Detect Me If You... Oh Wait. An Internet-Wide View of Self-Revealing Honeypots

Shun Morishita[1], Takuya Hoizumi[1], Wataru Ueno[1], Rui Tanabe[1],
Carlos Gañán[2], Michel J.G. van Eeten[2], Katsunari Yoshioka[1], and Tsutomu Matsumoto[1]

[1]Yokohama National University, Yokohama, Japan
[2]Delft University of Technology, Delft, Netherlands

***Abstract*—Open-source honeypots are a vital component in the protection of networks and the observation of trends in the threat landscape. Their open nature also enables adversaries to identify the characteristics of these honeypots in order to detect and avoid them. In this study, we investigate the prevalence of 14 open-source honeypots running more or less default configurations, making them easily detectable by attackers. We deploy 20 simple signatures and test them for false positives against servers for domains in the Alexa top 10,000, official FTP mirrors, mail servers in real operation, and real IoT devices running telnet. We find no matches, suggesting good accuracy. We then measure the Internet-wide prevalence of default open-source honeypots by matching the signatures with Censys scan data and our own scans. We discovered 19,208 honeypots across 637 Autonomous Systems that are trivially easy to identify. Concentrations are found in research networks, but also in enterprise, cloud and hosting networks. While some of these honeypots probably have no operational relevance, e.g., they are student projects, this explanation does not fit the wider population. One cluster of honeypots was confirmed to belong to a well-known security center and was in use for ongoing attack monitoring. Concentrations in an another cluster appear to be the result of government incentives. We contacted 11 honeypot operators and received response from 4 operators, suggesting the problem of lack of network hygiene. Finally, we find that some honeypots are actively abused by attackers for hosting malicious binaries. We notified the owners of the detected honeypots via their network operators and provided recommendations for customization to avoid simple signature-based detection. We also shared our results with the honeypot developers.**

I. Introduction

Honeypots have long been a valuable tool for network monitoring, helping to analyze and detect attacks against resources in the network. Beyond protecting a network, academia and the security industry have deployed larger collections of honeypots–i.e., honeynets–to observe attack trends across networks and the Internet as a whole. They expose what vulnerabilities are targeted, capture malicious binaries and enable observation of intruders.

When it comes to low-interaction honeypots, many operators rely on open-source software to emulate vulnerable network services. However, this use of emulation also implies a weaknesses, as the attacker might leverage discrepancies between the emulation and the actual service to detect and avoid these types of honeypots.

978-3-903176-15-7

Various tools and services for detecting honeypots have been released [43], [45], [50], [52]. These automated tools allow attackers to distinguish honeypots from real systems without having to dive into intricacies of how to leverage telltale signs for large-scale detection. The tools range from resource-intensive to lightweight and highly scalable. An example of the latter is the use of simple signatures that can be deployed via tools like Zmap [26], as most recently was done by Vetterl and Clayton [60]. Such lightweight approaches offer attackers a low-cost solution to detecting and avoiding honeypots. The question is, of course, how many honeypots are discoverable by simple signatures. In other words, how many honeypot developers make no effort to avoid discovery?

The objective of our study is to investigate the prevalence of honeypots that fail to take even the most basic precautions against detection by attackers. We present a simple signature-based detection method that looks for the characteristic responses–e.g., banners and web contents–of the default installations of open-source honeypots. (Though simple, we actually find a larger population of honeypots than the recent and more sophisticated approach of [60].) To put it differently, our main goal is to find out if, where, and why operators forsake even the most minimal effort to hide their honeypots. Rather than improving on existing honeypot detection techniques, we use 20 signatures for the characteristic responses of 14 open-source honeypot solutions. We first test these signatures for false positives and then run them against available scan data from Censys and against additional scan results that we collected ourselves. In sum, the contributions of this paper are:

- Using 20 signatures, we were able to discover 19,208 honeypots across 637 Autonomous Systems for which not even a minimal attempt was made to hide them. Concentrations are found in research networks, but also in enterprise, cloud and hosting networks. In terms of geography, we observe high prevalence of these honeypots in Taiwanese networks.
- We explore potential explanations for the prevalence of default honeypots by contacting their operators. Some of these are probably operationally irrelevant, but we also find honeypots deployed as part of professional security operations and project of a national research institute.
- We find that some of the honeypots pose security threats

themselves, as they are abused by attackers for hosting malicious binaries.
- We informed developers and operators of the easy discoverability of their honeypots and provided simple recommendations to prevent this.

The prevalence of easily discoverable honeypots is worrying. It seems prudent to assume that at least some attackers will leverage these evasion techniques or share their results in the form of blacklist. This means that all measurement projects and attack monitoring efforts based on these honeypots will be biased towards certain types of attacks and attackers – i.e., the less sophisticated ones. These biased observations potentially suck resources away from detecting and observing more skilled attackers. Last, but not least, we have also observed that poorly maintained honeypots can themselves become tools for the attackers. With our study, we aim to contribute to improving the security practices around honeypot deployment and safeguarding the validity of the measurement of attack trends.

The remainder of this paper is structured as follows. In Section II, we introduce a related work. In Section III, we outline a method to detect open-source honeypots. In Section IV, we describe our experiments. In Section V we discuss the experiment results. In Section VI, we describe ethical considerations and responsible disclosure. Finally, we conclude in Section VII.

II. Related Work

We use the term *honeypot* to refer a decoy system that are extensively leveraged to obtain threat information [3], [30], [31], [32], [47], capture malicious binaries [48], [62] and to detect previously-unseen attacks [46], [49]. Nawrocki *et al.* give a comprehensive overview of known honeypots [33].

Honeypot design: There is a wealth of work on honeypot design, roughly divided among on two types of designs: low-interaction and high-interaction honeypots. The latter provides a truly vulnerable system, thus allowing the attacker to interact with the application or service. The former type provides is less realistic, but it prevents hijacking and provides more control over what the attacker can do, as it emulate only a part of network services.

Many low-interaction honeypots are based on open-source. Table I provides an overview. The Honeynet Project is a popular organization that is dedicated to investigating the latest network attacks [57]. It has developed many open-source honeypots and voluntaries all over the world are deploying these honeypots. Similarly, there are many other services that share open-source honeypots. It is not surprising for operators to deploy these honeypots without changing their settings. For this reason, we focus on the discoverability open-source honeypots. Our findings provide input for design decisions regarding to their discoverability.

Honeypot detection: There is a rich literature on honeypots detection and evasion [5], [29]. Various studies have showed how to distinguish between regular servers and honeypots by detecting the virtual environment and debugger [27], by using the latency of the network links [61], by sending out unmodified malicious traffics [42], by collecting evidence of the machine [6] or by fingerprinting network data [10]. Over time, automated tools for detecting honeypots have been released. Honeypot Hunter [50] is a tool to validate proxy from honeypot. Nmap [40] is one of the most well-known scanner that has signatures to detect known honeypots. The famous vulnerability scan tool Metasploit [44] also has a module to detect Kippo [18]. There are also services that perform honeypot detection. Honeyscore [52] is a service that rates the likelihood of an IP address being connected to a honeypot by drawing on Shodan [53].

A related body of work is the extension of large-scale scanning tools and techniques such as ZMap [26], to detect honeypots. Various studies have showed how to detect honeypots by systematically generating fingerprints for 9 different honeypots and scanning the Internet [60], by finding publicly accessible Industrial Control Systems on the public IPv4 address [36]. Similar to these studies, we base our approach on Internet-wide scanning tools. We use Censys data [8], which logs the results of ongoing ZMap scans, and combine it with our own scans. Our study contributes to this literature; not by improving the detection capability as such, but by providing a simple lightweight approach that still manages to uncover a large population of honeypots – larger, in fact, than via the more novel approach of [60]. That being said, our main goal is not to advance detection, but rather to find out if, where, and why operators forsake even the most minimal effort to hide their honeypots.

Detection resistance: In light of improved detection techniques, researchers have developed more stealthy honeypots, in order to observe further attacks. Various studies have showed how to develop stealthy honeypots by revising a small part of the toolkit code of Honeyd [39] and appropriately patching the operating system to counter fingerprint attacks [61], by developing a system that redirect attacking service connections to the honeypot and redirect non-attacking service connections or probing connections to the production servers [51], by proposing a method to distinguish honeypots from real bots and provide a higher chance to join botnets [35], or by reacting to botnets with an intelligent deceptive response to improve the depth of deception [7].

Vise verse, researches have revealed how human actors are affected from the underlying environment by implementing honeypots with different properties [58], or by proposing a mathematical model of what would make a computer system to pretend as if a fake honeypot and scare away smarter attackers [38]. This demonstrates an ongoing arm race between honeypot operators and attackers. Indeed for sure, honeypot operators should take care of detection resistance.

To the best of our knowledge, very little research has been done for surveying the prevalence of honeypots across the Internet. Given the diversity of honeypots and their varying degrees of discoverability, this is a hard task. We aim to survey a specific subset of this population: open-source honeypots that

TABLE I: Open-source Honeypots and Listening Port (TCP).

Honeypot	Version (Installed date)	Listening Port (TCP)
Kippo [18]	(08/13/2017)	22*, 2222
Cowrie [13]	1.2.0	22*, 23*, 2222, 2223
telnetlogger [22]	0.2	23
MTPot [19]	(10/18/2017)	23
Telnet IoT honeypot [21]	(09/14/2017)	23*, 2222
Glastopf [15]	3.1.3-dev	80
Shockpot [20]	(10/18/2017)	80*, 8080
Wordpot [24]	(09/14/2017)	80
HoneyThing [17]	1.0.0	80, 7547
Conpot [12]	0.5.1-default-template	80, 102, 502
Nepenthes [4]	0.2.2	21, 25, 42, 80, 110, 135, 139, 143, 220, 443, 445, 465, 993, 995, 1023, 1025, 2103, 2105, 2107, 2745, 3127, 3140, 3372, 5000, 5554, 6129, 10000, 17300, 27347
Dionaea [14]	0.1.0	21, 42, 80, 135, 443, 445, 1433, 1723, 3306, 5060, 5061
Amun [11]	0.2.3-devel	21, 23, 25, 42, 80, 105, 110, 135, 139, 143, 443, 445, 554, 587, 617, 1023, 1025, 1080, 1111, 1581, 1900, 2101, 2103, 2954, 2967, 2968, 3127, 3128, 3268, 3372, 3389, 3628, 5000, 5168, 5554, 6070, 6101, 6129, 7144, 7547, 8080, 9999, 10203, 27347, 38292, 41523
HoneyPy [16]	0.6.3-linux-profile	7*, 8*, 21*, 22*, 23*, 25*, 53*, 80*, 88*, 110*, 111*, 139*, 143*, 389*, 443*, 636*, 873*, 2049, 3306, 5432, 6000, 10007, 10008, 10021, 10022, 10023, 10025, 10053, 10080, 10088, 10110, 10111, 10139, 10143, 10389, 10443, 10636, 10873

We note that port numbers with asterisk (*) indicate target port that honeypot operators additionally set up for further observation.

are running with the default configuration, as well as identify where concentrations of such honeypots occur. This raises new questions about the incentives of honeypot operators.

III. Signature-based Honeypot Detection

In this section, we outline our method to detect 14 open-source honeypots, listed in Table I. In order to perform Internet-wide scans for these honeypots, we focused on their characteristic responses that can be obtained by a single request packet so that the detection can be implemented with scalable scanners like ZMap [26].

We use 3 existing signatures implemented in Nmap for detecting 2 open-source honeypots, that is Nepenthes (FTP) and Dionaea (FTP, HTTP). For the remaining honeypots, we ran and scanned each of them in our local environment to find characteristic responses. We were able to create 8 signatures for 5 honeypots. For 7 honeypots, we could not find any characteristic response. We investigated their source code and were able to develop a further 9 signatures for these honeypots. In sum, we created 17 signatures for 12 open-source honeypots and by combining them with the 3 Nmap signatures for 2 open-source honeypots, we used 20 signatures in total for detecting 14 open-source honeypots. Table II summarize the signatures. We categorize our signatures into: banner, HTTP response, and error response. We show details of each signature in Table III.

Banner: Banner is the string that is returned first when connecting to a service. Services such as FTP, SSH, and Telnet can return a banner. Telnet service often sends negotiation option data before sending the actual banner string. For simplicity, we treat this option data as part of the banner. We find that the default banners of many open-source honeypots are unique enough to be used as signatures for their detection. We compared all FTP and Telnet banners of the honeypots with 694 FTP banners and 1,056 Telnet banners registered in *Nmap-service-probe* and found no match, indicating that these honeypot banners are indeed different from those common services that Nmap can identify. We note that it is technically

TABLE II: Signature Category of Open-source Honeypots.

Honeypot	Signature Category
Nepenthes (21/TCP, FTP)	*Banner
Dionaea (21/TCP, FTP)	*Banner
Amun (21/TCP, FTP)	Banner
Kippo (22/TCP, SSH)	Error response
Cowrie (22/TCP, SSH)	Error response
Cowrie (23/TCP, Telnet)	Banner
telnetlogger (23/TCP, Telnet)	Banner
MTPot (23/TCP, Telnet)	Banner
Telnet IoT honeypot (23/TCP, Telnet)	Banner
HoneyPy (23/TCP, Telnet)	Banner
Amun (25/TCP, SMTP)	Banner
Glastopf (80/TCP, HTTP)	HTTP response
Shockpot (80/TCP, HTTP)	HTTP response
Wordpot (80/TCP, HTTP)	HTTP response
HoneyThing (80/TCP, HTTP)	HTTP response
Conpot (80/TCP, HTTP)	HTTP response
Dionaea (80/TCP, HTTP)	*HTTP response
Amun (80/TCP, HTTP)	HTTP response
HoneyPy (80/TCP, HTTP)	HTTP response
Amun (143/TCP, IMAP)	Banner

We note that signatures with asterisk (*) are existing signatures registered in *Nmap-service-probe*.

easy to change the banners of the honeypots to avoid detection since they are indeed open source. Many of them even have a configuration file so that operators can customize the banner without changing the source code of the honeypot.

HTTP response: For those honeypots running HTTP services, the default HTTP response can be an easy signature for their detection. Indeed, we found that all 8 open-source honeypots that run HTTP service responded with some unique patterns if used in their default configuration (e.g., a hard-coded timestamp in HTTP header, unique HTML content, fixed response for any requests). Like banners, the HTTP response of these honeypots can be rather easily customized to avoid detection.

Error response: We found that intentionally erroneous requests were poorly handled by some of the honeypots and their responses are unique enough to be used as a signature. For example, an SSH negotiation with a non-existing SSH version would trigger a unique error message of some honeypots.

We note that our signature can be used for honeypot detection with sending just a single fixed request packet and capturing the corresponding response packet from the target host. This allows us to implement the detector with ZMap and thus Internet-wide scans are possible. Since SSH honeypots Kippo [18] and Cowrie [13] have similar error responses, we can distinguish these honeypots from each other with more interactions. We show details of the detection flow in figure 1.

IV. Experiments

In this section, we first evaluate the accuracy of the signatures to ensure a sufficiently low rate of false positives. We then match our signatures with scan data from Censys [8] and our own scans using ZMap [9], [26].

A. Evaluating Accuracy

As we derived our signatures from the default installations of the open-source honeypots and their source codes, it is confirmed that the signatures do match the characteristic responses of these honeypots. The key question regarding accuracy is, therefore, the rate of false positives, rather than false negatives. We evaluate the false positive rate by matching the signatures to four datasets of benign services that we assume to contain no honeypots.

Alexa Ranking: First, we used Alexa top 10,000 ranking [1] for our evaluation. It is highly unlikely that the high-ranked

TABLE III: Signatures of Open-source Honeypots.

Honeypot	Signature Category	Input Data	Response
Nepenthes (21/TCP, FTP) (Nmap)	Banner	NULL	220 ---freeFTPd 1\.0---warFTPd 1\.65---\r\n
Dionaea (21/TCP, FTP) (Nmap)	Banner	NULL	220 Welcome to the ftp service\r\n
Amun (21/TCP, FTP)	Banner	NULL	220 Welcome to my FTP Server\r\n
Cowrie (23/TCP, Telnet)	Banner	NULL	\xff\xfd\x1flogin:
telnetlogger (23/TCP, Telnet)	Banner	NULL	\xff\xfb\x03\xff\xfb\x01\xff\xfd\x1f\xff\xfd\x18\r\nlogin:
MTPot (23/TCP, Telnet)	Banner	NULL	\xff\xfb\x01\xff\xfb\x03\xff\xfc'\xff\xfe\x01\xff\xfd\x03\xff\xfe\"\xff\xfd'\xff \xfd\x18\xff\xfe\x1fUsername:
Telnet IoT honeypot (23/TCP, Telnet)	Banner	\r\n\r\n	\xff\xfd\x01Login: Password: \r\nWelcome to EmbyLinux 3\.13\.0-24-generic\r\n #
HoneyPy (23/TCP, Telnet)	Banner	NULL	Debian GNU/Linux 7\r\nLogin:
Amun (25/TCP, SMTP)	Banner	NULL	220 mail\.example\.com SMTP Mailserver\r\n
Glastopf (80/TCP, HTTP)	HTTP response	GET / HTTP/1.0\r\n\r\n	Blog Comments \n <label for=\"comment\">Please post your comments for the blog</label>\n \n <textarea name=\"comment\" id=\"comment\" rows=\"4\" columns=\"300\"></textarea>\n \n <input type=\"submit\" name=\"submit\" id=\"submit_comment\" value=\"Submit\" />\n
Shockpot (80/TCP, HTTP)	HTTP response	GET /nsE/2m9/hK9/fOy HTTP/1.0\r\n\r\n	<html><body> It Works! \n This is the default web page for this server\. \n The web server software is running but no content has been added, yet\. \n</body></html>\n
Wordpot (80/TCP, HTTP)	HTTP response	GET /wp-login.php?action =lostpassword HTTP/1.1\r\n\r\n	<input type=\"hidden\" name=\"testcookie\" value=\"1\" />\n\t \n</form>\n\n<p id=\"nav\">\nLost your password\?
HoneyThing (80/TCP, HTTP)	HTTP response	GET /Forms/login_security _1.html HTTP/1.0\r\n\r\n	<SCRIPT language=\"JavaScript\">\nif\(document\.Login_Form\.tipsFlag\.value == 1\)\nvar infoStr='Username or Password is incorrect, please try again\.';\ndocument\.getElementById\(\"tr1\"\)\.innerHTML = infoStr;\nelse if\(document\.Login_Form\.tipsFlag\.value == 2\)\ntimelast = document\.Login_Form\.timevalue\.value;\nwindow\.setInterval\(\"IncreaseSec\(\)\", 1000\);\n\n</SCRIPT>\n
Conpot (80/TCP, HTTP)	HTTP response	GET /index.html HTTP/1.0\r\n\r\n	Last-Modified: Tue, 19 May 1993 09:00:00 GMT
Dionaea (80/TCP, HTTP) (Nmap)	HTTP response	GET / HTTP/1.0\r\n\r\n	<!DOCTYPE html PUBLIC \"-//W3C//DTD HTML 3\.2 Final//EN\"><html>\n<title>Directory listing for /</title>\n<body>\n Directory listing for / \n
Amun (80/TCP, HTTP)	HTTP response	GET / HTTP/1.0\r\n\r\n	<!DOCTYPE HTML PUBLIC \"-//IETF//DTD HTML 2\.0//EN\"><html><head><title>It works!</title></head><html><body> It works! tim\.bohn@gmx\.net johan83@freenet\.de</body></html>\n\n
HoneyPy (80/TCP, HTTP)	HTTP response	GET / HTTP/1.0\r\n\r\n	Server: Apache/2\.4\.10 \(Debian\)\nConnection: close\nContent-Type: text/html\n\nOK!\n
Amun (143/TCP, IMAP)	Banner	\r\n\r\n	a001 OK LOGIN completed

We note that although some response patterns in the signatures seem legitimate, they indeed differ from the actual legitimate responses. For example, default banner for telnetd in Debian 7.0 is "`Debian GNU/Linux 7\r\ndebian login:`", which is different from those used by HoneyPy "`Debian GNU/Linux 7\r\nLogin:`".

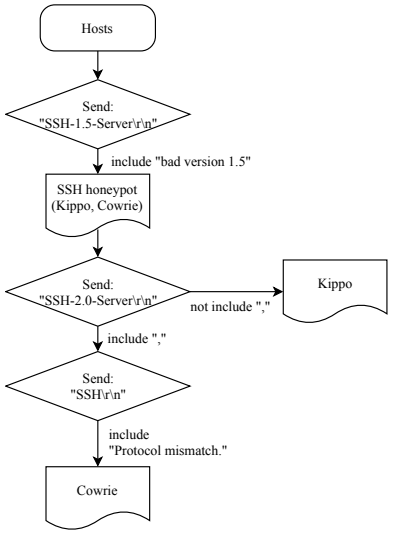

Fig. 1: Flow of SSH honeypot distinction.

domains are actually connected to a honeypot. We conducted a DNS lookup for the 10,000 domains, which results in 8,744 unique IP addresses. We then scanned these addresses and matched the results against our 20 signatures. We found open ports around services for which we have signatures. Since Alexa ranking is web site ranking, most of them were running on HTTP: 8,581. We also found 749 open ports for FTP, 1,128 for SSH, 53 for Telnet, 841 for SMTP, and 557 for IMAP. None of these results contained matches – in other words, we found no false positives.

Official FTP Mirrors: The second data source consisted of the official mirrors of Ubuntu [59], Apache [55], and CentOS [56] for evaluation of FTP honeypot signatures. We assume that these official mirrors contain no honeypots. We compiled a list of 531 unique domains running an FTP server: 299 domains for Ubuntu, 119 domains for Apache, and 258 domains for CentOS. The DNS lookup on these domains resulted in 457 IP addresses. Again, the results of our scans of these addresses contained no matches. Moreover, as described earlier, we confirmed that none of the honeypot FTP banners matched with 694 FTP banners registered in Nmap, showing honeypot banners are different from common FTP services that can be identified by Nmap.

University Email Domains: Third, we used university email domains for evaluation of the SMTP and IMAP honeypot signatures. We used the university domains of the GitHub repository [23] to make a list of domains and requested a DNS lookup to get their corresponding MX records. We collected 6,463 unique IP addresses where SMTP service was open and 1,683 unique IP addresses where IMAP service was open. We conducted a scan and confirmed that there were no false positives for any of these IP addresses.

IoT Device Telnet Services: Unlike other network services, Telnet service is often not meant to be provided for global use. Instead, it is running on many IoT devices like IP camera and routers [2]. We could only obtain six real IoT devices running Telnet services including four routers and two network storages. Our signatures did not create any false positives against the six devices. Moreover, we recall that our Telnet signatures did not match with any of the existing 1,056 banners registered in Nmap, showing their uniqueness.

B. Investigating Prevalence of Honeypots

After establishing that our approach has a sufficiently low false positive rate, we matched the signatures against Internet-

TABLE IV: Honeypot Detection Result using Censys and Our Own Scan data.

Honeypot signature	# Honeypots (Censys)	# Honeypots (Censys+Scan)	# Honeypots (Censys+Scan)	Survival Ratio	# Honeypots (Bitter Harvest)
Nepenthes (21/TCP, FTP) (Nmap)	**124**	-	119	96.0%	-
Dionaea (21/TCP, FTP) (Nmap)	**1,751**	-	1,324	75.6%	-
Amun (21/TCP, FTP)	**1,720**	-	745	43.3%	-
Cowrie (23/TCP, Telnet)	**1,796**	-	570	31.7%	938
telnetlogger (23/TCP, Telnet)	**2,984**	-	32	1.1%	-
MTPot (23/TCP, Telnet)	**226**	-	98	43.4%	216
HoneyPy (23/TCP, Telnet)	**2**	-	1	50.0%	-
Amun (25/TCP, SMTP)	**1,608**	-	430	26.7%	-
Glastopf (80/TCP, HTTP)	**2,487**	-	1,154	46.4%	3,371
Conpot (80/TCP, HTTP)	**89**	-	51	57.3%	87
Dionaea (80/TCP, HTTP) (Nmap)	**2,220**	-	569	25.6%	202
Amun (80/TCP, HTTP)	**944**	-	544	57.6%	-
HoneyPy (80/TCP, HTTP)	**19**	-	10	52.6%	-
Amun (143/TCP, IMAP)	**1,728**	-	686	39.7%	-
Kippo (22/TCP, SSH)	-	**505**	-	-	758
Cowrie (22/TCP, SSH)	-	**998**	-	-	2,021
Telnet IoT honeypot (23/TCP, Telnet)	-	**0**	-	-	11
Shockpot (80/TCP, HTTP)	-	**1**	-	-	-
Wordpot (80/TCP, HTTP)	-	**6**	-	-	-
HoneyThing (80/TCP, HTTP)	-	**0**	-	-	-
telnet-password-honeypot (23/TCP, Telnet)	-	-	-	-	1
Total	**17,698**	**1,510**	6,333	35.8%	7,605

wide scan data to measure the prevalence of default installations of open-source honeypots. While we could have scanned the whole Internet by ourselves using ZMap, we decided to take advantage of Censys data. The data contains periodic Internet-wide scan results using ZMap and ZGrab [25] and that we can reduce additional scans for the investigation.

The Censys data we used for the investigation was from Apr 9, 2018 to Apr 15, 2018. The Censys data contained 16.2M hosts with 21/tcp (FTP) open, 8.4M hosts with 23/tcp (Telnet) open, 13.6M hosts with 25/tcp (SMTP) open, 64.6M hosts with 80/tcp (HTTP), and 9.1M hosts with 143/tcp (IMAP) open, respectively. Among our 20 signatures, 14 of them could be directly matched with Censys data without additional scans (i.e. Censys data contained all necessary requests and responses to test these signatures). For the remaining 6 signatures, which could not be directly matched with Censys data, we first created more generic signatures to extract the candidate hosts from Censys data and then scanned the candidate hosts by ourselves using ZMap with the 6 signatures. Because the time when we performed our own scan (May 19, 2018) was about one month after Censys scan, we might underestimate the popularity of the honeypots due to their IP address churn. In order to measure the change over time, we also scanned the detected honeypots by the 14 signatures one month after the last Censys scan.

Table IV summarizes the detection results. The first column is our honeypot signature. The second column is the number of honeypots detected from Censys data. The third column is the number of honeypots detected by our own scans on the candidates extracted from Censys data. The fourth column is the number of honeypots detected by our own scan performed one month after the Censys scan. The fifth column is the surviving ratio of honeypots within the one month. The sixth column is the detection result of related work [60]. In summary, we detected 17,698 honeypots from Censys data by the 14 signatures. The most popular ones were telnetlogger with 2,984 hosts, Glastoph with 2,487 hosts, and Dionaea with 2,220 hosts. From the combination of Censys and our own scans with the 6 remaining signatures, we detected 1,510 honeypots with 2 popular solutions Cowrie with 998 hosts and Kippo with 505 hosts. In total, we detected 19,208 honeypots from 14 open-source honeypots. Compared to the related work [60] that detected 7,605 honeypots from 9 open-source honeypots, our result shows that our lightweight approach has the same or even better ability to detect honeypots.

TABLE V: Frequently observed FTP banner on the Internet.

Rank	IP	Ratio	Banner
1	5,436,627	33.5%	-
2	800,771	4.93%	`220 Microsoft FTP Service\r\n`
3	518,558	3.19%	`220 FTP Server ready.\r\n`
4	343,316	2.11%	`220 (vsFTPd 2.2.2)\r\n`
5	216,232	1.33%	`220 Ftp firmware update utility`
6	205,967	1.27%	`220 (vsFTPd 3.0.2)\r\n`
7	141,608	0.872%	`220 FTP service ready.\r\n`
8	119,395	0.736%	`220 Serv-U FTP Server v6.4 for WinSock ready... \r\n`
9	118,699	0.731%	`220-Microsoft FTP Service\r\n`
10	85,602	0.527%	`220 (vsFTPd 3.0.3)\r\n`
⋮	⋮	⋮	⋮
228	1,751	0.0108%	**Dionaea banner**
232	1,720	0.0106%	**Amun banner**
3,358	124	0.0008%	**Nepenthes banner**
Total	16,232,733	100%	

TABLE VI: Frequently observed Telnet banner on the Internet.

Rank	IP	Ratio	Banner
1	4,651,675	28.7%	-
2	176,908	2.09%	`{"banner":"\r\n\r\nUser Access Verification\r\n \r\nUsername: ","will":[{"name":"Echo","value" :1},{"name":"Suppress Go Ahead","value":3}],"do" :[{"name":"Terminal Type","value":24},{"name": "Negotiate About Window Size","value":31}]}`
3	120,561	1.43%	`{"banner":"\r\n%connection closed by remote host !\u0000"}`
4	102,623	1.21%	`{"banner":"\r\r\n(none) login: ","will":[{"name" :"Echo","value":1},{"name":"Suppress Go Ahead", "value":3}],"do":[{"name":"Echo","value":1},{ "name":"Negotiate About Window Size","value" :31}]}`
5	100,768	1.19%	`{"banner":"\r\n\r\nUser Access Verification\r\n \r\nPassword: ","will":[{"name":"Echo","value" :1},{"name":"Suppress Go Ahead","value":3}],"do" :[{"name":"Terminal Type","value":24},{"name": "Negotiate About Window Size","value":31}]}`
6	86,023	1.02%	`{"banner":"\r\nWelcome Visiting Huawei Home Gateway\r\nCopyright by Huawei Technologies Co. , Ltd.\r\n\r\nLogin:","will":[{"name":"Echo", "value":1},{"name":"Suppress Go Ahead","value" :3},{"name":"Terminal Type","value":24}]}`
7	64,764	0.767%	`{"banner":"\r\r\n(none) login: ","will":[{"name" :"Echo","value":1},{"name":"Suppress Go Ahead", "value":3}],"do":[{"name":"Echo","value":1},{ "name":"Negotiate About Window Size","value":31} ,{"name":"Remote Flow Control","value":33}]}`
8	53,498	0.633%	`{"banner":"\n\r\n\r\rAccount:","will":[{"name": "Echo","value":1}],"do":[{"name":"Terminal Type" ,"value":24}]}`
9	41,949	0.497%	`{"banner":"\r\nlogin: ","will":[{"name":"Echo", "value":1},{"name":"Suppress Go Ahead","value" :3}]}`
10	39,792	0.245%	`{"banner":"\r\nAccount:","will":[{"name":"Echo", "value":1}],"do":[{"name":"Terminal Type","value" :24}]}`
⋮	⋮	⋮	⋮
189	2,984	0.0353%	**telnetlogger banner**
261	1,793	0.0212%	**Cowrie banner**
1,080	214	0.00253%	**MTPot banner**
52,984	2	0.0000237%	**HoneyPy banner**
Total	16,232,733	100%	

Our re-scanning after the Censys scan revealed that at least 6,333 (35.8%) of 17,698 honeypots were still in operation after one month. Considering the lost honeypots by IP address churn, this gives us an impression of how many of these honeypots remained alive for more than a month. For most honeypots, a significant fraction was still running at the later time. There is one notable exception: almost all of the nearly 3,000 telnetlogger honeypots have disappeared. All of these honeypots were operated in the IP address range of a French research institution.

We also compared the FTP and Telnet banners of the honeypots against the total population of banners in the Censys data. Tables V and VI summarize how infrequent the banners of honeypots occur in the population. This implies that they can be used as detection signatures. Instead, an easy solution to prevent detection would be to use the popular banners.

V. Discussion

Discovering honeypots in the wild is a daunting task as by design they are meant to be deceptive. While our approach proved that it is possible to easily map off-the-self honeypots that are running default configuration, it is not feasible to map at scale all the honeypots that actually exist that are *well configured*. For example, Among the 14 open-source honeypots which we investigated, Conpot developers mention

TABLE VII: AS, Country and Network Type of Honeypot Population.

19,208 Honeypots (unique 13,417 IPs)					
IP	**AS Country**	**AS Type**	**IP per /16**	**AS Country**	**AS Type**
2,697	France	*Academic*	390.0	Mexico	*Academic*
783	Mexico	*Academic*	253.1	Taiwan	*ISP*
771	United States	*Hosting*	249.0	Greece	*Academic*
696	Taiwan	*ISP*	203.4	Taiwan	*ISP*
653	Japan	*Academic*	129.6	Japan	*Academic*
606	Taiwan	*ISP*	79.0	Taiwan	*ISP*
510	Italy	*Academic*	77.2	France	*Academic*
464	Taiwan	*ISP*	61.7	Taiwan	*ISP*
450	United States	*Hosting*	56.9	Taiwan	*Academic*
436	Taiwan	*Academic*	45.9	Sweden	*ISP and Hosting*
314	France	*Hosting*	41.1	France	*Academic*
277	Taiwan	*ISP*	21.2	United States	*Hosting*
271	Taiwan	*ISP*	20.0	Taiwan	*Academic*
249	Greece	*Academic*	19.8	Sweden	*Hosting*
247	France	*Academic*	18.8	Taiwan	*Academic*
187	Taiwan	*ISP*	18.4	Taiwan	*ISP*
171	United States	*Hosting*	15.9	Taiwan	*ISP*
154	Romania	*Other*	13.5	Taiwan	*ISP*
149	United States	*Hosting*	13.2	United States	*Hosting*
134	Taiwan	*ISP*	12.6	Taiwan	*ISP*

customization and explain how operators can tailor their honeypot (e.g. HTTP headers, response latency). We believe this is the main reason why we couldn't find many Conpot honeypots with default setting. We leave for future research detecting customized honeypots, and focus on understanding the prevalence of honeypots with default setting.

We now investigate the population of honeypots discovered via the scan data. We first turn to the locations of the main concentrations of honeypots can be found, both in terms of geography and Autonomous Systems (AS). We then conduct an investigation of their lifespan. Finally, we discuss feedback we received after reaching out to several of the network operators with significant concentrations of honeypots in their networks. We end with another angle of this problem: the active abuse of these honeypots by criminals.

A. Location

We used GeoIP2 ISP Database [34] to acquire AS as well as ipinfo.io [28] to acquire geographical information for the population of honeypots. Table VII summarize the number of IP addresses of the top 20 ASes, the country of the AS, and the type of the AS. We manually categorized AS types into four. Also, we examined the relative density of honeypots by calculating the ratio of the number of honeypots to the number of IP addresses owned by each AS.Since the number of IP addresses owned by each AS is different, we calculated and compared the number of honeypots per /16 (65,536 IP addresses). We summarize the number of honeypots per /16 of the top 20 ASes in Table VII, excluding ASes smaller than /16. Regarding AS types, we categorized universities and research institutions as "*Academic*", companies that provide Internet access as "*ISP*", companies that provide servers (e.g., shared or dedicated servers, VPSes, cloud services, etc.) as "*Hosting*", and other companies (e.g., a financial company, and a travel agency, etc.) as "*Other*". From the results, we found that some of the top ASes are universities and research institutions. In terms of geography, Taiwanese ASes are overrepresented. They occupy 8 spots among the top 20 ASes.

How can we explain this pattern? The concentrations inside research networks suggests that perhaps many of these honeypots are used for training purposes or student research projects. This fits with our observation that these honeypots appear to be running somewhat amateuristically, with their lack customization or even explicitly self-revealing settings. If these honeypots are mostly, say, student projects, then the impact of their discoverability is limited. They can still be abused by attacks, as we will see below, but at least they won't impact any professional operations to monitor networks or measure attack trends.

While this explanation undoubtedly accounts for a portion of the population we have found, it does not seem adequate to explain the overall pattern. First, a non-trivial amount of resources is being spent on these honeypots. As Table VII shows, certain networks allocate many hundreds of IP addresses to these honeypots. This scale seems to suggest a scientific measurement effort or industrial-grade attack monitoring, rather than a trial installation for a student project. Second, we also find concentrations in the commercial environment of hosting and ISP networks. Again, this allocation of resources suggests that these honeypots have operational function and value. Third, seeing these honeypots as non-operational installations cannot explain their high prevalence across different networks in Taiwan. To get a better sense of the factors that might explain these concentrations, we reached out to a number of network operators using public contacts.

B. Lifespan

To understand the lifespan of the honeypots, we analyze close to one year historical data of Censys data (Aug 8, 2017 to Jul 17, 2018). Figure 2 shows the number of detected honeypots in different ASes in one year. The number of detected honeypots is mostly stedy except that there are several events of bulk initiation and termination of honeypots in particular ASes. For example, in Nov 2017, 4,383 honeypots started operating in AS1, which is an academic network in Ireland, but in next month, all of them were shut down. Another huge termination of 6,917 honeypots was detected in Dec 2017 in AS2, which is an ISP in US.

Figure 3 shows the survival ratio of honeypots we detected on Aug 8, 2017. Counting from this first observation, the honeypots had an average lifespan of 200.42 days (SD=127.54). To see if lifespan varied across different networks types, we selected 53 ASes that had more than 10 detected IP addresses and manually categorized them into *Academic*, *ISP*, and *Hosting*. As a result, we have 5,553, 11,390, and 1,455 honypots detected on Aug 8, 2017 in Academic, ISP, Hosting networks, respectively. Remarkably, 80% of honeypots in academic networks were still operating after one year. These honeypots are obviously prime candidates for being detected and blacklisted by attackers. The honeypots in ISP and hosting networks show shorter lifespans, perhaps reflecting the higher economic value of the assigned resources. Still, about 20% of them run for at least one year in such a self-revealing manner. Note that the huge drop in ISP networks in Dec 2017 is due to the bulk termination of the honeypots in a U.S ISP. The drop in hosting networks in Apr 2018 comes from a bulk termination at a hosting service provider in U.S.

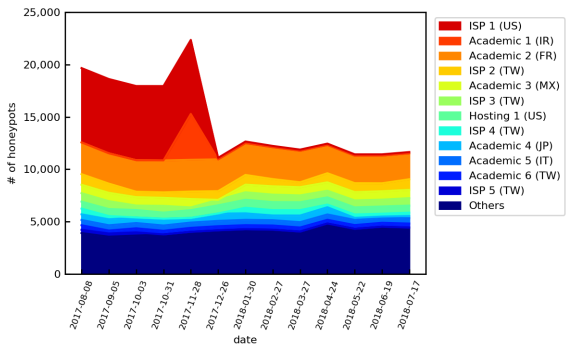

Fig. 2: # of Detected Honeypots in One Year.

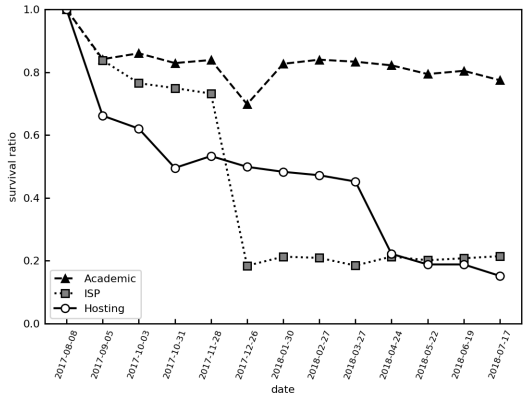

Fig. 3: Survival time of Honeypots Detected on August 8, 2017.

C. Operator Feedback

After we discovered many clusters of default honeypot installations, we were interested in understanding the reasons for running easily-discoverable honeypots. We contacted 11 operators of networks where these clusters were observed and received some insightful responses. We detailed the threat of honeypot detection (false negatives and other biases in monitoring and analysis) and offered suggestion on how to reduce discoverability. We asked the recipients, first, for confirmation of whether the IP addresses indeed hosted honeypots and, second, why these honeypots were running the default configurations, since that would make the easily detectable. However, we didn't receive any response from 7 operators. The lack of response already signals lack of network hygiene. So it is not surprising that these operators don't mind that their honeypots can be easily discovered. Other than the above, we received insightful responses from 4 operators.

One business ISP confirmed that there were indeed honeypots running on the IP addresses that we provided. As a courtesy, the ISP had temporarily routed some of its unused addresses space to a threat monitoring center of a world-famous security organization. The ISP put us in touch with the Chief Technical Officer of that center. When asked why they were running honeypots with default configurations, the CTO responded: 'Well, the quick answer is that detection by attackers is not a big issue.'

The operator of a national research network confirmed we did indeed correctly identified several clusters of honeypots in their network. One cluster was being operated by a researcher who was monitoring attacks on building automation. Another cluster was being operated by a security non-profit for threat monitoring 'to actively make the Internet more secure'. Other clusters were in address space delegated to universities.

The operator of a national research institute confirmed that we correctly identified honeypots in their network. Although they were not aware of honeypot detection issues, they explained that they are operating other type of honeypots (such as high-interaction ones) to observe more sophisticated attacks.

A researcher in a university in Taiwan, where we found a number of honeypots, explained that there has been a series of government supported cyber security projects to encourage the honeypots setups in different organizations and that it may be the reason for the disproportionately large concentration in this specific region.

D. Abuse

Prior work by Springall *et al.* reported that anonymous FTP servers were being abused by attackers to upload and distribute malware [54]. We also discovered 21 FTP honeypots that seems to be abused by an attacker. These honeypots had malicious binaries uploaded to them and other hosts were able to download them. We used VirusTotal to classify these binaries. We collected 54 malicious binaries (unique 17 malicious binaries): 12 CoinMiner, 2 Downloader, 2 Backdoor, and 1 Ransomware. This setup fits with a scenario where the hosts are used as download servers for when an attacker has established a foothold inside a network and then wants to download malware to the compromised host. We also note that these 21 abused honeypots were uploaded with similarly named files, so there is a possibility that the same attacker abused them. In short, it is clear that the poorly configured population of honeypots we uncovered can pose a direct security threat to the rest of the Internet.

VI. Ethics and Responsible Disclosure

Strictly speaking, our study could be considered a form of offensive research, in the sense that we reveal the presence of honeypots, which adversaries may use to evade monitoring. Note, however, that this approach is already available to anyone who is willing to look at the default installations and design simple signatures from their characteristics. We therefore consider the information in this paper to be public knowledge. However, for ethical reasons we remove all AS names and IP addresses when referring individual honeypots.

As part of the responsible disclosure process, we have contacted 6 honeypot developers who has a contact address listed on the Github page (Conpot, Glastopf, Cowrie, MTPot, Shockpot, and telnetlogger). We sent them an executive summary of our results and a full description of our methodology. We received responses from two developers associated with Conpot and Glastopf, one of whom works on both: Lukas

Rist. In light of our disclosure, he added a paragraph on customization to the Glastopf repository. For Conpot, he pointed out that there is an extensive section in the documentation on customizing the exposed content [37]. He also stated that while discoverability is inherent to low-interaction honeypots, it is correct to point out that a honeypot in its default configuration is low hanging fruit for an adversary. Hence, they usual recommend to customize honeypots when deploying them. Also, the developer suggested that an operator could handle this limitation by identifying when an adversary attempts to make the distinction between real and emulated.

We also contacted all network operators who had honeypots that seems to be abused by attackers. We used three different ways to contact each operator: an email address listed on their web sites, a contact form on their web site, or the email addresses listed in WHOIS. We sent messages detailing potential threats that can be posed by their honeypots and offered suggestions on how to reduce discoverability.

Related works have already provided how to build well configured honeypots [5]. We further explain recommendations for honeypot operators who run open-source honeypots.

1) Regarding open-source honeypots running FTP, Telnet, SMTP, and IMAP services: their default banners are different from those of common services. To avoid honeypot detection, it is effective to change their banner to a popular banner that is actually used in practice. The top 10 banners of FTP and Telnet are listed in Tables V and VI. Of course, it is possible to investigate other services from Censys data as well. Most honeypots can change their banner from the configuration file.
2) In case of HTTP service: their default Web contents are unique and that can be used for honeypot detection (e.g., a hard-coded timestamp in HTTP header, unique HTML content, fixed response for any requests). Most honeypots store HTML files in a prescribed directory and yet it is possible to change their Web contents.
3) For SSH service: their intentional erroneous requests are poorly handled. It is necessary to patch the source code to deal with this issue and that it may be more difficult than others cases.

VII. Conclusion

Our study has found that there are over 19,000 open-source honeypots online that can be detected with the simplest of signatures, as these systems have received no customization from their operators and are running in a basically self-revealing state. We found concentrations in research networks, but also in commercial hosting and access networks. In geographical terms, we found a high concentration in Taiwan.

The prevalence of these honeypots raise a number of questions. Why would anyone run a honeypot that is so easy to detect? One answer is that these honeypots serve no real operational purpose for ongoing security efforts. This explanation does not fit with the wider pattern, however. Thousands of honeypots are found in commercial networks, meaning that real resources are being spent on them. Furthermore, we confirmed that several clusters of honeypots were serving real operational purposes for threat monitoring, either as part of security operations or as part of academic research.

How problematic is this large-scale use of easily-discoverable honeypots? Operators should, of course, make their own tradeoffs in terms of how they set up their honeypots. We do want to point out, however, that the practice we uncovered presents serious problems to the field. First, for scientific research on the threat landscape, using such honeypots are likely to be biased towards the least sophisticated attacks and attackers. Nobody knows to what extent attackers care about evading honeypots. We know anecdotally that attackers share lists of honeypots. For example, we found a list of SSH honeypots on *Pastebin* [41]. In light of the uncertainty about the extent of avoidance by attackers, good scientific practice requires that researchers should assume that this is a source of bias in their results.

Second, and related, honeypots in operational security monitoring are also impacted by this bias – a bias, we might add, that will draw scarce attention towards attacks of low sophistication. Some professionals might argue, as one of our respondents did, that honeypot detection by attackers is not a big issue. It is unclear, however, what this assessment is based on and how much confidence we can award to it. We know of no systematic comparison of such honeypots. Perhaps it is because they still see plenty of traffic coming in to the honeypot. That does not tell them, however, what they are missing, which is likely to have higher value for their security operations that the traffic they do see. A third and final problem is that these honeypots themselves pose a security threat, as we found evidence of ongoing abuse by attackers.

In the end, the situation we have observed is the result of an incentive issue. Are honeypot operators incentivized to customize their installation, as the documentation tells them to do? Given that in thousands of cases the answer turns out to be negative, we see three paths forward to improve the situation. First, the installation procedures could incentivize more customization and perhaps even include some forms by default. Second, in the absence of customization, at least the default configuration could be using more generic characteristics – e.g., Telnet or FTP banners – that are widely shared by real hosts (see Section IV-B). Third, we should improve our understanding of the negative impacts of easily-discoverable honeypots – that is, to see when and where attackers try to detect and evade and how this influences our observations. This require rigorous protocols to measure the degree of bias that is introduced both in scientific results and in operational security monitoring and incident response.

Acknowledgements

A part of this work was funded by the WarpDrive: Web-based Attack Response with Practical and Deployable Research Initiative project, supported by the National Institute of Information and Communications Technology.

REFERENCES

[1] ALEXA. Alexa. https://www.alexa.com/.

[2] ANTONAKAKIS, M., APRIL, T., BAILEY, M., BERNHARD, M., BURSZTEIN, E., COCHRAN, J., DURUMERIC, Z., HALDERMAN, J. A., INVERNIZZI, L., KALLITSIS, M., ET AL. Understanding the mirai botnet. In *USENIX Security Symposium* (2017).

[3] ARIEL, B., BRACHA, S., LIOR, R., AND MOSHE, U. Identifying Attack Propagation Patterns in Honeypots using Markov Chains Modeling and Complex Networks Analysis. In *IEEE International Conference on Software Science, Technology and Engineering*, SWSTE'16.

[4] BAECHER, P., KOETTER, M., HOLZ, T., DORNSEIF, M., AND FREILING, F. The nepenthes platform: An efficient approach to collect malware. In *International Workshop on Recent Advances in Intrusion Detection* (2006), Springer, pp. 165–184.

[5] BLACKHAT. Breaking honeypots for fun and profit. https://www.blackhat.com/us-15/briefing.html#breaking-honeypots-for-fun-and-profit.

[6] CHARLES, C., SAM, C., BARBARA, E., P., AND DAVID, D. Dempster-shafer evidence combining for (anti)- honeypot technologies. In *International Conference on Cloud Security and Management*, ICCSM'15.

[7] CHARLES, C., SAM, C., BARBARA, E., P., AND DAVID, D. Hardening honeynets against honeypot-aware botnet attacks. In *International Conference on Cloud Security and Management*, ICCSM'15.

[8] DURUMERIC, Z., ADRIAN, D., MIRIAN, A., BAILEY, M., AND HALDERMAN, J. A. A Search Engine Backed by Internet-Wide Scanning. In *Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security*, CCS'15.

[9] DURUMERIC, Z., WUSTROW, E., AND HALDERMAN, J. A. ZMap: Fast Internet-wide Scanning and Its Security Applications. In *Proceedings of the 22nd USENIX Security Symposium*, USENIX'13.

[10] ELEAZAR, A., A., GINA, G., G., NICOLAS, S., L., AND LUIS A., V., V. A new procedure to detect low interaction honeypots. In *International Journal of Electrical and Computer Engineering* (2014), pp. 848–857.

[11] GITHUB. Amun. https://github.com/zeroq/amun.

[12] GITHUB. Conpot. https://github.com/mushorg/conpot.

[13] GITHUB. Cowrie. https://github.com/micheloosterhof/cowrie.

[14] GITHUB. Dionaea. https://github.com/rep/dionaea.

[15] GITHUB. Glastopf. https://github.com/mushorg/glastopf.

[16] GITHUB. HoneyPy. https://github.com/foospidy/HoneyPy.

[17] GITHUB. HoneyThing. https://github.com/omererdem/honeything.

[18] GITHUB. Kippo. https://github.com/desaster/kippo.

[19] GITHUB. MTPot. https://github.com/Cymmetria/MTPot.

[20] GITHUB. Shockpot. https://github.com/threatstream/shockpot.

[21] GITHUB. Telnet IoT honeypot. https://github.com/Phype/telnet-iot-honeypot.

[22] GITHUB. telnetlogger. https://github.com/robertdavidgraham/telnetlogger.

[23] GITHUB. University Domains and Names Data List & API. https://github.com/Hipo/university-domains-list.

[24] GITHUB. Wordpot. https://github.com/gbrindisi/wordpot.

[25] GITHUB. Zgrab. https://github.com/zmap/zgrab.

[26] GITHUB. Zmap. https://github.com/zmap/zmap.

[27] HOLZ, T., AND RAYNAL, F. Detecting honeypots and other suspicious environments. In *Proceedings of the 6th IEEE Information Assurance Workshop*, IAW'05.

[28] IPINFO.IO. ipinfo.io: IP Address API and Data Solutions - geolocation, company, carrier info, type and more. https://ipinfo.io/.

[29] JONI, U., SAMPSA, R., SAMUEL, L., AND VILLE, L. A survey on anti-honeypot and anti-introspection methods. In *World Conference on Information Systems and Technologies*, WorldCIST'17.

[30] KATERINA, G.-P., GOCE, A., ANA, D., RISTO, P., AND BRANDON, M. Characterization and classification of malicious Web traffic. In *Computers & Security* (2014), pp. 92–115.

[31] KOSTAS, G, A., STELIOS, S., PERIKLIS, A., KONSTANTINOS, X., EVANGELOS, P, M., AND ANGELOS, D. K. Detecting Targeted Attacks Using Shadow Honeypots. In *Proceedings of the 14th USENIX Security Symposium*, SSYM'05.

[32] LUKAS, K., JOHANNES, K., DAISUKE, M., TOMOMI, N., TAKASHI, K., KATSUNARI, Y., AND CHRISTIAN, R. AmpPot: Honeypot for Monitoring Amplification DDoS Attack. In *Proceedings of the 18th International Symposium on Research in Attacks, Intrusions and Defenses*, RAID'15.

[33] MARCIN, N., MATTHIAS, W., THOMAS, C., S., CHRISTIAN, K., AND JOCHEN, S. A Survey on Honeypot Software and Data Analysis. https://arxiv.org/pdf/1608.06249.pdf, 2016.

[34] MAXMIND. GeoIP2 ISP Database. https://www.maxmind.com/en/geoip2-isp-database.

[35] MEERAH, M., A.-H., AND MOSTAFA, H., D. Avoiding honeypot detection in peer-to-peer botnets. In *IEEE International Conference on Engineering and Technology*, ICETECH'15.

[36] MIRIAN, A., MA, Z., ADRIAN, D., TISCHER, M., CHUENCHUJIT, T., YARDLEY, T., BERTHIER, R., MASON, J., DURUMERIC, Z., HALDERMAN, ALEX, J., AND BAILEY, M. An Internet-wide view of ICS devices. In *14th Annual Conference on Privacy, Security and Trust*, PST'16.

[37] MUSHORG. Conpots documentation. https://conpot.readthedocs.io/en/latest/.

[38] NEIL, R., B., T., D., AND E., JOHN, C. Fake Honeypots: A Defensive Tactic for Cyberspace. In *Proceedings of the 7th IEEE Information Assurance Workshop*, IAW'06.

[39] NIELS, PROVOS. Honeyd Virtual Honeypot. http://www.honeyd.org/.

[40] NMAP.ORG. Nmap: the Network Mapper - Free Security Scanner. https://nmap.org/.

[41] PASTEBIN. ssh honeypot lists - Pastebin. https://pastebin.com/N2FUYg1x.

[42] PING, W., LEI, W., RYAN, C., AND CLIFF, C., Z. Honeypot detection in advanced botnet attacks. In *International Journal of Information and Computer Security* (2010), pp. 30–51.

[43] RAPID7. Kippo SSH Honeypot Detector. https://www.rapid7.com/db/modules/-auxiliary/scanner/ssh/detect_kippo.

[44] RAPID7. Metasploit: Penetration Testing Software, Pen Testing Security. https://www.metasploit.com/.

[45] RAPID7. Shodan Honeyscore Client. https://www.rapid7.com/db/modules/-auxiliary/gather/shodan_honeyscore.

[46] RESHMA, R., P., AND CHIRAG, S., T. Zero-Day Attack Signatures Detection Using Honeypot. In *Proceedings of the International Conference on Computer Communication and Networks*, COMNET'11.

[47] ROBERTO TANARA. Dionaea honeypot: from Conficker to WannaCry + SambaCry CVE 2017-7494. https://www.honeynet.org/node/1353.

[48] RYAN BARNETT. New Bot Malware (BoSSaBoTv2) Attacking Web Servers Discovered. https://www.trustwave.com/Resources/SpiderLabs-Blog/-Honeypot-Alert--New-Bot-Malware-(BoSSaBoTv2)-Attacking-Web-Servers-Discovered/.

[49] SAURABH, C., RAKESH, KUMAR, S., AND RAM, SWAROOP, M. Honeypot Baselining for Zero Day Attack Detection. In *International Journal of Information Security and Privacy* (2017), pp. 63–74.

[50] SEND-SAFE. Send-Safe Honeypot Hunter. http://www.send-safe.com/honeypot-hunter.html.

[51] SHIUE, L.-M., AND KAO, S.-J. Countermeasure for detection of honeypot deployment. In *International Conference on Computer and Communication Engineering*, ICCCE'08'.

[52] SHODAN. Honeyscore. https://honeyscore.shodan.io/.

[53] SHODAN. Shodan. https://www.shodan.io/.

[54] SPRINGALL, D., DURUMERIC, Z., AND HALDERMAN, J. A. Ftp: The forgotten cloud. In *International Conference on 46th Annual IEEE/IFIP Dependable Systems and Networks*, DSN'16.

[55] THE APACHE SOFTWARE FOUNDATION. the status of all apache mirrors. https://www.apache.org/mirrors/.

[56] THE CENTOS PROJECT. List of CentOS Mirrors. https://www.centos.org/download/mirrors/.

[57] THE HONEYNET PROJECT. About The Honeynet Project. https://www.honeynet.org/about.

[58] TIMOTHY, B., AND NICK, N. Picky attackers: Quantifying the role of system properties on intruder behavior. In *Proceedings of the 33rd Annual Computer Security Applications Conference*, ACSAC'17.

[59] UBUNTU. Official Archive Mirrors for Ubuntu. https://launchpad.net/ubuntu/+archivemirrors.

[60] VETTERL, A., AND CLAYTON, R. Bitter harvest: Systematically fingerprinting low- and medium-interaction honeypots at internet scale. In *12th USENIX Workshop on Offensive Technologies*, WOOT'18.

[61] XINWEN, F., WEI, Y., DAN, C., XUEJUN, T., KEVIN, S., AND STEVE, G. On recognizing virtual honeypots and countermeasures. In *Proceedings of the 2nd IEEE International Symposium on Dependable, Autonomic and Secure Computing*, DASC'06.

[62] YIN, MINN, P.-P., SHOGO, S., KATSUNARI, Y., TSUTOMU, M., TAKAHIRO, K., AND CHRISTIAN, R. IoTPOT: Analysing the Rise of IoT Compromises. In *Proceedings of the 9th USENIX Conference on Offensive Technologies*, WOOT'15.