Detection of Pluggable Transport Techniques Report *

Zuhayr Ansari (bfq6fs), Luke Del Giudice (wcc5ub) *University of Virginia*

Censorship suppresses the free distribution of information to protect users from harmful material. What material is deemed harmful by the governments and private organizations that conduct censorship varies widely and often is motivated by a need to exert political and social control. The right to access information anonymously is therefore at odds with limiting the spread of information deemed detrimental to society. One censorship circumvention technique is pluggable transport, using domain fronting to connect to allowed websites and then being redirected to a desired destination. Snowflake is an implementation of this diverting pluggable transport strategy, using domain fronting to a broker and receiving a snowflake proxy to connect to through WebRTC. Unfortunately, recent research suggests that the DTLS handshake used by Snowflake may be easily distinguishable from other WebRTC applications, providing an opportunity for censors. In this project we attempted to replicate this detection of Snowflake and alter the DTLS handshake procedure to increase the difficulty of detection.

Introduction

We first encountered the concept of fingerprinting Snowflake on a tor message board where a user referenced that Snowflake blocking is being done in Russia by fingerprinting and blocking DTLS Handshakes. This threat to user anonymity was obviously concerning, and further investigation showed that academia had also taken notice of this vulnerability in Snowflake. In particular, the work of

To test this result After investigating further, we found that these

Previous successful attempts of detecting Snowflake, including those made by Midtlien and Palma [2] and by MacMillan et al [3], have guided our approaches to this project. Government censors have attempted to block

"show your work" initiative *American Journal of Political Science* (*AJPS*)

`geometry:`

Snowflake has recently been used to circumvent censorship in Russia and Iran, whose governments have increased efforts to block access to Tor entry relays.

Results

Mini Results

The project consisted of four stages: procuring data both through self collection and using the Macmillan database [3], using a sklearn random forest classifier model to detect Snowflake WebRTC handshakes, altering the Snowflake handshakes' padding with scapy, and finally using go language to alter the handshake packets. The MacMillan database consisted of data of about 2000 WebRTC handshakes each from Google Meet, Facebook, Discord, and Snowflake. Simulation of WebRTC handshakes was done through selenium and collected with pyshark. This combined data, of around _____ WebRTC handshakes per application, were now in a pcapng format. These packet files were then converted to csv files, with columns representing features, for ease of use with pandas. The source of the packet was represented as a 1 for snowflake and a 0 for any other application. This matrix of packets and packet features could then be fed into an sklearn random forest classifier, where packet features were predictors and the source of the packet was

*Document format based on svmille template (http://github.com/svmiller).

the outcome trying to be predicted. The random forest classifier was chosen due to its robustness to noise, improved accuracy, and reduced overfitting from its combination of multiple decision trees. With k fold cross validation this yielded a balanced accuracy of 99.85%, precision of 0.9976, and a receiver operating characteristic area under the curve, roc-auc, score of 0.9999. Therefore our classifier was successful in distinguishing between the WebRTC handshake of Snowflake and other applications, a pattern that could be used by censors to detect and block Snowflake traffic.

Now that we could detect Snowflake usage, the question was whether we could alter this handshake to avoid detection. Giving Snowflake a new pattern would not prevent censors from adjusting to the change and recognizing the traffic as belonging to Snowflake. Therefore we aimed to try and conform the Snowflake traffic to that of another user specified application. Our previous random forest model could then be used as a test of whether or not Snowflake was distinguishable from that application.

For my template, I'm pretty sure this is mandatory.[1]

Diagram of Program

Perhaps the greatest intrigue of R Markdown comes with the `knitr` package provided by Xie (2013).

[1]The main reason I still use `pdflatex` (and most readers probably do as well) is because of LaTeX fonts. Unlike others, I find standard LaTeX fonts to be appealing.

References

Xie, Yihui. 2013. *Dynamic Documents with R and knitr.* Boca Raton, FL: CRC Press.