An Efficient and Usable Client-Side Cross Platform Compatible Phishing Prevention Application

SECOND REVIEW

Guide
Dr. Angelin Gladston
Associate Professor
Department of CSE

Submitted by

N. Dhanush	2016103021
G. Santhosh	2016103057
S. Ben Stewart	2016103513

OUTLINE

1. INTRODUCTION
2. OVERALL OBJECTIVE
3. LITERATURE SURVEY
4. PROPOSED SYSTEM
5. HIGH LEVEL BLOCK DIAGRAM
6. MODULE LIST
7. IMPLEMENTATION
8. EVALUATION METRICS
9. REFERENCES

INTRODUCTION

- Phishing
- Lists of such sites
- Time constraints
- Computational resources
- Vulnerabilities
- Cross platform

OVERALL OBJECTIVE

- Create a phishing list
- Cross Platform application
- Web browser add-on
- Provide temporal resilience
- Remove false positives from list

LITERATURE SURVEY

- Previous work by Samuel Marchal, Giovanni Armano, Tommi Grondahl, Kalle Saari, Nidhi Singh, and N. Asokan
- IEEE Trans. Comput., vol. 66, no. 10, pp. 1717-1733, Oct. 2017
- Implemented a client-side phishing prevention application.
- Had background tasks communicate with a browser add-on.
- Not platform independent.

AUTOMATIC PHISHING CLASSIFICATION

- Colin Whittaker, Brian Ryner and Marria Nazif for Google
- Proc. Netw. Distrib. Syst. Security Symp., 2010
- Features used
    1. The URL of the page
    2. The HTML page contents
    3. The host server details
- Needs blacklist updating.

CANTINA

- Guang Xiang, Jason Hong, Carolyn P. Rose and Lorrie Cranor
- ACM Trans. Inf. Syst. Secur., 2011
- Page similarity
- SHA 1 algorithm
- Easy to break
- Performance gains

AUTO UPDATED WHITELIST

- Ankit Kumar Jain and B. B. Gupta
- EURASIP J. Inf. Secur., vol. 2016, no. 1, Dec. 2016
- Whitelist
  a. the domain name
  b. the IP address
- Reverts to old system if not in whitelist

FUZZY ROUGH SET FEATURE SELECTION TO ENHANCE PHISHING ATTACK DETECTION

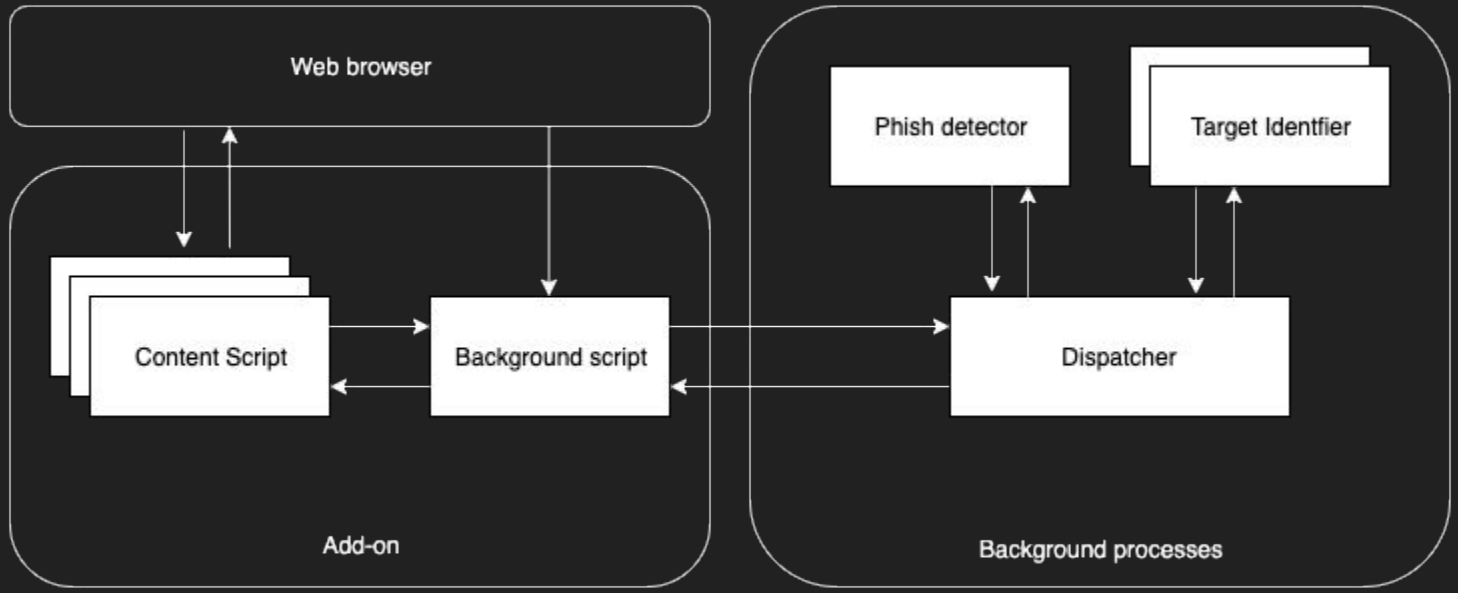
- Mahdieh Zabihimayvan and Derek Doran
- IEEE International Conference on Fuzzy Systems, June 2019
- Fuzzy Rough Set (FRS) theory
- Feature selection algorithm
- Random Forest classification
- No third party features

Paper	Journal/Conf. , Year	Contributions	Limitations
Large-Scale Automatic Classification of Phishing Pages	Proc. Netw. Distrib. Syst. Security Symp., 2010	Machine learning model can be used with reliable accuracy.	Needs blacklist for updating.
CANTINA: A feature-rich machine learning framework for detecting phishing Web sites	ACM Trans. Inf. Syst. Secur., 2011	SHA1 based similarity check for similar looking sites.	SHA1 could be manipulated.
A novel approach to protect against phishing attacks at client side using auto-updated white-list	EURASIP J. Inf. Secur., vol. 2016, no. 1, Dec. 2016	Auto-updated whitelist for faster detection of sites on average.	Not temporally resilient.
Fuzzy Rough Set Feature Selection to Enhance Phishing Attack Detection	IEEE International Conference on Fuzzy Systems, June 2019	Feature selection.	Not a user oriented application.

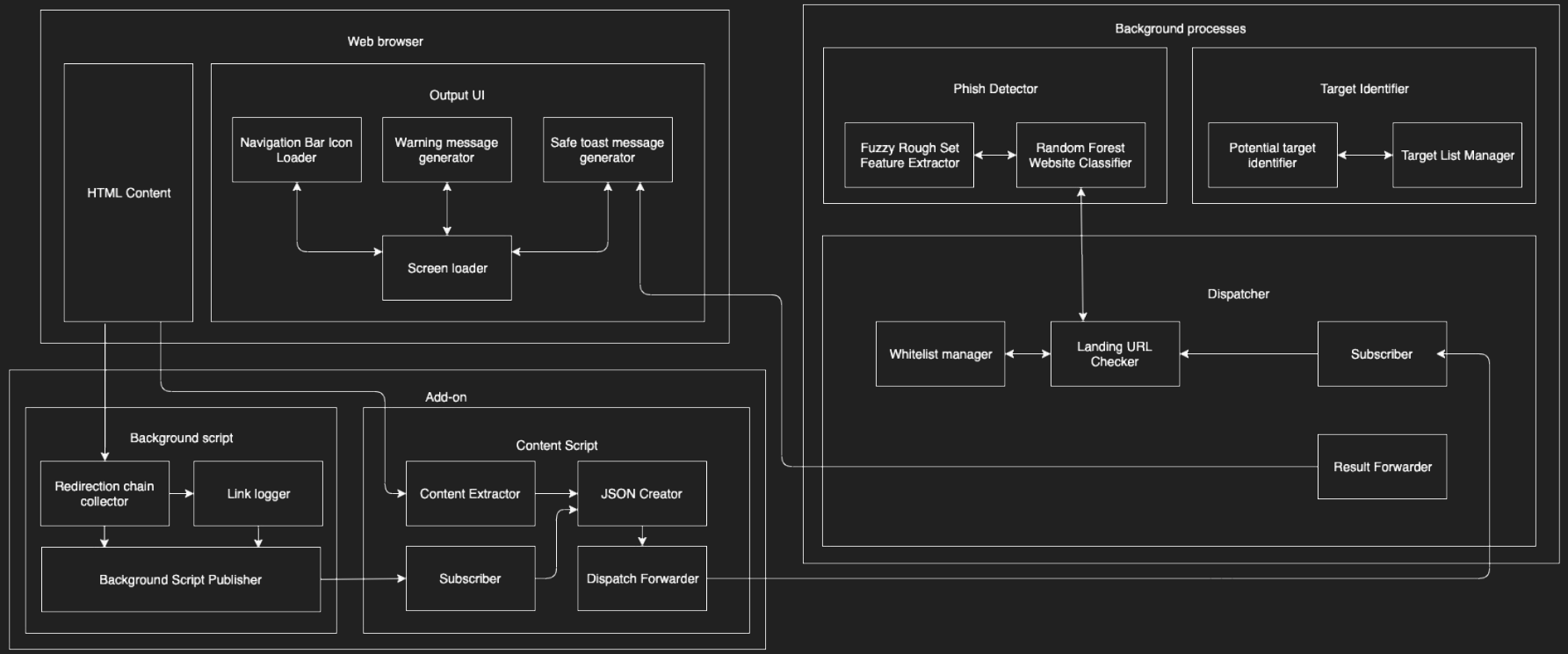
PROPOSED SYSTEM

- Platform independent
- Browser add-on
- Reduce false warnings
- Context independent detection
- Static observations

SYSTEM ARCHITECTURE



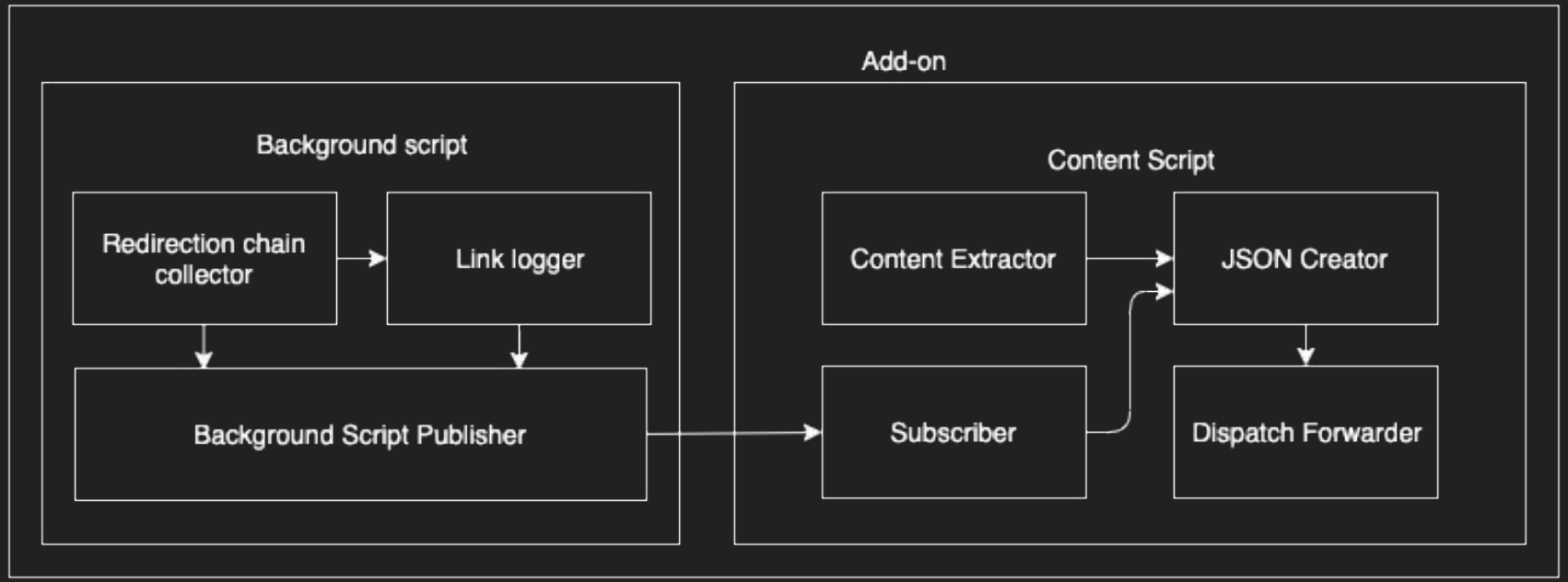
HIGH LEVEL BLOCK DIAGRAM



MODULE LIST

- Add on
  a. Background script
  b. Content script
- Background process
  a. Dispatcher
  b. Phish Detector
  c. Target Identifier
- Web Browser
  a. HTML content
  b. Output UI

ADD-ON


Add-on
Background script
Redirection chain collector
Link logger
Background Script Publisher
Content Script
Content Extractor
JSON Creator
Subscriber
Dispatch Forwarder

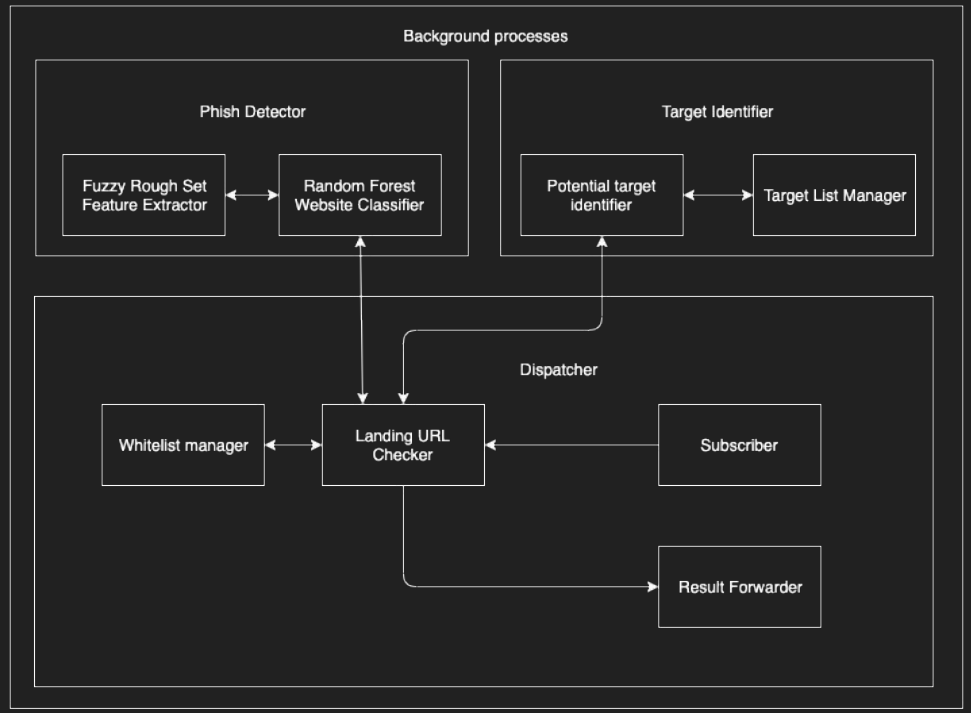
BACKGROUND SCRIPT

```
Begin
    For each page load redirect
        Add listener to that event
        Get the list of redirects from listener
        If page is fully loaded
            Send the list of redirects to content script
    Done
End
```

CONTENT SCRIPT

```
Begin
    For each page load redirect
        If page is fully loaded
            Get the URL from the tab
            Get the HTML content from innerHTML tag
            Get redirection list from background script
        Send them to the background process
    Done
End
```

BACKGROUND PROCESS



DISPATCHER

```
Begin
If page address is in whitelist
        Send the GREEN signal
Else
        Send content to phish detector
        Get results from phish detector
        If phish is FALSE
                Send the GREEN signal
Else
        Send the RED signal
        Send  content to target identifier
        If target is found
                Publish target
        Else
                No target matched
End
```

PHISH DETECTOR

Begin

For each page URL

Get the fuzzy set feature values for the URL

Load the saved random forest model

Publish the result

Done

End

FUZZY ROUGH SET

```
Begin
    Compute indiscernibility matrix M(A)
    Reduce M using absorption laws
    d - number of non-empty fields
    Initialise all fields
    For all fields
        Compute fields using formulas R=SUT
    Done
End
```

RANDOM FOREST MODEL

```
Begin
    For each record in dataset
        Get the fuzzy set feature values
        Create an arff file to save results
    Done
    Train the dataset with at least 7 splits as random forest
    Save the model as pkl file
End
```

TARGET IDENTIFIER

```
Begin
    Remove all href tags in page
    Get the hash value for page content
    Compare with values in hash list
    If match
        Display target
    Else
        No target found
End
```

SHA

Begin

Input is an array 8 items long where each item is 32 bits.

Calculate all the function boxes and store those values.

Store input, right shifted by 32 bits, into output.

Store the function boxes.

Store (Input H + Ch + ((Wt+Kt) AND 2^31)) AND 2^31 As mod1

Store (sum1 + mod1) AND 2^31 as mod2

Store (d + mod2) AND 2^31 into output E

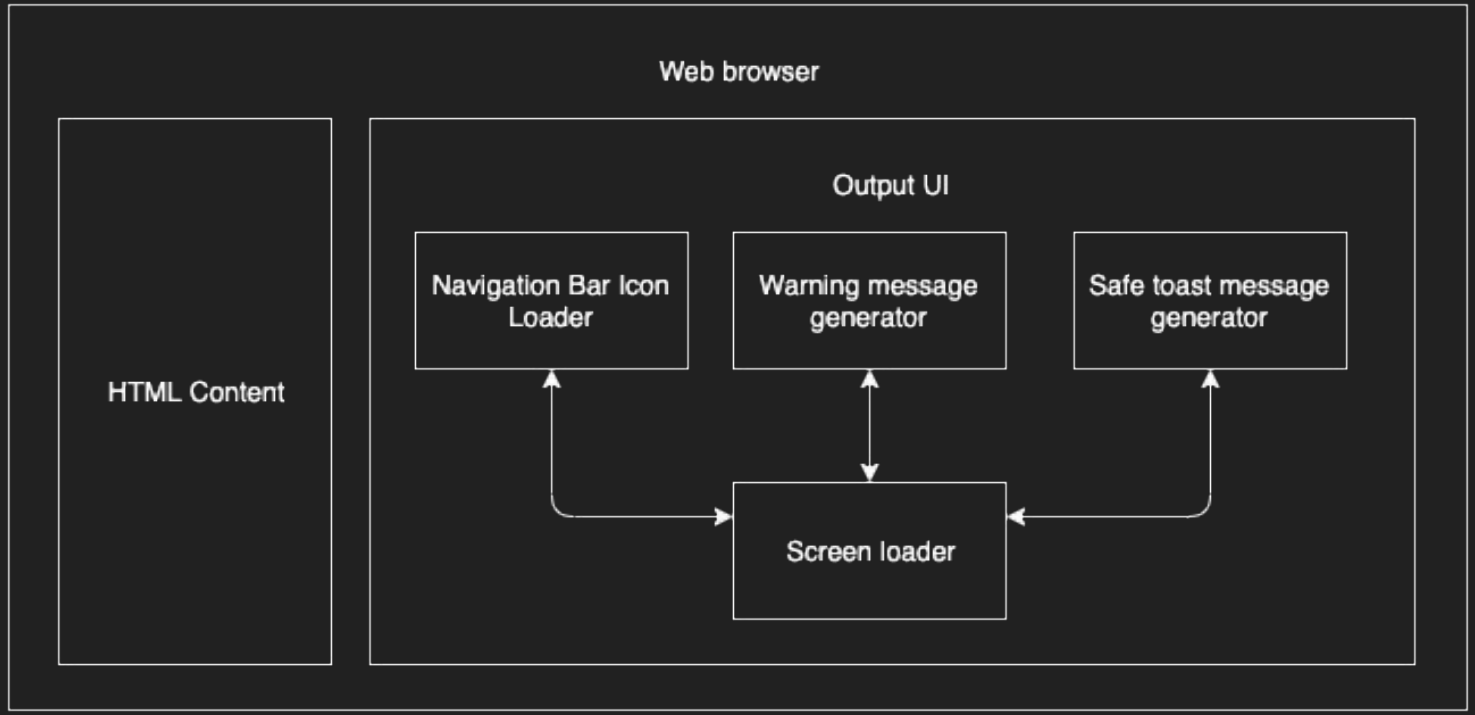
Store (MA + mod2) AND 2^31 as mod3

Store (sum0 + mod3) AND 2^31 into output A

Output is an array 8 items long where each item is 32 bits.

End

WEB BROWSER


Web browser
HTML Content
Output UI
Navigation Bar Icon Loader
Warning message generator
Safe toast message generator
Screen loader

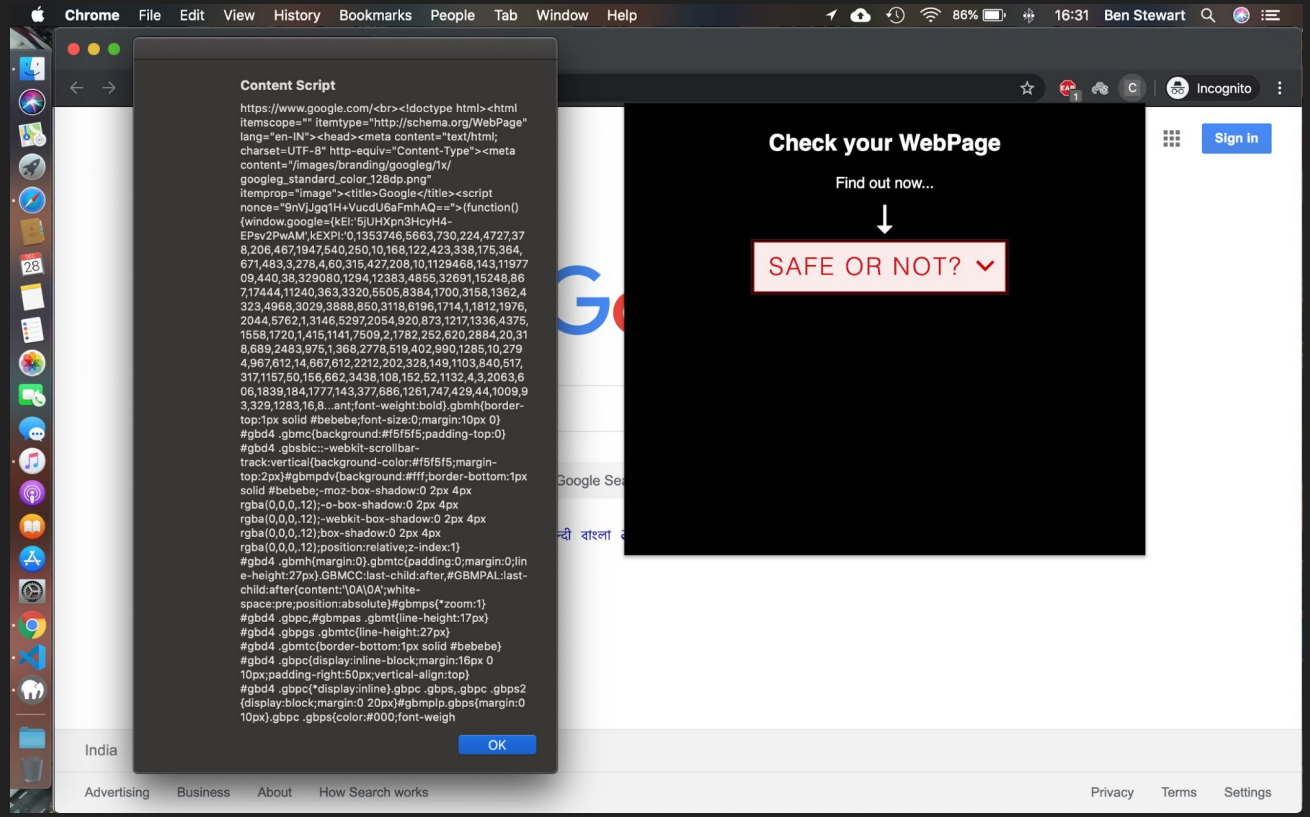
OUTPUT UI

```
Begin
    If site is phish
        Change icon to red
        Display warning message
        If site has target
            Display target link
        Else
            Display no target
    Else
        Change icon to green
        Display safe to proceed message
End
```

IMPLEMENTATION

- Add on
  a. Background script
  b. Content script
- Background process
  a. Dispatcher
  b. Phish Detector
  c. Target Identifier
- Web Browser
  a. HTML content
  b. Output UI

CONTENT SCRIPT

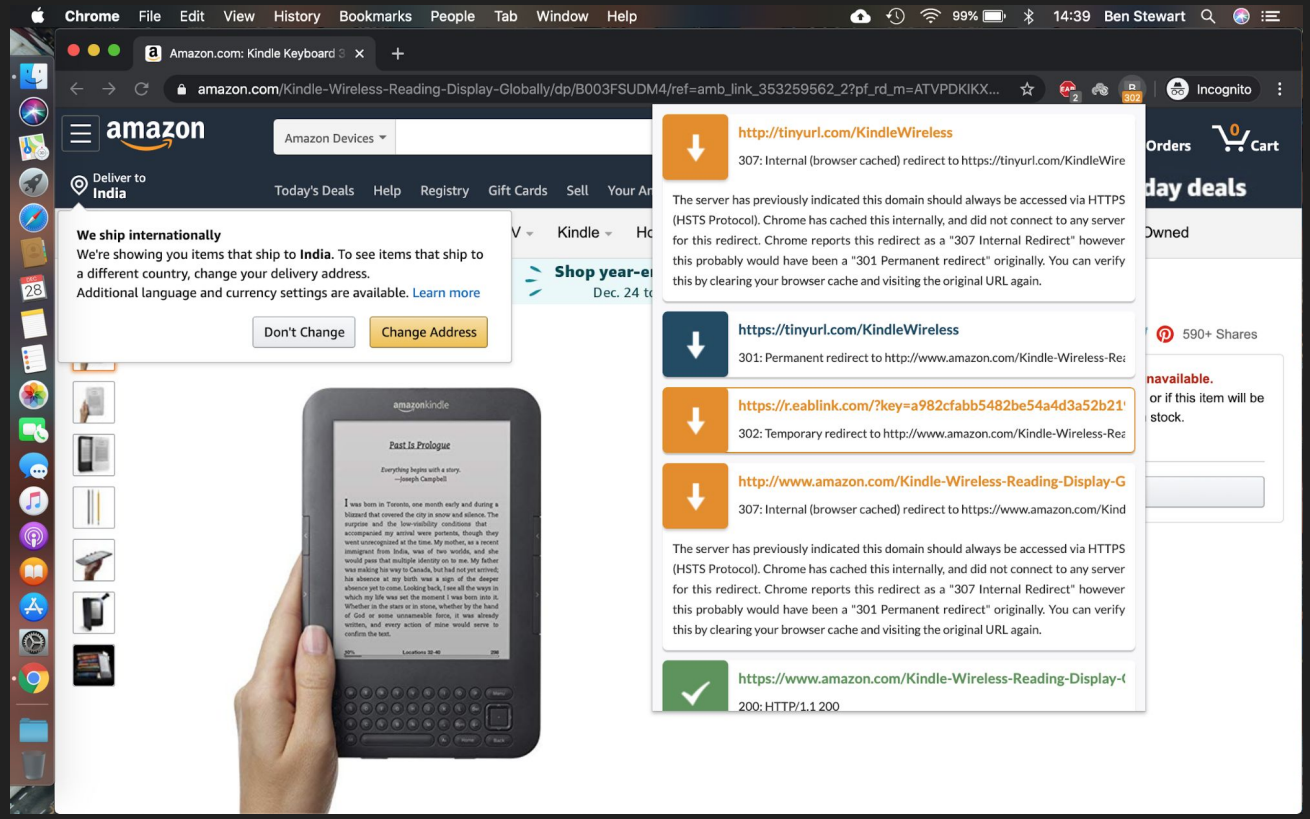


CONTENT SCRIPT

```
//Retrieve URL JS
tablink = tab.url;

//Retrieve Page content PHP
$site=$_POST['url'];
$html = file_get_contents($site);
```

BACKGROUND SCRIPT



BACKGROUND SCRIPT

```
//URL path item
url: pathItem.url,
status: pathItem.status_line,
redirect_type: pathItem.redirect_type,
redirect_url: pathItem.redirect_url,
meta_timer: pathItem.meta_timer
```

FEATURE SELECTION

```
benstewart@ben  ""/phish detector  master ●  python sample.py
(11054, 1)
(11054, 30)
[[-1]
 [-1]
 [-1]
 ...
 [-1]
 [-1]
 [-1]]
[[-1  1  1 ...  1  1 -1]
 [ 1  1  1 ...  1  1  1]
 [ 1  0  1 ...  1  0 -1]
 ...
 [-1  1  1 ...  1 -1  1]
 [ 1 -1  1 ...  1  0  1]
 [-1 -1  1 ...  1  1  1]]
```

FEATURE SELECTION

```
selector = RoughSetsSelector()
X_selected = selector.fit(X, y).transform(X)
```

RANDOM FOREST MODEL

```
[Parallel(n_jobs=1)]: Using backend SequentialBackend with 1 concurrent workers.
[Parallel(n_jobs=1)]: Done  10 out of  10 | elapsed:    0.0s finished
              precision    recall  f1-score   support

          -1       0.96      0.93      0.95       460
           1       0.95      0.97      0.96       594

   micro avg       0.95      0.95      0.95      1054
   macro avg       0.96      0.95      0.95      1054
weighted avg       0.95      0.95      0.95      1054

The accuracy is: 0.9544592030360531
[[429  31]
 [ 17 577]]
 benstewart@ben  ""/phish detector  master ●
```

RANDOM FOREST MODEL

```
//create model
clf4=RandomForestClassifier(min_samples_split=7)
clf4.fit(features_train, labels_train)
//save the model
joblib.dump(clf4, 'classifier/random_forest.pkl', compress=9)

//feature weightage
importances = clf4.feature_importances_
//confusion matrix
print metrics.confusion_matrix(labels_test, pred4)
```

TARGET IDENTIFIER

```
tags1 = get_tags(lxml.html.parse(path1))
tags2 = get_tags(lxml.html.parse(path2))
diff = difflib.SequenceMatcher()
diff.set_seq1(tags1)
diff.set_seq2(tags2)

params['url'] = url
response=requests.get(url, headers=headers, params=params)
```

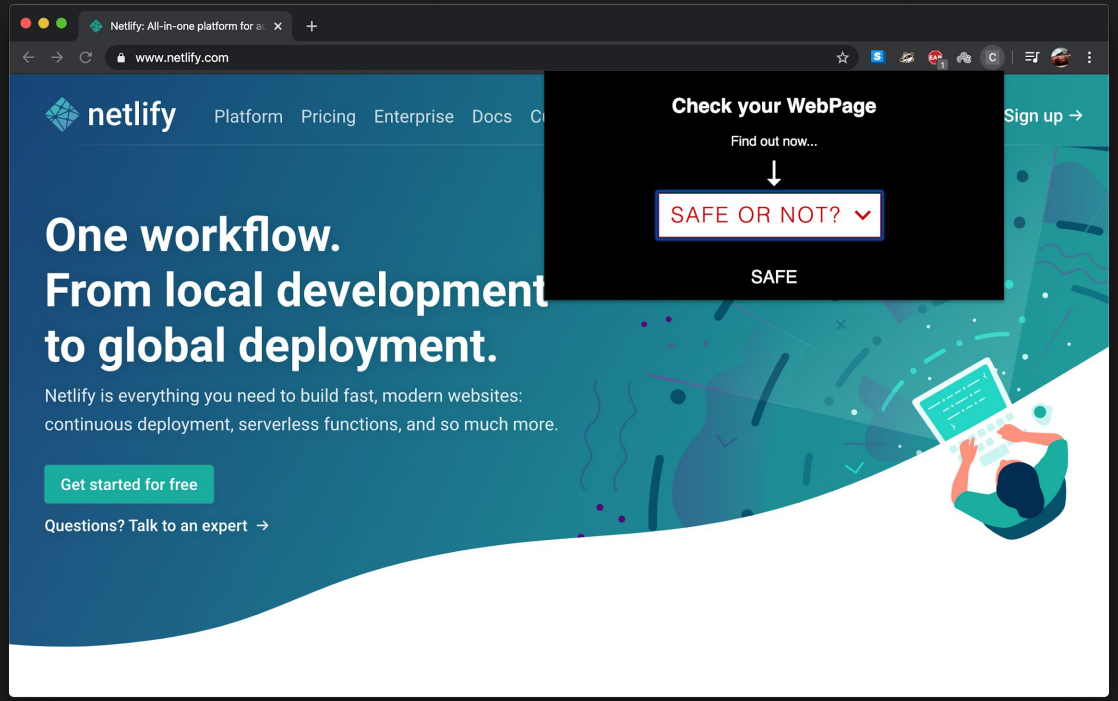
DISPATCHER

```
$decision=exec("python test.py $site 2>&1 ");
echo $decision;
```

OUTPUT UI


Check your WebPage
Find out now...
SAFE OR NOT?
PHISHING and the most similar site is verizon with
similarity 23.4957020057

HTML CONTENT



EVALUATION METRICS

1. Phish detection accuracy

Accuracy = (TP+TN)/(TP+TN+FP+FN)

2. Target detection ratio

Detection Ratio = (TP)/(TP+TN+FP+FN)

3. Memory usage profiling

Current total memory usage = Total Memory - (Free + Buffers + Cached)

4. Addon rendering time

Rendering Time = End time - Start time

5. Temporal resilience accuracy

Accuracy = (TP+TN)/(TP+TN+FP+FN)

REFERENCES

1. Mahdieh Zabihimayvan and Derek Doran, "Fuzzy Rough Set Feature Selection to Enhance Phishing Attack Detection", IEEE International Conference on Fuzzy Systems, June 2019.
2. S. Marchal, G. Armano, T. Gröndahl, K. Saari, N. Singh, N. Asokan, "Off-the-hook: An efficient and usable client-side phishing prevention application", IEEE Trans. Comput., vol. 66, no. 10, pp. 1717-1733, Oct. 2017.
3. A. K. Jain, B. B. Gupta, "A novel approach to protect against phishing attacks at client side using auto-updated white-list", EURASIP J. Inf. Secur., vol. 2016, no. 1, Dec. 2016.
4. G. Xiang, J. Hong, C. P. Rosé, L. Cranor, "CANTINA: A feature-rich machine learning framework for detecting phishing Web sites", ACM Trans. Inf. Syst. Secur., vol. 14, no. 2, 2011.
5. Implementation for the Usage of Google Safe Browsing APIs (v4), 2019, [online] Available: https://github.com/google/safebrowsing.
6. C. Whittaker, B. Ryner, and M. Nazif, “Large-scale automatic classification of phishing pages,” in Proc. Netw. Distrib. Syst. Security Symp., 2010, pp. 1–14.