EURECOM
Sophia Antipolis

Institut Eurécom

Research Report RR-09-226

HONEYPOT TRACES FORENSICS: THE OBSERVATION VIEW POINT MATTERS

February 12th, 2009

Van-Hau Pham* and Marc Dacier**
*Institute Eurécom, Sophia Antipolis, France
**Symantec Research Labs, Sophia Antipolis, France

Tel : (+33) 4 93 00 81 00
Fax : (+33) 4 93 00 82 00
Email : Van-Hau.Pham@eurecom.fr, Marc_Dacier@symantec.com

[1]Institut Eurécom's research is partially supported by its industrial members: BMW Group Research & Technology - BMW Group Company, Bouygues Télécom, Cisco Systems, France Télécom, Hitachi Europe, SFR, Sharp, STMicroelectronics, Swisscom, Thales.

HONEYPOT TRACES FORENSICS: THE OBSERVATION VIEW POINT MATTERS

Van-Hau Pham and Marc Dacier

Abstract

In this paper, we propose a method to identify and group together traces left on low interaction honeypots by machines belonging to the same botnet(s) without having any a priori information at our disposal regarding these botnets. In other terms, we offer a solution to detect new botnets thanks to very cheap and easily deployable solutions. The approach is validated thanks to several months of data collected with the worldwide distributed Leurré.com system. To distinguish the relevant traces from the other ones, we group them according to either the platforms, i.e. targets hit or the countries of origin of the attackers. We show that the choice of one of these two observations view points dramatically influences the results obtained. Each one reveals unique botnets. We explain why. Last but not least, we show that these botnets remain active during very long periods of times, up to 700 days, even if the traces they left are only visible from time to time.

Contents

List of Figures

1 Introduction

There is a consensus in the security community to say that botnets are today's plague of the Internet. A lot of attention has been paid to detect and eradicate them. Several approaches have been proposed for this purpose. By identifying the so called *Command and Control (C&C)* channels, one can keep track of all IPs connecting to it. The task is more or less complicated, depending on the type of *C&C* (IRC [2, 4, 6, 7, 14, 20], HTTP [3, 5, 23], fast-flux based or not [12, 16, 21], P2P [8, 13, 24, 26], etc.) but, in any case, one needs to have some insight about the channels and the capability to observe all communications on them. Another approach consists in sniffing packets on a network and in recognizing patterns of *bot-like* traffic. This is, for instance, the approach pursued by [9–11] and [22, 25]. The solutions mostly aim at detecting compromised machines in a given network rather than to study the botnets themselves as they only see the bots that exist within the network under study.

In this work, we are interested in finding a very general technique that would enable us to count the amount of various botnets that exist, their size and their lifetime. As opposed to previous work, we are not interested in studying a particular botnet in details or in detecting compromised nodes in a given network. We also do not want to learn the various protocols used by bots to communicate in order to infiltrate the botnets and obtain more precise information about them [20]. By doing so, we certainly will not be able to get as much in depth information about this or that botnet but our hope is to provide insights into the bigger picture of today's (and yesterday's) botnets activities.

Before describing our approach, it is crucial to understand the subtle difference that exists between counting the amount of machines launching a given attack and the amount of machines members of a given botnet. It is very misleading to believe that one can derive the latter from the former. Indeed, it is quite common to see several distinct botnets relying on the same attack vector to compromise more hosts. In such case, the total amount of machines observed using a given attack vector will be greater or equal to the sum of all members of all these botnets (it can be greater as machines not belonging to any botnet may also launch this attack). Clearly, any approach relying on simply summing up counters based on attack vectors characteristics (e.g. ID alerts, firewall logs, AV detection, etc.) is likely to grossly overestimate the size of botnets.

The solution described in the following is generic and simple to deploy widely. It relies on a distributed system of low interaction honeypots. Based on the traces left on these honeypots, we provide a technique that groups together the traces that are likely to have been generated by groups of machines controlled by a similar authority. Since we have no information regarding the *C&C* they obey to, we do not know if these machines are part of a single botnet or if they belong to several botnets that are coordinated. Therefore, to avoid any ambiguity, we write in the following that they are part of a *army of zombies*. An *army of zombies* can be a single botnet or a group of botnets the actions of which are coordinated during a given time interval.

In this paper, we propose a technique to identify and study the size as well as the lifetime of such *armies of zombies*. We show that armies can stay active for very long periods of time, up to 700 days, even if they manifest themselves only from time to time. The approach does not pretend to be able to identify all *armies of zombies* that could be found in our dataset. At the contrary, we show that, depending on how the dataset is preprocessed, i.e. depending on the observation viewpoint, different armies can be found. Exhaustiveness is not our concern at this stage but, instead, we are interested in offering an approach that could easily be widely adopted and that offers a much better picture of the reality of the problem.

The idea exposed here is similar, in its spirit, to the one presented in the paper coauthored by Allmann et al. [1]. However, instead of " [...] *leveraging the deep understanding of network detectives and the broad understanding of a large number of network witnesses to form a richer understanding of large-scale coordinated attackers*", our approach relies on a diverse yet limited number of low interaction honeypots. They do not need to be neither as smart as the network detectives nor as numerous as the network witnesses proposed in that work. Both approaches are quite complementary.

The reminder of the paper is organised as follows. Section 2 defines the terms used in the paper. Section 3 describes the dataset we have used and what we mean when we refer to the notion of *observation viewpoint*. It also explains why it matters when trying to identify *armies of zombies*. In Section 4, we describe the method itself that we have applied to find these armies, we provide the main characteristics of the results obtained as well as two precise, yet anecdotal, examples of armies detected thanks to our method. Finally, Section 5 concludes the paper.

2 Terminology

In order to avoid any ambiguity, we introduce a few terms that will be used throughout the text.

- **Platform**: A physical machine simulating, thanks to honeyd [19], the presence of three distinct machines. A platform is connected directly to the Internet and collects tcpdump traces that are fed on a daily basis into the centralized Leurré.com's database.

- **Leurré.com**: The Leurré.com project is a distributed system of platforms as defined earlier, deployed in more than 50 different locations in 30 different countries. More detailed information about it can be found in [15]

- **A Source** corresponds to an IP address that has sent at least one packet to, at least, one platform. It is important to understand that a given IP address can correspond to several distinct sources. Indeed, a given IP remains associated to a given source as long as there is no more than 25 hours between 2 packets received from that IP. After such a delay, a new source identifier will be assigned to the IP. By grouping packets by sources instead of by IPs, we minimize the risk of gathering packets sent by distinct physical machines that have been assigned the same IP dynamically after 25 hours.

- **A Cluster** is made of a group of sources that have left highly similar network traces on all platforms they have been seen on. Clusters have been precisely defined in [18]. They aim at grouping together attackers that are likely launching attacks with the very same attack tool.

- **A Cluster time series** $\Phi_{T,c}$ is a function defined over a period of time T, T being defined as a time interval (in days). That function returns the amount of sources per day associated to a cluster c.

- **An Observed cluster time series** $\Phi_{T,c,op}$ is a function defined over a period of time T, T being defined as a time interval (in days). That function returns the amount of sources per day associated to a cluster c that can be seen from a given *observation view point* op. The observation view point can either be a specific platform or a specific country of origin. In the first case, $\Phi_{T,c,platform_X}$ returns, per day, the amount of sources belonging to cluster c that have hit $platform_X$. Similarly, in the second case, $\Phi_{T,c,country_X}$ returns, per day, the amount of sources belonging to cluster c that are geographically located in $country_X$. Clearly, we always have: $\Phi_{T,c} = \sum^{\forall i \in countries} \Phi_{T,c,i} = \sum^{\forall x \in platforms} \Phi_{T,c,x}$

- **An attack event** is defined as a set of observed cluster time series exhibiting a particular shape during a limited time interval. This time interval typically lasts a couple of days but it can be as short as a single day in the case of observed cluster time series having a one day peak of activities. The existence of attack events highlights the coordinated activities of several attacking machines. It is important to notice that the set can be singleton. This is typically the case when the set is a peak of activities on a single day.

  We denote the attack event i as $e_i = (T_{start}, T_{end}, S_i)$ where the attack event starts at T_{start}, ends at T_{end} and S_i contains a set of observed cluster time series identifiers (c_i, op_i) such that all $\Phi_{[T_{star}-T_{end}),c_i,op_i}$ are strongly correlated to each other $\forall (c_i, op_i) \in S_i$. As an example, the top plot of Figure 1 represents the attack event 225 which consists of cluster 60332 (targeting port 5900 TCP) attacking seven platforms 5,8, 11, ...,31. Each curve represents the amount of sources of that cluster observed from one of these platforms. As we can observe, the attack event start at day 393 and ends at day 400. According to our convention, we have $e_{225} = (393, 400, \{(60232, 5), (60232, 8), ..., (60232, 31)\})$.

  Similarly, the bottom plot of Figure 1 represents the attack event 14 which consists of activities of cluster 0 on day 307 coming almost only from Spain. So, $e_{14} = (307, 307, \{(0, ES)\})$

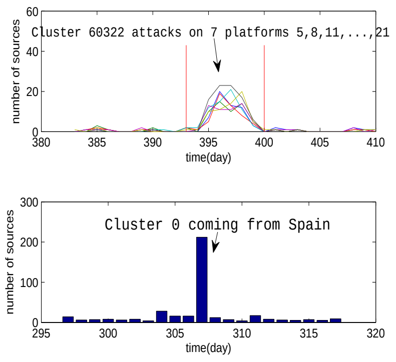


Figure 1: on the top plot, cluster 60232 attacks seven platforms from day 393 to day 400. On the bottom plot, peak of activities of cluster 0 from Spain on day 307

3 IMPACT OF OBSERVATION VIEW POINT

3.1 Dataset Description

In order to have a clean dataset for our experiments, we have selected the traces observed on 40 platforms out of the 50 that we had at our disposal. All these 40 platforms have been running for more than 800 days. During this period, none of them has been down for more than 10 times and each of them has been up continuously for at least 100 days at least once. They all have been up for a minimum of 400 days over that period. The total amount of sources observed, day by day, on all these 40 platforms can be denoted by the initial time series TS over a period of 800 days. We can split that time series per country[1] of origin of the sources. This gives us 231 time series TS_X where the i^{th} point of such time series indicates the amount of sources, observed on all platforms, located in country X. We represent by TS_L1 the set of all these Level 1 time series. To reduce the computational cost, we keep only the countries from which we have seen at least 10 sources on at least one day. This enables us to focus on 85 (the set of corresponding countries is called $big_{countries}$), instead of 231, time series. We represent by TS_L1' this refined set of Level 1 time series. Then, we split each of these time series by cluster to produce the final set of time series $\Phi_{[0-800),c_i,country_j} \forall c_i$ and $\forall country_j \in big_{countries}$. The i^{th} point of the time series $\Phi_{[0-800),X,Y}$ indicates the amount of sources originating from country Y that have been observed on day i attacking any of our platforms thanks to the attack defined by means of the cluster X. We represent by TS_L2 the set of all these Level 2 time series. In this case $|TS\_L2|$ is equal to 436,756 which corresponds to 3,284,551 sources.

[1]The geographical location is given to us thanks to the Maxmind product, based on the IP address. However, some IPs can not be mapped to any real country and are attached to labels not corresponding to any country, e.g. EU,A1,..

As explained in [17], time series that barely vary in amplitude over the 800 days are meaningless to identify attack events and we can get rid of them. Therefore, we only keep the time series that highlight important variations during the 800 days period. We represent by TS_L2' this refined set of Level 2 time series. In this case $|TS\_L2'|$ is equal to 2,420 which corresponds to 2,330,244 sources.

We have done the very same splitting and filtering by looking at the traces on a per platform basis instead of on a per country of origin basis. The corresponding results are given in Table 1.

TS consists of 3,477,976 sources		
OVP	country	platform
$\|TS\_L1\|$	231	40
$\|TS\_L1'\|$	85 (94,4% TS)	40 (100% TS)
$\|TS\_L2\|$	436,756	395,712
$\|TS\_L2'\|$	2,420	2,127
sources	2,330,244 (67% of TS)	2,538,922 (73% of TS)

Table 1: dataset description: TS: *all sources observed on the period under study*, OVP: *observation view point*, TS_L1: *set of time series at country/platform level*, TS_L1': *set of significant time series in* TS_L1, TS_L2 : *set of all cluster time series*, TS_L2' *set of strongly varying cluster time series*

3.2 Attack Event Detection

Having defined the time series we are interested in, we now want to find attack events, that is we now want to identify all time periods during which 2 or more of these observed cluster time series are correlated together.

To do this, in a first step, we fix the time period to a value of L days and we use a sliding window of size L to assess the correlation of all pairs of time series over such sliding window. Therefore, given N time series of length T, we must compute the correlation of N time series for T-L+1 time interval $\{[1, L], [2, L + 1], \dots [T - L, T]\}$. As a result, we obtain the correlated time intervals for every pair of time series in N. A correlated time interval of two cluster time series is the interval in which two time series are correlated. After this first step, we group together all pairs of cluster time series that are correlated together over the same period of time. Each group of correlated observed cluster time series over a given period of time constitutes what we have defined as an *attack event*.

It is worth noting that this method, which we refer to as $M1$ in the sequel, can not detect attack events made of one observed cluster time series. This is typically the case for peaks of activities occurring on a single day. In such simpler cases, it is much more efficient to apply another, less expensive, algorithm to identify the attack events. This is what we have done. For the sake of conciseness, we have decided not to include the description of this second method, $M2$, in the paper as it lies outside the scope of the message we are interested to deliver.

In the first case, the techniques used to find strongly correlated time series are classical ones developed within the signal processing community. We refer the interested reader to our previous work [17] where we have covered them in some more detail and have positioned them with respect to the state of the art in this domain. It is worth stressing that, in this earlier publication, the methodology used was very different as well as the results presented. Indeed, in that first work, we have presented a cheap algorithm, based on heuristics, to validate the mere existence of attack events whereas in this work, we have carried out an expensive, brute force approach, to study and analyze the relationships between all attack events one could find in a much larger dataset.

3.3 Impact of Observation View Point

3.3.1 Results on Attack Event Detection

We have applied the attack events identification techniques to our 2 distinct datasets, namely $TS_{country}$ and $TS_{platform}$. For the time series in $TS_{country}$, the first method M1 (resp. second method M2), i.e. the general one, has found 549 (resp. 43) attack events. The total amount of sources found in these attack events is 552,492 for the first method and 21,633 for the second one. Thus, all in all, sources participating to identified attack events account for 574,125 sources (corresponding to 16,5% of all sources contained in our initial dataset). Similarly, when working with the time series found in $TS_{platform}$, we end up with a total of 690 attack events this time, containing 578,372 sources. The results are given in Table 2

Table 2: Result on Attack Event Detection

	AE-set-I($TS_{country}$)		AE-set-II($TS_{platform}$)	
	No.AEs	No.sources	No.AEs	No.sources
M1	549	552,492	564	550,305
M2	43	21,633	126	28,067
Total	592	574,125	690	578,372

No.AEs: amount of attack events
M1,M2: methods represented in Section 3.2

3.3.2 Analysis

The table highlights the fact that depending on how we decompose the initial set of traces of attacks (i.e the initial time series TS), namely by splitting it by countries of origin of the attackers or by platforms attacked, different attacks events show up. To assess the overlap between attack events detected from different observation view points we use the *common source ratio, namely csr,* measure as follows:

$$csr(e, AE_{op'}) = \frac{\sum_{\forall e' \in AE_{op'}} |e \cap e'|}{|e|}$$

in which $e \in AE_{op}$ and $|e|$ is the amount of sources in attack event e, AE_{op} is $AE_{country}$ and $AE_{op'}$ is $AE_{platforms}$ (or vice versa).

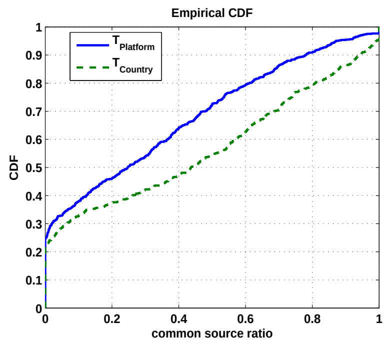


Figure 2: CDF common source ratio

Figure 2 represents the two cumulative distribution functions corresponding to this measure. The point (x, y) on the curve means that there are $y * 100\%$ of attack events obtained thanks to $T_{country}$ (resp $T_{platforms}$) that have less than $x * 100\%$ of sources in common with all attack events obtained thanks to $T_{platforms}$ (resp $T_{country}$). The $T_{country}$ curve represents the cumulative distribution obtained in this first case and the $T_{platforms}$ one represents the CDF obtained when starting from the attacks events obtained with the intial $T_{platforms}$ set of time series. As we can notice, around 23% (resp. 25%) of attack events obtained by starting from the $T_{country}$ (resp. $T_{platform}$) set of time series do not share any sources in common with any attack events obtained when starting the attack even identification process from the $T_{platform}$ (resp. $T_{country}$) set of time series. This corresponds to 136 (16,919 sources) and 171 (75,920 sources) attack events not being detected. In total, there are 288,825 (resp. 293,132) sources present in AE-Set-I (resp. AE-Set-II), but not in AE-Set-II (resp. AE-Set-I). As a final note, there are in total 867,248 sources involved in all the attack events detected from both datasets which correspond to 25% the attacks observed in the period under study. This number is coincidentally comparable with work in [20], in which, with a much more complicated technique, the authors claim that:*“[...] 27% of all malicious connection attempts observed from our distributed darknet can be directly attributed to botnet related spreading activity“*.

3.3.3 Explanation

There are good reasons that explain why we can not rely on a single viewpoint to detect all attacks events. They are described here below.

Split by country: Suppose we have one botnet B made of machines that are located within the set of countries $\{X, Y, Z\}$. Suppose that, from time to time, these machines attack our platforms leaving traces that are also assigned to a cluster C. Suppose also that this cluster C is a very *popular* one, that is, many other machines from all over the world continuously leave traces on our platforms that are assigned to this cluster. As a result, the activities specifically linked to the botnet B are lost in the noise of all other machines leaving traces belonging to C. This is certainly true for the cluster time series (as defined earlier) related to C and this can also be true for the time series obtained by splitting it by platform, $\Phi_{[0-800),C,platform_i} \forall platform_i \in 1..40$.However, by splitting the time series corresponding to cluster C by countries of origins of the sources, then it is quite likely that the time series $\Phi_{[0-800),C,country_i} \forall country_i \in \{X, Y, Z\}$ will be highly correlated during the periods in which the botnet present in these countries will be active against our platforms. This will lead to the identification of one or several attack events.

Split by platform: Similarly, suppose we have a botnet B' made of machines located all over the world. Suppose that, from time to time, these machines attack a specific set of platforms $\{X, Y, Z\}$ leaving traces that are assigned to a cluster C. Suppose also that this cluster C is a very *popular* one, that is, many other machines from all over the world continuously leave traces on all our platforms that are assigned to this cluster. As a result, the activities specifically linked to the botnet B' are lost in the noise of all other machines leaving traces belonging to C. This is certainly true for the cluster time series (as defined earlier) related to C and this can also be true for the time series obtained by splitting it by countries, $\Phi_{[0-800),C,country_i} \forall country_i \in big_{countries}$. However, by splitting the time series corresponding to cluster C by platforms attacked, then it is quite likely that the time series $\Phi_{[0-800),C,platform_i} \forall platform_i \in \{X, Y, Z\}$ will be highly correlated during the periods in which the botnet influences the traces left on the sole platforms concerned by its attack. This will lead to the identification of one or several attack events.

The top plot of Figure 3 represents the attack event 79. In this case, we see that the traces due to the cluster 175309 are highly correlated when we group them by platform attacked. In fact, there are 9 platforms involved in this case, accounting for a total of 870 sources. If we group the same set of traces by country of origin of the sources, we end up with the bottom curves of Figure 3 where the specific attack event identified previously can barely be seen. This highlights the existence of a botnet made of machines located all over the world that target a specific subset of the Internet.

4 On the armies of Zombies

So far, we have identified what we have called attack events which highlight the existence of coordinated attacks launched by a group of compromised machines,

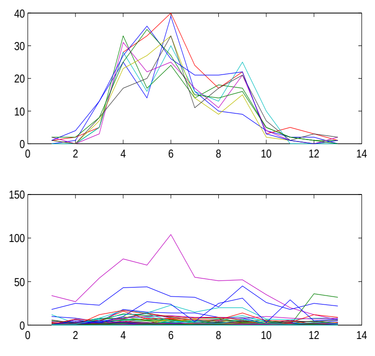


Figure 3: top plot represents the attack event 79 related to cluster 17309 on 9 platforms. The bottom plot represents the evolution of this cluster by country. Noise of the attacks to other platforms decrease significantly the correlation of observed cluster time series when split by country

i.e. a zombie army. It would be interesting to see if the very same army manifests itself in more than one attack event. To do this, we propose to compute what we call the *action sets*. An *action set* is a set of attack events that are likely due to same army. In this Section, we show how to build these action sets and what information we can derive from them regarding the size and the lifetime of the zombie armies.

4.1 Identification of the armies

4.1.1 Similarity Measures

In its simplest form, a zombie army is a classical botnet. It can also be made of several botnets, that is several groups of machines listening to distinct *C&C*. This is invisible to us and irrelevant. All that matters is that all the machines do act in a coordinated way. As time passes, it is reasonable to expect members of an army to be cured while others join. So, if the same army attacks our honeypots twice over distinct periods of time, one simple way to link the two attack events together is by noticing that they have a large amount of IP addresses in common. More formally, we measure the likelihood of two attacks events e_1 and e_2 to be linked to the same zombie army by means of their similarity defined as follows:

$$sim(e_1, e_2) = \begin{cases} max(\frac{|e_1 \cap e_2|}{|e_1|}, \frac{|e_1 \cap e_2|}{|e_2|}) & \text{if } |e_1 \cap e_2| < 200 \\ 1 & \text{otherwise} \end{cases}$$

We will say that e_1 and e_2 are caused by the same zombie army if and only if $sim(e_1, e_2) > \delta$. This only makes sens for *reasonable* values of δ. We address this issue in the coming subsections.

4.1.2 Action Sets

We now use the $sim()$ function to group together attack events into action sets. To do so, we build a simple graph where the nodes are the attack events. There is an arc between two nodes e_1 and e_2 if and only if $sim(e_1, e_2) > \delta$. All nodes that are connected by at least one path end up in the same action set. In other words, we have as many action sets as we have disconnected graphs made of at least two nodes; singleton sets are not counted as action sets.

We note that our approach is such that we can have an action set made of three attack events e_1, e_2 and e_3 where $sim(e_1, e_2) > \delta$ and $sim(e_2, e_3) > \delta$ but where $sim(e_1, e_3) < \delta$. This is consistent with our intuition that armies can evolve over time in such a way that the machines present in the army can, eventually, be very different from the ones found the first time we have seen the same army in action.

4.1.3 Results

To test the sensitivity of the threshold δ, we have computed the amount of action sets for the two datasets for different values of δ. The result is represented in top plot of Figure 4 (the bottom plot represent the corresponding amount of attack events involved in the armies). As we can see, at first, for the value of δ from 1% to 7%, the amount of action sets increases rapidly. Indeed, for very small values of δ all nodes remain connected together but, as δ increases, the initial graph loses arcs and more disconnected graphs appear, i.e. more action sets show up. This creation of action sets reaches a maximum after which action sets start disappearing with a growing δ value. This is due to the fact that some graphs are broken into isolated nodes that are not counting as attack sets anymore. The two curves reach their maximum values almost at the same position (when $\delta = 8\%$). Then they both start decreasing linearly.

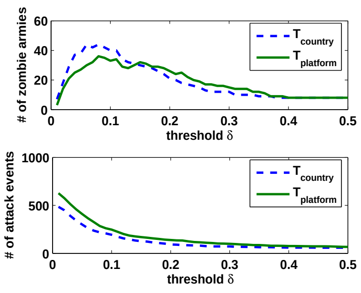


Figure 4: sensitivity check of threshold δ

In the context of this paper, we have arbitrarily chosen to investigate deeper the armies we can find when setting $\delta = 10\%$. We do not pretend that this number is optimal in any sense and, in fact, we do not really care. Indeed, our purpose, at this stage, is just to look at the results for one given value of δ and see if, yes or no, this theory of zombie armies seems to be valid or not, based on the characteristics of the ones we will find in that particular case. It can very well be that the attack events found in attack sets, as we have built them, have no underlying common cause and that they accidentally share common IPs.

For such value of δ we have identified 40 (resp. 33) zombie armies from AE-set-I (resp. AE-set-II) which have issued a total of 193 (resp. 247) attack events. Figure 5 represents the distribution of attack events per zombie army. Its top (resp. bottom) plot represents the distribution obtained from AE-set-I(resp. AE-set-II). We can see that the largest amount of attack events for an army is 53 (resp. 47) whereas 28 (resp. 20) armies have been observed only two times.

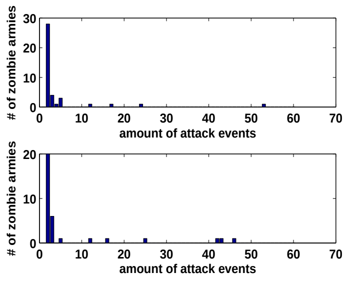


Figure 5: Zombie Army Size

4.2 Main Characteristics of the Zombie armies

In this section, we will analyze the main characteristic of the zombie armies.

Lifetime of Zombie Army Figure 6 represents the cumulative distribution of min-

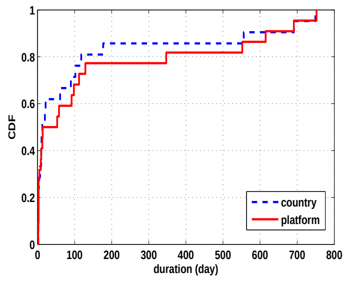
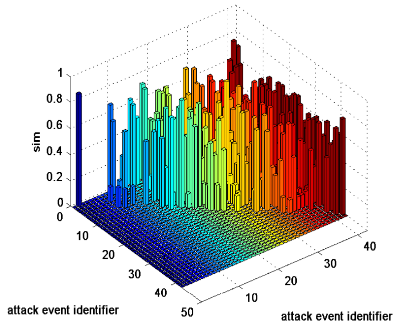


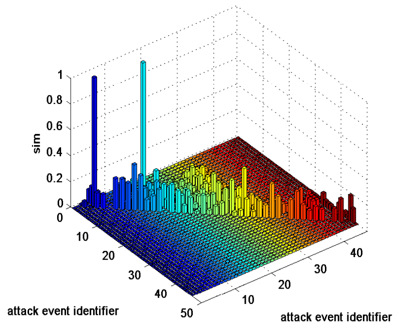
Figure 6: CDF duration

imum lifetime of zombie armies obtained from $TS_{platform}$ and $TS_{country}$ (see Section 4.1.3). According to the plot, around 20% of zombie armies have existed for more than 200 days. In the extreme case, two armies seems to have survived for 700 days! Such result seems to indicate that either i) it takes a long time to cure compromised machines or that ii) armies are able to stay active for long periods of time, despite the fact that some of their members disappear, by continuously compromising new ones.

Lifetime of Infected Host in Zombie Armies In fact, we can classify the armies into two classes as mentioned in the previous Section. For instance, Figure 7a represents the similarity matrix of zombie army 33, ZA33. To build this matrix, we first order its 42 attack events according to their occurred time. Then we represent their similarity relation under an 42×42 similarity matrix $\mathscr{M}$. The cell (i,j) represents the value of $sim()$ of the ordered attack event i^{th} and j^{th}. Since, $\mathscr{M}$ is a symmetric matrix, for the visibility, we represent only half of it. As we can see, we have a very high similarity measure between almost all the attacks events, around 60%. This is also true between the very first and the very last attack events. It is important to notice the time interval between the first and the last activities observed from this army is 753 days!



(a)



(b)

Figure 7: Renewal rate of zombie armies

Figure 7b represents an opposite case, the zombie army 31, ZA31, consisting of 46 attack events. We proceed as above to build its similarity matrix. As we can notice the important values are surrounded around the main diagonal of $\mathscr{M}$. It means that the attack event i^{th} has the same subset of infected machines with only few attack events happening not far from it in terms of time. Another important point to be noticed is that this army changes its attack vectors over time. In fact, it moves from attack against 4662 TCP, to 1025 TCP, then 5900 TCP, 1443 TCP, 2967 TCP, 445 TCP,...And the lifetime of this army is 563 days! It is clear, from these two cases, that the composition of armies evolves over time in different ways. More work remains to be done in order to understand the reasons behind these various strategies.

Attack Capacity By attack capacity, we refer to the amount of different attacks that a given army is observed lauching over time. The advanced worm, namely multi-headed worm, we have presented in our earlier work [17] is an example of worms that have many attack vectors and use them dynamically. The multi attack vectors allow the worms to have a large chance to propagate, and the varying in activity helps them to have multi attack traces which make it harder for IDS to detect them. This work reinforces the results we have earlier [17]. In fact, in previous work, we were able to detect multi-headed worms by the correlation of attack traces generated by different attack tools within an attack event. In this work, we have some even stronger evidence.Indeed, thanks to the notion of army, we observe several cases in which the same IP address has different behaviors in different attack events attached to a given army. As an example, the two attack events 128 and 131 consist of clusters 1378 and 2666 respectively. They both have 106 IP addresses in common and belong to the zombie army 12. All the attacks of attack event 128 are against port 64783 TCP whereas all the attacks of attack event 131 are against port 6211 TCP. The conclusion is that these 106 attacking machines mentioned earlier have dynamically changed their behavior. Finally, Figure 8 represents the distribution of number of distinct cluster per army. One zombie army has almost 120 clusters, yet not all of them are very different from each other.

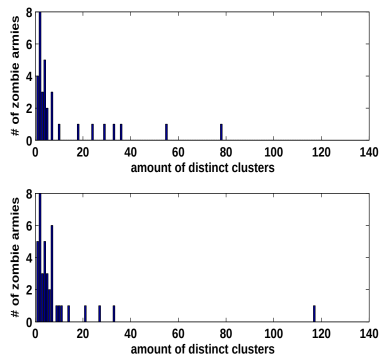


Figure 8: Zombie Army Attack Capacity

4.3 Illustrated Examples

After having offered a high level overview of the method and main characteristics of the results obtained, we feel it is important to give a couple of concrete, simple, examples of armies we have discovered. This should help the reader in better understanding the reality of two armies as well as what they look like. This is what we do in the next two subsections where we briefly present two representative armies.

4.3.1 Example 1

Zombie army 29, ZA-29, is an interesting example which has only been observed attacking a single platform. However, 16 distinct attack events are linked to that army! Figure 9a presents its two first activities corresponding to the two attack events 56 and 57. Figure 9b represents other four attack events. In each attack event, the army tries a number of distinct clusters such as 13882, 14635, 14647, 56608, 144028, 144044, 149357, 164877, 166477. These clusters try many combinations of Windows ports (135 TCP, 139 TCP, 445 TCP) and Web server (80 TCP). The time interval between the first and the last activities is 616 days !

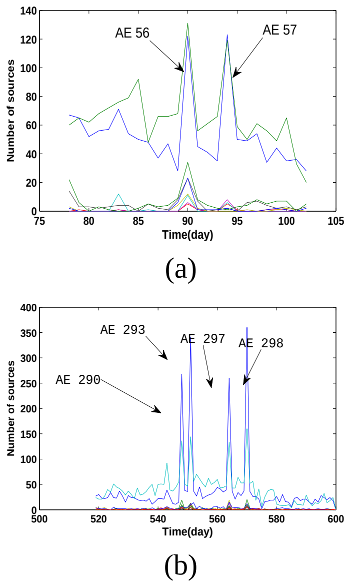


Figure 9: attack events of ZA29

4.3.2 Example 2

Last but not least, the zombie army 33, ZA-33, consisting of 42 attack events (already mentioned in Section 4.2) is an example of a multi-botnets zombie army. In fact, it seems that several botnets do different jobs and from time to time, they do some tasks together. In fact, in one hand, an important set of machines coming from Italy attacks several times one platform in China. As an example, the two top plots of Figure 10 are two examples of these attacks. The attack event 291 consisting of several clusters attack on port 64783T. And always coming from Italy, and targeting the same platform, but attack event 195 tries many clusters on port 9661 TCP. On the other hand, another component of ZA-33 consistently sends ICMP packets only, always coming from Greece and always targeting the same platform also located in Greece (see two plots in the middle of Figure 10). And as an example of coordination of two components of ZA33, the two plots in the bottom of Figure 10 represent two attack events (out of four) coming mostly from these two countries and attacking these two platforms. As a reminder, by design, there are always overlap between the attack events, for instance, attack event 483 share 41 IP address in common with AE 307, whereas 454 and 483 have 47 IP addresses in common.... The interval between the first and the last attack event issued by this zombie army is 753 days.

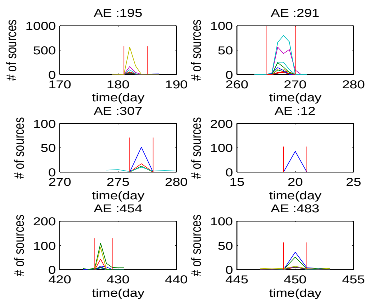


Figure 10: 6 attack events from zombie army 33

5 Conclusion

In this paper, we have addressed the important attack attribution problem. We have shown how low interaction honeypots can be used to track armies of zombies and characterize their lifetime and size. More precisely, this paper offers three main contributions. First of all, we propose a simple technique to identify, in a systematic and automated way, the so-called attack events in a very large dataset of traces. We have implemented and demonstrated experimentally the usefulness of this technique. Secondly, we have shown how, by grouping these attack events, we can identify long living armies of zombies. Here too, we have validated experimentally the soundness of the idea as well as the meaningfulness of the results it produces. Last but not least, we have shown the importance of the selection of the observation viewpoint when trying to group such traces for analysis purposes. Two such viewpoints have been considered in this paper, namely the geolocation of the attackers and the platform attacked. Results of the experiments have highlighted the benefits of considering more than one viewpoint as each of them offers unique insights into the attack processes. Future work needs to be done to consider other viewpoints as well as the possibility to combine these various viewpoints into a uniformed framework.

References

[1] M. Allman, E. Blanton, V. Paxson, and S. Shenker. Fighting coordinated attackers with cross-organizational information sharing. In *Hotnets 2006*, 2006.

[2] Paul Barford and Vinod Yegneswaran. An inside look at botnets. *Advances in Information Security*, 27:171–191, 2007.

[3] Ken Chiang and Levi Lloyd. A case study of the rustock rootkit and spam bot. In *First Workshop on Hot Topics in Understanding Botnets*, 2007.

[4] Evan Cooke, Farnam Jahanian, and Danny McPherson. The zombie roundup: understanding, detecting, and disrupting botnets. In *SRUTI'05: Proceedings of the Steps to Reducing Unwanted Traffic on the Internet on Steps to Reducing Unwanted Traffic on the Internet Workshop*, pages 6–6, Berkeley, CA, USA, 2005. USENIX Association.

[5] Neil Daswani and Michael Stoppelman. The anatomy of clickbot.a. In *HotBots'07: Proceedings of the First Workshop on Hot Topics in Understanding Botnets*, pages 11–11, Berkeley, CA, USA, 2007. USENIX Association.

[6] Felix C. Freiling, Thorsten Holz, and Georg Wicherski. Botnet tracking: Exploring a root-cause methodology to prevent distributed denial-of-service attacks. In *Lecture Notes in Computer Science*, pages 319–335. Springer-Verlag GmbH, September 2005.

[7] Jan Goebel and Thorsten Holz. Rishi: Identify bot contaminated hosts by irc nickname evaluation. In *Workshop on Hot Topics in Understanding Botnets 2007*, 2007.

[8] Julian B. Grizzard, Vikram Sharma, Chris Nunnery, Brent ByungHoon Kang, and David Dagon. Peer-to-peer botnets: overview and case study. In *HotBots'07: Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets*, pages 1–1, Berkeley, CA, USA, 2007. USENIX Association.

[9] Guofei Gu, Roberto Perdisci, Junjie Zhang, and Wenke Lee. Botminer: Clustering analysis of network traffic for protocol- and structure-independent botnet detection. In *USENIX Security '08*, 2008.

[10] Guofei Gu, Phillip Porras, Vinod Yegneswaran, Martin Fong, and Wenke Lee. Bothunter: Detecting malware infection through ids-driven dialog correlation. In *Proceedings of the 16th USENIX Security Symposium*, August 2007.

[11] Guofei Gu, Junjie Zhang, and Wenke Lee. Botsniffer: Detecting botnet command and control channels in network traffic. In *the 15th Annual Network and Distributed System Security Symposion*, 2008.

[12] Thorsten Holz, Christian Gorecki, Konrad Rieck, and Felix C. Freiling. Measuring and detecting fast-flux service networks. In *NDSS 2008*, 2008.

[13] Thorsten Holz, Moritz Steiner, Frederic Dahl, Ernst Biersack, and Felix Freiling. Measurements and mitigation of peer-to-peer-based botnets: a case study on storm worm. In *LEET'08: Proceedings of the 1st Usenix Workshop on Large-Scale Exploits and Emergent Threats*, pages 1–9, Berkeley, CA, USA, 2008. USENIX Association.

[14] Nicholas Ianelli and Aaron Hackworth. Botnets as a vehicle for online crime. In *18th Annual FIRST Conference*, May 2007.

[15] Corrado Leita, Van Hau Pham, Olivier Thonnard, Eduardo Ramirez Silva, Fabien Pouget, Engin Kirda, and Marc Dacier. The leurre.com project: collecting internet threats information using a worldwide distributed honeynet. In *1st WOMBAT workshop, April 21st-22nd, Amsterdam, The Netherlands*, Apr 2008.

[16] Emanuele Passerini, Roberto Paleari, Lorenzo Martignoni, and Danilo Bruschi. Fluxor: detecting and monitoring fast- flux service networks. In *DIMVA 2008*, 2008.

[17] Van-Hau Pham, Marc Dacier, Guillaume Urvoy Keller, and Taoufik En Najjary. The quest for multi-headed worms. In *DIMVA 2008, 5th Conference on Detection of Intrusions and Malware & Vulnerability Assessment, July 10-11th, 2008, Paris, France*, Jul 2008.

[18] Fabien Pouget and Marc Dacier. Honeypot-based forensics. In *AusCERT2004, AusCERT Asia Pacific Information technology Security Conference 2004, 23rd - 27th May 2004, Brisbane, Australia*, May 2004.

[19] Niels Provos. A virtual honeypot framework. In *Proceedings of the 12th USENIX Security Symposium*, pages 1–14, August 2004.

[20] Moheeb Rajab, Jay Zarfoss, Fabian Monrose, and Andreas Terzis. A multifaceted approach to understanding the botnet phenomenon. In *ACM SIGCOMM/USENIX Internet Measurement Conference*, October 2006.

[21] Anirudh Ramachandran, Nick Feamster, and David Dagon. Revealing botnet membership using dnsbl counter-intelligence. In *SRUTI'06: Proceedings of the 2nd conference on Steps to Reducing Unwanted Traffic on the Internet*, pages 8–8, Berkeley, CA, USA, 2006. USENIX Association.

[22] Günther Starnberger, Christopher Krügel, and Engin Kirda. Overbot - A botnet protocol based on Kademlia. In *SecureComm 2008, 4th International Conference on Security and Privacy in Communication Networks, September 22-25th 2008, Istanbul, Turkey*, Sep 2008.

[23] Joe Stewart. Bobax trojan analysis. http://www.secureworks.com/research/threats/bobax, May 2004.

[24] Joe Stewart. Phatbot trojan analysis. http://www.secureworks.com/research/threats/sinit/, March 2004.

[25] W. Timothy Strayer, Robert Walsh, Carl Livadas, and David Lapsley. Detecting botnets with tight command and control. *Local Computer Networks, Proceedings 2006 31st IEEE Conference on*, pages 195–202, Nov. 2006.

[26] Ping Wang, Sherri Sparks, and Cliff C. Zou. An advanced hybrid peer-to-peer botnet. In *HotBots'07: Proceedings of the first conference on First Workshop on Hot Topics in Understanding Botnets*, pages 2–2, Berkeley, CA, USA, 2007. USENIX Association.