Automated Website Fingerprinting through Deep Learning - Write-up*

Avishek Mondal

I. What has the paper contributed?

Contributed the following -

1) Automated feature engineering, using 3 deep learning methods for website fingerprinting (WF) attacks on the Tor network to deanonymise the destination that a Tor client wants to visit, and show that this is better than current "state-of-the-art" attacks
2) 2 datasets - closed-world dataset, and an open-world dataset. Each dataset consists of traffic traces of multiple visits to websites from Alexa Top Sites service.

II. What are the assumptions, and what are their strengths and weaknesses?

A. Threat Model Assumption

The paper assumes a local network adversary, who records network packets between client and guard relay. The guard relay is not pinned down, i.e. the client is free to choose different guard relays over the course of time.

1) Assessment: This model is more realistic and aggressive compared to studies where guard nodes are pinned, because this means any Tor user is vulnerable if this attack is successful. It also means the attack is not dependent on particular features of the experimental guard relay and is applicable to the whole Tor network

B. Data Collection Assumptions

Metadata from the traffic traces are captured, and the encrypted payload is discarded. The 3 broad features collected are -

1) Timing information
2) Direction
3) Size of TCP packet

Parsing is done based on the method proposed by [2], where TCP/IP packets are inspected for TLS records. Any TCP/IP packets that contain a TLS record is assumed to be a Tor cell. [2] was published in in 2013, but is still used. This was before Let's Encrypt was formed in 2016, and authentication using TLS became more widespread. If a user is also generating non-Tor network traffic, the models will mistakenly assume non-Tor traffic to also be Tor traffic.

1) Assessment: Is this still an accurate way of sniffing out Tor packets? In this use case yes it is, because all the computers are doing is running a Tor browser, so there's no other web traffic. But in a real world scenario, additional parsing will have to be done in order to make sure that it is indeed Tor cells that are being sniffed out. Additional parsing could be many things - as Prof Mittal mentioned, there are many ways of identifying Tor cells in the presence of non-Tor traffic. This includes matching destination IPs to known Tor relays etc. However, this means that the attack requires and additional layer of complexity in a real world setting.

*Paper is [1]

III. How effective is the attack?

A. How does the attack work?

The DL models are trained on a labelled dataset of traffic traces to known websites (authors call is "supervised multinomial classification"), where the target feature is the website the client visited, i.e. primary focus of target is to erode receiver anonymity.

1) Close world: Under close-world approximations, i.e. client visits sites that the adversary has trained the models on, this leads to a much better success rate of classification and lower loss rate compared to other known attacks.

2) Open World: The model output is a confidence interval as to whether the traffic generated by the client is to a receiver that is on a list of websites the adversary has already seen. This means that in an open world setting, the receiver anonymity set is essentially the universe of websites that the adversary has not trained their model on. Though this universe is large and thus preserves some level of receiver anonymity, it is likely that a nation-state adversary will have the computational resources required to significantly reduce this universe. However, the accuracy of identification **decreases** as the number of monitored websites increase. This preserves a degree of receiver annoynimity.

This attack is essentially most useful if the the adversary wants to know if a particular client has visited a particular destination. The paper assumes that all open world sites have the same prior probability of being visited. If this assumption is changed, i.e. based on the client, some websites will have a higher probability of being visited (for example a dissident visiting messaging platforms etc to communicate with the outside world), the models can predict the destination for the traffic with greater confidence.

Question to ask - How does this relate to DeepCorr?

References

[1] Vera Rimmer et al. "Automated website fingerprinting through deep learning". In: *Network & Distributed System Security Symposium (NDSS)*. 2018.

[2] Tao Wang and Ian Goldberg. "Improved website fingerprinting on tor". In: *Proceedings of the 12th ACM workshop on Workshop on privacy in the electronic society*. ACM. 2013, pp. 201–212.