Poster: Machine Learning-Based Fingerprinting of Network Traffic Using Programmable Forwarding Engines

Greg Cusack
University of Colorado
gregory.cusack@colorado.edu

Oliver Michel
University of Colorado
oliver.michel@colorado.edu

Eric Keller
University of Colorado
eric.keller@colorado.edu

***Abstract*—With the recent development of programmable forwarding engines (PFEs), systems designers are now able to extract information-rich, flow records at high rates of speed. The growth of PFEs and rich flow generation systems, provide us with the data and speed necessary for network, flow-based fingerprinting and classification. In this project, we explore the efficacy of classifying large amounts of network traffic using PFE-generated, rich flow records. We write a stream processor and use a random forest, binary classifier to utilize these flow records in fingerprinting ransomware and Shadowsocks, a censorship circumvention tool, without requiring deep packet inspection. Our ransomware classification model achieves a detection rate in excess of 0.86, while our Shadowsocks classifier achieves a detection rate close to 0.987. Our initial results show the efficacy of utilizing high-rate, PFE-generated, rich flow records to fingerprint various types of web traffic.**

I. Introduction

In recent years, we have seen the development of a few high rate stream processing systems, which employ switch hardware to generate network, information-rich flows [1], [4]. PFEs utilize switch hardware and dynamic memory caches to achieve high packet processing speeds while simultaneously providing rich flow records. These PFE-generated flow records, provide per-packet information and allow us to process flows for various applications in an accurate and scalable manner.

In this work, we test the efficacy of using PFE-generated flow records to classify large amounts of network traffic. In order to do so, we designed two applications that apply machine learning to identify different types of network flows. Our first application is a ransomware classifier. We chose to look at ransomware due to its recent emergence as one of the most prominent strains of cybercrime [2]. Unfortunately, malware delivery is shifting heavily to HTTPS as 37% of all malware now uses HTTPS as of June, 2017 [3]. This change in delivery method renders current deep packet inspection (DPI)-based ransomware classification techniques ineffective. As a result, we utilize PFE-generated flow records to monitor high-level flow features only available in TLS traffic.

For our second application, we look into classifying a recently designed censorship circumvention tool, Shadowsocks. Shadowsocks is a SOCKS5 proxy that has gained popularity in China due to its current effectiveness in circumventing the Great Firewall of China. Shadowsocks requires a user to set up her own proxy server with her own custom configurations. The customization of the proxy server on a user to user basis makes Shadowsocks a difficult circumvention technique to detect. However, by extracting enough features, we are able to identify Shadowsocks traffic with high certainty.

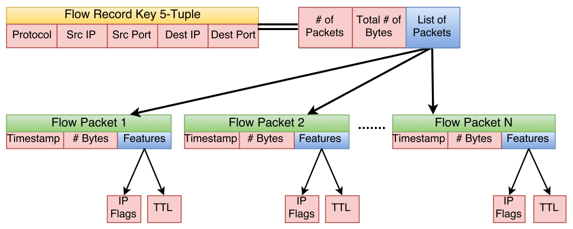


Fig. 1. Compact and per packet flow records created in a hierarchical manner. The 5-tuple serves as the key for matching packets in the same flow.

In order to provide the classifiers with testing and training data, we wrote a stream processor that extracts features from network flows. The promising results of both applications speak to the feasibility and effectiveness of using PFE-generated flow records to monitor and classify network traffic at high rates of speed. We outline the design of our stream processor in the following section.

II. Implementation

A. Flow Records and Processing Kernels

We wrote five kernels on top of the RaftLib framework for processing network data and creating compact and rich flow records. Figure 1 shows the structure of our flow record. The 5-tuple serves as a key for each flow, which links to the number of packets and bytes in the flow along with a reference to specific packet features. The packet features include the packet timestamp and the number of bytes in the packet. Each flow packet also contains a link to the packet's IP flags and time to live (TTL). We utilize the data in these flow records to extract features for our ransomware and Shadowsocks classifiers.

Figure 2 shows the kernels we wrote for flow generation and feature extraction. Normally, the per packet, flow records seen in Figure 1 would be generated in PFE hardware, but since we are reading from a PCAP, we wrote three kernels to simulate the rich, flow record generation process. The initial PCAP file reading kernel reads in a PCAP and outputs a raw packet. The raw packet parser extracts the 5-tuple from the packet and sends the 5-tuple along with the packet features as a key-value pair to the flow table kernel. We wrote a custom

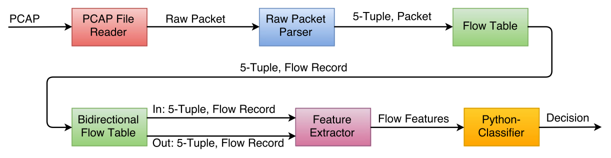


Fig. 2. All boxes except the Python-classifier are kernels we wrote for stream processing. We built the kernels to convert a PCAP to a set of flow records for feature extraction. Each kernel executes one step in the flow processing system.

flow table to do most of the packet processing and memory management. If an incoming 5-tuple already exists in the flow table, the incoming packet features are appended to the list of packets corresponding to the packet's 5-tuple key. On the other hand, a new entry is created in the flow table when a previously unseen 5-tuple is read.

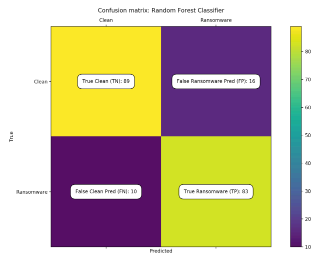
In a client's communication with a server, two flows are present. One flow corresponds to the client-to-server communication, and the other flow correlates with the server-to-client communication. In order to look at traffic burst patterns and other features requiring knowledge of corresponding flows in opposite directions, we wrote a bidirectional flow table kernel. The bidirectional flow table manages a list of flows and matches flow records with each other when an incoming flow record's source IP and source port match another flow record's destination IP and destination port and vice versa. If a flow match is not found, the incoming flow is added to the bidirectional flow table and waits for a match. If a match is found, the two flows are exported out of the bidirectional flow table to the feature extraction kernel. The feature extraction kernel takes in both flow records and performs calculations using the features as seen in Figure 1.

III. Results

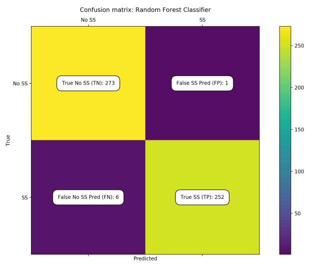
A. Classification Model

We tune our stream processor to extract features from our collected network traffic. The features are fed into each classifier, which first ensures the data contains the same number of target flows as clean flows in order to prevent classification bias. A 70-30 train/test split is used to train and test our model respectively. A 10-fold cross validation (CV) is performed on our data splitting to ensure our splitting model is unbiased. We select 28 features including packet interarrival times, inflow to outflow packet ratios, and burst lengths for classifying ransomware. In identifying Shadowsocks traffic, we extract 33 flow features, including packet interarrival times and payload entropy.

The confusion matrix in Figure 3a show the results of our ransomware classifier using 28 different features. Even with a smaller set of traffic data, ∼200MB, we are able to achieve a respectable recall of 0.87, a precision of 0.86, an F1 score of 0.87, and an AUC of the ROC of 0.92. The Shadowsocks classifier achieved stronger results as seen in the confusion matrix in Figure 3b. We achieve a precision of 0.996, a recall of 0.977, an F1 score of 0.987, and an AUC of the ROC of 0.999. The results from both classifiers show that both ransomware and Shadowsocks are vulnerable to machine learning-based attacks.



(a) Ransomware Confusion Matrix



(b) Shadowsocks Confusion Matrix

Fig. 3. Ransomware and Shadowsocks classifier confusion matrices – We achieve a precision and recall score of 0.86 and 0.87

IV. Conclusion & Discussion

Classification accuracies of 0.86 and 0.98 for the ransomware and Shadowsocks classifiers, respectively, show the efficacy of utilizing high-rate, PFE-generated, rich flow records to fingerprint different types of web traffic. However, the preliminary results also illustrate the vulnerabilities in ransomware delivery and Shadowsocks. In order to further our work, we first plan to write our classifiers in C++ to improve classification speed with the goal of fingerprinting network traffic at line rate. We will also continue to explore various classification techniques to improve our detection accuracy. It should be noted that some ransomware is delivered via UDP; therefore, adding UDP traffic support to our stream processor is vital.

V. Citations

This poster abstract is partly based off of a recently published workshop paper:

G. Cusack, O. Michel, and E. Keller. Machine Learning-based Detection of Ransomware using SDN. *ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization.* 2018. https://www.cs.clemson.edu/nss/sdnfvsec2018/program.html

VI. Acknowledgements

This work was supported in part by the NSF grants 1652698 (CAREER) and 1406192 (SaTC), and by the NSF and VMWare grant 1700527 (SDI-CSCS).

References

[1] P. Bosshart, G. Gibb, H.-S. Kim, G. Varghese, N. McKeown, M. Izzard, F. Mujica, and M. Horowitz, "Forwarding metamorphosis: Fast programmable match-action processing in hardware for sdn," in *Proceedings of the ACM SIGCOMM 2013 Conference on SIGCOMM*, ser. SIGCOMM '13. New York, NY, USA: ACM, 2013, pp. 99–110. [Online]. Available: http://doi.acm.org/10.1145/2486001.2486011

[2] Europol, "Internet organised crime assessment 2016 iocta," 2017. [Online]. Available: https://www.europol.europa.eu/activities-services/main-reports/internet-organised-crime-threat-assessment-iocta-2017

[3] A. Magnúsardóttir, "Malware is moving heavily to https," 2017. [Online]. Available: https://www.cyren.com/blog/articles/over-one-third-of-malware-uses-https

[4] S. Narayana, A. Sivaraman, V. Nathan, P. Goyal, V. Arun, M. Alizadeh, V. Jeyakumar, and C. Kim, "Language-directed hardware design for network performance monitoring," in *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*. ACM, 2017, pp. 85–98.

Machine Learning-Based Fingerprinting of Network Traffic Using Programmable Forwarding Engines

Greg Cusack, Oliver Michel, Eric Keller

University of Colorado Boulder

Goal

Explore the efficacy of classifying large amounts of network traffic using PFE-generated, rich flow records in two separate applications

- Ransomware identification and classification
- Censorship circumvention traffic fingerprinting

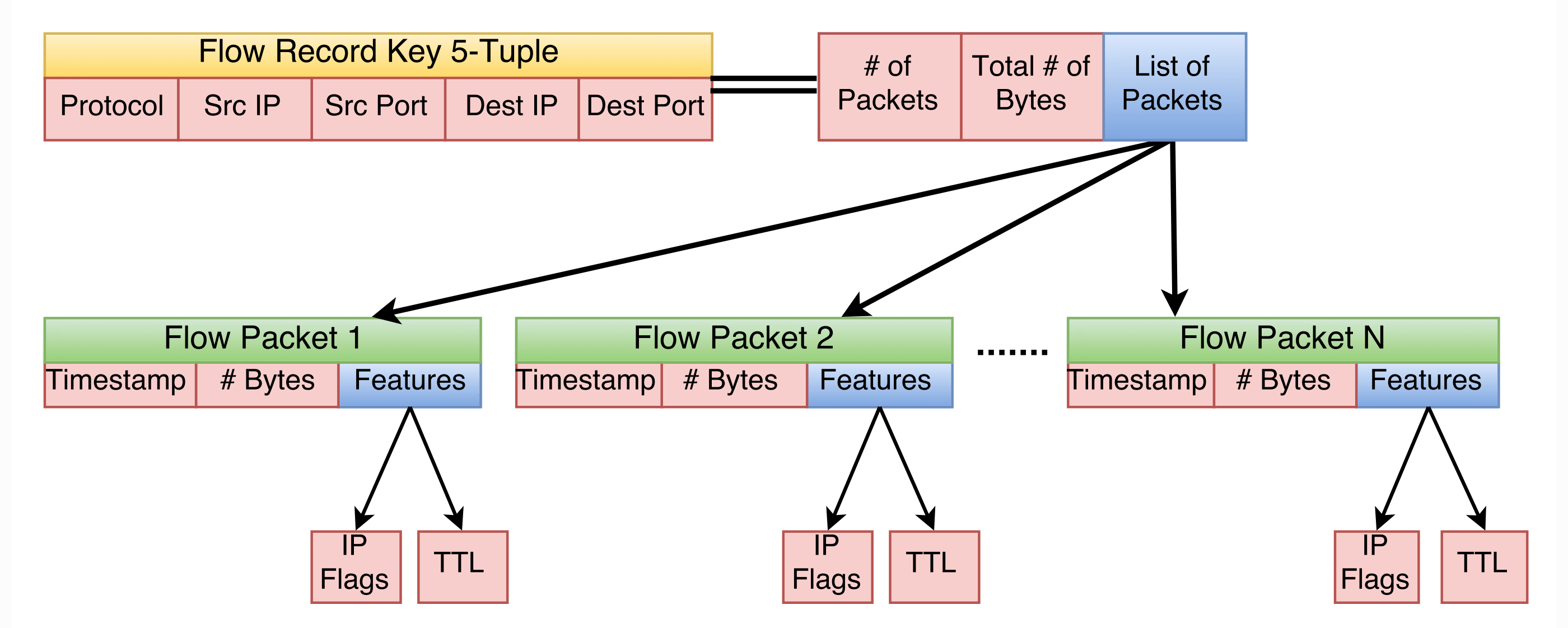
Programmable Forwarding Engines (PFEs)

- Allow commodity network equipment to support the scalable generation of rich flow records
- Stream processing systems utilize PFE switch hardware to process network data at high-rates of speed and extract vital, per-packet flow information
- Provide system designers with the data and speed necessary for network, flow-based traffic analysis and fingerprinting

Compact, per-packet flow records

PFE flow record overview

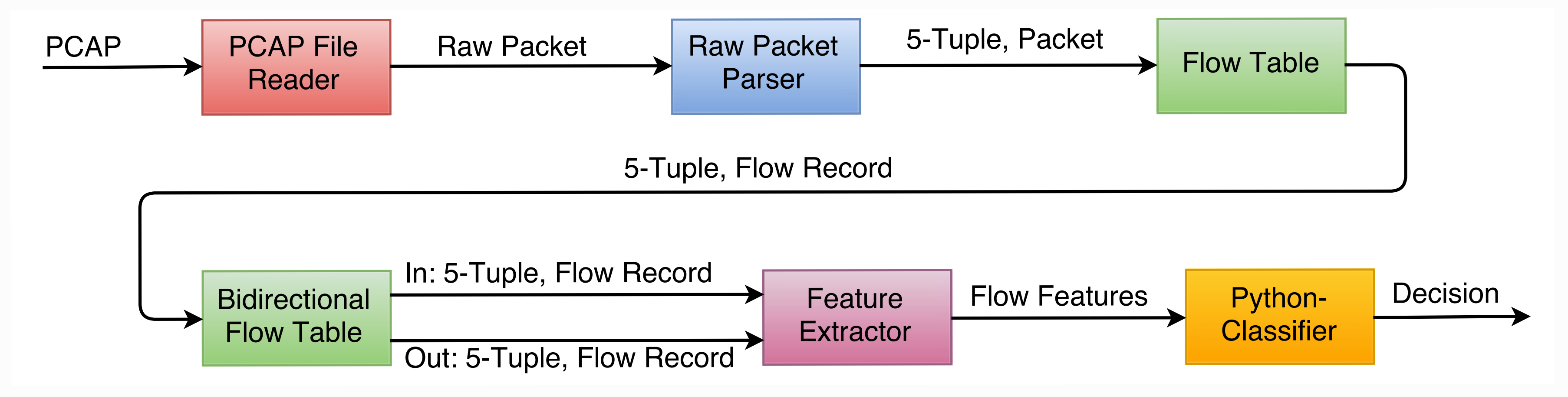
- The data extracted from a flow can be tailored to fit a user's specific application



Flow Records and Processing

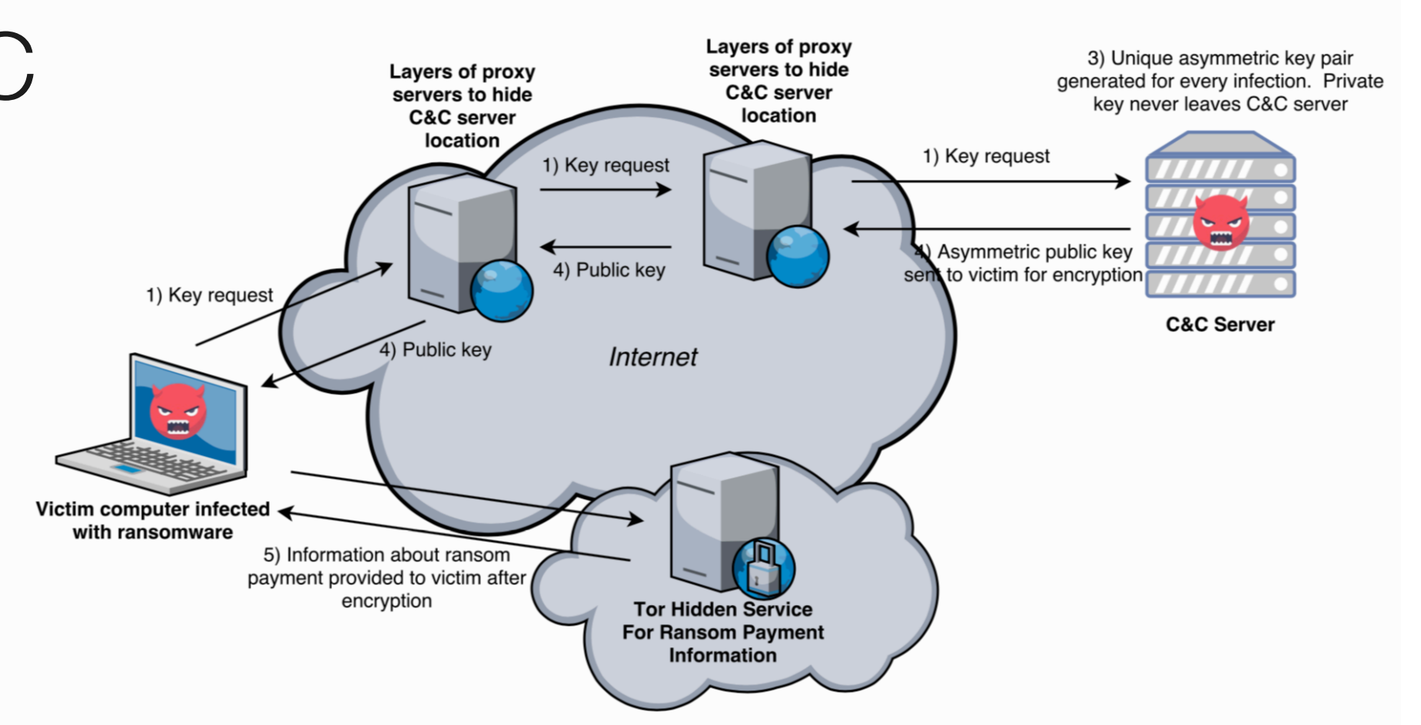
Stream processor

- 5 kernel stream processor
- Simulates PFE-generated compact, rich flow records
- Extracts vital features from flow records
- Feeds into Python classifier



Ransomware Overview

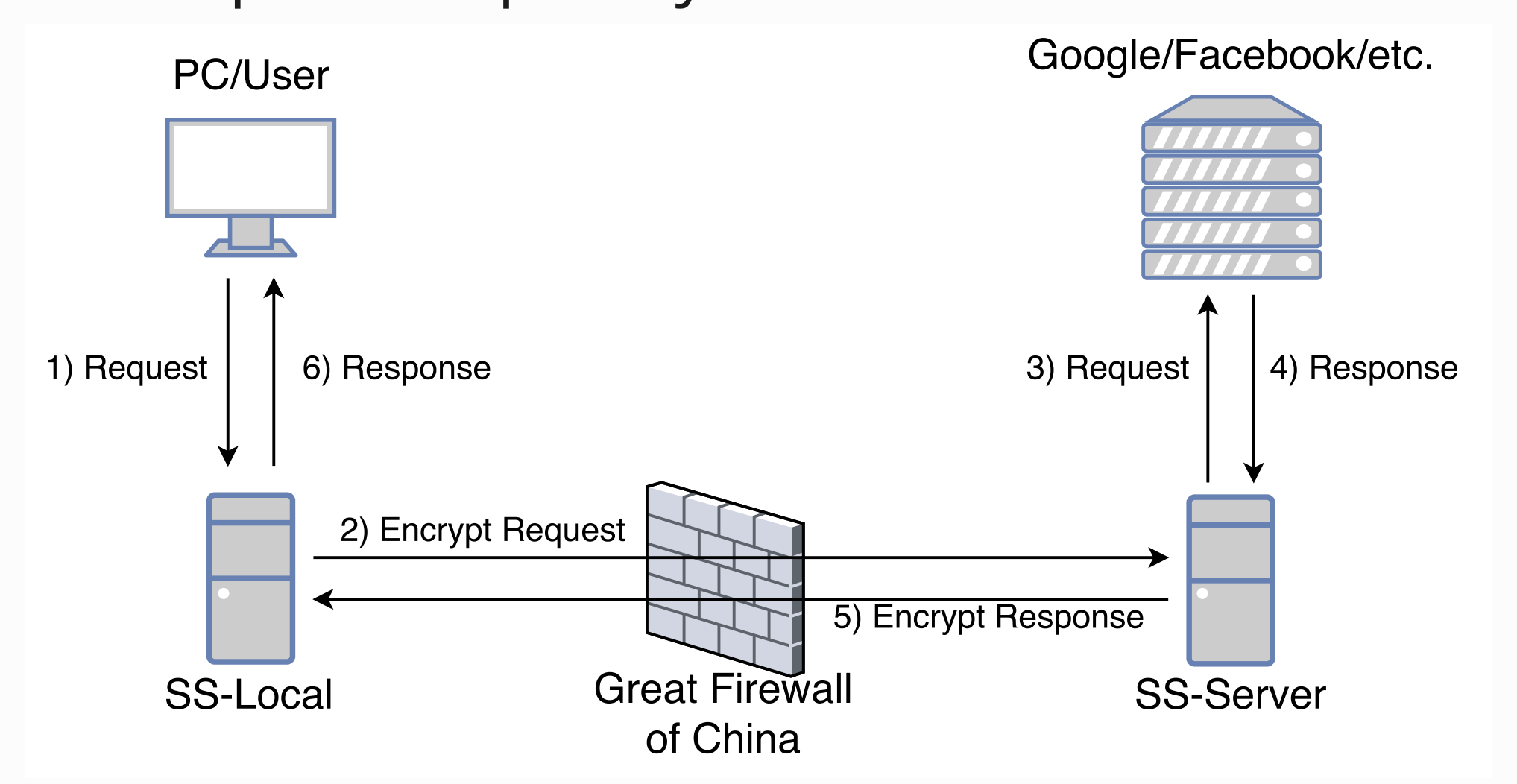
- Victim makes initial key request to C&C server
- C&C server returns encryption key
- Tor hidden service communicates method of payment


Layers of proxy servers to hide C&C server location
Layers of proxy servers to hide C&C server location
3) Unique asymmetric key pair generated for every infection. Private key never leaves C&C server
1) Key request
1) Key request
1) Key request
4) Public key
4) Public key
4) Asymmetric public key sent to victim for encryption
C&C Server
Internet
Victim computer infected with ransomware
5) Information about ransom payment provided to victim after encryption
Tor Hidden Service For Ransom Payment Information

[Cabaj, Mazurczyk. *CoRR* '16]

Shadowsocks (SS) Overview

- Censorship circumvention tool
- User sets up own proxy server outside GFW domain



Ransomware Classification Results

Goal

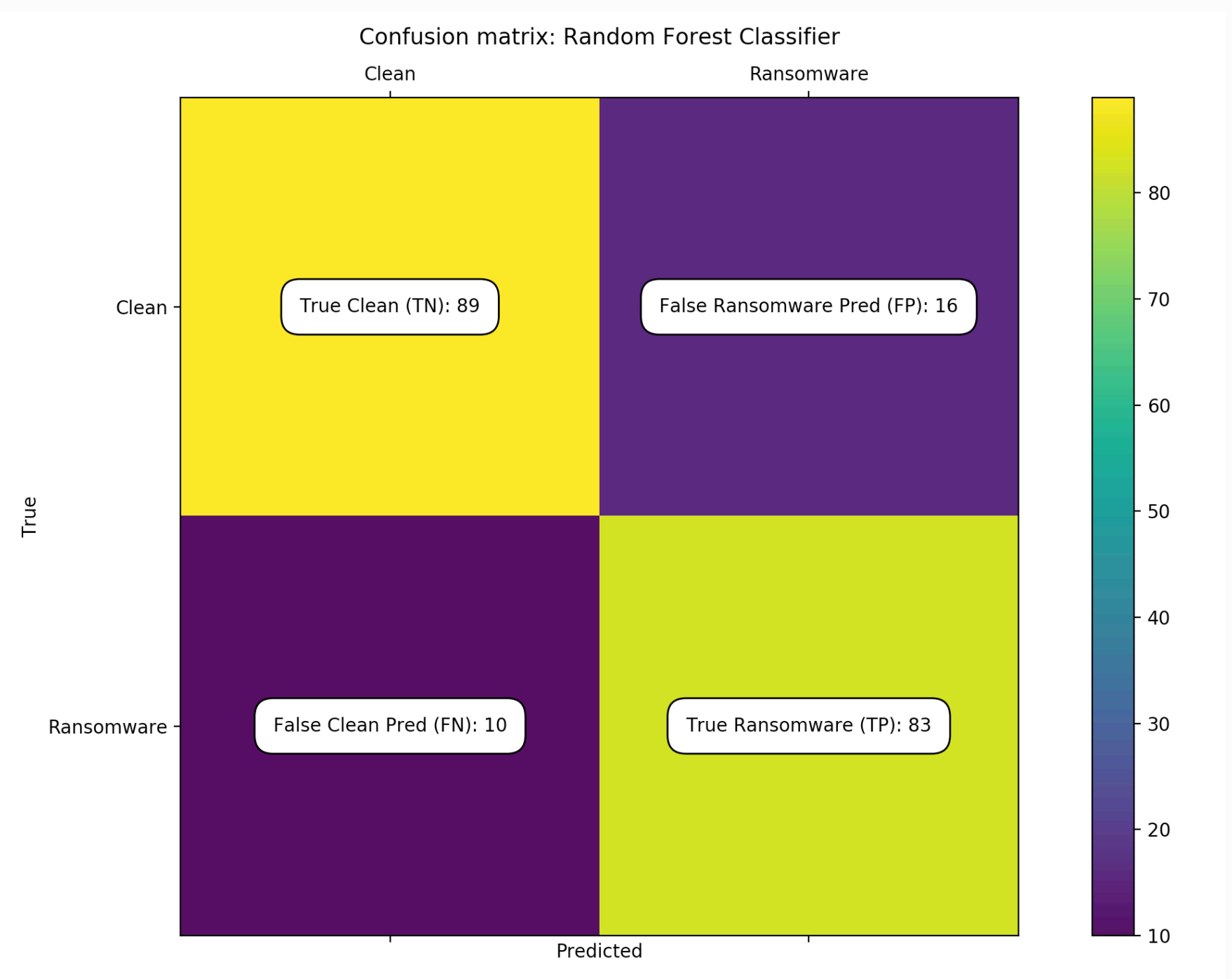
- Minimize false negatives
- Balance FNR and FPR

Random Forest Classifier

- 40 decision trees with depth 15

Performance

Precision:	0.89	F1 Score:	0.87
Recall:	0.83	AUC of ROC:	0.93



Key Flow Features

- Packet interarrival times
- Inflow to outflow packet ratios
- Burst lengths
- Flow duration

Shadowsocks Classification Results

Goal

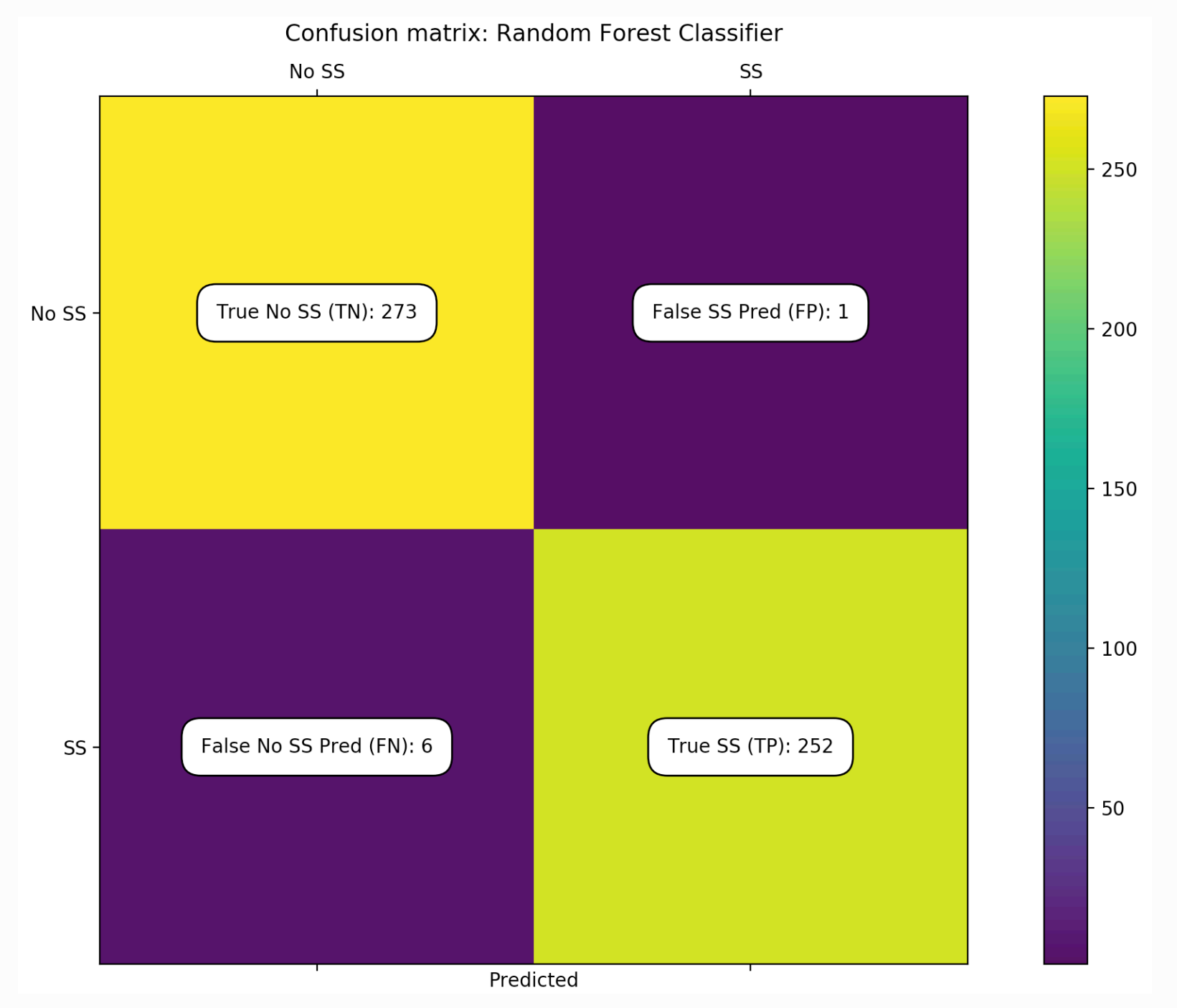
- Minimize false positives

Random Forest Classifier

- 10 decision trees with depth 10

Performance

Precision:	0.996	F1 Score:	0.987
Recall:	0.977	AUC of ROC:	0.999



Key Flow Features

- Packet interarrival times
- Traffic latency
- Payload entropy

Discussion

Takeaway

- Preliminary results show efficacy of utilizing high-rate PFE-generated, rich flow records to fingerprint different types of web traffic

Future Work

- Write classifiers in C++ for line rate classification
- Continue to explore other classification techniques

Acknowledgements

This work was supported in part by the NSF grants 1652698 (CAREER) and 1406192 (SaTC), and by the NSF and VMWare grant 1700527 (SDI-CSCS).