A Honeyfarm Data Control Mechanism: Design, Implementation, Evaluation and Forensic Study

Wei YIN, Hongjian ZHOU, Mingyang WANG and Zhiwen JIN,

North China Institute of Computing Technology, China

Summary
Data control for the honeyfarm should protect the Internet from being attacked by compromised honeypots in the honeyfarm, while providing a controlled environment for malware behavior study. This paper presents such a honeyfarm and focuses on the design of a Data cOntrol mechanism based on Intrusion detection and Data redirection (DOID). The horizontal port scanning problem and DDoS attack problem are addressed in the proposed honeyfarm. Comprehensive experiments including attack event tracing, worm behavior study, forensic analysis, DDoS monitoring and performance evaluation display that DOID is an effective tool for attack monitoring and forensic analysis, with minimal overhead.

Key words:
Data Control, Honeyfarm, honeynet, forensic analysis

1. Introduction

Honeypot [1], representing as a vulnerable system, attracts hackers to probe, explore and attack. In the meantime, using monitoring software, security professionals can record intruders' behaviors on the compromised system for forensic analysis, so that we can better understand their motivations, toolkits and tactics [2]. This can raise the awareness of threat intelligence, leading to better design and development of anti-malware software and intrusion detection systems.

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 are not available. Also, attack monitoring and analysis are non-trivial and maintenance of honeypots in various locations introduces high cost. This motivates the honeyfarm architecture [3], as shown in Fig. 1. 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.

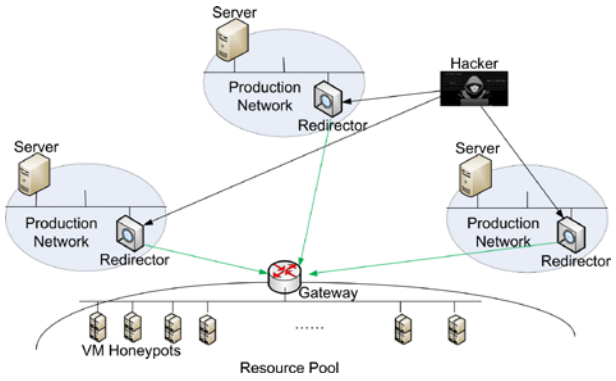


Fig. 1 Traditional honeyfarm architecture

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 [3] and Potemkin [4]. Collapsar realizes 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 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 utilized 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 behaviors. 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 behaviors cannot be blocked or contained. Consequently, hackers' behaviors in a honeyfarm need to be fine-grained controlled. However, in both realizations, the containment problem is not well addressed, or not even mentioned. Simply blocking all outbound connections may not work efficiently [5], because a hacker may download and install software on a compromised system. Blocking such non-attack connections may not be appropriate for studying the

Manuscript received June 5, 2018
Manuscript revised June 20, 2018

hacker's behaviors. Restricting outbound traffic rate and the number of outgoing connections [5] 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 [6], we propose such a data control mechanism for the honeyfarm. in which the intrusion detection system (IDS) and the reverse firewall are introduced. The IDS is used to recognize attack traffic and redirects the attack traffic to an emulated target to study hacker's further behaviors. Non-attack traffic is not restricted, hence toolkits downloaded by hackers can be captured. We also present attack event tracing, worm behavior study and a forensic study using the honeyfarm.

In this paper, we extend it by (1) proposing a mechanism to address the horizontal port scanning problem to provide fairness, efficiency and robustness, (2) proposing a mechanism to address the DDoS problem to ensure that DDoS traffic is blocked from going out of the honeyfarm, not to do any harm to Internet servers and hosts, (3) performance evaluation of the data control mechanism from two aspects: the overhead of the DOID honeyfarm and the experimental study of the mechanism to address the horizontal port scanning problem, (4) another forensic study in which attackers exploit the SQL injection and MS16-016 vulnerability, (5) DDoS attack monitoring and analysis using the honeyfarm and (6) the implementation of the data control mechanism.

The rest of the paper is organized as follows. Section 2 discusses the DOID data control mechanism. The implementation of this mechanism is discussed in section 3. The experimental setup and results with the deployed DOID architecture is discussed in section 4. The performance study is discussed in section 5. The related work is presented in section 6. The paper concludes in section 7.

2. Data Control Mechanism

In this section, we present the DOID containment architecture, and the defined policies for processing incoming/outcoming packets by the DOID gateway, followed by the two mechanisms to address the horizontal port scanning problem and the DDoS attack problem.

2.1 Containment Architecture

In the traditional honeyfarm architecture as shown in Fig. 1, redirectors in production networks transfer 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 behaviors should be contained and monitored in the honeyfarm so that their features can be studied and understood. In our design as shown in Fig. 2, 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.

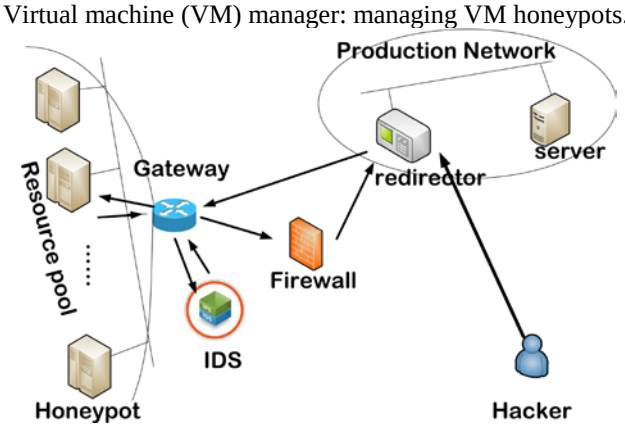


Fig. 2 DOID honeyfarm architecture

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 utilized for both inbound and outbound traffic checking. The inbound traffic will be checked by the IDS before forwarding to honeypots in the honeyfarm and known attack packets can be dropped 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.

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 packets will be dropped if the DOID gateway is configured to drop known attack packets. 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, if compromised, 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 Drop 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 drop known attack packets 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 gaeway 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 utilized to recognize 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 behaviors can be monitored as well.

Packet Drop Policy: Horizontal port scanning generats 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.

2.3 Address horizontal port scanning

Horizontal scanning is a type of port scan that targets at the same port on a group of hosts. Scanning internal addresses is considered attack traffic in our honeyfarm, therefore the DOID gateway will redirect the scanning traffic to emulated honeypots. Since horizontal scanning can generate a large number of honeypots, without restriction, this can quickly consume all honeyfarm resource.

Four measures are combined in the DOID architecture to deal with the resource overuse problem, including limiting the number of honeypots generated by an attack source to a certain threshold, recycling inactive honeypots which do not have communication for a period of time, filtering horizontal port scanning which allows only a percentage of scanning packets to generate VM honeypots, and limiting the total number of honeypots to a percentage of the maximum supported.

Limiting the maximum number of honeypots generated by each attack source, the number of generated honeypots will remain constant even if the group of attack sources launches more scanning requests. However, this number will increase if the number of attack sources increases. So the recycling policy to reclaim honeypots and the maximum total honeypot limitation policy can be used as the safety guard.

Table 1: Tables and supported chains in iptables

Chain	***Supported Table***	***Which packets to apply***
PREROUTING	Raw, NAT, Mangle	Packets that comes to the network interface
IINPUT	Filter, Mangle	Packets targeted at a local socket
FORWARD	Filter, Mangle	Packets that are being routed through the server
OUTPUT	Raw, Filter, Mangle, NAT	Packets that the server generates
POSTROUTING	NAT, Mangle	Pakcets leaving the server

Recycling the honeypot after a period of inactive time, the number of honeypots is going to decrease as time elapses. However, if a very large number of scanning requests are issued within a very short time and the requests continue to go on, this approach alone cannot guarantee safety. So we need the filtering approach and the maximum honeypot limitation measure.

In the filtering approach, because the DOID gateway responses to only a percentage of the scanning packets, the number of honeypots generated could be small. However, as more port scanning requests are issued, this approach alone cannot prevent the honeyfarm from resource overuse. Therefore other approaches, such as limiting the total number of honeypots, should be combined to prevent it from growing.

Limiting the total number of honeypots to a percentage of the maximum supported, the number of honeypots stays constant once the maximum is reached. The limitation of this approach is that it cannot generate further honeypots for subsequent requests from other attack sources. So the fairness problem should be addressed, which can be provided by limiting the maximum honeypots per attack source. The recycling policy could help as well as it destroys outdated honeypots.

As discussed above, each approach has benefits and drawbacks, so we combine those approaches to provide efficiency, fairness and security.

2.4 Address DDoS attacks

The compromised honeypots in the honeyfarm should not be used to launch attacks to Internet servers and hosts. A reverse firewall, implemented by iptables, is utilized as the tool to address the DDoS attack problem. The rules are implemented for outgoing packets instead of incoming packets. There are four different tables in the iptables firewall including Filter, NAT, Mangle and Raw. The Filter table is the default table that configuration rules will be applied if the table is not specified explicitly by the -t option. The NAT table is responsible for network address translation. When a new connection is created, the NAT table will be checked for rules. The Mangle table can be used to modify packets including headers. The Raw table can exclude packets from connection tracking. Each above tables support a set of iptables chains which may include PREROUTING, INPUT, FORWARD, OUTPUT and POSTROUTING. Their relationship is shown in Table 1.

Those packets that come and leave the iptables server will be our concern so we are interested in three types of chains including PREROUTING, FORWARD and POSTROUTING. Most TCP-based DDoS attacks have a very high data rate, so it is important to consider the performance of iptables rules. The first chain that can be applied to a packet is the PREROUTING chain. So it is better to filter malicious packets in this chain without causing any processing overhead for other chains. However, the PREROUTING chain is not supported in the Filter table. So we resort to the Mangle table to make our anti-DDoS iptables rules.

DDoS attacks are so complex that it is almost impossible to implement a signature-based approach against all of them. A nf_conntrack kernel module which is capable of connection tracking can assist us with mitigating almost all seemingly legitimate TCP-based DDoS attacks that do not use SYN packets. These attacks include ACK, SYN-ACK DDoS attacks and DDoS attacks using the bogus TCP flags. Using the conntrack module, we first set a rule with the Mangle PREROUTING chain to block any packet which is not an SYN packet and does not belong to an established TCP connection.

The TCP SYN packet that uses an uncommon TCP MSS value is blocked. This can mitigate dumb SYN floods. We also block packets with various bogus TCP flags, which are not used in legitimate packets. A spoofed packet that seems to be originated from private subnets is filtered by iptables as well. We also configure the PREROUTING chain to block all ICMP packets to prevent the Ping of Death attack, ICMP flood and ICMP fragmentation flood.

The connection from inside honeypots that have more than 100 established connections is blocked. This number can be tuned if needed. This approach is useful to deal with connection attacks. Fragmented packets are blocked to mitigate UDP fragmentation flood. We also filter the TCP RST packets to mitigate TCP RST floods.

In UDP reflection attacks including DNS and NTP reflection attacks, attackers usually send a short request with a forged source IP address to the server which responses with a large reply. The reply is sent to the victim that the forged source address represents. With a plenty of forged requests, the victiim's bandwidth will be overwhelmed. The iptables is configured to deny UDP reflection attacks by not allowing packets with the corresponding UDP ports to cross the boundary. We

finally set the rate limit for general traffic below a limit, as the last resort to DDoS attacks.

3. Implementation

Click [7] that is a framework to build configurable and flexible routers is used as the basic component for constructing our DOID gateway. The Click router is comprised of a number of packet processing modules named elements. A special description language, which is called Click configuration, can specify how each element is connected together to shape a directed graph. The graph describes how a packet is processed by elements within the router. In our implementation, around 9900 lines of element code and roughly 1500 lines of configuration code are added to the Click module.

All VM (virtual machine) honeypots are installed in a cloud environment, running on top of Open vSwitch that is a multilayer virtual switch. In order to isolate every VM honeypot from others, each honeypot is configured in a separate VLAN.

IDS is implemented by the snort intrusion detection system. The DOID gateway forwards packets to IDS for attack detection. IDS responses to the DOID gateway after comparing the packet with a signature database. The reverse firewall is implemented by iptables but the policy is enforced on outgoing packets instead of incoming packets.

A redirector is a single board computer installed on a network to be monitored. It is running the Linux system configured with an unused IP address across the network. A piece of software is installed to capture the traffic destinated at the configured IP address. If the traffic is not from the DOID gateway, the redirector encapsulates and sends it to the DOID gateway. Otherwise, it decapsulates it and gets the payload which is then sent to the destination.

4. Experiments with DOID Honeyfarm

In this section, we discuss the experiment environment setup and the evaluation results including attack event tracing, worm behavior study, forensic analysis, and DDoS attack monitoring

4.1 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 2 shows the configuration of these honeypots. The monitoring starts from 1st Feb 2017 to 20th July 2017.

4.2 Aggregate Statistics

Fig. 3 reveals 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 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 behaviors can be monitored. This confirms that our design is appropriate for these scenarios.

For 134 outgoing attacks (not counting DDoS 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

Table 2: Tables and supported chains in iptables

City	***Honeypot configuration***	
	Operating system	***Running service***
Shanghai (SH)	Ubuntu metasploitable2 [8]	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 3.0 with mod-cgi feature). Bash not patched	HTTP
Wuhan(WH)	Win 7. Patched without fixing the MS15-067 vulnerability	RDP
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

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 [9] make the source tracing very difficult.

4.3 Worm propagation analysis

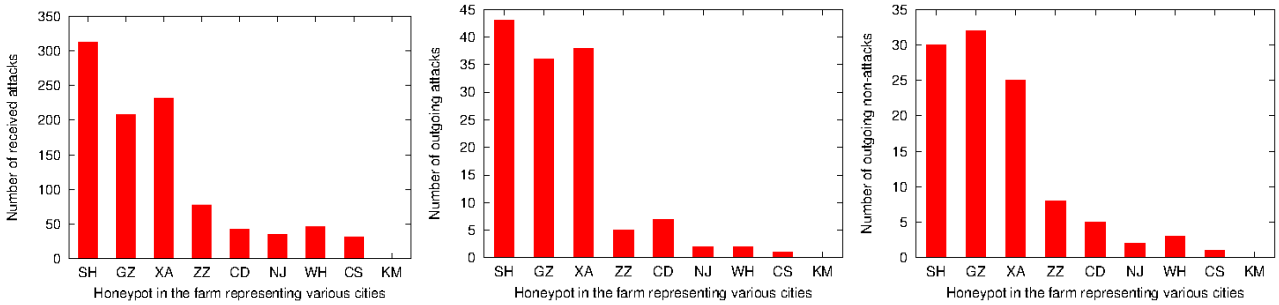
In the design of the DOID architecture, attack behaviors are contained and monitored so that it can be utilized to capture, contain and study worm propagation behaviors. Simply blocking attack traffic fails to give worms freedom to propagate. Allowing worms to propagate to the Internet causes the liability issue and worm behaviors 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. Fig. 4(1) 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 recognizes different types of worms and corresponding vulnerable honeypots are generated in order to study the behaviors of each worm type. Fig. 4(2) shows the time 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.

4.4 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



(1) Incoming attacks (2) Outgoing attacks (3) Outgoing non-attacks

Fig. 3 Outgoing attack and non-attack traffic

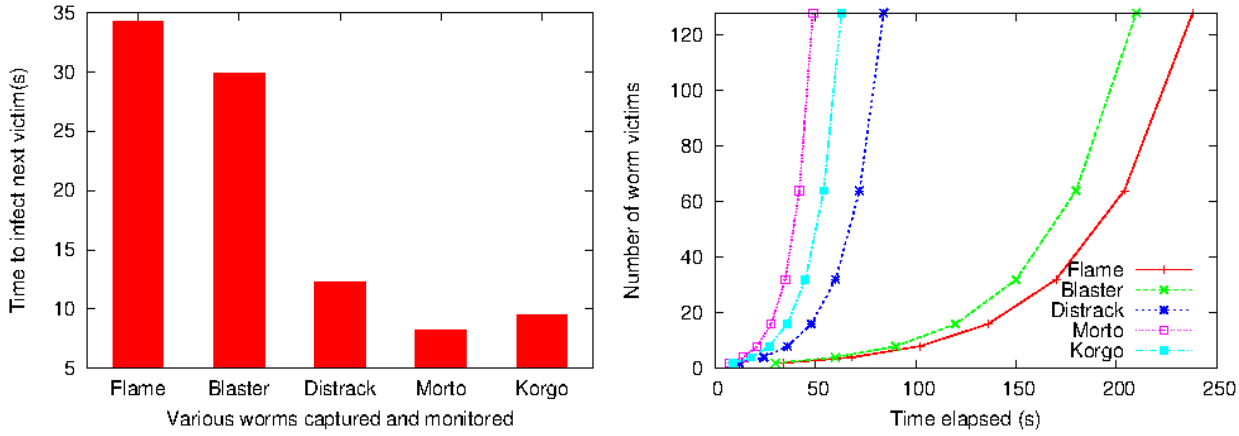
is before 4.3 has the shellshock vulnerability. The definition of "var =() {:;}; command" defines a variable var as a function in Bash scripts, and the following "command" will be executed when the Bash sentence is interpreted. So if the attacker assigns "*() {:;}; command*" to the '*HTTP_USER_AGENT*' 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 *shv5* 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 analyzer, e.g., tcpdump, is unable to analyze 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:\textbackslash Windows\textbackslash System32\textbackslash config* directory are downloaded. After that, the attacker scanned the production network. Finally, the attacker deleted all system and application logs in the *C:\textbackslash Windows\textbackslash System32\textbackslash winevt\textbackslash* Logs directory.

SQL injection and MS16-016 exploit The windows server 2008 is installed on the Changsha honeypot, which runs a web application. The Web application has a SQL



(1) Time to infect next victim (2) Worm propagation activity

Fig. 4 Statistics of Worm propagation

injection vulnerability. Through the vulnerability, a successful SQL injection exploit can read sensitive data from the database, modify database data and execute administration operations on the database. Also, the server has a MS16-016 vulnerability, which could allow privilege elevation. Combining these two vulnerabilities, the attacker can compromise the honeypot and get the root privilege.

The honeypot was deployed in the honeyfarm at 2:30 pm on 6th Feb 2017 and compromised at 10:23 am on 8th Feb 2017. The attacker first connected to the honeypot and started a port scanning. Since the honeypot had FTP, IIS and MySQL services started, the attacker tried weak password testing on port 21 and 1433. After detecting there were no weak passwords, he conducted a full web application vulnerability scanning. Three hours later, the scanning was finished and the hacker requested to a webpage, which is a SQL injection point, using the SQLMap tool. The attacker first extracted the database name, then dragged all table names from the database. After this, he dumped the username and password columns from the user table. The user table had an administrator username and the corresponding password in it. The password is encrypted by the MD5 hash algorithm. Then he tried to open a shell using the SQLMap tool but failed. The hacker started another quick scanning and stopped it when the background login page was found. After this, the hacker logan on the website using the administrator username and the decrypted password. Then the hacker browsed the management interface of the website and found a file upload point. He uploaded a shell.php file containing a piece of code as "*<?php @eval(\$_POST['pass']*);?>". The malicious code could enable the hacker to run any PHP code given by the *pass* parameter. For example, all files in the current directory could be listed by calling this shell.php file as "*http://website/shell.php?pass=system("dir")*". However, this shell.php file was detected and deleted by the antivirus software. Three minutes later, the hacker uploaded another shell.jpg file. This file included the same malicious code in it. Another connect.php file was uploaded afterwards, which contained a piece of code as "<!-\#include file="shell.jpg" ->". In this case, the malicious code was hidden in the shell.jpg file to avoid detection and the connect.php file was linked to the shell.jpg file. When requesting the connect.php file, the malicious code in the shell.jpg file would be executed.

Then the attacker connected to the web server through the PHP trojan horse using the China Chopper software, so that the hacker could manage the file system and execute system commands. The hacker executed the "whoami" command and found he was just a guest user. The hacker also executed the "systeminfo" command and found out the detailed system information. He uploaded the Windows Credentials Editor (WCE) software, attempting to obtain cleartext passwords entered by users at logon. But this file was deleted by the antivirus software. Then the hacker uploaded an MS16-016 exploit. It was executed and the hacker obtained the system privilege.

4.5 Monitored DDoS attacks

We totally captured 165 DDoS attack events. The distribution of DDoS attack types is illustrated in Fig. 5. One observation is that reflection attacks occur the most frequently. Reflection attacks accounted for 80.5% of total DDoS attacks, It increased by 30% in July 2017 compared to that in Feb 2017. Among reflection attacks, NTP reflection attacks increased most significantly, followed by SSDP reflection attacks, CHARGEN reflection attacks and SNMP reflection attacks.

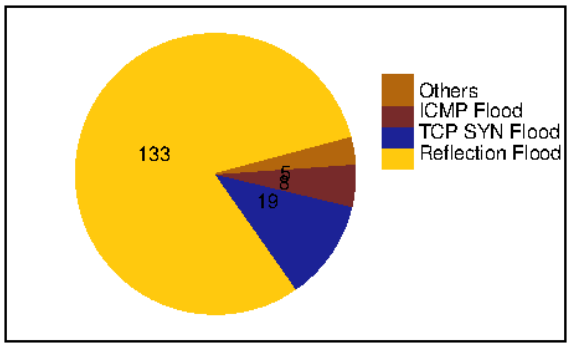


Fig. 5 Distribution of DDoS attack types.

The second observation is that China was the main target for DDoS attacks. In those DDoS attacks, China was the victim for most of the time, accounting for 43% of the attacks, followed by the USA with 21.4% of the attacks. But looking at the monthly statistics separately. The number of attacks targeting China was decreased by 14% from Feb 2017 to July 217, while for the USA this figure slightly increased. Looking at China's statistics more closely, the most attacked cities in China were Beijing, Shanghai and Shenzhen.

Another observation is that the during of attacks was increasing. In Feb 2017, the average DDoS attack duration was 1.5 hours. However, the during time continued to increase, reaching 7 hours in July 2017. The most durable DDoS attack lasted for 2 days, generating 21 GB of traffic in total.

5. Performance Evaluation

The DOID architecture provides effective support for data containment. However, the use of IDS and reverse firewall inevitably introduces performance degradation. In this section, we study the delay introduced by the DOID architecture. The DOID architecture combines four approaches to address the honeypot overuse problem caused by horizontal port scanning. In this section, we also study the performance of the proposed approaches.

5.1 Introduced Delay

We construct a TCP source using a honeypot in the resource pool and a TCP sink using a host on the Internet. The sink is two hops away from the source. The DOID gateway is configured to use IDS, use the reverse firewall (RF), use both and use neither. The TCP source repeatedly transmits a 4.2GB video file to the TCP sink. We vary the TCP packet size from 400 to 1400 bytes. The end-to-end delay is measured at the TCP source and the experiment is carried out for ten times. The results are averaged and shown in Fig. 6.

It can be observed from Fig. 6 that both IDS and RF introduce a delay to TCP packets because the DOID gateway passes packets to these systems for processing. IDS compares packets with a malware fingerprint database and the reverse firewall just compares packet header information with configured policies, hence IDS introduces more delay than the reverse firewall. Totally IDS and the reverse firewall cause around 6 ms delay to the system. We use Gigabyte links to connect the TCP source and sink, hence the difference of transmission time for a 400 and 1400 byte packet is not significant. Furthermore, the sink is only two hops away from the source, hence packets are transmitted for twice before reaching the sink. For these two reasons, the end-to-end delay does not vary much according to different packet sizes.

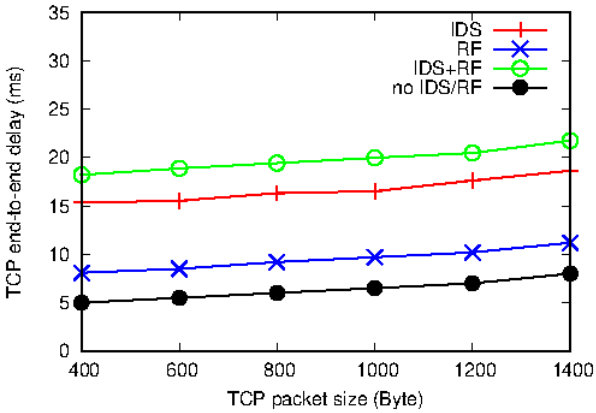


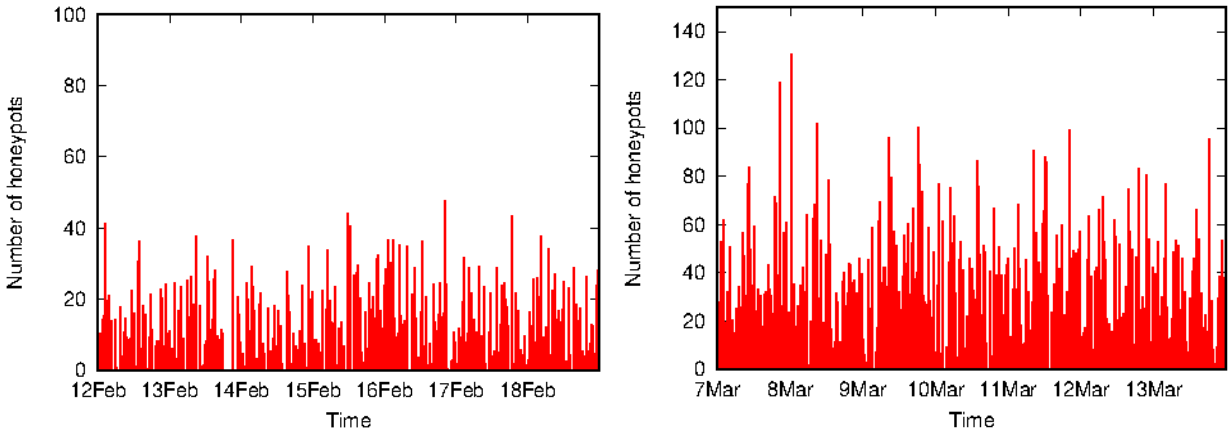
Fig. 6 End-to-end delay introduced by DOID.

5.2 Anti Resource Overuse

As discussed in section 2.3, four measures are combined in the DOID architecture to deal with the resource overuse problem. In deployment, by default, we limit the number of honeypots generated by an attack source to 128. The honeyfarm recycles inactive honeypots which do not have communication for more than 30 minutes. Scanning requests are filtered and we allow only 5% scanning packets to generate VM honeypots. The allowable total number of honeypots in the honeyfarm is set to 812, which is 80% of the maximum supported. With such a configuration, we run the honeyfarm for a period of a week. Then we configure the recycling time to be 120 minutes and run it for another week. We monitored the number of honeypots every 5 minutes and plot the statistics in Fig. 7.

As shown from Fig. 7, the average number of honeypots for the 30-minute case is lower than the 120-minute case. The number of honeypots for the peak time is different for each case as well. It is 45 for the first case and around 129

for the second case. This is because honeypots can live longer in the second case.



(1) Recycling time of 30 minutes. (2) Recycling time of 120 minutes.

Fig. 7 Number of honeypots by varying recycling time.

We also vary the percentage of scanning requests (10% and 20%) to generate honeypots. Other configurations are set to the defaulit. The statistics are plotted in Fig. 8. As shown from this figure, the average number of honeypots for the 20% case is higher than the 10% case. The value at the peak time is higher for the 20% case as well. This is because for the same number of requests, the number of honeypots generated is larger for the 20% case.

To draw a conclusion from Fig. 7 and Fig. 8, the utilization of honeyfarm can be controlled by tunning the parameters. Therefore, we can keep the honeyfarm safe.

6. Related Work

Honeyfarm is related with, but defers from the honeypot [10], honeynet [11] and distributed honeynet [12] architectures. Data control is a research issue in those architectures.

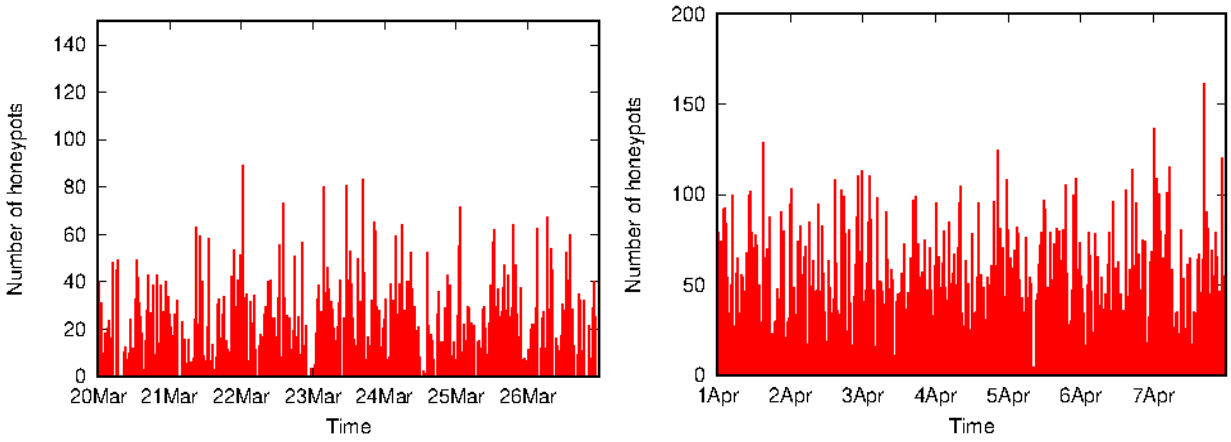
Gen Ⅰ honeynet [5] presents two alternatives for data control. The first method is to deny all outbound connections. This ensures safety, but this approach cannot study worm and botnet behaviours, as their behaviours are restricted. The second approach allows a certain number of outbound connections 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 [5] 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 [13]. 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. [14] 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 [15] 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. [16] and Nassif et al. [17] respectively present a data clustering approach to speed up the forensic analysis process. Most recent forensic analysis focus on instant messaging applications [18][19][20] and malware



(1) Generate honeypots for 10% of scanning requests. (2) Generate honeypots for 20% of scanning requests

Fig. 8 Number of honeypots by varying generation percentage

[21] 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.

7. 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 behaviors for study, the traffic must be controlled effectively. We presented and implemented a honeyfarm system, DOID, for such a purpose. We also addressed the problem caused by horizontal port scanning and DDoS attacks. We deployed the system on the Internet and conducted comprehensive experiments including attack event tracing, worm behavior capture, forensic analysis, DDoS study and performance evaluation, which confirm DOID is an effective tool in attack monitoring and forensic analysis, with reasonable overhead.

Acknowledgments

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

References

[1] Zhenxin Zhan andÂa MaochaoXuÂa and ShouhuaiXu. Characterizinghoneypot-captured cyber attacks: Statistical framework and casestudy. IEEE Transactions on Information Forensics and Security,8(11):1775âAS–1789, Nov 2013.

[2] Matthias Wählisch, Sebastian Trapp, Christian Keil, Jochen Schönfelder,Thomas C. schmidt, and Jochen Schiller. First insights froma mobile honeypot. In Proceedings of the ACM SIGCOMM 2012, pages 305–306, Helsinki, Finland, 2012. ACM.

[3] Xuxian Jiang, Dongyan Xu, and Yi-Min Wang. Collapsar:avmbased honeyfarm and reverse honeyfarm architecture for network attack capture and detention. Journal of parallel and distributed computing, 66:1165âAS–1180, Apr 2006.

[4] Michael Vrable, Justin Ma, Jay Chen, David Moore, Erik Vandekieft, Alex C. Snoeren, Geoffrey M. Voelker, and Stefan Savage. Scalability, fidelity, and containment in the potemkin virtual honeyfarm. In Proceedings of the ACM Symposium on Operating Systems Principles, Brighton, United Kingdom, Oct. 2005. ACM.

[5] Lance Spitzner. Honeypots: Tracking hackers. Addison Wesley, pages 0–480, 2002.

[6] Wei Yin, Hongjian Zhou, and Chunlei Yang. A honeyfarm data control mechanism and forensic study. In Proceedings of EAI ChinaCOM, pages 1–11, 2017.

[7] Eddie Kohler, Robert Morris, Benjie Chen, John Jannotti, and M. Frans Kaashoek. The click modular router. ACM Transactions on Computer Systems, 18(3):263–297, aug 2000.

[8] Metasploitable2. https://community.rapid7.com/docs/DOC-1875.
[9] Tor technology. http://www.torproject.org/.
[10] Ratinder Kaur and Maninder Singh. A survey on zero-day polymorphic worm detection techniques. IEEE Communications Surveys & Tutorials, 16(3):1520 – 1549, March 2014.
[11] Fahim H. Abbasi and R. J. Harris. Experiences with a generation III virtual honeynet. In Proceedings of ATNAC, pages 1–9, 2009.
[12] Sanjeev Kumar, Paramdeep Singh, Rakesh Sehgal, and J. S. Bhatia. Distributed honeynet system using Gen III virtual honeynet. International Journal of Computer Theory and Engineering, 4(4):537–541,2012.
[13] I.S. Kim and M.H. Kim. Agent-based honeynet framework for protecting servers in campus networks. IET Information Security, 6(3):202 – 211, Sep 2012.
[14] Xing-Yun He, Kwok-Yan Lam, Siu-Leung Chung, Chi-Hung Chi, and Jia-Guang Sun. Real-time emulation of intrusion victim in honeyFarm, pages 143–154. Springer Berlin Heidelberg, Berlin, Heidelberg, 2004.
[15] Christian Kreibich, Nicholas Weaver, Chris Kanich, Weidong Cui, and Vern Paxson. GQ: Practical containment for measuring modern malware systems. In Proceedings of the 2011 ACM SIGCOMMIMC, pages 397–412, Toronto, Canada, 2011. ACM.
[16] M. Al Fahdi,N. L. Clarke, F. Li, and S. M. Furnell. Asuspect-oriented intelligent and automated computer forensic analysis. Digital Investigation, 18:65 – 76, Aug 2016.
[17] Luis Filipe da Cruz Nassif and Eduardo Raul huschka. Document clustering for forensic analysis: An approach for improving computer inspection. IEEE Transaction on Information Forensics and Security, 8(1):46âAS–54, Jan 2013.
[18] kenneth M. Ovens and Gordon morison. Forensic analysis of kik messenger on ios devices. Digital Investigation, 17:40 – 52, Apr 2016.
[19] Daniel Walnycky, Ibrahim Baggili, Andrew Marrington, Jason Moore, and Frank Breitinger. Network and device forensic analysis of android social-messaging applications. Digital Investigation, 14:77 – 84, 2015.
[20] Cosimo Anglano. Forensic analysis of whatsapp messenger on android smartphones. Digital Investigation, 11:201 – 213, May 2014.
[21] Michael Vrable, Justin Ma, Jay Chen, David Moore, Erik Vandekieft, Alex C. Snoeren, Geoffrey M.Voelker, and Stefan Savage. Aforensic 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

Wei YIN received his Ph.D degrees from the University of Queensland, Australia in 2012. He is a research engineer in North China Institute of Computing Technology. His research interest include rate adaptation in wireless networks, honeypots and honey encryption.