Time Signature Based Matching for Data Fusion and Coordination Detection in Cyber Relevant Logs

Lauren Deason, PUNCH Cyber*

October 25, 2017

The ability to detect automated behavior within cyber log data is a useful tool for the network defender, as malicious activity executed by scripts or bots is likely to leave behind identifiable traces in logs. This paper presents a methodology for detecting certain types of automated activity within cyber logs based on matching observed temporal patterns. This methodology is scalable, overcoming the infeasibility of brute force methods to identify groups of nearest neighbors in large datasets by implementing a locality sensitive hashing algorithm. This coordination detection methodology applied to cyber log data can be used to develop features for input into further analysis such as anomaly detection to flag potentially malicious activity or unsupervised clustering to characterize classes of automated behavior. Alternatively, the methodology could be used as a means to fuse together disparate data sources by generating a 'temporal signature' key and allowing for fuzzy matching on this key. Examples of each type of application are presented using a dataset of billions of records of netflow data.

1 Overview

This paper presents a general methodology for identifying similar temporal patterns across large sets of discrete time series data at scale, with examples of results for the specific application to cyber log data analysis. Part I describes the problem and the analytic methodology implemented to solve it, with specific pre-processing details for the application to cyber log data. Part II then presents specific examples of results obtained from applying this methodology to netflow data.

Part I

Problem Statement and Methodology

The general problem addressed in this paper is how to identify similar temporal patterns across entities within large sets of time series data. This is a particular instance of the 'nearest neighbor' problem, well known to be intractable by brute force methods at scale due to the n^2 size of the set of pairwise computations needed.

*Contact email: lauren@punchcyber.com. This research was developed with funding from the Defense Advanced Research Projects Agency (DARPA). The views, opinions and/or findings expressed are those of the author and should not be interpreted as representing the official views or policies of the Department of Defense or the U.S. Government. I would like to thank John Lankau, Kyle Smith, and Mike Geide of PUNCH Cyber for their input as Cyber Subject Matter Experts in the examples presented in this paper.

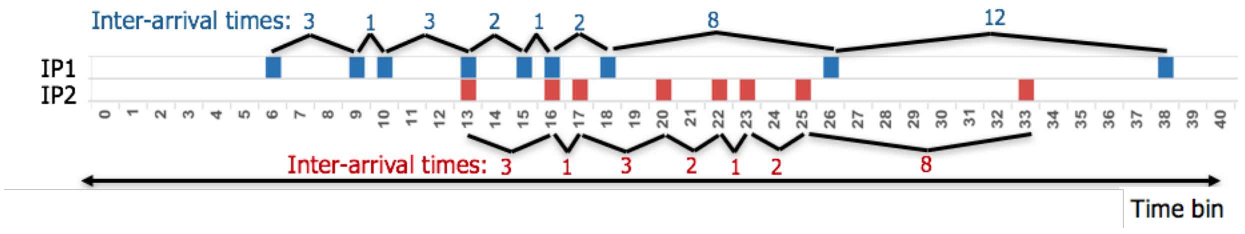
Fortunately, there are approximate techniques that allow us to find the desired set of nearest neighbors with high probability, such as Locality Sensitive Hashing (LSH), the technique employed here. In this paper, I define appropriate temporal feature spaces and metrics relevant for identifying temporal patterns of interest in discrete time series data and present an algorithm using locality sensitive hashing implemented via Apache Spark that is able to identify similar temporal behavior from across millions to billions of time series. The methodology presented here is very general and could be applied to a wide variety of discrete time series pattern matching problems. The focus here is an application to cyber relevant log data aiming to achieve two primary goals: 1) Identify automated and coordinated activity across entities that may be indicative of malicious activity; and 2) Provide a means to merge together disparate time series datasets, even when no common join key is present.

2 Temporal Feature Spaces and Metrics

2.1 Time Signature

I construct a time signature for each entity which is the set of all time bins in which this entity has activity (this can be extended, as shown later, to a time series where the magnitude of activity at a given time is also tracked). Figure 1 depicts an example where each entity is an IP that exhibits connection activity in

Figure 1: Time Signature Examples of Network Activity by IP



netflow logs at distinct points in time. Formally, for entity i, with records r at times t, we construct the time signature for i as $TS_i = \{t | \exists r_{i,t}\}$ so that in this example, we have that $TS_{IP1} = \{6, 9, 10, 13, 15, 16\}$ and $TS_{IP2} = \{13, 16, 17, 20, 22\}$.[1] In order to define a metric on the space of time signatures, it is sensible to assign a distance of zero to entities that have identical time signatures, however, there may also be cases where we are interested to know that two entities are nearly similar (in the case of the IP example here, this could be due to two sensors tracking the same traffic, but for whatever reason a small percentage of observations are not observed by one of the sensors. To obtain a metric that allows us to compare time signatures in this way, we use one minus the Jaccard similarity of the sets of times in two time signatures to represent the 'time signature distance' between two entities: $D_{TS}(i,j) = 1 - \frac{|TS_i \cap TS_j}{|TS_i \cup TS_j|}$. Thus, the distance between any two time series ranges between zero and one, and for the example in Figure 1 we have that $D_{TS}(IP1, IP2) = 1 - \frac{2}{15} = 0.867$.

2.2 Interarrival Shingles

In addition to flagging activity that exhibits similar temporal activity based on a large overlap in the set of timebins with activity, we also wish to identify similar temporal behavior across entities which may be

[1]Note that this can be thought of as a sparse vector representation of the time series for entity i in dataset d, and in fact this is the data structure in which each time series is stored within an RDD in the Spark implementation.

offset by some lag. In the case of looking for automated activity associated with malicious behavior, this would allow us to find temporal signatures of malware that are identical modulo some time offset due to malware scripts initiated by a user action, such as clicking a link or opening a file, that could cause the start times to differ by arbitrary lags across machines. In the case of fusing together disparate data sources using temporal patterns, allowing for arbitrary lags can compensate for time zone differences or sensors that are not synchronized. The example in Figure 1 shows a case where although the time signatures of the two entities (in this case, the two IPs) are not close, there does appear to be coordination across the two evident by the fact that the time signatures would line up (nearly) perfectly if IP2's timeline were to be shifted 7 units to the left. One way to see this is to compute the series of interarrival times for each entity, which, as shown in the figure are 3,1,3,2,1,2,8,12 for IP1 and 3,1,3,2,1,2,8 for IP2. Because the order in which interarrival times appear in a series matters, and because I wish to again allow for imperfect or 'fuzzy' matching of interarrival series, I define a metric not over pairs of full interarrival time series, but over pairs of sets of 'shingles' of interarrival times. An n-shingle is defined to be an n-tuple subsequence of interarrival times. Thus, in the example above, the set of 3-shingles of interarrival times for IP1 would be $IA^3_{IP1} = (3,1,3),(1,3,2),(3,2,1),(2,1,2),(1,2,8),(2,8,12)$ and that for IP2 would be $IA^3_{IP2} = (3,1,3),(1,3,2),(3,2,1),(2,1,2),(1,2,8)$. I then define the 'n-shingle interarrival distance' between entities i and j as $D^n_{IA}(i,j) = 1 - \frac{|IA^n_i \cap IA^n_j|}{|IA^n_i \cup IA^n_j|}$. As in the case of time signature distance, this will range between zero and 1 and a small time signature distance will imply a small interarrival distance, however, the converse is not true, as there may be time series with no time bins in common that exhibit identical temporal patterns offset by some lag. The size of the shingle chosen will affect the sensitivity of the distance metric to deviations from identical interarrival time sequences and can be chosen based on the desired 'fuzziness' of matching time series; in general a large value of n will result in a very strict matching criteria where one deviation in an interarrival time could cause two entities to be at the maximum interarrival distance, while at the other extreme, a value of n=1 would allow entities with very different temporal patterns to match.

3 Data Pre-processing for Cyber Relevant Log Analysis

3.1 Time Discretization

In order to define feature spaces and distances on discrete time series data, we must of course first have time series data that has been discretized. In the case of cyber logs, time series are already presented as discrete, with timestamp precision that may go down the nanosecond level, however, there are several considerations in deciding the level at which to discretize time for the purpose of time pattern matching studied here. Network data logged by sensors is often, if not always, subject to jitter, that is, an event may not be logged at the exact time it occurs but rather be subject to some lag. For a maximum such lag of δ seconds, events which occur at a distance of P seconds from each other, could then be observed in log data with interarrival times ranging between $P - 2\delta$ to $P + 2\delta$ (with an expected interarrival time of P, if the observed lag is stochastic with independent and stationary distribution). The ability of our analytic to detect coordinated behavior in the presence of jitter will depend on the the time aggregation level chosen. Aggregating at a very fine level may cause us to miss coordinated activity that fails to exhibit matching time patterns due to jitter in logging times (false negatives), however aggregating at too high a level may cause us to flag as coordinated activity that is in reality unrelated (false positives). The time aggregation level chosen will be influenced by risk preferences for false negatives/positives and prior knowledge about jitter expected by a given sensor.

3.2 Entity Specification

For tabular data with a timestamp field, in order to extract a set of time series for multiple entities, one must first specify the entity definition (that is, a group-by key). In the case of the application to to cyber data, and netflow logs in particular presented below, the entity of interest may be some combination of IPs, ports, and protocols, as in the examples below, where data is grouped by the four-tuple source IP, destination IP, protocol, and destination port in order to define a discrete time series of binary indicators indicating connections observed in netflow logs.

3.3 Windowing

For a given set of time series data, we may wish to identify entities with similar temporal patterns not only over the entire time period but also those which display very similar patterns during only smaller windows of time. In cyber log applications such as those presented in Part II, this may be due to a desire to flag specific time signatures associated with malicious activity that occur only sporadically and interspersed with other, legitimate activity. Another reason to allow for matching temporal behavior over shorter windows of time is to permit entity definitions which may be dynamic over longer periods of time such as dynamically assigned IP address, for example, which are commonly implemented via DHCP on networks. For both reasons, in the applications below, data is pre-processed to define one time series for each entity-day, so that entities with changing IP addresses or temporarily coordinated behavior will not go undetected due to failure of the time patterns to match over long periods.

4 Identifying Temporally Similar Entity Groups

Given the feature spaces and associated metrics described in section 2, the problem of finding the set of all neighbors within δ of a given entity is trivially accomplished by computing the distance between the entity of interest and all others in the dataset. When there is no one entity known to be of interest, but rather the goal is to identify instances of any pairs or groups of entities that lie within a distance δ of each other, the problem becomes impractically large to compute by brute force pairwise distance computations as the number of these computations scales with the square of the number of entities. In the case of the application considered here, cyber log data, where we consider time series for each four tuple entity defined by Source IP, Destination IP, Protocol, Destination Port, the number of entities is in the millions, making direct computation infeasible. This is a specific instance of a more general class of 'nearest neighbor' problems, known to suffer from scale difficulties. Luckily, there are approximate methods available to solve this class of problems, including the method employed here, Locality Sensitive Hashing. This algorithm avoids the need to compute all n^2 pairwise distances for a set of n entities by first applying a cleverly designed hash function to each entity such that entities near each other in the metric space of interest are likely to hash into the same bucket while entities which are far from each other are unlikely to do so. The number of pairwise comparisons that needs to be made to find the set of nearest neighbors is then bounded above by $n * k^2$ where k is the maximum number of entities hashing into any one bucket, and can be tuned via parameter choice inputs to the algorithm. In the Jaccard distance, a minhash is used, taking advantage of the fact that the probability of two sets receiving the same minhash value after a random permutation of element ordering is equal to their Jaccard similarity. For a detailed description of the algorithm, the reader is referred to Chapter 3 of [3] which provides an excellent introduction to the concept of locality sensitive hashing and its applications to big data analysis.

One unique aspect of the application of this methodology to matching of time series patterns for the purpose of identifying actors that appear to be acting in a coordinated manner is the problem of matches that occur by random chance rather than because of any actual relationship between actors. As a concrete example, suppose a time period composed of 1024 discrete time bins is observed for a set of millions of entities, where 1000 of these entities exhibit activity in exactly two discrete time bins, and these time bins are chosen by each entity at random. In this case, the probability of an exact match occurring between these length-2 time signatures for two or more entities is approximately 23%. This is a direct generalization of the well known birthday problem describing the probability of two or more people in a room sharing a birthday where the number of 'days' in the 'year' is the number of ways to choose k distinct time bins from a set of d possible time bins and the number of people in the room is the number of entities that have activity in exactly k time bins. Using Stirling's approximation formula to compute the binomial coefficient $\binom{d}{k}$ for $d \gg k$, I define a probability threshold below which I consider temporal matches to be possible random chance collisions, and therefore not of interest. This has the effect of filtering out entities with very short time signatures, as matches for these cannot confidently be attributed to collusion/coordination rather than chance collisions.[2]

Part II

Applications and Examples from Network Log Analysis

In this section I present some illustrative results and examples from this algorithm applied to Silk data grouped by the 4-tuple of Source IP, Destination IP, Protocol, and Destination Port.[3] Coordinated activity across such 4-tuples can identify various types of scanning behavior as well as other types of unexpected coordination across IPs. The dataset used in the examples below consists of 1.4 billion Silk records spanning over 650,000 source IPs, 6.6 million destination IPs, and covers a time period of 150 days. Because I have only Silk records, and no other log types with which to enrich the netflow data or provide further context, the examples presented here show the types of automated and coordinated activity that can be detected using only netflow data, and the types of post-processing that may be useful to distinguish between expected or potentially malicious automated activity on a given network. Each example summarizes the observed activity and provides an educated guess based on Cyber Subject Matter Expert (SME) input as to the underlying activity on the network; however a definitive determination as to the nature of the activity in these examples is generally not possible without further context. My goal is to present the set of temporal features I am able to produce at scale for such data, with a discussion of how one might take the next step

[2]In fact, the chance of random chance collisions of time signatures is higher than assumed by this set-up in which I have assumed entities to choose discrete time bins at random. In reality, some patterns of time activity (such as persistent activity over the course of several consecutive time bins) are more likely than others and so I may still obtain some accidental matches using this criteria.

[3]I have chosen not to include source port in the definition of entity as this is typically not as informative as the other four fields in characterizing a connection. It could, however, easily be added in and would not change the methodology.

of performing anomaly detection on these features, with the final goal of sending unexpected automated or coordinated activity detected on a network up to human Cyber analysts for triage.

5 Temporal Group Features

In order to characterize the different types of coordinated activity we see in netflow, I define a set of additional features on time series groups, where each group is the set of neighbors within a distance of 0.1 of each other under the Jaccard similarity of time series metric. These feature definitions are driven by domain knowledge

Table 1: Summary of Temporal Group Features Explored in Examples

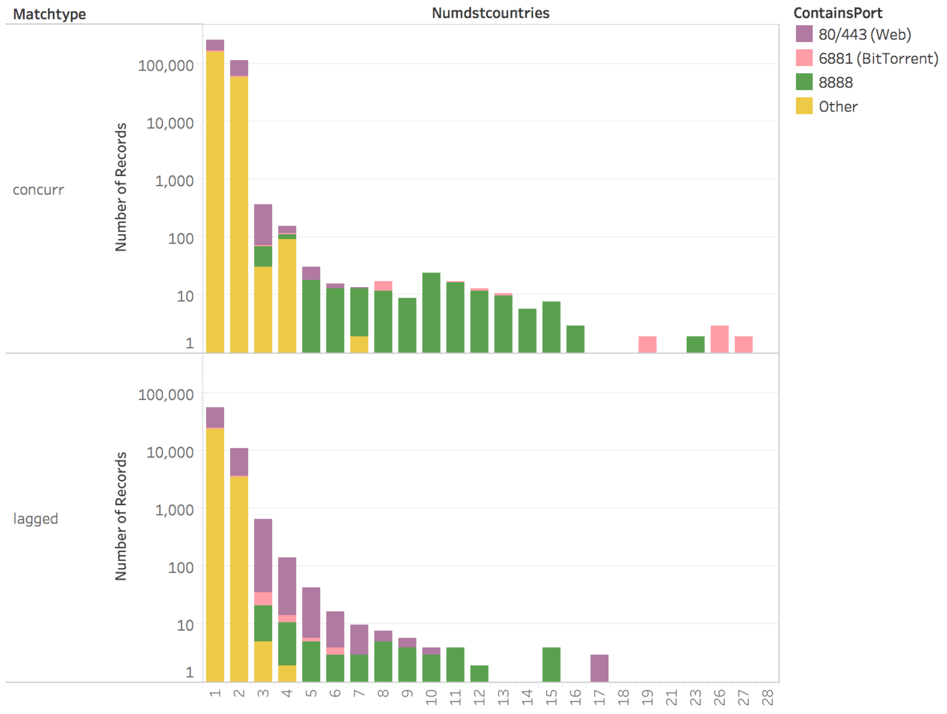
Category	**Feature**	**Definition**
Communication Type	Share Int2Int	Share of entities in a given time series group that are categorized by Silk as having Source IP internal to network and Destination IP internal to network
	Share In	Share of entities in a given time series group that are categorized by Silk as having Source IP external to network and Destination IP internal to network, not on port 80 or 443
	Share Inweb	Share of entities in a given time series group that are categorized by Silk as having Source IP external to network and Destination IP internal to network, on port 80 or 443
	Share Out	Share of entities in a given time series group that are categorized by Silk as having Source IP internal to network and Destination IP external to network, not on port 80 or 443
	Share Outweb	Share of entities in a given time series group that are categorized by Silk as having Source IP internal to network and Destination IP external to network, on port 80 or 443
	Share Ext2ext	Share of entities in a given time series group that are categorized by Silk as having Source IP external to network and Destination IP external to network
Coordination Type	Share Targetted Port Scanning	Share of entities in a given group for which there is at least one other entity within the group that shares the same source IP and destination port but different destination IP
	Share Horizontal Port Scanning	Share of entities in a given group for which there is at least one other entity within the group that shares the same source and destination IP but different destination ports
	Share Coordinated Source Activity	Share of entities in a given group for which there is at least one other entity within the group that shares the same source IP
	Share Coordinated Destination Activity	Share of entities in a given group for which there is at least one other entity within the group that shares the same destination IP
Time Series Length	Timedelt	Length of time (in seconds) between the first and last observed activity for the average entity in the given time series group
	NumBins	Number of discrete time bins in which the average entity in the given group has activity
Temporal Regularity	Interarrival Shingle Entropy	$\sum_{s \in IA_i^n} p \ln p$ where p is the empirical probability of randomly selecting shingle s from the full set of interarrival shingles. Thus, time series that exhibit a high degree of regularity (such as periodic behavior) will have low entropy.
Other	NumDstCountries	Number of distinct countries included in the geotagged set of destination IPs for entities within the given group

and exploration of similar temporal patterns found by the above algorithm when applied to 4-tuple entity (Source IP, Destination IP, Protocol, and Destination Port) time series netflow data. One common type of activity that tends to be flagged for similar temporal behavior by this analytic is targeted port scanning, that is, a source IP that connects to multiple destination IPs on the same destination port in a coordinated fashion. Thus, grouping within each time series group by Source IP and destination Port and then counting how many of these Source IP-Dest Port pairs correspond to more than one distinct Destination IP in that group, we can characterize the share of entities in a given group that exhibit similar temporal behavior that could be thought of as ’targeted port scanning’. Similarly, we can group a set of coordinated entities to identify IPs engaged in horizontal port scanning, that is, a source IP that connects to multiple destination ports on the same destination IP in a coordinated fashion. For entities composed of 4 fields, there are obviously 4! distinct such ’types’ of coordination that could be identified, with varying levels of interpretability and cyber relevance. In the examples presented here, I include statistics on the share of entities within a group that exhibit either type of ’port scanning’ as well as the share that exhibit a more general ’coordinated source IP’ or ’coordinated destination IP’ activity. Clearly, there is overlap in these groups, and there may be other cyber relevant classifications that I do not include here. In addition to these characterizations of each temporal group by ’coordination type’, I compute other aggregate statistics for each group based on features of the individual entities and time series that are included in each, such as ’Communication Type’ (whether the logged activity represents internal to external traffic, internal to internal, etc.), ’Time Series Length’, ’Interarrival Shingle Entropy’, and measures of the number of companies or countries included in the geolocations of associated IPs. These features are summarized in Table 1.

5.1 Anomalies with Respect to Number of Destination Countries

Figure 2 shows a histogram of temporal groups (for both concurrent and time lagged matches) broken out by the number of countries spanned by the set of destination IPs contained in each temporal group. As

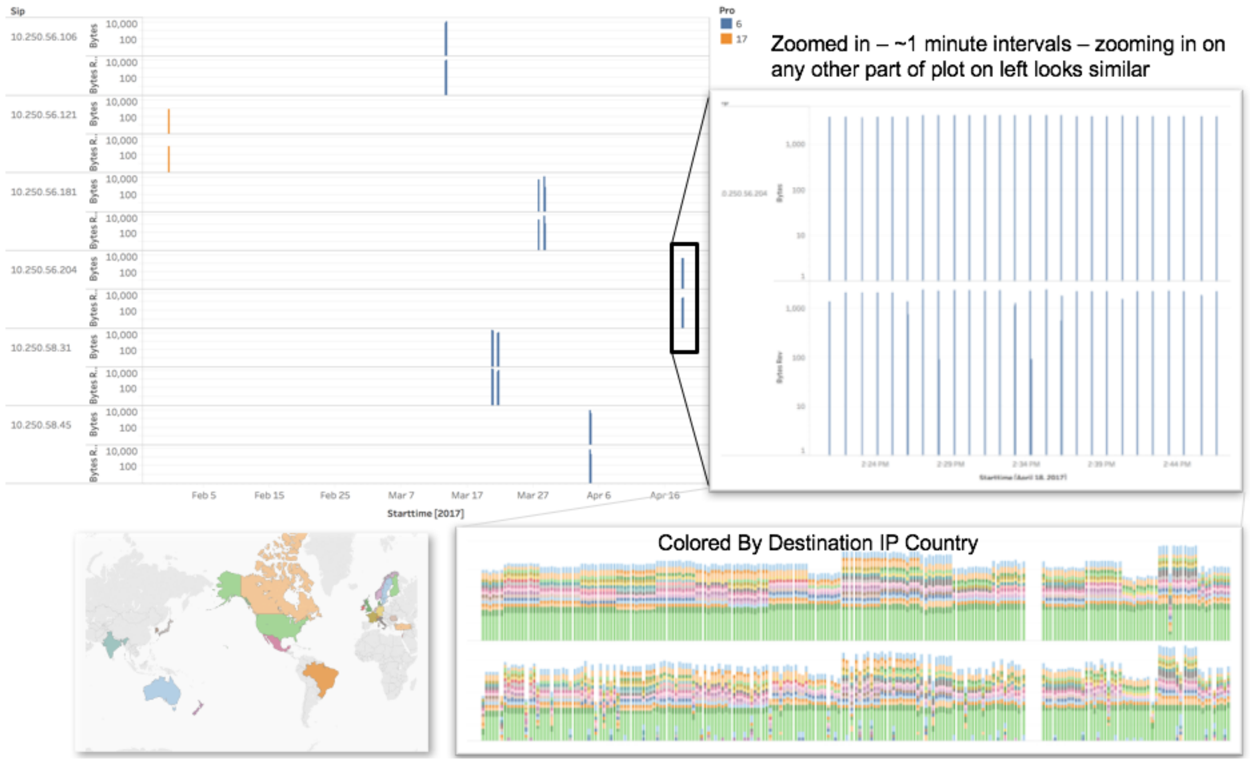
Figure 2: Histogram of Temporal Groups by Number of Destination Countries



one might expect, the majority of coordinated netflow activity for 4-tuples of source IP, dest IP, protocol, and port occurs across sets of 4-tuples where the IPs involved in the coordinated activity are located in the same country (as identified by MaxMind’s geotagging of IPs). Figure 2 also depicts the breakdown by temporal groups that contain any entity with a destination port in either 80/443 (web traffic), 6881

(BitTorrent), or 8888. There is a clear correlation between the number of destination countries spanned by a temporal group and the indicator for whether each of these destination ports is present within the group. Groups containing the largest number of destination countries (up to 28) tend to include traffic over port 6881, which is commonly used by BitTorrent, and in this case appears to be a logical guess as to the underlying activity as the internal IP connects to dozens of hosts around the globe to transfer files over the P2P network. The next most common destination port observed for groups containing destination IPs across multiple countries is 8888. Pulling down the raw data associated with this traffic (see Figure 3), we see that there are six internal IPs with similar behavior: at various points in time, they make periodic (roughly 1 minute period) connections over port 8888 to a set of 1407 external IPs spanning 22 companies and 26 countries. Each burst of such periodic activity lasts roughly 3 hours and connections are made to

Figure 3: Periodic, Coordinated Connections to Multiple Destinations on Port 8888

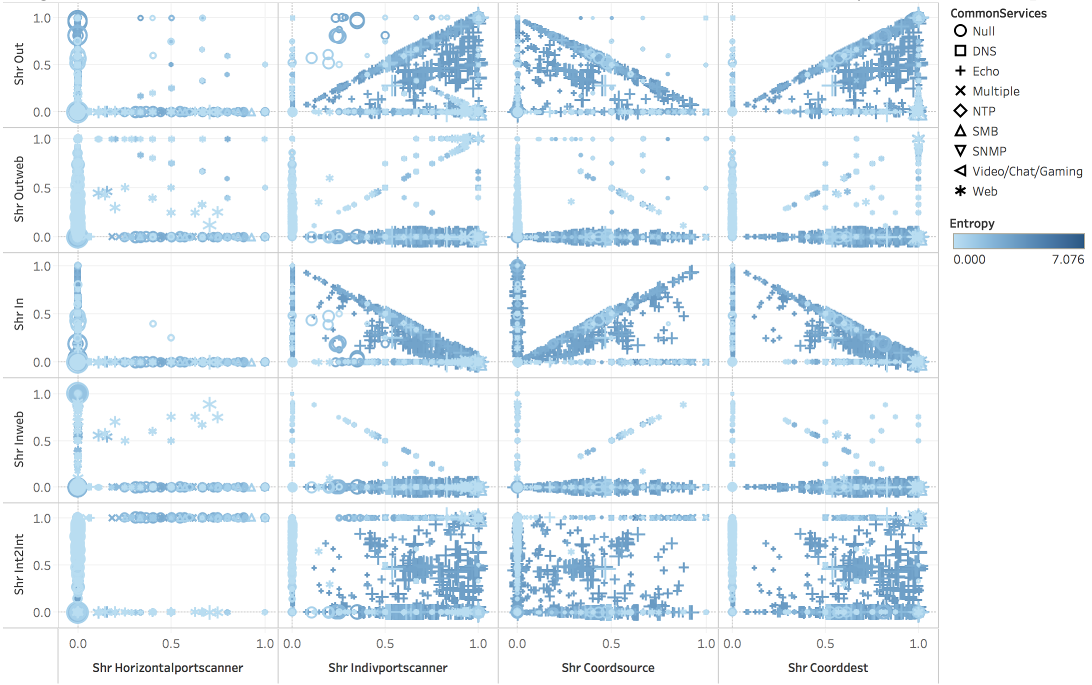


multiple destination IPs (at multiple companies and countries) simultaneously each minute. Approximately every 10 minutes, the set of destination IPs changes, while the set of companies remains constant (that is, every 10 minutes or so, the periodic connections switch from one set of destination IPs to a new set, within the same set of destination companies). A small sample of these destination IPs reviewed in Shodan were hosting 301 redirects to www[.]privateinternetaccess[.]com on port 8888 and so these logs are likely related to legitimate VPN service. There were VirusTotal malware samples associated with this address as well, and so this finding is an example of the type of automation that would need to be investigated further using additional context and network specific knowledge.

5.2 Exploring Pairwise Correlations of Output Features

The example in the previous section investigated activity yielding apparently anomalous or unusual behavior in one dimension (number of destination countries). Here, I begin to explore the pairwise correlations between multiple combinations of the features computed for each temporal group (see Table 1) in order to begin developing intuition for the relationships exhibited between these features for this dataset, with an aim to eventually move away from such manual inspection to a more automated anomaly detection framework applied to relevant features. Figure 4 presents scatter plots of groups of 4-tuple netflow entities found to

Figure 4: Cross-correlation of Netflow Coordination Features on One Day Window Outputs



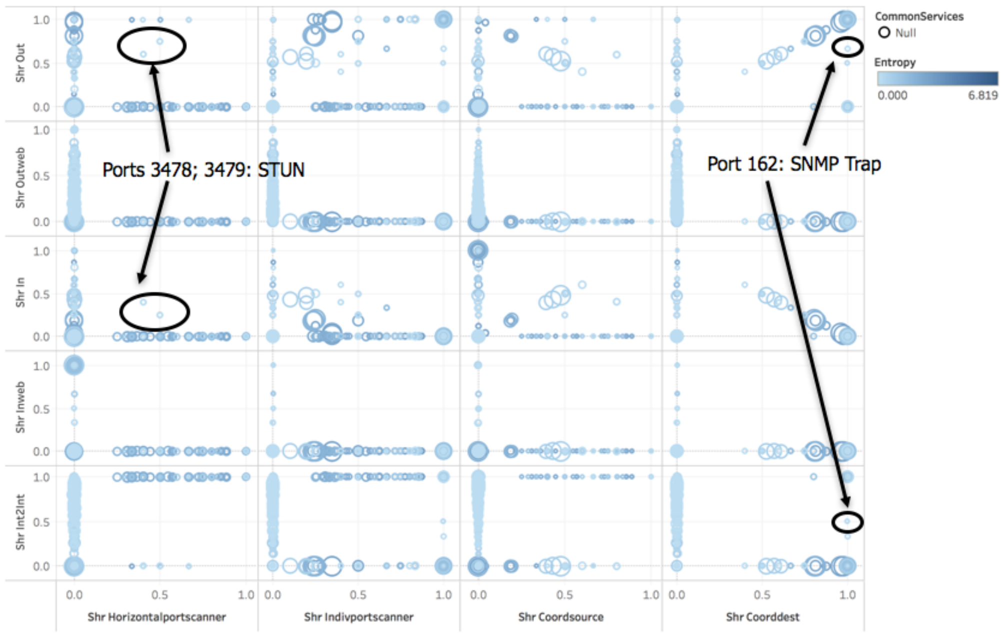
For this particular dataset, the most common ports appearing in coordination and my assigned service labels are as follows: 443-Web, 80-Web, 123-NTP, 2048-icmp echo request, 0-icmp echo reply, 53-DNS, 161-SNMP, 5228-Video/Chat/Gaming (specifically google playstore, google talk), 445-SMB, 7-TCP Echo, 135-Microsoft EPMAP (remotely manages services), and 5223 - Video/Chat/Gaming (Apple push notifications for MobileMe, FaceTime, etc. as well as other gaming). Null indicates none of these ports are included in the group, while multiple indicates that more than one of them is.

lie within a Jaccard distance of 0.1 of each other in the time signature space for a time window of length one day. Each point is shaded by the average interarrival shingle entropy value of the group so that lighter colored points indicate higher degrees of temporal regularity (with the extreme of entropy = 0 indicating perfectly periodic activity). In this plot, groups have also been labelled according to whether they include any entities with one of the 12 most common destination ports, where these ports are labelled and grouped together according to the service(s) commonly associated with each.

There are some strong pairwise correlations evident in these plots; namely, a concentration of groups along the positive 45 degree line for the share of coordinated sources and the share of entities with communication types of 'In' or 'Inweb', with no instances exhibiting positive 'Shr Coordsource' values where 'Shr In' or 'Shr Inweb' exceeds 'Shr Coordsource'. This 45 degree line is largely a an artifact of the unidirectional nature of SiLK flows.

The full set of pairwise scatters presented in Figure 4 allows us to see some high level patterns that are prevalent within this dataset in terms of characterizing the types of netflow activity that is identified as

Figure 5: Cross-correlation of Netflow Coordination Features on One Day Window Outputs Filtering out Groups with Common Ports

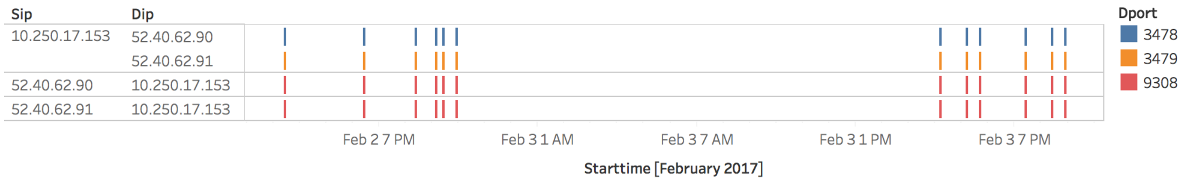


happening contemporaneously across entities. In order to uncover automated or coordinated activity that is not typical for the network, we can examine the same scatters after filtering out the more common types of coordinated activity. Figure 5 presents the same plots, now having filtered out all groups that contain any entities with one of the twelve most common destination ports, as a crude way to begin a manual search for anomalies and to continue to develop intuition about the types of activity the analytic flags. Filtering out these groups greatly reduces the noise in the scatters and reveals a few points that appear unusual relative to the majority, generally those lying on the interior region of a plot and not on a 45 degree line. Two such points are investigated further and detailed in subsections 5.2.1 and 5.2.2.

5.2.1 STUN Traffic to AWS Servers

After filtering out groups containing services commonly seen to occur within contemporaneous temporal groups, the only groups that contain entities categorized as 'horizontal port scanning' that also include

Figure 6: STUN Traffic with AWS Servers

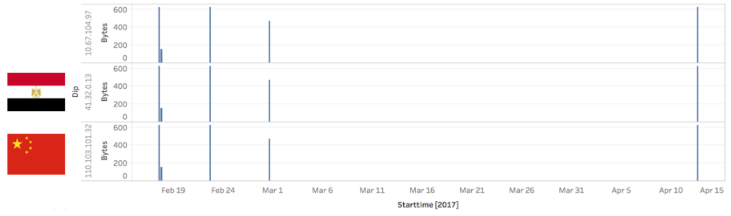


both 'In' and 'Out' communications types are those called out in the top left quadrant of figure 5. These groups consist of communications between an internal IP and two external IPs hosting Amazon AWS servers. While traffic between internal IPs and AWS servers is not uncommon on this network, it is uncommon to see bidirectional communications with an external IP on multiple ports simultaneously outside of the common ports filtered out here. Figure 6 shows the timestamps of communications for these two entity-days (one on Feb 2 and one on Feb 3). The destination ports 3478 and 3479 seen here are associated with Session Traversal Utilities for NAT (STUN), a service that allows for an internal device to determine what its external IP is by contacting a server that is sitting outside the Network Area Translator (NAT). Three possible uses of this service are: 1) End-to-end communication devices that need to know their external IP so they can be contacted (e.g. VoiP phones, some gaming devices, etc.); 2) Malware looking for where it is, by checking its IP using STUN; or 3) A research project run by Dan Kaminsky at WhiteOps to detect bots using STUN.[4] While the specific activity in this case involves destinations hosted on AWS, the observed activity could be any of the three and cannot be determined from Silk data. Based on RiskIQ reporting 52.40.62.80 being associated with stun.www.np.cn.s0.playstation.net and a common association of port 3479 with the Playstation Network, this instance is most likely due to someone playing video games on the network.

5.2.2 SNMP Traffic to Unusual Destinations

Another case in which filtering out the more commonly coordinated service ports yields an outlier along the feature dimensions of communication type and coordination type is a group of entities detected to have simultaneous activity from the same source IP to different destination IPs where these include both Internal to Internal ('Int2Int') and Internal to External ('Out') traffic. One of two such instances is a set of communications over port 162 depicted in Figure 7. Port 162 is typically used by Simple Network

Figure 7: Simultaneous SNMPTrap (Port 162) Traffic from 10.101.2.207 to Unusual Destinations



Management Protocol (SNMP) devices to send status updates to an SNMP management device. In this particular case, what is unusual is not that such communications occur simultaneously across multiple machines (indeed, this may be expected on a network where machines are configured to report status at specified times), but rather that these simultaneous communications include communications from an internal IP to external IPs as well as another apparently internal IP. The two external IPs in this case belong to telecom companies in Egypt and China while the internal destination IP does not appear to correspond to a real device on the network as it never initiates any connections and only ever receives connections from 10.101.2.207 and another internal device that appears to perform ping sweeps across the internal space. Thus, this particular anomaly is likely due to a network misconfiguration (or a device intended for another network being connected to this one), but provides an example of the type of unusual activity that can be flagged using these temporal features and sent up to analysts with knowledge of the network to rule out malicious activity.

[4] https://isc.sans.edu/forums/STUNtraffic/745/2/

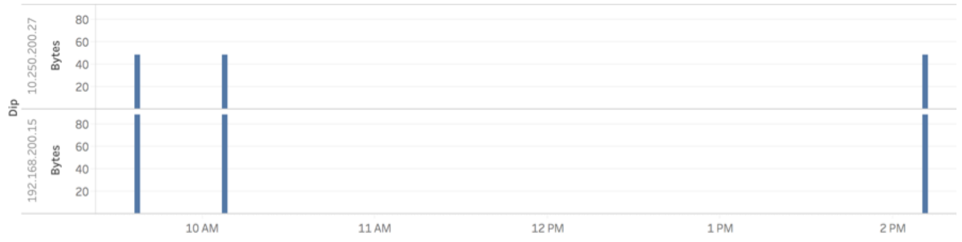
6 Disparate Dataset Matching

The results presented in section 5 show one way in which the time series pattern matching algorithm described above could be used in cyber data analysis: namely, by defining features over a unit of analysis which is derived from the output of that analytic (entity groups with similar temporal patterns), and then identifying groups that are unusual relative to the majority of observed network activity with respect to these features. A different potential use of the analytic could be to use the temporal matching algorithm in order to merge together disparate datasets that lack a common key but can be joined using entities present in multiple datasets that are identifiable by their temporal behavior. In principle, this could be used on different types of cyber logs to match network with host based data when a common key is missing, or more generally on any sets of multimodal data where one expects temporal signatures of activity from an entity to be present in multiple data sources. The examples presented in this section continue to use only the Silk data described above; as such, these are not actual examples of merging together disparate data sources but show how the same entity (IP, process, etc) might appear under different keys within a dataset, but can be identified based on temporal patterns. It is then straightforward to imagine cases where the activity of one entity appears in different logs or data sources under different keys, and can be matched together, thereby providing for a way to merge the datasets on this ’temporal signature’ key.

6.1 Translated IPs

One example of an entity being represented multiple times under different keys within flow data could be an instance of flows recorded under different IPs that actually represent the same host as appears to be the case in Figure 8.[5] Although the logs provide no explicit information directly linking these two internal IPs, we

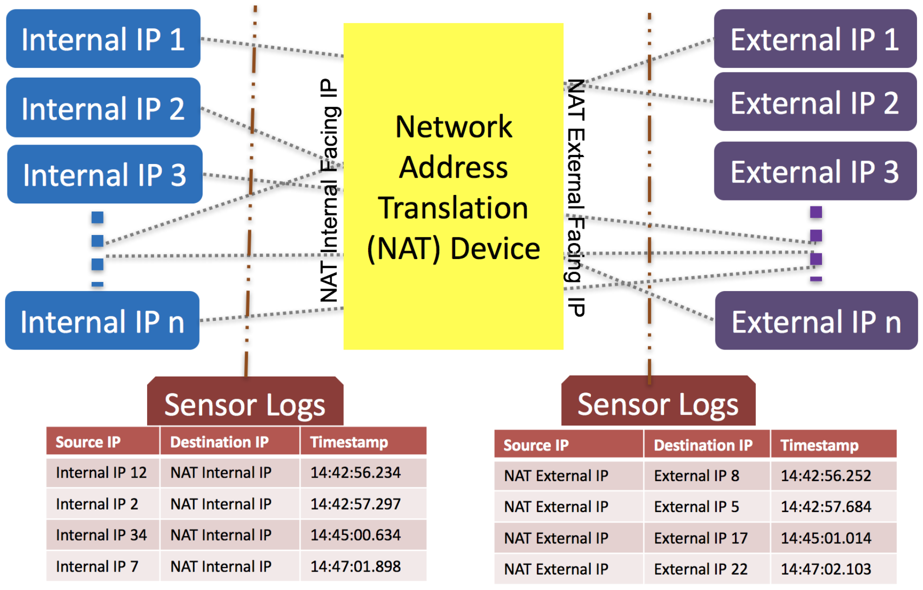
Figure 8: Identifying dual homed or NATTed IPs: Source IP 5.255.87.147 has simultaneous activity to two internal IPs



can infer from the matching time signatures of activity that they are actually the same host. Figure 9 depicts a hypothetical example of a network and sensor layout where multiple internal hosts connect to external IPs via one external facing IP address. With the sensor placement as shown, there would be no direct way to merge together the two sets of logs on a common join key unless we could assume that timestamps were unique and not subject to jitter (both likely false assumptions in practice). Using the methodology described above, however, rather than matching on individual timestamps, we can perform a fuzzy match on time signatures, also allowing for time offsets between sensors. This would then allow us to infer the flow

[5]It is not clear in this dataset without further knowledge of sensor placement and network architecture why logs would record in this way. One hypothesis is that a sensor which has visibility on both sides of a device that translates IPs (such as a Network Address Translator (NAT) device or firewall, for example) logs a given flow once for each address. If, in fact, these two internal IPs are distinct hosts, then the finding is more interesting as a rare occurence of an external host engaging in activity with two internal hosts in a coordinated manner.

Figure 9: Matching NATTed IP Flows



Source IP	Destination IP	Timestamp
Internal IP 12	NAT Internal IP	14:42:56.234
Internal IP 2	NAT Internal IP	14:42:57.297
Internal IP 34	NAT Internal IP	14:45:00.634
Internal IP 7	NAT Internal IP	14:47:01.898

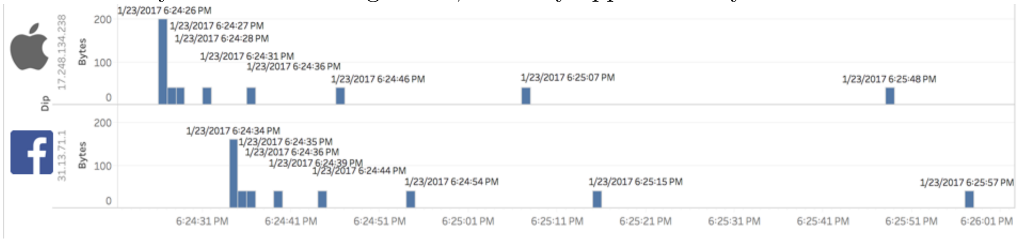
Source IP	Destination IP	Timestamp
NAT External IP	External IP 8	14:42:56.252
NAT External IP	External IP 5	14:42:57.684
NAT External IP	External IP 17	14:45:01.014
NAT External IP	External IP 22	14:47:02.103

connections represented by the dotted gray lines in the figure which are not explicitly captured in either individual log.

6.2 Related Processes

Expanding the notion of entity from a specific four-tuple or even IP/host to include higher level entities such as a process or other action that leaves behind traces in log data, we can think about merging together disparate datasets in which a given process might leave matching time signatures across datasets. As an example from only Silk data, consider the activity shown in Figure 10. Here, we see traffic over port 443 from an internal IP to two destination IPs belonging to different companies, where the temporal pattern of activity across the two series is nearly identical, modulo a lag of 8 seconds. Based on Cyber SME input, traffic to domains like Apple and Facebook with such temporal patterns are common, especially if someone leaves a web page up in their browser for long periods of time. In this instance, communications are mainly 40 bytes per instance, and likely indicate a 'heartbeat' keep-alive transmission for services that want to remain online and interactive. Because of the matching temporal pattern, we can infer that the same service is communicating with both.

Figure 10: Identifying Related Processes: Source IP 10.0.160.133 connects over port 443 to an Apple and a Facebook IP with nearly identical time signature, offset by approximately 8 sec



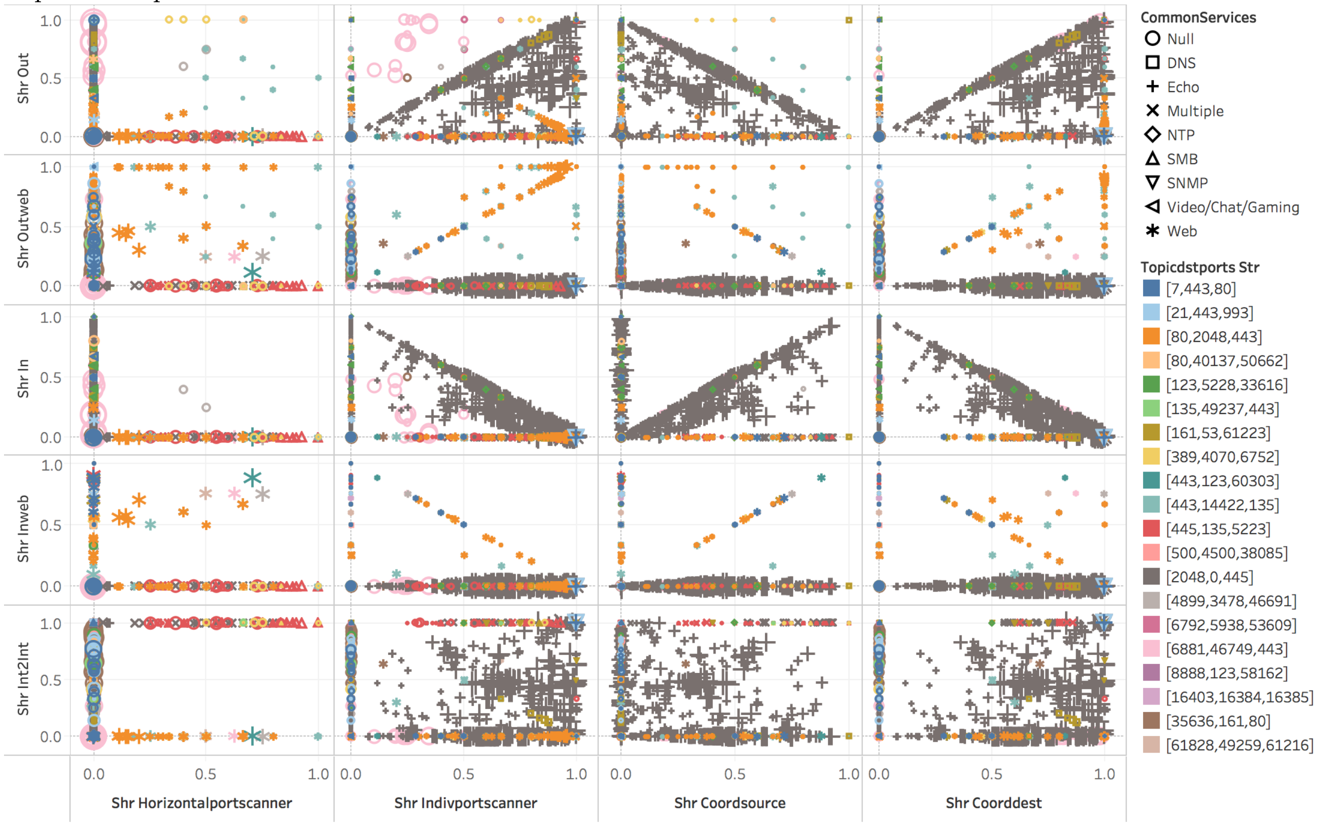
7 Future Work

Much of the development thus far of the analytic presented above has involved implementing the methodology in Apache Spark, building on the open source package spark-neighbors [2].[6] Moving forward, the focus of the research will be on developing second stage analysis pipelines to automatically flag potentially malicious behavior and applying the framework to new data types and use cases.

7.1 Moving Towards Automated Anomaly Detection on Temporal Group Features

The examples presented in section 5 were investigated due to one or more apparently anomalous features discovered by ad-hoc manual inspection of different feature distributions, filters, and pairwise correlations. Such examples are instructive and can inform future iterations of feature selection at both the entity definition stage as well as the specification of temporal group features defined on the groups of temporal neighbors identified by the Locality Sensitive Hashing algorithm described above. In future work, I plan to move

Figure 11: Cross-correlation of Netflow Coordination Features on One Day Window Outputs: Colored by Top LDA Topic Cluster



towards an automated framework where a second stage of anomaly detection or unsupervised learning is performed on the output units of the first stage coordination detection. Figure 11 shows the same scatter plots of output features as were depicted in figure 4, where now each point is colored by its assignment to a primary 'topic' as revealed by a second stage clustering analytic where a Latent Dirichlet Allocation

[6]Since this research effort began, Spark has added an implementation of LSH for Jaccard distance within its base ML library.

model is fit to the data to estimate the distribution of destination port 'topics' evident in each temporal group. Currently, this second stage clustering is still used to explore the output manually by observing correlations visually between topic clusters and other data features. In future research, however, I plan to develop this second stage to include automatic flagging of anomalies where the feature set feeding into this second stage analysis is informed by cyber expertise. In addition to the LDA port 'topic' identification presented here, I also plan to experiment with other unsupervised algorithms to classify temporal groups based on a multidimensional feature space that embeds information relevant to detecting malicious activity.

7.2 Combine Output with that of Other Temporal Analysis

The algorithm presented here applied to netflow data for the purpose of identifying coordinated behavior will tend to find automated activity, the vast majority of which is not linked to any malicious activity. For example, a mail server which is set up to send out messages to multiple IPs at the same time may appear in the 'targeted port scanning' results. Similarly, any host connecting to a specific destination or port at regular intervals (which could be the case for automated updates, for example) will exhibit a time pattern that matches (with some lag) all other such periodic check-in behavior. Other specific analytic approaches (in particular, using Fourier Transforms) can be applied to systematically detect and categorize such periodic behavior in large scale time series data (see [4]), and so one dimension along which we may wish to filter flagged coordinated activity is the regularity of the time pattern identified for a given coordinated set of entities. This can be measured, for example, by the entropy of the the interarrival shingles defined in 2.2.

7.3 New Data Types, Use Cases, and Other Extensions

The examples of applications of the analytic presented in this paper represent a small subset of the possible applications to cyber log data, which in turn represent a small subset of all possible applications. The methodology of performing anomaly detection on sets of entities identified via the coordination detection analytic to flag unusual instances of automated or coordinated activity can apply more broadly to any application in which automation detection is of interest, while the data fusion application can be extended even more widely. The specific network results presented here were restricted to the sample of netflow data used in this analysis, however, there are several other examples of cyber log data where one could imagine relevant use cases to detect malicious activity. One example of an application that I hope to research in the future would be to apply the temporal matching algorithm to time series of passive DNS records. Such records would identify the approximate timing of when a given domain changes the IP to which it resolves. In theory, malicious domains whose IPs change regularly to avoid detection might follow similar patterns of when these changes are made, if a certain actor re-uses the same automated process to update them. By obtaining passive DNS records covering a large swatch of domains on the internet over a sufficiently long period of time, I would be able to apply this analytic to filter down to sets of domains that exhibit similar temporal signatures, thereby possibly identifying a new temporal 'signature' associated with specific malware.

In addition to extending this research to new data types and use cases, I also plan to extend the functionality of the method as currently implemented to match not only binary time series indicators of activity, but to take into account magnitude of observed time series values, such as bytes transferred. The extension of the current methodology to allow for this is straightforward, and builds on pattern matching algorithms used for audio signal matching (see [6, 7] for example) as well as discrete time series matching algorithms such as dynamic time warping, and recent research developing a 'Sketch, Shingle, Hash' approach very closely related to the methodology presented here [5].

8 Conclusion

The ability to detect automated behavior within cyber relevant log data is a useful tool for the network defender, as malicious activity executed by scripts or bots is likely to leave behind traces in logs. The methodology presented here provides a means to detect certain types of automated activity within cyber logs based on matching observed temporal patterns. This methodology is scalable, overcoming the infeasibility of brute force methods to identify groups of nearest neighbors in large datasets by implementing a locality sensitive hashing algorithm. The coordination methodology presented here can be thought of as a means of feature generation for input into further analysis such as anomaly detection or unsupervised clustering to characterize automated behavior. Alternatively, the methodology could be used as a means to fuse together disparate data sources by generating a 'temporal signature' key and allowing for fuzzy matching on this key. There remain several avenues of future research open in applying and extending this methodology.

References

[1] CERT Software Engineering Institute, Carnegie Mellon University. System for Internet-Level Knowledge (SiLK). https://tools.netsa.cert.org/silk/

[2] https://github.com/karlhigley/spark-neighbors

[3] Leskovec, Jure, Anand Rajaraman, and Jeffrey David Ullman. Mining of massive datasets. Cambridge university press, 2014.

[4] Deason, Lauren. "Detecting Automated and Periodic Activity of Varying Time Scales (at Scale) in Cyber Relevant Log Data", working paper.

[5] Luo and Shrivastava (2016) "SSH (Sketch, Shingle, & Hash) for Indexing Massive-Scale Time Series" - http://proceedings.mlr.press/v55/luo16.pdf

[6] A. L.-C. Wang, "An Industrial Strength Audio Search Algorithm". (Shazam) - http://www.csie.ntu.edu.tw/~r95162/An%20Industrial-Strength%20Audio%20Search%20Algorithm.pdf

[7] J. Haitsma and T. Kalker, "A highly robust audio fingerprinting system," in International Conference on Music Information Retrieval, Paris, France, 2002. - http://www.ismir2002.ismir.net/proceedings/02-FP04-2.pdf