://creativecommons.org/licenses/by/3.0/>
 1: <http://creativecommons.org/terms>
 2: <http://r00t.org/>
 3: <http://www.noiseprotocol.org/>
 4: <http://www.shal-online.com/>
 5: <http://www.thespanner.co.uk/2014/03/21/rpo/>
 6: <https://amzn.to/2GAQAgG>
 7: <https://amzn.to/2LMySF2>
 8: <https://amzn.to/2Ysdsm>
 9: <https://api.pwnedpasswords.com/range/8cb22>.
 10: <https://creativecommons.org/licenses/by-sa/4.0/>
 11: <https://developers.google.com/web/tools/lighthouse/audits/noopener>
 12: https://en.wikipedia.org/wiki/Address_space_layout_randomization
 13: https://en.wikipedia.org/wiki/Buffer_overflow#History
 14: <https://en.wikipedia.org/wiki/K-anonymity>
 15: <https://example.com>
 16: <https://fedoramagazine.org/build-a-virtual-private-network-with-wireguard/>
 17: <https://fedoramagazine.org/howto-use-sudo/>
 18: <https://gist.github.com/masarson/lbeca41f42599db1c6d48a89e135f653>
 19: <https://github.com/Iskuri>
 20: <https://hal.inria.fr/hal-02100345>
 21: <https://haveibeenpwned.com/>
 22: <https://haveibeenpwned.com/API/v2#PwnedPasswords>.
 23: <https://httpd.apache.org/docs/2.4/logs.html#accesslog>
 24: <https://keepers.space>
 25: <https://krebsonsecurity.com>
 26: <https://krebsonsecurity.com/2021/08/microsoft-patch-tuesday-august-2021-edition/>
 27: <https://krebsonsecurity.com/2021/08/new-anti-anti-money-laundering-services-for-crooks/>
 28: <https://krebsonsecurity.com/2021/08/phishing-sites-targeting-scammers-and-thieves/>
 29: <https://krebsonsecurity.com/2021/08/ransomware-gangs-and-the-name-game-distraction/>
 30: <https://phrack.org/issues/49/14.html>
 31: <https://portswigger.net/research/detecting-and-exploiting-path-relative-stylesheet-import-prssi-vulnerabilities>
 32: <https://root-me.org>
 33: <https://stackoverflow.com/questions/10446819/do-apache-access-logs-ever-miss-requests>
 34: <https://threatpost.com>
 35: <https://threatpost.com/adload-malware-apple-xprotect/168634/>
 36: <https://threatpost.com/aggah-wordpress-spearphishing/168657/>
 37: <https://threatpost.com/amazons-track-worker-keystrokes/168687/>
 38: <https://threatpost.com/black-hat-novel-dns-hack/168636/>
 39: <https://threatpost.com/cyberattackers-captchas-phishing-malware/168684/>
 40: <https://threatpost.com/exchange-servers-attack-proxyshell/168661/>
 41: <https://threatpost.com/qr-code-scammers-bitcoin-atms/168621/>
 42: <https://threatpost.com/ransomware-payments-quadruple-extortion/168622/>
 43: <https://threatpost.com/rogue-marketplace-alphabay-reboots/168648/>
 44: <https://threatpost.com/solarwinds-financial-crisis-podcast/168677/>
 45: https://unsplash.com/@blackzheng?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText
 46: https://unsplash.com/@shanerounce?utm_source=unsplash&utm_medium=referral&utm_content=creditCopyText
 47: [https://www.danielmills.org/downloads\(buffer_overflow/escalate.cpp](https://www.danielmills.org/downloads(buffer_overflow/escalate.cpp)
 48: [https://www.danielmills.org/downloads\(buffer_overflow/shellcode.txt](https://www.danielmills.org/downloads(buffer_overflow/shellcode.txt)
 49: <https://www.darkreading.com>
 50: https://www.darkreading.com/application-security/http-2-implementation-errors-exposing-websites-to-serious-risks/d/d-id/1341593?mc=rss_x_drr_edt_aud_dr_x_x-rss-simple
 51: https://www.darkreading.com/application-security/researchers-find-significant-vulnerabilities-in-macos-privacy-protections/d/d-id/1341590?mc=rss_x_drr_edt_aud_dr_x_x-rss-simple
 52: https://www.darkreading.com/careers-and-people/organizations-still-struggle-to-hire-and-retain-infosec-employees-report/d/d-id/1341587?mc=rss_x_drr_edt_aud_dr_x_x-rss-simple
 53: <https://www.darkreading.com/cloud/new-normal-demands-new-security-leadership-structure/d/d>

[-] KEYWORD Collections.Log List \${ALL_DISCOVERED_FORMS}

REPORT

00:00

Documentation: Logs the length and contents of the `list` using given `level`.

Start / End / Elapsed: 20210816 18:12:34.134 / 20210816 18:12:34.136 / 00:00:00.002

18:12:34.136 INFO List length is 9 and it contains following items:
 0: <form id="passwordTester" onsubmit="event.preventDefault(); checkPassword();">
 <input id="passwordTesterPassword" placeholder="Password" type="text"/>
 <input type="submit" value="Check"/>
 </form>
 1: <form action="/auth" method="post">
 <input name="token" type="hidden" value="0293ebd342495f54cd66ea5e5d2a392b05d3292531df552d4d0f22553e3adccc"/>
 <div class="input">
 <label for="login-username">Username</label>
 <input autocapitalize="none" autocorrect="off" id="login-username" name="username" placeholder="Username" type="text" value="">
 </div>
 <div class="input">
 <label for="login-password">Password</label>
 <input id="login-password" name="password" placeholder="Password" type="password"/>
 </div>
 <div class="input input--checkbox">
 <input id="login-remember" name="remember" type="checkbox"/>
 <label for="login-remember">Remember me</label>
 </div>
 <div class="row">
 <div class="six columns">
 Forgotten password?
 </div>
 <div class="six columns">
 <button class="button button--main button--wide" type="submit">Log in</button>
 </div>
 </div>
</form>
 2: <form action="/auth/signup" method="post">
 <input name="token" type="hidden" value="0293ebd342495f54cd66ea5e5d2a392b05d3292531df552d4d0f22553e3adccc"/>
 <div class="input">
 <label for="signup-username">Username</label>
 <input autocapitalize="none" autocorrect="off" id="signup-username" name="username" placeholder="Username" type="text" value="">
 </div>
 <div class="input">
 <label for="signup-email">Email</label>
 <input autocapitalize="none" autocorrect="off" id="signup-email" name="email" placeholder="Email" type="email" value="">
 </div>
 <div class="input">
 <label for="signup-password">Password</label>
 <input autocomplete="new-password" id="signup-password" name="password" placeholder="Password" type="password"/>
 </div>
 <p>By providing my information and clicking on the sign up button, I confirm that I have read and agree to this website's terms of use and privacy policy.</p>
 <p>All members must abide by the Code of Conduct. Failing to do so will result in your account being terminated.</p>
 <div class="row">
 <div class="six columns">
 </div>
 </div>
 <div class="six columns">
 <button class="button button--main button--wide" type="submit">Sign up</button>
 </div>
</form>

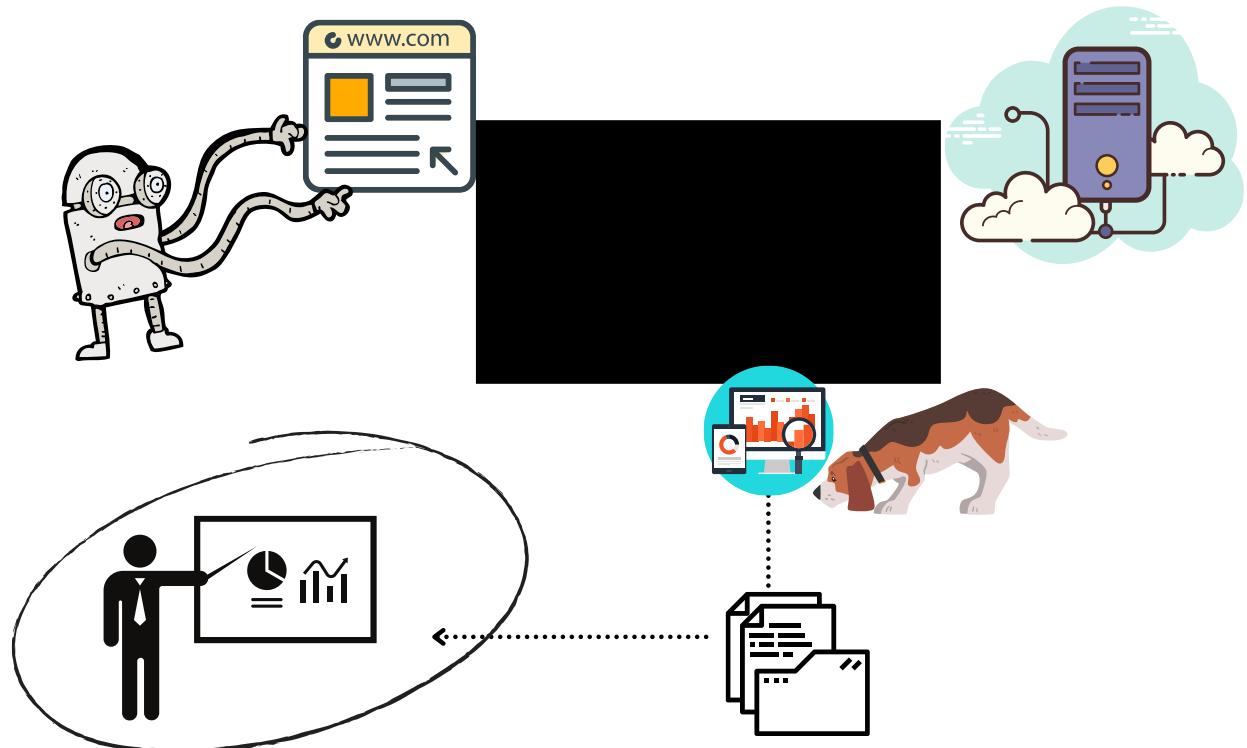
KEYWORD	Collections.Log List \${ALL_FORMS_ATTRIBUTES_AND_INPUT_ELEMENTS}	00:00:00.001	
Documentation:	Logs the length and contents of the <code>list</code> using given <code>level</code> .		
Start / End / Elapsed:	20210816 18:12:34.136 / 20210816 18:12:34.137 / 00:00:00.001		
18:12:34.137	INFO	List length is 9 and it contains following items: 0: {'Form attributes': {'id': 'passwordTester', 'onsubmit': 'event.preventDefault(); checkPassword();'}, 'Input elements': [{id': 'passwordTesterPassword', 'placeholder': 'Password', 'type': 'text'}, {'type': 'submit', 'value': 'Check'}]} 1: {'Form attributes': {'action': '/auth', 'method': 'post'}, 'Input elements': [{name': 'token', 'type': 'hidden', 'value': '0293ebd342495f54cd66ea5e5d2a392b05d3292531df552d4d0f22553e3adccc'}, {'autocapitalize': 'none', 'autocomplete': 'off', 'id': 'login-username', 'name': 'username', 'placeholder': 'Username', 'type': 'text', 'value': ''}, {'id': 'login-password', 'name': 'password', 'placeholder': 'Password', 'type': 'password'}, {'id': 'login-remember', 'name': 'remember', 'type': 'checkbox'}]} 2: {'Form attributes': {'action': '/auth/signup', 'method': 'post'}, 'Input elements': [{name': 'token', 'type': 'hidden', 'value': '0293ebd342495f54cd66ea5e5d2a392b05d3292531df552d4d0f22553e3adccc'}, {'autocapitalize': 'none', 'autocomplete': 'off', 'id': 'signup-username', 'name': 'username', 'placeholder': 'Username', 'type': 'text', 'value': ''}, {'autocapitalize': 'none', 'autocomplete': 'off', 'id': 'signup-email', 'name': 'email', 'placeholder': 'Email', 'type': 'email', 'value': ''}, {'autocomplete': 'new-password', 'id': 'signup-password', 'name': 'password', 'placeholder': 'Password', 'type': 'password'}]} 3: {'Form attributes': {'action': '/auth/signup', 'method': 'post'}, 'Input elements': [{name': 'token', 'type': 'hidden', 'value': 'la6ea3d367810788b47f6507266d6c732ea9167765815ccb2f468f70cdda3f95'}, {'autocapitalize': 'none', 'autocomplete': 'off', 'id': 'signup-username', 'name': 'username', 'placeholder': 'Username', 'type': 'text', 'value': ''}, {'autocapitalize': 'none', 'autocomplete': 'off', 'id': 'signup-email', 'name': 'email', 'placeholder': 'Email', 'type': 'email', 'value': ''}, {'autocomplete': 'new-password', 'id': 'signup-password', 'name': 'password', 'placeholder': 'Password', 'type': 'password'}]} 4: {'Form attributes': {'action': '/auth', 'method': 'post'}, 'Input elements': [{name': 'token', 'type': 'hidden', 'value': 'la6ea3d367810788b47f6507266d6c732ea9167765815ccb2f468f70cdda3f95'}, {'autocapitalize': 'none', 'autocomplete': 'off', 'id': 'login-username', 'name': 'username', 'placeholder': 'Username', 'type': 'text', 'value': ''}, {'id': 'login-password', 'name': 'password', 'placeholder': 'Password', 'type': 'password'}, {'id': 'login-remember', 'name': 'remember', 'type': 'checkbox'}]} 5: {'Form attributes': {'action': '/shop/donate', 'class': 'form--join-the-list', 'enctype': 'multipart/form-data', 'method': 'POST'}, 'Input elements': [{class': 'u-full-width', 'id': 'token', 'maxlength': '', 'name': 'token', 'placeholder': '', 'type': 'hidden', 'value': 'f5bedc7a648fda02af47e054ffffb96c5e6375d22ed752b33ecb552487916534'}, {class': 'u-full-width', 'id': 'formid', 'maxlength': '', 'name': 'formid', 'placeholder': '', 'type': 'hidden', 'value': '8b7f3d48b3735616b721f804fc8aa06a'}, {class': 'u-full-width', 'id': 'product', 'maxlength': '', 'name': 'product', 'placeholder': '', 'type': 'hidden', 'value': '78cb2d7256563076ef63fa19a941b4bf'}, {class': 'u-full-width', 'id': 'amount', 'maxlength': '', 'name': 'amount', 'placeholder': 'Amount', 'type': 'text', 'value': ''}]} 6: {'Form attributes': {'action': '?contact-us', 'class': 'form--contact-us', 'enctype': 'multipart/form-data', 'method': 'POST'}, 'Input elements': [{class': 'u-full-width', 'id': 'token', 'maxlength': '', 'name': 'token', 'placeholder': '', 'type': 'hidden', 'value': 'd4c1185f930d25194e170350e0b5740b06c8004786e3808858adfe9022cd4fd5'}, {class': 'u-full-width', 'id': 'formid', 'maxlength': '', 'name': 'formid', 'placeholder': '', 'type': 'hidden', 'value': '78cb2d7256563076ef63fa19a941b4bf'}, {class': 'u-full-width', 'id': 'email', 'maxlength': '', 'name': 'email', 'placeholder': 'Email', 'type': 'email', 'value': ''}]} 7: {'Form attributes': {'action': '/auth/signup', 'method': 'post'}, 'Input elements': [{name': 'token', 'type': 'hidden', 'value': 'd4c94b813466a77d52cf1e8ed1d2d750878ab603ffe476ba7aa6a6788f905ff'}, {'autocapitalize': 'none', 'autocomplete': 'off', 'id': 'signup-username', 'name': 'username', 'placeholder': 'Username', 'type': 'text', 'value': ''}, {'autocapitalize': 'none', 'autocomplete': 'off', 'id': 'signup-email', 'name': 'email', 'placeholder': 'Email', 'type': 'email', 'value': ''}, {"autocomplete": "new-password", "id": "signup-password", "name": "password", "placeholder": "Password", "type": "password"}]} 8: {'Form attributes': {'action': '/auth', 'method': 'post'}, 'Input elements': [{name': 'token', 'type': 'hidden', 'value': 'd4c94b813466a77d52cf1e8ed1d2d750878ab603ffe476ba7aa6a6788f905ff'}, {'autocapitalize': 'none', 'autocomplete': 'off', 'id': 'login-username', 'name': 'username', 'placeholder': 'Username', 'type': 'text', 'value': ''}, {'id': 'login-password', 'name': 'password', 'placeholder': 'Password', 'type': 'password'}, {"id": "login-remember", "name": "remember", "type": "checkbox"}]}	

Questions:

- If you are asked to **repeat** the test steps, which method would you prefer? Would you prefer manual approach, or would you just re-trigger test execution while you comfortably sip a cup of coffee (or whatever it is you like).
- Do you see how the second approach could provide more information in less time, while also leaving you less exhausted, more refreshed and focused for the next step (which is to identify entry points)?

11. mitmproxy: Intercepting API Requests & Responses

11.1 Scenario



11.2 Install mitmproxy

1. Install [mitmproxy](#) tool
2. Set LANG environment variable

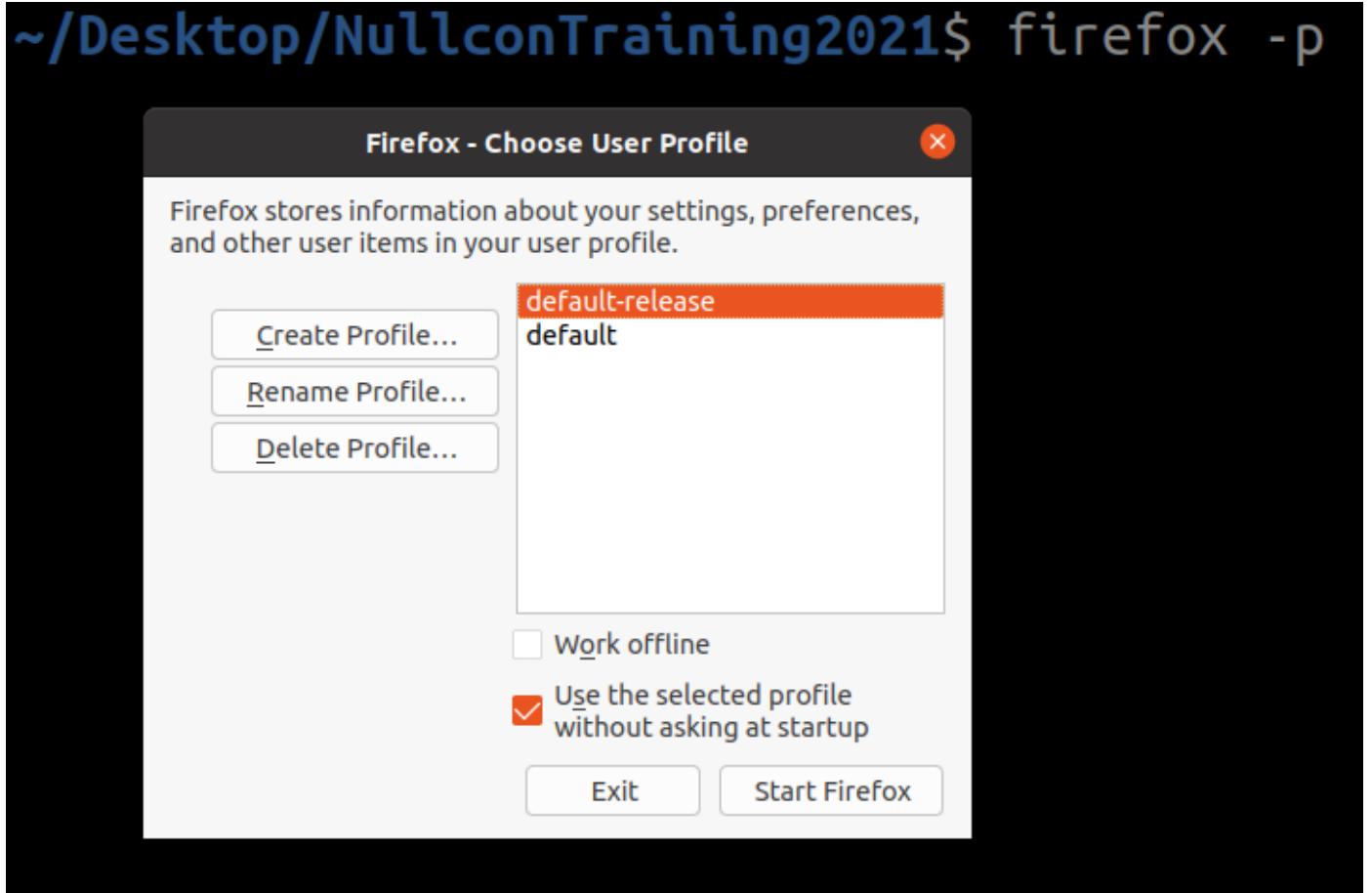
```
export LANG=en_US.UTF-8
```

```
secqation@mirage:/$ mitmproxy
Error: mitmproxy requires a UTF console environment.
Set your LANG environment variable to something like en_US.UTF-8
secqation@mirage:/$ export LANG=en_US.UTF-8
secqation@mirage:/$
secqation@mirage:/$
secqation@mirage:/$ mitmproxy
secqation@mirage:/$ mitmdump
Proxy server listening at http://*:8080
```

11.3 Create Browser Profile

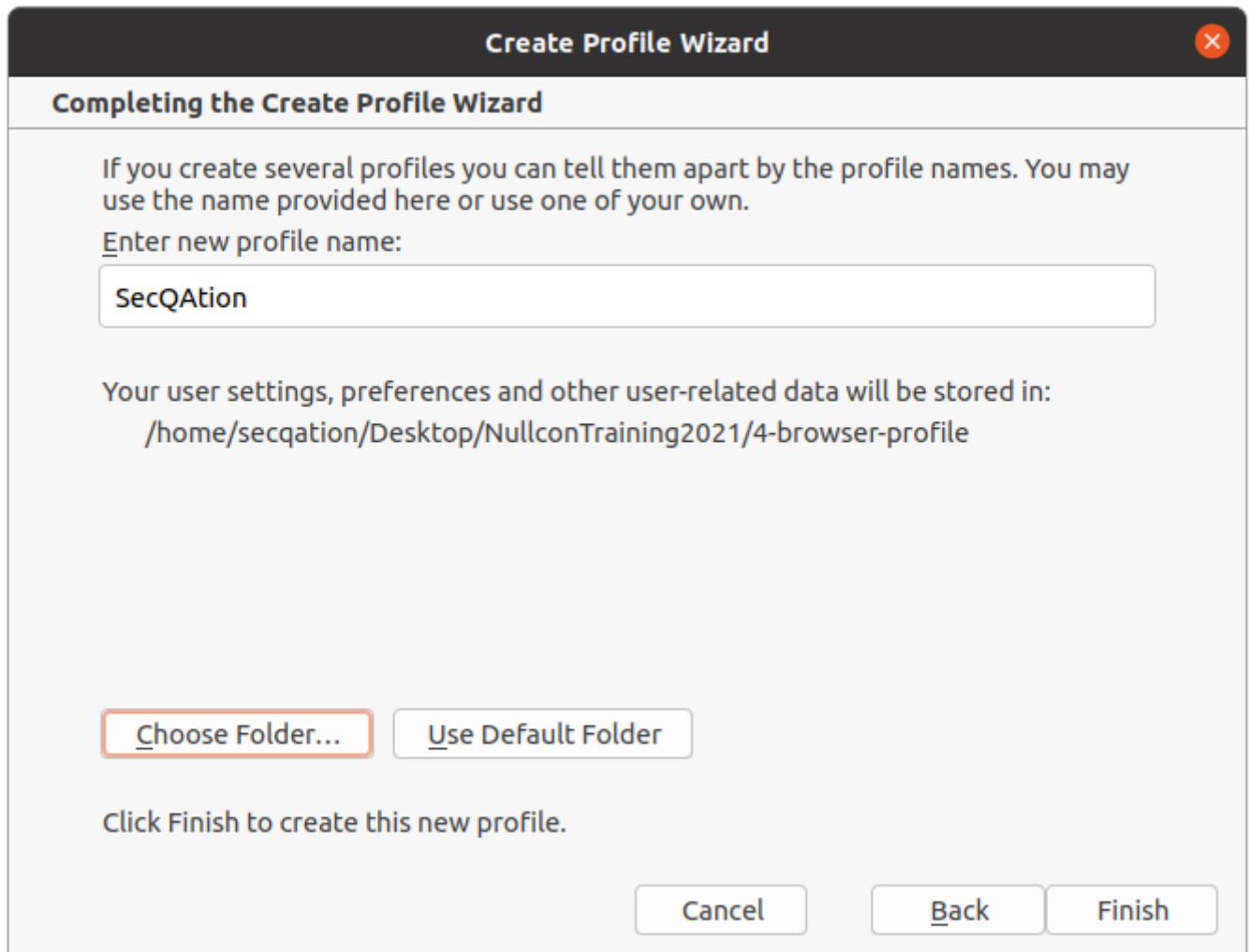
1. Ensure Firefox browser is installed
2. Run following command

```
firefox -p
```



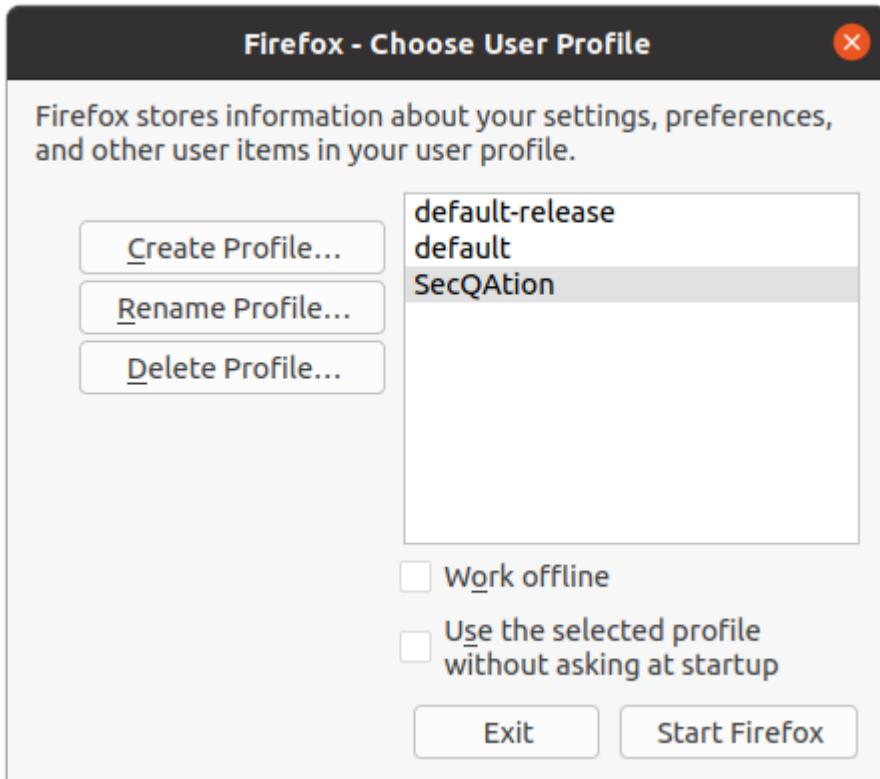
3. Click on "Create Profile" > "Next"
4. Enter a profile name (e.g. `SecQAtion`)
5. Choose a folder path

6. Click on "Finish"

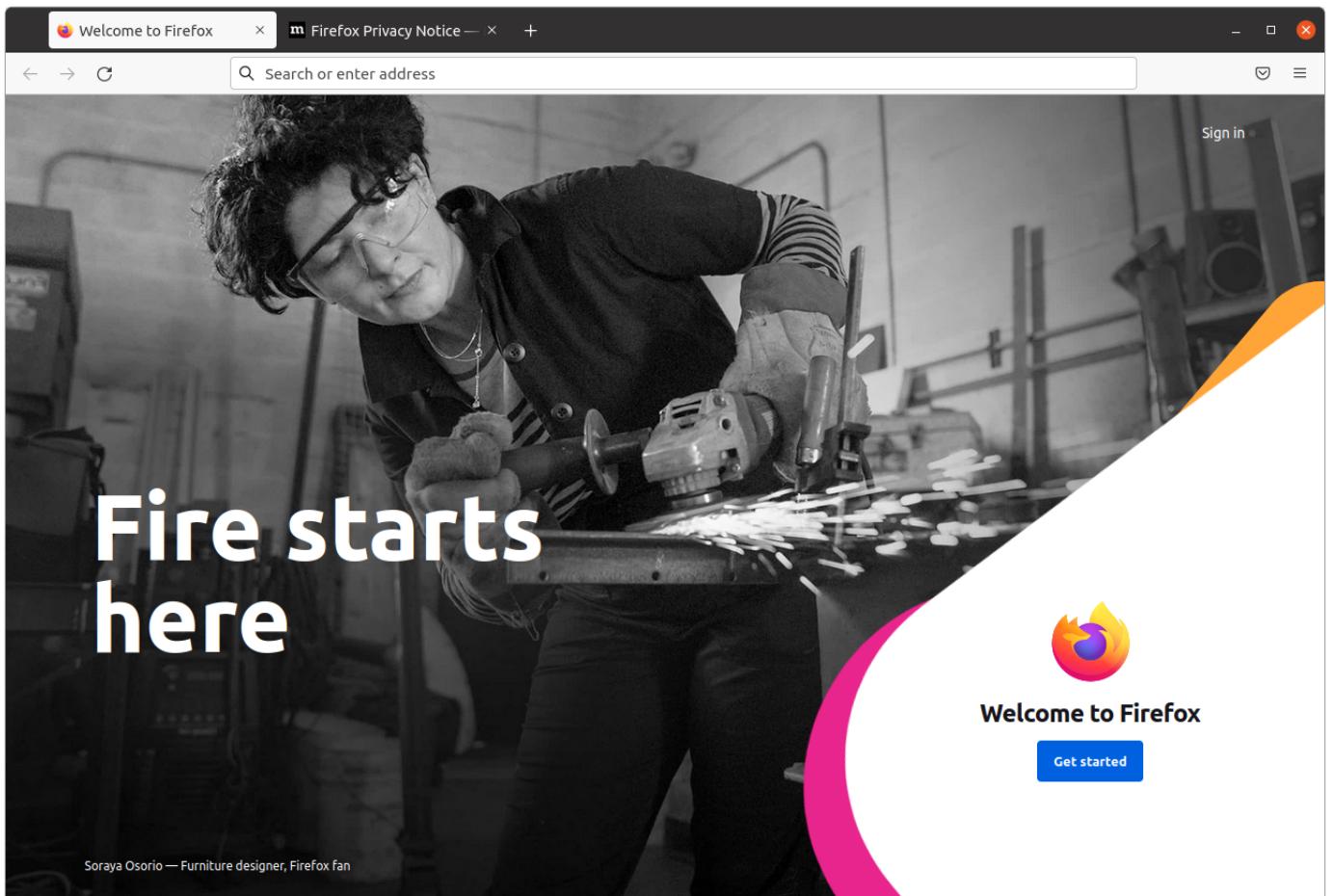


7. Select newly created profile

8. Uncheck **Use the selected profile without asking at startup** checkbox



9. Click on "Start Firefox"

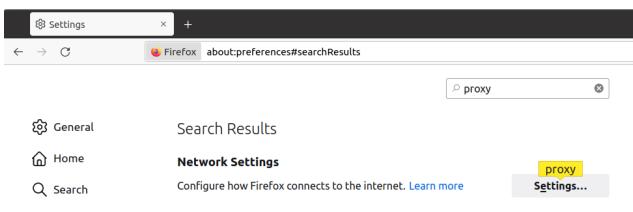


11.4 Configure Browser Profile

11.4.1 Enable Proxy

1. In the Firefox browser, go to "Settings" page

2. Search for "Network Settings"



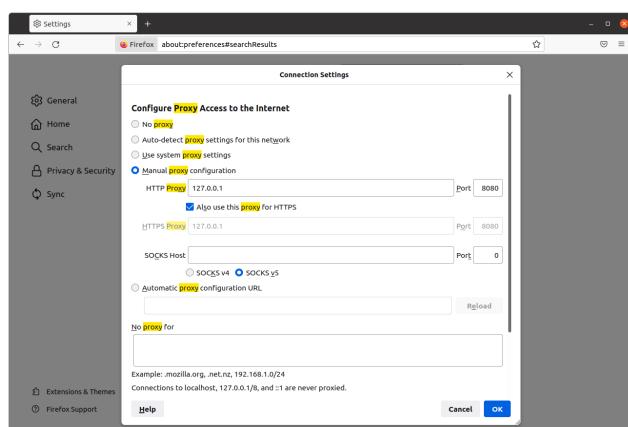
3. Click on "Settings..." button

4. Select "Manual proxy configuration" radio button

5. Enter **127.0.0.1** as HTTP Proxy value

6. Enter **8080** as HTTP Port value

7. Check "Also use this proxy for HTTPS" checkbox



8. Click on "OK" button

11.4.2 Install mitmproxy CA Certificate

Mitmproxy can decrypt encrypted traffic on the fly, as long as the client trusts mitmproxy's built-in certificate authority. Usually this means that the mitmproxy CA certificate has to be installed on the client device.

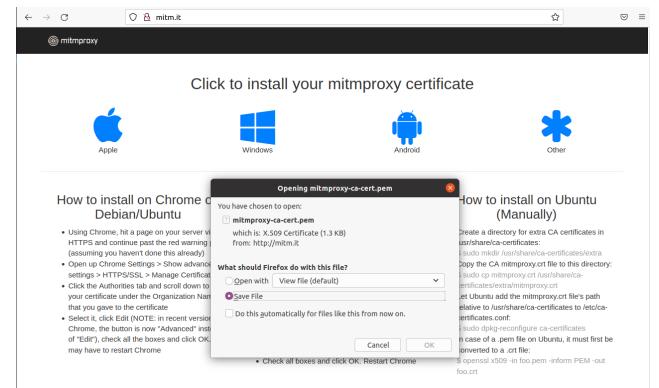
Ref:

<https://docs.mitmproxy.org/stable/concepts-certificates/>

mitmdump

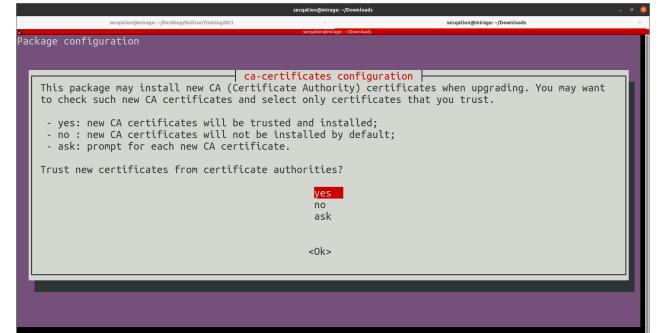
```
secqation@mirage:~$ mitmdump
Proxy server listening at http://*:8080
```

10. In the Firefox browser, navigate to mitm.it
11. Download relevant certificate

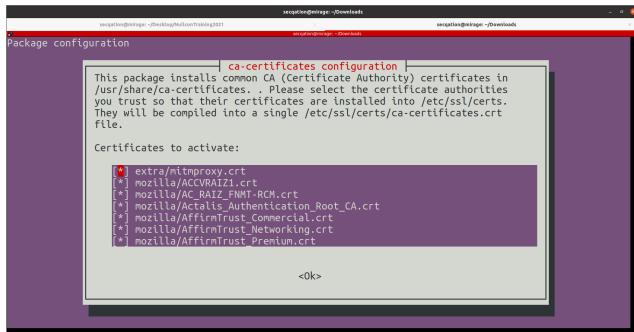


12. Follow certificate installation instructions

```
$ cd ~/Downloads $ openssl x509 -in mitmproxy-ca-cert.pem -inform PEM -c
```



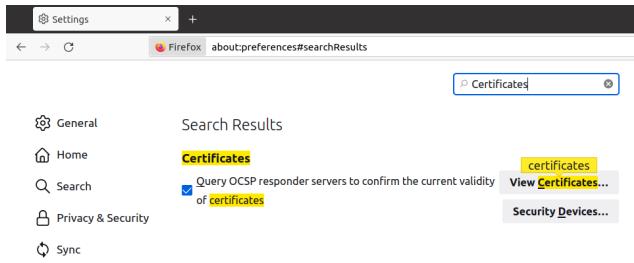
13. Select "yes" when asked to trust new certificates from certificate authorities
14. Select the newly added certificate by pressing [TAB] button,



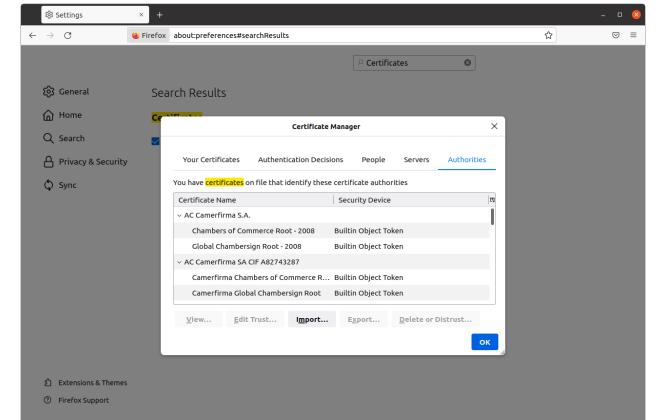
15. Press **[TAB]** and **[ENTER]** to select "Ok"

```
secquestion@mirage:~/Downloads$ ls mitmproxy-ca-cert.pem
secquestion@mirage:~/Downloads$ openssl x509 -in mitmproxy-ca-cert.pem -inform PEM -out mitmproxy-ca-cert.crt
secquestion@mirage:~/Downloads$ ll mitmproxy-ca-cert.crt
-rw-r--r-- 1 secquestion secquestion 1318 Aug 17 01:59 mitmproxy-ca-cert.pem
secquestion@mirage:~/Downloads$ sudo mkdir /usr/share/ca-certificates/extra
[sudo] password for secquestion:
secquestion@mirage:~/Downloads$ sudo cp mitmproxy-ca-cert.crt /usr/share/ca-certificates/extra/mitmproxy.crt
secquestion@mirage:~/Downloads$ sudo dpkg-reconfigure ca-certificates
Processing triggers for ca-certificates (20210119-20.04.1) ...
Updating certificates in /etc/ssl/certs...
0 added, 0 removed; done.
Running hooks in /etc/ca-certificates/update.d...
done.
secquestion@mirage:~/Downloads$
```

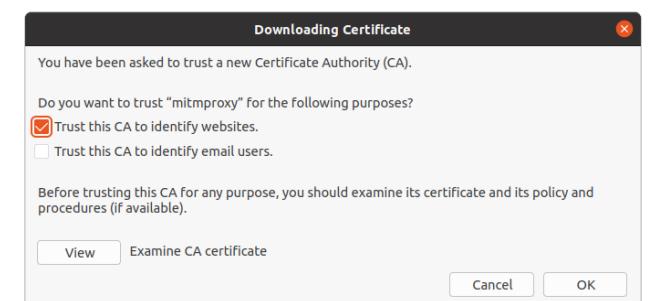
16. In Firefox browser, go to "Settings" page 17. Search for "Certificates"



18. Click on "View Certificates" 19. Click on "Import"



20. Choose the CA certificate "mitmproxy-ca-cert.crt" 21. Select the checkbox labeled as "Trust this CA to identify websites."



22. Click on "OK"

11.5 Verify Setup

1. Open a terminal window by pressing **[CTRL]+[ALT]+T**
2. Start mitmproxy by running following command:

mitmproxy

3. Open Firefox browser, and load the newly created browser profile (with proxy mode enabled)
4. Navigate to <https://defendtheweb.net/>
5. The secure website should load successfully

6. You should see the captured server requests and responses in mitmproxy screen

The screenshot shows two windows side-by-side. On the left, a terminal window titled 'Flows' displays a list of network requests and responses. The requests include GETs to Google's search autocomplete endpoint and Mozilla's telemetry submission endpoint. The responses show various status codes like 200 and 304. On the right, a web browser window is open to the URL 'https://defendtheweb.net'. The page title is 'Defend the Web: An Interactive Cyber Security Platform'. It features a green button labeled '[Get started]' and a link to 'Login | Sign up'. Below the main content, there is a section titled 'Let's protect the web together' with a brief description of the platform's purpose and a 'Learn' section with a small icon.

7. Your browser is configured and ready for use.

11.6 What Next?

1. mitmproxy should intercept all server requests and responses while automated test cases get executed by the robot framework
2. Irrelevant browser traffic should be ignored
3. All of the captured traffic should be saved to a file
4. It should be possible to open the saved output file and analyze the intercepted traffic

11.7 Prepare Bash Script

init.sh

```
mitmdump -p 8080 -w +traffic.mitm "!" ~u firefox|ocsp|mozilla|googleapis" & robot -d ~/PycharmProjects/secqation/nullcon2021/Results ~/PycharmProjects/secqation
```

Note: You must update the **absolute path** to "Results" and "Tests" folder as per your own directory structure.

```
secqation@mirage:~/PycharmProjects/secqation/nullcon2021/BashScripts$ ll
total 12
drwxr-xr-x 2 secqation seqation 4096 Aug 17 10:39 .
drwxr-xr-x 8 secqation seqation 4096 Aug  9 19:39 ..
-rw-r--r-- 1 secqation seqation 225 Aug 17 10:39 init.sh
secqation@mirage:~/PycharmProjects/secqation/nullcon2021/BashScripts$ cat init.sh
mitmdump -p 8080 -w +traffic.mitm "!" ~u firefox|ocsp|mozilla|googleapis" &
robot -d ~/PycharmProjects/secqation/nullcon2021/Results ~/PycharmProjects/secqation/nullcon2021/Tests/App.
robot
pkill mitmdump
echo "Test Complete."
```

11.8 Run Bash Script

Run following two commands, and take a short break:

```
chmod +x init.sh ./init.sh
```

```
secqation@mirage:~/PycharmProjects/secqation/nullcon2021/BashScripts$ chmod +x init.sh
secqation@mirage:~/PycharmProjects/secqation/nullcon2021/BashScripts$ ./init.sh
=====
App
=====
Proxy server listening at http://*:8080
Information Gathering | PASS |
-----
App | PASS |
1 test, 1 passed, 0 failed
=====
Output: /home/secqation/PycharmProjects/secqation/nullcon2021/Results/output.xml
Log:   /home/secqation/PycharmProjects/secqation/nullcon2021/Results/log.html
Report: /home/secqation/PycharmProjects/secqation/nullcon2021/Results/report.html
Test Complete.
secqation@mirage:~/PycharmProjects/secqation/nullcon2021/BashScripts$ ll
total 12
drwxr-xr-x 2 secqation seqation 4096 Aug 17 10:45 .
drwxr-xr-x 8 secqation seqation 4096 Aug  9 19:39 ..
-rwxr-xr-x 1 secqation seqation 225 Aug 17 10:39 init.sh*
-rw-r--r-- 1 secqation seqation     0 Aug 17 10:45 traffic.mitm
secqation@mirage:~/PycharmProjects/secqation/nullcon2021/BashScripts$
```

Observation:

- "traffic.mitm" file was created
- File size is 0, indicating no traffic was captured
- What went wrong?

11.9 Update Browser Settings in Robot Framework

1. Locate the **folder** that stores data related to custom Firefox browser profile (that we created earlier), e.g.,

```
/home/secqation/Desktop/NullconTraining2021/4-browser-profile
```

2. In PyCharm, open "Common.robot" file under "Resources" folder
3. Update the value of **global variable** `${FIREFOX_PROXY_PROFILE}`. Set it as absolute path to Firefox profile directory.

```

1 *** Settings ***
2 Library           SeleniumLibrary
3
4 *** Variables ***
5 ${FIREFOX_PROXY_PROFILE} = /home/secqation/Desktop/NullconTraining2021/4-browser-profile
6
7 *** Keywords ***
8 Begin Web Assessment
9     Open Browser  about:blank  ff_profile_dir=${FIREFOX_PROXY_PROFILE}
10    Maximize Browser Window
11    Set Selenium Speed  0
12    Set Selenium Implicit Wait  8 seconds
13    Set Selenium Timeout  10 seconds
14    Delete All Cookies
15
16 End Web Assessment
17     Capture Page Screenshot
18     Close All Browsers

```

11.10 Re-run Bash Script

1. Re-run the bash script

```
./init.sh
```

```

App | PASS |
1 test, 1 passed, 0 failed
=====
Output: /home/secqation/PycharmProjects/secqation/nullcon2021/Results/output.xml
Log:   /home/secqation/PycharmProjects/secqation/nullcon2021/Results/log.html
Report: /home/secqation/PycharmProjects/secqation/nullcon2021/Results/report.html
Test Complete.
secqation@mirage:~/PycharmProjects/secqation/nullcon2021/BashScripts$ ll
total 5388
drwxr-xr-x 2 secqation secqation 4096 Aug 17 14:19 .
drwxr-xr-x 8 secqation secqation 4096 Aug  9 19:39 ..
-rw-rxr-x 1 secqation secqation 225 Aug 17 14:18 init.sh*
-rw-r--r-- 1 secqation secqation 5504528 Aug 17 14:30 traffic.mitm
secqation@mirage:~/PycharmProjects/secqation/nullcon2021/BashScripts$ 
```

2. Notice that server traffic has been captured and saved successfully in the file

```
traffic.mitm
```

3. Run following command to analyze the saved traffic

```
mitmweb -r traffic.mitm
```

The screenshot shows the mitmproxy browser interface. On the left, a list of network requests is displayed, including various GET and POST requests to the defendtheweb.net domain. On the right, a detailed view of a selected POST request is shown. The request details pane includes fields for method (POST), URL (https://defendtheweb.net/api/1/articles/view), headers (Content-Type: application/x-www-form-urlencoded; charset=UTF-8, User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:91.0) Gecko/20100101 Firefox/91.0, etc.), and body (id: 142). The status bar at the bottom indicates "auto" and "URLEncoded form".

4. Also, access `log.html` file from Results folder, to view data scraped from the target website

The screenshot shows the mitmproxy browser interface with the "App Log" tab selected. The log displays the results of a web assessment, including test cases for information gathering, login, and password recovery. Key entries include:

- TEST Information Gathering**: Full Name: App.Information.Gathering, Status: PASS.
- KEYWORD Crawl https://defendtheweb.net, defendtheweb.net**: Start / End / Elapsed: 2021/08/17 14:28:35.671 / 2021/08/17 14:30:07.727 / 00:01:32.056.
- KEYWORD Crawl global parameters**: 00:00:00.013.
- KEYWORD Crawl Set target domain \${target_Domain}**: 00:00:00.003.
- KEYWORD Crawl Start crawling \${Target_URL}, \${target_Domain}**: 00:01:31.699.
- KEYWORD Crawl Print crawl status**: 00:00:00.336.
- INFO List length is 9 and it contains following items:** Documentation: Logs the length and contents of the list using given level.
- INFO Start / End / Elapsed: 2021/08/17 14:30:07.726 / 2021/08/17 14:30:07.726 / 00:00:00.000**.

11.11 Apply Human Intelligence Now

While the robots were at work, it was a break for you. With a refreshed mind, now is the time to look at all the gathered data and **identify potential attack vectors**.

The current data set can give you a good understanding about the target application.

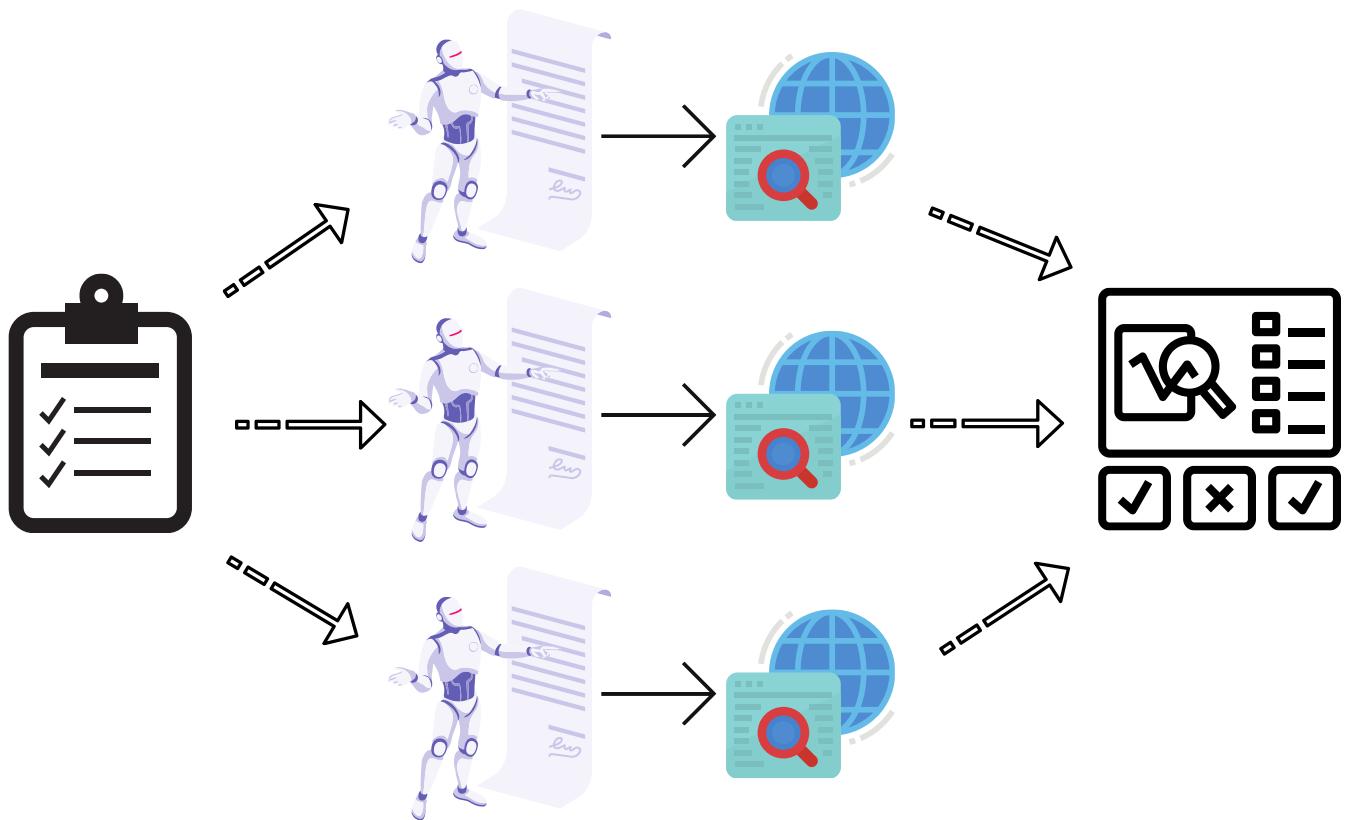
After having a first look at the target application, if there's a need, the robot test cases can be enhanced further to automate the form filling process, or any other manual task, including reading emails from your email account or running commands in your bash shell, etc.

If you have clarity in your mind regarding the exact steps that you wish to perform, writing robot test cases is really a trivial process.

12. Pabot: Parallel Processing

12.1 Pabot

It's a parallel executor for Robot Framework tests. Split one execution into multiple and save test execution time.



Scenario:

What if we wish to perform a set of operations across different pages of a website? Shall we scroll through the pages one-by-one, or is there a way to spin up multiple processes that perform same set of tasks but on different endpoints...

12.2 Installation

Run following command to install Pabot libraries:

```
pip install -U robotframework-pabot
```

12.3 Syntax

- By default, two parallel processes would run:

```
pabot --pabotlib -d ./OutputFolder/ ./TestSuite/
```

- Use `--processes` option to specify specific number of parallel processes that should be run:

```
pabot --pabotlib --processes 6 --testlevelsplits -d ./OutputFolder/ ./TestSuite/
```

12.4 New Keywords

1. Run Setup Only Once
2. Acquire Lock
3. Release Lock
4. Set Parallel Value For Key
5. Get Parallel Value For Key
6. etc.

12.5 Exercise

1. Add a test step to export all in-scope links, gathered during the crawling process
2. Consider this list of URLs as an input to another set of test cases
3. Start parallel processes such that each process would pick one unique URL from the input list
4. Use `Pabot` to analyze several URLs at the same time, performing same test steps but on different endpoints

12.6 Hint

Check the `DeepCrawl` test suite in `Tests` folder.

```
secqation@mirage:~/Desktop/NullconTraining2021/2-example$ ll
total 32
drwxr-xr-x 8 secqation seqation 4096 Aug 19 14:58 .
drwxr-xr-x 7 secqation seqation 4096 Aug 19 12:04 ..
drwxr-xr-x 2 secqation seqation 4096 Aug 17 14:57 BashScripts/
drwxr-xr-x 3 secqation seqation 4096 Aug 18 16:21 CustomLibraries/
drwxr-xr-x 3 secqation seqation 4096 Aug 18 12:06 Data/
drwxr-xr-x 3 secqation seqation 4096 Aug 19 12:03 Resources/
drwxr-xr-x 3 secqation seqation 4096 Aug 19 12:03 Results/
drwxr-xr-x 3 secqation seqation 4096 Aug 19 14:56 Tests/
secqation@mirage:~/Desktop/NullconTraining2021/2-example$ tree Tests/Tests/
└── App.robot
    ├── DeepCrawl
    │   ├── DeepCrawl.robot
    │   ├── __init__.robot
    │   ├── parallel_test1.robot
    │   ├── parallel_test2.robot
    │   ├── parallel_test3.robot
    │   ├── parallel_test4.robot
    │   └── parallel_test5.robot
    └── demo.robot

1 directory, 9 files
secqation@mirage:~/Desktop/NullconTraining2021/2-example$
```

1. Access the sample code

```
cd /home/secqation/Desktop/NullconTraining2021/2-example
```

2. Run following command to trigger parallel test execution

```
pabot --pabotlib --processes 6 --testlevelsplits -d Results/ Tests/DeepCrawl/
```

```
seqcation@mirage:~/PycharmProjects/secqation/nullcon2021$ pabot --pabotlib --processes 6 --testlevelsplits -d Results/ Tests/DeepCrawl/
ing Process-2 after 35.0 seconds
2021-08-19 19:31:06.898829 [PID:79469] [1] [ID:1] PASSED DeepCrawl.Parallel Test2.Information Gathering Pro
cess-1 in 43.5 seconds
2021-08-19 19:31:22.812249 [PID:79477] [5] [ID:5] PASSED DeepCrawl.Parallel Test2.Information Gathering Pro
cess-3 in 59.2 seconds
2021-08-19 19:31:23.581107 [PID:79472] [2] [ID:2] still running DeepCrawl.Parallel Test1.Information Gather
ing Process-2 after 60.0 seconds
2021-08-19 19:31:23.639166 [PID:79475] [4] [ID:4] still running DeepCrawl.Parallel Test2.Information Gather
ing Process-2 after 60.0 seconds
2021-08-19 19:31:25.359767 [PID:79475] [4] [ID:4] PASSED DeepCrawl.Parallel Test2.Information Gathering Pro
cess-2 in 61.7 seconds
2021-08-19 19:31:32.754451 [PID:79472] [2] [ID:2] PASSED DeepCrawl.Parallel Test1.Information Gathering Pro
cess-2 in 69.1 seconds
6 tests, 6 passed, 0 failed, 0 skipped.
=====
Output: /home/secqation/PycharmProjects/secqation/nullcon2021/Results/output.xml
Log:   /home/secqation/PycharmProjects/secqation/nullcon2021/Results/log.html
Report: /home/secqation/PycharmProjects/secqation/nullcon2021/Results/report.html
Stopping PabotLib process
Robot Framework remote server at 127.0.0.1:8270 stopped.
PabotLib process stopped
Total testing: 4 minutes 52.39 seconds
Elapsed time: 1 minute 10.49 seconds
seccation@mirage:~/PycharmProjects/secqation/nullcon2021$ pabot --pabotlib --processes 6 --testlevelsplits -d Results/ Tests/DeepCrawl/
```

3. View the execution log by accessing **log.html** in your browser

```
file:///home/secqation/PycharmProjects/secqation/nullcon2021/Results/log.html
```

Test Execution Log

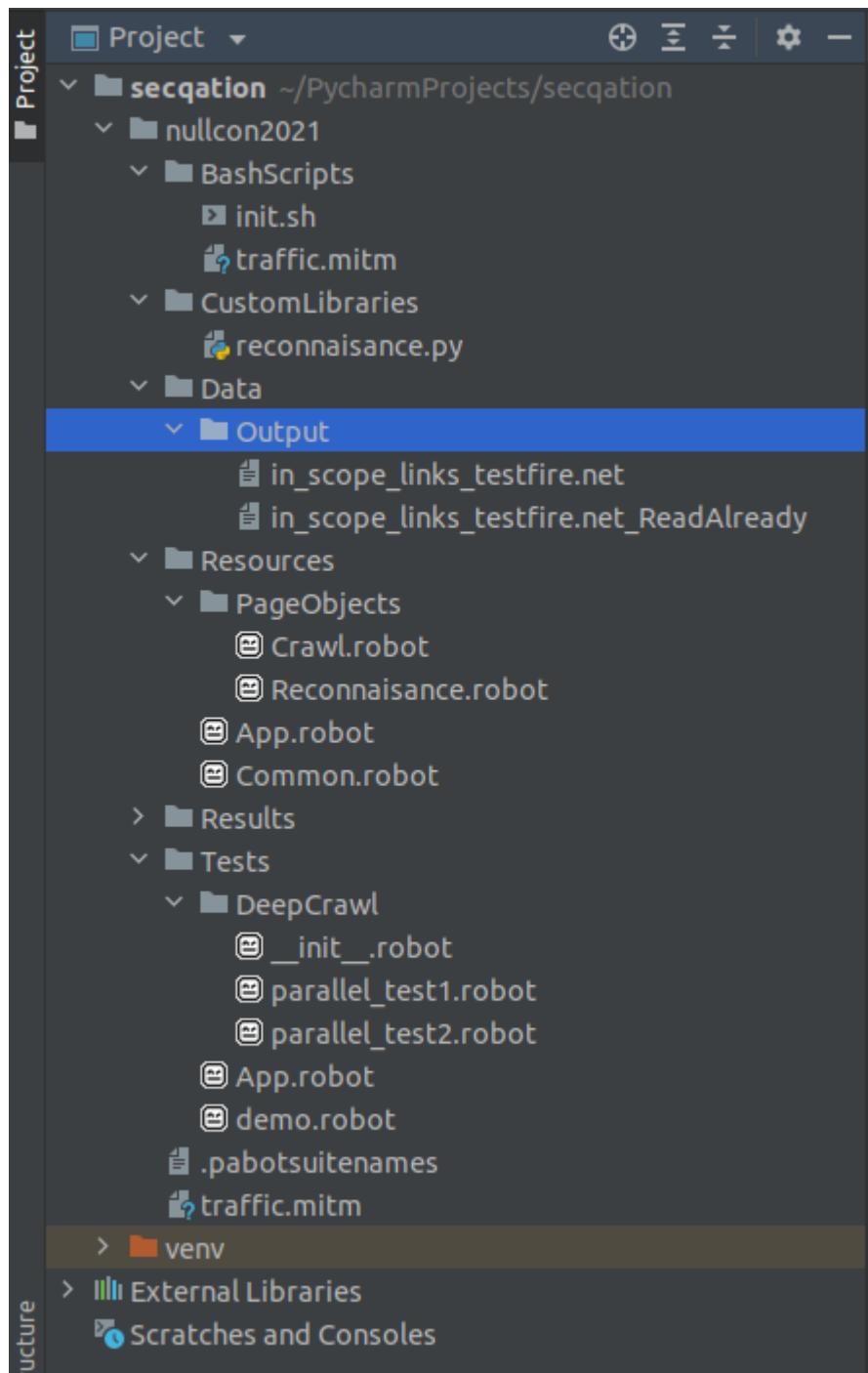
- SUITE DeepCrawl		00:01:03.810			
Full Name:	DeepCrawl				
Documentation:	Pabot result from 6 executions.				
Source:	/home/secqation/PycharmProjects/secqation/nullcon2021/Tests/DeepCrawl				
Start / End / Elapsed:	20210819 12:12:54.100 / 20210819 12:13:57.910 / 00:01:03.810				
Status:	6 tests total, 6 passed, 0 failed, 0 skipped				
+ SETUP	pabot.PabotLib.Run Setup Only Once	Setup Selenium	00:00:00.120		
+ TEARDOWN	Common.End Web Assessment		00:00:00.040		
- SUITE Parallel Test1		00:00:59.219			
Full Name:	DeepCrawl.Parallel Test1				
Source:	/home/secqation/PycharmProjects/secqation/nullcon2021/Tests/DeepCrawl/parallel_test1.robot				
Start / End / Elapsed:	20210819 12:12:56.559 / 20210819 12:13:55.778 / 00:00:59.219				
Status:	3 tests total, 3 passed, 0 failed, 0 skipped				
+ TEST	Information Gathering Process-1		00:00:58.760		
+ TEST	Information Gathering Process-2		00:00:59.154		
+ TEST	Information Gathering Process-3		00:00:58.145		
- SUITE Parallel Test2		00:00:59.847			
Full Name:	DeepCrawl.Parallel Test2				
Source:	/home/secqation/PycharmProjects/secqation/nullcon2021/Tests/DeepCrawl/parallel_test2.robot				
Start / End / Elapsed:	20210819 12:12:56.549 / 20210819 12:13:56.396 / 00:00:59.847				
Status:	3 tests total, 3 passed, 0 failed, 0 skipped				
+ TEST	Information Gathering Process-1		00:00:59.603		
+ TEST	Information Gathering Process-2		00:00:58.218		
+ TEST	Information Gathering Process-3		00:00:59.365		

4. Expand keywords to see captured data

00:0 [RE]

-	TEST	Information Gathering Process-1	
Full Name:	DeepCrawl.Parallel Test1.Information Gathering Process-1		
Start / End / Elapsed:	20210819 12:12:56.670 / 20210819 12:13:55.430 / 00:00:58.760		
Status:	PASS		
+ SETUP	Common.Begin Parallel Test Execution		00:00:47.043
- KEYWORD	App.Deep crawl individual in-scope URLs testfire.net		00:00:07.279
Start / End / Elapsed:	20210819 12:13:43.737 / 20210819 12:13:51.016 / 00:00:07.279		
+ KEYWORD	pabotPabotLib.Acquire Lock read_url		00:00:00.196
+ KEYWORD	Crawl.Read an unread link from file \${OUTPUT_FOLDER_PATH}in_scope_links_\${Domain_Name}		00:00:00.388
+ KEYWORD	pabotPabotLib.Release Lock read_url		00:00:00.012
- KEYWORD	SeleniumLibrary.Go To \${WORKING_ON_URL}		00:00:05.253
Documentation:	Navigates the current browser window to the provided url.		
Start / End / Elapsed:	20210819 12:13:44.446 / 20210819 12:13:49.699 / 00:00:05.253		
12:13:44.451 INFO	Opening url ' http://demo.testfire.net/index.jsp?content=pr/20060413.htm '		
+ KEYWORD	Crawl.Find HTML forms \${WORKING_ON_URL}		00:00:00.003
+ KEYWORD	Crawl.Find Javascript Files \${WORKING_ON_URL}		00:00:00.014
+ KEYWORD	Crawl.Check for presence of sensitive keywords \${WORKING_ON_URL}, admin, password, version, secret, hidden		00:00:00.063
+ KEYWORD	SeleniumLibrary.Capture Element Screenshot xpath://body		00:00:01.230
+ TEARDOWN	SeleniumLibrary.Close Browser		00:00:04.372
-	TEST	Information Gathering Process-2	00:00:59.154
Full Name:	DeepCrawl.Parallel Test1.Information Gathering Process-2		
Start / End / Elapsed:	20210819 12:12:56.567 / 20210819 12:13:55.721 / 00:00:59.154		
Status:	PASS		
+ SETUP	Common.Begin Parallel Test Execution		00:00:47.278
- KEYWORD	App.Deep crawl individual in-scope URLs testfire.net		00:00:06.865
Start / End / Elapsed:	20210819 12:13:43.934 / 20210819 12:13:50.799 / 00:00:06.865		
+ KEYWORD	pabotPabotLib.Acquire Lock read_url		00:00:00.293
+ KEYWORD	Crawl.Read an unread link from file \${OUTPUT_FOLDER_PATH}in_scope_links_\${Domain_Name}		00:00:00.251
+ KEYWORD	pabotPabotLib.Release Lock read_url		00:00:00.022
- KEYWORD	SeleniumLibrary.Go To \${WORKING_ON_URL}		00:00:04.304
Documentation:	Navigates the current browser window to the provided url.		
Start / End / Elapsed:	20210819 12:13:44.792 / 20210819 12:13:49.096 / 00:00:04.304		
12:13:44.840 INFO	Opening url ' http://demo.testfire.net/index.jsp?content=pr/20060518.htm '		
+ KEYWORD	Crawl.Find HTML forms \${WORKING_ON_URL}		00:00:00.306

5. Observe changes in the files stored in the Output folder, before and after test execution completes successfully. Explain your observations.



6. Also check a modified version of the code at following location:

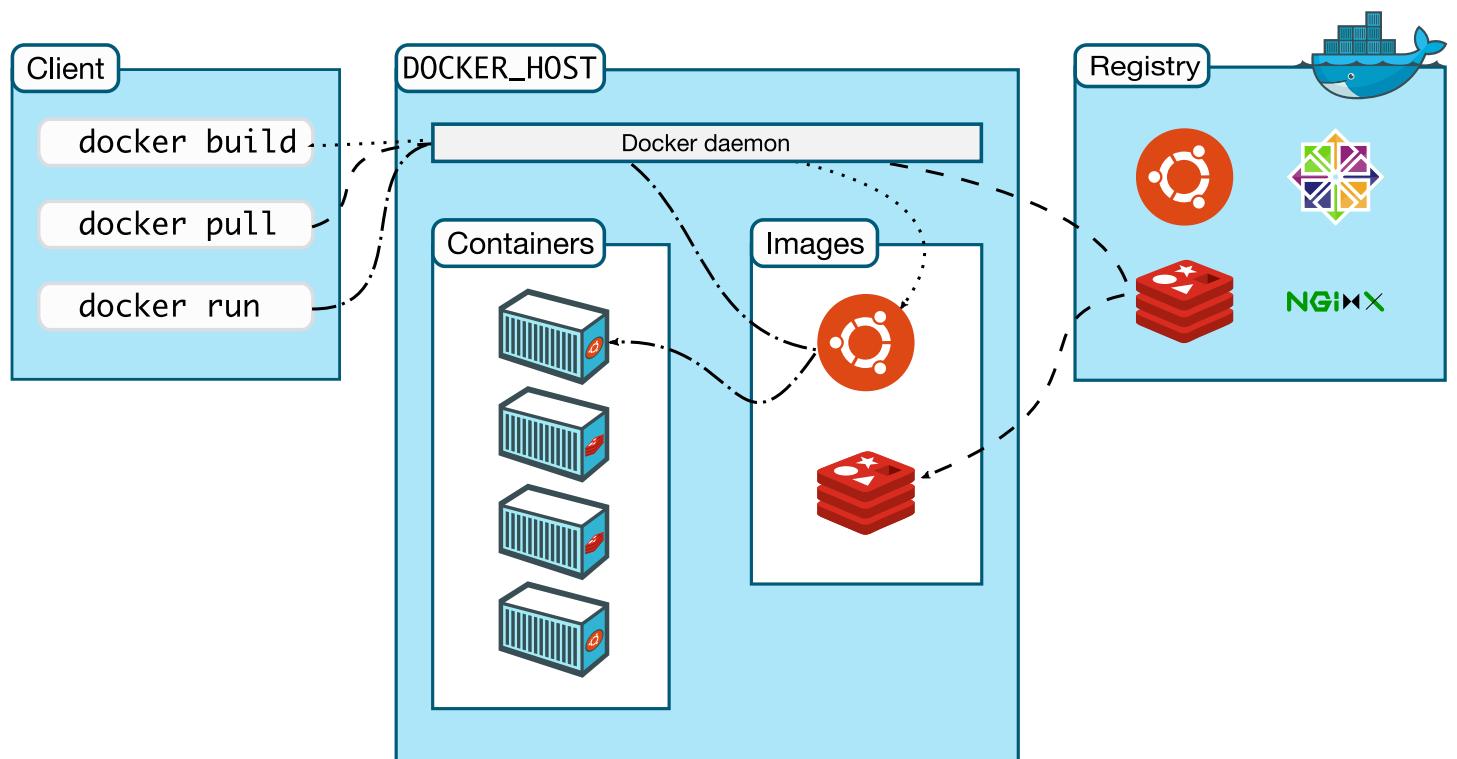
</home/secqation/Desktop/NullconTraining2021/3-example-pabot/>

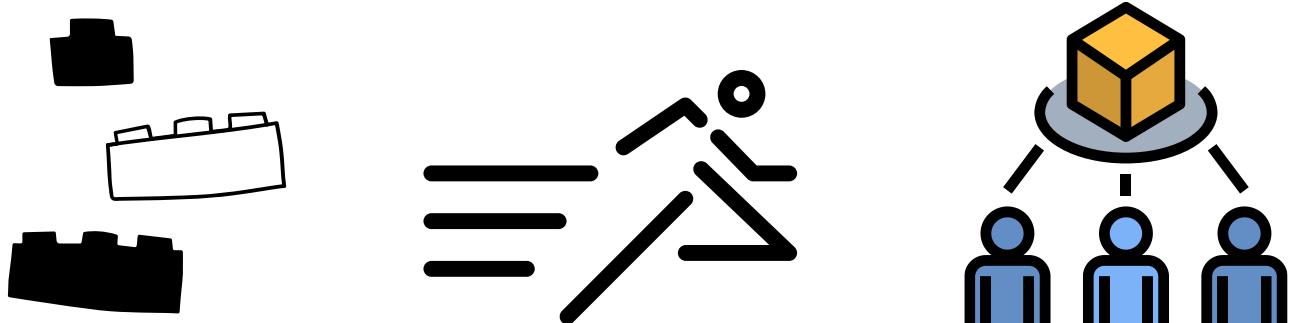
13. Understanding Docker and Docker Compose

Docker provides the ability to **package and run an application** in a loosely isolated environment called a container. The isolation and security allow you to run many containers simultaneously on a given host. Containers are lightweight and contain everything needed to run the application, so you do not need to rely on what is currently installed on the host. You can easily share containers while you work, and be sure that everyone you share with gets the same container that works in the same way.

<https://docs.docker.com/get-started/overview/>

Compose is a tool for **defining and running multi-container Docker applications**. With Compose, you use a YAML file to configure your application's services. Then, with a single command, you create and start all the services from your configuration.





13.1 Pre-requisites:

- Install [Docker](#)

13.2 1. Create Dockerfile

1. Create a new file, called **Dockerfile**
2. Paste following contents

```
FROM python:3.9.5-alpine3.13 LABEL maintainer="Riddhi Shree" ENV PYTHONUNBUFFERED=1 RUN pip install robotframework RUN pip install selenium RUN pip in
```

3. Create **healthcheck.sh** file
4. Paste following contents

```
#!/usr/bin/env bash robot -d ./test_results ./robo-tests
```

5. Create a folder called **demo-test-suite**

```
$ mkdir demo-test-suite $ cd demo-test-suite
```

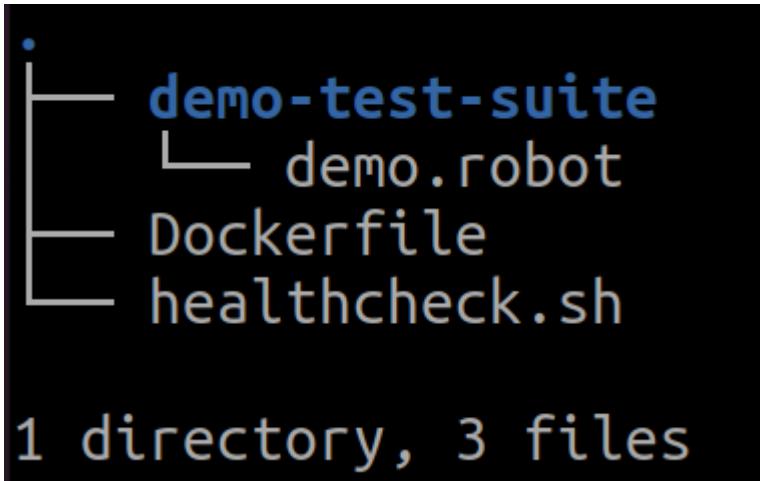
6. Inside **demo-test-suite** folder, create **demo.robot** file

```
$ touch demo.robot
```

7. Paste following code in **demo.robot** file

```
*** Settings ***
Library SeleniumLibrary
Variables ${BASE_URL} = https://www.winja.site
Keywords
Custom keyword Log This is a custom keyword
```

8. The final structure should look like this:



Note: To access readymade files, check following path:

```
/home/secqation/Desktop/NullconTraining2021/4-example-docker/1-Dockerfile/
```

13.3 2. Build Docker Image

1. Run following command to build a new Docker image from **Dockerfile**

```
$ docker build -t demo .
```

```
secqation@mirage:~/Desktop/NullconTraining2021/4-example-docker$ docker build -t demo .
Sending build context to Docker daemon 121.9kB
Step 1/13 : FROM python:3.9.5-alpine3.13
3.9.5-alpine3.13: Pulling from library/python
540db60ca938: Pull complete
d037ddac5dde: Pull complete
629719f9106a: Pull complete
f9ef3a05a91e: Pull complete
2407aae58672: Pull complete
Digest: sha256:cc416abe30153111c005e8c7d69a6817d5e1977512939336ec37d92e4bce1769
Status: Downloaded newer image for python:3.9.5-alpine3.13
--> 46a196bf50ae
Step 2/13 : LABEL maintainer="Riddhi Shree"
--> Running in cce5b6bb0ca0
Removing intermediate container cce5b6bb0ca0
--> 72f38b7916ae
Step 3/13 : ENV PYTHONUNBUFFERED=1
--> Running in efe78b2b8906
Removing intermediate container efe78b2b8906
--> bdf1f486e252
Step 4/13 : RUN pip install robotframework
--> Running in 395ffd4b03e0
Collecting robotframework
  Downloading robotframework-4.1-py2.py3-none-any.whl (657 kB)

```

- Check if image was created successfully

\$ docker images

REPOSITORY	TAG	IMAGE ID	CREATED	SIZE
demo	latest	d9ffb8a44e18	2 minutes ago	59.6MB
python	3.9.5-alpine3.13	46a196bf50ae	7 weeks ago	45MB

13.4 3. Run Docker Container

- Start a running container from the Docker image

\$ docker run -itd demo \$ docker ps -a

```
secqation@mirage:~/Desktop/NullconTraining2021/4-example-docker$ docker run -itd demo
53945980fc4a80031419e67633a759b6d3bb831294234f10bb30dedf669cf381
secqation@mirage:~/Desktop/NullconTraining2021/4-example-docker$ docker ps -a
CONTAINER ID   IMAGE      COMMAND           CREATED          STATUS
S
53945980fc4a   demo      "/bin/sh -c 'sh ./he..."   7 seconds ago   Exited (1) 5 seconds ago
xed_hertz
```

- Read the container logs

\$ docker logs <CONTAINER_ID>

```
secqation@mirage:~/Desktop/NullconTraining2021/4-example-docker$ docker ps -a
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS
S
53945980fc4a      demo                "/bin/sh -c 'sh ./he..."   7 seconds ago     Exited (1) 5 seconds ago
xed_hertz

secqation@mirage:~/Desktop/NullconTraining2021/4-example-docker$ docker logs 53
=====
Robo-Tests
=====
Robo-Tests.Demo
=====
Test Case 1 | FAIL |
WebDriverException: Message: 'geckodriver' executable needs to be in PATH.
-----
Robo-Tests.Demo | FAIL |
1 test, 0 passed, 1 failed
-----
Robo-Tests | FAIL |
1 test, 0 passed, 1 failed
-----
Output: /robo-vault/test_results/output.xml
Log: /robo-vault/test_results/log.html
Report: /robo-vault/test_results/report.html
secqation@mirage:~/Desktop/NullconTraining2021/4-example-docker$ █
```

3. What did you observe?
4. The tests have failed. But, why? Can you fix this?

13.5 4. Create Docker Compose file

1. Download [docker-compose-v3.yml](https://raw.githubusercontent.com/SeleniumHQ/docker-selenium/trunk/docker-compose-v3.yml)

```
$ wget https://raw.githubusercontent.com/SeleniumHQ/docker-selenium/trunk/docker-compose-v3.yml
```

2. Modify the file as follows

```
version: "3" services: firefox_1: image: selenium/node-firefox:4.0.0-rc-1-prerelease-20210804 shm_size: 2gb depends_on: - selenium-hub environment: -
```

3. Modify the contents of **healthcheck.sh** file as follows

```
#!/usr/bin/env bash echo "Checking if hub is ready - $HUB_HOSTNAME" while [[ "$(curl -s http://$HUB_HOSTNAME:4444/wd/hub/status | jq .value.ready)" != "true" ]]
```

4. Add following two lines in **Dockerfile**

```
... RUN apk add curl RUN curl -OL https://github.com/stedolan/jq/releases/download/jq-1.6/jq-linux64 -o ./jq-Linux64 && chmod a+x ./jq-Linux64 && mv
```

5. Run following command to start Selenium Hub and to execute robot test cases:

```
$ docker-compose build $ docker-compose up -d $ docker-compose ps
```

```
secqation@mirage:~/Desktop/NullconTraining2021/4-example-docker/2-docker-compose$ docker-compose -f docker-compose-v3.yml up -d
Creating network "robo-network" with the default driver
Creating selenium-hub ... done
Creating 2-docker-compose_firefox_1_1 ... done
Creating 2-docker-compose_firefox_2_1 ... done
Creating 2-docker-compose_robotframework_1 ... done
secqation@mirage:~/Desktop/NullconTraining2021/4-example-docker/2-docker-compose$ docker-compose -f docker-compose-v3.yml ps
      Name           Command     State    Ports
-----+-----+-----+-----+
2-docker-compose_firefox_1_1   /opt/bin/entry_point.sh   Up      0.0.0.0:6902->5900/tcp,:::6902->5900/tcp
2-docker-compose_firefox_2_1   /opt/bin/entry_point.sh   Up      5900/tcp
2-docker-compose_robotframework_1
selenium-hub                  /bin/sh -c sh ./healthcheck.sh Up      0.0.0.0:4442->4442/tcp,:::4442->4442/tcp, 0.0.0.0:4443->4443/tcp,:::4443->4443/tcp, 0.0.0.0:4444->4444/tcp,:::4444->4444/tcp
secqation@mirage:~/Desktop/NullconTraining2021/4-example-docker/2-docker-compose$ █
```

6. **Selenium Grid** dashboard can be accessed at <http://127.0.0.1:4444>

The screenshot shows the Selenium Grid interface running in a browser window. The URL is 127.0.0.1:4444/ui/index.html#. The main content area displays two grid nodes:

- Node 1:** URI: http://172.24.0.4:5555. Sessions: 0. Max. Concurrency: 1. Queue size: 0.
- Node 2:** URI: http://172.24.0.3:5555. Sessions: 0. Max. Concurrency: 1. Queue size: 0.

A sidebar on the left shows navigation links: Overview, Sessions, and Help. A bottom sidebar shows Concurrency: 0% and 0 / 2.

7. Check the logs to see if robot test cases got executed successfully, or not.

```
$ docker logs 2-docker-compose_robotframework_1
```

```
secqation@mirage:~/Desktop/NullconTraining2021/4-example-docker/2-docker-compose$ docker logs 2-docker-compose_robotframework_1
Checking if hub is ready - selenium-hub
=====
Robo-Tests
=====
Robo-Tests.Demo
=====
Test Case 1 | FAIL |
WebDriverException: Message: 'geckodriver' executable needs to be in PATH.
-----
Robo-Tests.Demo | FAIL |
1 test, 0 passed, 1 failed
=====
Robo-Tests | FAIL |
1 test, 0 passed, 1 failed
=====
Output: /robo-vault/test_results/output.xml
Log: /robo-vault/test_results/log.html
Report: /robo-vault/test_results/report.html
```

8. Something is still missing. Let's check what has gone wrong!

13.6 References

- <https://docs.docker.com/get-started/overview/>
- <https://docs.docker.com/compose/>
- <https://docs.docker.com/compose/install/>
- https://hub.docker.com/r/selenium/hub/tags?page=1&ordering=last_updated
- <https://raw.githubusercontent.com/SeleniumHQ/docker-selenium/trunk/docker-compose-v3.yml>
- <https://robotframework.org/SeleniumLibrary/SeleniumLibrary.html>

14. Dockerizing Selenium Test Execution Environment

14.1 The Problem

So far, we have seen that our **robot test cases are failing** even though we are trying to run them in a Dockerized environment.

```
secqation@mirage:~/Desktop/NullconTraining2021/4-example-docker/2-docker-compose$ docker logs 2-docker-compose_robotframework_1
Checking if hub is ready - selenium-hub
=====
Robo-Tests
=====
Robo-Tests.Demo
=====
Test Case 1 | FAIL |
WebDriverException: Message: 'geckodriver' executable needs to be in PATH.
-----
Robo-Tests.Demo | FAIL |
1 test, 0 passed, 1 failed
=====
Robo-Tests | FAIL |
1 test, 0 passed, 1 failed
=====
Output: /robo-vault/test_results/output.xml
Log:   /robo-vault/test_results/log.html
Report: /robo-vault/test_results/report.html
```

On checking the [documentation for SeleniumLibrary](#), we find that all **drivers must be explicitly installed**.

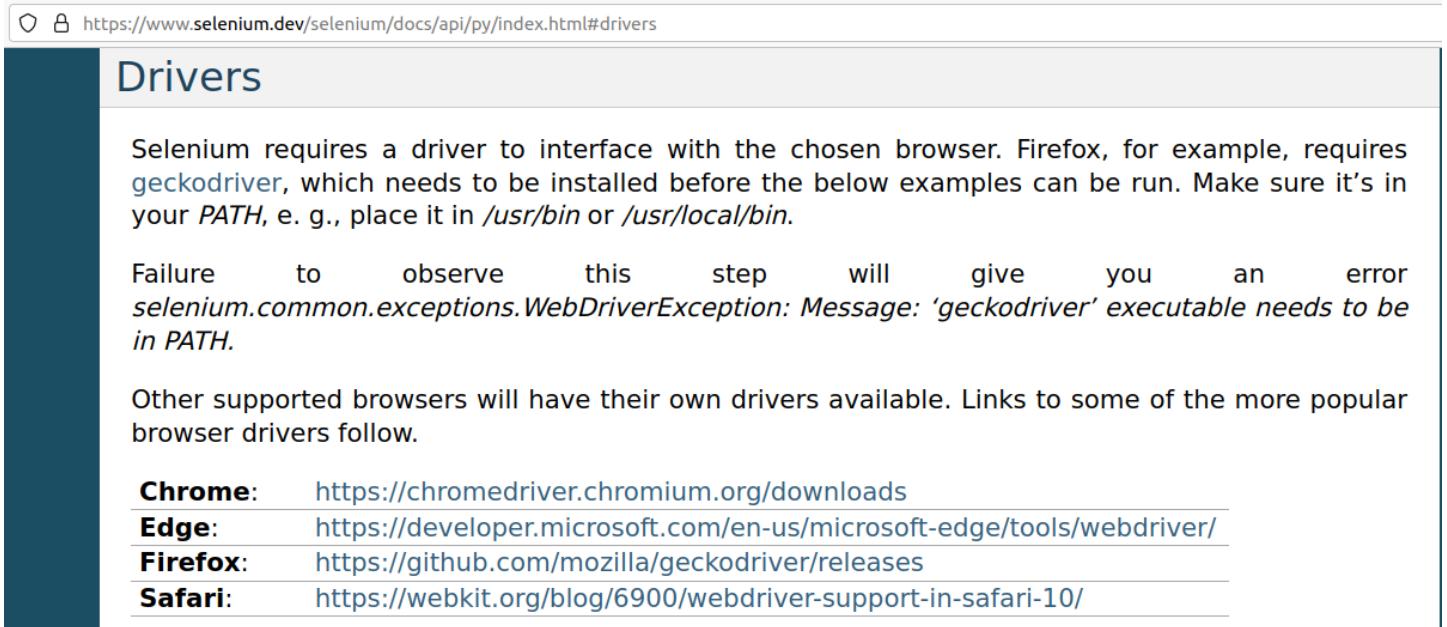
Browser drivers

After installing the library, you still need to install browser and operating system specific browser drivers for all those browsers you want to use in tests. These are the exact same drivers you need to use with Selenium also when not using SeleniumLibrary. More information about drivers can be found from [Selenium documentation](#).

The general approach to install a browser driver is downloading a right driver, such as chromedriver for Chrome, and placing it into a directory that is in [PATH](#). Drivers for different browsers can be found via Selenium documentation or by using your favorite search engine with a search term like selenium chrome browser driver. New browser driver versions are released to support features in new browsers, fix bug, or otherwise, and you need to keep an eye on them to know when to update drivers you use.

Furthermore, on checking the [Selenium documentation](#), we find an explanation for the exact error that we saw earlier, i.e.,

`WebDriverException: Message: 'geckodriver' executable needs to be in PATH`



The screenshot shows a browser window with the URL <https://www.selenium.dev/selenium/docs/api/py/index.html#drivers>. The page title is "Drivers". The content discusses the requirement for a driver to interface with the chosen browser, specifically mentioning geckodriver for Firefox. It also notes that failure to observe this step will result in an error message: "selenium.common.exceptions.WebDriverException: Message: 'geckodriver' executable needs to be in PATH." Other supported browsers like Chrome, Edge, Firefox, and Safari are mentioned with their respective driver download links.

Browser drivers can be installed using **webdrivermanager**:

```
$ pip install webdrivermanager $ webdrivermanager firefox chrome --linkpath /usr/local/bin
```

Under normal circumstances, it would have been required for us to [manually install drivers](#) for any browser that our robot test cases are configured to use.

But, in current situation, as we are consuming the Selenium and Firefox **Docker services**, should we not have everything **preconfigured**?

15. Solution: Headless Browser

In our test case, we need to modify the `Open Browser ${BASE_URL}` test step to open a headless browser, instead.

Check [SeleniumBrowser documentation](#) for various **arguments** that can be passed to **Open Browser** keyword.

Open Browser

Arguments

```

url          = None      <str>
browser     = firefox    <str>
alias       = None      <str>
remote_url  = False     <bool> or <str>
desired_capabilities = None    <dict> or <None> or <str>
ff_profile_dir = None    <FirefoxProfile> or <str> or <None>
options     = None      <Any>
service_log_path = None    <str>
executable_path = None    <str>

```

Documentation

Opens a new browser instance to the optional url.

The browser argument specifies which browser to use. The supported browsers are listed in the table below. The browser names are case-insensitive and some browsers have multiple supported names.

Browser	Name(s)
Firefox	firefox, ff
Google Chrome	googlechrome, chrome, gc
Headless Firefox	headlessfirefox
Headless Chrome	headlesschrome
Internet Explorer	internetexplorer, ie
Edge	edge

We can configure a **headless Firefox browser** as shown below:

```
... ${firefox_options}= Evaluate sys.modules['selenium.webdriver'].FirefoxOptions() sys, selenium.webdriver Call Method ${firefox_options} add_argument headless
```

After modification, our files should have following contents:

1. demo-test-suite/demo.robot

```
*** Settings *** Library SeleniumLibrary *** Variables *** ${BASE_URL} = https://www.winja.site ${SELENIUM_HUB} = http://%{HUB_HOSTNAME}:4444/wd/hub
```

2. Dockerfile

```
FROM python:3.9.5-alpine3.13 LABEL maintainer="Riddhi Shree" ENV PYTHONUNBUFFERED=1 RUN pip install --upgrade pip RUN pip install robotframework RUN p
```

3. docker-compose.yml

```
version: "3.9" services: firefox_1: image: selenium/node-firefox:4.0.0-rc-1-prerelease-20210804 shm_size: 2gb container_name: firefox_1 depends_on: -
```

4. healthcheck.sh

```
#!/usr/bin/env bash echo "Checking if hub is ready - $HUB_HOSTNAME" while [[ "$(curl -s http://$HUB_HOSTNAME:4444/wd/hub/status | jq .value.ready)" !=
```

15.1 Test Execution

- Run following commands to execute robot test cases in a headless Firefox browser, inside a Docker network:

```
$ docker-compose build $ docker-compose up
```

```

secqation@mirage: ~/Desktop/NullconTraining2021/4-example-docker/2-docker-compose$ docker-compose up
selenium-hub      | 04:57:41.494 INFO [LocalDistributor.newSession] - Session created by the distributor. Id: d531d946-5e39-4c9a-bb7d-804309f46ffd, Caps: Capabilities {acceptInsecureCerts: true, browserName: firefox, browserVersion: 90.0.2, moz:accessibilityChecks: false, moz:buildID: 20210721174149, moz:geckodriverVersion: 0.29.1, moz:headless: false, moz:processID: 156, moz:profile: /tmp/rust_mozprofileNFBNAj, moz:shutdownTimeout: 60000, moz:useNonSpecCompliantPointerOrigin: false, moz:webdriverClick: true, pageLoadStrategy: normal, platformName: linux, platformVersion: 5.11.0-27-generic, proxy: {}, se:vnc: ws://192.168.48.3:5555/session..., se:vncEnabled: true, se:vncLocalAddress: ws://localhost:7900/webssockify, setWindowRect: true, strictFileInteractivity: false, timeouts: {implicit: 0, pageLoad: 300000, script: 30000}, unhandledPromptBehavior: dismiss and notify}
firefox_2          | 1629608265908     Marionette      INFO    Stopped listening on port 35073
selenium-hub      | 04:57:46.891 INFO [LocalSessionMap.lambda$new$0] - Deleted session from local session map, Id: d531d946-5e39-4c9a-bb7d-804309f46ffd
robotframework      | Selenium In Docker Setup Should Work Correctly | PASS |
robotframework      | -----
robotframework      | Robo-Tests.Demo | PASS |
robotframework      | 1 test, 1 passed, 0 failed
robotframework      | -----
robotframework      | Robo-Tests | PASS |
robotframework      | 1 test, 1 passed, 0 failed
robotframework      | -----
robotframework      | Output: /robo-vault/test_results/output.xml
robotframework      | Log: /robo-vault/test_results/log.html
robotframework      | Report: /robo-vault/test_results/report.html
robotframework exited with code 0

```

- A new folder called **report** must have been created in the working directory. This folder contains the test report that was auto-generated by the robot framework.

```

secqation@mirage:~/Desktop/NullconTraining2021/4-example-docker/2-docker-compose$ tree report/
report/
├── log.html
├── output.xml
└── report.html
└── selenium-screenshot-1.png
└── selenium-screenshot-2.png

0 directories, 5 files

```

- Open **log.html** file in a browser to view the test logs

→ C file:///home/secqation/Desktop/NullconTraining2021/4-example-docker/2-docker-compose/report/log.html

Start / End / Elapsed: 20210822 10:27:29.580 / 20210822 10:27:46.923 / 00:00:17.337
Status: 1 test total, 1 passed, 0 failed, 0 skipped

- **TEST** Selenium In Docker Setup Should Work Correctly

Full Name: Robo-Tests.Demo.Selenium In Docker Setup Should Work Correctly
Start / End / Elapsed: 20210822 10:27:29.875 / 20210822 10:27:46.881 / 00:00:17.006
Status: **PASS**

+ **KEYWORD** Begin Web Assessment
+ **KEYWORD** SeleniumLibrary.Go To \${BASE_URL}
- **KEYWORD** SeleniumLibrary.Capture Page Screenshot

Documentation: Takes a screenshot of the current page and embeds it into a log file.
Start / End / Elapsed: 20210822 10:27:44.722 / 20210822 10:27:45.870 / 00:00:01.148
10:27:45.869 **INFO**

Home Past Events Live on Twitter! Upcoming Tags

Winja
Embracing Diversity.

About Us
Winja is a virtual community, run by volunteers. We aim to embrace diversity. How?! By experimenting with ideas. We aim to create a thriving space that allows people to come together, think together and create awareness around latest technological trends. We aim to build capabilities in the field of Information Security.

Various events that we have conducted so far include capture-the-flag (CTF) competitions, blog-writing workshop, and online technical talks. Past Winja events can be [viewed here](#). To start a conversation, join us on our [Discord server](#).

COMING NEXT...

Nullcon Presents **WINJA TALKS**



+ **KEYWORD** SeleniumLibrary.Close All Browsers

15.2 References

- <https://robotframework.org/SeleniumLibrary/>
- <https://www.selenium.dev/selenium/docs/api/py/index.html#drivers>
- <https://github.com/rasjani/webdrivermanager>
- <https://robotframework.org/SeleniumLibrary/SeleniumLibrary.html>

16. CI/CD Build Pipeline

16.1 Git

1. Create a new Git repository in github.com

Quick setup — if you've done this kind of thing before

or [HTTPS](https://github.com/riddhi-shree/secQAtion-jenkins-pipeline.git) [SSH](https://github.com/riddhi-shree/secQAtion-jenkins-pipeline.git)

Get started by creating a new file or [uploading an existing file](#). We recommend every repository include a [README](#), [LICENSE](#), and [.gitignore](#).

...or create a new repository on the command line

```
echo "# secQAtion-jenkins-pipeline" >> README.md
git init
git add README.md
git commit -m "first commit"
git branch -M main
git remote add origin git@github.com:riddhi-shree/secQAtion-jenkins-pipeline.git
git push -u origin main
```

...or push an existing repository from the command line

```
git remote add origin git@github.com:riddhi-shree/secQAtion-jenkins-pipeline.git
git branch -M main
git push -u origin main
```

2. Create a new directory locally, and run the setup commands

```
$ git init $ git remote add origin git@github.com:riddhi-shree/secQAtion-jenkins-pipeline.git
```

16.2 Personal Access Token

1. Login to github.com
2. Go to **Settings > Developer settings > Personal access tokens**
3. Click on **Generate new token** button

[Settings / Developer settings](#)

New personal access token

Personal access tokens function like ordinary OAuth access tokens. They can be used instead of a password for Git over HTTPS, or can be used to [authenticate to the API over Basic Authentication](#).

Note

secqation-jenkins

What's this token for?

Expiration *

30 days The token will expire on Sat, Sep 25 2021

Select scopes

Scopes define the access for personal tokens. [Read more about OAuth scopes](#).

<input checked="" type="checkbox"/> repo	Full control of private repositories
<input checked="" type="checkbox"/> repo:status	Access commit status
<input checked="" type="checkbox"/> repo_deployment	Access deployment status
<input checked="" type="checkbox"/> public_repo	Access public repositories
<input checked="" type="checkbox"/> repo:invite	Access repository invitations
<input checked="" type="checkbox"/> security_events	Read and write security events
<input type="checkbox"/> workflow	Update GitHub Action workflows
<input type="checkbox"/> write:packages	Upload packages to GitHub Package Registry
<small>Download packages from GitHub Package Registry</small>	

4. Enter a descriptive text in **Note** input field, to uniquely identify the token
5. Select **repo** checkbox
6. Click on **Generate token** button
7. Copy the generated personal access token value, and use this secret to access GitHub repository from Jenkins

16.3 Jenkinsfile

1. Create a file called **Jenkinsfile**
2. Add following contents

```
pipeline { agent any stages { stage('Build') { steps { echo 'Building..' } } stage('Test') { steps { echo 'Testing..' } } stage('Deploy') { steps { echo 'Deploying..!' } } }
```

16.4 Jenkins Server

1. In a separate folder, create a **Dockerfile** with following contents

```
FROM jenkins/jenkins:lts USER root RUN mkdir -p /tmp/download && \ curl -L https://download.docker.com/linux/static/stable/x86_64/docker-18.03.1-ce.tg
```

2. Build Jenkins Docker image

```
docker build -t jenkins-docker .
```

3. Run following commands to successfully start the Jenkins server

```
$ mkdir -p /var/jenkins_home $ chown -R 1000:1000 /var/jenkins_home/ $ docker run -p 8080:8080 -p 50000:50000 -v /var/jenkins_home:/var/jenkins_home -
```

CONTAINER ID	IMAGE	COMMAND	CREATED	STATUS	PORTS	NAMES
a67458de0da3	jenkins-docker	"/sbin/tini -- /usr/..."	3 hours ago	Up 3 hours	0.0.0.0:8080->8080/tcp, :::8080->8080/tcp, 0.0.0.0:50000->50000/tcp, :::50000->50000/tcp	jenkins

4. Navigate to <http://localhost:8080/>

The screenshot shows the Jenkins dashboard at localhost:8080. The left sidebar has links for New Item, People, Build History, Manage Jenkins, My Views, Open Blue Ocean, Lockable Resources, and New View. Under Build Queue, it says "No builds in the queue." Under Build Executor Status, it lists 1 Idle and 2 Idle. The main content area says "Welcome to Jenkins!" and "Start building your software project" with a "Create a Job" button. It also has sections for "Set up a distributed build" with "Set up an agent" and "Configure a cloud" buttons, and a link to "Learn more about distributed builds". The top right shows the user "Riddhi Shree" and a log out button.

5. Click on **New Item** option in left navigation menu

6. Enter an item name, e.g., `SecQAtion`

7. Select **Pipeline** option, and click **OK**

The screenshot shows the Jenkins interface for creating a new item. At the top, there's a header bar with the Jenkins logo, a search bar, and user information. Below it, the main content area has a title 'Enter an item name' and a required input field containing 'SecQAtion Pipeline'. A 'Required field' message is displayed below the input field. The page lists several project types with their icons and descriptions:

- Freestyle project**: This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for something other than software build.
- Pipeline**: Orchestrates long-running activities that can span multiple build agents. Suitable for building pipelines (formerly known as workflows) and/or organizing complex activities that do not easily fit in free-style job type.
- Multi-configuration project**: Suitable for projects that need a large number of different configurations, such as testing on multiple environments, platform-specific builds, etc.
- Bitbucket Team/Project**: Scans a Bitbucket Cloud Team (or Bitbucket Server Project) for all repositories matching some defined markers.
- Folder**: Creates a container that stores nested items in it. Useful for grouping things together. Unlike view, which is just a filter, a folder is a separate namespace, so you can have multiple things of the same name as long as they are in different folders.

At the bottom of the list, there are two buttons: 'OK' and 'Cancel'.

8. Scroll down to **Advanced Project Options > Pipeline** 9. Select **Pipeline script from SCM** option in **Definition** dropdown list

Build Triggers

- GitHub hook trigger for GITScm polling
- Poll SCM
- Disable this project
- Quiet period
- Trigger builds remotely (e.g., from scripts)

Advanced Project Options

Pipeline

Definition

Pipeline script from SCM

SCM

None

Script Path

Jenkinsfile

Lightweight checkout

Pipeline Syntax

Buttons

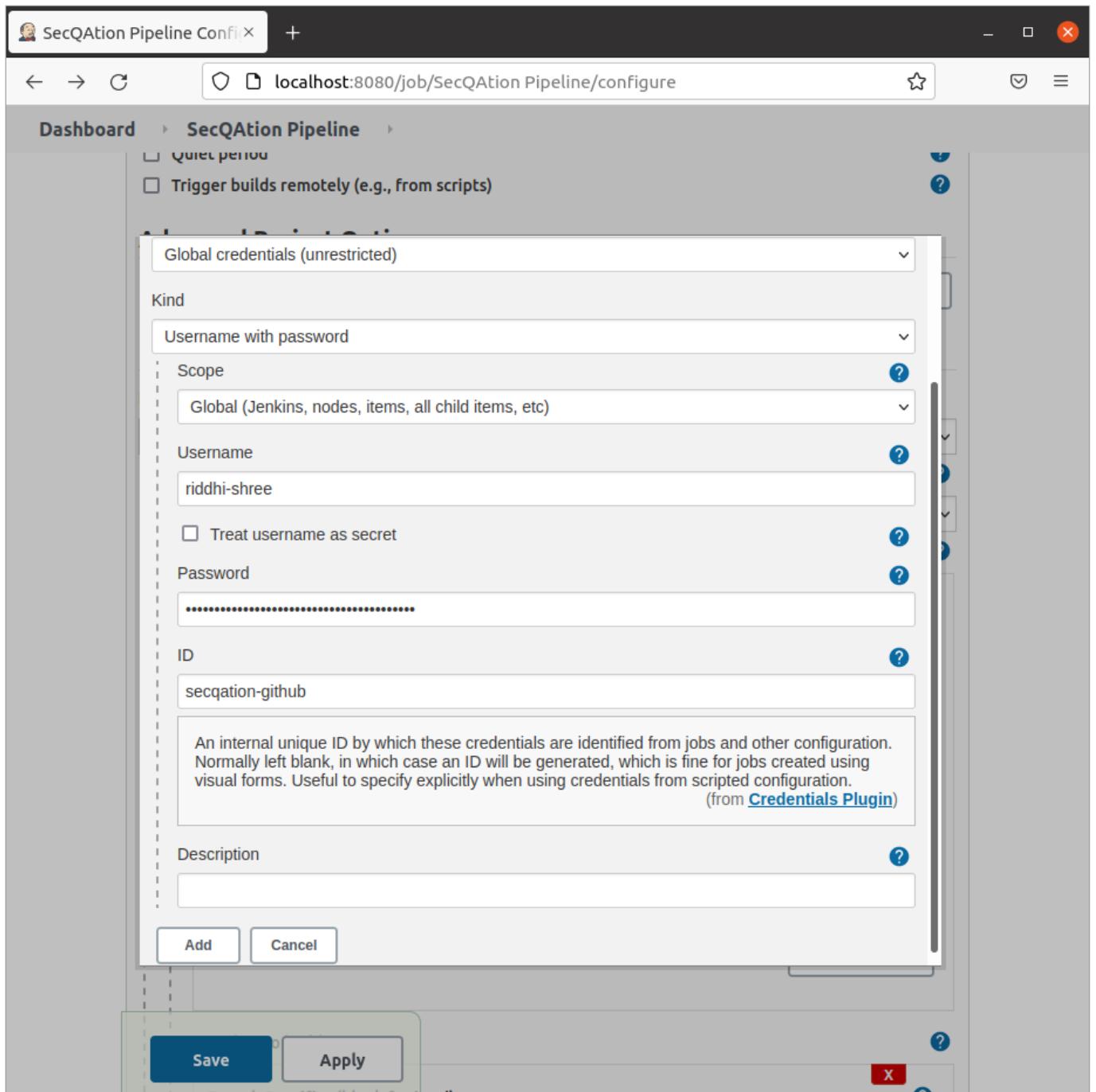
Save Apply

[REST API](#) Jenkins 2.303.1

10. From **SCM** dropdown, select **Git** option 11. Enter your Git repository URL, e.g.,
<https://github.com/riddhi-shree/secQAtion-jenkins-pipeline.git>

The screenshot shows the Jenkins configuration interface for a pipeline named "SecQAction Pipeline". Under the "Pipeline" section, the "Definition" is set to "Pipeline script from SCM". The "SCM" section is configured to "Git", and the "Repository URL" is set to "https://github.com/riddhi-shree/secQAction-jenkins-pipeline.git". A red error message indicates a failed connection: "Failed to connect to repository : Command "git ls-remote -h https://github.com/riddhi-shree/secQAction-jenkins-pipeline.git HEAD" returned status code 128: stdout: stderr: remote: Invalid username or password. fatal: Authentication failed for 'https://github.com/riddhi-shree/secQAction-jenkins-pipeline.git/'". In the "Credentials" section, there is a dropdown menu set to "- none -" and a button labeled "Add". A tooltip for "Jenkins" is visible over the "Add" button. Below the "Add" button is a box labeled "Jenkins Credentials Provider". At the bottom of the screen, there are "Save" and "Apply" buttons.

12. Select **Credentials > Add > Jenkins**
13. Enter GitHub username
14. Enter your GitHub **Personal Access Token** in **Password** field
15. Enter a unique name in **ID** field, e.g. `secqation-github`



8. Validate **Script Path** that specifies path to the **Jenkinsfile** in GitHub repository
9. Uncheck **Lightweight checkout** checkbox

Script Path

Lightweight checkout

If selected, try to obtain the Pipeline script contents directly from the SCM without performing a full checkout. The advantage of this mode is its efficiency; however, you will not get any changelogs or polling based on the SCM. (If you use `checkout scm` during the build, this will populate the changelog and initialize polling.) Also build parameters will not be substituted into SCM configuration in this mode. Only selected SCM plugins support this mode.

(from [Pipeline: Groovy](#))

Pipeline Syntax

Save **Apply**

10. Click on **Apply** button

11. Click on **Save** button

localhost:8080/job/SecQAtion Pipeline/

Jenkins

Dashboard > SecQAtion Pipeline >

- Back to Dashboard
- Status
- Changes
- Build Now
- Configure
- Delete Pipeline
- Full Stage View
- Open Blue Ocean
- Rename
- Pipeline Syntax

Build History trend ^

find X

Atom feed for all Atom feed for failures

Pipeline SecQAtion Pipeline

Recent Changes

No data available. This Pipeline has not yet run.

Stage View

No data available. This Pipeline has not yet run.

Permalinks

16.5 ngrok

1. Download [ngrok](#)

2. Unzip ngrok

```
unzip /path/to/ngrok.zip
```

3. Sign up and login to [ngrok dashboard](#)

<https://dashboard.ngrok.com/get-started/setup>

Download ngrok

ngrok is easy to install. Download a single binary with zero run-time dependencies.

[Download for Linux](#)

1. Unzip to install

On Linux or Mac OS X you can unzip ngrok from a terminal with the following command. On Windows, just double click ngrok.zip to extract it.

```
$ unzip /path/to/ngrok.zip
```

2. Connect your account

Running this command will add your auth token to the default `ngrok.yml` configuration file. This will grant you access to more features and longer session times. Running tunnels will be listed on the [status page](#) of the dashboard.

```
$ ./ngrok authtoken
```

4. Copy your [authtoken](#)

5. Connect your account

```
./ngrok authtoken <your_auth_token>
```

6. Start a HTTP tunnel forwarding to your local port 8080 (where Jenkins server is running)

```
./ngrok http 8080
```

```
ngrok by @inconshreveable

Session Status           online
Account                  Riddhi Shree (Plan: Free)
Version                 2.3.40
Region                  United States (us)
Web Interface           http://127.0.0.1:4040
Forwarding              http://ff2d-49-37-75-29.ngrok.io -> http://localhost:8080
Forwarding              https://ff2d-49-37-75-29.ngrok.io -> http://localhost:8080

Connections             ttl     opn     rt1     rt5     p50     p90
                        12      0       0.00    0.00    7.38   10.58

HTTP Requests
-----
POST /github-webhook/                200 OK
POST /github-webhook/                200 OK
GET  /static/89c5c2ca/images/rage.png 200 OK
GET  /static/89c5c2ca/favicon.ico   200 OK
GET  /static/89c5c2ca/images/material-icons/svg-sprite-action-symbol.svg 200 OK
GET  /static/89c5c2ca/images/jenkins-header-logo-v2.svg                 200 OK
GET  /static/89c5c2ca/images/title.png                                     200 OK
GET  /adjuncts/89c5c2ca/lib/layout/breadcrumbs.js                         200 OK
```

7. Take a note of the **forwarding URL**, e.g., <http://ff2d-49-37-75-29.ngrok.io>

The screenshot shows the ngrok dashboard at <https://dashboard.ngrok.com/endpoints/status>. The left sidebar has a dark theme with the 'Endpoints' section selected. The main area is titled 'Online Tunnel Endpoints' and lists two entries:

Region	URL
US	http://ff2d-49-37-75-29.ngrok.io
US	https://ff2d-49-37-75-29.ngrok.io

16.6 Webhook

1. Login to your [GitHub](#) account
2. Go to the project repository
3. Click on **Settings > Webhooks > Add webhook**
4. Enter payload URL, e.g. <http://ff2d-49-37-75-29.ngrok.io/github-webhook>

The screenshot shows the GitHub Settings page for a repository named 'secQAtion-jenkins-pipeline'. The 'Webhooks' section is active, indicated by a red border around the 'Webhooks' link in the sidebar. The main form is titled 'Webhooks / Add webhook'. It contains the following fields:

- Payload URL ***: `http://ff2d-49-37-75-29.ngrok.io/github-webhook/`
- Content type**: `application/x-www-form-urlencoded`
- Secret**: A randomly generated string: `somerandomstringofcharactersreallyrealllllyooooooooooooong`
- Which events would you like to trigger this webhook?**
 - Just the push event.
 - Send me everything.
 - Let me select individual events.
- Active**: A checked checkbox with the note: "We will deliver event details when this hook is triggered."
- Add webhook** button

Note: Appending the URL path `/github-webhook` to ngrok forwarding URL (e.g., <http://ff2d-49-37-75-29.ngrok.io>) will give us the **Payload URL** for GitHub webhook 5. Optionally, enter a **Secret** 6. Select **Just the push event** option 7. Click on **Add webhook** button

Webhooks

[Add webhook](#)

Webhooks allow external services to be notified when certain events happen. When the specified events happen, we'll send a POST request to each of the URLs you provide. Learn more in our [Webhooks Guide](#).

[http://ff2d-49-37-75-29.ngrok.io/...](http://ff2d-49-37-75-29.ngrok.io/) (push)

[Edit](#) [Delete](#)

16.7 Configure Build Trigger

1. Login to Jenkins server
2. Select your pipeline from the Dashboard

The screenshot shows the Jenkins dashboard at localhost:8080. On the left sidebar, there are links for New Item, People, Build History, Project Relationship, Check File Fingerprint, and Manage Jenkins. The main area displays a list of jobs with the following details:

Icon	Name	Icon
	SecQAtion Pipeline	

Below the table, it says "Icon: S M L".

3. Click on **Configure > Build Triggers**

4. Select the option **GitHub hook trigger for GITScm polling**

The screenshot shows the configuration page for the "SecQAtion Pipeline" job at localhost:8080/job/SecQAtion%20Pipeline/configure. The "Build Triggers" tab is selected. The configuration options are:

- Build after other projects are built
- Build periodically
- GitHub hook trigger for GITScm polling

A detailed description of the "GitHub hook trigger for GITScm polling" option is provided:

When Jenkins receives a GitHub push hook, GitHub Plugin checks to see whether the hook came from a GitHub repository which matches the Git repository defined in SCM/Git section of this job. If they match and this option is enabled, GitHub Plugin triggers a one-time polling on GITScm. When GITScm polls GitHub, it finds that there is a change and initiates a build. The last sentence describes the behavior of Git plugin, thus the polling and initiating the build is not a part of GitHub plugin.

(from [GitHub plugin](#))

Below this, there are additional options:

- Poll SCM
- Disable this project
- Quiet period
- Trigger builds remotely (e.g., from scripts)

5. Click on **Save** button

16.8 Git Commit

1. Open a terminal and navigate to your local Git repository

```
cd /home/secqation/Desktop/NullconTraining2021/6-example-jenkins
```

2. Modify the **echo** command in **Jenkinsfile** (just for testing purpose)

```
pipeline { agent any stages { stage('Build') { steps { echo 'Building..' echo 'Added webhook. Testing connection!!!' } } stage('Test') { steps { echo
```

3. Push the changes to remote GitHub repository

```
$ git add . $ git status $ git commit -m "Your comment" $ git push
```

```
Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
  (use "git restore <file>..." to discard changes in working directory)
    modified:   Jenkinsfile

no changes added to commit (use "git add" and/or "git commit -a")
secqation@mirage:~/Desktop/NullconTraining2021/6-example-jenkins$ tree
.
└── Jenkinsfile
    └── README.md

0 directories, 2 files
secqation@mirage:~/Desktop/NullconTraining2021/6-example-jenkins$ git add .
secqation@mirage:~/Desktop/NullconTraining2021/6-example-jenkins$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes to be committed:
  (use "git restore --staged <file>..." to unstage)
    modified:   Jenkinsfile

secqation@mirage:~/Desktop/NullconTraining2021/6-example-jenkins$ git commit -m "Modified the echo command"
[master a4c6331] Modified the echo command
 1 file changed, 1 insertion(+), 1 deletion(-)
secqation@mirage:~/Desktop/NullconTraining2021/6-example-jenkins$ git push
```

4. Notice that a new build is triggered automatically

Dashboard > SecQAtion Pipeline >

[Back to Dashboard](#)

Status

- [Changes](#)
- [Build Now](#)
- [Configure](#)
- [Delete Pipeline](#)
- [Full Stage View](#)
- [Open Blue Ocean](#)
- [Rename](#)
- [Pipeline Syntax](#)
- [GitHub Hook Log](#)

Pipeline SecQAtion Pipeline

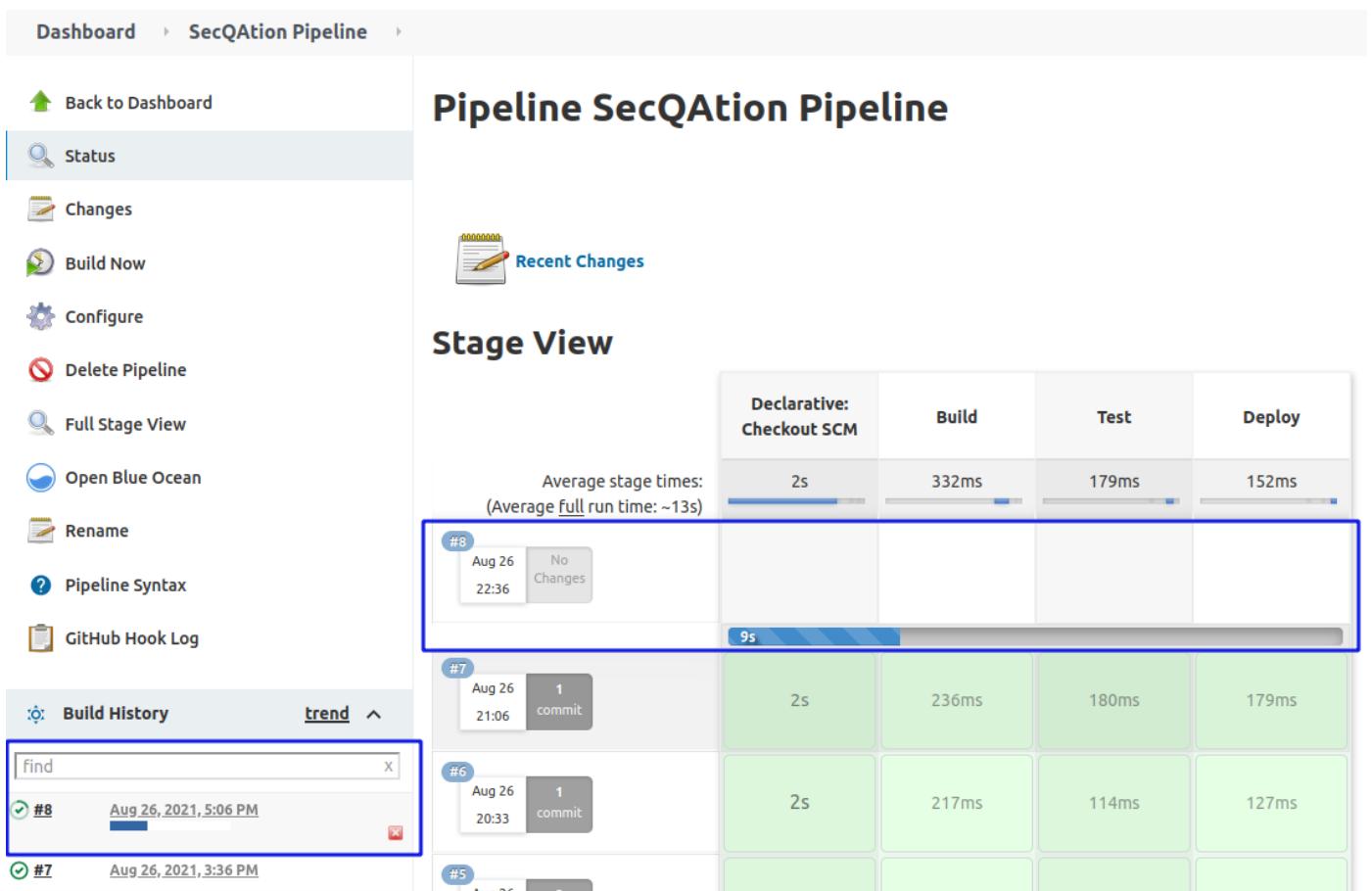
 [Recent Changes](#)

Stage View

Declarative: Checkout SCM	Build	Test	Deploy	
Average stage times: (Average full run time: ~13s)	2s	332ms	179ms	152ms
#8 Aug 26 22:36 No Changes				
#7 Aug 26 21:06 1 commit	2s	236ms	180ms	179ms
#6 Aug 26 20:33 1 commit	2s	217ms	114ms	127ms
#5 Aug 26 20:26 2 commits				

Build History [trend](#) ^

find #8 Aug 26, 2021, 5:06 PM #7 Aug 26, 2021, 3:36 PM



5. Click on the **build number**

6. Click on **Console Output**

localhost:8080/job/SecQAtion Pipeline/8/

Jenkins

Dashboard > SecQAtion Pipeline > #8

Back to Project Status Changes Console Output Edit Build Information Delete build '#8' Polling Log Git Build Data Open Blue Ocean Restart from Stage Replay Pipeline Steps Workspaces Previous Build

Build #8 (Aug 26, 2021, 5:06:20 PM)

Changes

1. Modified the echo command ([details](#) / [githubweb](#))

Started by GitHub push by riddhi-shree

Revision: a4c633181a7b37444e7a3455664ee4c61bde3edb
Repository: <https://github.com/riddhi-shree/secQAtion-jenkins-pipeline.git>

- refs/remotes/origin/master

The screenshot shows the Jenkins interface for a pipeline named 'SecQAtion Pipeline'. The build number is #8. The left sidebar contains various Jenkins management links like 'Back to Project', 'Status', 'Changes', 'Console Output' (which is selected), 'View as plain text', 'Edit Build Information', 'Delete build #8', 'Polling Log', 'Git Build Data', 'Open Blue Ocean', 'Restart from Stage', 'Replay', 'Pipeline Steps', 'Workspaces', and 'Previous Build'. The main right panel is titled 'Console Output' with a green checkmark icon. It displays the Jenkins pipeline logs. A blue box highlights the commit message: 'Commit message: "Modified the echo command"'. Another blue box highlights the output: 'Added webhook. Testing connection!!!'. The logs also show standard Git operations like cloning the repository and fetching upstream changes.

```

Started by GitHub push by riddhi-shree
Obtained Jenkinsfile from git https://github.com/riddhi-shree/secQAtion-jenkins-pipeline.git
Running in Durability level: MAX_SURVIVABILITY
[Pipeline] Start of Pipeline
[Pipeline] node
Running on Jenkins in /var/jenkins_home/workspace/SecQAtion Pipeline
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Declarative: Checkout SCM)
[Pipeline] checkout
Selected Git installation does not exist. Using Default
The recommended git tool is: NONE
using credential secqation-github
> git rev-parse --resolve-git-dir /var/jenkins_home/workspace/SecQAtion Pipeline/.git # timeout=10
Fetching changes from the remote Git origin
> git config remote.origin.url https://github.com/riddhi-shree/secQAtion-jenkins-pipeline.git # timeout=10
Fetching upstream changes from https://github.com/riddhi-shree/secQAtion-jenkins-pipeline.git
> git --version # timeout=10
> git --version # 'git version 2.30.2'
using GIT_ASKPASS to set credentials
> git fetch --tags --force --progress -- https://github.com/riddhi-shree/secQAtion-jenkins-pipeline.git +refs/
> git rev-parse refs/remotes/origin/master^{commit} # timeout=10
Checking out Revision a4c633181a7b37444e7a3455664ee4c61bde3edb (refs/remotes/origin/master)
> git config core.sparsecheckout # timeout=10
> git checkout -f a4c633181a7b37444e7a3455664ee4c61bde3edb # timeout=10
Commit message: "Modified the echo command"
> git rev-list --no-walk a5b30915505e060da30605040e9bde7b0f91813e2 # timeout=10
[Pipeline] }
[Pipeline] // stage
[Pipeline] withEnv
[Pipeline] {
[Pipeline] stage
[Pipeline] { (Build)
[Pipeline] echo
Building..
[Pipeline] echo
Added webhook. Testing connection!!!
[Pipeline] }
[Pipeline] // stage
[Pipeline] stage
[Pipeline] { (Test)
[Pipeline] echo
Testing..

```

16.9 Triggering Robot Test Case Execution

We can now successfully trigger a build in Jenkins for every `git push` event in GitHub. So, what's pending then? We are yet to execute the automated test cases and fetch test results.

The final **folder structure** of our Git repository might look similar to following:

```
secqation@mirage:~/Desktop/NullconTraining2021/6-example-jenkins$ tree
.
└── demo-test-suite
    └── demo.robot
    ├── docker-compose.yml
    ├── Dockerfile
    ├── healthcheck.sh
    └── init.sh
    └── Jenkinsfile
    └── README.md
```

Let's confirm the contents of each file at this point.

16.9.1 demo.robot

```
*** Settings ***
Library SeleniumLibrary
Variables
${BASE_URL} = http://
```

16.9.2 docker-compose.yml

```
version: "3.9"
services:
  firefox_1:
    image: selenium/node-firefox:4.0.0-rc-1-p
```

16.9.3 Dockerfile

```
FROM python:3.9.5-alpine3.13
LABEL maintainer="Riddhi Shree"
ENV PYTHONUNBUFFERED=1
```

16.9.4 healthcheck.sh

```
#!/usr/bin/env bash
echo "Checking if hub is ready - $HUB_HOSTNAME" while [[
```

16.9.5 init.sh

```
docker-compose down
docker rm `docker ps -aq`
```

16.9.6 Jenkinsfile

```
pipeline {
    agent any
    stages {
        stage('Test Execution') {
            steps {
                sh 'rm -r ./reports'
            }
        }
    }
}
```

16.9.7 View Build Output

Once a new commit is pushed to remote **GitHub** repository, a build is triggered in Jenkins server. Steps mentioned in **Jenkinsfile** get executed.

The Jenkins interface shows the Pipeline SecQATION Pipeline. The Stage View displays four stages: Declarative: Checkout SCM, Test Execution, Publish Test Output, and another unnamed stage. The Build History shows five builds (#45 to #41) with their respective run times and outcomes. Build #45 is green (2s), #44 is red (1min 4s, failed), #43 is green (1min 3s), #42 is green (1min 1s), and #41 is green (1s). The pipeline URL is 127.0.0.1:8080/job/SecQATION%20Pipeline/45/console.

```
Successfully built 0d532a6a1427
Successfully tagged secqationpipeline:robotframework:latest
Creating network "secqationpipeline_default" with the default driver
Creating selenium-hub ...
Creating selenium-hub ... done
Creating selenium-hub ... done
Creating firefox_1 ...
Creating firefox_1 ... done
Creating firefox_2 ...
Creating firefox_2 ... done
Creating firefox_1 ... done
Creating robotframework ...
Creating robotframework ... done
Test execution in-progress...
Checking if hub is ready - selenium-hub
=====
Robo-Tests
=====
Robo-Tests-Demo
=====
Selenium In Docker Setup Should Work Correctly | PASS |
-----
Robo-Tests-Demo | PASS |
1 test, 1 passed, 0 failed
-----
Robo-Tests | PASS |
1 test, 1 passed, 0 failed
-----
Output: /robo-vault/test_results/output.xml
Log: /robo-vault/test_results/log.html
Report: /robo-vault/test_results/report.html
Stopping firefox_1 ...
Stopping firefox_1 ... done
Stopping firefox_2 ...
Stopping firefox_2 ... done
Stopping selenium-hub ...
Stopping selenium-hub ... done
Removing robotframework ...
Removing robotframework ... done
Removing firefox_1 ...
Removing firefox_1 ... done
Removing selenium-hub ...
Removing selenium-hub ... done
Removing robotframework ... done
```

Note: To run this example yourself, look

at the code available at location

`/home/secqation/Desktop/NullconTraining2021/6-example-jen`

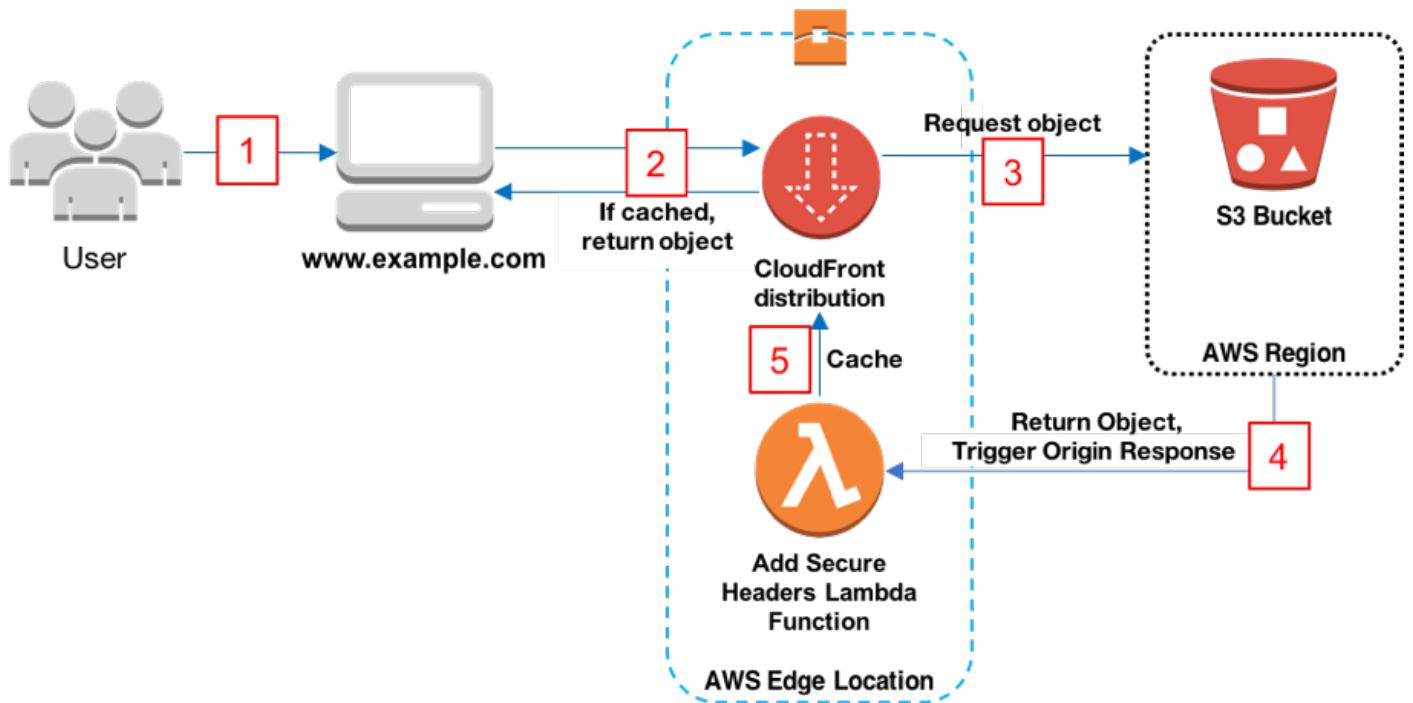
16.10 What's next?

How do we view the detailed test results...

16.11 References

- `https://blog.knoldus.com/opsinit-adding-a-github-webhook-in-jenkins-pipeline/`
- `https://github.com/`
- `https://ngrok.com/downLoad`
- `https://dashboard.ngrok.com/login`
- `https://dashboard.ngrok.com/get-started/setup`

17. Serving test report via S3 and CloudFront



17.1 Steps

1. Purchase a **domain name** (e.g., `secqation.xyz`)

The screenshot shows the Namecheap dashboard. At the top, it displays the URL "ap.www.namecheap.com". On the left sidebar, there are several menu items: "Dashboard" (selected), "Expiring / Expired", "Domain List", "Hosting List", "Private Email", and "SSL Certificates". The main area of the dashboard shows a greeting "Hello Riddhi Shree" and a note that the user last logged in on Sep 5, 2021, at 03:07 PM (EST). Below this, there is a search bar with the placeholder "Search for your next domain". A section titled "Recently Active in Your Account" lists "All" products. One item in this list is highlighted with a blue border: "secqation.xyz". To the right of this item is a small icon of a house with a green checkmark.

2. Create a private S3 bucket, having same name as the purchased domain name (i.e., [secqation.xyz](#))

Permissions overview

Access
Bucket and objects not public

Block public access (bucket settings)

Public access is granted to buckets and objects through access control lists (ACLs), bucket policies, access point policies, or all. In order to ensure that public access to all your S3 buckets and objects is blocked, turn on Block all public access. These settings apply only to this bucket and its access points. AWS recommends that you turn on Block all public access, but before applying any of these settings, ensure that your applications will work correctly without public access. If you require some level of public access to your buckets or objects within, you can customize the individual settings below to suit your specific storage use cases. [Learn more](#)

Edit

Block all public access

- On
 - Block public access to buckets and objects granted through *new* access control lists (ACLs)
 - On
 - Block public access to buckets and objects granted through *any* access control lists (ACLs)
 - On
 - Block public access to buckets and objects granted through *new* public bucket or access point policies
 - On
 - Block public and cross-account access to buckets and objects through *any* public bucket or access point policies
 - On

Feedback English (US) ▾

© 2008 - 2021, Amazon Internet Services Private Ltd. or its affiliates. All rights reserved.

[Privacy Policy](#) [Terms of Use](#)[Cookie preferences](#)

3. Create an IAM user having permission to upload files to S3 bucket

Policies > secqation-s3-access

Summary

Policy ARN arn:aws:iam:::policy/secqation-s3-access 

Description

Permissions Policy usage Tags Policy versions Access Advisor

Policy summary  JSON Edit policy

```
1 {  
2   "Version": "2012-10-17",  
3   "Statement": [  
4     {  
5       "Sid": "VisualEditor0",  
6       "Effect": "Allow",  
7       "Action": "s3>ListAllMyBuckets",  
8       "Resource": "arn:aws:s3:::"  
9     },  
10    {  
11      "Sid": "VisualEditor1",  
12      "Effect": "Allow",  
13      "Action": "s3:*",  
14      "Resource": [  
15        "arn:aws:s3:::secqation.xyz/*",  
16        "arn:aws:s3:::secqation.xyz"  
17      ]  
18    }  
19  ]  
20 }
```

17.2 CloudFront

1. Login to AWS account
2. Got to [CloudFront console](#)
3. Click on **Create Distribution** button

4. Fill the **create distribution form**:

1. Origin Domain: Choose your **S3 bucket** from dropdown list
2. S3 bucket access: Select **Yes use OAI (bucket can restrict access to only CloudFront)**
3. Click on **Create new OAI > Create**
4. Bucket policy: Select **Yes, update the bucket policy**

Origin domain
Choose an AWS origin, or enter your origin's domain name.

🔍

×

Origin path - optional Info
Enter a URL path to append to the origin domain name for origin requests.

Name
Enter a name for this origin.

S3 bucket access Info
Use a CloudFront origin access identity (OAI) to access the S3 bucket.

Don't use OAI (bucket must allow public access)
 Yes use OAI (bucket can restrict access to only CloudFront)

Origin access identity
Select an existing origin access identity (recommended) or create a new identity.

access-identity-secqation.xyz.s3.us-east-1.amazonaws.com

▾

[Create new OAI](#)

Bucket policy
Update the S3 bucket policy to allow read access to the OAI.

No, I will update the bucket policy
 Yes, update the bucket policy

5. Viewer protocol policy: Select **Redirect HTTP to HTTPS**

6. Allowed HTTP methods: Select **GET, HEAD**

5. Scroll down to "Settings" and select an option (e.g., **Use North America, Europe, Asia, Middle East, and Africa**)

6. Select **Alternate domain name (CNAME) >> Add item**, and enter target domain name (e.g., `secqation.xyz`)

7. Select **Custom SSL certificate >> Request certificate**

Request a certificate

- Step 1: Add domain names**
- Step 2: Select validation method
- Step 3: Add tags
- Step 4: Review
- Step 5: Validation

AWS Certificate Manager logs domain names from your certificates into public certificate transparency (CT) logs when renewing certificates. You can opt out of CT logging. [Learn more](#)

You can use AWS Certificate Manager certificates with other AWS Services.

Add domain names

Type the fully qualified domain name of the site you want to secure with an SSL/TLS certificate (for example, www.example.com). Use an asterisk (*) to request a wildcard certificate to protect several sites in the same domain. For example: *.example.com protects www.example.com, site.example.com and images.example.com.

Domain name*
*At least one domain name is required

secqation.xyz

*.secqation.xyz

Add another name to this certificate
You can add additional names to this certificate. For example, if you're requesting a certificate for "www.example.com", you might want to add the name "example.com" so that customers can reach your site by either name. [Learn more](#).

[Cancel](#) **Next**

8. Fill and submit the "Request a certificate" form
9. Return to "Create distribution" form
10. In the **Custom SSL certificate** dropdown field, select the certificate that was just created.

Alternate domain name (CNAME) - optional

Add the custom domain names that you use in URLs for the files served by this distribution.

secqation.xyz

Remove

Add item

i To add a list of alternative domain names, use the [bulk editor](#).

Custom SSL certificate - optional

Associate a certificate from AWS Certificate Manager. The certificate must be in the US East (N. Virginia) Region (us-east-1).

arn:aws:acm:us-east-1:948778895882:certificate/159b5f22-7a6e-4ca4-b067-(



Request certificate

11. Click on **Create Distribution** button
12. Wait until the CloudFront distribution creation status becomes **Enabled**

CloudFront > Distributions

Distributions (2) Info		C	Enable	Disable	Delete	Create distribution
		<input type="text"/> Search all distributions		< 1 >		
<input type="checkbox"/>	ID	Description	Domain Name	Alternate domain names	Origins	Status
<input type="checkbox"/>	EDKE5Y8BDGZGO	-	d39zsgs2...	secqation.xyz, www.secqation.xyz	secqation.xyz.s3.us-east-1.amazonaws.com	Enabled

17.3 Hosted Zone

- When a certificate is issued for a domain, user is required to add a CNAME record to their domain to prove the domain ownership.

Status

Domain	Validation status
secqation.xyz	Success

Add the following CNAME record to the DNS configuration for your domain. The procedure for adding CNAME records depends on your DNS service Provider. [Learn more](#).

Name	Type	Value
_116d57b485d9c88d03ca4a73c9855b04.secqation.xyz.	CNAME	_28cd3083ea506a132f6db668ba78bb05.lkwmzfhcjn.acm-validations.aws.

Note: Changing the DNS configuration allows ACM to issue certificates for this domain name for as long as the DNS record exists. You can revoke permission at any time by removing the record. [Learn more](#).

[Create record in Route 53](#) [Amazon Route 53 DNS Customers](#) ACM can update your DNS configuration for you. [Learn more](#).

Domain	Validation status
*.secqation.xyz	Success

- After the CNAME record is added successfully to the domain, it takes a while to update the DNS validation status in AWS. Wait until the "Validation status" shows as **Success**, and certificate status shows as **Issued**.

Request a certificate		Import a certificate	Actions ▾	Manage certificate events															
<input type="checkbox"/>	Name ▾	Domain name ▾	Additional names	Status ▾															
<input type="checkbox"/>	-	secqation.xyz	*.secqation.xyz	Issued															
<table border="1"> <thead> <tr> <th colspan="2">Status</th> </tr> </thead> <tbody> <tr> <td>Status</td> <td>Issued</td> </tr> <tr> <td>Detailed status</td> <td>The certificate was issued at 2021-09-05T12:02:38UTC</td> </tr> <tr> <th>Domain</th> <th>Validation status</th> </tr> <tr> <td>secqation.xyz</td> <td>Success</td> </tr> <tr> <td>*.secqation.xyz</td> <td>Success</td> </tr> </tbody> </table>								Status		Status	Issued	Detailed status	The certificate was issued at 2021-09-05T12:02:38UTC	Domain	Validation status	secqation.xyz	Success	*.secqation.xyz	Success
Status																			
Status	Issued																		
Detailed status	The certificate was issued at 2021-09-05T12:02:38UTC																		
Domain	Validation status																		
secqation.xyz	Success																		
*.secqation.xyz	Success																		
 Export DNS configuration to a file		You can export all of the CNAME records to a file																	

- Go to Route 53 dashboard

4. Click on **Hosted zones > Create hosted zone**
5. Enter **Domain name**, e.g., `secqation.xyz`

Create hosted zone Info

Hosted zone configuration

A hosted zone is a container that holds information about how you want to route traffic for a domain, such as example.com, and its subdomains.

Domain name Info
This is the name of the domain that you want to route traffic for.

Valid characters: a-z, 0-9, ! " # \$ % & ' () * + , - / ; < = > ? @ [\] ^ _ ` { | } . ~

Description - optional Info
This value lets you distinguish hosted zones that have the same name.

The hosted zone is used for...

The description can have up to 256 characters. 0/256

Type Info
The type indicates whether you want to route traffic on the internet or in an Amazon VPC.

Public hosted zone
A public hosted zone determines how traffic is routed on the internet.

Private hosted zone
A private hosted zone determines how traffic is routed within an Amazon VPC.

6. Click on **Create hosted zone** button
7. You will be displayed with a list of **AWS nameservers**.

Records (2) Info DNSSEC signing Hosted zone tags (0)

Records (2) Info
Automatic mode is the current search behavior optimized for best filter results. [To change modes go to settings.](#)

<input type="checkbox"/>	Record name	Type	Routing policy	Differences	Value/Route traffic to
<input type="checkbox"/>	secqation.xyz	NS	Simple	-	ns-140.awsdns-17.com. ns-1446.awsdns-52.org. ns-1796.awsdns-32.co.uk. ns-763.awsdns-31.net.
<input type="checkbox"/>	secqation.xyz	SOA	Simple	-	ns-140.awsdns-17.com. awsdns-hostmaster.am

8. Change your domain's default nameserver to these **external nameservers**.
9. Wait until the nameservers are updated successfully for your domain. Check the status by running following command:

`dig NS secqation.xyz`

10. Add records in your hosted zone to redirect a user to the CloudFront endpoint, that would fetch website contents from an S3 bucket
11. Click on **Create record** button
12. Leave **Record name** as empty
13. Switch on the **Alias** option
14. In "Route traffic to" dropdown list, select **Alias to CloudFront distribution**

Route 53 > Hosted zones > secqation.xyz > Create record

Quick create record [Info](#) [Switch to wizard](#)

Record 1 [Delete](#)

Record name Info <code>blog</code>	Record type Info A – Routes traffic to an IPv4 address and so...	Route traffic to Info <input checked="" type="radio"/> Alias Alias to CloudFront distribution
Valid characters: a-z, 0-9, !# \$ % & '()*+,-/:;<=>@[@\]^_`{ }.~		
<input type="text" value="d39zsqs284w5jb.cloudfront.net"/>		
Routing policy Info Simple routing	Evaluate target health <input type="radio"/> No	Add another record

[Cancel](#) [Create records](#)

15. Select the appropriate CloudFront distribution endpoint
16. Click on **Add another record** button
17. In the **Record name** field, enter `www`
18. In "Route traffic to" dropdown list, select **Alias to another record in this hosted zone**
19. Select the record that was just created, i.e. `secation.xyz`.

Route 53 > Hosted zones > secqation.xyz > Create record

Quick create record [Info](#) [Switch to wizard](#)

Record 1 [Delete](#)

Record name Info www	Record type Info A – Routes traffic to an IPv4 address and so...	Route traffic to Info Alias Alias to another record in this hosted zone US East (N. Virginia)
Valid characters: a-z, 0-9, ! " # \$ % & ' () * + , - / ; < = > ? @ [\] ^ _ ` { } . ~		
An alias to a CloudFront distribution and an alias to another record in the same hosted zone are global and available only in US East (N. Virginia).		
<input type="text" value="secqation.xyz"/> X		
Routing policy Info Simple routing	Evaluate target health <input checked="" type="checkbox"/> Yes	Add another record
		Cancel Create records

20. Click on **Create records** button

Hosted zone details [Edit hosted zone](#)

[Records \(4\)](#) [DNSSEC signing](#) [Hosted zone tags \(0\)](#)

Records (4) [Info](#)
Automatic mode is the current search behavior optimized for best filter results. [To change modes go to settings.](#)

<input type="checkbox"/>	Record name	Type	Routing policy	Differences	Value/Route traffic to
<input type="checkbox"/>	secqation.xyz	A	Simple	-	www.secqation.xyz. ns-140.awsdns-17.com. ns-1446.awsdns-52.org. ns-1796.awsdns-32.co.uk. ns-763.awsdns-31.net.
<input type="checkbox"/>	secqation.xyz	NS	Simple	-	ns-140.awsdns-17.com. awsdns-hostm...
<input type="checkbox"/>	secqation.xyz	SOA	Simple	-	ns-140.awsdns-17.com. awsdns-hostm...
<input type="checkbox"/>	www.secqation.xyz	A	Simple	-	d39zsgs284w5jb.cloudfront.net.

21. Open a browser window and enter your domain name to access the static website hosted on S3 bucket

<https://www.secqation.xyz>

Robo-Tests Log

Generated
20210905 22:41:00 UTC+05:30
13 hours 36 minutes ago

Test Statistics

Total Statistics		Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
All Tests		1	1	0	0	00:00:22	<div style="width: 100%;"> </div>

Statistics by Tag		Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
No Tags							<div style="width: 0%;"> </div>

Statistics by Suite		Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
Robo-Tests		1	1	0	0	00:00:23	<div style="width: 100%;"> </div>
Robo-Tests.Demo		1	1	0	0	00:00:23	<div style="width: 100%;"> </div>

Test Execution Log

- **SUITE** Robo-Tests
 - Full Name: Robo-Tests
 - Source: /robo-vault/robo-tests
 - Start / End / Elapsed: 20210905 22:40:37.653 / 20210905 22:41:00.200 / 00:00:22.547
 - Status: 1 test total, 1 passed, 0 failed, 0 skipped
- **SUITE** Demo
 - Full Name: Robo-Tests.Demo
 - Source: /robo-vault/robo-tests/demo.robot
 - Start / End / Elapsed: 20210905 22:40:37.693 / 20210905 22:41:00.195 / 00:00:22.502
 - Status: 1 test total, 1 passed, 0 failed, 0 skipped
- **TEST** Selenium In Docker Setup Should Work Correctly
 - Full Name: Robo-Tests.Demo.Selenium In Docker Setup Should Work Correctly
 - Start / End / Elapsed: 20210905 22:40:38.025 / 20210905 22:41:00.186 / 00:00:22.161
 - Status: **PASS**
 - + **KEYWORD** Begin Web Assessment
 - + **KEYWORD** SeleniumLibrary.Go To \${BASE_URL}
 - + **KEYWORD** SeleniumLibrary.Capture Page Screenshot
 - + **KEYWORD** SeleniumLibrary.Close All Browsers

17.4 S3 Bucket: Content Update

17.4.1 AWS Vault

AWS Vault is a tool to securely store and access AWS credentials in a development environment. AWS Vault stores IAM credentials in your operating system's

secure keystore and then generates **temporary credentials** from those to expose to your shell and applications.

1. Download **aws-vault** binary

```
$ wget https://github.com/99designs/aws-vault/releases/download/v6.3.1/aws-vault-linux-amd64 $ chmod +x aws-vault-linux-amd64 $ mv aws-vault-linux-amd64 /usr/local/bin/aws-vault
```

2. Create a new AWS **profile**, and save your IAM credentials securely

```
aws-vault add secqation
```

```
secqation@mirage:~/Desktop/NullconTraining2021/7-static-website-hosting/terraform$ aws-vault add secqation
Enter Access Key ID: test
Enter Secret Access Key:
Added credentials to profile "secqation" in vault
secqation@mirage:~/Desktop/NullconTraining2021/7-static-website-hosting/terraform$
```

3. Execute a command (using temporary credentials)

```
aws-vault exec secqation -- aws s3 ls
```

4. Authenticate with your AWS IAM account

```
aws-vault exec --duration=12h secqation
```

Note: This should set your AWS credentials in your current Terminal session.

5. View the temporary AWS credentials

```
aws-vault exec secqation -- env | grep AWS
```

17.4.2 AWS CLI

- Single file upload

```
aws s3 cp ./index.html s3://amazon-cloudfront-secure-static-site-s3bucket
```

- Multiple files upload

```
aws s3 cp ./test-suite/ s3://amazon-cloudfront-secure-static-site-s3bucketroot-9z7rxapslwau/ --recursive
```

17.5 CloudFront: Caching Policy Update

If changes are not reflecting on your website, **invalidate** the cached files, and update the chaching policy.

1. Go to **CloudFront** console
2. Select your distribution
3. Select **Behaviors** tab
4. **Edit** behavior
5. In **Edit Behavior** screen, scroll down to **Cache key and origin requests** section
6. Choose **Legacy cache settings**
7. Under **Object caching** section, choose **Customize**
8. For **Minimum TTL**, enter **0**
9. For **Maximum TTL**, enter **1**
10. For **Default TTL**, enter **1**

Note: After this update, your website will instantaneously reflect any changes made to the files stored in S3 bucket.

17.6 Update Dockerfile for Jenkins server

1. Add steps to install 4 new packages, namely **aws-vault**, **pass**, **GnuPG** and **AWS CLI** on our Jenkins server. These packages would help us in uploading files to S3 bucket in a secure manner.

```
FROM jenkins/jenkins:lts
USER root
RUN mkdir -p /tmp/download && \
curl -L https://download.docker.com/linux/static/stable/x86_64/docker-18.03.1-ce.tg
```

Note: Sample code is available in following path -

```
/home/secqation/Desktop/NullconTraining2021/4-example-docker/3-jenkins
```

2. Stop and remove the running Jenkins container

```
$ docker stop jenkins
$ docker rm jenkins
$ docker ps -a
```

3. Re-build the Jenkins Docker image

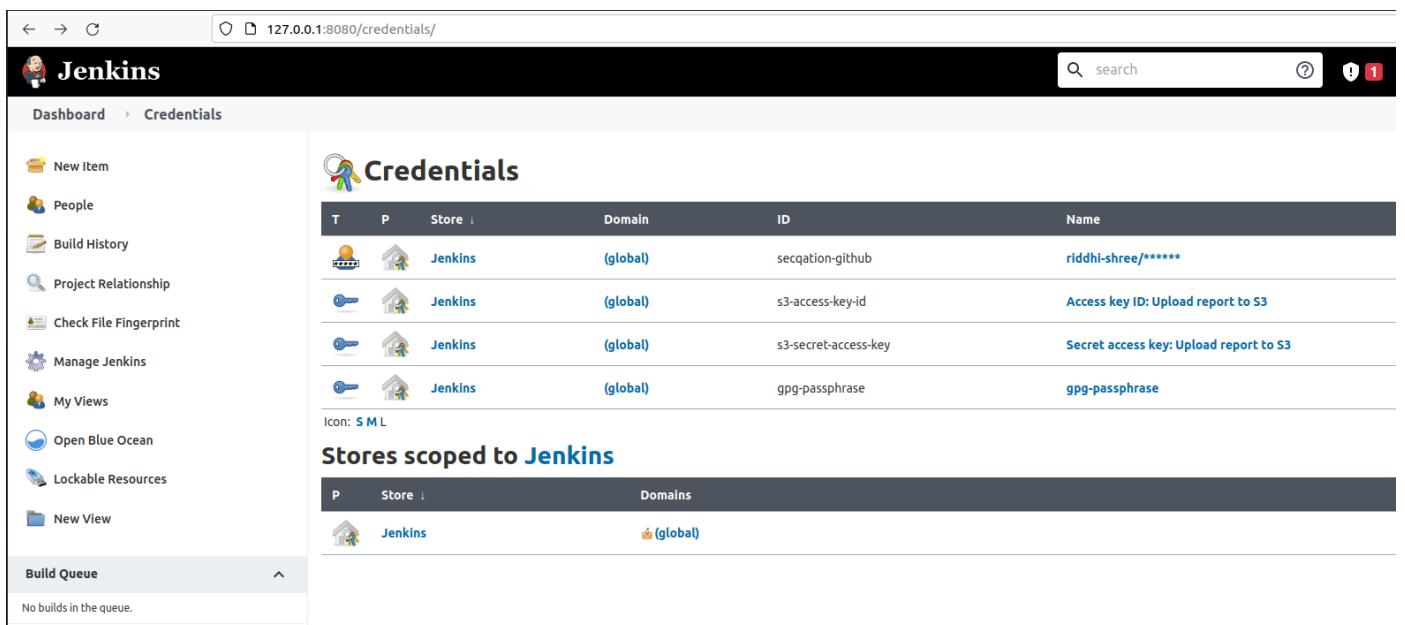
```
docker build -t jenkins-docker .
```

4. Start a new container for Jenkins server

```
docker run -p 8080:8080 -p 50000:50000 -v /var/jenkins_home:/var/jenkins_home -v /var/run/docker.sock:/var/run/docker.sock --name jenkins -d jenkins-docker
```

17.7 Add Secrets in Jenkins' Credentials Store

1. Go to <http://127.0.0.1:8080/>
2. Click on **Manage Jenkins > Manage Credentials**
3. Under **Domain** column, click on a link labeled as **(global)**



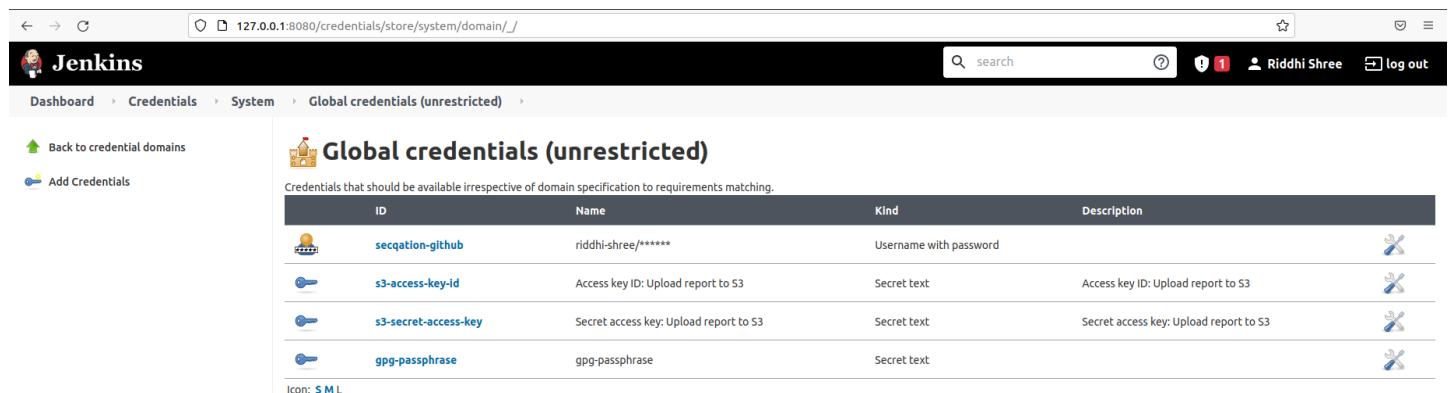
The screenshot shows the Jenkins 'Credentials' page at <http://127.0.0.1:8080/credentials/>. The left sidebar includes links for New Item, People, Build History, Project Relationship, Check File Fingerprint, Manage Jenkins, My Views, Open Blue Ocean, Lockable Resources, and New View. The main content area has a title 'Credentials' with a sub-section 'Stores scoped to Jenkins'. A table lists four global credentials:

T	P	Store	Domain	ID	Name
		Jenkins	(global)	secqation-github	riddhi-shree/*****
		Jenkins	(global)	s3-access-key-id	Access key ID: Upload report to S3
		Jenkins	(global)	s3-secret-access-key	Secret access key: Upload report to S3
		Jenkins	(global)	gpg-passphrase	gpg-passphrase

Below the table, it says 'Icon: S M L' and 'Stores scoped to Jenkins' with a table showing 'Domains' for Jenkins.

8. In the **Global credentials** page, click on **Add Credentials** link 9. From **Kind** dropdown list, select **Secret text** option 10. In the **Secret** input field, enter your secret value 11. In the **ID** input field, enter an identifying name 12. Click on **OK** button

Repeat above steps for storing AWS access key ID, AWS secret access key, and any other secrets.



The screenshot shows the Jenkins 'Global credentials (unrestricted)' page at http://127.0.0.1:8080/credentials/store/system/domain/_/. The left sidebar includes links for Back to credential domains and Add Credentials. The main content area has a title 'Global credentials (unrestricted)' with a subtitle 'Credentials that should be available irrespective of domain specification to requirements matching.' A table lists four credentials:

ID	Name	Kind	Description
secqation-github	riddhi-shree/*****	Username with password	
s3-access-key-id	Access key ID: Upload report to S3	Secret text	Access key ID: Upload report to S3
s3-secret-access-key	Secret access key: Upload report to S3	Secret text	Secret access key: Upload report to S3
gpg-passphrase	gpg-passphrase	Secret text	

Below the table, it says 'Icon: S M L'.

17.8 Update Jenkinsfile

1. In the **Jenkinsfile**, add steps to upload the robot test execution results to S3 bucket.

```
pipeline{ agent any stages{ stage('Cleanup'){ steps{ sh 'docker stop selenium-hub firefox_1 firefox_2 robotframework || true' sh 'docker rm selenium-hub firefox_1 firefox_2' } } }
```

Note: Sample code is available in following path -

```
/home/secqation/Desktop/NullconTraining2021/7-static-website-hosting
```

17.9 Push Changes to GitHub Repository

1. Start **ngrok**

```
./ngrok http 8080
```

```
secqation@mirage: ~/Downloads/ngrok-stable-linux-amd64
secqation@mirage: ~/Downloads/ngrok-stable-linux-amd64
ngrok by @inconshreveable

Session Status           online
Account                  Riddhi Shree (Plan: Free)
Version                 2.3.40
Region                  United States (us)
Web Interface            http://127.0.0.1:4040
Forwarding               http://0efc-49-37-75-37.ngrok.io -> http://localhost:8080
Forwarding               https://0efc-49-37-75-37.ngrok.io -> http://localhost:8080

Connections             ttl     opn      rt1      rt5      p50      p90
                         6       0       0.00    0.00    3.31    13.15

HTTP Requests
-----
POST /github-webhook/                200 OK
POST /github-webhook/                200 OK
GET  /login                           200 OK
GET  /                                403 Forbidden
GET  /favicon.ico                     200 OK
GET  /static/687d9c0b/images/jenkins.svg 200 OK
GET  /static/687d9c0b/css/simple-page-forms.css 200 OK
GET  /static/687d9c0b/css/simple_page_theme.css 200 OK
```

2. Update the **Payload URL** for webhook in GitHub

Options
Manage access
Security & analysis
Branches
Webhooks
Notifications
Integrations
Deploy keys

Webhooks / Manage webhook

Settings

Recent Deliveries

We'll send a POST request to the URL below with details of any subscribed event: receive (JSON, x-www-form-urlencoded, etc). More information can be found in the GitHub API documentation.

Payload URL *

http://0efc-49-37-75-37.ngrok.io/github-webhook/

Content type

application/x-www-form-urlencoded

3. Push local changes to remote GitHub repository

```
$ git add . $ git commit -m "Test S3 upload" $ git push
```

```
secqation@mirage:~/Desktop/NullconTraining2021/7-static-website-hosting$ git status
On branch master
Your branch is up to date with 'origin/master'.

Changes not staged for commit:
  (use "git add <file>..." to update what will be committed)
    (use "git restore <file>..." to discard changes in working directory)
      modified: Jenkinsfile

no changes added to commit (use "git add" and/or "git commit -a")
secqation@mirage:~/Desktop/NullconTraining2021/7-static-website-hosting$ git add .
secqation@mirage:~/Desktop/NullconTraining2021/7-static-website-hosting$ git commit -m "Test S3 upload"
[master c05ecaa] Test S3 upload
 1 file changed, 19 deletions(-)
secqation@mirage:~/Desktop/NullconTraining2021/7-static-website-hosting$ git push
Enumerating objects: 5, done.
Counting objects: 100% (5/5), done.
Delta compression using up to 2 threads
Compressing objects: 100% (3/3), done.
Writing objects: 100% (3/3), 292 bytes | 146.00 KiB/s, done.
Total 3 (delta 2), reused 0 (delta 0)
remote: Resolving deltas: 100% (2/2), completed with 2 local objects.
To github.com:riddhi-shree/secQAtion-jenkins-pipeline.git
 9081421..c05ecaa master -> master
secqation@mirage:~/Desktop/NullconTraining2021/7-static-website-hosting$
```

4. Wait for a while and visit your website at https://www.secqation.xyz to view the updated website contents.

17.10 Is there a problem?

Unauthenticated users can access our sensitive security report...

17.11 References

- <https://github.com/aws-samples/amazon-cloudfront-secure-static-site#user-content-amazon-cloudfront-secure-static-website>
- <https://docs.aws.amazon.com/Route53/latest/DeveloperGuide/dns-configuring.html>
- <https://aws.amazon.com/premiumsupport/knowledge-center/prevent-cloudfront-from-caching-files/>
- <https://docs.aws.amazon.com/AmazonCloudFront/latest/DeveloperGuide/Invalidation.html#invalidation-specifying-objects-paths>
- <https://aws.amazon.com/blogs/networking-and-content-delivery/adding-http-security-headers-using-Lambdaedge-and-amazon-cloudfront/>
- https://infosec.mozilla.org/guidelines/web_security
- <https://docs.aws.amazon.com/AmazonS3/latest/userguide/WebsiteAccessPermissionsReqd.html>
- <https://aws.amazon.com/premiumsupport/knowledge-center/cloudfront-serve-static-website/>
- <https://londonappdeveloper.com/2021/04/12/how-to-use-terraform-via-docker-compose-for-professional-developers/>
- <https://github.com/99designs/aws-vault>
- <https://docs.aws.amazon.com/cli/latest/userguide/install-cliv2-linux.html>
- https://registry.terraform.io/providers/hashicorp/aws/latest/docs/resources/cloudfront_distribution
- <https://www.jenkins.io/doc/book/pipeline/syntax/#declarative-sections>
- <https://medium.com/cuc4/aws-credential-hygiene-on-dev-machines-ed9e8793ea67>
- <https://blog.chapagain.com.np/gpg-remove-keys-from-your-public-keyring/>
- <https://www.passwordstore.org/>
- https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/6/html/security_guide/sect-security_guide-encryption-gpg-creating_gpg_keys_using_gpg
- <https://stackoverflow.com/questions/49072403/suppress-the-passphrase-prompt-in-gpg-command>
- <https://docs.github.com/en/github/authenticating-to-github/managing-commit-signature-verification/generating-a-new-gpg-key>
- <https://stackoverflow.com/questions/39596446/how-to-get-gpg-public-key-in-bash>
- <https://www.passwordstore.org/>

18. Implementing authentication via Lamda Edge

18.1 Private S3 Bucket

1. Login to AWS account
2. Go to <https://s3.console.aws.amazon.com/>
3. Click on "Create bucket" button
4. Enter a unique bucket name
5. Click on "Create bucket" button
6. A private S3 bucket should be created successfully.

18.2 File Upload

1. Create "index.html" file

```
<html>My index file</html>
```

2. Go to <https://s3.console.aws.amazon.com/>
3. Select the newly created S3 bucket
4. Click on "Upload" button
5. Click on "Add files" button
6. Select your "index.html" file
7. Click on "Upload" >> "Close"

18.3 CloudFront

1. Go to <https://console.aws.amazon.com/cloudfront>
2. Click on "Create distribution" button
3. Under "Origin domain", select your newly created private S3 bucket
4. Under "S3 bucket access", select [Yes use OAI \(bucket can restrict access to only CloudFront\)](#)
5. Click on "Create new OAI"
 1. Accept default entry, or, enter a new name
 2. Click on "Create" button
6. Under "Bucket policy", select [Yes, update the bucket policy](#)
7. Under "Viewer protocol policy", select [HTTPS only](#)
8. Under "Allowed HTTP methods", select [GET, HEAD](#)
9. Under "Settings" > "Price class", select a suitable option
10. Under "Settings" > "Default root object", enter [index.html](#)

11. Click on "Create distribution" button
12. Note down the CloudFront distribution ID

18.4 Lambda@Edge

Lambda@Edge functions must be created in `us-east-1` region.

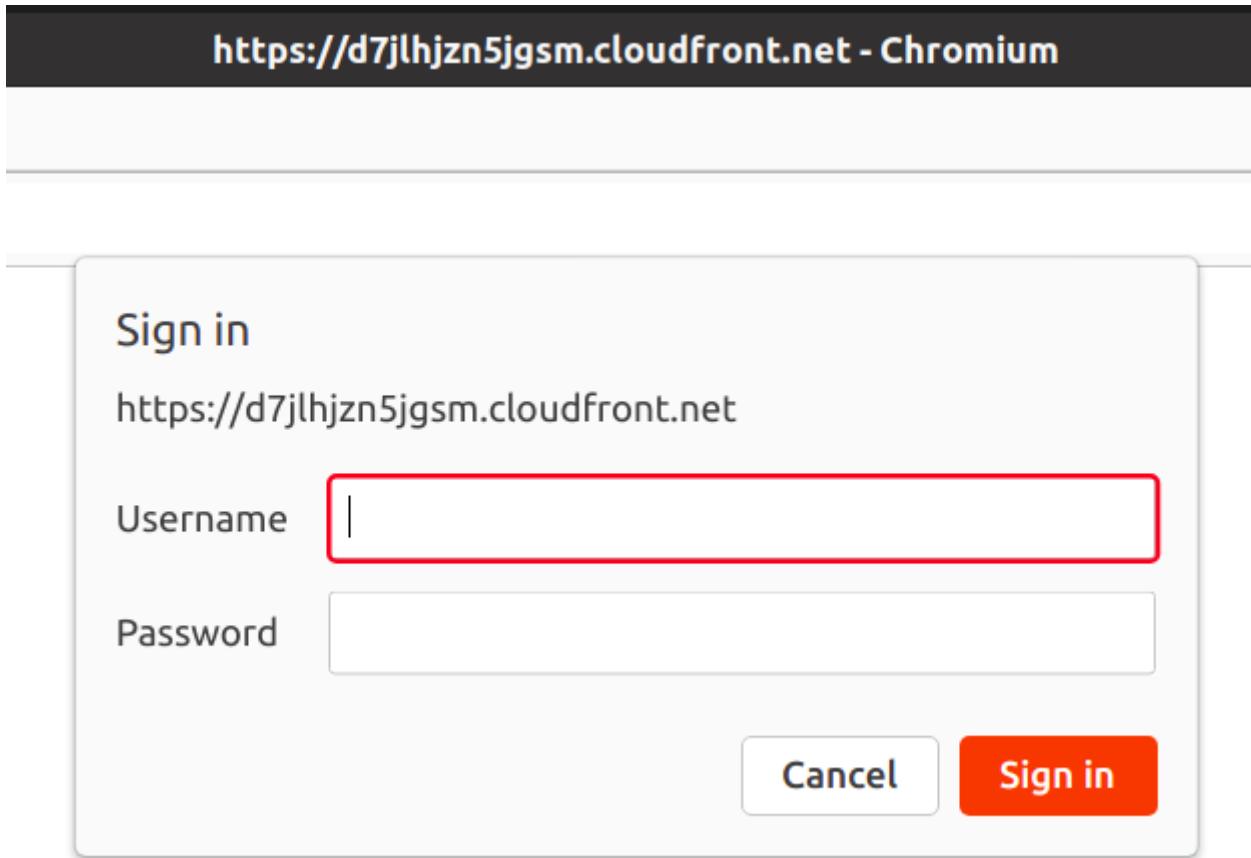
1. Go to <https://console.aws.amazon.com/Lambda/home?region=us-east-1>
2. Click on "Create function" button
3. Select **Author from scratch** option
4. Enter a function name of your choice
5. Accept the default runtime, i.e. **Node.js 14.x**
6. Click on "Create function" button
7. Scroll down to "Code source" section
8. Replace the contents of "index.js" file with

```
exports.handler = (event, context, callback) => { // Get the request and its headers
  const request = event.Records[0].cf.request;
  const headers = request.headers;
  const body = request.body;
  const querystring = require('querystring');
  const qs = querystring.parse(body);
  const url = qs.url;
  const method = qs.method;
  const params = qs.params;
  const headersObject = {};
  const parsedHeaders = headers.map(header => {
    const [key, value] = header.value.split(': ');
    return { key, value };
  });
  parsedHeaders.forEach(header => {
    if (header.key === 'Content-Type') {
      headersObject['Content-Type'] = header.value;
    } else {
      headersObject[header.key] = header.value;
    }
  });
  const response = {
    status: 200,
    statusDescription: 'OK',
    headers: [
      { key: 'Content-Type', value: 'text/html' },
      ...headersObject
    ],
    body: `Hello, ${url}! Method: ${method}, Params: ${JSON.stringify(params)}`
  };
  callback(null, response);
}
```

9. Click on "Deploy" button
10. Click on "Configuration" tab >> "Permissions"
11. Under "Execution role", click on the role name
12. On the IAM role page, click on "Trust relationships" >> "Edit trust relationship"
13. Replace the line **"Service": "Lambda.amazonaws.com"** with

```
"Service": ["Lambda.amazonaws.com", "edgeLambda.amazonaws.com"]
```

14. Click on "Update Trust Policy" button
15. Return to the Lambda@Edge function page
16. Click on "Actions" button
17. Click on **Publish new version** >> "Publish"
18. Click on "Add trigger"
19. Under "Trigger configuration", select **CloudFront**
20. Under "Configure CloudFront trigger" >> "Distribution", select the correct CloudFront distribution ID
21. Under "CloudFront event", select **Viewer request**
22. Select the **Confirm deploy to Lambda@Edge** checkbox
23. Click on "Add" button
24. Navigate to <https://console.aws.amazon.com/cloudfront>
25. Wait until the CloudFront distribution status changes from **Deploying...** to **Enabled**
26. This process might take a few minutes to complete
27. Once enabled, copy the **distribution domain name** and paste it in a browser window, e.g.
<https://d7j1hjzn5gsm.cloudfront.net>
28. You would be prompted to enter a valid username and password



29. Only after correct credentials are provided, user would be allowed to access the [index.html](#) page

18.5 Cleanup

When not needed anymore, make sure the S3, CloudFront and Lambda@Edge AWS resources are deleted. This will prevent incurring unnecessary costs.

CloudFront

1. Select your CloudFront distribution
2. Click on "Disable" button
3. Once disabled completely, click on "Delete" button
4. Click "Delete" in the confirmation box

S3 Bucket

1. Select your S3 bucket
2. Click on "Empty" button
3. Enter
4. Click on "Empty" >> "Exit"
5. Select your emptied S3 bucket
6. Click on "Delete" button
7. Enter name of your S3 bucket in the confirmation box
8. Click on "Delete bucket" button

Lambda@Edge

1. Wait for a few hours
2. Select your lambda@edge function
3. Click on "Actions" >> "Delete"
4. [Refer this link](#)

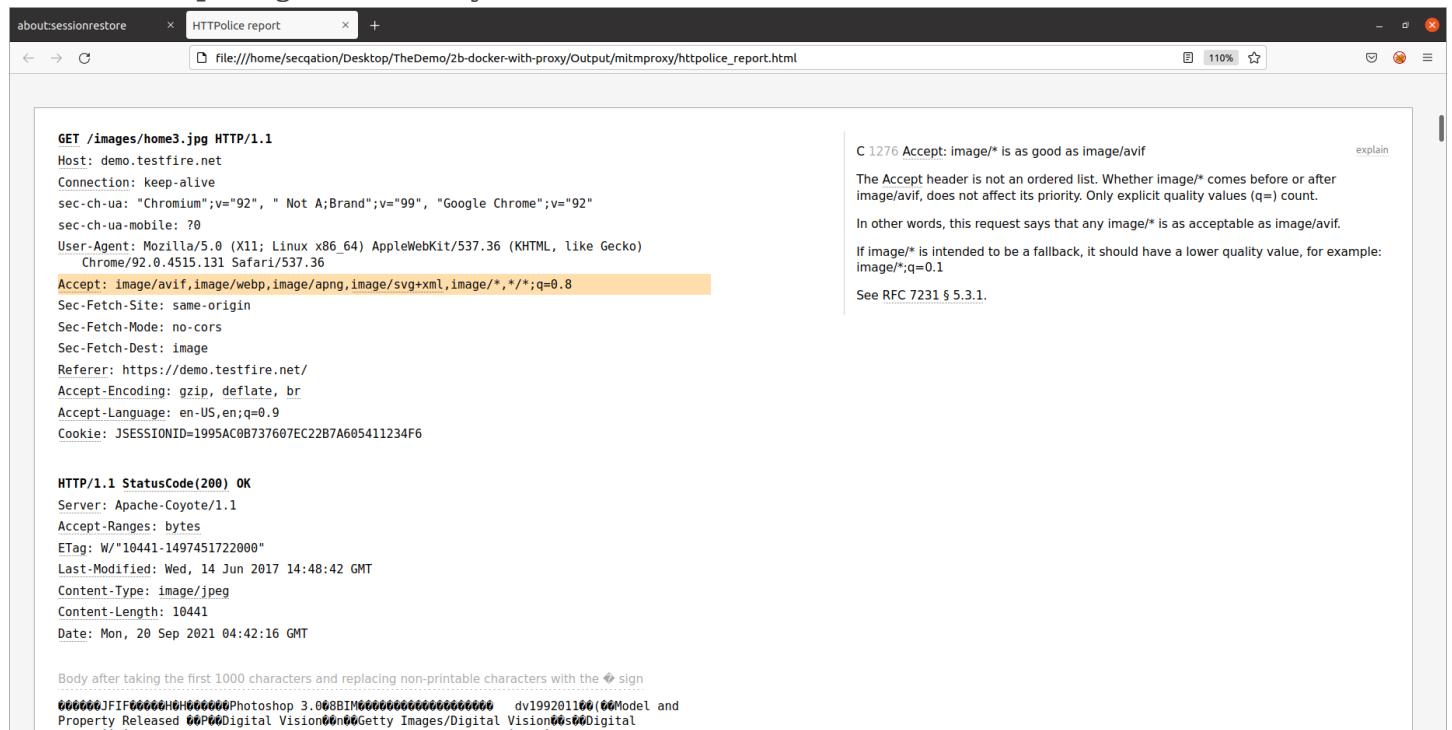
Note: Replicated functions cannot be deleted instantly

19. Leveraging HTTPPolice

HTTPPolice is a validator or “linter” for **HTTP requests and responses**. It can spot bad header syntax, inappropriate status codes, and other potential problems in your HTTP server or client.

19.1 Sample Report

The final report generated by HTTPPolice looks like this:



about:sessionrestore x HTTPPolice report +
file:///home/secquestion/Desktop/TheDemo/2b-docker-with-proxy/Output/mitmproxy/httppolice_report.html
110% ⚡

GET /images/home3.jpg HTTP/1.1
Host: demo.testfire.net
Connection: keep-alive
sec-ch-ua: "Chromium";v="92", "Not A;Brand";v="99", "Google Chrome";v="92"
sec-ch-ua-mobile: ?0
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko)
Chrome/92.0.4515.131 Safari/537.36
Accept: image/avif,image/webp,image/apng,image/svg+xml,image/*,*/*;q=0.8
Sec-Fetch-Site: same-origin
Sec-Fetch-Mode: no-cors
Sec-Fetch-Dest: image
Referer: https://demo.testfire.net/
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: JSESSIONID=1995AC0B737607EC22B7A605411234F6

HTTP/1.1 StatusCode(200) OK
Server: Apache-Coyote/1.1
Accept-Ranges: bytes
ETag: W/"10441-1497451722000"
Last-Modified: Wed, 14 Jun 2017 14:48:42 GMT
Content-Type: image/jpeg
Content-Length: 10441
Date: Mon, 20 Sep 2021 04:42:16 GMT

Body after taking the first 1000 characters and replacing non-printable characters with the ⚡ sign
000000JFIF0000000000Photoshop 3.008BIM000000000000dv199201160/00Model and
Property Released 00P00Digital Vision00n00Getty Images/Digital Vision00s00Digital

C 1276 Accept: image/* is as good as image/avif
explain
The Accept header is not an ordered list. Whether image/* comes before or after image/avif, does not affect its priority. Only explicit quality values (q=) count.
In other words, this request says that any image/* is as acceptable as image/avif.
If image/* is intended to be a fallback, it should have a lower quality value, for example:
image/*;q=0.1
See RFC 7231 § 5.3.1.

19.2 Installation

```
pip3 install mitmproxy-HTTPPolice
```

19.3 Integrating HTTPPolice with mitmproxy

The network traffic sniffed by mitmproxy tool can be filtered and passed as an input to HTTPPolice, which in turn can generate an HTML report containing full requests and responses. Analysing, storing and sharing of a well structured auto-generated HTML report is a lot easier than trying to organize the data manually.

- Run following commands:

```
$ cd /home/secqation/Desktop/TheDemo/3b-jenkins-pipeline/ $ ./init.sh
```

- Check the contents of **init.sh** file. What is it doing?

```
rm Input/mitmproxy/* rm Output/mitmproxy/* rm Output/robotframework/* docker-compose down docker-compose up -d --build mitmproxy docker-com
```

- Check the contents of **docker-compose.yml** file. How did we configure proxy?

Notice the use of environment variables **HTTP_PROXY** and **HTTPS_PROXY**

```
version: "3.9" services: firefox: image: selenium/node-firefox:4.0.0-rc-1-prerelease-20210804 shm_size: 2gb container_name: firefox depends_on: - sele
```

- Also take a note of the volume mappings (especially for "mitmproxy" service)

- Check the contents of **Dockerfile** for **mitmproxy** service.

```
FROM mitmproxy/mitmproxy RUN apt-get update RUN apt-get install procps curl -y ENV LANG=en_US.UTF-8 VOLUME /home/mitmproxy/.mitmproxy CMD ["mitmdump",
```

Notice that the intercepted traffic is being saved into a file called **traffic.mitm**. This file is saved in an **Output** folder that has a corresponding volume mapping in our host machine.

- Once the **init.sh** script execution completes successfully, locate the **traffic.mitm** output file in path `/home/secqation/Desktop/TheDemo/3b-jenkins-pipeline/Output/mitmproxy`

- Check the contents of **docker-compose-httpolice.yml** file

```
version: "3.9" services: HTTPPolice: build: context: . dockerfile: ./Input/HTTPPolice/Dockerfile image: httppolice:latest container_name: httppolice volum
```

- Run HTTPPolice in **interactive mode** using Docker Compose **run** command

```
docker-compose -f docker-compose-httpolice.yml run HTTPPolice
```

```

secqation@mirage: ~/Desktop/TheDemo/3b-jenkins-pipeline
secqation@mirage: ~/Desktop/TheDemo/3b-jenkins-pipeline

Flows
>> POST https://accounts.google.com/ListAccounts?gpsia=1&source=ChromiumBro...
    ← 200 application/json 43b 108ms
GET https://demo.testfire.net/
    ← 200 text/html 9.15k 284ms
GET https://demo.testfire.net/style.css
    ← 200 text/css 1.22k 278ms
GET https://demo.testfire.net/images/logo.gif
    ← 200 image/gif 4.87k 337ms
GET https://demo.testfire.net/images/gradient.jpg
    ← 200 image/jpeg 894b 280ms
GET https://demo.testfire.net/images/home3.jpg
    ← 200 image/jpeg 10.2k 339ms
GET https://demo.testfire.net/images/header_pic.jpg
    ← 200 image/jpeg 15.83k 282ms
GET https://demo.testfire.net/images/pf_lock.gif
    ← 200 image/gif 76b 335ms
GET https://demo.testfire.net/images/home1.jpg
    ← 200 image/jpeg 7.71k 270ms
GET https://demo.testfire.net/images/home2.jpg
    ← 200 image/jpeg 5.77k 263ms
GET https://demo.testfire.net/favicon.ico
↓ [1/86] [scripts:1]
: set view_filter '!'

```

9. Filter out unwanted traffic. Press **[F]** key on keyboard and enter a [filter expression](#)

```
: set view_filter '! ~u firefox|mozilla' : set view_filter '! (~t "image/")' : set view_filter '(~t text/)'
```

```
secqation@mirage: ~/Desktop/TheDemo/3b-jenkins-pipeline
secqation@mirage: ~/Desktop/TheDemo/3b-jenkins-pipeline

Flows
>> GET https://demo.testfire.net/style.css
    ← 200 text/css 1.22k 278ms
GET https://demo.testfire.net/favicon.ico
    ← 404 text/html 6.76k 263ms
GET https://demo.testfire.net/
    ← 200 text/html 9.15k 268ms
GET https://demo.testfire.net/
    ← 200 text/html 9.15k 268ms
GET https://demo.testfire.net/style.css
    ← 200 text/css 1.22k 259ms
GET https://demo.testfire.net/cgi.exe
    ← 404 text/html 6.76k 265ms
GET https://demo.testfire.net/default.jsp?content=security.htm
    ← 404 text/html 6.76k 274ms
GET https://demo.testfire.net/feedback.jsp
    ← 200 text/html 8.3k 262ms
GET https://demo.testfire.net/index.jsp
    ← 200 text/html 9.15k 262ms
GET https://demo.testfire.net/index.jsp?content=business.htm
    ← 200 text/html 8.29k 263ms
↓ [3/47] [f:! (~t "image/"))][scripts:1]
: set view_filter '! (~t "image/")'
```

10. Export the filtered HTTP requests and responses using HTTPPolice command

```
: httpolice.report.html @shown /home/mitmproxy/Output/httpolice_report.html
```

```
secqation@mirage: ~/Desktop/TheDemo/3b-jenkins-pipeline
secqation@mirage: ~/Desktop/TheDemo/3b-jenkins-pipeline

Flows
>>04:42:14 HTTPS POST ...nts.google.com /ListAccounts?gpsia=1&source=Chr... 200 ...plication/json 43b 108ms
04:42:15 HTTPS GET ...o.testfire.net /
200 text/html 9.15k 284ms
04:42:16 HTTPS GET ...o.testfire.net /style.css
200 text/css 1.22k 278ms
04:42:17 HTTPS GET ...o.testfire.net /favicon.ico
404 text/html 6.76k 263ms
04:42:17 HTTPS GET ...o.testfire.net /
200 text/html 9.15k 268ms
04:42:18 HTTPS GET ...o.testfire.net /
200 text/html 9.15k 268ms
04:42:18 HTTPS GET ...o.testfire.net /style.css
200 text/css 1.22k 259ms
04:42:19 HTTPS GET ...o.testfire.net /cgi.exe
404 text/html 6.76k 265ms
04:42:20 HTTPS GET ...o.testfire.net /default.jsp?content=security.htm 404
text/html 6.76k 274ms
04:42:21 HTTPS GET ...o.testfire.net /feedback.jsp
200 text/html 8.3k 262ms
04:42:21 HTTPS GET ...o.testfire.net /index.jsp
200 text/html 9.15k 262ms
04:42:22 HTTPS GET ...o.testfire.net /index.jsp?content=business.htm 200
text/html 8.29k 263ms
04:42:23 HTTPS GET ...o.testfire.net /index.jsp?content=business_card...
200 text/html 7.84k 312ms
04:42:23 HTTPS GET ...o.testfire.net /index.jsp?content=business_depo...
200 text/html 7.66k 267ms
04:42:24 HTTPS GET ...o.testfire.net /index.jsp?content=business_insu...
200 text/html 7.53k 267ms
04:42:25 HTTPS GET ...o.testfire.net /index.jsp?content=business_lend...
200 text/html 7.66k 266ms
04:42:26 HTTPS GET ...o.testfire.net /index.jsp?content=business_othe...
200 text/html 7.51k 263ms
04:42:27 HTTPS GET ...o.testfire.net /index.jsp?content=business_reti...
200 text/html 6.76k 261ms
04:42:27 HTTPS GET ...o.testfire.net /retirement.htm
200 text/html 1.09k 261ms
04:42:28 HTTPS GET ...o.testfire.net /index.jsp?content=inside.htm
200 text/html 8.17k 262ms
04:42:28 HTTPS GET ...o.testfire.net /index.jsp?content=inside_about...
200 text/html 7.73k 265ms
04:42:29 HTTPS GET ...o.testfire.net /index.jsp?content=inside_career...
200 text/html 8.11k 263ms
↓ [1/47] [f:! (~t image/))][scripts:1]
: httpolice.report.html @shown /home/mitmproxy/Output/httpolice_report.html
```

11. Once the **HTML report** is exported, open it in a browser (and analyze)

```

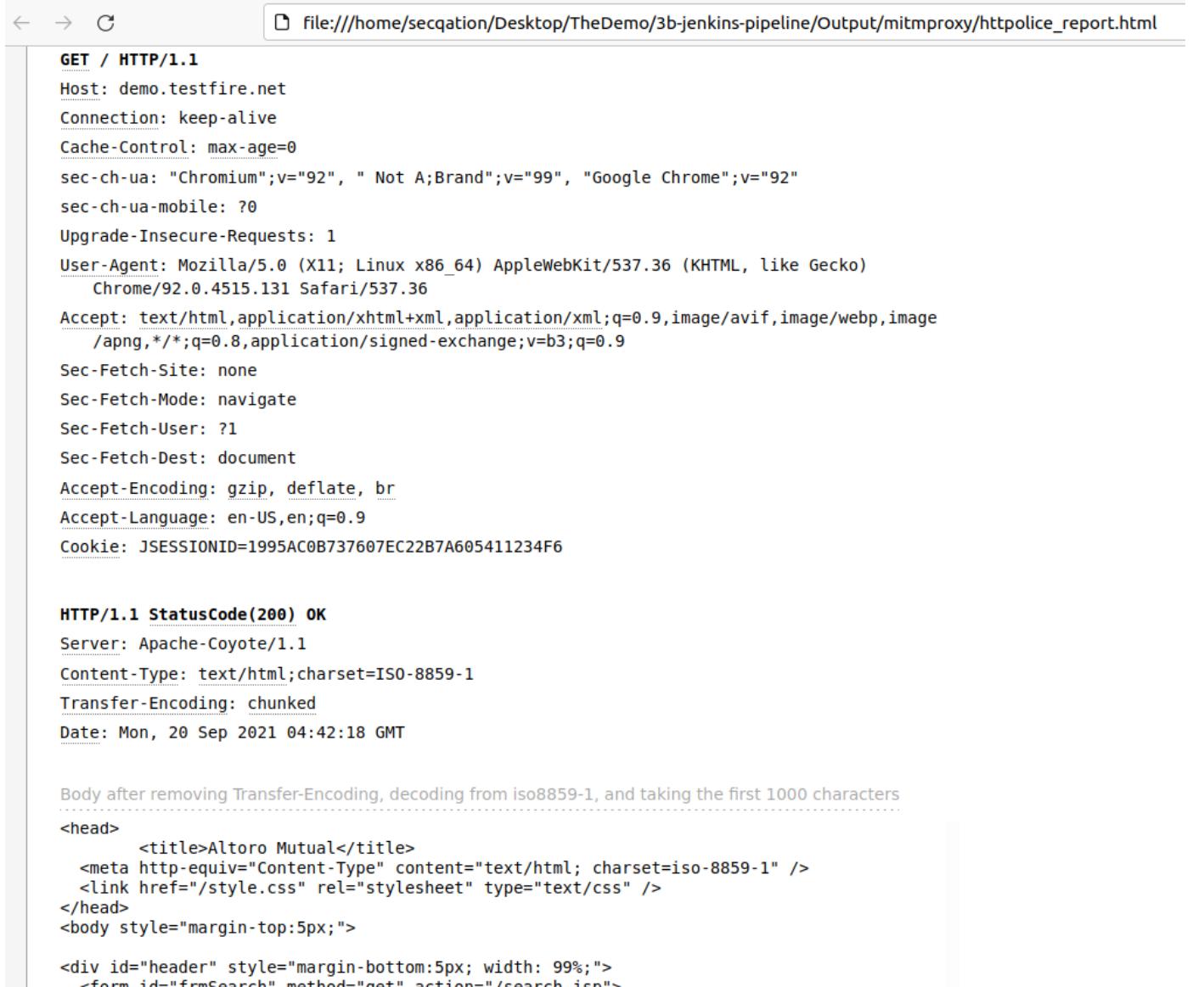
secqation@mirage: ~/Desktop/TheDemo/3b-jenkins-pipeline
secqation@mirage: ~/Desktop/TheDemo/3b-jenkins-pipeline

Flows
>>04:42:14 HTTPS POST ...nts.google.com /ListAccounts?gpsia=1&source=Chr... 200 ...plication/json 43b 108ms ●
04:42:15 HTTPS GET ...o.testfire.net / 200 text/html 9.15k 284ms
04:42:16 HTTPS GET ...o.testfire.net /style.css 200 text/css 1.22k 278ms
04:42:17 HTTPS GET ...o.testfire.net /favicon.ico 404 text/html 6.76k 263ms ●
04:42:17 HTTPS GET ...o.testfire.net / 200 text/html 9.15k 268ms
04:42:18 HTTPS GET ...o.testfire.net / 200 text/html 9.15k 268ms
04:42:18 HTTPS GET ...o.testfire.net /style.css 200 text/css 1.22k 259ms
04:42:19 HTTPS GET ...o.testfire.net /cgi.exe 404 text/html 6.76k 265ms
04:42:20 HTTPS GET ...o.testfire.net /default.jsp?content=security.htm 404 text/html 6.76k 274ms
04:42:21 HTTPS GET ...o.testfire.net /feedback.jsp 200 text/html 8.3k 262ms
04:42:21 HTTPS GET ...o.testfire.net /index.jsp 200 text/html 9.15k 262ms
04:42:22 HTTPS GET ...o.testfire.net /index.jsp?content=business.htm 200 text/html 8.29k 263ms
04:42:23 HTTPS GET ...o.testfire.net /index.jsp?content=business_card... 200 text/html 7.84k 312ms
04:42:23 HTTPS GET ...o.testfire.net /index.jsp?content=business_depo... 200 text/html 7.66k 267ms
04:42:24 HTTPS GET ...o.testfire.net /index.jsp?content=business_insu... 200 text/html 7.53k 267ms
04:42:25 HTTPS GET ...o.testfire.net /index.jsp?content=business_lend... 200 text/html 7.66k 266ms
04:42:26 HTTPS GET ...o.testfire.net /index.jsp?content=business_othe... 200 text/html 7.51k 263ms
04:42:27 HTTPS GET ...o.testfire.net /index.jsp?content=business_reti... 200 text/html 6.76k 261ms
04:42:27 HTTPS GET ...o.testfire.net /retirement.htm 200 text/html 1.09k 261ms
04:42:28 HTTPS GET ...o.testfire.net /index.jsp?content=inside.htm 200 text/html 8.17k 262ms
04:42:28 HTTPS GET ...o.testfire.net /index.jsp?content=inside_about... 200 text/html 7.73k 265ms
04:42:29 HTTPS GET ...o.testfire.net /index.jsp?content=inside_career... 200 text/html 8.11k 263ms
↓ [1/47] [f:! (~t image/)][scripts:1]
Alert: HTTPPolice: wrote report on 47 flows to /home/mitmproxy/Output/httpolice_report.html

```

```
$ firefox httpolice_report.html
```

```
secqation@mirage:~/Desktop/TheDemo/3b-jenkins-pipeline/Output/mitmproxy$ ll
total 2964
drwxr-xr-x 2 secqation seqation 4096 Sep 26 14:08 .
drwxr-xr-x 4 secqation seqation 4096 Sep 26 09:33 ..
-rw-r--r-- 1 secqation seqation 307522 Sep 26 14:06 httpolice_report.html
-rw-r--r-- 1 secqation seqation 2715309 Sep 26 09:33 traffic.mitm
secqation@mirage:~/Desktop/TheDemo/3b-jenkins-pipeline/Output/mitmproxy$ firefox httpolice_report.html
secqation@mirage:~/Desktop/TheDemo/3b-jenkins-pipeline/Output/mitmproxy$
```



```
GET / HTTP/1.1
Host: demo.testfire.net
Connection: keep-alive
Cache-Control: max-age=0
sec-ch-ua: "Chromium";v="92", " Not A;Brand";v="99", "Google Chrome";v="92"
sec-ch-ua-mobile: ?0
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (X11; Linux x86_64) AppleWebKit/537.36 (KHTML, like Gecko)
    Chrome/92.0.4515.131 Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/avif,image/webp,image
    /apng,*/*;q=0.8,application/signed-exchange;v=b3;q=0.9
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br
Accept-Language: en-US,en;q=0.9
Cookie: JSESSIONID=1995AC0B737607EC22B7A605411234F6

HTTP/1.1 StatusCode(200) OK
Server: Apache-Coyote/1.1
Content-Type: text/html;charset=ISO-8859-1
Transfer-Encoding: chunked
Date: Mon, 20 Sep 2021 04:42:18 GMT

Body after removing Transfer-Encoding, decoding from iso8859-1, and taking the first 1000 characters
<head>
    <title>Altoro Mutual</title>
    <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
    <link href="/style.css" rel="stylesheet" type="text/css" />
</head>
<body style="margin-top:5px;">

    <div id="header" style="margin-bottom:5px; width: 99%;">
        <form id="frmSearch" method="get" action="/search isn">
```

12. Press **[Q]** key on keyboard. When prompted, press **[Y]** key to exit the interactive mitmproxy shell

```
04:42:28 HTTPS GET ...o.testfire.net
04:42:29 HTTPS GET ...o.testfire.net
04:42:30 HTTPS GET ...o.testfire.net
↓ [6/43] [f:(~t text/)][scripts:1]
Quit (yes,no)? 
```

19.4 Reference

- <https://httpolice.readthedocs.io/en/latest/index.html>
- <https://httpolice.readthedocs.io/en/latest/concepts.html>
- <https://httpolice.readthedocs.io/en/latest/quickstart.html#using-mitmproxy>
- <https://docs.mitmproxy.org/stable/concepts-filters/>
- <https://buildmedia.readthedocs.org/media/pdf/httpolice/0.7.0/httpolice.pdf>

20. Basics of Python programming

20.1 Interpreter

1. Open a terminal by pressing **[CTRL]+[ALT]+[T]** keys
2. Type `python3` and press **[ENTER]**

```
secqation@mirage:~/Desktop/NullconTraining2021$ python3
Python 3.8.10 (default, Jun 2 2021, 10:49:15)
[GCC 9.4.0] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>> 
```

3. Try out below mentioned examples, and understand what's happening.
4. To exit the interpreter, type `quit()` and press **[ENTER]**

20.2 Numerical Operators

- `+`, `-`, `*`, `/`, `//` (floor division), `%` (remainder), `**` (power), `=`

20.3 Strings

1. Run following two commands and explain the difference in output

```
>>> print('C:\some\name') >>> print(r'C:\some\name')
```

Which of the above represents a **raw string**?

2. String literals can span **multiple lines** by using triple-quotes: `'''...'''` or `'''...'''`.
3. Strings can be **concatenated** with the `+` operator, and **repeated** with `*`

```
>>> 3 * 'un' + 'ium'
```

4. Slicing

```
>>> word = 'Python' >>> word[0:2] # characters from position 0 (included) to 2 (excluded) >>> word[2:5] # characters from position 2 (included) to 5 (excluded)
```

Note how the start is always included, and the end always excluded. This makes sure that `s[:i] + s[i:]` is always equal to `s`:

+-----+	-+-----+	-+-----+	-+-----+	-+-----+	-+-----+	-+-----+
P	y	t	h	o	n	
+-----+	-+-----+	-+-----+	-+-----+	-+-----+	-+-----+	-+-----+
0	1	2	3	4	5	6
-6	-5	-4	-3	-2	-1	

20.4 Lists

```
>>> cubes = [1, 8, 27, 65, 125] >>> cubes.append(216) # add the cube of 6
```

```
>>> letters = ['a', 'b', 'c', 'd', 'e', 'f', 'g'] >>> letters[2:5] = ['C', 'D', 'E'] >>> letters[2:5] = [] >>> letters[:] = [] >>> len(letters)
```

20.5 Control Flow Tools

20.5.1 if

20.5.4 range()

```
>>> x = int(input("Please enter an integer: ")) Please enter an integer: 42 > >>> for i in range(5): ... print(i) ... e changed to zero' ... elif x == 0:
```

20.5.2 for

```
>>> list(range(0, 10, 3)) [0, 3, 6, 9]
```

```
>>> # Measure some strings: ... words = ['cat', 'window', 'defenestrate'] >>> for w in words: ... print(w, len(w)) ...
```

20.5.5 break

20.5.3 while

```
>>> for n in range(2, 10): ... for x in range(2, n): ... if n % x == 0: ... p
```

```
>>> # Fibonacci series: ... a, b = 0, 1 >>> while a < 1000: ... print(a, end=' ') ... a, b = b, a+b ... 0,1,1,2,3,5,8,13,21,34,55,89,144,233,377,610,987,
```

20.5.6 continue

```
>>> for num in range(2, 10): ... if num % 2 == 0: ... print("Found an even nu
```

20.6 Functions

```
>>> def fib(n): # write Fibonacci series up to n ... """Print a Fibonacci series up to n.""" ... a, b = 0, 1 ... while a < n: ... print(a, end=' ') ... a,
```

20.7 Default Argument Values

```
def ask_ok(prompt, retries=4, reminder='Please try again!'): while True: ok = input(prompt) if ok in ('y', 'ye', 'yes'): return True if ok in ('n', 'no', 'n
```

This function can be called in several ways:

1. By giving only the **mandatory** argument:

```
ask_ok('Do you really want to quit?')
```

2. By giving one of the **optional** arguments:

```
ask_ok('OK to overwrite the file?', 2)
```

3. By giving **all** arguments:

```
ask_ok('OK to overwrite the file?', 2, 'Come on, only yes or no!')
```

20.8 Further Reading

- <https://docs.python.org/3/tutorial/introduction.html>

21. Example: Custom Keywords Library

1. Create a python file, e.g., `reconnaissance.py`

2. Add python functions

```
import requests from bs4 import BeautifulSoup from tldextract import extract import re try: from urlparse import urljoin, urlparse # Python2 except ImportError:
```

3. Import the library in your `.robot` file

```
3b-jenkins-pipeline > Input > robotframework > demo-test-suite > Resources > PageObjects > Reconnaissance.robot
1 *** Settings ***
2 Library      SeleniumLibrary
3 Library      ../../CustomLibraries/reconnaissance.py
4
5 *** Keywords ***
6 Read Source Code
7   [Arguments]  ${Target_URL}
8   ${current_url} =  Get Location
9   ${status} =  Run Keyword And Return Status    Should Be Equal    ${Target_URL}
10  IF  ${status} != ${True}
11  |  Go To    ${Target_URL}
12  END
13  ${source_code} =  Get Source
14  Return From Keyword  ${source_code}
15
16 Google search
17   [Arguments]  ${Target_Domain}
18   Go To    https://www.google.com/
19   ${search_box_element} =  Set Variable  xpath://input[@role="combobox"]
20   Input Text  ${search_box_element}    site:*.${Target_Domain}
21   Press Keys  ${search_box_element}  ENTER
22   Wait Until Page Contains  results
23   Capture Element Screenshot  xpath://body
24
```

4. Call custom keywords

```

181  Get list of links discovered but out of scope
182      ${not_in_scope_links} =  Extract Links Not In Scope  ${ALL_DISCOVERED_LINKS}  ${IN_SCOPE_DOMAIN}
183      Sort List  ${not_in_scope_links}
184      @{OUT_OF_SCOPE_LINKS} =  Remove Duplicates  ${not_in_scope_links}
185      Set Global Variable  ${OUT_OF_SCOPE_LINKS}
186      Log  ${OUT_OF_SCOPE_LINKS}
187
188  Get list of links discovered but not yet visited
189      ${not_visited_links} =  Retrieve not yet visited links
190      Sort List  ${not_visited_links}
191      @{DISCOVERED_BUT_NOT_VISITED_LINKS} =  Remove Duplicates  ${not_visited_links}
192      Set Global Variable  ${DISCOVERED_BUT_NOT_VISITED_LINKS}
193      Log  ${DISCOVERED_BUT_NOT_VISITED_LINKS}
194
195  Get list of all links discovered that are in-scope
196      Log List  ${ALL_DISCOVERED_LINKS}
197      Sort List  ${ALL_DISCOVERED_LINKS}
198      @{sorted_unique_links} =  Remove Duplicates  ${ALL_DISCOVERED_LINKS}
199      Log List  ${sorted_unique_links}
200      @{in_scope_links} =  Extract Links In Scope  ${sorted_unique_links}  ${IN_SCOPE_DOMAIN}
201      ${ALL_IN_SCOPE_LINKS} =  Set Variable  ${in_scope_links}

```

5. Keyword output will be part of the final report

- **KEYWORD** Crawl.Print crawl status
Start / End / Elapsed: 20210919 13:37:20.256 / 20210919 13:37:20.529 / 00:00:00.273

+ **KEYWORD** Crawl.Collect final statistics

+ **KEYWORD** Collections.Log List \${VISITED_LINKS}

- **KEYWORD** Collections.Log List \${DISCOVERED_BUT_NOT_VISITED_LINKS}
Documentation: Logs the length and contents of the `list` using given `level`.
Start / End / Elapsed: 20210919 13:37:20.526 / 20210919 13:37:20.526 / 00:00:00.000
13:37:20.526 INFO List length is 25 and it contains following items:
0: <https://demo.testfire.net/Privacypolicy.jsp?sec=Careers&template=US>
1: <https://demo.testfire.net/admin/clients.xls>
2: <https://demo.testfire.net/disclaimer.htm?url=http://www.microsoft.com>
3: <https://demo.testfire.net/disclaimer.htm?url=http://www.netscape.com>
4: https://demo.testfire.net/high_yield_investments.htm
5: https://demo.testfire.net/index.jsp?content=inside_benefits.htm
6: https://demo.testfire.net/index.jsp?content=inside_community.htm
7: https://demo.testfire.net/index.jsp?content=inside_contact.htm#ContactUs
8: https://demo.testfire.net/index.jsp?content=inside_executives.htm
9: https://demo.testfire.net/index.jsp?content=inside_internships.htm
10: https://demo.testfire.net/index.jsp?content=inside_jobs.htm
11: https://demo.testfire.net/index.jsp?content=inside_trainee.htm
12: <https://demo.testfire.net/index.jsp?content=pr/20060413.htm>
13: <https://demo.testfire.net/index.jsp?content=pr/20060518.htm>
14: <https://demo.testfire.net/index.jsp?content=pr/20060720.htm>
15: <https://demo.testfire.net/index.jsp?content=pr/20060817.htm>
16: <https://demo.testfire.net/index.jsp?content=pr/20060921.htm>
17: <https://demo.testfire.net/index.jsp?content=pr/20060928.htm>
18: <https://demo.testfire.net/index.jsp?content=pr/20061005.htm>
19: <https://demo.testfire.net/index.jsp?content=pr/20061109.htm>
20: <https://demo.testfire.net/index.jsp?content=security.htm#top>
21: https://demo.testfire.net/inside_points_of_interest.htm
22: <https://demo.testfire.net/my%20documents/JohnSmith/Bank%20Site%20Documents/grouplife.htm>
23: <https://demo.testfire.net/pr/communityannualreport.pdf>
24: https://demo.testfire.net/survey_questions.jsp?step=a

+ **KEYWORD** Collections.Log List \${OUT_OF_SCOPE_LINKS}

+ **KEYWORD** Collections.Log List \${ALL_IN_SCOPE_LINKS}

+ **KEYWORD** Collections.Log List \${ALL_DISCOVERED_FORMS}

+ **KEYWORD** Collections.Log List \${ALL_FORMS_ATTRIBUTES_AND_INPUT_ELEMENTS}

22. When and How to Use Burp Suite

1. General Setup and Tool Overview
2. Attacking Web Applications using Burp Suite
3. Specific Attack Scenarios

22.1 Corresponding VM download links

- [bsides-workshop-v0.2.0.ova](#)
- [c0c0n-training.ova](#)

23. DVWA: Get Your Hands Dirty

Damn Vulnerable Web App (DVWA) is a PHP/MySQL web application that is damn vulnerable. Let's solve [DVWA](#).

23.1 Pre-requisites

- Add following line to **/etc/hosts** file: `127.0.1.1 secqation.local`

23.2 Purely Manual Approach

1. Start your own vulnerable instance of DVWA

```
docker run --rm -it -p 80:80 vulnerables/web-dvwa
```

2. Go to <http://secqation/login.php>



The DVWA login form is displayed. It features two input fields: one for "Username" and one for "Password", both with placeholder text. Below the password field is a "Login" button.

Username	<input type="text"/>
Password	<input type="password"/>
<input type="button" value="Login"/>	

3. Start hacking.

23.3 SecQAtion Approach

Let's make hacking DVWA a bit more interesting...

1. Go to `/home/secqation/Desktop/TheDemo/1-Local-execution/demo-test-suite` folder path
2. Check the contents of **init.sh** file in **BashScripts** folder

```
mitmdump -p 8080 -w +traffic.mitm "! ~u firefox|ocsp|mozilla|googleapis" & robot -d ~/Desktop/TheDemo/1-local-execution/demo-test-suite/Results ~/Des
```

Notice the use of `${1:-App.robot}`. It says that if no parameter value is passed when calling the bash script, then **App.robot** test case should be executed by default. In case a specific filename is passed, then that specific test suite should be executed.

3. Check the contents of **Tests** folder

```
$ tree Tests/ Tests/ 01_setup_dvwa.robot 02_teardown_dvwa.robot 03_recon_base_url.robot App.robot
```

4. Check the contents of `.robot` files

1. **Tests/01_setup_dvwa.robot**

```
*** Settings *** Resource ../Resources/App.robot Resource ../Resources/Common.robot Resource ../Data/Variables/global.robot Suite Setup Begin Web
```

2. **Tests/02_teardown_dvwa.robot**

```
*** Settings *** Resource ../Resources/App.robot Resource ../Resources/Common.robot *** Test Cases *** Destroy DVWA Running Docker Instance Stop
```

3. **Tests/03_recon_base_url.robot**

```
*** Settings *** Resource ../Resources/App.robot Resource ../Resources/Common.robot Resource ../Data/Variables/global.robot Suite Setup Begin Web
```

4. **Tests/App.robot**

```
*** Settings *** Resource ../Resources/App.robot Resource ../Resources/Common.robot Resource ../Data/Variables/global.robot Suite Setup Setup for
```

5. Run following commands to setup DVWA and do a quick scan of the base URL

```
./BashScripts/init.sh 01_setup_dvwa.robot ./BashScripts/init.sh 03_recon_base_url.robot
```

6. Open the **log.html** file (from Results folder) in a browser window

<file:///home/secqation/Desktop/TheDemo/1-local-execution/demo-test-suite/Results/log.html>

Recon Base Url Log

Generated
20210927 07:59:29 UTC+05:30
11 minutes 20 seconds ago

Test Statistics

Total Statistics		Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
All Tests		1	1	0	0	00:00:17	<div style="width: 100%; background-color: #2e7131;"></div>

Statistics by Tag		Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
No Tags							<div style="width: 0%; background-color: #cccccc;"></div>

Statistics by Suite		Total	Pass	Fail	Skip	Elapsed	Pass / Fail / Skip
Recon Base Url		1	1	0	0	00:00:32	<div style="width: 100%; background-color: #2e7131;"></div>

Test Execution Log

- **SUITE** Recon Base Url
 - Full Name: Recon Base Url
 - Source: /home/secqation/Desktop/TheDemo/1-local-execution/demo-test-suite/Tests/03_recon_base_url.robot
 - Start / End / Elapsed: 20210927 07:58:56.506 / 20210927 07:59:28.976 / 00:00:32.470
 - Status: 1 test total, 1 passed, 0 failed, 0 skipped
 - + **SETUP** Common.Begin Web Assessment
 - + **TEARDOWN** Common.End Web Assessment
 - + **TEST** Authenticate and Crawl DVWA

7. Click on **Authenticate and Crawl DVWA**
8. Click on **App.Crawl \${Base_URL}, \${Target_Domain}**
9. Click on **Crawl.Print crawl status**
10. Click on **Collections.Log List \${ALL_IN_SCOPE_LINKS}**

TEST	Authenticate and Crawl DVWA	
Full Name:	Recon Base Url.Authenticate and Crawl DVWA	
Documentation:	Assumptions - 1) DVWA server is up 2) DVWA setup has been completed successfully	
Start / End / Elapsed:	20210927 07:59:10.938 / 20210927 07:59:28.190 / 00:00:17.252	
Status:	PASS	
+ KEYWORD	App.Login to DVWA	
- KEYWORD	App.Crawl \${BASE_URL}, \${Target_Domain}	
Start / End / Elapsed:	20210927 07:59:19.746 / 20210927 07:59:28.190 / 00:00:08.444	
+ KEYWORD	Crawl.Reset global parameters	
+ KEYWORD	Crawl.Set target \${Target_Domain}	
+ KEYWORD	Builtin.Log \${CURRENT_URL}	
+ KEYWORD	Builtin.Log \${CURRENT_PAGE_SOURCE_CODE}	
+ KEYWORD	SeleniumLibrary.Capture Element Screenshot xpath://body	
+ KEYWORD	Crawl.Start crawling \${Target_URL}, \${Target_Domain}	
- KEYWORD	Crawl.Print crawl status	
Start / End / Elapsed:	20210927 07:59:27.541 / 20210927 07:59:28.190 / 00:00:00.649	
+ KEYWORD	Crawl.Collect final statistics	
+ KEYWORD	Collections.Log List \${VISITED_LINKS}	
+ KEYWORD	Collections.Log List \${DISCOVERED_BUT_NOT_VISITED_LINKS}	
+ KEYWORD	Collections.Log List \${OUT_OF_SCOPE_LINKS}	
- KEYWORD	Collections.Log List \${ALL_IN_SCOPE_LINKS}	
Documentation:	Logs the length and contents of the <code>list</code> using given <code>level</code> .	
Start / End / Elapsed:	20210927 07:59:28.186 / 20210927 07:59:28.186 / 00:00:00.000	
07:59:28.186	INFO	List length is 21 and it contains following items: 0: http://secqation.local/ 1: http://secqation.local/about.php 2: http://secqation.local/instructions.php 3: http://secqation.local/logout.php 4: http://secqation.local/phpinfo.php 5: http://secqation.local/security.php 6: http://secqation.local/setup.php 7: http://secqation.local/vulnerabilities/brute/ 8: http://secqation.local/vulnerabilities/captcha/ 9: http://secqation.local/vulnerabilities/csp/ 10: http://secqation.local/vulnerabilities/csrf/ 11: http://secqation.local/vulnerabilities/exec/ 12: http://secqation.local/vulnerabilities/fi/?page=include.php 13: http://secqation.local/vulnerabilities/javascript/ 14: http://secqation.local/vulnerabilities/sql/ 15: http://secqation.local/vulnerabilities/sql_blind/ 16: http://secqation.local/vulnerabilities/upload/ 17: http://secqation.local/vulnerabilities/weak_id/ 18: http://secqation.local/vulnerabilities/xss_d/ 19: http://secqation.local/vulnerabilities/xss_r/ 20: http://secqation.local/vulnerabilities/xss_s/

11. Copy the list of all in-scope URLs

12. Open **Tests/App.robot** test file, and pass this copied list of URLs as an argument to the **Perform Authenticated Crawling** test template

```

*** Settings ***
Resource      ./Resources/App.robot
Resource      ./Resources/Common.robot
Resource      ./Data/Variables/global.robot
Suite Setup    Setup for Authenticated Crawling of DVWA Pages
Suite Teardown End Web Assessment

*** Test Cases ***
Crawl Authenticated DVWA Pages
[Documentation] Assumptions -
...
1) DVWA server is up
...
2) DVWA setup has been completed successfully
...
3) User is logged in
[Template] Perform Authenticated Crawling
http://secqation.local/
http://secqation.local/about.php
http://secqation.local/instructions.php
http://secqation.local/phpinfo.php
http://secqation.local/security.php
http://secqation.local/setup.php
http://secqation.local/vulnerabilities/brute/
http://secqation.local/vulnerabilities/captcha/
http://secqation.local/vulnerabilities/csp/

```

13. Switch to command line interface and run following command

```
$ cd ~/Desktop/TheDemo/1-local-execution/demo-test-suite $ ./BashScripts/init.sh
```

14. Refresh **log.html** page in your browser to see updated results

← → ⌂ file:///home/secqation/Desktop/TheDemo/1-local-execution/demo-test-suite/Results/log.html

TEST Crawl Authenticated DVWA Pages

Full Name: App.Crawl Authenticated DVWA Pages
Documentation: Assumptions - 1) DVWA server is up 2) DVWA setup has been completed successfully 3) User is logged in
Start / End / Elapsed: 20210927 08:28:02.791 / 20210927 08:28:20.522 / 00:00:17.731
Status: PASS

- + **KEYWORD** App.Perform Authenticated Crawling http://secqation.local/
- + **KEYWORD** App.Perform Authenticated Crawling http://secqation.local/about.php
- + **KEYWORD** App.Perform Authenticated Crawling http://secqation.local/instructions.php
- + **KEYWORD** App.Perform Authenticated Crawling http://secqation.local/phpinfo.php
- + **KEYWORD** App.Perform Authenticated Crawling http://secqation.local/security.php
- + **KEYWORD** App.Perform Authenticated Crawling http://secqation.local/setup.php
- + **KEYWORD** App.Perform Authenticated Crawling http://secqation.local/vulnerabilities/brute/
- + **KEYWORD** App.Perform Authenticated Crawling http://secqation.local/vulnerabilities/captcha/
- + **KEYWORD** App.Perform Authenticated Crawling http://secqation.local/vulnerabilities/csp/
- + **KEYWORD** App.Perform Authenticated Crawling http://secqation.local/vulnerabilities/csrf/
- + **KEYWORD** App.Perform Authenticated Crawling http://secqation.local/vulnerabilities/exec/
- + **KEYWORD** App.Perform Authenticated Crawling http://secqation.local/vulnerabilities/fi/?page=include.php
- + **KEYWORD** App.Perform Authenticated Crawling http://secqation.local/vulnerabilities/javascript/
- + **KEYWORD** App.Perform Authenticated Crawling http://secqation.local/vulnerabilities/sql/
- + **KEYWORD** App.Perform Authenticated Crawling http://secqation.local/vulnerabilities/sql_blind/
- + **KEYWORD** App.Perform Authenticated Crawling http://secqation.local/vulnerabilities/upload/
- + **KEYWORD** App.Perform Authenticated Crawling http://secqation.local/vulnerabilities/weak_id/
- + **KEYWORD** App.Perform Authenticated Crawling http://secqation.local/vulnerabilities/xss_d/
- + **KEYWORD** App.Perform Authenticated Crawling http://secqation.local/vulnerabilities/xss_r/
- **KEYWORD** App.Perform Authenticated Crawling http://secqation.local/vulnerabilities/xss_s/

Start / End / Elapsed: 20210927 08:28:19.580 / 20210927 08:28:20.522 / 00:00:00.942

- + **KEYWORD** App.Begin Data Processing \${Target_URL}
- + **KEYWORD** App.Print current page statistics
- **KEYWORD** Crawl.Print crawl status

Start / End / Elapsed: 20210927 08:28:19.910 / 20210927 08:28:20.522 / 00:00:00.612

- + **KEYWORD** Crawl.Collect final statistics
- + **KEYWORD** Collections.Log List \${VISITED_LINKS}
- + **KEYWORD** Collections.Log List \${DISCOVERED_BUT_NOT_VISITED_LINKS}
- + **KEYWORD** Collections.Log List \${OUT_OF_SCOPE_LINKS}
- + **KEYWORD** Collections.Log List \${ALL_IN_SCOPE_LINKS}
- + **KEYWORD** Collections.Log List \${ALL_DISCOVERED_FORMS}
- + **KEYWORD** Collections.Log List \${ALL_FORMS_ATTRIBUTES_AND_INPUT_ELEMENTS}

15. Expand respective keywords to see detailed scan report of various URLs that were passed to the test template **Perform Authenticated Crawling**

```

+ [KEYWORD] App.Perform Authenticated Crawling http://secqation.local/vulnerabilities/xss_r/
- [KEYWORD] App.Perform Authenticated Crawling http://secqation.local/vulnerabilities/xss_s/
  Start / End / Elapsed: 20210927 08:28:19.580 / 20210927 08:28:20.522 / 00:00:00.942
  + [KEYWORD] App.Begin Data Processing ${Target_URL}
  - [KEYWORD] App.Print current page statistics
    Start / End / Elapsed: 20210927 08:28:19.793 / 20210927 08:28:19.910 / 00:00:00.117
    - [KEYWORD] BuiltIn.Log ${CURRENT_URL}
      Documentation: Logs the given message with the given level.
      Start / End / Elapsed: 20210927 08:28:19.794 / 20210927 08:28:19.794 / 00:00:00.000
      08:28:19.794 INFO http://secqation.local/vulnerabilities/xss_s/
    + [KEYWORD] BuiltIn.Log ${CURRENT_PAGE_SOURCE_CODE}
    + [KEYWORD] Collections.Log List ${ALL_DISCOVERED_LINKS}
    - [KEYWORD] SeleniumLibrary.Capture Element Screenshot xpath://body
      Documentation: Captures a screenshot from the element identified by locator and embeds it into log file.
      Start / End / Elapsed: 20210927 08:28:19.796 / 20210927 08:28:19.909 / 00:00:00.113
      08:28:19.909 INFO

```



```

- [KEYWORD] Crawl.Print crawl status
  Start / End / Elapsed: 20210927 08:28:19.910 / 20210927 08:28:20.522 / 00:00:00.612
  + [KEYWORD] Crawl.Collect final statistics
  + [KEYWORD] Collections.Log List ${VISITED_LINKS}
  + [KEYWORD] Collections.Log List ${DISCOVERED_BUT_NOT_VISITED_LINKS}
  + [KEYWORD] Collections.Log List ${OUT_OF_SCOPE_LINKS}
  + [KEYWORD] Collections.Log List ${ALL_IN_SCOPE_LINKS}
  + [KEYWORD] Collections.Log List ${ALL_DISCOVERED_FORMS}
  + [KEYWORD] Collections.Log List ${ALL_FORMS_ATTRIBUTES_AND_INPUT_ELEMENTS}

```

16. Locate `mitmproxy_httprobe.py` file

```
$ sudo apt install mlocate $ locate mitmproxy_httprobe.py
```

17. Check the contents of `docker-compose-httprobe.yml` file. Ensure that the **volume mappings** are correct.

```
version: "3.9" services: HTTPPolice: build: context: . dockerfile: ./Input/HTTPPolice/Dockerfile image: httppolice:latest container_name: httppolice volume
```

18. Run following command

```
docker-compose -f docker-compose-httppolice.yml run HTTPolice
```

19. Press **[F]** key to set a filter expression

20. Enter **'! (~q | ~t css|javascript|icon)'** as the filter expression. This will remove any request without a response or any request that has its content type set as css, javascript or icon.

```
secqation@mirage: ~/Desktop/TheDemo/1-local-execution/demo-test-suite
secqation@mirage: ~/Desktop/TheDemo/1-local-execution/demo-test-suite

Flows
02:58:03 HTTP GET secqation.local /about.php 200 text/html 2.06k 10ms
02:58:03 HTTP GET secqation.local /instructions.php 200 text/html 4.98k 15ms
02:58:04 HTTP GET secqation.local /phpinfo.php 200 text/html ...31k 59ms
02:58:07 HTTP GET secqation.local /security.php 200 text/html 2.05k 7ms
02:58:07 HTTP GET secqation.local /dvwa/images/lock.png 200 image/png 761b 8ms
02:58:07 HTTP GET secqation.local /setup.php 200 text/html 1.92k 11ms
02:58:07 HTTP GET secqation.local /dvwa/images/spanner.png 200 image/png 464b 19ms
02:58:08 HTTP GET secqation.local /vulnerabilities/brute/ 200 text/html 1.4k 9ms
02:58:09 HTTP GET secqation.local /vulnerabilities/captcha/ 200 text/html 1.52k 8ms
02:58:10 HTTP GET secqation.local /vulnerabilities/csp/ 200 text/html 1.44k 9ms
02:58:11 HTTP GET secqation.local /vulnerabilities/csrf/ 200 text/html 1.37k 10ms
02:58:12 HTTP GET secqation.local /vulnerabilities/exec/ 200 text/html 1.33k 12ms
02:58:12 HTTP GET secqation.local /vulnerabilities/fi/?page=includ... 200 text/html 1.28k 11ms
02:58:13 HTTP GET secqation.local /vulnerabilities/javascript/ 200 text/html 3.19k 10ms
02:58:14 HTTP GET secqation.local /vulnerabilities/sqli/ 200 text/html 1.39k 12ms
02:58:15 HTTP GET secqation.local /vulnerabilities/sqli_blind/ 200 text/html 1.39k 12ms
02:58:16 HTTP GET secqation.local /vulnerabilities/upload/ 200 text/html 1.36k 8ms
02:58:16 HTTP GET secqation.local /vulnerabilities/weak_id/ 200 text/html 1.15k 11ms
02:58:17 HTTP GET secqation.local /vulnerabilities/xss_d/ 200 text/html 1.52k 10ms
02:58:18 HTTP GET secqation.local /vulnerabilities/xss_r/ 200 text/html 1.35k 14ms
>>02:58:19 HTTP GET secqation.local /vulnerabilities/xss_s/ 200 text/html 1.59k 10ms

↓ [110/110][f:! (~q | ~t
css|javascript|icon)][scripts:1]
: set view_filter '! (~q | ~t css|javascript|icon)'
```

21. Run following command to export the HTML report for filtered traffic

```
: httpolice.report.html @shown /home/mitmproxy/Output/httpolice/httpolice_report.html
```

22. Press **[Q]** key followed by **[Y]** key to exit mitmproxy

23. Access the exported HTML report from path **[./Results/httpolice/httpolice_report.html](#)**

The screenshot shows a browser window titled "HTTPolice report". The address bar contains "file:///home/secqation/Desktop/TheDemo/1-local-execution/demo-test-suite/Results/httppolice/httppolice_report.html". The main content area displays an HTTP request and its response.

```

GET /login.php HTTP/1.1
Host: secqation.local
User-Agent: Mozilla/5.0 (X11; Ubuntu; Linux x86_64; rv:92.0) Gecko/20100101 Firefox/92.0
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,*/*;q=0.8
Accept-Language: en-US,en;q=0.5
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1

HTTP/1.1 StatusCode(200) OK
Date: Wed, 15 Sep 2021 18:05:05 GMT
Server: Apache/2.4.25 (Debian)
Set-Cookie: PHPSESSID=tejpuhn34g0p5n608eoccjbn5; path=/
Expires: Tue, 23 Jun 2009 12:00:00 GMT
Cache-Control: no-cache, must-revalidate
Pragma: no-cache
Set-Cookie: PHPSESSID=tejpuhn34g0p5n608eoccjbn5; path=/
Set-Cookie: security=low
Vary: Accept-Encoding
Content-Encoding: gzip
Content-Length: 699
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html;charset=utf-8

Body after removing Content-Encoding and taking the first 1000 characters
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN" "http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd>

```

A callout box highlights the "Pragma: no-cache" header with the identifier "C 1162 Pragma: no-cache is for requests". It includes a link to "explain" and a note: "This response contains the 'no-cache' pragma directive, which is defined by RFC 7234 § 5.4 only for requests."

24. If you find an interesting request, copy and paste it in Burp Suite's Repeater/ Intruder tool

25. Ask doubts, if any.

Hope you have learned something interesting.

Happy Hacking... the SecQAtion way!!

23.4 References:

- <https://github.com/digininja/DVWA>

