Modeling, Analysis, and Characterization of Periodic Traffic on a Campus Edge Network

Mackenzie Haffey Martin Arlitt Carey Williamson
University of Calgary

***Abstract*—Traffic in today's edge networks is diverse, exhibiting many different patterns. This paper focuses on periodic network traffic, which is often used by known network services (e.g., Network Time Protocol, Akamai CDN) as well as by malicious applications (e.g., botnets, vulnerability scanning). We use a simple and flexible SQL-based approach as our computational model for detecting periodic traffic, and apply it to the analysis of seven weeks of Bro connection logs from a campus edge network. Our results show that periodic traffic analysis is effective for detecting P2P, gaming, cloud, scanning, and botnet traffic flows, which often exhibit periodic network communications. We present a classication taxonomy for periodic traffic, and provide an in-depth characterization of this traffic on our campus edge network.**

I. Introduction

Traffic flows in modern edge networks are diverse and voluminous, with many different patterns [8], [12], [23], [26]. These patterns can arise from users interacting with network-based services, often in a diurnal fashion [18], or by automated systems that respond to specific events [26]. Network operators need an understanding of these patterns in order to differentiate between normal and abnormal behaviors on their networks.

The pattern of interest in our work is *periodic traffic*, in which communication events occur repeatedly at regular intervals within a specified time frame. Periodic network traffic is especially relevant in network security [4], [5], [9], [22], since it may indicate anomalous [17] or malicious [29], [30] activities. For example, many botnets are known to use periodic communications for command and control channels [10], [14], [15]. For this reason, periodicity detection is an important component in many intrusion detection systems [7], [13], [19].

Periodicity analysis can use different types of network data, such as netflow information [7], control plane information [2], and application-level information [10]. Detection techniques include statistical methods [20], spectral analysis [5], [9], and autocorrelation [30], [35]. Detecting the absence of periodicity is also important, when systems that should be producing periodic traffic cease doing so [3].

Periodicity detection is a well-established area of research. In most works, periodicity is represented as a numerical score derived from metadata associated with a set of flows. Some periodicity detection methods make a binary classification of flows as periodic or not [2], [5], [35], by comparing their periodicity score to a threshold value. Other techniques consider periodicity as a continuous gradient [10], [13], in which flows range from weakly to strongly periodic.

Despite substantial prior research on detecting periodic traffic, relatively little effort has been devoted to understanding the pervasiveness of periodic network traffic in edge networks. For example, network security researchers sometimes evaluate new detection techniques with artificially-generated datasets that exclude legitimate periodic traffic [2], [5], [9], [20]. We argue that simply detecting periodic traffic is insufficient, and that deeper understanding is needed. Such analysis is usually only a secondary consideration in prior research [12], [17], [26], while it is the central focus in our work.

In this paper, we focus explicitly on the analysis and characterization of periodic network traffic. We use a simple and flexible SQL-based method to extract periodic traffic from empirical connection-level summary data. We use our empirical data analysis to illustrate how periodic traffic reflects the typical operation of an edge network, as well as to identify malicious traffic activities on the network.

The main contributions of our paper are: (1) the implementation of a flexible, procedural, and highly-parallelizable computational model for the extraction of periodic network traffic using SQL; (2) an evaluation of the sensitivity and robustness of our SQL-based approach; (3) a taxonomical classification of periodic network traffic based on its structural properties; and (4) an in-depth characterization of the periodic traffic composition on a modern campus edge network. We also discuss the limitations of our approach, and some of its technical challenges for scalability and robustness.

The main insights that emerge from our work are:

- a modular SQL-based approach provides a flexible and powerful means by which to analyze periodic traffic;
- periodic traffic is pervasive in modern network applications, including gaming, CDN, P2P, and cloud-based services, as well as malicious traffic;
- periodic traffic is diverse in its structural properties, and often very transient in its existence, making it a challenge to detect; and
- P2P applications account for about half of the periodic traffic detected in our empirical study.

The remainder of this paper is organized as follows. Section II discusses prior related work. Section III introduces our empirical dataset. Section IV describes our method for identifying periodic traffic. Section V gives an overview of our traffic analysis results, and presents our traffic classification taxonomy. Section VI characterizes the periodic traffic that we observed. Finally, Section VII concludes the paper.

II. Background and Related Work

The study of periodic behaviors is foundational in many scientific disciplines, including astronomy [21], biology [34], and computer science [25], [35]. In our context, we are interested in periodicity that is present in network-level communication patterns.

The simplest periodicity detection techniques use statistical analysis on selected features of network traffic. Hubballi and Goyal [20] proposed a method that computes the standard deviation of the time between successive flows on the network. If the value is below a threshold, the traffic is deemed periodic; otherwise, it is not. This method was extended by van Splunder [35] to analyze netflow data from a large edge network. Eslahi et al. [10] used statistical methods to detect periodic HTTP traffic.

Spectral analysis methods are also popular for periodicity detection. AsSadhan and Moura [2] calculated the Discrete Fourier Transform (DFT) for a time series and then generated a periodogram, whose peak is then tested for significance. Similarly, Heard et al. [18] used the Fast Fourier Transform (FFT) to calculate a periodogram for time series connection data. Huynh et al. [22] computed the FFT of a time series of netflow records to produce a frequency spectrum analysis. Chen et al. [9] augmented the DFT approach by incrementally expanding the duration of the time series used, in order to find periodic patterns. Bartlett et al. [5] used wavelets to transform time series netflow data into the spectral domain for analysis.

Autocorrelation is another technique for periodicity detection. For example, van Splunder [35] calculated the autocorrelation of a time series of netflow data to identify potential periods. The periodicity of the traffic was then tested using the method presented by Hubballi and Goyal [20]. Similarly, Qiao et al. [30] used the circular autocorrelation function to identify candidate periods, and assess the periodicity with different algorithms. Gu et al. [15] calculated autocorrelation on time series data, focusing on the number of peaks. Traffic is considered periodic if the autocorrelation is strong enough.

While there has been considerable research on periodicity detection, less attention has been devoted to understanding periodic behavior. For instance, in their characterization of video game traffic, Feng et al. [12] indicated that this traffic was highly periodic. Periodicity also arises in some video streaming services [31]. Nikaein et al. [26] identified periodic traffic as a subset of machine-type communications. He et al. [17] noted that periodic traffic can indicate network congestion. While periodicity was observed in these specific domains, it was not explored more broadly.

A few efforts have been made to understand periodicity in general, but these studies are rather limited. Bartlett et al. [4], [5] developed a periodicity detector, and identified several broad classes of periodic traffic, including regular OS updates, P2P traffic, and adware. Others used these insights to refine periodicity detection [29], [30], though the characterization of this traffic was not explored further. Heard et al. [18] recognized the need to explore the composition of periodic traffic. However, they only observed the traffic of a single desktop computer for one week. Such a study is too small for a comprehensive understanding of periodic traffic on a modern edge network, such as a campus edge network.

III. Empirical Dataset

In this paper, we study the University of Calgary's network, as an example of a typical campus edge network. Our network is used by 32,000 undergraduate/graduate students and 3,000 faculty/staff. Our data was collected from a mirrored stream of all the traffic that passes through the edge routers on our campus network. Data collection was conducted for a 7-week period from January 1, 2017 to February 17, 2017. The winter academic term started on January 8, and the mid-term break occurred on February 18, so classes were in session throughout most of the data collection period.

Figure 1 shows a stacked graph view of our edge network usage during our measurement period. Inbound traffic (upper axis) peaks daily near 4 Gbps, while outbound traffic (lower axis) rarely exceeds 1 Gbps. Traffic volumes are highest on weekdays when students are on campus. TCP is the dominant transport protocol, with UDP a distant second.

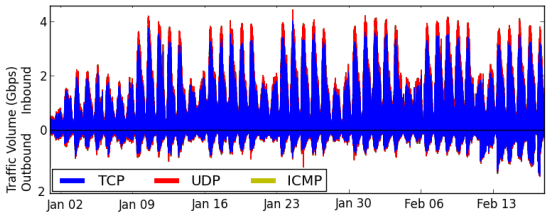


Fig. 1. Traffic profile on University of Calgary network.

This traffic was processed in real-time into connection summaries using Bro [28], which is an open-source network security monitoring tool often used for intrusion detection. The connection-level metadata was recorded for all TCP, UDP, and ICMP traffic. Our logs included over 15 billion connection summaries, resulting in a 3.5 TB dataset. Each connection summary represents communication from a sending host h_s to a receiving host h_r, where one of the hosts is on the campus network, and the other elsewhere on the Internet. (Our monitor does not see connections between hosts within our network.) Relevant fields from the connection-level data were then loaded into a Vertica[1] database, and analyzed for periodicity using SQL queries, as described in the next section. An important feature of Vertica is that it automatically exploits parallelism in query execution, without any extra effort required from the network analyst. We used the (free) community edition of Vertica on a three-node compute cluster to analyze our dataset.

IV. Periodic Traffic Model

This section discusses the modeling methodology that we used to detect periodic traffic in our empirical data. For ease

[1] https://www.vertica.com/

of exposition, we draw upon the terminology from simulation modeling to describe our approach [24]. Specifically, we start from a high-level conceptual model, which is then refined into a specification model, followed by a computational model. Finally, we discuss several extensions of the model, as well as its sensitivity and robustness.

A. Conceptual Model

Conceptually, periodic traffic is simple. It consists of repeated occurrences of an "event" at regularly-spaced time intervals. For example, the event could be the occurrence of a packet transmission or a network connection attempt.

Figure 2 shows several pedagogical examples of periodic and non-periodic traffic. These graphs are time series representations, showing the timing of connections along the horizontal axis, as well as the byte volumes sent (lower half of plot) and received (upper half) on each connection.

Figure 2(a) shows the periodic communication pattern between two NTP servers (one on campus, one elsewhere), communicating daily for 48 days. Since the NTP service is for time clock synchronization, one would expect a strongly periodic pattern in the communications, and that is indeed what is observed. This pattern is distinctly different from the non-periodic traffic illustrated in Figure 2(b). In the latter graph, two transport-level endpoints initiate connections at seemingly arbitrary points in time over a span of three hours. This pattern is non-periodic.

Despite the simplicity of the periodic traffic concept, there are many variations on its patterns. Figure 2(c) shows a pair of entities that initiate connections every half-hour for 3 hours. This traffic has periodic structure. However, Figure 2(d) shows a different pair of communicants that initiate contact only three times (rather than six) within the same 3-hour duration of observation. Whether this is deemed periodic or not depends on how you define periodicity, which is what we discuss next.

B. Specification Model

Several decisions are necessary to define period traffic precisely. These decisions involve the number and type of network events to be detected, the time duration over which detection is to be performed, the minimum and maximum periods to be detected, and the laxity with which periodicity is to be considered. Table I summarizes our definitions.

TABLE I
DEFAULT PARAMETER VALUES FOR PERIODICITY DETECTION

Item	Setting
Network event	Connection attempt
Minimum number of events	4
Minimum period	10 seconds
Maximum period	12 days
Variance threshold (laxity)	$\theta_{var} < 0.5$

In our work, we define network events at the *connection level*, rather than the packet level. For TCP, a connection attempt involves a SYN packet being transmitted from one transport-level endpoint (i.e., source IP address and port) to another (i.e., destination IP address and port). Depending on the application, connection attempts may or may not be answered by the recipient; this does not matter in our definition. Successive connection attempts may use a different port number, either at the source or the destination, but not both, so that at least one of the transport-level endpoints (i.e., address-port pair) remains the same. An analagous definition can be used for connection-less UDP-based traffic, as well as for network-level ICMP echo traffic, by treating packets between new communication endpoints as a connection initiation.

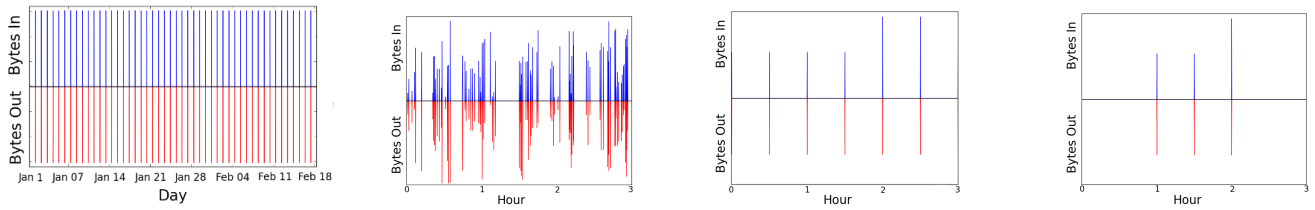
We are interested in the timing relationship between connection attempts. We require at least four connection attempts (i.e., three full periods of the periodic pattern) in order to assess the variability in the connection inter-arrival time. In our current implementation, we restrict the period to be no shorter than 10 seconds, and no longer than 12 days. However, these are easily settable configuration parameters in our model.

The final parameter in our periodic traffic model is the laxity for the periodicity detection. We use a parameter θ_{var} as a threshold for the variability permitted in the periodic pattern. Setting $\theta_{var} = 0$ is the strictest definition of periodicity, with exactly periodic behavior, while setting $\theta_{var} > 0$ allows some laxity in the timing between events. Note that θ_{var} can be defined in several ways, including variance of the connection inter-arrival times, relative variance of the connection inter-arrival times (i.e., coefficient of variation), and so on. We use absolute variance as our default, but also consider relative variance in our work [16].

C. Computational Model

To extract periodic traffic from our empirical data, we use an SQL-based implementation of the periodicity detection method proposed by Hubballi and Goyal [20]. Algorithm 1 below shows the pseudo-code for our computational model, the details of which are described in the following paragraphs.

```
Algorithm 1: Periodic Traffic Extraction                    1
                                                            2
WITH                                                        3
M AS (                                                      4
SELECT                                                      5
h_s || '/' || h_r || '/' || port || '/' || proto AS key, ts 6
FROM connection_table                                       7
),                                                          8
counts AS (                                                 9
SELECT key AS id, COUNT(key) AS n                           10
FROM M                                                      11
GROUP BY id                                                 12
),                                                          13
pruned AS (                                                 14
SELECT M.key, counts.id, M.ts AS ts                         15
FROM M, counts                                              16
WHERE M.key = counts.id                                     17
AND counts.n BETWEEN θ_min AND θ_max                        18
),                                                          19
diffs AS (                                                  20
SELECT key, ts-prev_ts AS diff                              21
FROM (                                                      22
SELECT key, ts,                                             23
LAG(ts,1) OVER (                                            24
PARTITION BY key                                            25
```



(a) Network Time Protocol (NTP) (b) Non-periodic traffic (c) Continuous periodic traffic (d) Transient periodic traffic

Fig. 2. Examples of periodic and non-periodic patterns in network traffic.

```
ORDER BY key ASC, ts ASC) AS prev_ts                    26
FROM pruned                                             27
)                                                       28
AS sub0                                                 29
GROUP BY key, ts, prev_ts                               30
ORDER BY key                                            31
),                                                      32
summary AS (                                            33
SELECT key, AVG(diff) AS Period,                        34
VAR_SAMP(diff) AS V                                     35
FROM diffs                                              36
GROUP BY key                                            37
)                                                       38
                                                        39
SELECT key, Period, V                                   40
FROM summary                                            41
WHERE V < θ_var;                                        42
```

Our analysis starts with Bro's connection-level[2] summaries. Bro defines each connection using a standard 5-tuple: source IP and port, destination IP and port, and transport protocol. Each connection summary produced by Bro includes this 5-tuple, as well as a timestamp indicating when the first packet in the connection was seen, plus other additional information, such as the number of bytes transferred.

Our analysis requires only a subset of the fields in each connection summary record. We denote each connection as $c = (ts, proto, h_s, h_r, port)$, where the port is the destination port on the "server" end of the connection summary record. That is, regardless of the server's location, we focus on successive connections issued by a host (on different ports) when contacting a particular transport-level endpoint (i.e., server/port). When referring to an attribute of a specific connection, we use dot notation. For example, $c_i.ts$ refers to the timestamp of connection c_i.

We define a *candidate connection set* to be a set S of n connections $S = \{c_1, c_2, ...c_n\}$ all with the same h_s, h_r, protocol, and port. Within each set S, the connections are timestamp-ordered (i.e., $c_i.ts \leq c_{i+1}.ts$ for all $0 < i < n$)

Periodicity detection begins by grouping all connections into distinct connection sets based on their attributes. This produces a set $M = \{S_1, S_2, ...S_m\}$ of m connection sets, where m is the number of distinct h_s/h_r/protocol/port combinations.

We then prune the connection sets, since many sets contain either too few or too many connections to manifest periodicity. To do this, we specify a minimum period (p_{min}) and maximum period (p_{max}) for periodicity detection within a duration T. From these values, we calculate $\theta_{min} = T/p_{max}$, as well as $\theta_{max} = T/p_{min}$. The cardinality n_j of each connection set $S_j \in M$ is then calculated, and any S_j with $n_j < \theta_{min}$ or $n_j > \theta_{max}$ is removed from M.

After the pruning step, for each set $S_j \in M$, we calculate the elapsed time between successive connections $c_i \in S_j$, for $0 < i < n$. We denote this as $D_j = \{d_1, d_2, ...d_{n-1}\}$ where $d_i = c_{i+1}.ts - c_i.ts$. Finally, for each set D_j of timestamp differences, we calculate the sample variance Var_j. If Var_j is below a specified threshold θ_{var}, then the connection C_j is classified as periodic. Otherwise, it is not.

An example of an SQL-like query that implements this process is presented in Algorithm 1. While there may be more computationally-efficient ways to extract the same information, we prefer this step-by-step procedural formulation for its intuitive simplicity. The modular structure also makes it easy to explore parameter sensitivity, by recording the cardinality of the sets generated at each intermediate step. Furthermore, Vertica automatically exploits the inherent parallelism in the query execution.

The WITH query in Algorithm 1 contains multiple clauses. The first clause (lines 4-7) constructs M, the set of m candidate connection sets. The second clause (lines 9-12) computes the cardinality of each connection set, used for pruning in the third clause (lines 14-18). The fourth clause (lines 20-31) calculates the time differences between successive connections within a set. The fifth clause (lines 33-37) calculates the mean and variance of the time differences. Finally, the SELECT clause at the end (lines 40-42) retrieves the sets that satisfy the variance threshold.

D. Model Sensitivity

To understand how our model parameters affect the results, we evaluated the effect of tuning the variance threshold.

Figure 3 illustrates how the variance threshold affects periodic traffic detection on one day's worth of traffic. For these sensitivity tests, we set $p_{min} = 10$ seconds and $p_{max} = 21,600$ (6 hours). This means that a candidate connection set must contain at least four events (i.e., three periods) in order to be considered for periodicity analysis.

As the variance threshold increases, the percentage of connection sets classified as periodic also increases. Figure 3 shows that the detection rate increases quickly when $\theta_{var} < 1$, and then grows slowly until $\theta_{var} = 10$. When $\theta_{var} > 10$, the

[2]A description of the format of Bro's conn log is available at https://www.bro.org/sphinx/scripts/base/protocols/conn/main.bro.html#type-Conn::Info

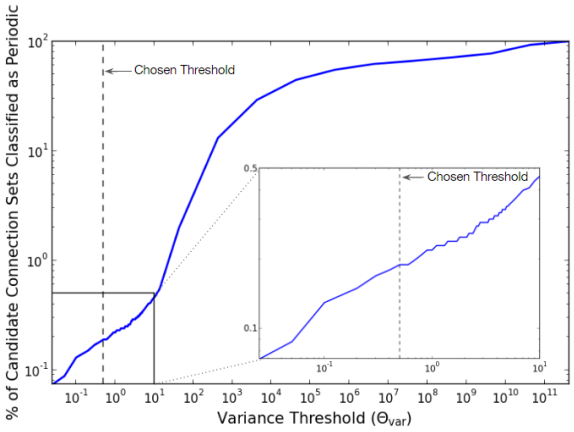


Fig. 3. Sensitivity to variance threshold.

detection rate increases rapidly before leveling off again. The rapid increase beyond $\theta_{var} = 10$ suggests that many non-periodic connection sets are being classified as periodic (i.e., false positives).

To reduce false positives in our periodic traffic detection, we used Figure 3 as a guide to select $\theta_{var} = 0.5$ as our arbitrary variance threshold. For our data analysis, we set $p_{min} = 10$ seconds and $p_{max} = 1,036,800$ (12 days), again to ensure that at least three full periods are observed. These settings are used in all remaining analyses in the paper. Further rationale for these settings, and additional experiments with other settings, are provided in [16].

E. Baseline Model Results

Table II on the next page provides a statistical summary of our empirical dataset. Logs are collected hourly, but we analyze them at different granularities: hourly, daily, and the full 7-week duration. At each granularity, we report the number of connections observed, the number of candidate connection sets (before pruning), and the number of instances of periodic traffic.

Table II shows that the periodic traffic constitutes only a tiny fraction of the aggregate network traffic. However, these periodic signals provide surprisingly valuable insights into network operations and malicious traffic activities, as demonstrated throughout the rest of the paper.

Tradeoffs exist for periodicity detection at different time granularities. Hourly logs are "cleaner", with less churn in the set of active users/devices in the network. However, it is not possible to detect long-period or long-duration periodic traffic in a short log. Furthermore, many of the same periodicities appear in successive log files of short duration, and should not be counted multiple times. We thus need to merge the results from individual logs to produce longer periodic traffic instances without duplicates.

As the log duration is increased, however, periodicities can become obfuscated, due to transient connectivity, mobility of users, diurnal effects, network outages, data collection issues, or other disruptions to the traffic patterns. Fortunately, the analysis of a long-duration log enables the identification of intermittent periodicities, which can be detected, merged, and aggregated appropriately.

The final row of Table II shows that our overall analysis more than doubles the number of periodicities detected, compared to analyzing a single composite log (first row of data in Table II). The rest of the paper focuses on these 244K instances of periodic traffic, representing 2,462,352 connection events.

V. Periodic Traffic Taxonomy

This section provides a taxonomical classification of periodic traffic, as illustrated in Figure 4.

A. Network Traffic Overview

The aggregate traffic from which we extract periodic traffic is complex. As Table II shows, periodic traffic is only a tiny component of the overall network traffic at any given timescale. Thus, identifying this traffic is challenging.

Table III provides an overview of the traffic composition on our edge network, For this summary, we provide a breakdown of the aggregate traffic at the protocol and protocol/port level. At the protocol level, Table III shows that the aggregate traffic (on a connection basis) is 73% TCP, 23% UDP, and 4% ICMP. At the port level, we classify ports as *system*, *user*, or *dynamic* ports, based on the IANA[3] definition. Table III shows that most of the aggregate traffic uses system ports, for well-known protocols such as HTTP, HTTPS, DNS, or ICMP.

TABLE III
Protocol/Port Analysis for Network Traffic.

Protocol	Port	All Traffic	Periodic Traffic
TCP	System	55%	22%
	User	17%	18%
	Dynamic	1%	5%
UDP	System	17%	3%
	User	5%	42%
	Dynamic	1%	4%
ICMP	-	4%	6%

Our goal is to understand the periodic traffic ecosystem. To this end, we examined all periodic traffic originating from or directed to our network, doing so at both the protocol and protocol/port levels.

Table III shows that the protocol breakdown for periodic traffic differs significantly from that for the aggregate traffic. UDP is the most prevalent protocol for periodic traffic (49%), followed by TCP (45%) and ICMP (6%).

We investigated why the periodic traffic differed so greatly from the aggregate traffic. We found that some hosts were involved in anomalously many periodic communications. These hosts were generally specific services, such as the Akamai CDN node on our campus network, and P2P applications, including P2P botnets. These services produced much of the periodic traffic in the user/dynamic port ranges, and primarily

[3]https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml

TABLE II
STATISTICAL SUMMARY OF EMPIRICAL DATASET AND PERIODIC TRAFFIC DETECTED.

Time Granularity	Num Logs	Total Connections			Candidate Connection Sets			Periodic Traffic Connections			Periodic Traffic Instances		
		Min	Mean	Max	Min	Mean	Max	Min	Mean	Max	Min	Mean	Max
7 Weeks	1	15.2 B			5.1 B			1,312,157			115,655		
1 Day	48	225 M	317 M	405 M	99 M	125 M	163 M	25,905	51,299	206,345	2,046	5,019	7,614
1 Hour	1,152	6 M	13.2 M	27 M	3.7 M	5.9 M	13 M	321	2,137	52,642	37	187	988
Merged	1	15.2 B			18 B			2,462,352			244,569		

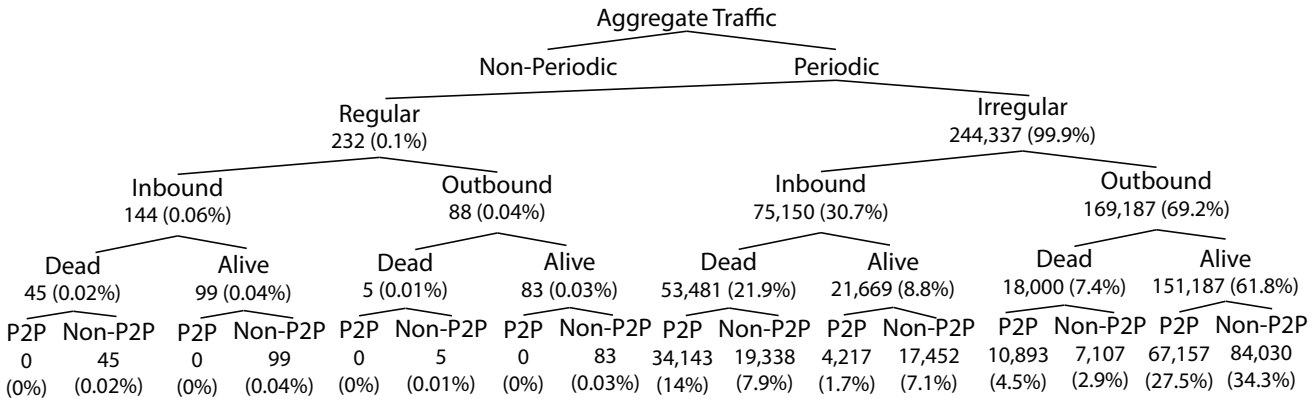


Fig. 4. Taxonomical classification of periodic traffic detected on our campus edge network.

used UDP. Their presence skews the protocol distribution for periodic traffic.

The protocol/port combinations for periodic traffic also differ substantially from those for the aggregate traffic. TCP periodic traffic was almost evenly-divided between system ports (22% of all periodic traffic) and user ports (18%), unlike the aggregate TCP traffic, which was mostly on system ports (55%). As in the aggregate traffic, TCP/80 and TCP/443 are the most popular well-known protocol/port combinations, representing 95% of all periodic traffic in the system port range. Port usage in the user range varies widely, though ports such as TCP/5223 (Apple's Push Notification Service) show lots of periodic traffic.

For UDP, periodic traffic is more prevalent in the user port range than the system port range. This differs greatly from the aggregate traffic, and suggests that relatively little of this periodic traffic is for well-known services. UDP periodic traffic is widely distributed across user/dynamic ports, but within the system port range, it is heavily concentrated on 53 (DNS), 137 (NetBIOS), and 443 (HTTPS over UDP).

Inspection of the ICMP traffic indicates that it is more prominent and varied in the periodic traffic than in the aggregate traffic. For ICMP, 'Echo Reply' and 'Host Unreachable' messages are both prominent in the periodic traffic. To a lesser extent, but still noticable, are ICMP 'Port Unreachable' messages. The volume and variety of this ICMP periodic traffic indicates a lot of scanning activities.

The main observation from this initial overview is that the composition of periodic traffic is quite different from that of the aggregate traffic. In particular, there are notable differences in the relative usage of TCP and UDP, and in the types of ports being used. As will be shown later, these differences arise primarily from the presence of P2P applications.

B. Regularity

In the simplest case, periodic behavior is persistent and continuous, occurring at regular intervals for the entire duration of the log. We define periodic traffic to be *regular* if its lifespan is within two periods of the length of the observational period. The *lifespan* is defined as the elapsed time between the first and last observed connection attempts.

Two examples of regular periodic traffic in our edge network are NTP and our Akamai CDN node. NTP creates an easily identifiable periodic pattern, as seen earlier in Figure 2(a). Our Akamai node exhibits periodic traffic using multiple protocols. For example, it uses ICMP echo queries to ping other Akamai nodes periodically to monitor Internet latencies, which are later reported to a central server.

In our 7-week log, we observed 232 instances of regular periodic traffic, with 228 of these having protocol/port combinations in the system port range. The most prominent of these was NTP, representing 112 instances of periodic traffic. Other well-known protocols were HTTP, HTTPS, DNS, NetBIOS, SSDP, and SNMP.

HTTP periodicity can be classified into two broad categories: update checks, and data posts. Update checks are probes that periodically check for updates from a server. On our edge network, we found many types of update checks (e.g., software, operating systems, anti-virus, databases, security certificates). Conversely, data posts involve periodically uploading data to a server. We observed several different examples of data posts, including backups and logging.

The remaining regular periodic traffic instances were primarily from scanning activities. In our dataset, 8 out of 11 DNS periodicities, 8 out of 17 NetBIOS instances, and all of the SSDP and SNMP periodic traffic came from Internet-scale scanning projects at Ruhr University [32].

Regular periodic traffic tends to have long periods with intuitive values, such as daily (49%) and weekly (41%) patterns. The longest period observed was 8.8 days; this was a Web proxy validating a cached object.

Contrary to the simple regularity discussed above, we found that the vast majority of periodic traffic is transient and *irregular*. This irregularity happens since computers can have transient Internet connectivity, change IP addresses, be rebooted, or be shut off overnight. In addition, some applications only generate periodic traffic part of the time.

Transient periodicities are to be expected on a modern edge network, which services a myriad of devices and purposes. The majority of the devices on our network are for personal use, and are therefore more likely to follow usage patterns that produce irregular periodic traffic. Thus, having many transient periodicities on an edge network like ours is normal.

Network middleboxes, such as DHCP and NAT, can cause irregular periodicities too. For example, a DHCP server dynamically assigns IP addresses, so that at a different point in the logs, the same laptop may have a different IP, and the previously observed IP may represent a different laptop. This DHCP churn is quite common and has several underlying causes [27]. As another example, a NAT box relays/forwards traffic from multiple other hosts on a network. This causes irregularity since the traffic from multiple hosts becomes interleaved on connection records with a common IP address.

The main insight from this regularity analysis is that the vast majority of periodic traffic is irregular. As shown in Figure 4, only 0.1% of the periodic traffic satisfied the strict definition of regularity, while 99.9% was irregular in some way. The reasons for this include user behavior, network middleboxes, and P2P applications.

C. Directionality

Although periodic traffic often involves two-way communication, it implicitly has a directional orientation, which we define relative to our edge network. Specifically, outbound periodic traffic originates from a host on our edge network, while inbound ones originate from outside our network.

Outbound periodic traffic exhibits protocol/port usage similar to the aggregate traffic in Table III. This traffic typically corresponds to well-known services, with system port usage concentrated on ports 80 and 443. TCP periodic traffic on user/dynamic ports was generated almost entirely by our Akamai node and a Sality P2P botnet that we discovered on our network due to its use of periodic communications. UDP periodic traffic on system ports was primarily NTP and Google's QUIC protocol (UDP/443). For user/dynamic ports, the UDP instances were mostly P2P, gaming, and Akamai-related traffic.

The protocol/port usage of inbound periodic traffic differs significantly from that for outbound. The few TCP periodicities on system ports were primarily for University-related services, however there was also periodic scanning for Telnet-capable hosts. TCP periodic traffic on user/dynamic ports used a wide range of ports, and were primarily related to external services interacting with hosts on our edge network. UDP periodic traffic in the system port range was primarily NTP-related. However, there was also significant DNS and NetBIOS traffic. The periodicities in the dynamic port range were mostly related to P2P traffic and our Akamai node. ICMP periodicities were more prominent in inbound traffic than outbound traffic. These instances included our Akamai node, regular echo requests, and scanning.

Figure 4 shows that regular periodic traffic was mostly inbound. These instances primarily involved services offered by hosts on our edge network. By contrast, irregular periodic traffic was mostly outbound. This traffic was generated primarily by end-user applications, in particular P2P applications. Many of the services and peers that end-user applications interact with reside outside our edge network.

The key observation here is that there are structural differences in the composition of inbound periodic traffic and outbound periodic traffic. These differences include the volume and variety of periodic traffic, as well as the protocol and port usage. The underlying reasons for the differences include the network services being offered, the applications being used, and user behavior on the network.

D. Liveness

Another attribute of periodic traffic is whether the connection attempt elicits a response or not. The terms unidirectional (one-way) or bidirectional (two-way) periodic traffic are sometimes used to describe this characteristic. However, we instead use the term *liveness* to refer to this attribute, to avoid confusion with directionality. If the originator receives zero bytes from the responder over the lifespan of the periodic traffic, we classify it as *dead*. Otherwise, we consider it *alive*. Liveness is an important feature, since Internet traffic that has no recipient is often useful for identifying odd behaviors [6].

Figure 4 shows that the majority of the periodic traffic that we observed showed liveness. Such traffic is clearly most useful when both hosts are aware that the other is receiving the probe. These instances have protocol/port usage patterns similar to the overall pattern, particularly for TCP.

Despite most periodic traffic instances being alive, we observed surprisingly many dead instances: 71,531 (29%). Many of these, particularly on the system ports for TCP and UDP, were scanning hosts for specific services, such as HTTP, HTTPS, Telnet, NetBIOS, DNS, NTP, and SNMP. Those in the user/dynamic port ranges were typically related to specific services or applications. Almost all of the TCP periodic traffic in this range was generated by service vendors attempting to communicate with hosts on our network. A very small proportion was related to P2P applications. Conversely, the dead UDP instances in this range were almost entirely related

to P2P traffic. There were also a few scanning for specific (vulnerable) services like SSDP.

The prevalence of dead (unidirectional) periodic traffic is related to churn. If a host establishes communication with a vendor/peer, and later changes IP address or sleeps, it can no longer respond. These dead instances continue for some time until the vendor/peer decides to halt communication with the inactive host.

Many ICMP instances in this category were related to host/-port scanning. A total of 6,269 dead ICMP periodicities had error return codes of either host, network, or port unreachable. The rest had return codes indicating that the ICMP request was prohibited.

As shown in Figure 4, the majority of unidirectional periodic instances are inbound. This is intuitive based on the observations above. The larger proportion of dead, inbound, irregular periodicities is also sensible considering that many are related to P2P applications and service vendors. However, there is one notable anomaly. Our Akamai node produced many unidirectional periodic instances over ports 80, 443, 11640, and 12347. In total, this single host produced 5,877 dead outbound instances of periodic traffic. This behavior likely indicates some stale configuration information.

In summary, the main observation from this analysis is that most periodic traffic (about 70%) involves bidirectional communication, while the rest (about 30%) is only unidirectional. The latter category results primarily from scanning activities, churn in P2P applications, and possibly stale network configuration information.

E. P2P Traffic

P2P applications generated lots of periodic traffic, so we used P2P as another attribute for classification. We used heuristic techniques to identify P2P applications, and packet payload captures to confirm our hypotheses [16].

Some hosts on our network were involved in a lot of periodic traffic. A few of these hosts (such as our Akamai node) were identified as specific legitimate services. However, many internal hosts were exchanging probes with a globally distributed set of hosts. Further investigation demonstrated that many of these were related to P2P applications. We were able to positively identify four P2P applications: BitTorrent, PPStream, the ZeroAccess botnet, and a strain of the Sality P2P botnet. We first noticed the latter (Sality) due to its periodic communications, presumably to share neighbor information, or other tasks related to maintaining the P2P network. We then identified that it was related to Sality by comparing its behavior to Sality's known behavior [11], and checking IP addresses in the Virus Tracker[4] database. These P2P applications were primarily on Bring Your Own Device (BYOD) subnets.

A deeper study of these P2P applications, and their periodic traffic, led to two observations. First, the identified P2P applications tend to use a specific port or limited range of ports for this traffic. Second, these P2P applications generated traffic with specific periods, or a small set of periods. Other researchers have had similar findings [1], [5], [33].

In our empirical dataset, P2P applications accounted for 48% of all periodic traffic instances detected. In most P2P applications, each peer generates a periodic probe to each other known peer. For a large P2P network, the total number of probes is high. P2P probes are often conducted using UDP on user/dynamic ports. This is why UDP is the most frequently-observed transport protocol for periodic traffic, especially on user/dynamic ports.

Figure 4 shows the prevalence of P2P periodic traffic. P2P probes were all irregular, and were prominent in both inbound and outbound traffic. In both cases, P2P traffic made up the majority of the unidirectional periodicities. This is likely due to churn in P2P networks, when peers go offline without informing the network. Even if they did inform the network, there would be a delay for that information to propagate.

For non-P2P applications, the periodic traffic had similar protocol/port usage as the aggregate traffic. The majority of the periodicities are TCP-based and concentrated in the system port range, though the user/dynamic ports are used more often than in the aggregate traffic. UDP is less prevalent in non-P2P periodic traffic, and no ICMP periodicities were P2P-related.

The key takeaway from this analysis is that approximately half of the periodic traffic on our campus edge network is from P2P applications. These applications are diverse and ubiquitous, typically using UDP for their periodic communication, and doing so using well-defined periods and a limited range of ports.

VI. Periodic Traffic Characterization

In this section, we present an in-depth characterization of the periodic traffic that we detected. We focus on structural properties, temporal characteristics, subnet-related differences, vendor-specific properties, and visual characteristics.

A. Structural Characteristics

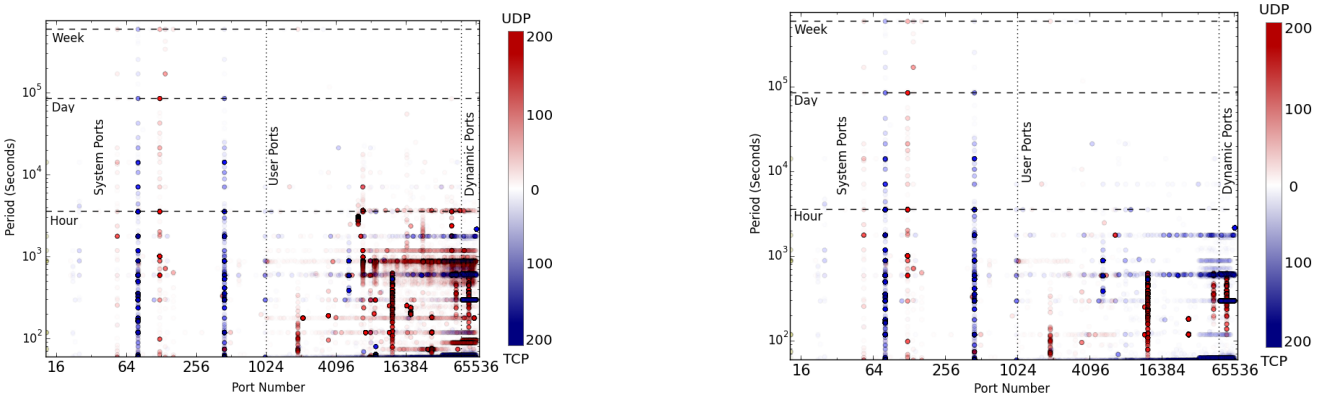
To identify clusters in the periodic traffic, we produced scatterplots of the period and port. Figure 5 presents these results, using logarithmic scales on both axes. Figure 5(a) shows the scatterplot for all periodic traffic, while Figure 5(b) shows the results for non-P2P periodic traffic. Blue denotes TCP traffic, and red denotes UDP. Individual points are drawn with low opacity, thus the more opaque an area is, the more instances of periodic traffic it represents.

Figure 5 shows three different types of visual clustering: points, horizontal bands, and vertical bands.

Points represent specific services that generate periodic traffic using a specific port. Darker points are formed when many instances share the same period and port number. For instance, the point for UDP/12350 with a period of 9 minutes is for Skype[5], while the point for TCP/5223 with a 15-minute period represents Apple's Push Notification Service.

[4]https://virustracker.net/

[5]Skype is transitioning from a classic P2P design to a centralized Microsoft Azure service [37]. Both appear in our logs. Skype can also use dynamic ports.



(a) All Periodic Traffic

(b) Non-P2P Periodic Traffic

Fig. 5. Period/port scatterplots for periodic traffic. Red is for UDP, while blue is for TCP. Darker points represent greater intensity of periodic traffic.

Horizontal bands indicate specific services that use a range of different ports. Different applications use different port ranges, so the length of horizontal bands can vary. Since the ports vary, these applications typically use ports in the user/dynamic port range. As a result, the horizontal bands tend to appear on higher-numbered ports, as seen in Figure 5.

These horizontal bands arise from specific network applications. This becomes apparent when comparing Figure 5(a) and (b). In Figure 5(a), there are multiple horizontal bands for (red) UDP periodic traffic. These horizontal bands diminish in Figure 5(b), when P2P applications are removed. Horizontal bands are not exclusive to P2P applications, though. In Figure 5(b), for example, there is a clear horizontal band across TCP ports 45,000-65,000 with a period of 9 minutes. This band represents periodic traffic generated by Google Hangouts.

Vertical bands indicate commonly used ports, but not specific applications. For example, vertical bands appear at ports 80 (HTTP), 123 (NTP), 443 (HTTPS), and 1900 (SSDP). Inspection of the periodic traffic within these bands reveals that they represent several distinct applications. For example, inspection of the HTTP logs showed that many different applications generated periodic traffic on these ports, and those with different periods were unrelated to one another.

In Figure 5(b), the three vertical bands for port numbers beyond 10,000 were generated by the Akamai node on our edge network. These instances were all directed to other Akamai nodes, and are used for internal testing and reporting.

B. Temporal Characteristics

Periodic traffic has two salient temporal attributes: period (time between successive connections) and lifespan (time between the first and last connections).

Figure 6 shows CDFs of the period and the lifespan for the periodic traffic that we observed. Figure 6(a) includes all periodic traffic, while Figure 6(b) excludes P2P traffic. The upper line (blue) is for the period. The lower line (green) is for the lifespans. Both are measured in seconds. Note that the horizontal axis is log scale, and that the vertical dashed lines denote one minute, one hour, and one day periods.

The observed periods span a wide range, with the shortest being 10 seconds (recall Table I), and the longest being 8.8 days. Despite this large range, periods are concentrated at the lower end of the distribution, with periods up to 1 minute representing 40% of all instances, and periods under one hour comprising 97%. One-minute periods were common, especially for P2P applications. The most frequent longer periods were 15, 30, 50, and 60 minutes. Periods longer than one day were quite rare ($< 0.1\%$).

The lifespans for periodic traffic also tend to be short, but have a broad range. The shortest was only 30 seconds, based on our parameter settings in Table I, and the longest was 47.9 days. About 6% have a lifespan under 1 minute, and 71% have a lifespan under one hour. Lifespans vary more widely than periods, but there is a significant peak around 10 minutes for P2P. About 30% of the lifespans exceed one hour.

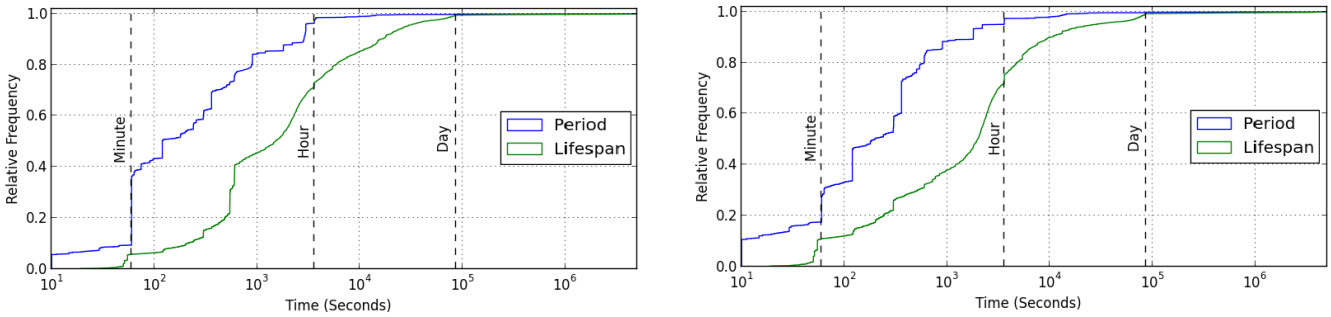
From this analysis, we conclude that "typical" periodic traffic has short periods and lifespans. Short lifespans can be due to the applications, or reflect normal end-user behavior. End-users tend to start and stop applications, relocate, or shut down their devices, particularly in a BYOD environment. This behavior contributes to the short(er) lifespans.

C. Subnet-based Analysis

Figure 7 shows a breakdown of the periodic traffic observed on several different subnets within our campus network. We selected the five busiest managed subnets from our department (based on connections), the five busiest wireless BYOD subnets, and the five busiest wired BYOD subnets for comparison. Figure 7(a) is for outbound periodic traffic, while Figure 7(b) is for inbound periodic traffic. On each graph, the subnet numbers are anonymized, but are in the same position in each graph for comparison.

Figure 7 shows several differences in the periodic traffic observed in managed and unmanaged (BYOD) subnets.

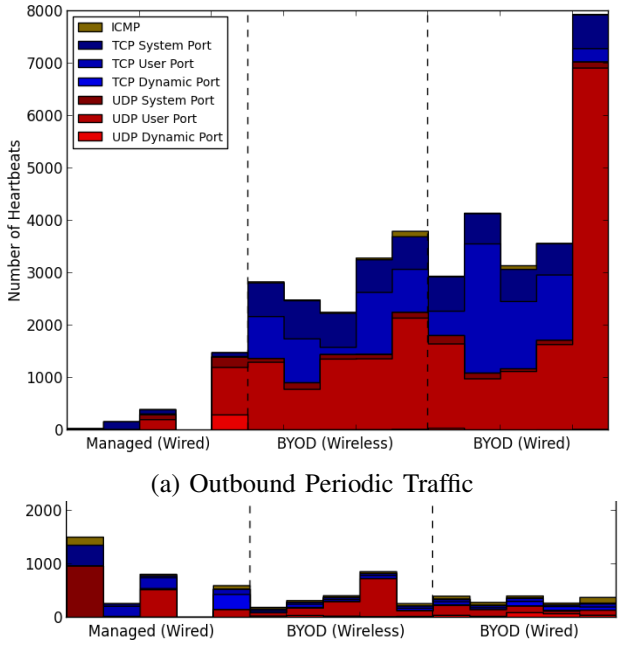
The managed portions of the network produce relativity few instances of outbound periodic traffic. Those that do exhibit periodic traffic differ depending on what the subnet is used for. Subnet 1 houses key infrastructure servers such as



(a) All Periodic Traffic

(b) Non-P2P Periodic Traffic

Fig. 6. CDFs of periods and lifespans for periodic traffic.



(a) Outbound Periodic Traffic

(b) Inbound Periodic Traffic

Fig. 7. Subnet-based analysis of periodic traffic.

DNS and Web servers, and produces only a few instances of periodic traffic, all of which are on system ports. Subnet 2 has periodic traffic directed primarily toward managed services, while subnet 3 has periodic traffic mainly related to games and end-user applications. Subnet 5 contains a NAT device that accounts for all periodic traffic on this subnet. It forwards traffic from end-user devices, and thus manifests a significant amount of periodic traffic for end-user applications, including P2P. Thus, this subnet is a hybrid of a managed and BYOD subnet.

The unmanaged subnets tend to have many more instances of outbound periodic traffic. This traffic is primarily UDP-based, however TCP also has a significant presence. The UDP instances are primarily on the user port range, reflecting P2P applications. However, there is also lots of gaming-related periodic traffic. The TCP periodic traffic in the system port range uses ports 80 and 443 almost exclusively. This periodic traffic is directed primarily to software vendors, managed (cloud) services, Web pages, CDNs, and various service providers. Those in the user port range are composed primarily of P2P traffic, however there are also lots of periodic traffic instances with Apple, Skype, and managed service providers. The few ICMP periodicities represent innocuous echo requests.

The inbound periodic traffic differs depending on the purpose of the managed portions of the network. Though subnet 1 produced the least outbound periodic traffic, Figure 7(b) shows that it receives the most inbound periodic traffic. The TCP periodic traffic is directed to Web servers and a Linux mirror site for OS updates. The UDP periodic traffic is mostly NTP-related, with a few being DNS-related. Subnet 2 receives periodic traffic from various service providers. The periodic traffic received by subnet 3 is primarily P2P traffic directed to a specific host. The inbound periodic traffic for subnet 5 is related to the NAT devices, and has a similar composition to the outgoing periodic traffic. The periodicities in ICMP traffic on managed subnets are mostly ICMP echo requests, however there are also lots of port unreachable messages.

On the unmanaged (BYOD) subnets, outbound periodic traffic dominates the inbound periodic traffic. However, the composition of each is similar, with both mostly composed of UDP traffic. The UDP periodic traffic is almost all related to P2P applications. A small portion is related to (persistent) scanning that was attempting to locate NetBIOS-capable hosts. Unlike the outbound periodic traffic, there were very few instances of TCP periodic traffic. The TCP periodic traffic was mostly generated by software vendors and managed services, although there is little usage of ports 80 and 443. Most of the TCP periodic traffic occurred on user/dynamic ports.

D. Service-Related Characteristics

Many of the periodicities observed in non-P2P traffic were related to well-known service vendors. For this analysis, we selected five representative service/software vendors for in-depth analysis. These vendors represent the four major categories observed: software vendors, service vendors, managed hosting vendors, and CDNs.

Figure 8 shows that each vendor has a distinct protocol/port usage profile. These differences reflect the diversity of services provided by each vendor.

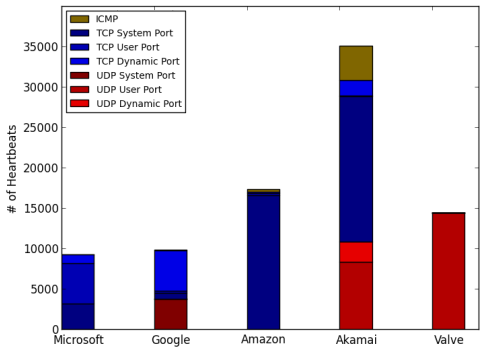


Fig. 8. Vendor-based analysis of periodic traffic.

Microsoft is a software/service vendor for which the periodic traffic uses TCP almost exclusively. Periodic traffic in the system port range used only ports 80, 443, and 993. Instances over ports 80 and 443 were related to Skype, Web application hosting, software updates, and user information collection, while those over TCP/993 were for Microsoft's Outlook e-mail service. They used periodic traffic on user/dynamic ports to communicate with hosts on our network every 60 seconds.

Google's periodic traffic relates to specific services. Instances in the system port range for both TCP and UDP were concentrated on port 443 and directed towards Google's servers. Periodic TCP traffic in the user/dynamic port range was directed towards our network, and provided Google's services (e.g., Hangouts) to hosts on our network.

Amazon uses periodic traffic for managed services. This traffic primarily used TCP ports 80 and 443. Inspection of the HTTP logs showed that these instances were not related to Amazon's hosting services themselves. Rather, the periodicities were generated by the wide variety of applications and Web pages that were being hosted on the servers. There were also some ICMP echo requests directed to Amazon's servers.

The Akamai periodic traffic had several distinct patterns. The ICMP instances were other Akamai hosts making ICMP echo requests to the Akamai node, and vice versa. Periodic traffic in the user/dynamic port ranges for both UDP and TCP were used by Akamai for internal testing and reporting. TCP periodicities in the system port range were all conducted over ports 80 and 443. Many of those over port 80 were related to the Akamai Netsession Interface[6], which periodically posts logs to Akamai servers.

Valve is a multimedia vendor that provides video games and game-related services. The periodic traffic was produced by Valve's Steam video game client. These instances occur with a period of 1.5-2 minutes on UDP/27,000-UDP/27,037. These port numbers[7] are used by Steam for game client traffic, game match-making, and in-home streaming.

[6]https://www.akamai.com/us/en/products/mediadelivery/netsession-interface-faq.jsp

[7]https://support.steampowered.com/kb_article.php?ref=8571-GLVN-8711

E. Visual Characteristics

Visualization can play a key role in the interpretation of periodic traffic. To demonstrate this, we used Gephi[8] to create four examples of node-link diagrams in Figure 9. Each node represents a host that either sends or receives periodic traffic. Hosts internal to our network are light grey, while external hosts are color-coded according to the protocol/port used by the periodic traffic. Edges are the same color as the source, thus outbound instances have grey edges, and inbound ones have colored edges.

Figure 9(a) illustrates the results for our Akamai node. This node exchanges periodic traffic with many external hosts, using TCP (blue), UDP (red), and ICMP (yellow). These instances form a dense mesh of primarily outbound edges.

P2P applications tend to form complex networks, as shown in Figure 9(b) for BitTorrent. Each internal host has its own cluster of external peers, but some nodes provide periodic traffic between clusters.

Figure 9(c) illustrates the periodic traffic of the Sality P2P botnet that we identified on our network. The internal hosts of this P2P network generate periodic traffic directed to many external hosts. Furthermore, many of these external hosts receive probes from multiple internal hosts, thus creating the complex relationships shown. These networks are quite distinctive [36], and are easily identifiable.

Figure 9(d) shows an example of the ZeroAccess botnet. In this example, the internal host is interacting with external peers that are not shared with any other internal hosts. As a result, the diagram resembles that for an internal service. However, the use of UDP on non-system ports, and the primarily inbound edges, make it visually distinct from Figure 9(a).

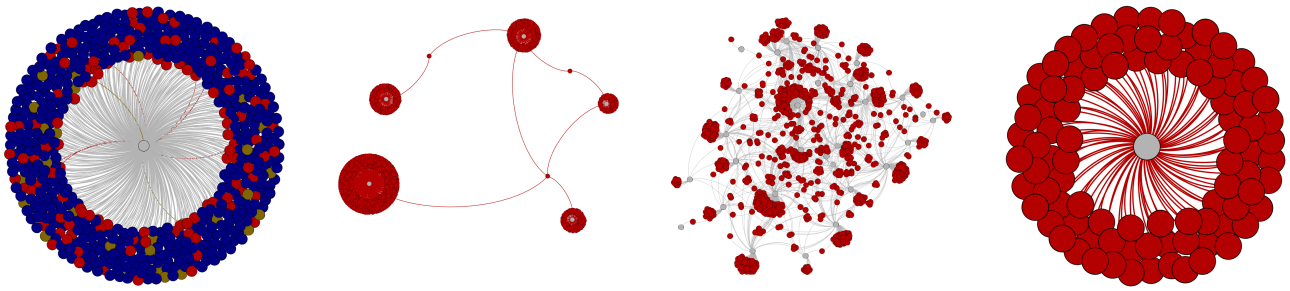
VII. Conclusions

In this paper, we provide a modeling methodology for the detection, analysis, and characterization of periodic network traffic. Our approach can be used to assess the state of well-known network services (e.g., NTP, DNS, CDN), and also detect P2P, gaming, cloud, scanning, and botnet traffic flows. Despite its simplicity, our SQL-based method is surprisingly powerful, offering deep insights into the characteristics of periodic network traffic.

The main conclusions from our work are as follows. First, periodic traffic is pervasive in modern network applications, including gaming, CDN, P2P, cloud-based services, and malicious traffic. Second, periodic traffic is diverse in its structural properties, and often very transient in its existence. Third, P2P applications account for about half of the periodic traffic detected in our empirical study. We hope that our taxonomical classification facilitates better understanding of the periodic traffic ecosystem on a modern campus edge network.

There are several high-level implications from our work. First, system administrators of managed infrastructure need to know if any of their (critical) systems are exchanging periodic traffic with unexpected places. Such communication patterns

[8]https://gephi.org/



(a) Akamai Node (b) BitTorrent (c) Sality Botnet (d) ZeroAccess Botnet

Fig. 9. Example visualizations of periodic network traffic.

may indicate compromised systems, or potential vulnerabilities. Second, security analysts need to know if there are any periodic probes from external organizations (e.g., on a security blacklist) that are scanning their network. Such traffic could indicate network reconnaissance prior to a potential attack. Finally, we need to make periodic traffic information accessible and useful for network operators. For these purposes, we regularly run our periodic traffic analysis scripts on a daily basis, and report suspicious/malicious activities to our network security team.

Future work is needed to further enhance our understanding of periodic traffic. First, the irregularities in periodic traffic warrant further investigation, since they dominate the periodic traffic ecosystem, and are not detected easily. These irregularities arise from network middleboxes (e.g., DHCP, NAT, wireless APs) and user behavior (e.g., diverse applications and devices, mobility, transient network connectivity), and are challenging to analyze. Second, better methods are needed for interpreting periodic traffic. It is often assumed that periodic behavior is inherently suspicious [7], [13], [19], or that manual review of this traffic is feasible [18], [22]. Our work indicates that interpreting periodic traffic is not simple, and better automated methods are required. Finally, obfuscation of periodic traffic is the next logical step for malware designers. We need robust detection techniques for malicious traffic once these periodic signals vanish.

Acknowledgements

Financial support for this work was provided by Canada's Natural Sciences and Engineering Research Council (NSERC). The authors are grateful to University of Calgary Information Technologies (UCIT) for facilitating the collection of our network traffic measurement data, and for permission to publish this work. The authors thank the IEEE MASCOTS 2018 reviewers, who provided constructive feedback and insightful comments on an earlier version of this paper.

References

[1] D. Andriesse, C. Rossow, and H. Bos, "Reliable Recon in Adversarial Peer-to-Peer Botnets", *Proceedings of ACM Internet Measurement Conference*, pp. 129–140, Tokyo, Japan, October 2015.

[2] B. AsSadhan and J. Moura, "An Efficient Method to Detect Periodic Behavior in Botnet Traffic by Analyzing Control Plane Traffic", *Journal of Advanced Research*, Vol. 5, No. 4, pp. 435–448, July 2014.

[3] R. Barbosa, R. Sadre, and A. Pras, "Towards Periodicity Based Anomaly Detection in SCADA Networks", *Proceedings of IEEE Conference on Emerging Technologies and Factory Automation*, pp. 1–4, Krakow, Poland, September 2012.

[4] G. Bartlett, J. Heidemann, and C. Papadopoulos, "Using Low-rate Flow Periodicities for Anomaly Detection: Extended" Technical Report, USC/Information Sciences Institute, 2009.

[5] G. Bartlett, J. Heidemann, and C. Papadopoulos, "Low-rate, Flow-level Periodicity Detection", *Proceedings of the 14th IEEE Global Internet Symposium*, pp. 804–809, Shanghai, China, April 2011.

[6] K. Benson, A. Dainotti, A. Snoeren, and M. Kallitsis, "Leveraging Internet Background Radiation for Opportunistic Network Analysis", *Proceedings of ACM Internet Measurement Conference*, pp. 423-436, Tokyo, Japan, October 2015.

[7] L. Bilge, D. Balzarotti, W. Robertson, E. Kirda, and C. Kruegel, "Disclosure: Detecting Botnet Command and Control Servers through Large-scale Netflow Analysis", *Proceedings of the 28th Annual Computer Security Applications Conference*, pp. 129–138, Orlando, FL, December 2012.

[8] V. Bolotin, J. Coombs-Reyes, D. Heyman, Y. Levy, and D. Liu, "IP Traffic Characterization for Planning and Control", *Proceedings of International Teletraffic Congress*, Vol. 16, pp. 425–436, Edinburgh, Scotland, June 1999.

[9] L. Chen, S. Hsiao, M. Chen, and W. Liao, "Slow-paced Persistent Network Attacks Analysis and Detection using Spectrum Analysis", *IEEE Systems Journal*, Vol. 10, No. 4, pp. 1326–1337, December 2016.

[10] M. Eslahi, M. Rohmad, H. Nilsaz, M. Naseri, N. Tahir, and H. Hashim, "Periodicity Classification of HTTP Traffic to Detect HTTP Botnets", *IEEE Symposium on Computer Applications and Industrial Electronics*, pp. 119–123, Langkawi, Malaysia, April 2015.

[11] N. Falliere, "Sality: Story of a Peer-to-Peer Viral Network", *Technical Report*, Symantec, July 2011.

[12] W. Feng, F. Chang, W. Feng, and J. Walpole, "A Traffic Characterization of Popular On-line Games", *IEEE/ACM Transactions on Networking*, Vol. 13, No. 3, pp. 488–500, June 2005.

[13] S. Garcia, "Modelling the Network Behavior of Malware to Block Malicious Patterns: The Stratosphere Project: A Behavioral IPS", *Virus Bulletin*, pp. 1–8, Prague, Czech Republic, October 2015.

[14] J. Gardiner, M. Cova, and S. Nagaraja, "Command and Control: Understanding, Denying, and Detecting – A Review of Malware C2 Techniques, Detection and Defences", *arXiv preprint arXiv:1408.1136*, 2014.

[15] G. Gu, J. Zhang, and W. Lee, "Botsniffer: Detecting Botnet Command and Control Channels in Network Traffic", *Proceedings of Network and Distributed System Security Symposium*, pp. 1–18, San Diego, CA, February 2008.

[16] M. Haffey, *Characterization of Periodic Network Traffic*, MSc Thesis, Department of Computer Science, University of Calgary, August 2017.

[17] X. He, C. Papadopoulos, J. Heidemann, U. Mitra, and U. Riaz, "Remote Detection of Bottleneck Links using Spectral and Statistical Methods" *Computer Networks*, Vol. 53, No. 3, pp. 279–298, February 2009.

[18] N. Heard, P. Delanchy, and D. Lawson, "Filtering Automated Polling Traffic in Computer Network Flow Data", *Proceedings of IEEE Joint Intelligence and Security Informatics Conference*, pp. 268-271, Hague, Netherlands, September 2014.

[19] X. Hu, J. Jang, M. Stoecklin, T. Wang, D. Schales, D. Kirat, and J. Rao, "Baywatch: Robust Beaconing Detection to Identify Infected Hosts in Large-scale Enterprise Networks", *IEEE/IFIP International Conference on Dependable Systems and Networks*, pp. 479–490, June 2016.

[20] N. Hubballi and D. Goyal, "Flowsummary: Summarizing Network Flows for Communication Periodicity Detection", *International Conference on Pattern Recognition and Machine Intelligence*, pp. 695–700, Kolkata, India, December 2013.

[21] P. Huijse, P. Estevez, P. Protopapas, P. Zegers, and J. Principe, "An Information Theoretic Algorithm for Finding Periodicities in Stellar Light Curves", *IEEE Transactions on Signal Processing*, Vol. 60, No. 10, pp. 5135–5145, October 2012.

[22] N. Huynh, W. Ng, A. Ulmer, and J. Kohlhammer, "Uncovering Periodic Network Signals of Cyber Attacks", *IEEE Symposium on Visualization for Cyber Security*, pp. 1–8, Baltimore, MD, USA, October 2016.

[23] A. Lakhina, K. Papagiannaki, M. Crovella, C. Diot, E. Kolaczyk, and N. Taft, "Structural Analysis of Network Traffic Flows", *Proceedings of ACM SIGMETRICS*, pp. 61–72, New York, NY, June 2004.

[24] L. Leemis and S. Park, *Discrete-Event Simulation: A First Course*, Pearson Prentice Hall, 2006.

[25] Z. Li, B. Ding, J. Han, R. Kays, and P. Nye, "Mining Periodic Behaviors for Moving Objects", *Proceedings of ACM International Conference on Knowledge Discovery and Data Mining*, pp. 1099–1108, Washington, DC, July 2010.

[26] N. Nikaein, M. Laner, K. Zhou, P. Svoboda, D. Drajic, M. Popovic, and S. Krco, "Simple Traffic Modeling Framework for Machine Type Communication", *International Symposium on Wireless Communication Systems*, pp. 783–787, Ilmenau, Germany, October 2013.

[27] R. Padmanabhan, A. Dhamdhere, E. Aben, and N. Spring, "Reasons Dynamic Addresses Change", *Proceedings of ACM Internet Measurement Conference*, pp. 183–198, Santa Monica, CA, USA, November 2016.

[28] V. Paxson, "Bro: A System for Detecting Network Intruders in Real-time", *Computer Networks*, Vol. 31, No. 23, pp. 2435–2463, December 1999.

[29] Y. Qiao, Y. Yang, J. He, B. Liu, and Y. Zeng, "Detecting Parasite P2P Botnet in Emule-like Networks through Quasi-periodicity Recognition", *International Conference on Information Security and Cryptology*, pp. 127–139, Seoul, Korea, December 2011.

[30] Y. Qiao, Y. Yang, J. He, C. Tang, and Y. Zeng, "Detecting P2P Bots by Mining the Regional Periodicity", *Journal of Zhejiang University SCIENCE C*, Vol. 14, No. 9, pp. 682–700, September 2013.

[31] A. Rao, A. Legout, Y. Lim, D. Towsley, D. Barakat, and W. Dabbous, "Network Characteristics of Video Streaming Traffic", *Proceedings of the 7th ACM International Conference on Emerging Networking Experiments and Technologies*, Article 25, Tokyo, Japan, December 2011.

[32] C. Rossow, "Amplification Hell: Revisiting Network Protocols for DDoS Abuse", *Proceedings of Network and Distributed System Security Symposium*, pp. 1–15, San Diego, CA, USA, February 2014.

[33] C. Rossow, D. Andriesse, T. Werner, B. Stone-Gross, D. Plohmann, C. Dietrich, and H. Bos, "SoK: P2PWNED – Modeling and Evaluating the Resilience of Peer-to-Peer Botnets", *IEEE Symposium on Security and Privacy*, pp. 97–111, San Francisco, CA, USA, May 2013.

[34] B. Seaton, "The Detection of Periodicity in Irregular Data", *Journal of Theoretical Biology*, Vol. 63, No. 2, pp. 311–324, December 1976.

[35] J. van Splunder, "Periodicity Detection in Network Traffic", Technical Report, Mathematisch Instituut Universiteit Leiden, 2015.

[36] P. Wang, S. Sparks, and C. Zou, "An Advanced Hybrid Peer-to-Peer Botnet', *IEEE Transactions on Dependable and Secure Computing*, Vol. 7, No. 2, pp. 113–127, April-June 2010.

[37] Wikipedia, "Skype", https://en.wikipedia.org/wiki/Skype