Detection of IP Gangs: Strategically Organized Bots

Tianyue Zhao[1,3(✉)] and Xiaofeng Qiu[2]

[1] Henry M. Gunn High School, Palo Alto, CA 94306, USA
zhaotianyue@outlook.com
[2] Beijing University of Posts and Telecommunications, Beijing, China
[3] NSFOCUS Inc., Santa Clara, CA 95054, USA

Abstract. Botnets, groups of malware-infected computers (bots) that perform cybersecurity attacks on the Internet, pose one of the most serious cybersecurity threats to many industries, including smart infrastructure [9, 10], Internet based companies, [11] and Internet of Things (IoT) [8]. There are many unconventional methods of organizing bots that are potentially advantageous to attackers. "Botnet", as a technical term, cannot effectively describe these methods. With the vast amounts of Internet traffic data collected by security appliances, it is possible to reveal novel behavior of bots using data analysis algorithms. In this paper, we propose a concept called IP Gang to describe groups of bots from the perspective of the attacker's business – we define IP Gangs to be groups of bots that often perform attacks together during a period of time. Crucially, we developed a fast, high-compatibility detection algorithm that can be deployed in wide-scale, industrial applications to effectively defend against IP Gangs. The detection algorithm is inspired by single-linkage clustering and optimized for large quantities of data. A test on a month (1.5 GB) of real life DDoS log data detected 21 IP Gangs, with 13916 bots in total. To analyze the behavior of the Gangs, we visualized the activity of each Gang with diagrams named "attack fingerprints" and confirmed that 15 of the detected Gangs displayed behavior that the concept of "botnet" alone cannot describe.

Keywords: IP gang · Botnet · Cybersecurity · Big data

1 Introduction

Botnets, groups of malware-infected computers (bots) that perform cybersecurity attacks on the Internet, pose one of the most serious cybersecurity threats to many industries. Smart infrastructure such as power grids have been attacked to deny hundreds of thousands of people basic services [9, 10]. Internet-based industries have been hit with massive Distributed Denial of Service (DDoS) attacks that can render large websites inoperative for entire hours [11]. Internet of Things (IoT) devices such as webcams are regularly hijacked to form bots [8], disabling them in their original purpose and severely disrupting IoT industry operations. With vast amounts of Internet traffic data collected by security appliances, it is possible to reveal novel behavior of bots using data analysis algorithms. Many aspects of botnets have been researched quite thoroughly [1–3], such as detection of botnets and communication patterns

P. Perner (Ed.): ICDM 2018, LNAI 10933, pp. 254–265, 2018.
https://doi.org/10.1007/978-3-319-95786-9_19

between bots and command/control (C&C) servers, but these are all technical studies, while few researches consider the perspective of the thriving botnet industry, which conducts cybersecurity attacks as a service.

The botnet industry seeks a high volume of DDoS attacks botnets can perform at any given time, low cost, and resistance to detection algorithms. These goals can be better achieved by organizing bots with unconventional methods, such as flexible organization.

Here are several scenarios that demonstrate the advantages of flexibly organizing bots from the point of view of the botnet industry: (1) Bots can be controlled as multiple small botnets with distinct technical properties – such as separate C&C servers and different C&C protocols – to evade detection. Many detection algorithms classify large groups of confirmed bots with identical technical properties as a botnet, so these small botnets with distinct properties are much harder to completely detect, and therefore have much better survivability. One way of implementing this is through the "super-botnet" structure proposed and analyzed by Vogt et al. [3]. (2) An attacker can utilize deceitful, advanced attack strategies, which are much more costly and time-consuming to defend against. Botnets could take distinct roles in a composite attack strategy. A known composite strategy is the usage of DDoS attacks as smoke-screens [4, 5] to draw defenders' attention and cover up other attacks. (3) Bots in places where it is night may be turned off. Making bots in places where it is day attack together allows for maximum guaranteed attack volume.

We recognize that the term botnet is not enough to describe the organization of bots. The definition of botnets is from a technical perspective, yet these advantageous scenarios of organizing bots can be achieved in many technical ways: by creating a network of small botnets each with its own C&C server, by dividing a large botnet into separately managed portions, and more. Therefore, the concept "botnet" is ill-suited at describing these new methods.

This necessitates a new, broad, industry-oriented concept that describes these ways of organizing bots. We propose the concept of IP Gang to meet this demand.

Analyzing IP Gangs allows for smarter, strategic defences. Analysis from the perspective of the cyber-crime industry allows defenders to study, truly understand, and most importantly strategically defend against the behaviors of the attackers. For example, attackers have threatened to launch attacks unless a ransom is paid [7], and the defender can decide to pay or not in a more informed manner thanks to the additional knowledge on the IP Gang. In another situation, if some bots belonging to an IP Gang starts attacking, it would save precious time to immediately quarantine other bots of the IP Gang.

In this paper, we proved the existence of IP Gangs in real life Internet traffic, and developed a fast, high-compatibility detection algorithm that can be deployed in wide-scale, industrial applications to effectively defend against IP Gangs.

For compatibility, we only use the start time, source IP address, and target IP address describing events in easily-obtainable Internet event log data, which widely deployed network security appliances generally output. Other parameters describing the events (such as bytes per packet) are optional, and may help with accuracy if present.

The algorithm is based on the principle of single-linkage clustering [6], but with an additional "packaging" step that reduces the number of nodes to be clustered to increase speed. The detection algorithm outputs each detected IP Gang as a list of IP addresses. The complexity of the algorithm is $O(n^2)$ with a small constant, where n is the number of events in the data. Our test on one month's events from a DDoS attack log detected 21 IP gangs, and showed that our algorithm is fast enough to be used to detect and help defend against IP Gangs on a large scale.

The rest of this paper is structured as follows. Section 2 presents previous work on botnet structure and botnet detection with Internet traffic. In Sect. 3, the formal definition of IP Gang is introduced and its relationship to botnets is analyzed. The principle and algorithm used to detect IP Gangs are detailed in Sect. 4. Next, the test results on real data are presented. We conclude by discussing future work.

2 Related Work

Botnet structures that may provide advantages similar to those of IP Gang's have been studied. Most notably, Vogt et al. [3] proposed the "super-botnet", a network of small, centralized botnets that can perform coordinated attacks, and provided detailed technical analysis of super-botnets. Individual botnets in a super-botnet can be detected, but it is very hard to detect the entire super-botnet. This additional resilience allows attackers to accumulate enough bots to perform very large-scale attacks. However, Vogt et al. did not provide experimentation on real life data. The concept of super-botnet is possibly related to the concept of IP Gang, but is still fundamentally different - it is still a technical definition, while the definition of IP Gang is business-oriented.

There had been a lot of researches on the detection of botnets based on Internet traffic. Gu et al. [1] was one of the first to propose and test on real life traffic data a clustering-based botnet detection model, which provides a variety of advantages over previous models that detect botnets by scanning for Command and Control (C&C) traffic between bots and the attacker. Gu et al. provided experimentation on real life data and analysis of the results.

3 Definition of IP Gang

As discussed in Sect. 1, the concept of IP Gangs and botnets are not comparable. IP Gangs and botnets consider the business and technical perspectives of groups of bots respectively. The former is concerned with how the attacker organizes his bots to his advantage, while the latter is instead mainly concerned with how the bots communicate with each other and with the controller.

Definition: *An IP Gang is a group of malware-infected computers (bots) that are controlled by the same attacker and often launch attacks directed at the same target within a short period of time t.*

A botnet is a group of bots that are organized by a certain network architecture and controlled by the same C&C (Command and Control) protocol by a logically centralized C&C server. Usually, the bots in a botnet have the same behavior from a technical point of view.

It is worth noting that, by definition, all the bots in a botnet always receive the same command, while bots in an IP Gang are not subject to such constraint.

The concept of IP Gang is suitable for describing the spatial and temporal features of organized bots, which could be in a same botnet or belongs to different botnets (Fig. 1). IP Gangs can be intentionally formed by attackers, or unintentionally formed due to logistic conditions. Bots of a botnet that are located in the same time zone may frequently be available to attack at the same time, thereby qualifying as an IP Gang.

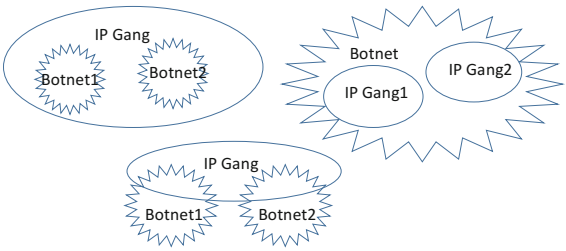

Fig. 1. IP gang and botnet

4 IP Gang Detection Algorithm

4.1 Overview

The target of the algorithm is to cluster the IP addresses of bots based on cybersecurity event log data to detect IP Gangs. We designed the method to meet the following challenges:

(1) *Speed: the method must be fast enough to be deployed on networks which produce large volumes of log data.*
(2) *Use as little information as possible: to ensure compatibility, the method must use only the most basic attack event information found in virtually all event logs: start time, source IP address, and destination IP address.*

At the core of the detecting algorithm is a clustering algorithm inspired by single-linkage clustering. Clustering algorithms are inherently time-intensive, and ours yields a complexity of $O(N^2)$, where N is the number of nodes to be clustered. Subsequently, reducing the number of nodes is crucial to the speed of the detection algorithm, and we add a packaging step before the clustering step to accomplish this. The algorithm consists of three steps, as illustrated in Fig. 2.

(1) *Packaging: Events reported by security devices are grouped into Organized Attack Events (OAEs).*

(2) *Clustering: OAEs are clustered using a method inspired by single-linkage clustering with each finished cluster, named OAE cluster, representing a Gang.*
(3) *Analyzing: OAE clusters are analyzed to find Gangs of IP addresses. This can also be seen as a "de-packaging" step, extracting IP addresses from OAE clusters.*

Each step of our procedure is analyzed in greater detail in the rest of this section, and the performance of the procedure when run on real life data is discussed in the experimentation section.

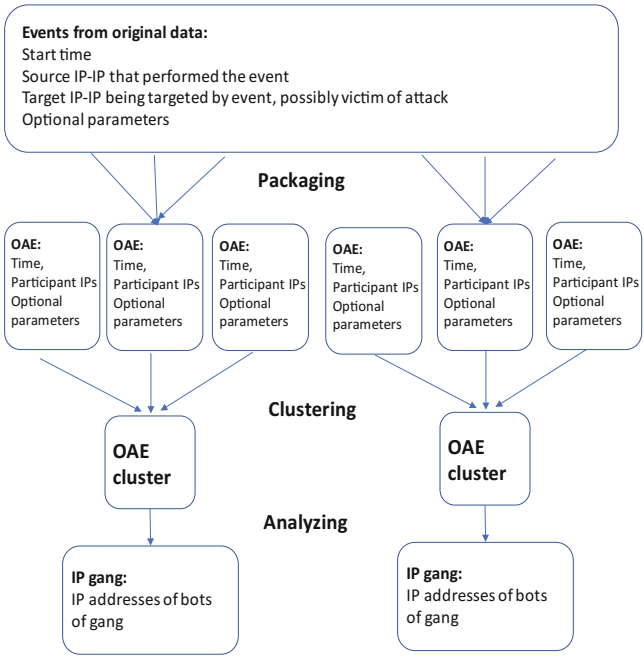

Fig. 2. Procedure of IP gang detection algorithm

4.2 Packaging: Constructing Organized Attack Events (OAEs) from Original Data

To decrease the number of nodes that need to be clustered in the clustering stage, we designed a way of packaging individual events with the same target IP address and starting in a time interval t into entities called Organized Attack Events (OAEs), which will be nodes in the clustering step.

The definition of an OAE is:

Given that A is a group of attack events, A_{IP} is the set of all of the source IP address in A. A is an OAE iff

$$\begin{aligned} &\forall_{a_1,a_2} \in A, \\ &|a_{1.starttime} - a_{2.starttime}| \leq t \\ &\& \; a_{1.targetIP} == \; a_{2.targetIP} \\ &\& \; |A_{.IP}| \geq min\_size \end{aligned} \tag{1}$$

The set $A_{.IP}$ contains the IP addresses of all the bots that launched this set of Organized Attack Events (OAE).

The optimal value of time t varies between log data types.

Generally, attackers will simultaneously utilize a large number of bots in each organized attack for maximum effectiveness. Here we use this property to keep organized attacks – which are of use to us – and discard unorganized, individual attacks by filtering out OAEs with less than *min_size* events in them.

The pseudocode for the packaging process is:

```
id=0, and cur_OAE is initialized to empty
for event in D: #D denotes the input data
    if(|event.start_time-cur_OAE.start_time|<=1 minute)...
        and (event.target_IP==cur_OAE.target_IP):
        cur_OAE.size+=1
        cur_OAE.IP.append(event.source_IP)
        optional_params_tmp.append(event.optional_params)
    else:
        if(cur_OAE.size>=minimum_size): #OAE is disposed if not big enough
            cur_OAE.optional_params=take_most_common_values(optional_params_tmp)
                #if the most common set of values for the
            #optional params do not account for at least
            #50% of the entries in the optional_params_tmp
            #list, cur_OAE.optional_params is set to None

            A.append(cur_OAE)
        cur_OAE.id=id
        id+=1
        cur_OAE.size=1 #initialization with only current event
        cur_OAE.IP=[event.source_IP]
        cur_OAE.optional_params=None
        cur_OAE.start_time=event.start_time
        cur_OAE.target_IP=event.target_IP
```

At the start, the set of current OAE, *cur_OAE*, is initialized to contain only the first event. Afterwards, if the event being processed starts within t of *cur_OAE*, and has the same target IP address, it is added to *cur_OAE*. *cur_OAE* is added to the list of OAEs if the number of events it contains exceeds *min_size*. Otherwise, it is discarded and re-initialized.

This step has linear time complexity, and therefore is insignificant in terms of runtime. However, by using OAEs, instead of individual events in the clustering step, we decreased the number of nodes to be clustered by a very large factor, without sacrificing accuracy.

4.3 Clustering: Clustering OAEs to Form OAE Clusters

The OAEs formed in the packaging step are then clustered to form OAE clusters. Given A_1, A_2 are two OAEs, $A_{1.IP}$ as the set of all of the source IP address in A_1, A_1 and A_2 must be put in the same cluster if:

$$s = \frac{|A_{1.IP} \cap A_{2.IP}|}{|A_{1.IP}|} \geq combining\_threshold,$$
$$assuming\ |A_{1.IP}| \leq |A_{2.IP}| \tag{2}$$

In (2), s measures the normalized similarity of bots in two OAEs. Two OAEs with similarity larger than *combining.threshold* should be put in the same OAE cluster.

The clustering algorithm is inspired by single-linkage clustering, but is different in a crucial way. Single linkage clustering merges the two most similar clusters in each merge step, and stops performing merge steps when a reasonable total number of clusters have been reached. In contrast, we perform all possible merges of OAE clusters satisfying Eq. (2). In words, the clustering algorithm merges pairs of clusters that contain OAEs with a similarity score higher than the combining threshold but not yet in the same cluster. We proceed until no such pair exists.

The clustering algorithm is performed with a disjoint set data structure. For each OAE, the clustering algorithm computes the similarity scores between this OAE and all other OAEs. If the similarity score of two OAEs is higher than the combining threshold and the two OAEs are not yet in the same OAE cluster, the OAE clusters of the two OAEs are merged with a *union* operation.

The pseudo code is as follows:

```
1 for OAE A1 in set of all p OAEs:
2     count={} #counts the number of IP addresses
3              #each OAE has with the current OAE
3     for each IP address ip1 in A1.IP:
4         relevant=find(every OAE that contains ip1)
5         #The time this operation takes is near
6         #constant when a key-value database is used
7
8         #Each entry in "relevant" is an OAE that has
9         #at least 1 IP address in common with A1
10
11        #The number of entries in "relevant" is
12        #proportional to q, the average number
13        #of OAEs an IP address contributes to
14        for OAE A2 in relevant:
15            if(A1 and A2 are already in same cluster):
16                skip iteration
17            if(A2.id is in count):
18                count[A2.id]+=1
19            else:
20                count[A2.id]=1
21        for A2 in count:
22            if(A1 and A2 are already in same cluster):
23                continue
24            if(|A1.IP|<=|A2.IP|):
25                s=count[A2]/|A1.IP|
26            else:
27                s=count[A2]/|A2.IP|
28            if(s>=combining threshold):
29                merge the OAE clusters A1 and A2 are in
30                       with "union" operation
```

In short, the clustering step puts OAEs that are performed by bots of the same gang into the same OAE cluster. This is achieved through computing the percentage of participating IP addresses two OAEs have in common and then merging the clusters the two OAEs are in if the percentage is higher than a threshold.

A NOSQL database is integral to our clustering method, as it greatly speeds up our method. With a relational database, the query at line 4 is very time-consuming, but NOSQL databases can perform this in a near constant time.

The complexity of this implementation is $O(n^2)$, where n is dataset size – the number of individual events in the log data. The number of iterations in the *for* loop in line 1 is p, the total amount of OAEs. The *for* loop in line 3 is independent from data size. The number of iterations in the *for* loops of lines 14 and 21 are both q, the average number of OAEs an IP address contributes to. Therefore, the overall complexity is $O(pq)$. It is apparent that $p \propto n$, as the average number of events in each OAE does not change. We expect $q \propto n$, though the correlation between the two is less strong. Therefore, $pq \propto n^2$, and the complexity is $O(n^2)$.

4.4 Analyzing: Identifying Gangs by Analyzing OAE Clusters

After OAE clusters are formed in the previous step, we analyze the OAE clusters to identify gangs of bots, with each bot represented by an IP address. We collect all the IP addresses that have participated in an OAE of an OAE cluster, and only retain IP addresses that have participated in enough OAEs of that cluster.

In words, for each OAE cluster, we calculate the percentage of OAEs in the cluster each IP address contributed to. IP addresses that only contribute to a very small percentage of OAEs may be treated as noise, as they are generally not worthy of studying. A threshold named *validation.threshold* is set in Sect. 5 of this paper and only the IP address with a contribution percentage larger than the threshold is retained into a gang.

5 Experimentation on Real Life Data

5.1 Overview

The data we used for experimentation is a DDoS log consisting of individual DDoS attack events collected from January 1st, 2016 to January 31st, 2016. The reports of DDoS attacks are collected from several dozens of NSFOCUS Network Traffic Analyzers (NTAs) and Anti-DDoS Systems (ADSs). NTAs and ADSs are deployed at the sites of the customers of NSFOCUS, and construct the DDoS log by analyzing netflow, an industry standard type of metadata.

Our algorithm was written in Python, used the Neo4j graph database, and was ran on a 2012 Thinkpad X230i laptop with hyper-threading disabled. The clustering step took 48 h with the full 1.5 GB of data, and the other steps were insignificant in terms of runtime.

With a *validation.threshold* of 0.05 and a *combining.threshold* of 0.6, our algorithm detected 17350 OAEs and 21 IP Gangs that has at least 10 OAEs. In total, there are 13916 valid bots in all these gangs.

On average, each OAE contained 183 individual events. This means that the packaging step decreased the runtime of the clustering step by a factor of 183^2.

5.2 Discussion of the Parameters

In the "packaging" step, the optimal time t in Eq. (1) for our data is found to be 1 min. We observed in our log data that large groups of events that have the same target IP address typically have start times within 1 min of each other. Running the packaging step on our data with several different t values confirm $t = 1\ min$ as the optimal value. We determined the optimal value of *min_size* in Eq. (1) to be 50 with a similar procedure. In fact, we conducted tests with $t = 1, 3, 5\ min$ and $min\_size = 20, 30, 50$, achieving very similar results in each test. We therefore chose $t = 1\ min$ and $min\_size = 50$ to maximize accuracy.

In the "Analyzing" step, the value of *validation.threshold* greatly influences the final output of IP addresses in an IP gang. As shown in Fig. 3, different values of *validation.threshold* resulted in large variations in the total number of IP addresses in these 21 gangs. *Validation.threshold* provides a mechanism to look into an IP gang in different levels of granularity. For example, a larger *validation.threshold* will only output core members of an IP gang so that the defender could monitor the IP gang more efficiently. On the other hand, a smaller threshold will help the defender to get more detailed information on an IP gang.

5.3 Visualization and Discussion

We developed a type of diagram that visualizes the attack patterns of IP Gangs, which we denote the "attack fingerprint". Each attack fingerprint represents a Gang, and each red dot on attack fingerprint represents an individual attack event by a bot of the Gang. The *ID* numbers of OAEs are assigned in time order, so the Y-axis is practically a relative measure of time. The X-axis is the *ID* of the bot, so each column of the figure represents the temporal behavior of a bot. The fingerprints in Fig. 4(a), (b), and (c) have $validation.threshold = 0.05$, while the one in Fig. 4(d) has $validation.threshold = 0.1$.

In 6 of the 21 fingerprints, we see that all the columns have nearly the exact same appearance. This tells us that each bot in the gang participated in almost the same OAEs. The attack fingerprints in Fig. 4(c) is an example. This kind of fingerprints can be explained with the conventional concept of botnets, because all the bots are behaving in the same way.

The other 15 fingerprints are difficult to describe with only the concept of botnets, because the behavior of bots often differ from each other. As shown in Fig. 4(a) and (b), the columns take a small number of distinct but still similar appearances. In certain rows, all columns have red dots, but the columns take several distinct patterns in other rows. This shows that the bots are not always behaving in the same way. Notably, different values of *validation.threshold* allows for different parts of the Gang to be analyzed in detail. Figure 4(d) demonstrates this, as it describes the same Gang as

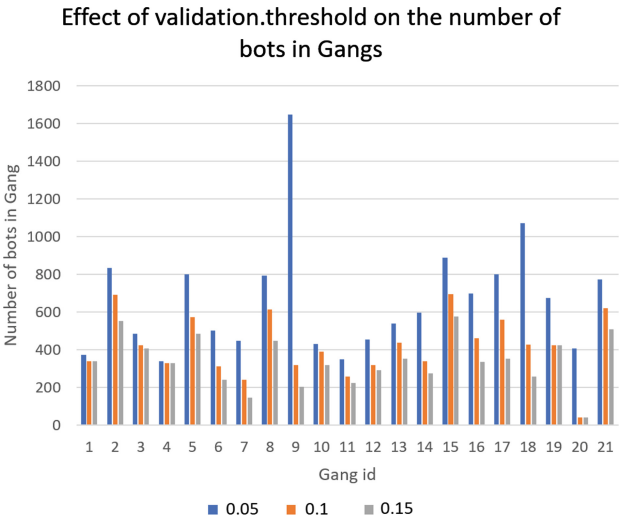

Fig. 3. Influence of *validation.threshold*

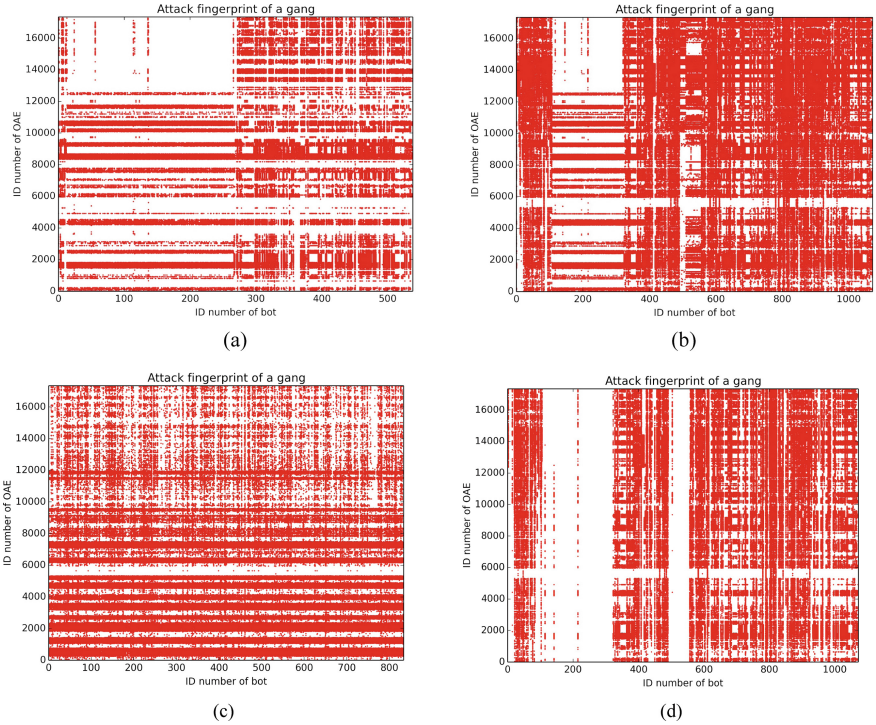

Fig. 4. Fingerprints of gangs (Color figure online)

Fig. 4(b), but is drawn with *validation.threshold = 0.1*. Clearly, the bots with *id* between 100 and 300 are omitted from the graph, while the other parts are preserved. There are several plausible explanations for this phenomenon. For example, in Fig. 3 (a), there may be two botnets, one with bots *ID < 280*, and another with bot *ID > 280*. Sometimes they attack together as shown by the dense horizontal lines, and sometimes they attack separately as shown by the upper part of the fingerprint with OAE *ID > 12000*. Another explanation is that these bots belong to the same botnet, but sometimes only part of the botnet are able to successfully carry out the attack.

6 Conclusions and Future Work

We demonstrate that IP gangs exist on the internet and are actively being used by attacker to perform DDoS attacks. They are detectable using clustering-based algorithms and can be distinguished from conventional botnets.

Analyzing the behavior of IP gangs will be highly beneficial. Doing so can make for a better understanding of the operation, structure, and performance of IP gangs. A more thorough understanding of these is necessary to accurately assess the threat that IP gangs pose to cybersecurity, and to defend against IP gangs. For defenders, knowing more about the behavior of Gangs can allow for smarter, strategic defences against Internet attacks.

References

1. Gu, G., Perdisci, R., Zhang, J., Lee, W.: BotMiner: clustering analysis of network traffic for protocol-and structure-independent botnet detection. In: USENIX Security Symposium, vol. 5, no. 2, pp. 139–154 (2008)
2. Khattak, S., Ramay, N.R., Khan, K.R., Syed, A.A., Khayam, S.A.: A taxonomy of botnet behavior, detection, and defense. IEEE Commun. Surv. Tutor. **16**(2), 898–924 (2014)
3. Vogt, R., Aycock, J., Jacobson, M.: Army of botnets. In: Proceedings of NDSS 2007 (2007)
4. Arbor Networks: DDoS as a smokescreen for fraud and theft, 3 February 2016. https://www.arbornetworks.com/blog/insight/ddos-as-a-smokescreen-for-fraud-and-theft/
5. Kaspersky Lab: Research reveals hacker tactics: Cybercriminals use DDoS as smokescreen for other attacks on business, 22 November 2016. https://www.kaspersky.com/about/press-releases/2016_research-reveals-hacker-tactics-cybercriminals-use-ddos-as-smokescreen-for-other-attacks-on-business
6. Stanford Natural Language Processing Group: Single-link and complete-link clustering (n.d.). https://nlp.stanford.edu/IR-book/html/htmledition/single-link-and-complete-link-clustering-1.html
7. WeLiveSecurity: Spammed-out emails threaten websites with DDoS attack on September 30th, 25 September 2017. https://www.welivesecurity.com/2017/09/25/email-ddos-threat/
8. Kolias, C., Kambourakis, G., Stavrou, A., Voas, J.: DDoS in the IoT: mirai and other botnets. Computer **50**(7), 80–84 (2017)
9. Pultarova, T.: Cyber security - Ukraine grid hack is wake-up call for network operators. Eng. Technol. **11**(1), 12–13 (2016)

10. Khan, R., Maynard, P., McLaughlin, K., Laverty, D., Sezer, S.: Threat analysis of BlackEnergy malware for synchrophasor based real-time control and monitoring in smart grid. In: Janicke, H., Jones, K., Brandstetter, T. (eds.) 4th International Symposium for ICS & SCADA Cyber Security Research 2016, pp. 53–63 (2016)
11. Kaspersky Lab: Attack on Dyn explained. https://www.kaspersky.com/blog/attack-on-dyn-explained/13325/