A Honeyfarm Data Control Mechanism and Forensic Study

Wei Yin(✉), Hongjian Zhou, and Chunlei Yang

North China Institute of Computing Technology, Beijing, China
yinweihappy168@yahoo.com

Abstract. Honeyfarm is a model to deploy honeypots for global network attack monitoring, correlation and forensic analysis. Data control is a fundamental problem in the honeyfarm to protect the Internet from being attacked by compromised honeypots in the honeyfarm, while providing a controlled environment for worm behaviour study. However, this problem is not well addressed in a limited number of existing implementations. This paper presents a honeyfarm system and focuses on the design of a data control mechanism based on Intrusion detection and Data redirection (DOID). Comprehensive experiments including attack event tracing, worm behaviour study and forensic analysis display that DOID is a good tool for attack monitoring and forensic analysis.

Keywords: Honeyfarm · Data control · Forensic analysis

1 Introduction

Honeypot, representing as a vulnerable system, attracts hackers to probe, explore and attack. A single honeypot or multiple independently deployed honeypots can only provide a limited local view of network attacks, and global network attack monitoring, correlation and trend prediction is not available. Also, attack monitoring and analysis is non-trivial and maintenance of honeypots in various locations introduces high cost. This motivates the honeyfarm architecture, as shown in Fig. 1(a). It puts all honeypots into a resource pool located in one single area. A redirector is installed on each monitored production network, which redirects attack traffic to the corresponding honeypot in the resource pool. Therefore, only one security personnel, instead of one personal per location, is required at the central location to manage all honeypots.

However, there are only a limited number of honeyfarm prototypes in the literature. As far as we know, two most famous honeyfarm prototypes are Collapsar [1] and Potemkin [2]. Collapsar realises the traditional honeyfarm vision as well as the reverse honeyfarm vision. In the reverse honeyfarm, honeypots act as vulnerable clients, e.g. a web browser, exploited by malicious servers, e.g. a

The work is supported by the NSFC project 61702542 and the China Postdoctoral Science Foundation project 2016M603017.

B. Li et al. (Eds.): ChinaCom 2017, LNICST 237, pp. 362–372, 2018.
https://doi.org/10.1007/978-3-319-78139-6_37

web server. Potemkin aims to improve honeyfarm scalability by memory sharing of VM (Virtual Machine) honeypots on a guest operating system. On one hand, a honeyfarm contains thousands of honeypots so it should not be utilised to launch attacks against other hosts on the Internet after being compromised. Therefore, attack traffic should be at least blocked. Or even better, to give hackers a certain degree of freedom so that attack actions can be contained and monitored within the honeyfarm. This can lead to a better understanding of hackers' actions and worm propagation behaviours. On the other hand, hackers may download toolkits from the Internet to conduct subsequent actions or the compromised host may need to receive commands from a master on the Internet. Therefore, these behaviours cannot be blocked or contained. Consequently, hackers' behaviours in a honeyfarm need to be fine-grained controlled. However, in both realisations, the containment problem is not well addressed, or not even mentioned. Simply blocking all outbound connections may not work efficiently [3], because a hacker may download and install software on a compromised system. Blocking such non-attack connections may not be appropriate for studying the hacker's behaviours. Restricting outbound traffic rate and the number of outgoing connections [3] cannot stop all outgoing attacks and the risk for hackers to attack other hosts still exists. Therefore, a proper containment mechanism is essential for the honeyfarm architecture. In this paper, we propose a data control mechanism, in which the intrusion detection system (IDS) and the reverse firewall are introduced. The IDS is used to recognise attack traffic and redirects the attack traffic to an emulated target to study hacker's further behaviours. Non-attack traffic is not restricted, hence toolkits downloaded by hackers can be captured. Outgoing traffic will be further checked by the reverse firewall and DDoS attacks are filtered to avoid the liability issue.

The paper achieves the following contributions.

First, this paper is a real implementation of the honeyfarm concept. One advantage of our mechanism is that attack traffic is not simply blocked, but redirected to an emulated target in order to capture hacker's subsequent behaviours. Another advantage is that non-attack traffic can be recognised and forwarded to the Internet. Therefore attackers' non-attack activities, e.g. communicating with C&C (Command & Control) server and downloading toolkits, can be monitored as well.

Second, we deployed the honeyfarm on the Internet to monitor a number of production networks. A large number of attack statistics are recorded, gathered and analysed. It is proved that our implemented honeyfarm is an effective tool for attack event tracing [4], worm behaviour study [5] and forensic analysis [6].

The rest of the paper is organised as follows. Section 2 discusses the DOID data control mechanism. The experimental setup and results with the deployed DOID architecture is discussed in Sect. 3. The related work is presented in Sect. 4. The paper concludes in Sect. 5.

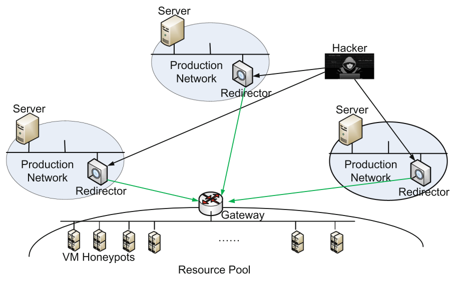
2 Data Control Mechanism

In this section, we present the DOID containment architecture and the defined policies for processing incoming and outcoming packets by the DOID gateway.

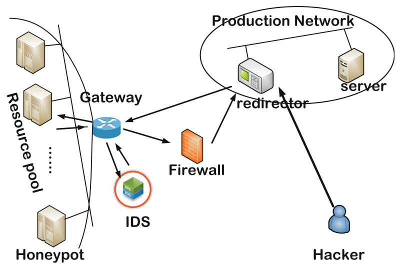
2.1 Containment Architecture

In the traditional honeyfarm architecture as shown in Fig. 1(a), redirectors in production networks transfer attack traffic to the honeyfarm gateway which redistributes the traffic to the corresponding honeypot in the resource pool. Responding packets are forwarded by the gateway to the redirectors and no other functionality is performed by the gateway in the traditional honeyfarm. When an attacker breaks a system, he may launch attacks to hosts on the Internet and initiate non-attack traffic to download toolkits. Therefore, outgoing traffic should be differentiated and controlled respectively to mitigate risks as well as providing freedom for the attacker to behave normally. In addition, worm behaviours should be contained and monitored in the honeyfarm so that their feature can be studied and understood. In our design as shown in Fig. 1(b), the DOID gateway has four components in order to achieve a good data control purpose: (1) Containment: implementing policies, e.g. dropping and forwarding, on incoming and outgoing traffic; (2) ARP responder: the gateway configured on the honeypot does not exist in the resource pool so that the DOID gateway should respond to the ARP request for the configured network gateway; (3) Monitoring: listening for configuration requests and making changes to DOID gateway configurations; (4) Virtual machine (VM) manager: managing VM honeypots.

Two external components are required to assist data control. One is intrusion detection system (IDS) that differentiates attack and non-attack traffic. The other is a reverse firewall, which functions as a firewall, but it implements policies on outbound traffic instead of inbound traffic so that it prevents the outside world from being attacked by honeypots in the honeyfarm. IDS is utilised for both inbound and outbound traffic checking. The inbound traffic will be checked by the IDS before forwarding to honeypots in the honeyfarm and the payload of known attacks can be modified to fail attacks so that hackers are encouraged to try various methods to penetrate the system, which enhances the security value of honeypots. The outbound traffic will be processed by multiple DOID gateway components and examined by IDS. Only non-attack traffic is forwarded to the Internet and attack traffic is redirected to the honeypots in the resource pool. After IDS, the non-attack traffic will be further checked by the reverse firewall for clearing DDoS attack.



(a) Traditional honeyfarm architecture



(b) DOID Honeyfarm Architecture

Fig. 1. Honeyfarm architectures

2.2 Containment Policy

The containment component implements strategies on incoming and outgoing traffic.

Inbound Traffic Strategies. Incoming traffic are forwarded by redirectors. They are packets from attack sources or responding packets of outgoing non-attack traffic. Since packets are encapsulated by the redirector and forwarded to the DOID gateway, they are decapsulated by the DOID gateway and the packet payload is checked by the IDS. The attack payload will be modified if the DOID gateway is configured to modify known attack payload. Otherwise, the packet is forwarded to the corresponding honeypot directly.

Packet Filtering Policy. The goal of the packet filtering policy is to prevent the honeyfarm from the Denial of Service (DoS) attack. The honeyfarm gateway generates a honeypot for each packet with a unique destination address. The redirector can be misused to deliver a large number of packets with different destination addresses to the honeyfarm gateway. Due to limited hardware resource in the honeyfarm, a bulk of requests cause exhaustion of hardware resource. To solve this issue, the honeyfarm gateway maintains a white list that lists all monitored IP addresses in production networks. When outgoing non-attack packets are forwarded, the destination addresses are recorded in another list named non-attack address list. All packets, of which the destination address does not belong to the white list or the source address does not belong to the non-attack address list, are filtered and no honeypot is generated. Consequently, the DoS attack is avoided under such a condition.

Packet Distribution Policy. For packets of which the destination address belongs to the white list or the source address belongs to the non-attack address list, they are distributed to the corresponding honeypot.

Packet Modification Policy. The value of a honeypot can be maximised by encouraging hackers to try zero-day exploits. This is done by enabling the DOID gateway to modify known attack payloads to fail an attack.

Outbound Traffic Strategies. Outbound packets include ARP requests, packets responding to the attack source, outgoing non-attack and attack packets initiated from honeypots.

Packet responding Policy. When the DOID gateway receives an ARP request, it uses its MAC address as the response to allow honeypots to send packets to it for forwarding.

Packet Encapsulation Policy. In order to monitor attackers' behaviours, including downloading toolkits and browsing websites, their outgoing non-attack traffic is forwarded instead of being blocked. Responding packets to the attack source are also forwarded. However, reverse firewall checking is done before forwarding to mitigate DDoS attacks from the Honeyfarm to the Internet. Finally these packets are encapsulated and forwarded to the redirector which decapsulates and forwards them to the Internet. They are not forwarded by the honeyfarm gateway to the attack source directly to avoid the inconsistency problem because the network gateway on the redirector's production network may use NAT.

Packet Redirection Policy. After compromised, the honeypot may be used as a drawboard to attack other hosts on the Internet. Therefore, the honeyfarm could be an incubator for malicious software and an accelerator for network worms. The IDS is utilised to recognise outgoing attacks and they are redirected to honeypots in the honeyfarm, preventing Internet hosts from being attacked from compromised honeypots in the honeyfarm. In this way, subsequent attack behaviours can be monitored as well.

Packet Drop Policy. Horizontal port scanning generates a bulk of packets targeted at different destinations. This causes overuse of honeypot resource. To solve this problem, we use a scan filter to limit the number of outbound scanning packets from a given honeypot in the honeyfarm. Also, we maintain a list of honeypots that are generated directly or indirectly by each attack source. If the number of generated honeypots exceeds a certain amount, it stops generating honeypots and subsequent packets will be dropped.

3 Experiments with DOID Honeyfarm

In this section, we discuss the experiment environment setup and the evaluation results including attack event tracing, worm behaviour study and forensic analysis. We also conduct performance evaluation of the system.

3.1 Experiment Environment

The resource pool is built in Beijing and a redirector is installed in 9 cities respectively. The traffic that is redirected from a particular city is forwarded to a certain honeypot in the honeyfarm, which has different types of system vulnerabilities. Table 1 shows the configuration of these honeypots. The monitoring starts from 1st Feb 2017 to 20th July 2017.

3.2 Aggregate Statistics

We capture the number of incoming attacks targeted at these honeypots in the resource pool, the number of outgoing attacks and non-attacks initiated from the resource pool. Due to the variety of running operating systems, open services and exposed vulnerabilities, the number of received attacks is not evenly distributed over these honeypots. We find that honeypots representing Shanghai, Guangzhou and Xi'an attract most attacks. Among 1101 attack attempts, 752 attempts are targeted at these three honeypots. This is because they expose more vulnerabilities than other honeypots. Most attacks are carried out between 9 am and 6 pm. This may indicate that most hackers are professionals and make

Table 1. Honeypot configuration for various cities

City	Honeypot configuration	
	Operating system	Running service
Shanghai (SH)	Ubuntu metasploitable 2 [7]	FTP, SSH, TELNET, SMTP, HTTP, RPC, NETBIOS-SSN, MICROSOFT-DS
Guangzhou (GZ)	Unpatched Win XP SP3	MSRPC, NETBIOS-SSN, MICROSOFT-DS
Xi'an (XA)	Unpatched Win server 2003	FTP, HTTP, RPC, NETBIOS-SSN
Zhengzhou (ZZ)	Patched Ubuntu metasploitable 2	SSH, FTP
Chengdu (CD)	Patched Win XP SP3	MSRPC
Nanjing (NJ)	Redhat 7.0 with apache web service (version 1.3 with mod-cgi feature). Bash not patched	HTTP
Wuhan (WH)	Win 7. Patched without fixing the MS15-067 vulnerability	RDP (Remote Desktop Protocol)
Changsha (CS)	Unpatched Win server 2008, running a web app with a SQL injection vulnerability	FTP, IIS, MySQL
Kunming (KM)	Patched Win 10	No service

a living on it. Using compromised honeypots as drawboards, hackers launch a number of attacks to the outside world. They also establish non-attack connections to the outside world to download files through FTP or use ICMP echo requests to probe hosts on the Internet. From the aggregate statistics, we find that the more vulnerabilities a honeypot has, the more attacks it attracts, and the more likely it is used as the drawboard to compromise the outside world. The captured outgoing non-attacks and attacks indicate that differentiating outgoing traffic types is essential for the data control architecture in the honeyfarm. On one hand, outgoing non-attack connections cannot be blocked so that hackers' toolkit download actions can be captured. On the other hand, outgoing attacks should be redirected to the emulated target in the honeyfarm so that the liability problem is avoided as well as hackers' subsequent behaviours can be monitored. This confirms that our design is appropriate for these scenarios.

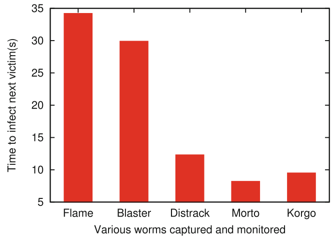
For 134 outgoing attacks, we also monitor the target geographic distribution. Most of the outgoing attacks are targeted at intra-network hosts. This means most of the time hackers use the compromised honeypot as the entry point to explore and penetrate hosts in the inner network. Therefore most hackers are interested in data residing in the network where the redirector is located. We also find that famous Japan companies including Sony and Nikon, and European multi-national corporations, e.g., Benz and Siemens, become the drawboard targets. This means the motivation of most hackers could probably be stealing commercial secrets from those companies and they sell commercial documents to their competitors for a living.

According to the statistic analysis, we totally captured 948 different attacking sources. They belong to 59 different countries. Over half of the source addresses are from China. Most of these addresses are from prefixes of 123.233/16, 27.224/16 and 27.115/16. Japan ranks the second, followed by the US and Europe. Although statistics show attacks are from these addresses, it does not mean attacks are actually from those areas, because technologies such as VPNs and the Tor technology [8] make the source tracing very difficult.

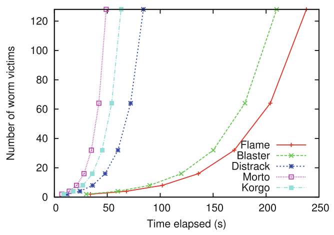
3.3 Worm Propagation Analysis

In the design of the DOID architecture, attack behaviours are contained and monitored so that it can be utilised to capture, contain and study worm propagation behaviours. Simply blocking attack traffic fails to give worms freedom to propagate. Allowing worms to propagate to the Internet causes the liability issue and worm behaviours cannot be monitored. Over five-month monitoring, our farm system captured a number of worms including Flame, Morto and Blaster. We studied the propagation speed of various worms. Figure 2(a) illustrates the time taken for each worm to infect the next victim after it first time infects a honeypot in the farm. We find that the infection speed for Morto is the fastest. It takes 7 s to infect the next honeypot. The speed for Flame and Blaster is comparatively slow, around 30 s. Morto propagates itself through weak password in the remote desktop protocol (RDP) and its goal is to gain remote desktop access authority. In our farm, the password for RDP is null. No wonder the Morto worm propagates so fast. Flame propagates itself through network shared files and its goal is to gather information through screen-shots, keystroke and network traffic record. Blaster propagates itself by an RPC (Remote Procedure Call) vulnerability. Therefore, vulnerability scanning and exploitation takes longer time.

In our data control mechanism for the farm, we configured that an attack source can generate at most 128 honeypots. So we give worms a certain degree of freedom to infect other honeypots. The intrusion detection system recognises different types of worms and corresponding vulnerable honeypots are generated in order to study the behaviours of each worm type. Figure 2(b) shows the time



(a) Time to infect next victim



(b) Worm propagation activity

Fig. 2. Statistics of worm propagation

line of propagation for each worm monitored. We find that worm propagates exponentially. The Morto worm is the first one finishing infecting 128 honeypots. It takes around 50 s. The Flame and Blaster worms are slower.

3.4 Forensic Analysis

Over five-month deployment period, we captured more than 1000 attacks. Here, we present the forensic details about the first successful exploit event to the Nanjing, Wuhan and Changsha honeypots.

Shellshock Exploit. We installed Redhat operating system (Enterprise version 7.0) on the Nanjing honeypot. The Apache web server (version 1.3) is running on top of the operating system and the mod-cgi feature is enabled. The Bash version is 4.2. We run the Sebek client module on the operating system. Bash of which the version is before 4.3 has the shellshock vulnerability. The definition of *"var =() {:;}; command"* defines a variable *var* as a function in Bash scripts, and the following "command will be executed when the Bash sentence is interpreted. So if the attacker assigns *'() {:;}; command"* to the '*HTTP_USER_AGENT*' field in the HTTP request, the *HTTP_USER_AGENT* will be passed to Bash by the web server and then Bash executes the *command* followed by the function definition. In this case, the *HTTP_USER_AGENT* is no longer taken as a meaningless string and is likely to be exploited by hackers as a malicious input to the web server. Hence risk exists.

The honeypot was deployed in the honeyfarm at 1:10 pm on 2nd Feb 2017. It was compromised at 10:34 am on 4th Feb 2017. The attacker first connected to the honeypot and scanned the website for vulnerabilities such as SQL injection, XSS (cross site scripting). After finding no such vulnerabilities, the attacker fabricated a malicious HTTP request containing *"HTTP_USER_AGENT=(){:;}; uname -a"* and tested whether our web server has the shellshock vulnerability. After confirming the existence of the vulnerability, the attacker used a HTTP request containing *HTTP_USER_AGENT=() { :;}; /bin/bash -c "cd /tmp; wget http://123.*.*.*/download/shv4.tar.gz.tar; tar xzf shv5.tar.gz.tar;./setup alice 6543"* to download the *shv*5 rootkit and installed the ssh server. The ssh server was configured with password alice, and the server port is set to 6543. Then the attacker connected to the ssh server at port 6543. The connection between the honeypot and the attacker was encrypted. Traditional packet analyser, e.g., tcpdump, is unable to analyse encrypted data. However, the Sebek module can capture keystrokes, as it hijacks the $SYS\_read()$ function. Through analysis of the keystrokes, we found that the attacker downloaded a BSSH2 script from "http://sshbruteforce.com". It is an ssh brute-force attacking tool. Using BSSH2, the attacker executed brute-force attack against IP addresses ranging from 122.96.0.1 to 122.96.255.255. The attacker then modified a number of binary files including ps, ifconfig, netstat, top, ls, find and md5sum by shv5. As a consequence, when the system user calls those programs, the output of these programs shield information about that the system has been compromised. For example, when we ran the netstat command and we found that the ssh server which was running on the port 6543 was hidden. The MD5 hash values of modified files were stored in .shmd5. Therefore, when running the *md5sum* command, the modified md5sum program obtained the original MD5 value from the .shmd5 file to avoid being detected.

MS15-067 Exploit. The Windows 7 operating system is installed on the Wuhan honeypot, which has the MS15-067 vulnerability. It is a critical remote code execution vulnerability in the remote desktop protocol (RDP). Through exploiting the vulnerability, attackers can execute codes with the administrator privilege. After compromising the system, attackers can perform various tasks including installing software, modifying user data and creating users.

The honeypot was deployed in the honeyfarm at 10:15 am on 5th Feb 2017 and first compromised at 3:50 pm on 6th Feb 2017. The attacker first established a TCP reverse shell and reversely connected to the attacker from the honeypot. Then the attacker listed all running processes on the honeypot and inserted its process into *iexplorer.exe* and hid its existence. Then the attacker shutdown anti-virus software and the firewall. A persistent connection was established and the system would reversely connect to the attacker's machine every 10 seconds after the system rebooted. The attacker searched *.pdf, *.doc, *.jpg files on the honeypot and downloaded some files. Then all files in the *C:\ Windows\System32\config* directory are downloaded. After that, the attacker scanned the production network. Finally, the attacker deleted all system and application logs in the *C:\ Windows\System32\winevt\Logs* directory.

4 Related Work

Honeyfarm is related with, but defers from the honeypot [9], honeynet [10] and distributed honeynet [11] architectures. Data control is a research issue in those architectures.

Gen I honeynet [3] presents two alternatives for data control. The first method is to deny all outbound connections. This ensures safety, but this approach cannot study worm and botnet behaviours, as their behaviours are restricted. The second approach allows a certain number of outbound connections and all subsequent connections are forbidden after the maximum number is reached. This method still has a risk to harm a certain number of hosts on the Internet.

Gen II and III honeynets [3] use rate limiting to reduce the outbound rate. They refuse attack traffic to go outside of the honeynet. Therefore, this method cannot study malware further behaviours [12]. Our mechanism redirects the attack traffic to the honeyfarm honeypots in order to capture subsequent activities, hence worm propagation and botnet behaviours can be well monitored.

He et al. [13] propose a data control mechanism to prevent hackers attacking websites on the Internet using a compromised honeypot. The first connection to the website is allowed, but the rate is limited in order to gain enough time to clone the same website in the honeyfarm. Subsequent access to the website is directed to the honeyfarm. This architecture requires installing a honeypot and a corresponding adjacent honeyfarm in the monitored network, while our architecture only installs a redirector on the monitored network and a central honeyfarm for all monitored networks, which is lightweight.

The GQ architecture is proposed in [14] to control malware in the honeyfarm. In the architecture, the GQ gateway split the honeyfarm and the Internet. Policies are implemented at the gateway to control outgoing connections, which includes forwarding, rate-limiting, dropping, redirecting, reflecting and rewriting. However, due to paper length limitations, the detail about under which condition to apply each policy is not discussed.

Forensic analysis is a hot topic in the security area. Fahdi et al. [15] and Nassif et al. [16] respectively present a data clustering approach to speed up the forensic analysis process. Most recent forensic analysis focus on instant messaging applications [17–19] and malware [20] on mobile devices, Although mobile applications attract so much attention, exploitation of traditional desktop vulnerabilities is still an important issue as new vulnerabilities are discovered every day. Our paper presents a comprehensive forensic study on a new honeyfarm architecture and we find that: (1) It is essential for the honeyfarm gateway to differentiate attack and non-attack traffic to perform fine-grained data control. (2) Worms propagate exponentially. (3) Hackers break a system through one or multiple vulnerabilities and therefore mitigating vulnerabilities can prevent hackers from breaking in.

5 Conclusion

Honeyfarm, a conceptual idea for honeypot deployment, is a promising tool for global network attack monitoring, correlation, forensic analysis and trend prediction. In order to protect the Internet from being attacked by compromised honeypots in the honeyfarm as well as being able to capture attack behaviours for study, the traffic must be controlled effectively. We presented and implemented a honeyfarm system, DOID, for such a purpose. We deployed the system on the Internet and conducted comprehensive experiments including attack event tracing, worm behaviour capture and forensic analysis, which confirm DOID is a good tool in attack monitoring and forensic analysis. We also summarised observations based on these experiments.

References

1. Jiang, X., Xu, D., Wang, Y.-M.: Collapsar: AVM-based honeyfarm and reverse honeyfarm architecture for network attack capture and detention. J. Parallel Distrib. Comput. **66**, 1165–1180 (2006)
2. Vrable, M., Ma, J., Chen, J., Moore, D., Vandekieft, E., Snoeren, A.C., Voelker, G.M., Savage, S.: Scalability, fidelity, and containment in the potemkin virtual honeyfarm. In: Proceedings of the ACM Symposium on Operating Systems Principles. ACM, Brighton, October 2005

3. Spitzner, L.: Honeypots: Tracking Hackers, pp. 1–480. Addison Wesley, Boston (2002)
4. Burkhart, M., Schatzmann, D., Trammell, B., Boschi, E., Plattner, B.: The role of network trace anonymization under attack. SIGCOMM Comput. Commun. Rev. **40**(1), 5–11 (2010)
5. Jain, P., Sardana, A.: Defending against internet worms using honeyfarm. In: Proceedings of the CUBE International Information Technology Conference, CUBE 2012, Pune, India, pp. 795–800. ACM (2012)
6. Krishnan, S., Snow, K.Z., Monrose, F.: Trail of bytes: efficient support for forensic analysis. In: Proceedings of the 17th ACM CCS, Chicago, Illinois, USA, pp. 50–60. ACM (2010)
7. Metasploitable2. https://community.rapid7.com/docs/DOC-1875
8. Tor technology. http://www.torproject.org/
9. Kaur, R., Singh, M.: A survey on zero-day polymorphic worm detection techniques. IEEE Commun. Surv. Tutor. **16**(3), 1520–1549 (2014)
10. Abbasi, F.H., Harris, R.J.: Experiences with a generation III virtual honeynet. In: Proceedings of ATNAC, pp. 1–9 (2009)
11. Kumar, S., Singh, P., Sehgal, R., Bhatia, J.S.: Distributed honeynet system using Gen III virtual honeynet. Int. J. Comput. Theory Eng. **4**(4), 537–541 (2012)
12. Kim, I.S., Kim, M.H.: Agent-based honeynet framework for protecting servers in campus networks. IET Inf. Secur. **6**(3), 202–211 (2012)
13. He, X.-Y., Lam, K.-Y., Chung, S.-L., Chi, C.-H., Sun, J.-G.: Real-time emulation of intrusion victim in honeyfarm. In: Chi, C.-H., Lam, K.-Y. (eds.) AWCC 2004. LNCS, vol. 3309, pp. 143–154. Springer, Heidelberg (2004). https://doi.org/10.1007/978-3-540-30483-8_18
14. Kreibich, C., Weaver, N., Kanich, C., Cui, W., Paxson, V.: GQ: practical containment for measuring modern malware systems. In: Proceedings of the 2011 ACM SIGCOMM IMC, Toronto, Canada, pp. 397–412. ACM (2011)
15. Al Fahdi, M., Clarke, N.L., Li, F., Furnell, S.M.: A suspect-oriented intelligent and automated computer forensic analysis. Digit. Invest. **18**, 65–76 (2016)
16. da Cruz Nassif, L.F., Huschka, E.R.: Document clustering for forensic analysis: an approach for improving computer inspection. IEEE Trans. Inf. Forensics Secur. **8**(1), 46–54 (2013)
17. Ovens, K.M., Morison, G.: Forensic analysis of Kik messenger on iOS devices. Digit. Invest. **17**, 40–52 (2016)
18. Walnycky, D., Baggili, I., Marrington, A., Moore, J., Breitinger, F.: Network and device forensic analysis of android social-messaging applications. Digit. Invest. **14**, 77–84 (2015)
19. Anglano, C.: Forensic analysis of whatsapp messenger on android smartphones. Digit. Invest. **11**, 201–213 (2014)
20. Vrable, M., Ma, J., Chen, J., Moore, D., Vandekieft, E., Snoeren, A.C., Voelker, G.M., Savage, S.: A forensic analysis of android malware how is malware written and how it could be detected? In: Proceedings of IEEE 38th Annual International Computer, Software and Applications Conference (2014)