**Browser, Operating System and Application (BOA) Next Generation Detection- Report**

Omri Yonatani 206265233

Maxim Raykher 311620108

**What has been done In the field in the past?:**

According to the article, some researchers have tried to classify and analysis encrypted traffic or data, by different methods. For example:

**Liberatore and Levine** showed the effectiveness of two traffic analysis techniques for the identification of encrypted HTTP streams.

One of them based on a naïve-Bayes classifier and the other on Jaccard's coefficient similarity measure.

**Panchenko and his colleagues** showed that a SVM classifier is able to correctly identify web pages, even when the user uses encryption and anonymization networks such as Tor.

**Cai et al** presented a web page fingerprinting attack and showed that it is able to overcome defenses such as a application level defense HTTPOS and randomized pipelining over Tor.

**Saltaformaggio et al** presented NetScope, a passive framework for identifying user activities within the wireless network traffic base on inspecting IP headers.

**Conti and his coworkers** devised a highly accuracy classification frameworks for various user mobile actions and applications using network features such as size, direction (incoming/outgoing) and timing.

**Anderson and McGrew** presented an effective approach to OS passive fingerprinting that uses a combination of encrypted data (TLS+TCP/IP) with non-encrypted data (HTTP) over multi-sessions. In addition, they showed OS classification based on protocol parameters (TCP handshake) or non-encrypted data.

Etc..

**Based on that related works**, we use Anderson and McGrew method. Anderson and McGrew used OS fingerprinting and protocol parameters like TCP handshake to classify the OS.

In section 2 we used 1CNN, 2CNN and Distiller models, with or without the handshake of TCP/TLS.

The handshake data can point for example to the app's category, OS fingerprint and browser's type.

Further more we can see the results, when we filtered the handshake- we got ~60% accuracy instead of ~90%.

**Data exploration-**

Our data doesn't need some data exploration, because we used the same features and methods of the original articles.

In section 1,2 and 3- we used BOA features.

In section 4- we used MAppGraph features.

(just because we compared the articles and for reliable results we need to use the same features)

**Summary Results –**

**Section 1: 1CNN, 2CNN and Distiller models on BOA's dataset**

Train:Test ratio is 80:20 in all our work

Number of samples for each label-

A picture containing text, receipt, document

Description automatically generated

**M1CNN-**

Features: udps.n\_bytes- Is the first 784 payload bytes of the flow\session

K-Fold: 5-fold

Number of epochs: 35 epochs

We used python libraries such as:

SKLearn, Tensorflow, Numpy and Pandas.

Confusion matrix:

Chart

Description automatically generated

Accuracy: 0.9406

Precision: 0.8480

Recall: 0.8618

F1: 0.8491

The model classify application and OS very bad when the OS is Mac.

However when the OS is Mac, OS and browser classify is very good.

The model classify windows OS good, but have a problem with the browser and application.

**M2CNN**-

Features: udps.n\_bytes- Is the first 784 payload bytes of the flow\session

K-Fold: 5-fold

Number of epochs: 35 epochs

We used python libraries such as:

SKLearn, Tensorflow, Numpy and Pandas.

Confusion matrix:

Chart

Description automatically generated

Accuracy: 0.9609

Precision: 0.7841

Recall: 0.8139

F1: 0.7753

This model is very similar to M1CNN, the problem is visible when we try to classify Mac OS with application.

The browser got good results.

This models results is little bit difference than M1CNN.

**Distiller**-

Features: udps.n\_bytes, udps.protocol\_headers, udps.stnn\_image.

Udps.n\_bytes- Is the first 784 payload bytes of the flow\session

Udps.protocol\_headers- 32X4 matrix that contains:

direction, payload size, delta time and tcp window size of the 32 first packets.

udps.stnn\_image- 5X14 matrix that contains statistical features.

K-Fold: 5-fold

Number of epochs: 30 epochs at most of parts, except from the fine tuning.

We used python libraries such as:

SKLearn, Tensorflow, Numpy and Pandas.

Confusion matrix:

Chart

Description automatically generated

Accuracy: 0.9827

Precision: 0.9412

Recall: 0.9460

F1: 0.9421

Distiller model classify perfectly Linux and Mac OS. Windows is the problematic with the browser and application.

**Conclusion**:

All the models classify the OS relatively well, but the browser and application less.

Distiller, classify in addition the browser.

**Section 2: With TCP and TLS handshake VS without**

We will show the results for each model:

**M1CNN- Filtered**

Table

Description automatically generated

Accuracy: 0.6093 (Unfiltered 0.9406)

Precision: 0.2897

Recall: 0.3453

F1: 0.2831

The model classify Mac OS good only with the Safari browser, and don’t successful classify the application.

Windows and Linux got good results, but the browser and application is problematic.

* As you can see, in the right bottom corner- the blue area, that shows about Windows & IExplorer.

**M2CNN- Filtered**

Table

Description automatically generated with medium confidence

Accuracy: 0.6073 (Unfiltered 0.9609)

Precision: 0.2675

Recall: 0.3366

F1: 0.2639

Same as M1CNN. Little bit better..

As we can see, the handshake changes the accuracy up to 36%, although this only 6 packets of each PCAP file. 1CNN decrease in 34%, while 2CNN in 36%.

**Distiller- Filtered**

Timeline

Description automatically generated

Accuracy: 0.8247 (Unfiltered 0.9827)

Precision: 0.6569

Recall: 0.6653

F1: 0.6534

This model was adversely affected in 16%.

The model classify Mac OS and browsers good.

Linux and Windows are better in OS classify, but the browser and application is problematic.

**Conclusion**:

The handshake is significant to the Machine learning models, because of the details, fingerprints and the properties of each tuple parts (Browser, OS, Application).

For example, the handshake different from OS to OS and there are some fingerprints on the beginning of the packets.

**Section 3: BOA's result VS separated models result**

Train:Test ratio is 70:30

Labels type-

Text

Description automatically generated

**Text

Description automatically generated**

**Text

Description automatically generated**

**Text

Description automatically generated**

**Browser-**

**Table

Description automatically generated**

Chart, bar chart

Description automatically generated

As you can see, the Safari browser got the best classification result, with the Explorer behind him with scores: 0.985 and 0.983 respectively.

**Random forest (the best classifier) confusion matrix:**

**Chart, waterfall chart

Description automatically generated**

**Table

Description automatically generated**

**Operating-system-**

Text

Description automatically generated

Chart, bar chart

Description automatically generated

As you can see, the Windows OS got the best classification result, with the Linux behind him with scores: 0.997 and 0.97 respectively.

**Random forest (the best classifier) confusion matrix:**

**Chart

Description automatically generated**

**Table

Description automatically generated**

**Application-**

**Table

Description automatically generated**

Chart, bar chart

Description automatically generated

As you can see, the Twitter App got the best classification result, with 0.908 score.

**SVM-rbf (the best classifier) confusion matrix:**

Chart, waterfall chart

Description automatically generated

Table

Description automatically generated

**Browser, Operating-system and Application together-**

Table

Description automatically generated

Chart, bar chart

Description automatically generated

**SVM-rbf (the best classifier) confusion matrix:**

**Chart, scatter chart

Description automatically generated**

**Table

Description automatically generated**

**Conclusion:**

**When we run the models separately-**

Browser- Safari is the best

OS- Windows is the best

Application- Twitter is the best

* The difference between Safari and Explorer is 0.02%, is a minor detail.

**And when we run all the models together-**

Browser- Explorer is the best

OS- Windows is the best

Application- Twitter is the best

(The Tuple- <Windows, Explorer, Twitter>)

* Run time: when we run the models together the run time is increasing. At the separately run- is the same labels, but the multiclass is smaller.
* The number of the samples for each machine is unbalanced and this can be the reason to the changes between the results. (Safari vs Explorer).

**Section 4: MAppGraph VS BOA**

Train:Test ratio is 80:20

Number of epochs: 150

Number of features: 63

The features:

Table

Description automatically generated

The features is classified to 4 categories:

Aggregated Features: features like sum, total, mac, min and values basic features..

Temporal Features: flow duration, mean and standard deviation of flow duration.

Statistical Features: mean, median absolute deviation, variance, etc..

Categorical Features: transmission protocol and IP address.

**Results:**

**We didn’t succeeded to complete this section due to lack of time and resources and unexpected conditions.**

**These results are on MAppGraph reduced data-set**

Graphical user interface

Description automatically generated with medium confidence

We can see while more number of epochs, the accuracy increase and the loss decrease.

**Confusion-Matrix:**

A picture containing text

Description automatically generated

The accuracy of the model is 93% (0.9324) on 16,561 samples.