Filtering automated polling traffic in NetFlow data

Nicholas Heard
Imperial College London &
Heilbronn Institute, University of Bristol
n.heard@imperial.ac.uk

Patrick Rubin Delanchy
Heilbronn Institute,
University of Bristol
pr12244@bristol.ac.uk

Daniel Lawson
Heilbronn Institute,
University of Bristol
madjl@bristol.ac.uk

***Abstract*—Detecting polling behaviour in a computer network has two important applications. First, the polling behaviour can be indicative of malware beaconing, where an undetected software virus sends regular communications to a controller. Second, the polling behaviour may not be malicious, and correspond to regular automated update requests permitted by the client; to build models of normal host behaviour for signature-free anomaly detection, this polling behaviour needs to be understood. This article presents a simple Fourier analysis technique for identifying polling behaviour, and focuses on the second application: modelling the normal behaviour of a host, using real data collected from the computer network of Imperial College London.**

I. Introduction

Monitoring computer network traffic for intrusion events is of growing importance in industry, academia and government, as dependence upon technology increasingly underpins capability. Standard rule-based computer network monitoring tools screen the traffic for matches with known malicious behavioural signatures; for example, the internet protocol (IP) addresses of internet connections can be matched against blacklists of known malicious hosts. Signature-based methods rely upon efficient updating of blacklists and have limited applicability for detecting *zero-day* malware behaviours that were previously unseen.

Monitoring for generically unusual activity offers a promising complement to signature based anti-virus tools [1] . The aim is to build models of the normal behaviour of each host on the network, and then monitor to detect significant deviations from the model. A common difficulty is that the analyst will be swamped with traffic for each host, much of it automatically generated. The focus of this article is on detecting automated polling behaviour.

Evenly time-spaced beaconing or polling behaviour from a host on a computer network can indicate that host has been infected with malware, and the regular connections are communications with a command and control server. The connections can be directed to one or many internet protocol (IP) addresses, and can go undetected amongst the bulk of other traffic flowing through that host. Not all polling behaviour in a computer network will correspond to malware. Hosts will also periodically connect to legitimate servers to query the presence of new content and keep long term connections alive. When present, legitimate polling traffic will typically account for a large proportion of a host's network connections. Understanding polling behaviour, and separating this from user-driven traffic, is therefore an important part of the model-building process when proposing statistical models of normal behaviour of hosts.

The approach presented here for detecting periodic behaviour is to sequentially apply a Fourier analysis to the counting process martingale of the events process for each IP to IP edge associated with a given host. A dominant frequency in the calculated periodogram for an edge indicates polling. The edges showing polling behaviour are separated out from the bulk data, so that the remaining data should have the characteristics of human behaviour and would admit suitable modelling. The approach is demonstrated on real data from Imperial College London's computer network, separating out the automatically generated traffic emanating from what is believed to be an uninfected host.

II. Imperial College network data

The analysis in this paper will focus on a single IP address within the computer network of Imperial College London, referred to from this point as IP X. Imperial has a range of 345,098 IP addresses, of which approximately one seventh are typically active. The average level of traffic flow on the network equates to approximately 1.3 billion NetFlow records per day. For this analysis, 97 days of NetFlow router data were collected between November 2013 and February 2014. Figure 1 plots the distribution of the time of day of the NetFlow events, and shows a typically strong sinusoidal diurnal pattern.

In contrast to this global pattern from the network, the corresponding distribution of client connection events for IP X is shown in Figure 2. This is a relatively active node, acting as the client in 2,644,780 NetFlow events within the 97 day data collection period. As such, this IP was likely to be responsible for a large amount of automated traffic. This observation is reflected by the distribution in Figure 2 which is much flatter than Figure 1 and has unusual peaks which are not consistent with diurnal human behaviour.

III. Detecting Periodicity

For two IP addresses X and Y, and for $t \geq 0$, let $N_{XY}(t)$ be the counting process of NetFlow events with source IP X and destination IP Y occurring by time t. If the internet is regarded as a graph with IP addresses as nodes, then $N_{XY}(t)$ monitors the activity on one edge of that graph. $N_{XY}(t)$ will be treated as a discrete time process: although the underlying event process may operate in continuous time, the recorded data will be rounded to some fixed level of accuracy and coincidental values will be possible. For the data analysed in this article, all event times were originally recorded to the nearest millisecond, but for computational tractability they were further rounded to the second. Given

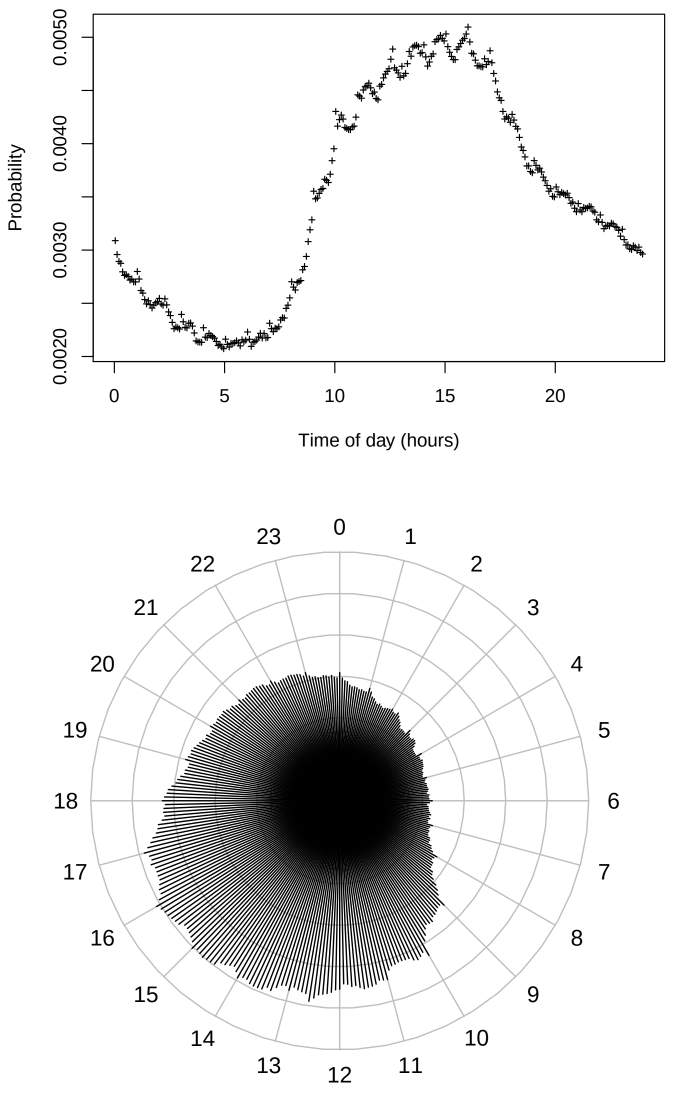

Fig. 1. The distribution of daily arrival times of NetFlow events across the whole network, estimated to five minute bins using the data obtained over 97 days, shown on [0,24] hours (top) to demonstrate the sinusoidal nature, and correctly as a circular distribution (bottom).

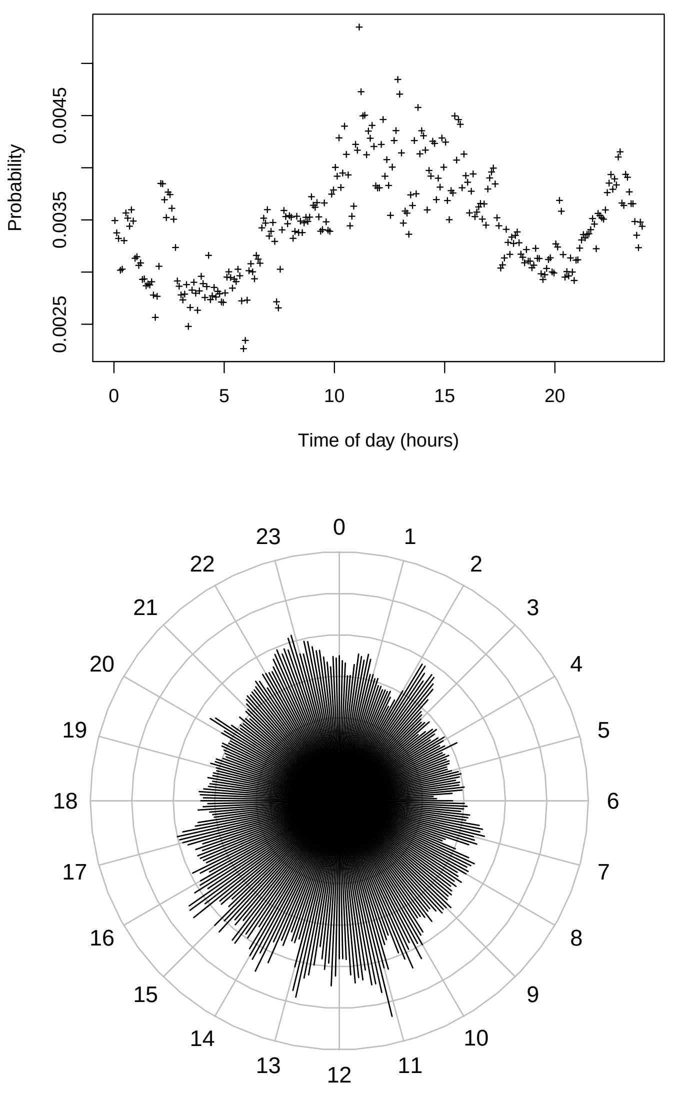

Fig. 2. The distribution of unfiltered client event times for IP X, estimated to five minute bins using the data obtained over 97 days.

that $N_{XY}(t)$ is a discrete-time process, define the increment $dN_{XY}(t) = N_{XY}(t) - N_{XY}(t-1)$.

Consideration of node (rather than edge) level activity would require a simple extension of this notation. Let $N_{X\cdot}(t) = \sum_Y N_{XY}(t)$ be the counting process of all recorded NetFlow events with source IP X, and similarly let $N_{\cdot Y}(t) = \sum_X N_{XY}(t)$ be the counting process of all events with destination IP Y. These counting processes monitor the connection activity of a node in the internet graph, either from a client or server perspective respectively. Finally, let $N_X(t) = N_{X\cdot}(t) + N_{\cdot X}(t)$ count all of the events involving IP X. All of these quantities are potentially worthy of consideration when looking to detect anomalous behaviour, depending upon the nature of the actual anomaly. However, in the subsequent analysis here attention is restricted to edges (X,Y) from the fixed client IP X of interest.

After observing $N_{XY}(t)$ for T units of time of length Δt seconds, the periodogram at frequency $f > 0$ is defined via the discrete Fourier transform [2],

$$S_{XY}(f) = \left| \sum_{t=1}^{T} (dN_{XY}(t) - N_{XY}(T)/T) e^{-i2\pi f t} \right|^2 \Big/ T. \quad (1)$$

Strong periodic behaviour at frequency f will correspond to a large value of $S(f)$. The fast Fourier transform allows $S(f)$ to be calculated efficiently for the Fourier frequencies $f \in \mathcal{F} = \{0, 1/(T\Delta t), 2/(T\Delta t), \ldots, (T/2-1)/(T\Delta t)\}$.

For testing simple periodicities of a single frequency, Fisher's g-test [3] provides a p-value of significance based on the test statistic

$$g_{XY} = \max_{f \in \mathcal{F}} S_{XY}(f) \Big/ \sum_{f' \in \mathcal{F}} S_{XY}(f'), \quad (2)$$

and this provides the basis for classifying the edge (X,Y) as periodic or non-periodic. Full details are given in [4], in the different context of identifying periodicity in gene expression time series data.

IV. Practical Considerations

The first week of data in which each edge from client IP X occurs are used to assess periodicity. With a sampling frequency of one second, this equates to series of length $T = 604{,}800$ which makes computation simple and provides a window in which the edge behaviour might be expected to be fairly stationary. Similar results were also achieved using the first two weeks of data for each edge.

In practice, rather than counting the actual number of connections observed in each interval, it proved more useful to treat the increments of the discrete time counting process $N_{XY}(t)$ as binary variables, with each increment $dN_{XY}(t) \in \{0, 1\}$ according to the presence or absence of connections from X to Y at time interval t. This way, high activity levels at a small number of time points are not able to strongly influence (1); the aim of this analysis is to detect long term periodic behaviour. As an illustrative example, the IP address 64.238.147.53 (dl.acm.org, the ACM Digital Library) appeared as the server in 176 connections from IP X in the data, but all of these connections occurred over three very short sessions of length approximately one or two seconds. The three sessions were 64 seconds and 48 seconds apart. Without taking a binary perspective, this behaviour translated to a small number of very large increments at these time points, and the frequency of 1/(16 seconds) appeared strongly represented and highly statistically significant.

The method in Section III can be readily implemented using standard software packages. Here, the fast Fourier transform calculation of (1) was performed using the python library *scipy.fftpack* (http://docs.scipy.org/doc/scipy/reference/fftpack.html) and the p-value of Fisher's g-statistic (2) was calculated using the function *fisher.g.test* from the R library GeneCycle (http://cran.r-project.org/web/packages/GeneCycle/GeneCycle.pdf).

V. Results

Table I gives details of the edges from client IP X which showed significant automated periodic behaviour according to the method from Section III, along with the corresponding periodicities. The periodic connections were largely a mixture of: connections to Imperial College administrative servers; polling requests for file updates from a running Dropbox client; and regular automatic refreshing of a live.com page in a web browser.

To illustrate the strength of the periodicity present in these edges, a typical example is displayed in Figure 3 which shows the periodogram (1) calculated for connections from IP X to a server at dropbox.com. Note the sharp peak in the periodogram at $f = 0.018 = 1/55.56$, the reciprocal of the dominant periodicity noted in Table I.

Connections from IP X to the servers identified as periodic in Table I accounted for 2,163,936 NetFlow events in the data, which was over 80% of the total client connections made by X. Figure 4 shows the time of day distribution for the remaining 480,844 client events from IP X after filtering out the periodic connections from Table I. This filtered distribution gives a picture much more consistent with human behaviour compared to Figure 2, with activity levels fairly flat outside the working hours of 9am and 6pm. It is clear that much (although not all) of the automated behaviour has now been extracted.

TABLE I. Detected periodic behaviours

Domain/description	IP addresses	Periodicities (s)
IC DNS	155.198.142.{7,8}	30.0, 30.0
IC NFS	155.198.63.{17,99}	4.00, 30.05
IC Dropbox LAN sync	155.198.199.255	31.88
IC UnixLDAP	155.198.142.5	280.00
dropbox.com	108.160.163.35	55.65
	108.160.162.98	55.66
messenger.live.com	157.56.116.202	7.71
	157.56.192.{34,48}	15.52,15.52
	157.56.126.150	15.56
	157.56.126.89	20.70
	157.56.192.160	31.18
	157.56.126.90,157.56.126.73	62.00,62.06
	157.56.192.{63,82,146}	62.03,62.26,62.26
	157.56.126.55	66.03
	157.56.126.199	119.96
	157.56.126.136	4045.20
	157.56.192.{83,95,96}	each 4086.49
	157.56.126.134	5018.00
microsoft	157.56.52.34	20.00
	157.56.52.{27,26}	136.75,604.70
	157.55.130.165	304.03
mail.live.com	213.199.179.174	307.63

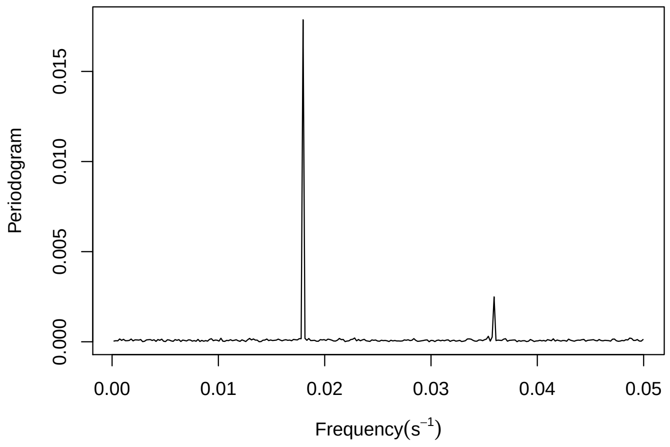

Fig. 3. Periodogram of connections from IP X to 108.160.162.98 (dropbox.com), showing strong periodicity at the frequency 1/(55.56s) (see Table I).

VI. Conclusion

A simple method using Fourier analysis has been presented for filtering automated polling behaviour from bulk NetFlow data. The method has been successfully demonstrated on real data from a client machine within an organisational computer network. Implementation was straightforward, using freely available software libraries.

The purposes of adopting this technique are two-fold. The first application is modelling normal behaviour of an IP for anomaly detection. Normal connectivity is a mixture of

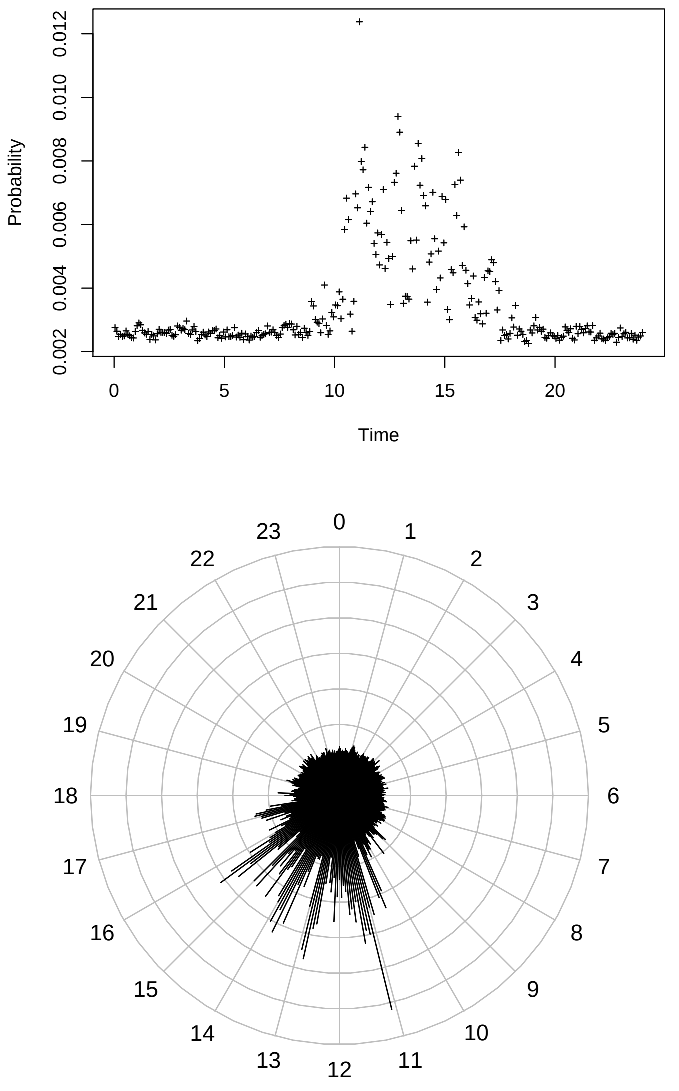

Fig. 4. The revised daily distribution of client event times for IP X after filtering out the polling behaviour from Table I.

automated and human behaviours, and separating the two will improve the analyst's ability to model both components. The second application is detecting beaconing behaviour which cannot be explained by legitimate reasons, as this potentially provides compelling evidence of the presence of malware.

REFERENCES

[1] A. Lazarevic, A. Ozgur, L. Ertoz, J. Srivastava, and V. Kumar, "A comparative study of anomaly detection schemes in network intrusion detection," in *Proceedings of the Third SIAM International Conference on Data Mining*, 2003, pp. 25–36.

[2] D. Halliday and J. Rosenberg, "Time and frequency domain analysis of spike train and time series data," in *Modern Techniques in Neuroscience Research*, U. Windhorst and H. Johansson, Eds. Springer Berlin Heidelberg, 1999, pp. 503–543.

[3] R. A. Fisher, "Tests of significance in harmonic analysis," *Proceedings of the Royal Society of London. Series A*, vol. 125, no. 796, pp. 54–59, 1929.

[4] S. Wichert, K. Fokianos, and K. Strimmer, "Identifying periodically expressed transcripts in microarray time series data," *Bioinformatics*, vol. 20, no. 1, pp. 5–20, 2004.