

Configuration Manual

MSc Research Project
Cyber Security

Hardik Solanki
Student ID: x21117659

School of Computing
National College of Ireland

Supervisor: Prof. Vikas Sahni

National College of Ireland
MSc Project Submission Sheet
School of Computing




Student Name: Hardik Solanki
Student ID: x21117659
Programme: MSc. In Cyber Security **Year:** 2022-2023
Module: MSc Research Project
Lecturer: Prof. Vikas Sahni
Submission Due Date: 6th January 2023
Project Title: Limiting Attack Surface for Infrastructure Applications using Custom YAML Templates in Nuclei Automation
Word Count: 1378 **Page Count:** 20

I hereby certify that the information contained in this (my submission) is information pertaining to research I conducted for this project. All information other than my own contribution will be fully referenced and listed in the relevant bibliography section at the rear of the project.

ALL internet material must be referenced in the bibliography section. Students are required to use the Referencing Standard specified in the report template. To use other author's written or electronic work is illegal (plagiarism) and may result in disciplinary action.

Signature:



Date: 6th January, 2023

PLEASE READ THE FOLLOWING INSTRUCTIONS AND CHECKLIST

Attach a completed copy of this sheet to each project (including multiple copies)	<input type="checkbox"/>
Attach a Moodle submission receipt of the online project submission, to each project (including multiple copies).	<input type="checkbox"/>
You must ensure that you retain a HARD COPY of the project, both for your own reference and in case a project is lost or mislaid. It is not sufficient to keep a copy on computer.	<input type="checkbox"/>

Assignments that are submitted to the Programme Coordinator Office must be placed into the assignment box located outside the office.

Office Use Only	
Signature:	
Date:	
Penalty Applied (if applicable):	

Configuration Manual

Hardik Solanki
Student ID: x21117659

1 Introduction

The purpose of this manual is to provide an overview of the workflow for the Nuclei Automation in Infrastructure research project. In addition, it provides procedures for setting up the environment and integrating the YAML template into Nuclei GitHub.

In order to be successful, the project must be downloaded, installed, and configured properly. The stages below describe the installation and integration process.

2 Environmental Setup

Configurations	Version
Kali Linux (Linux Operating System) ¹	2022.3
Golang (Go Programming Language) ²	go1.19.4 linux/amd64

Table 1: Environmental Setup

```
(kali@kali)-[~]
└─$ lsb_release -a
No LSB modules are available.
Distributor ID: Kali
Description:    Kali GNU/Linux Rolling
Release:       2022.3
Codename:      kali-rolling
```

Figure 1: Installed latest version of Linux

3 Nuclei Automation Installation Guide & Usage in Linux

- **Golang:** It is necessary to install the latest GO version in order for Nuclei to work properly.
 - **Golang installation command:** `sudo apt-get install -y golang`

¹ <https://www.kali.org/docs/installation/hard-disk-install/>

² <https://go.dev/doc/install>

```
(kali@kali)-[~]
└─$ sudo apt-get install -y golang
[sudo] password for kali:
Sorry, try again.
[sudo] password for kali:
Reading package lists... Done
Building dependency tree... Done
Reading state information... Done
The following packages were automatically installed and are no longer require
d:
  cpp-11 gcc-11-base
Use 'sudo apt autoremove' to remove them.
The following additional packages will be installed:
  binutils binutils-common binutils-x86-64-linux-gnu cpp cpp-12 g++ g++-12
  gcc gcc-12 gcc-12-base golang-1.19 golang-1.19-doc golang-1.19-go
  golang-1.19-src golang-doc golang-go golang-src libasan8 libatomic1
  libbinutils libc-bin libc-dev-bin libc-dev-tools libc-l10n libc6 libc6-dev
  libcc1-0 libcrypt-dev libcrypt1 libctf-nobfd0 libctf0 libgcc-12-dev
  libgcc-s1 libgfortran5 libgomp1 libgprofng0 libitm1 liblsan0 libnsl-dev
  libpkgconf3 libquadmath0 libstdc++-12-dev libstdc++6 libtirpc-common
  libtirpc-dev libtirpc3 libtsan2 libubsan1 linux-image-6.0.0-kali6-amd64
  linux-image-amd64 linux-libc-dev locales manpages manpages-dev pkg-config
  pkgconf pkgconf-bin rpcsvc-proto
Suggested packages:
  binutils-doc cpp-doc gcc-12-locales cpp-12-doc g++-multilib
  g++-12-multilib gcc-12-doc gcc-multilib make autoconf automake libtool
  flex bison gdb gcc-doc gcc-12-multilib bzip2 | brz mercurial subversion
  glibc-doc libnss-nis libnss-nisplus libstdc++-12-doc linux-doc-6.0
  debian-keyring-handbook
```

Figure 2: Installing Golang in Linux

```
(kali@kali)-[~]
└─$ go version
go version go1.19.4 linux/amd64
```

Figure 3: Installed latest version of Golang

- **Nuclei Automation Tool Installation Command:**
 - go install -v github.com/projectdiscovery/nuclei/v2/cmd/nuclei@latest

```
(kali@kali)-[~]
└─$ go install -v github.com/projectdiscovery/nuclei/v2/cmd/nuclei@latest
go: downloading github.com/projectdiscovery/nuclei v1.1.7
go: downloading github.com/projectdiscovery/nuclei/v2 v2.8.3
go: downloading github.com/projectdiscovery/gologger v1.1.5
go: downloading github.com/projectdiscovery/goflags v0.1.5
go: downloading github.com/projectdiscovery/interactsh v1.0.6-0.20220827132222-460cc6270053
go: downloading github.com/projectdiscovery/utils v0.0.4-0.20221201124851-f8524345b6d3
go: downloading github.com/alecthomas/chroma v0.10.0
go: downloading github.com/apex/log v1.9.0
go: downloading github.com/blang/semver v3.5.1+incompatible
go: downloading github.com/corpix/uarand v0.2.0
go: downloading github.com/go-playground/validator/v10 v10.11.1
go: downloading github.com/google/go-github v17.0.0+incompatible
go: downloading github.com/klauspost/compress v1.15.12
go: downloading github.com/logrusorgru/aurora v2.0.3+incompatible
go: downloading github.com/olekukonko/tablewriter v0.0.5
go: downloading github.com/pkg/errors v0.9.1
go: downloading github.com/projectdiscovery/hmap v0.0.2
go: downloading github.com/projectdiscovery/ratelimit v0.0.2
go: downloading github.com/projectdiscovery/retryablehttp-go v1.0.5-0.20221202084821-c1a692a64751
go: downloading github.com/projectdiscovery/stringsutil v0.0.2
go: downloading github.com/remeh/sizedwaitgroup v1.0.0
go: downloading github.com/tj/go-update v2.2.5-0.20200519121640-62b4b798fd68+incompatible
go: downloading go.uber.org/atomic v1.10.0
```

Figure 4: Nuclei Installation

```

(kali@kali)-[~]
└─$ nuclei --version

nuclei v2.8.3
projectdiscovery.io

[INF] Current Version: 2.8.3

```

Figure 5: Nuclei Tool latest version

- **Nuclei Automation Tool Usage:**

- Following exhibits shows the Nuclei Automation tool usage with all respective listed commands:

```

(kali@kali)-[~/Desktop/Nuclei Automation]
└─$ nuclei -help
Nuclei is a fast, template based vulnerability scanner focusing
on extensive configurability, massive extensibility and ease of use.

Usage:
nuclei [flags]

Flags:
TARGET:
  -u, -target string[]      target URLs/hosts to scan
  -l, -list string         path to file containing a list of target URLs/hosts to scan (one per line)
  -resume string           resume scan using resume.cfg (clustering will be disabled)
  -sa, -scan-all-ips      scan all the IP's associated with dns record
  -iv, -ip-version string[] IP version to scan of hostname (4,6) - (default 4)

TEMPLATES:
  -nt, -new-templates      run only new templates added in latest nuclei-templates release
  -ntv, -new-templates-version string[] run new templates added in specific version
  -as, -automatic-scan     automatic web scan using wappalyzer technology detection to tags mapping
  -t, -templates string[]  list of template or template directory to run (comma-separated, file)
  -tu, -template-url string[] list of template urls to run (comma-separated, file)
  -w, -workflows string[]  list of workflow or workflow directory to run (comma-separated, file)
  -wu, -workflow-url string[] list of workflow urls to run (comma-separated, file)
  -validate                validate the passed templates to nuclei
  -nss, -no-strict-syntax  disable strict syntax check on templates
  -td, -template-display   displays the templates content
  -tl                      list all available templates

FILTERING:
  -a, -author string[]      templates to run based on authors (comma-separated, file)
  -tags string[]           templates to run based on tags (comma-separated, file)
  -etags, -exclude-tags string[] templates to exclude based on tags (comma-separated, file)
  -itags, -include-tags string[] tags to be executed even if they are excluded either by default or configuration
  -id, -template-id string[] templates to run based on template ids (comma-separated, file)
  -eid, -exclude-id string[] templates to exclude based on template ids (comma-separated, file)
  -it, -include-templates string[] templates to be executed even if they are excluded either by default or configuration
  -et, -exclude-templates string[] template or template directory to exclude (comma-separated, file)
  -em, -exclude-matchers string[] template matchers to exclude in result

```

Figure 6: Nuclei Command Usage

```

-s, --severity value[] templates to run based on severity. Possible values: info, low, medium, high, critical, unknown
-es, --exclude-severity value[] templates to exclude based on severity. Possible values: info, low, medium, high, critical, unknown
-pt, --type value[] templates to run based on protocol type. Possible values: dns, file, http, headless, network, workflow, ssl, websocket, whois
-ept, --exclude-type value[] templates to exclude based on protocol type. Possible values: dns, file, http, headless, network, workflow, ssl, websocket, whois
5
-tc, --template-condition string[] templates to run based on expression condition

OUTPUT:
-o, --output string output file to write found issues/vulnerabilities
-resp, --store-resp store all request/response passed through nuclei to output directory
-srd, --store-resp-dir string store all request/response passed through nuclei to custom directory (default "output")
-silent display findings only
-nc, --no-color disable output content coloring (ANSI escape codes)
-json write output in JSONL(ines) format
-rrr, --include-rr include request/response pairs in the JSONL output (for findings only)
-mm, --no-meta disable printing result metadata in cli output
-ts, --timestamp enables printing timestamp in cli output
-rdb, --report-db string nuclei reporting database (always use this to persist report data)
-ms, --matcher-status display match failure status
-me, --markdown-export string directory to export results in markdown format
-se, --sarif-export string file to export results in SARIF format

CONFIGURATIONS:
-config string path to the nuclei configuration file
-fr, --follow-redirects enable following redirects for http templates
-fhr, --follow-host-redirects follow redirects on the same host
-mr, --max-redirects int max number of redirects to follow for http templates (default 10)
-dr, --disable-redirects disable redirects for http templates
-rc, --report-config string nuclei reporting module configuration file
-H, --header string[] custom header/cookie to include in all http request in header:value format (cli, file)
-V, --var value custom vars in key-value format
-r, --resolvers string file containing resolver list for nuclei
-sr, --system-resolvers use system DNS resolving as error fallback
-dc, --disable-clustering disable clustering of requests
-passive enable passive HTTP response processing mode
-fh2, --force-http2 force http2 connection on requests
-ev, --env-vars enable environment variables to be used in template
-cc, --client-cert string client certificate file (PEM-encoded) used for authenticating against scanned hosts
-ck, --client-key string client key file (PEM-encoded) used for authenticating against scanned hosts

```

Figure 6.1: Nuclei Command Usage

```

-ca, --client-ca string client certificate authority file (PEM-encoded) used for authenticating against scanned hosts
-sml, --show-match-line show match lines for file templates, works with extractors only
-ztls use ztls library with autofallback to standard one for tls13
-smi string tls smi hostname to use (default: input domain name)
-sandbox sandbox nuclei for safe templates execution
-i, --interface string network interface to use for network scan
-at, --attack-type string type of payload combinations to perform (batteringram,pitchfork,clusterbomb)
-sip, --source-ip string source ip address to use for network scan
-config-directory string override the default config path ($HOME/.config)
-rsr, --response-size-read int max response size to read in bytes (default 10485760)
-rss, --response-size-save int max response size to read in bytes (default 1048576)

INTERACTSH:
-i-server, --interactsh-server string interactsh server url for self-hosted instance (default: oast.pro,oast.live,oast.site,oast.online,oast.fun,oast.me)
-itoken, --interactsh-token string authentication token for self-hosted interactsh server
-interactions-cache-size int number of requests to keep in the interactions cache (default 5000)
-interactions-eviction int number of seconds to wait before evicting requests from cache (default 60)
-interactions-poll-duration int number of seconds to wait before each interaction poll request (default 5)
-interactions-cooldown-period int extra time for interaction polling before exiting (default 5)
-ni, --no-interactsh disable interactsh server for OAST testing, exclude OAST based templates

UNCOVER:
-uc, --uncover enable uncover engine
-uq, --uncover-query string[] uncover search query
-ue, --uncover-engine string[] uncover search engine (shodan,shodan-1db,fofa,censys,quake,hunter,zoomeye,netlas) (default shodan)
-uf, --uncover-field string uncover fields to return (ip,port,host) (default "ip:port")
-ul, --uncover-limit int uncover results to return (default 100)
-ucd, --uncover-delay int delay between uncover query requests in seconds (0 to disable) (default 1)

RATE-LIMIT:
-rl, --rate-limit int maximum number of requests to send per second (default 150)
-rlm, --rate-limit-minute int maximum number of requests to send per minute
-bs, --bulk-size int maximum number of hosts to be analyzed in parallel per template (default 25)
-c, --concurrency int maximum number of templates to be executed in parallel (default 25)
-hbs, --headless-bulk-size int maximum number of headless hosts to be analyzed in parallel per template (default 10)
-headc, --headless-concurrency int maximum number of headless templates to be executed in parallel (default 10)

```

Figure 6.2: Nuclei Command Usage

```

OPTIMIZATIONS:
-timeout int time to wait in seconds before timeout (default 10)
-retries int number of times to retry a failed request (default 1)
-ldp, --leave-default-ports leave default HTTP/HTTPS ports (eg. host:80,host:443)
-mhe, --max-host-error int max errors for a host before skipping from scan (default 30)
-project use a project folder to avoid sending same request multiple times
-project-path string set a specific project path (default "/tmp")
-spm, --stop-at-first-match stop processing HTTP requests after the first match (may break template/workflow logic)
-stream stream mode - start elaborating without sorting the input
-irt, --input-read-timeout duration timeout on input read (default 3m0s)
-nh, --no-htpdx disable htpdx probing for non-url input
-no-stdin disable stdin processing

HEADLESS:
-headless enable templates that require headless browser support (root user on Linux will disable sandbox)
-page-timeout int seconds to wait for each page in headless mode (default 20)
-sb, --show-browser show the browser on the screen when running templates with headless mode
-sc, --system-chrome use local installed Chrome browser instead of nuclei installed
-lha, --list-headless-action list available headless actions

DEBUG:
-debug show all requests and responses
-dreq, --debug-req show all sent requests
-dresp, --debug-req show all received responses
-p, --proxy string[] list of http/socks5 proxy to use (comma separated or file input)
-pi, --proxy-internal proxy all internal requests
-ldf, --list-dsl-function list all supported DSL function signatures
-tlog, --trace-log string file to write sent requests trace log
-elog, --error-log string file to write sent requests error log
-version show nuclei version
-hm, --hang-monitor enable nuclei hang monitoring
-v, --verbose show verbose output
-profile-mem string optional nuclei memory profile dump file
-vv display templates loaded for scan
-svd, --show-var-dump show variables dump for debugging
-ep, --enable-pprof enable pprof debugging server
-tv, --templates-version shows the version of the installed nuclei-templates
-hc, --health-check run diagnostic check up

```

Figure 6.3: Nuclei Command Usage

```

-vv, -show-var-dump      display templates loaded for scan
-ep, -enable-pprof      show variables dump for debugging
-tv, -templates-version enable pprof debugging server
-hc, -health-check      shows the version of the installed nuclei-templates
                        run diagnostic check up

UPDATE:
-ut, -update-templates  update nuclei-templates to latest released version
-ud, -update-template-dir string custom directory to install / update nuclei-templates
-duc, -disable-update-check disable automatic nuclei/templates update check

STATISTICS:
-stats                 display statistics about the running scan
-sj, -stats-json       write statistics data to an output file in JSONL(ines) format
-si, -stats-interval int number of seconds to wait between showing a statistics update (default 5)
-m, -metrics           expose nuclei metrics on a port
-mp, -metrics-port int port to expose nuclei metrics on (default 9092)

CLOUD:
-cloud                run scan on nuclei cloud
-cs, -cloud-server string nuclei cloud server to use (NUCLEI_CLOUD_SERVER) (default "https://cloud-dev.nuclei.sh")
-ak, -cloud-api-key string api-key for the nuclei cloud server (NUCLEI_CLOUD_APIKEY)
-ls, -list-scan        list previous cloud scans
-ns, -no-store         disable scan/output storage on cloud
-ds, -delete-scan string delete scan/output on cloud by scan id
-so, -scan-output string display scan output by scan id

```

Figure 6.4: Nuclei Command Usage

4 Implementation - Integrating a custom YAML template into Nuclei's GitHub main repository (Publicly)

This section illustrates how YAML have been merged into Nuclei's GitHub main repository (Production/Publicly) using a “Git Pull Request”³.

Step 1: Nuclei Maintainers Team (Project Discovery Parent Company) has its own GitHub main repository, as shown in the exhibit below.

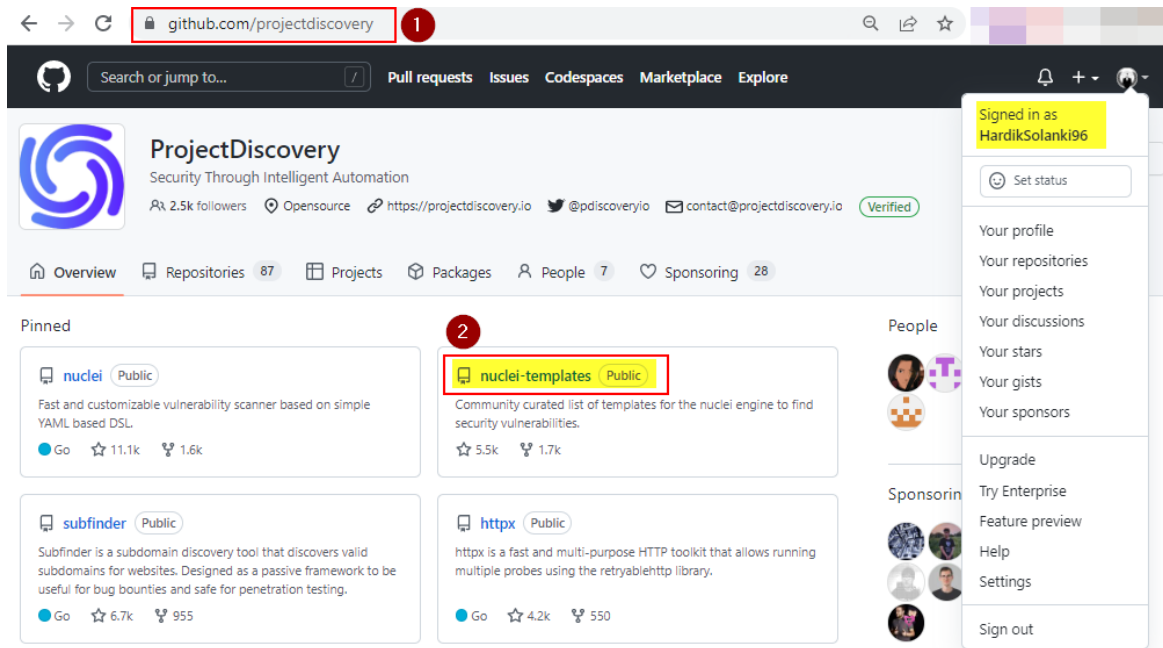


Figure 7: Project Discovery Parent Company of Nuclei Tool

³ <https://github.com/projectdiscovery/nuclei-templates>

Step 2: Click "Fork" and fork into Authors' (HardikSolanki96) GitHub account from Nuclei GitHub main repository, which is from "https://github.com/projectdiscovery/nuclei-templates", as shown in the exhibit below.

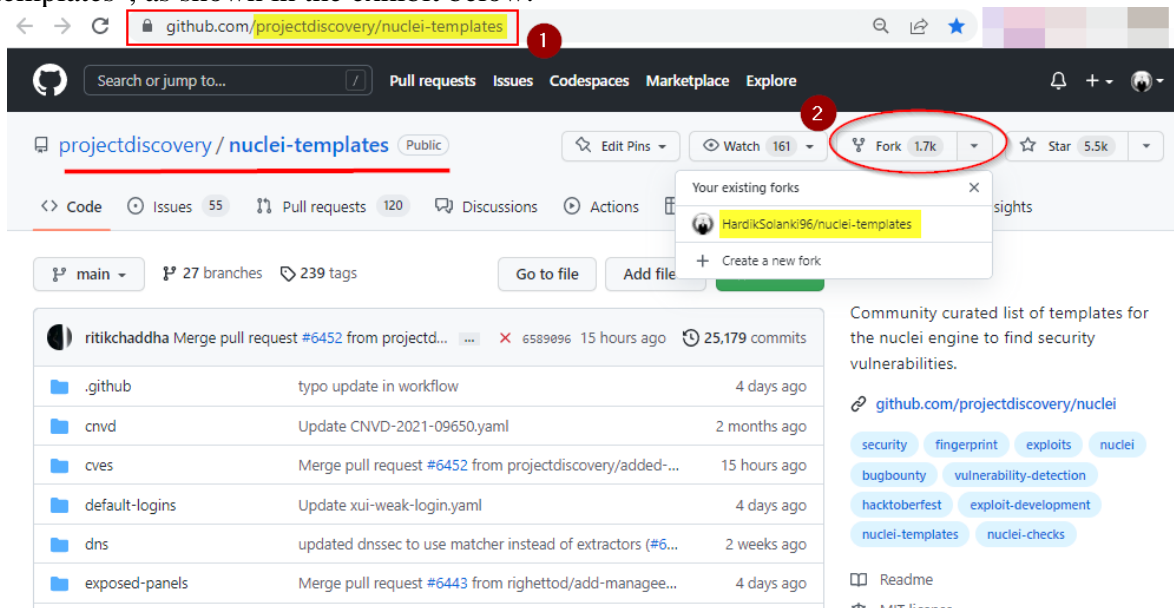


Figure 7.1: Fork the repository

The Nuclei GitHub repository has been forked in the author's GitHub account (HardikSolanki96), from where the author can create a Git Pull request in the Nuclei Main GitHub repository to submit the created custom YAML templates. This is shown in the exhibit below.

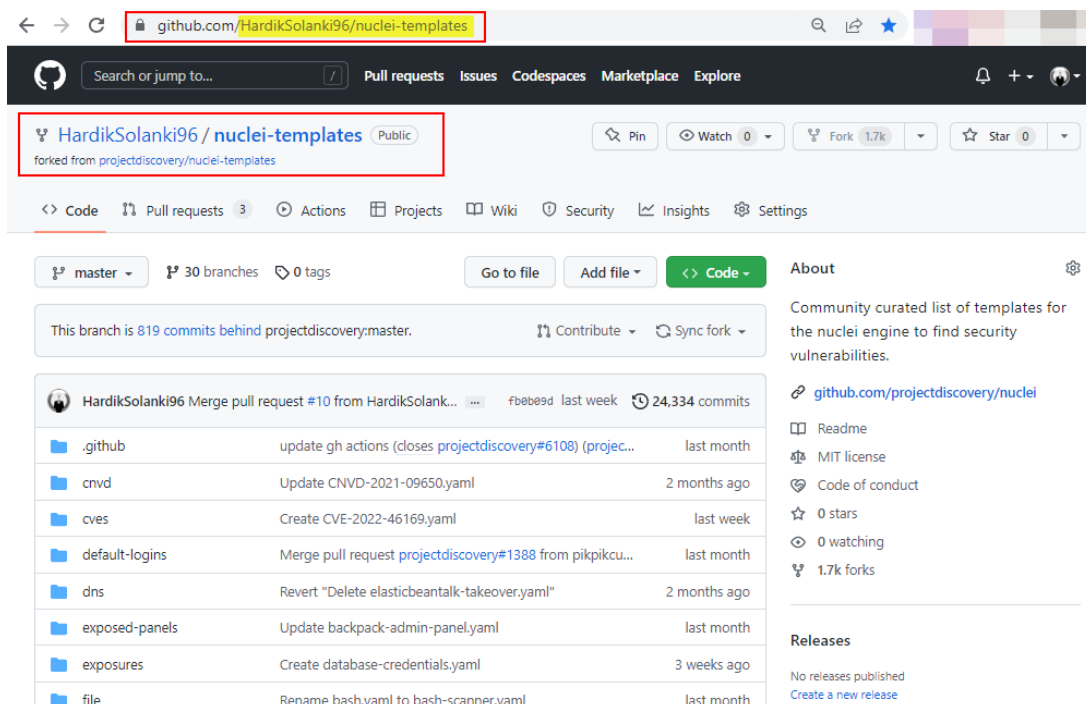


Figure 7.2: Nuclei Repository Forked

Step 3: To submit the created custom YAML Template, navigate to the forked repository in “Authors” account, which is “HardikSolanki96/nuclei-templates” and click on “Add file -> Create new file”, as shown in the below exhibit.

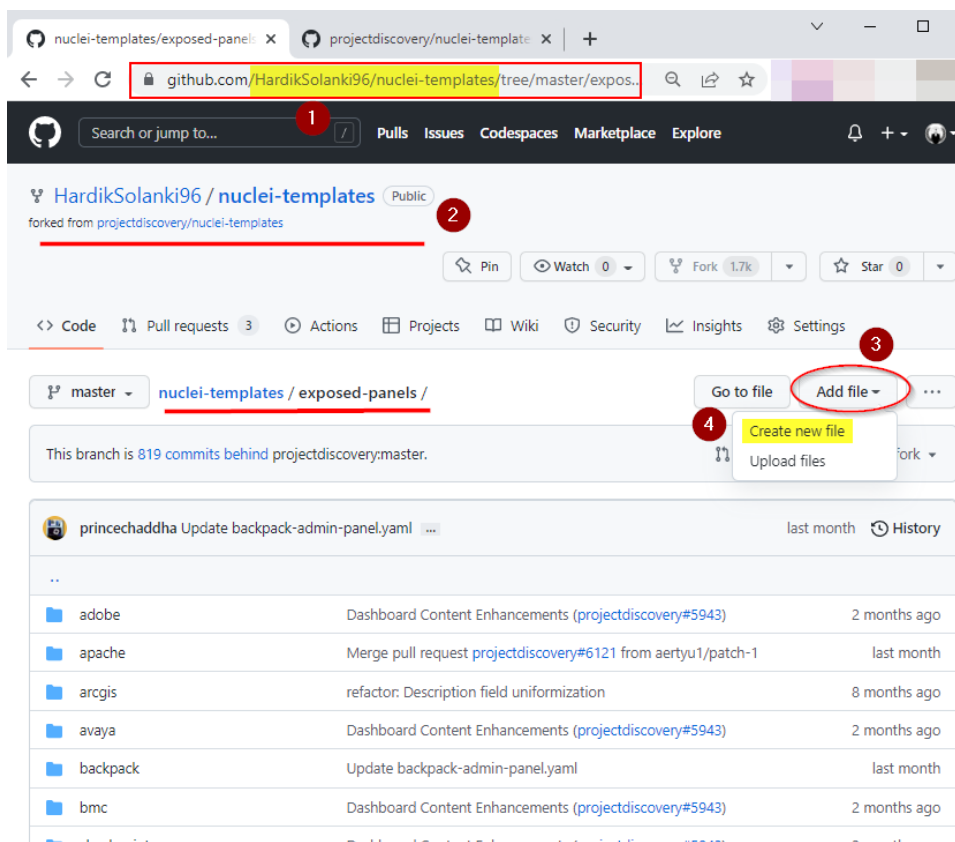


Figure 7.3: Create YAML template

Now, create or paste the custom YAML template which was developed in this research, and name it "mpftvc-admin-panel.yaml" and click on "Propose new file" to create a Git Pull Request in the Nuclei’s Main GitHub Repository. These are shown below in the exhibits.

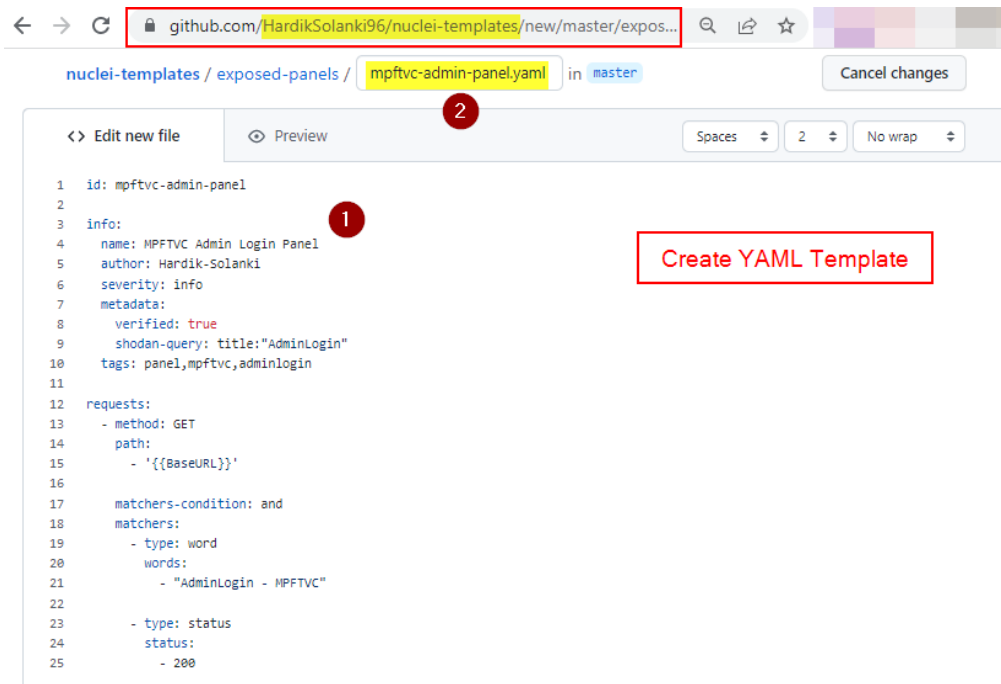


Figure 7.4: Created YAML template

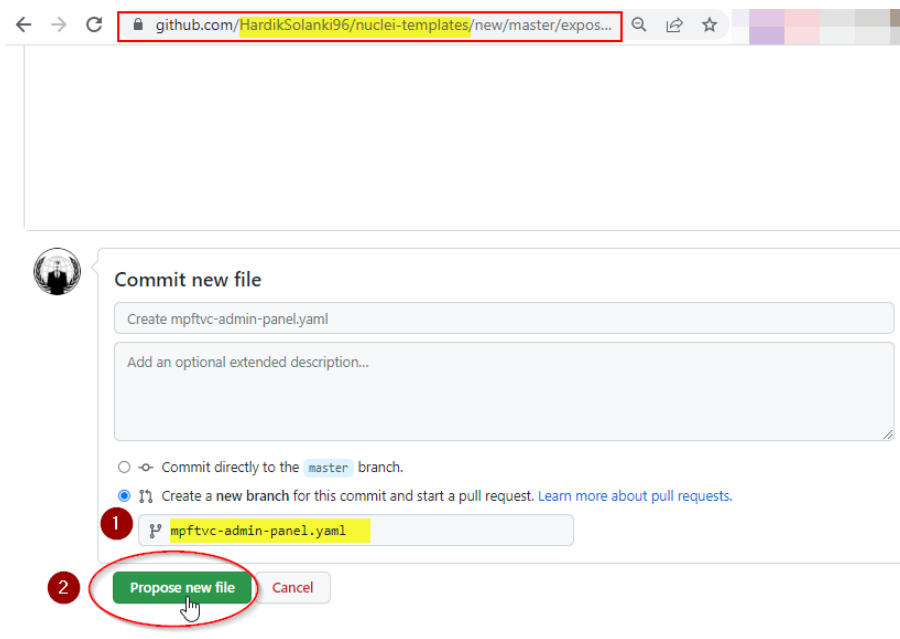


Figure 7.5: Propose YAML template

Step 4: Navigate to the actual Nuclei GitHub main repository i.e., "project discovery/nuclei-templates" from where the YAML template author can access the "Compare & pull request" pop-up to pull the request from Authors forked Nuclei GitHub repository to actual Nuclei GitHub repository by "project discovery" as shown in exhibit below.

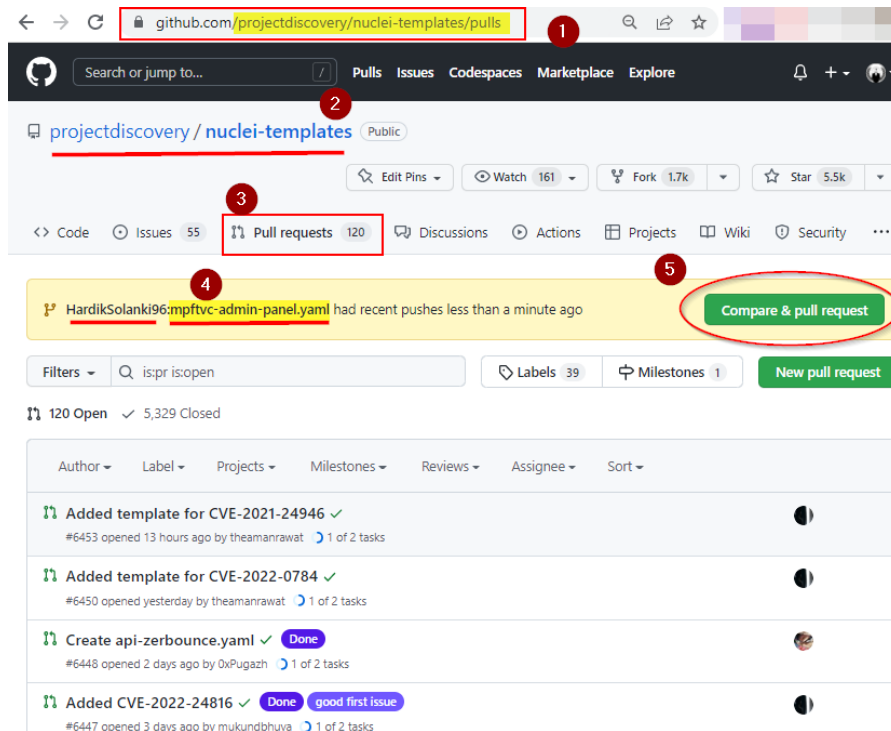


Figure 7.6: Compare pull request from main nuclei repository

After that, name the YAML template or rename/change the name of the template according to the YAML template, then click on “Create pull request”.

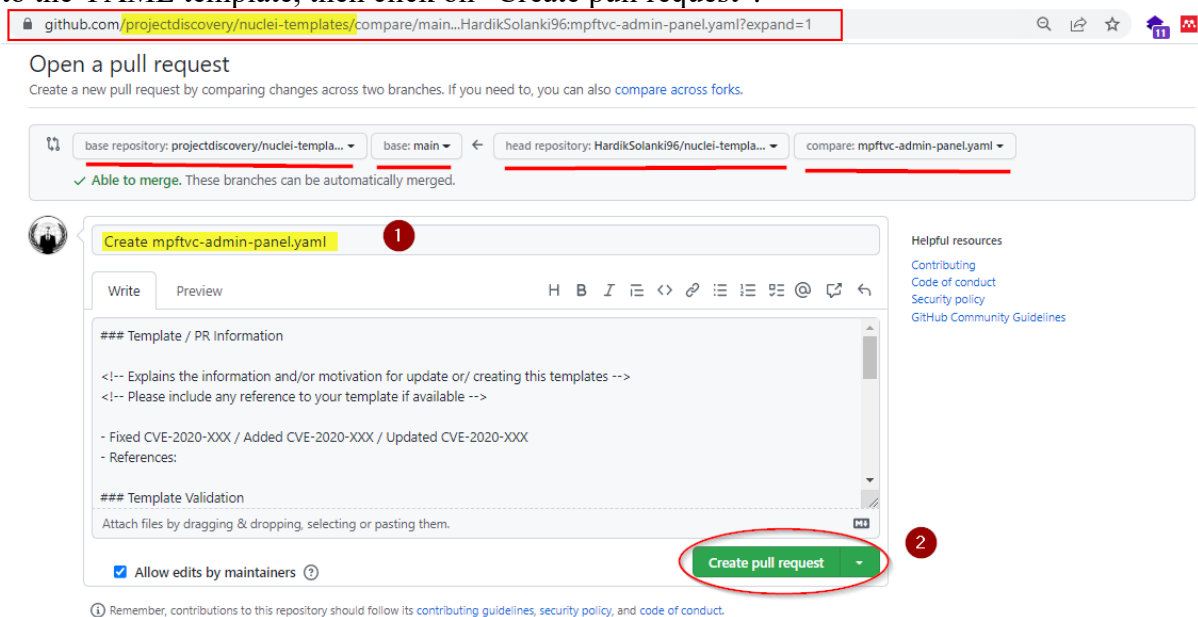


Figure 7.7: Create Pull Request

In conclusion, the following exhibits show that the Git Pull request was successfully created and the Pull request number is "6455" and that the custom YAML template designed for this project has been submitted to Nuclei's main GitHub repository for review, verification, and validation by Nuclei Maintainers.

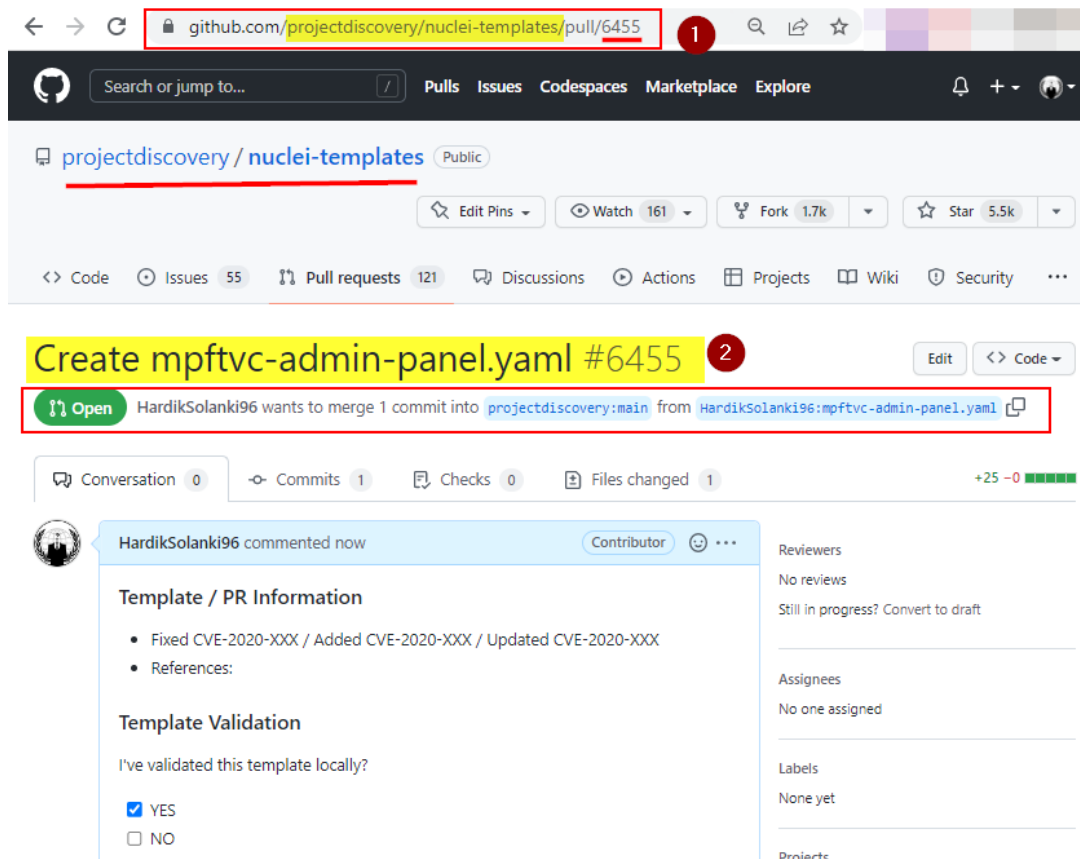


Figure 7.8: Pull Request created and submitted to Nuclei Maintainer team

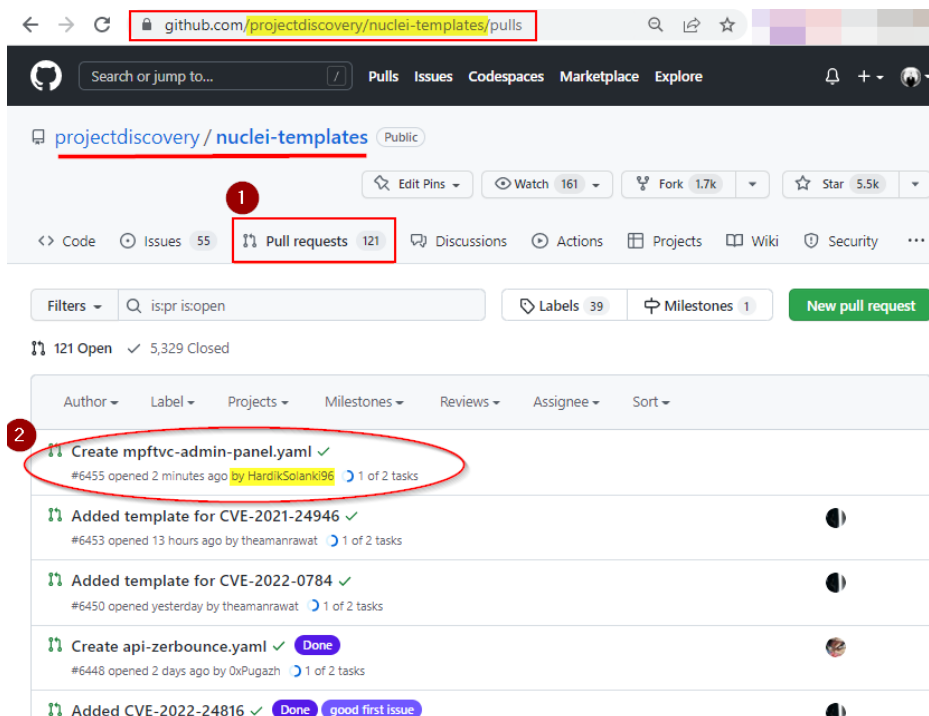


Figure 7.9: Created Pull request

Step 5: At this time, the Custom YAML templates can be reviewed by 2 reviewers from Nuclei Maintainers Team.

The 1st reviewer will be to check for duplicate YAML templates or if additional information is needed related to the submitted YAML template from the author (Hardik-Solanki). Once the YAML template is updated and correct as sent to the Nuclei team in first review, it will be passed on to the second reviewer. This is because they will re-check, verify and validate the submitted YAML template before it can merge into Nuclei’s main GitHub repository (Publicly).

Now, according to the following exhibit, the first review has been completed successfully by reviewer "DhiyaneshGeek" from the nuclei maintainers team (No duplicates & no additional information was required from the Author for the first review) and the second request has been forwarded to reviewer "pussycat0x".

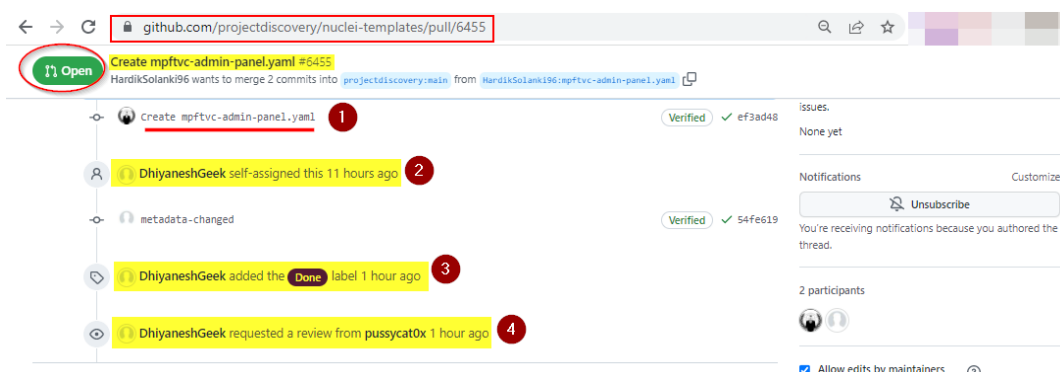


Figure 7.10: 1st review done by “DhiyaneshGeek”



Figure 7.11: Email Notification after 1st review has been done by Nuclei Maintainers team

Since the status shown in the above figure is still OPEN, it means that the submitted YAML template has not yet been merged and implemented into the Nuclei GitHub repository (Production/Publicly). This is also shown in the below figure, where no such “mpftvc-admin-panel” YAML template has been merged and implemented publicly under author (hardik-solanki).

```
(kali@kali)-[~]
└─$ nuclei -a Hardik-Solanki -vv

nuclei v2.8.3
projectdiscovery.io

[INF] Using Nuclei Engine 2.8.3 (outdated)
[INF] Using Nuclei Templates 9.3.3 (latest)
[INF] Templates added in last update: 238
[INF] Templates loaded for scan: 21
[INF] Targets loaded for scan: 0
[CVE-2022-3768] WPSmartContracts < 1.3.12 - Author SQLi (@hardik-solanki) [high]
[CVE-2022-4260] WP-Ban < 1.69.1 - Admin Stored XSS (@hardik-solanki) [high]
[dqs-superadmin-panel] DQS Superadmin Login Panel (@hardik-solanki) [info]
[flahscookie-superadmin-panel] Flahscookie Superadmin Login (@hardik-solanki) [info]
[gyra-master-admin] GYRA Master Admin (@hardik-solanki) [info]
[superadmin-ui-panel] Superadmin UI Login (@hardik-solanki) [info]
[vodafone-voxui-panel] Vodafone Vox UI Panel (@hardik-solanki) [info]
[xfinity-panel] Xfinity Panel (@hardik-solanki) [info]
[golangci-config] GolangCI-Lint Configuration File Exposure (@hardik-solanki) [low]
[stestr-config] Stestr Configuration File Exposure (@hardik-solanki) [info]
[cloud-config] Cloud Config File Exposure (@dhiyaneshdk,@hardik-solanki) [medium]
[database-credentials] Database Credentials File Exposure (@hardik-solanki) [low]
[kubernetes-etcd-keys] Kubernetes etcd Keys Exposure (@hardik-solanki) [medium]
[svn-wc-db] SVN wc.db File Exposure (@hardik-solanki) [medium]
[badarg-log] Badarg Log File Exposure (@hardik-solanki) [low]
[error-logs] Common Error Log Files (@geeknik,@daffainfo,@elsfa7110,@hardik-solanki) [low]
[firebase-debug-log] Firebase Debug Log File Exposure (@hardik-solanki) [low]
[npm-debug-log] NPM Debug Log Disclosure (@hardik-solanki) [low]
[ws-ftp-log] WS FTP File Disclosure (@hardik-solanki) [low]
[carel-plantvisor-panel] CAREL Pl@ntVisor Panel (@hardik-solanki) [info]
[hue-personal-wireless-panel] HUE Personal Wireless Lighting Panel (@hardik-solanki) [info]
[INF] No results found. Better luck next time!
```

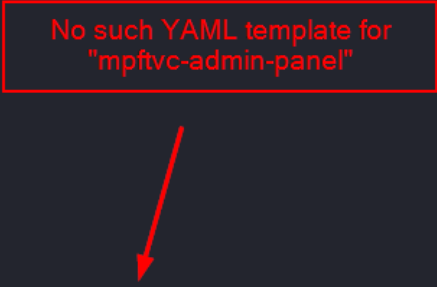


Figure 7.12: custom YAML template is not yet publicly released

When the 2nd reviewer confirms everything is fine without requiring any additional information from the YAML template author, it will successfully change the status from "Open" to "Merged" by Nuclei's Maintainers Team.

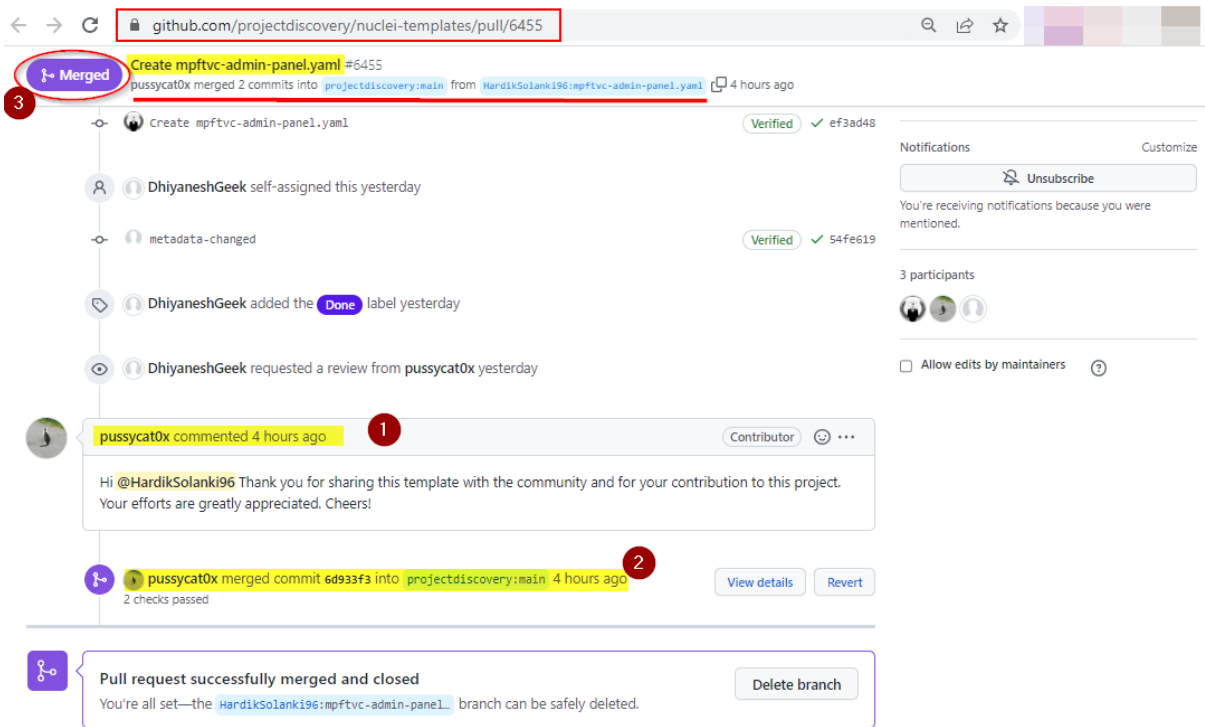


Figure 7.13: 2nd review done by “pussycat0x”

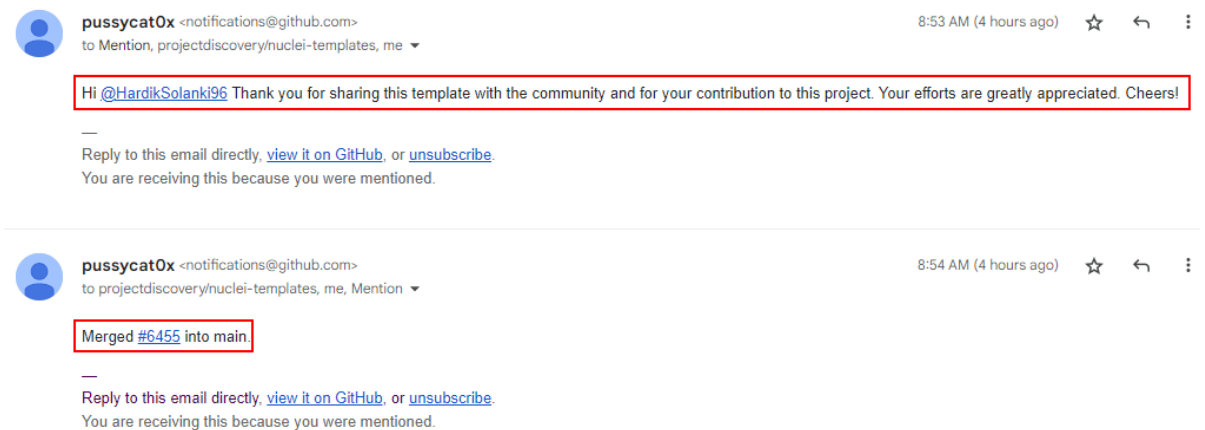


Figure 7.14: Email notification from 2nd reviewer “pussycat0x”

Thus, this status indicates that the created custom YAML Template in this research has been successfully merged and implemented into the Nuclei Main GitHub repository and is now ready for public use (Production Environment).

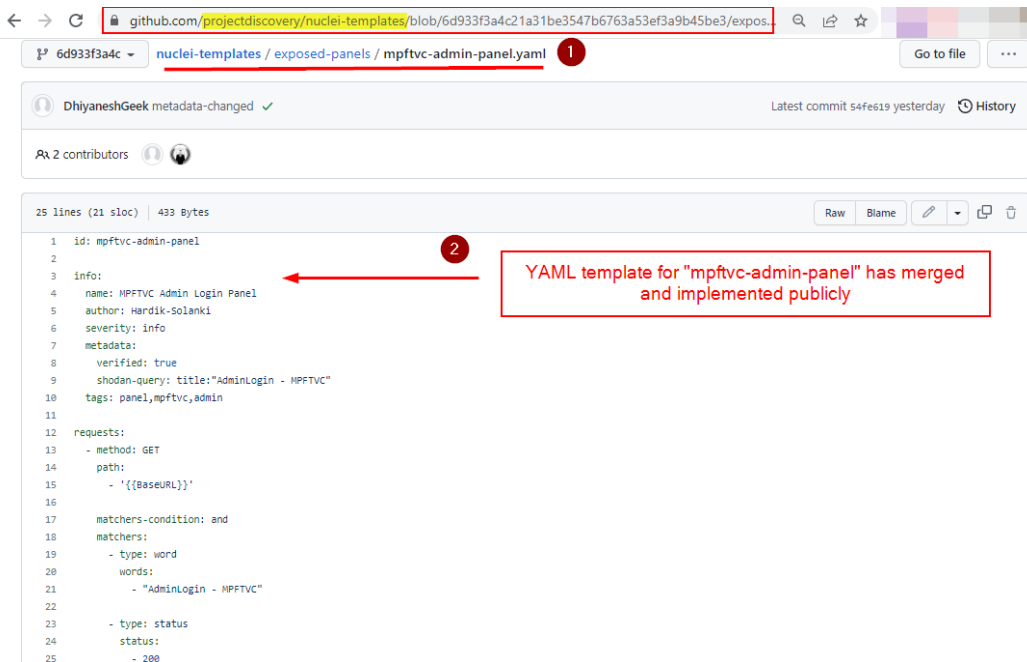


Figure 7.15: "mpftvc-admin-panel" has merged and implemented publicly

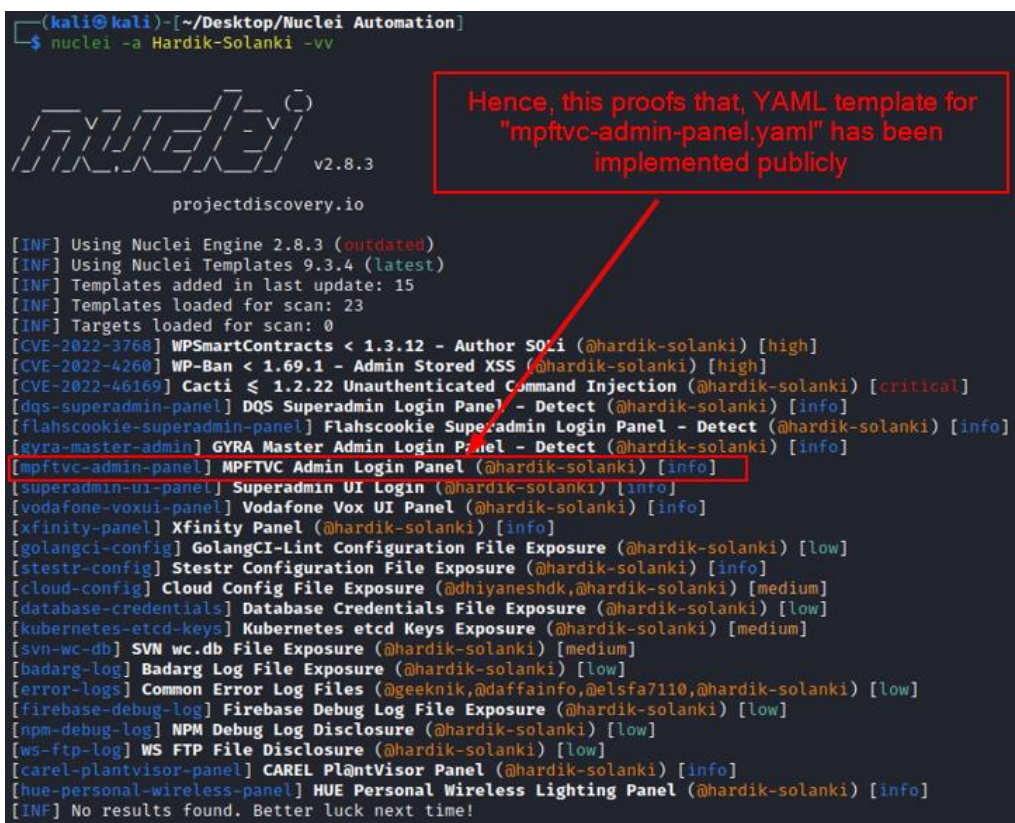


Figure 7.16: Successfully merged and publicly available now.

5 Evaluation

A key objective of this research project was to automate the Vulnerability Assessment process using Nuclei Automation, based on developed custom YAML templates, and manage

the attack surface of the infrastructure applications. This research project was successfully implemented after creating a custom YAML template and implementing into Nuclei's main GitHub repository (publicly).

In the below exhibit 8, demonstrated how to run the custom created and implemented YAML into Organizations Infrastructure with a YAML template "mpftvc-admin-panel.yaml" followed by a list of Infrastructure domains:

Command usage has been listed in above section 3 from exhibit 6 to 6.4.

```
(kali@kali)-[~/Desktop/Nuclei Automation]
└─$ nuclei -t mpftvc-admin-panel.yaml -l Infrastructure-domains.txt -vv

nuclei v2.8.3
projectdiscovery.io

[INF] Supplied input was automatically deduplicated (1 removed).
[INF] Using Nuclei Engine 2.8.3 (outdated)
[INF] Using Nuclei Templates 9.3.3 (latest)
[INF] Templates added in last update: 238
[INF] Templates loaded for scan: 1
[INF] Targets loaded for scan: 69
[mpftvc-admin-panel] MPFTVC Admin Login Panel (@hardik-solanki) [info]
[mpftvc-admin-panel] [http] [info] http://3.1.153.231/
[mpftvc-admin-panel] [http] [info] http://3.1.153.231:90/
```

Figure 8: Verify vulnerable domains using "mpftvc-admin-panel" YAML template

Followed by the "Infrastructure domains" list and YAML template, "mpftvc-admin-panel". It was observed that, out of 69 Infrastructure targets, 2 were vulnerable and exposes the Internal "MPFTVC Admin Login Panel" publicly whose severity is "info". As shown in the below exhibit 8.1.

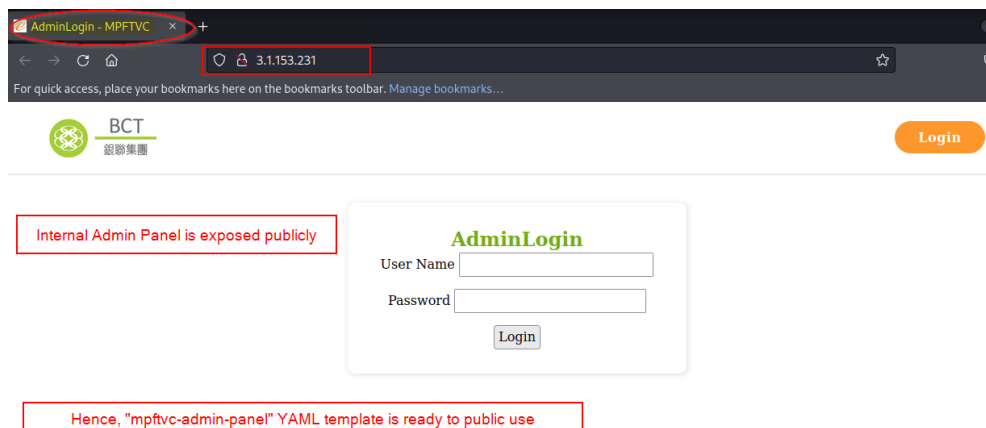


Figure 8.1: "mpftvc-admin-panel" vulnerable internal domain

6 Monthly Internship Activity Report

Student Name: **Hardik Solanki**

Student number: **x21117659**

Company: **Kontex Security Ltd**

Month Commencing: **October 2022**

My research thesis project topic is “Limiting Attack Surface for Infrastructure Applications using Custom YAML Templates in Nuclei Automation”.

The purpose of selecting this topic is to monitor and managing the attack surface before any publicly attacks occurs on the organization's infrastructure.

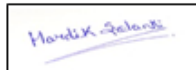
Hence, I am going to write custom YAML templates based on publicly available vulnerabilities for Web Application, Mobile Application, Sensitive files, CVEs, Network & cloud environment/application and will integrate into Open-Source tool called “Nuclei” Scanner (Scanner Built by the Project Discovery Team). “Nuclei” is automated Vulnerability Assessment/Management scanner tool that runs only based on user Custom defined YAML Templates.

For this project, I've started with the project and abstract planes. By October 2022, I've finished my abstract plane and other information gathering for my project.

Employer comments

Hardik has started the project execution phase, finished his project plans, and written the project's abstract. He will now work on the project's brief introduction, related work, research methodology, design specification & implementation, and evaluation phases.

Student Signature:



Date: **31st October 2022**

Industry Supervisor Signature: **Michal Skackov**
Skackov
(Director, Kontex Security)

Date: **31st October 2022**

Figure 9: Activity report for October 2022

Student Name: **Hardik Solanki**

Student number: **x21117659**

Company: **Kontex Security Ltd**

Month Commencing: **November 2022**

I have completed the Abstract of my project and also complete my brief introduction related to this Project.

Furthermore, I will be working on "Research Methodology, Design & Implementation, and Evaluation".

I have followed AGILE methodology for this research:

1. **Requirement:** Will gather the required data and information from open source (OSINT) regarding vulnerabilities, vulnerable end points, etc.
2. **Design:** Start creating the YAML templates based on gathered information.
3. **Development:** Template author personally validate the YAML template and create a pull request to submit the designed template in the Nuclei GitHub repository for further development.
4. **Review/Test:** Custom templates will be tested and validated from Nuclei Team members.
5. **Deploy:** Once the templates are validated and its correct, then deployed in Nuclei production environment.
6. **Review/Launch:** Once deployed, templates are further ready to use publicly for Vulnerability Assessment Scanning Automation, Monitoring, and managing attack surfaces.

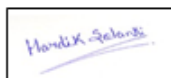
Environment Requirements:

- **OS:** Kali Linux
- **Language:** Golang should be installed in Linux System.
- **Usage:** git clone <https://github.com/projectdiscovery/nuclei-templates.git>

Employer comments

Hardik is currently working on the reporting and documentation after completing the "Research Methodology, Design & Implementation, and Evaluation."

Student Signature:



Date: **30th November 2022**

Industry Supervisor Signature: Michal Skackov
Skackov
(Director, Kontex Security)

Date: **30th November 2022**

Figure 9.1: Activity report for November 2022

Student Name: **Hardik Solanki**

Student number:

x21117659

Company: **Kontex Security Ltd**

Month Commencing:

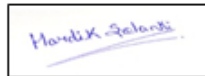
December 2022

I've created 15+ YAML based templates to Automate the attack Surface and manage the Security Testing (Vulnerability Assessment) and Integrated in Production Environment of Nuclei Automation scanner tool, and hence, this way I have completed the Research Methodology, Design & Implementation, and Evaluation. Now I will be working on reporting Phase and Manual Configuration documentation.

Employer comments

Hardik has finished the "Research Methodology, Design & Implementation, and Evaluation" and is now working on the reporting and documentation.

Student Signature:



Date: **23rd December 2022**

Industry Supervisor Signature: **Michal Skackov**

Skackov
(Director, Kontex Security)

Date: **23rd December 2022**

Figure 9.2: Activity report for December 2022