v1.0.0
Jul 7 2021

Research on methods of detecting defacement attack based on deep learning

Edchina, Cu64, Klone

CONTENT

1. Motivation
2. Building model Machine Learning
3. Alert system development
4. Experiment and evaluation

1 Motivation

2 Building model Machine Learning

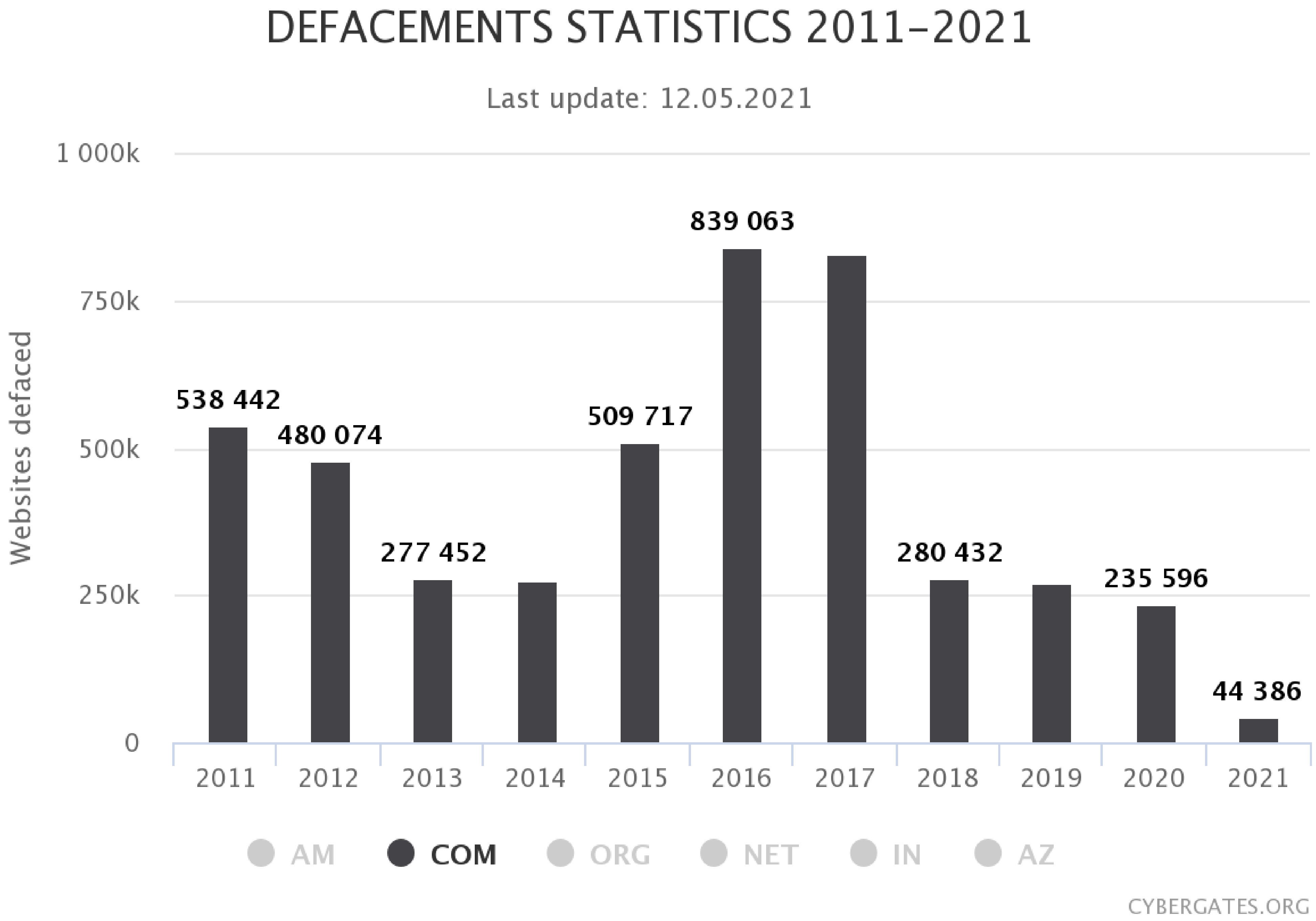
3 Alert system development

4 Experiment and evaluation

MOTIVATION

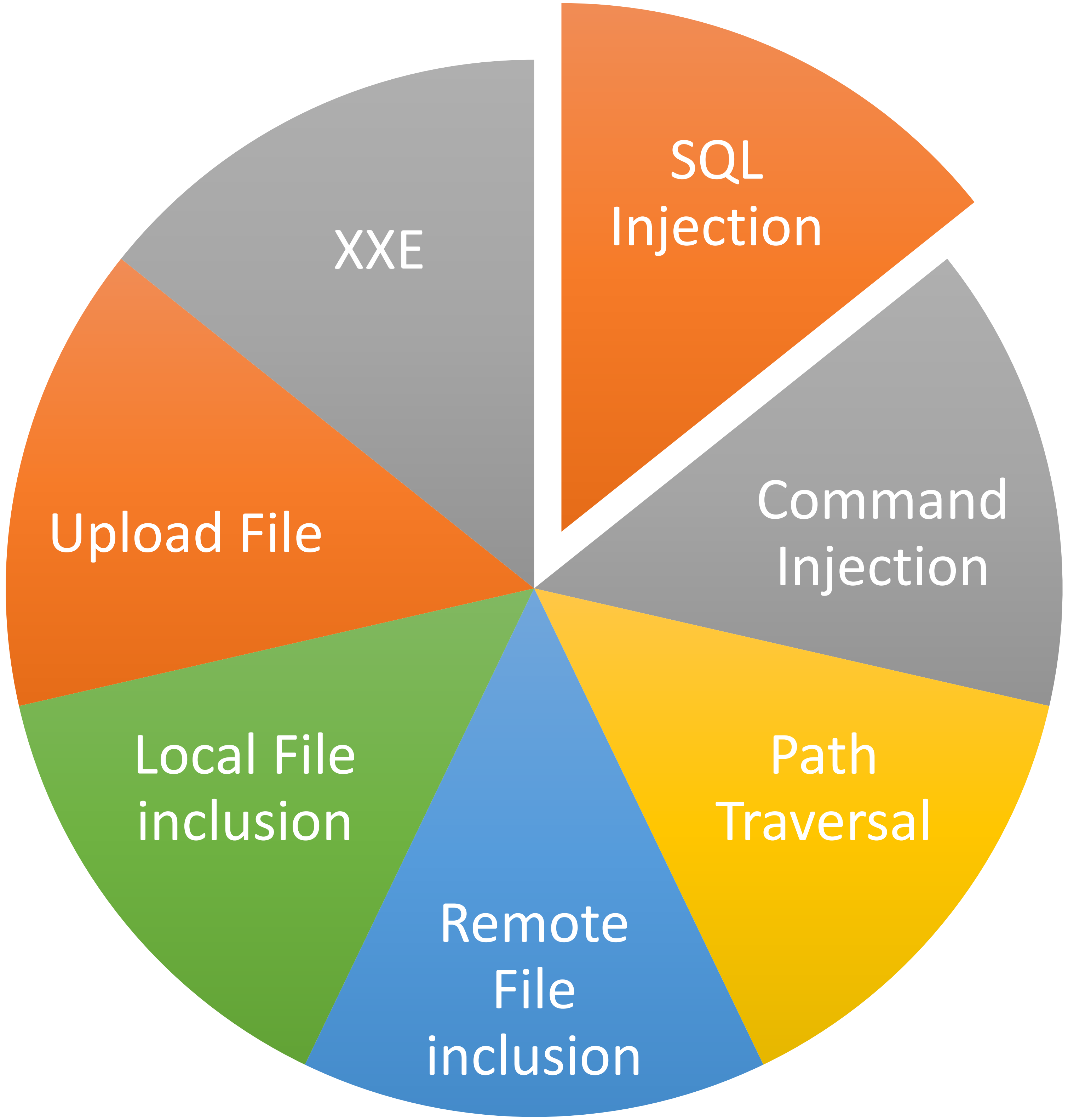
❖ Goals:

- Cause political conflict
- To show off
- Warn the system administrator



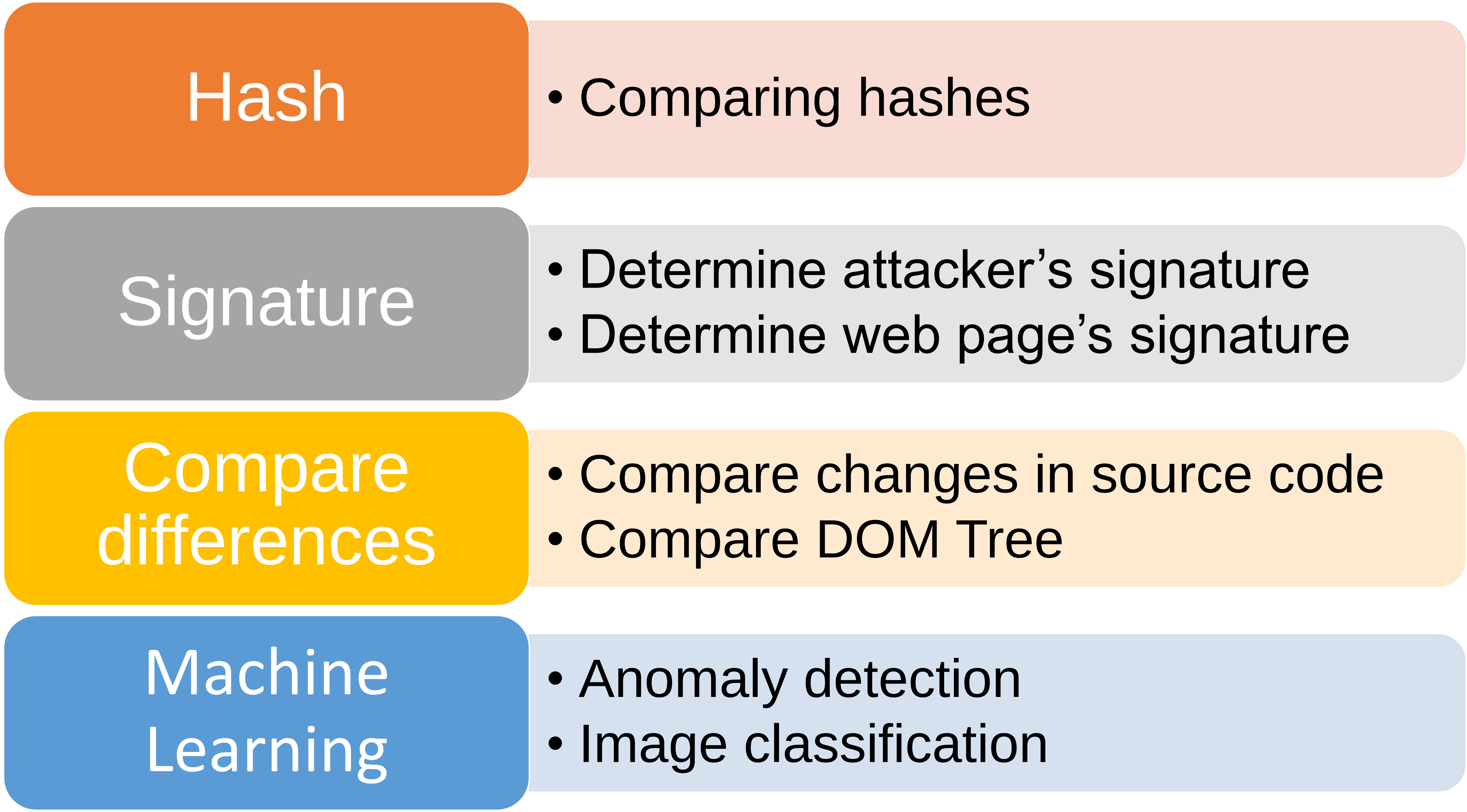
The amount of websites under the .com domain that faced defacement attacks from 2011 to 2021

SOME ATTACK TECHNIQUES



Some critical web vulnerabilities

DETECTION TECHNIQUES



Hash

- Comparing hashes

Signature

- Determine attacker’s signature
- Determine web page’s signature

Compare differences

- Compare changes in source code
- Compare DOM Tree

Machine Learning

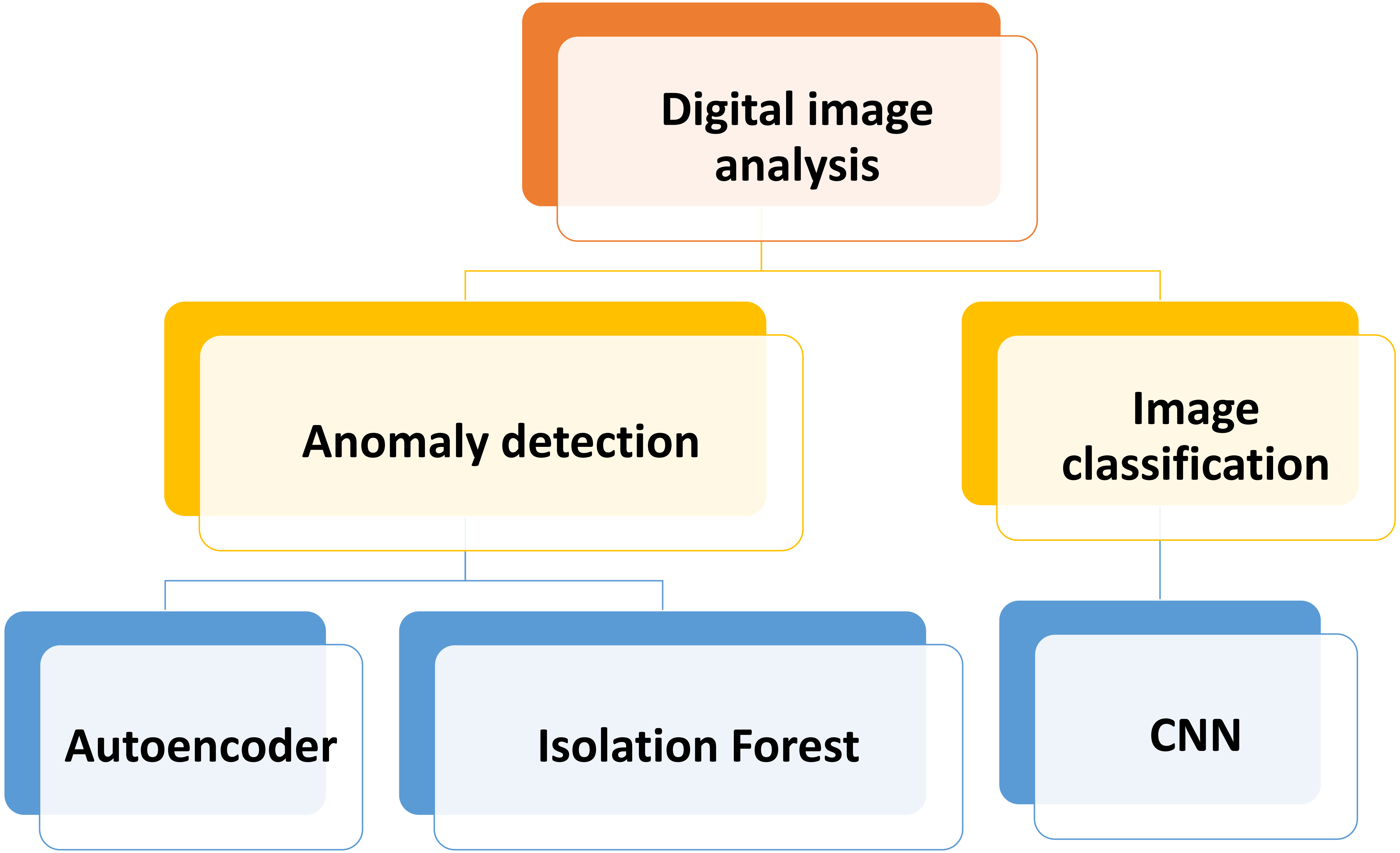
- Anomaly detection
- Image classification

INTRODUCTION

"RESEARCH ON METHODS OF DETECTING DEFACEMENT ATTACKS BASED ON DEEP LEARNING"

1. Motivation
2. Building model Machine Learning
3. Alert system development
4. Experiment and evaluation

APPROACH



Digital image analysis

Anomaly detection

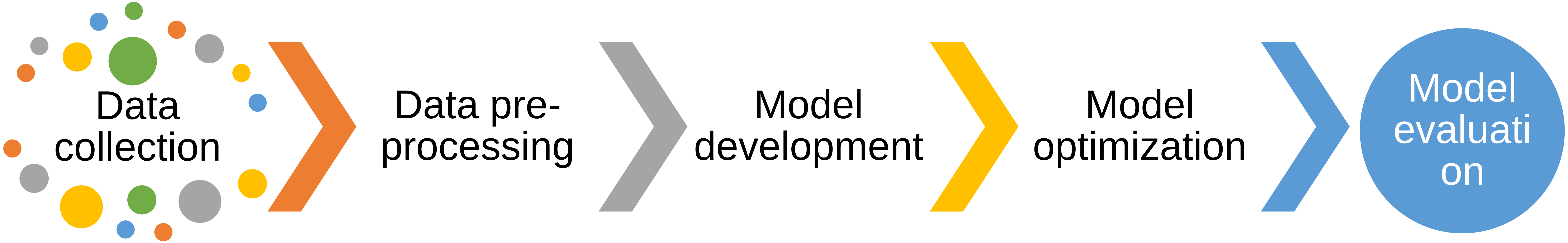
Image classification

Autoencoder

Isolation Forest

CNN

APPROACH



Data collection

Data pre-processing

Model development

Model optimization

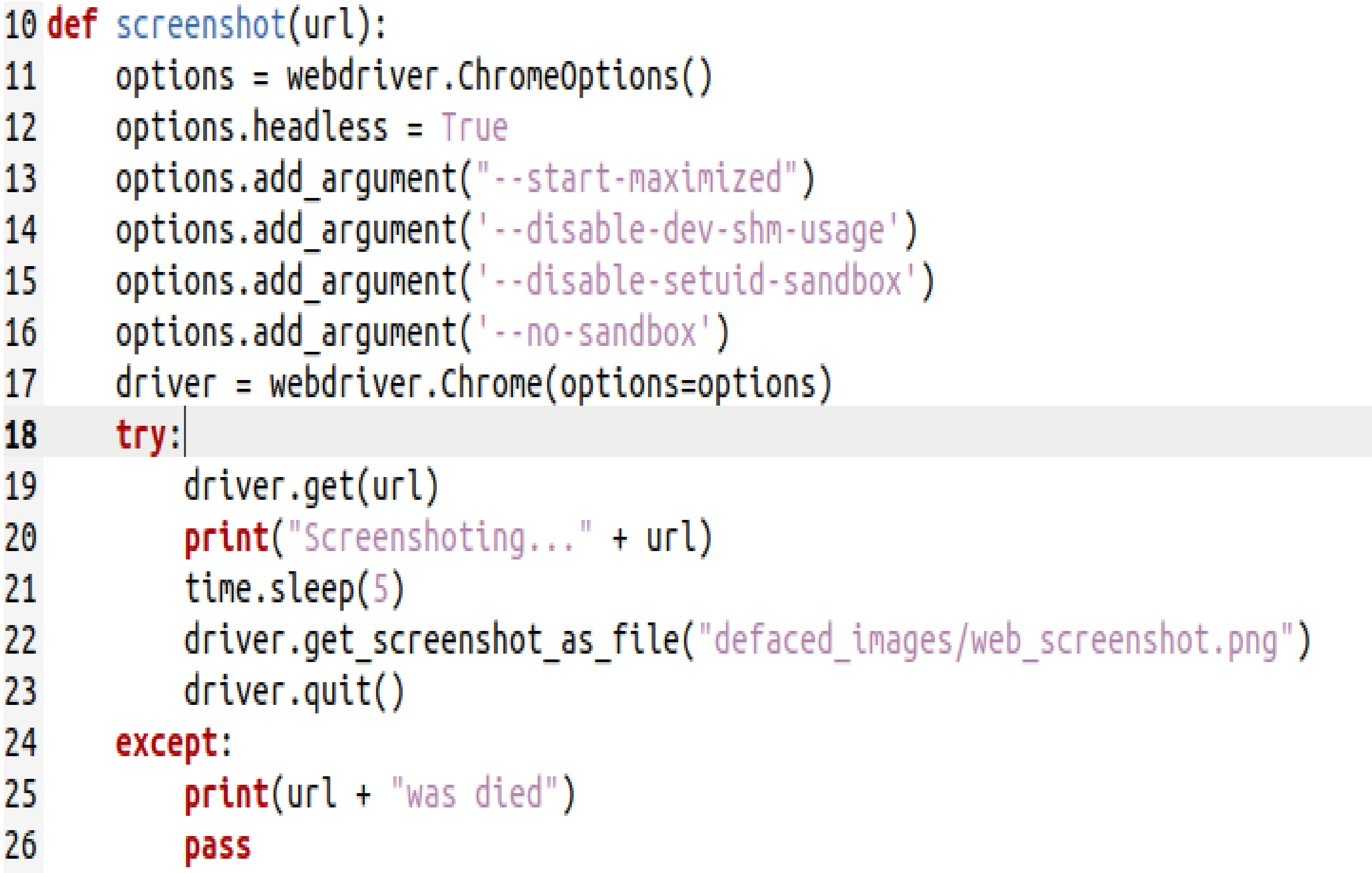
Model evaluati on

INPUT DATA COLLECTION

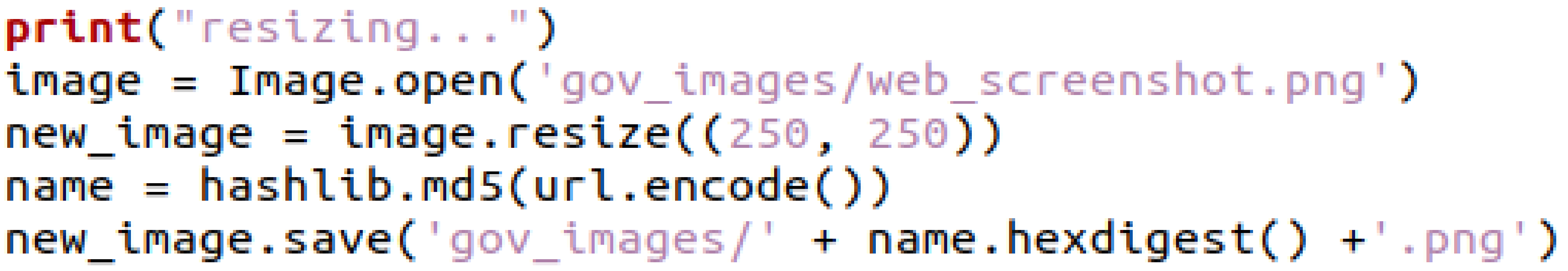
- ❖ Normal website data
  - moz.com/top500
  - github.com/GSA/govt-urls

- ❖ Defaced website data
  - mirror-h.org
  - zone.kurd-h.org
  - www.zone-h.org
  - www.xatrix.org/defac.php

Screenshot: Python-selenium & Chromium



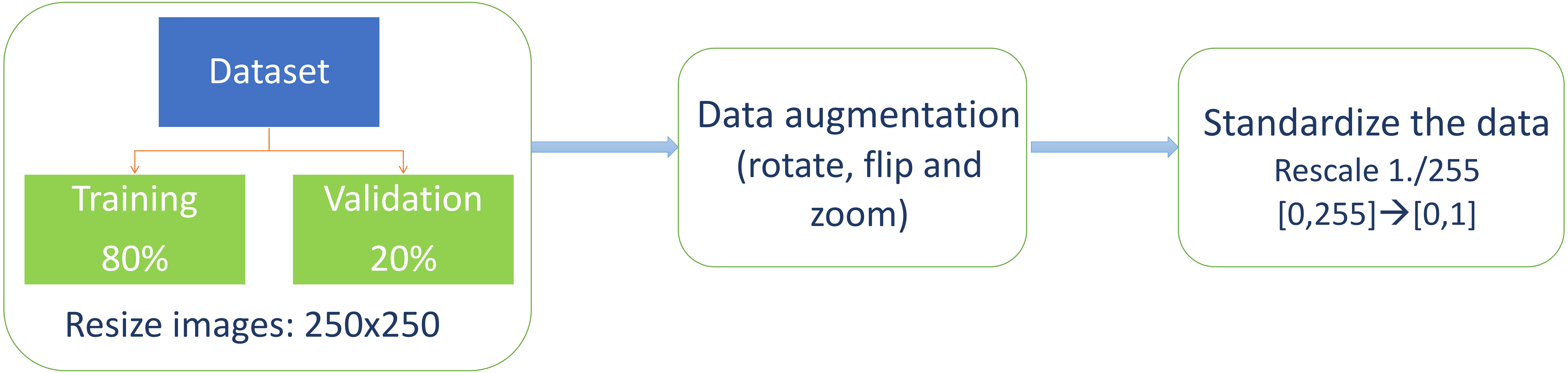
```python
def screenshot(url):
    options = webdriver.ChromeOptions()
    options.headless = True
    options.add_argument("--start-maximized")
    options.add_argument('--disable-dev-shm-usage')
    options.add_argument('--disable-setuid-sandbox')
    options.add_argument('--no-sandbox')
    driver = webdriver.Chrome(options=options)
    try:
        driver.get(url)
        print("Screenshoting..." + url)
        time.sleep(5)
        driver.get_screenshot_as_file("defaced_images/web_screenshot.png")
        driver.quit()
    except:
        print(url + "was died")
        pass
```



```python
print("resizing...")
image = Image.open('gov_images/web_screenshot.png')
new_image = image.resize((250, 250))
name = hashlib.md5(url.encode())
new_image.save('gov_images/' + name.hexdigest() +'.png')
```

DATA PRE-PROCESSING

- Dataset
  - Clean: 6,796 pics
  - Deface: 4,917 pics



Dataset

Training

80%

Validation

20%

Resize images: 250x250

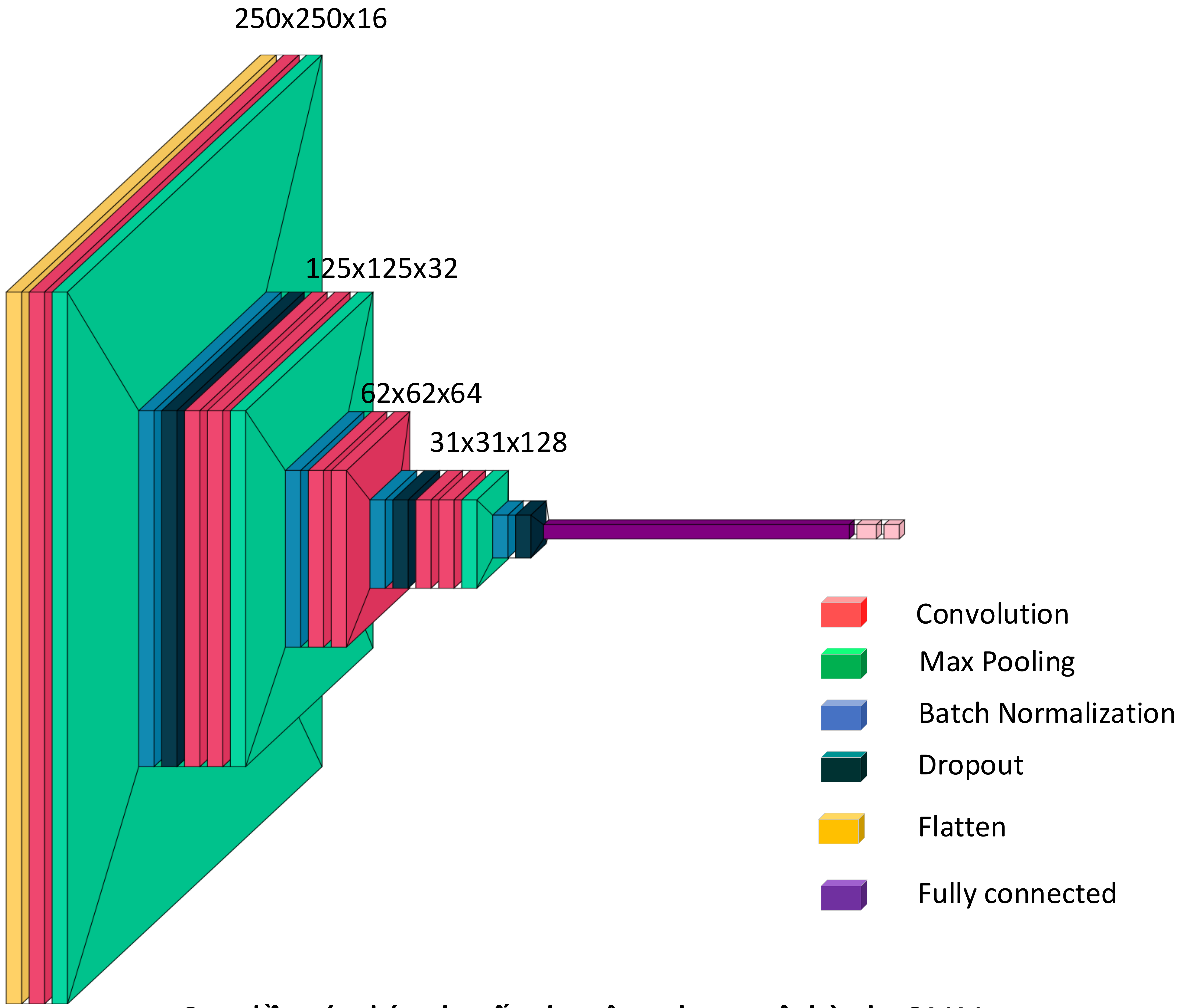
Data augmentation (rotate, flip and zoom)

Standardize the data

Rescale 1./255

[0,255]→[0,1]

BUILDING MODEL



Sơ đồ các lớp huấn luyện cho mô hình CNN

MODEL OPTIMIZATION

- Reduce overfitting
  - Add Dropout layers
  - Data augmentation
  - Add Batch Normalization layers

MODEL EVALUATION

- Evaluation criteria:

Accuracy

$$Accuracy = \frac{TP+TN}{TP+TN+FP+FN} = 98.01$$

True positive rate

$$TPR = \frac{TP}{TP+FN} = 98.12$$

False positive rate

$$FPR = \frac{FP}{FP+TN} = 3.73$$

False negative rate

$$FNR = \frac{FN}{FN+TP} = 1 - TPR = 1.88$$

Positive predictive value

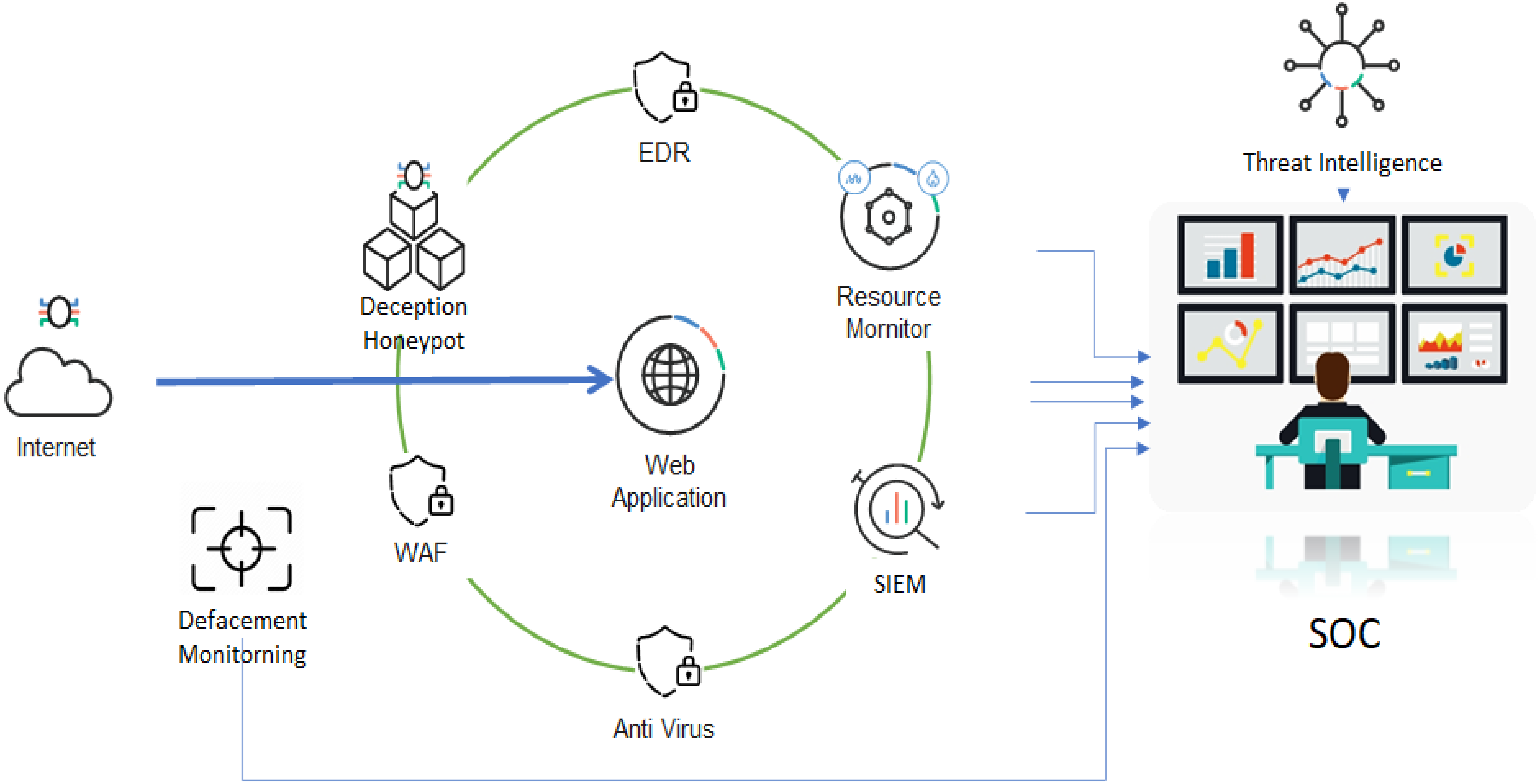
$$PPV = \frac{TP}{TP+FP} = 94.00$$

F1 Score

$$F1 = 2 \times \frac{PPV \ \times TPR}{PPV + TPR} = 96.02$$

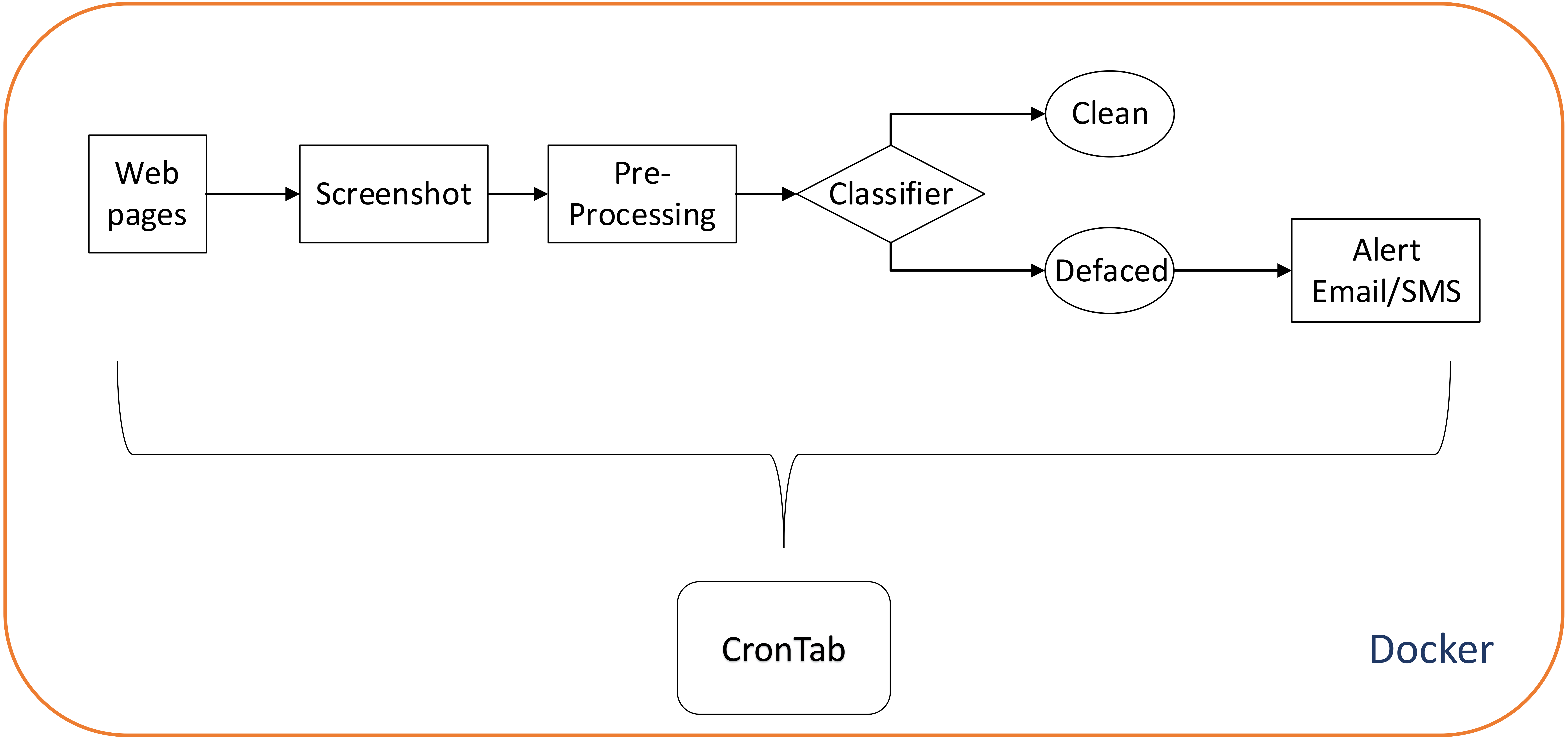
1. Motivation
2. Building model Machine Learning
3. Alert system development
4. Experiment and evaluation

MOTIVATION



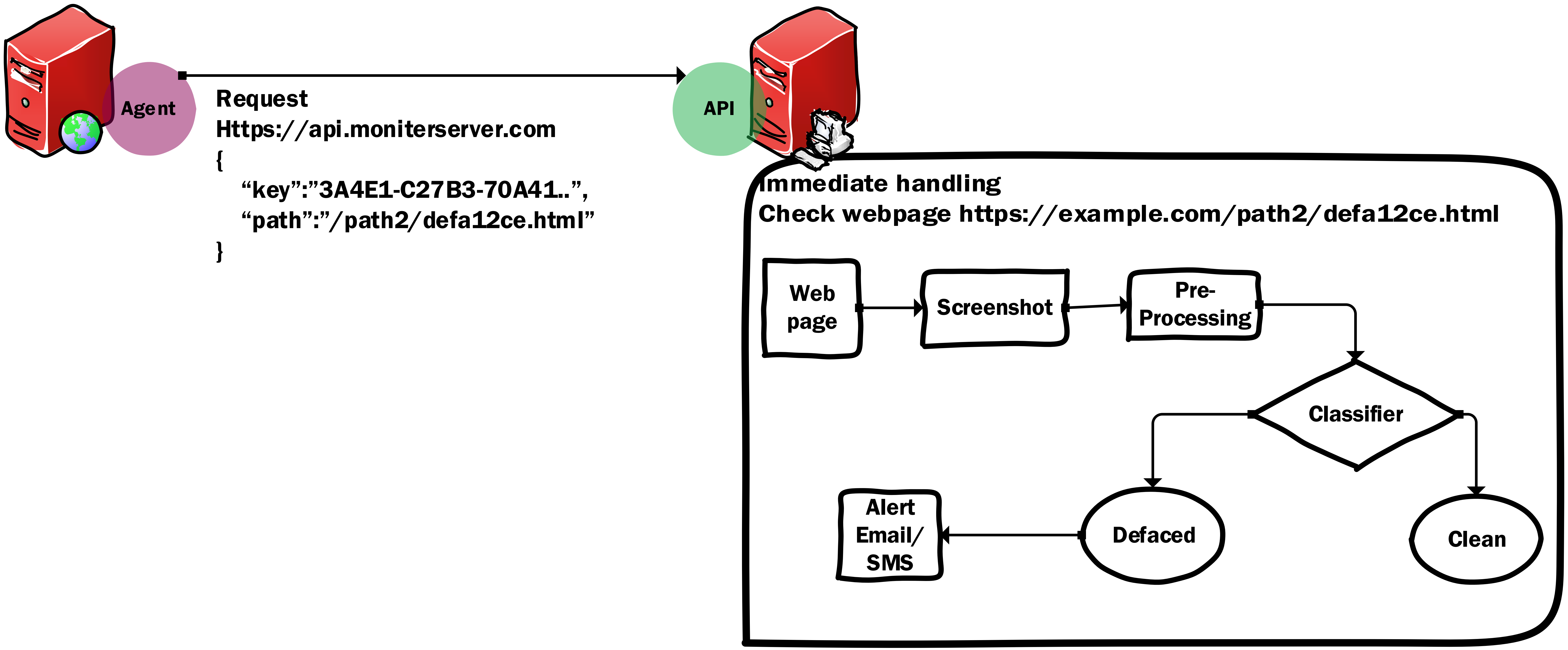
Some security solutions for a web application

ALERT SYSTEM DEVELOPMENT



Detection and warning diagram

AGENT DEPLOYMENT



Request
Https://api.moniterserver.com

```
{
   “key”:”3A4E1-C27B3-70A41..”,
   “path”:”/path2/defa12ce.html”
}
```

Immediate handling
Check webpage https://example.com/path2/defa12ce.html

Event handling flow

1. Motivation
2. Building model Machine Learning
3. Alert system development
4. Experiment and evaluation

INSTALLATION

Cloning the repository

```
git clone https://github.com/J4FSec/In0ri.git
cd In0ri
```

Configuring email credentials to send notifications and agent keys from

Edit the file FlaskApp/sendEmail.py

```
EMAIL_ADDRESS = "foo@gmail.com"
EMAIL_PASSWORD = "$uper$ecurePa$$word"
```

Configure Telegram notification

Edit the file chatbot.py

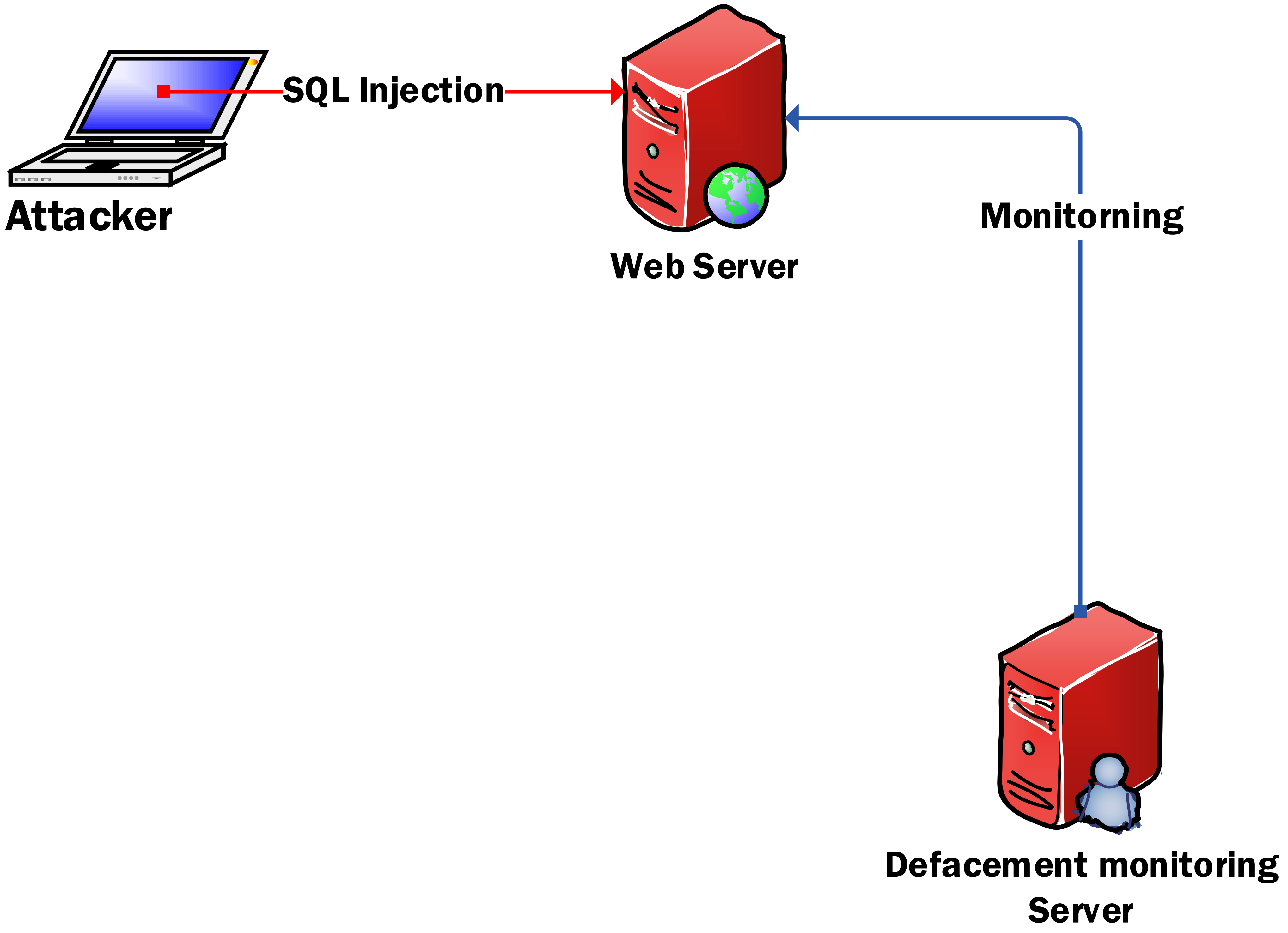
```
CHAT_ID= 'foo' // Channel ID to send notifications to
TOKEN = 'bar' // Bot token retrieved from @BotFather
```

Starting In0ri

```
docker-compose up -d
```

SCENARIO 1

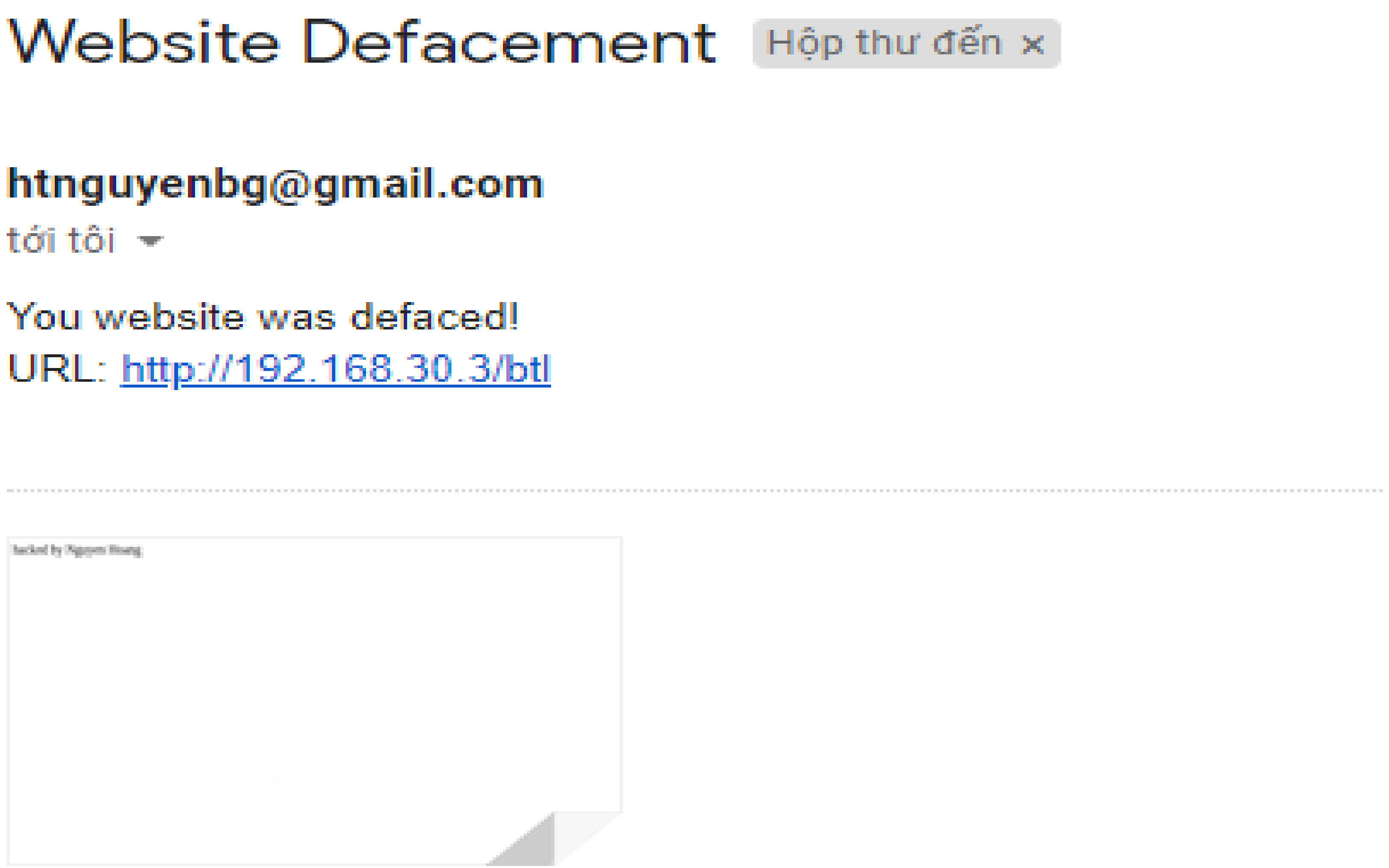
Scenario content



Scenario 1 diagram

SCENARIO 1

Result



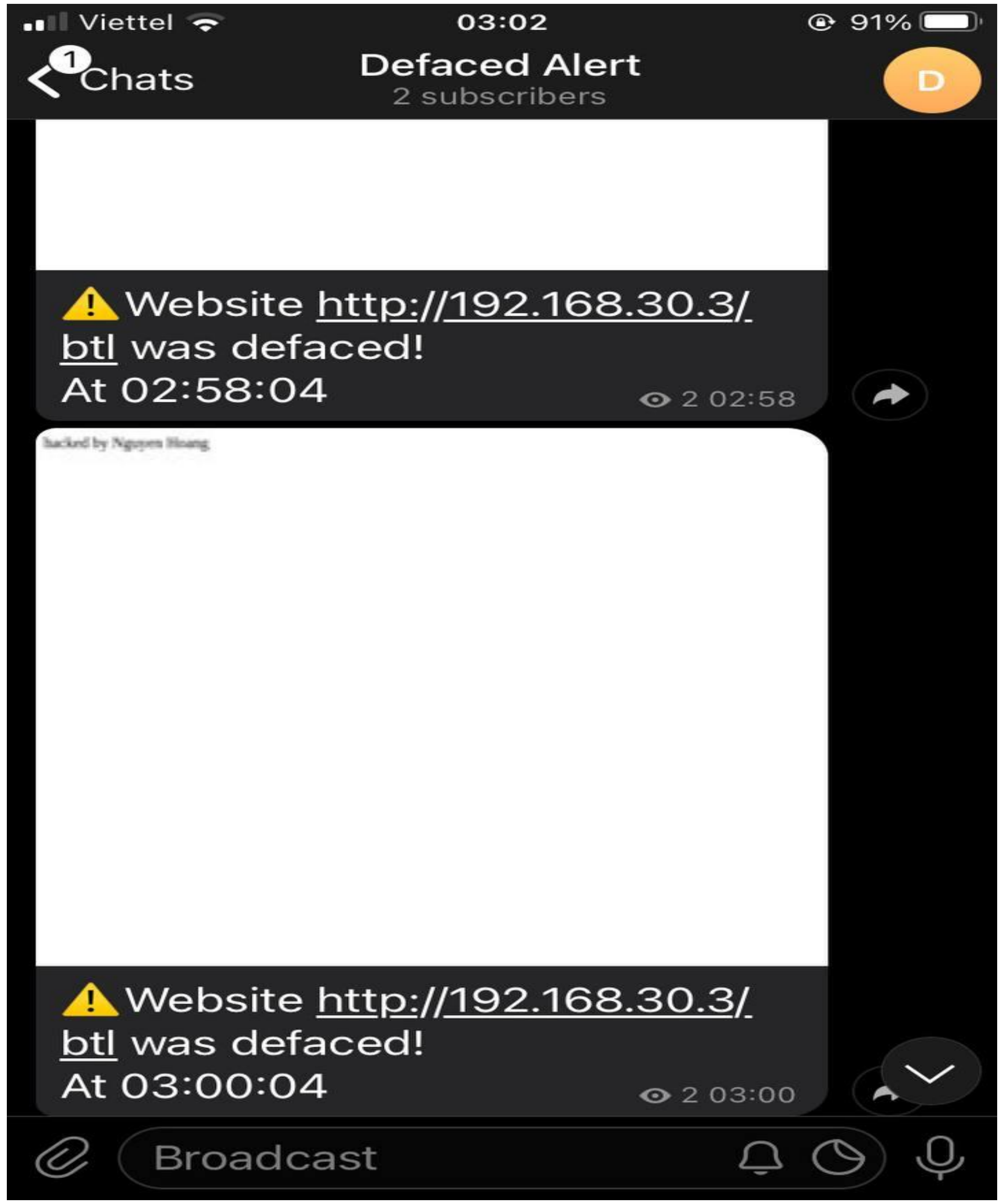
Website Defacement

htngmuyenbg@gmail.com

tới tôi

You website was defaced!
URL: http://192.168.30.3/btl

Email Alert



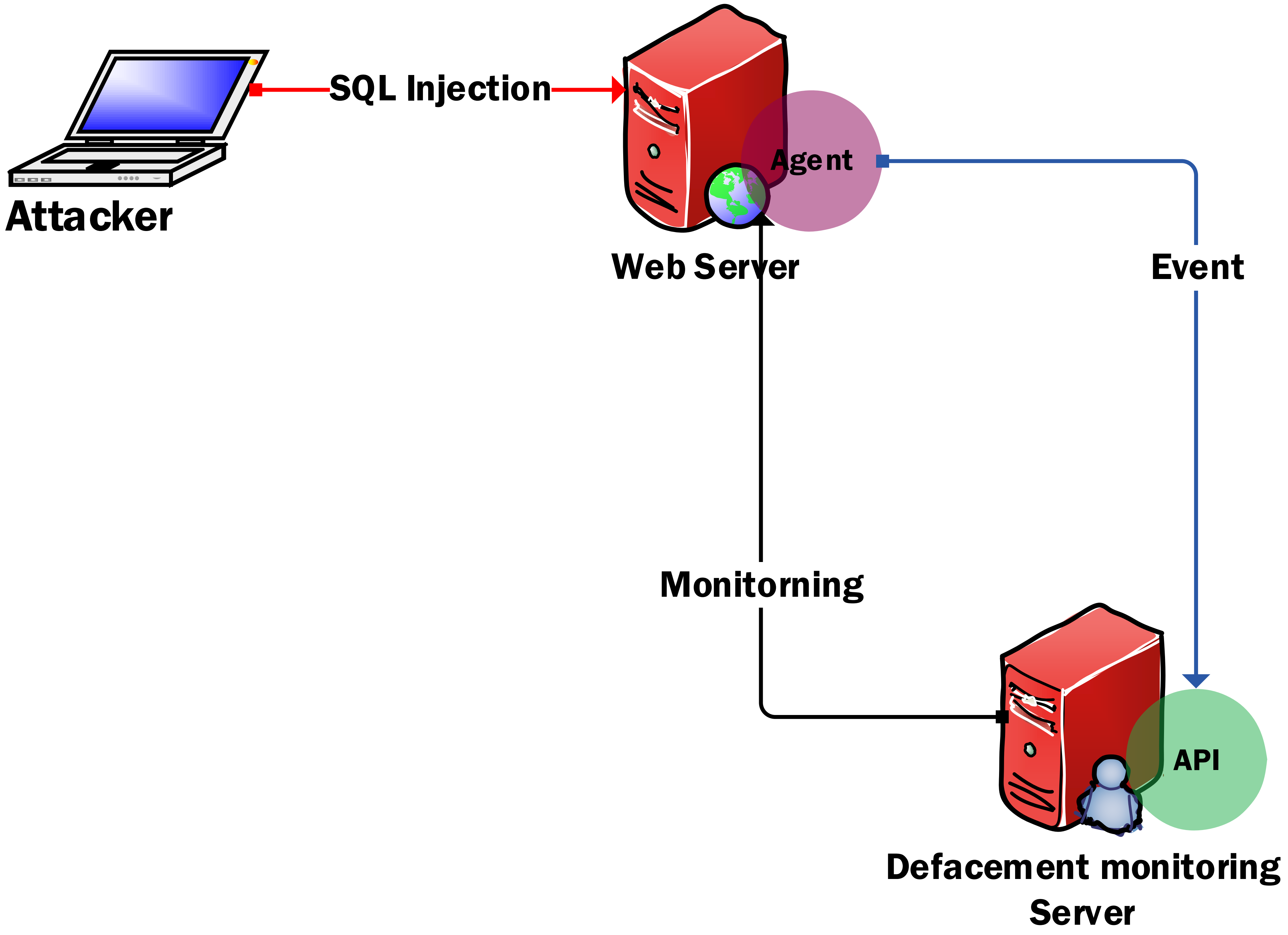
⚠Website http://192.168.30.3/btl was defaced!
At 02:58:04 2 02:58

⚠Website http://192.168.30.3/btl was defaced!
At 03:00:04 2 03:00

Telegram Alert

SCENARIO 2

Scenario content

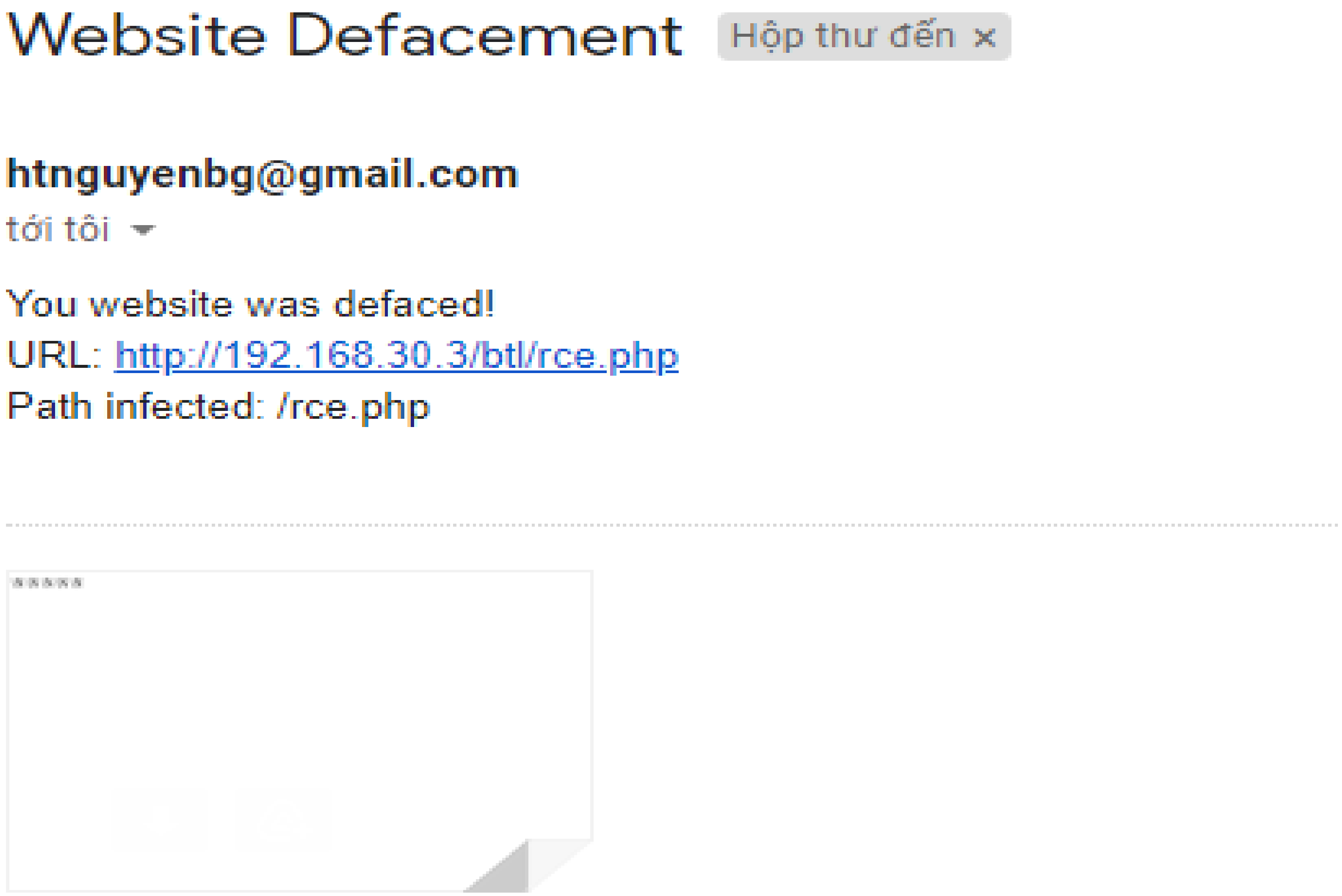


Scenario 2 diagram

SCENARIO 2

Result 1

- The system detected that an suspicious file has been created on the server and send out a warning



Website Defacement

htnguyenbg@gmail.com

tới tôi

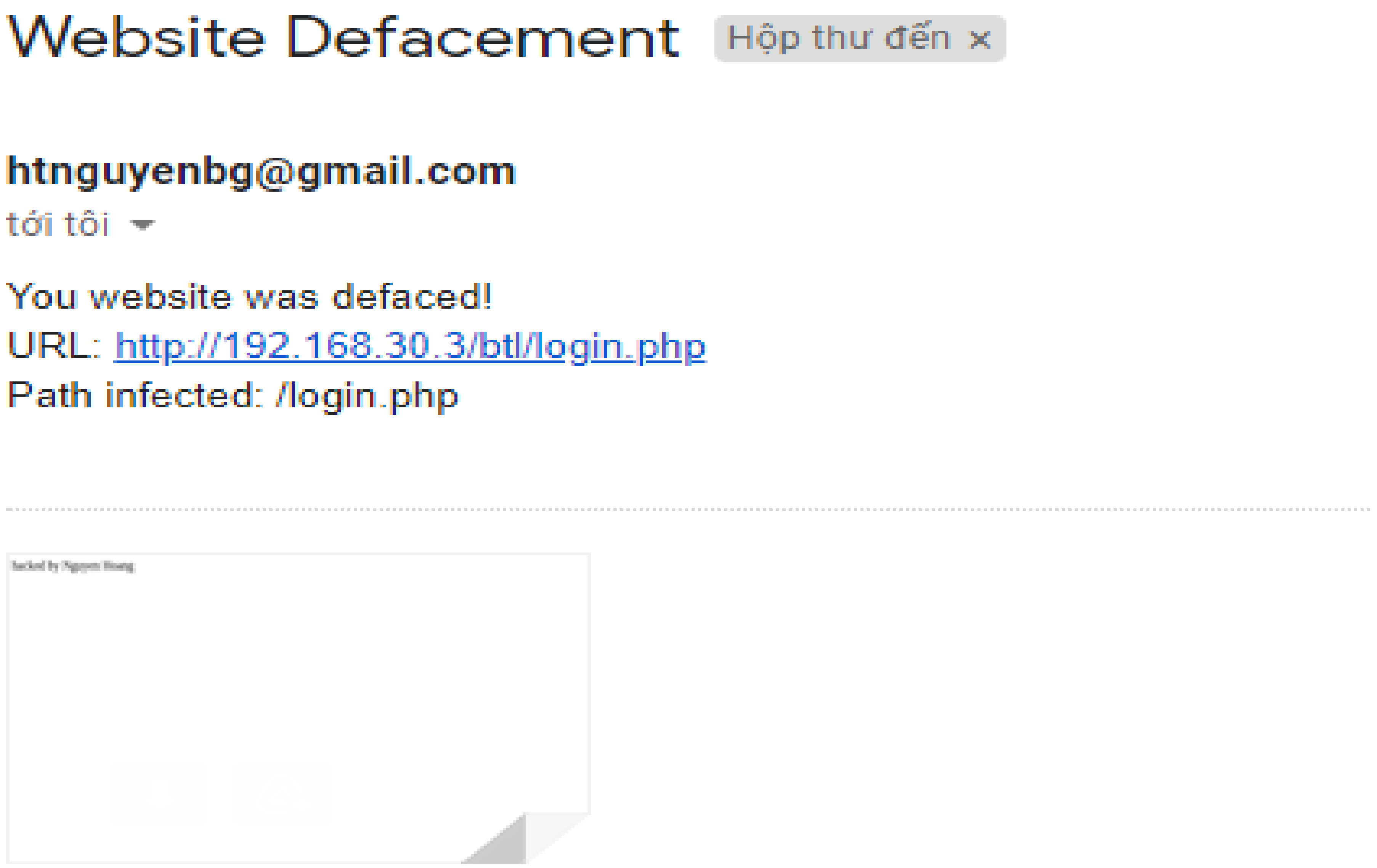
You website was defaced!
URL: http://192.168.30.3/btl/rce.php
Path infected: /rce.php

Email Alert

SCENARIO 2

Result 2

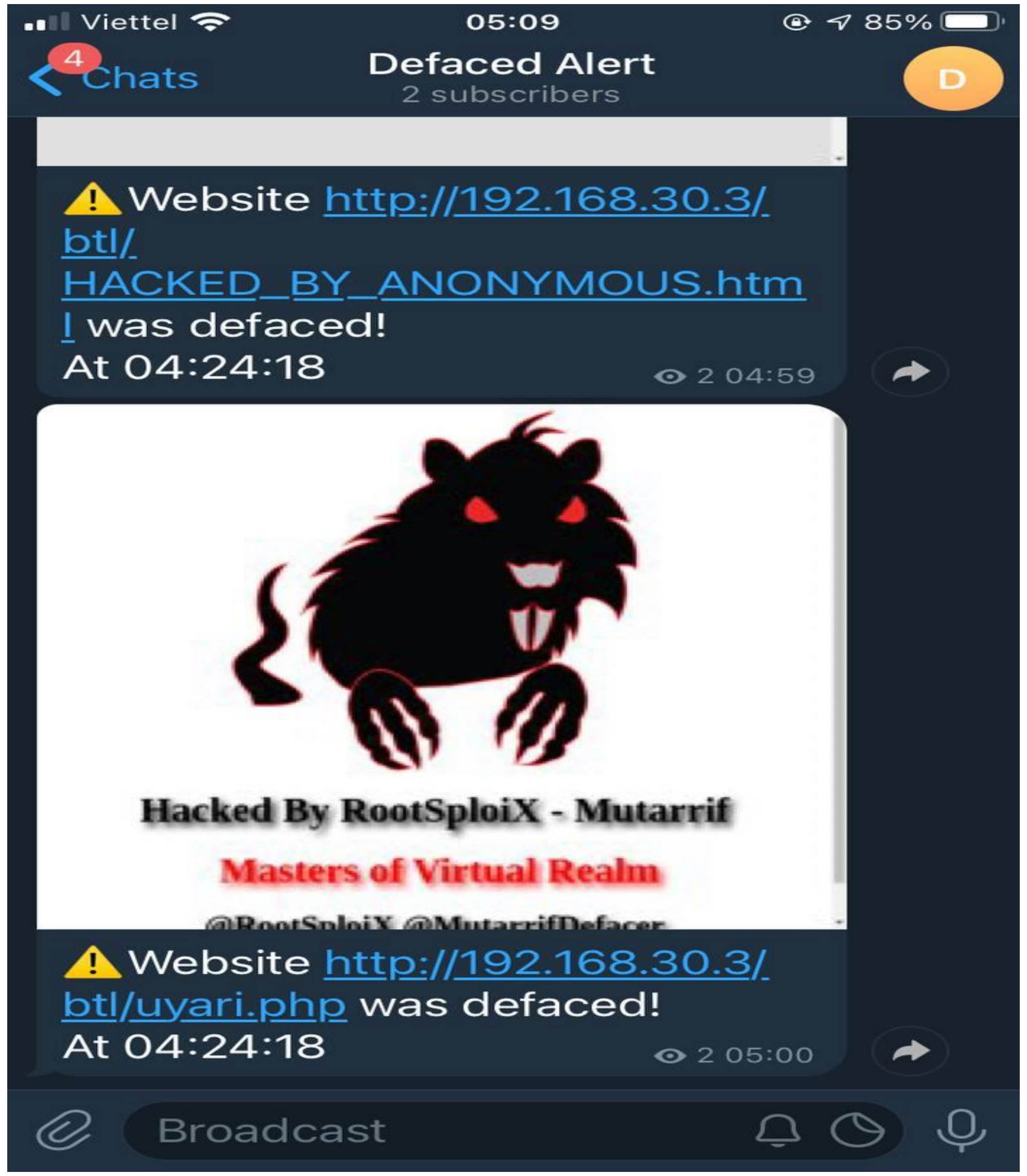
- Detect any file without registering them previously



htnguyenbg@gmail.com

You website was defaced!
URL: http://192.168.30.3/btl/login.php
Path infected: /login.php

Email Alert



⚠️Website http://192.168.30.3/btl/HACKED_BY_ANONYMOUS.html was defaced!
At 04:24:18
2 04:59

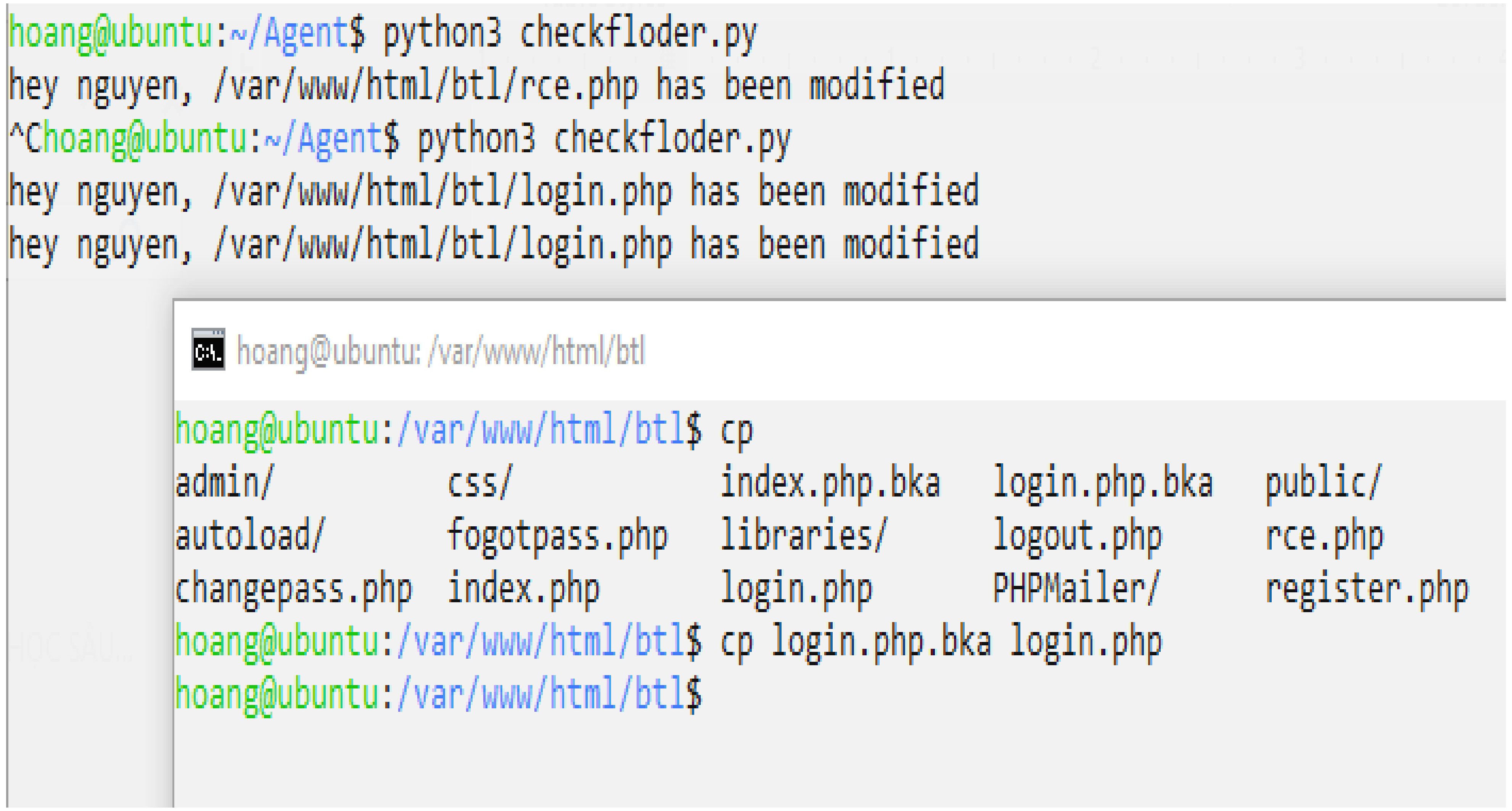
⚠️Website http://192.168.30.3/btl/uyari.php was defaced!
At 04:24:18
2 05:00

Telegram Alert

SCENARIO 2

Result 3

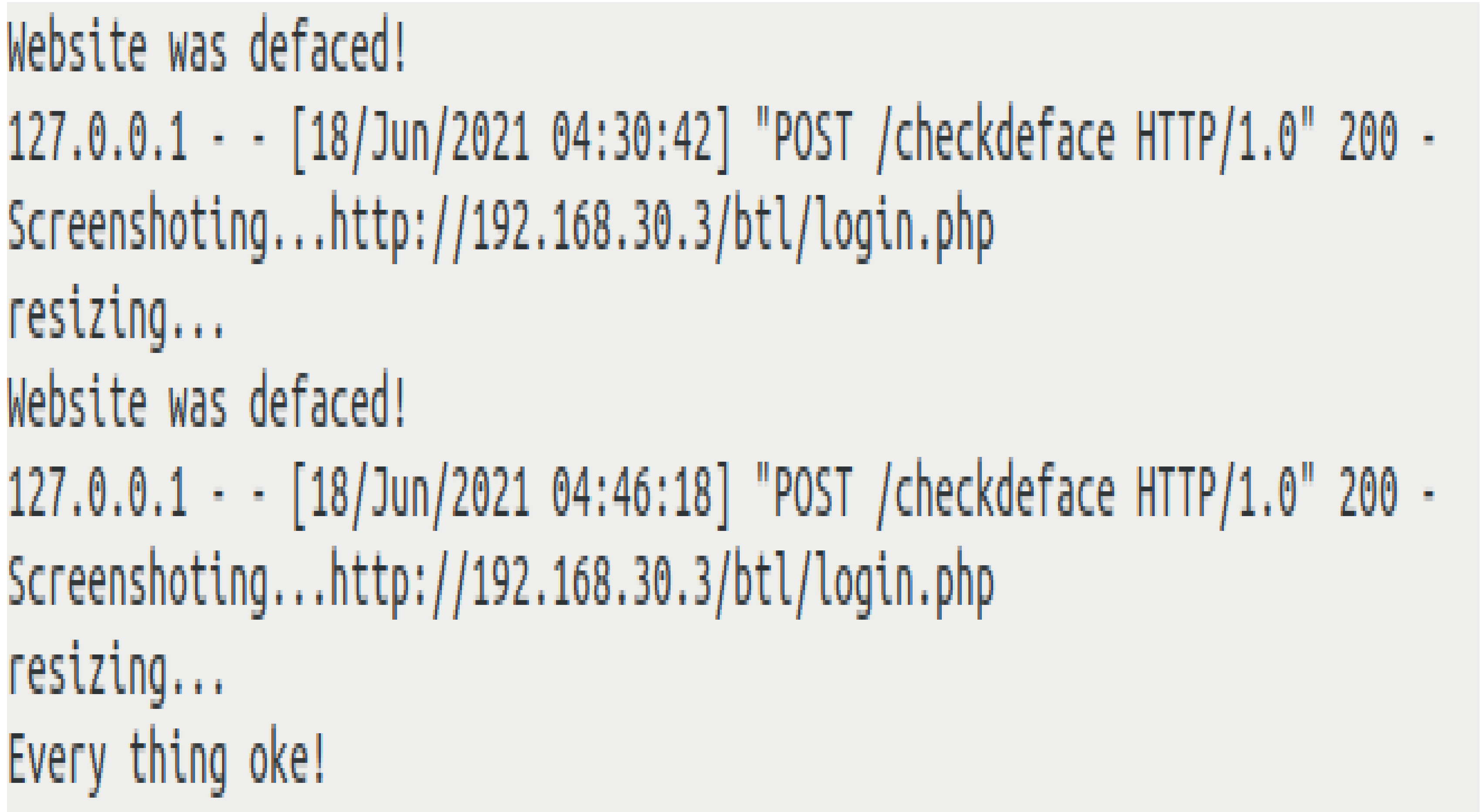
- The system does not alert when legitimate changes are made.



```
hoang@ubuntu:~/Agent$ python3 checkfloder.py
hey nguyen, /var/www/html/btl/rce.php has been modified
^Choang@ubuntu:~/Agent$ python3 checkfloder.py
hey nguyen, /var/www/html/btl/login.php has been modified
hey nguyen, /var/www/html/btl/login.php has been modified
```

```
hoang@ubuntu:/var/www/html/btl$ cp
admin/         css/           index.php.bka  login.php.bka  public/
autoload/      fogotpass.php  libraries/     logout.php     rce.php
changepass.php index.php      login.php      PHPMailer/     register.php
hoang@ubuntu:/var/www/html/btl$ cp login.php.bka login.php
hoang@ubuntu:/var/www/html/btl$
```

Backup to the original file



```
Website was defaced!
127.0.0.1 - - [18/Jun/2021 04:30:42] "POST /checkdeface HTTP/1.0" 200 -
Screenshoting...http://192.168.30.3/btl/login.php
resizing...
Website was defaced!
127.0.0.1 - - [18/Jun/2021 04:46:18] "POST /checkdeface HTTP/1.0" 200 -
Screenshoting...http://192.168.30.3/btl/login.php
resizing...
Every thing oke!
```

The server receives the event and run a check

QnA