2014

Finding evidence of wordlists being deployed against SSH honeypots – implications and impacts

Priya Rabadia
Edith Cowan University, prabadia@our.ecu.edu.au

Craig Valli
Edith Cowan University, c.valli@ecu.edu.au

Recommended Citation

Rabadia, P., & Valli, C. (2014). Finding evidence of wordlists being deployed against SSH honeypots – implications and impacts. DOI: https://doi.org/10.4225/75/57b3e7d5fb882

DOI: 10.4225/75/57b3e7d5fb882

12th Australian Digital Forensics Conference. Held on the 1-3 December, 2014 at Edith Cowan University, Joondalup Campus, Perth, Western Australia.
This Conference Proceeding is posted at Research Online.
https://ro.ecu.edu.au/adf/141

FINDING EVIDENCE OF WORDLISTS BEING DEPLOYED AGAINST SSH HONEYPOTS – IMPLICATIONS AND IMPACTS

Priya Rabadia and Craig Valli
Edith Cowan University, Security Research Institute, Perth, Australia
prabadia@our.ecu.edu.au, c.valli@ecu.edu.au

Abstract

This paper is an investigation focusing on activities detected by three SSH honeypots that utilise Kippo honeypot software. The honeypots were located on the same /24 IPv4 network and configured as identically as possible. The honeypots used the same base software and hardware configurations. The data from the honeypots were collected during the period 17th July 2012 and 26th November 2013, a total of 497 active day periods. The analysis in this paper focuses on the techniques used to attempt to gain access to these systems by attacking entities. Although all three honeypots are have the same configuration settings and are located on the same IPv4 /24 subnet work space, there is a variation between the numbers of activities recorded on each honeypots. Automated password guessing using wordlists is one technique employed by cyber criminals in attempts to gain access to devices on the Internet. The research suggests there is wide use of automated password tools and wordlists in attempts to gain access to the SSH honeypots, there are also a wide range of account types being probed.

Keywords: Cyber Security, SSH, Secure Shell, Honeypots

INTRODUCTION

This paper is an investigation focusing on activities detected by three Secure Shell (SSH) honeypots that utilise the Kippo honeypot software. This paper is part of an ongoing investigation with initial work conducted in 2012 and 2013 (Valli, 2012; Valli, Rabadia, & Woodward, 2013). This investigation is similar to the research conducted by Owens and Matthews, however this paper focusing on the use of wordlist being deployed to again access to a system. All three SSH honeypots were configured identically using Ubuntu 11 Long Term Support (LTS) server as their base operating systems. Additionally, they were located on the same IPv4 /24 subnet work space on virtual private servers (VPS) (Valli et al., 2013). The Kippo SSH honeypots are referred to as Mopoke, Quokka and Lair. The data examined in this paper were collected over a 497 day period from the 17th July 2012 until 26th November 2013.

The focus of this particular research is to identify evidence of automated attacks using password wordlists being implemented to login and gain access to the three Kippo SSH honeypots. All three honeypots have the same username and password databases that contain multiple "correct" login password combinations. These "correct" combinations are part of the deception that is presented to the attacking entity by the Kippo SSH honeypot. The passwords in these lists are drawn from well known "bad" password lists.

OVERVIEW OF THE SETUP OF THE KIPPO SSH HONEYPOTS

The Kippo SSH honeypot is a medium interaction honeypot, in that the honeypot imitates some functions that are exhibited by a 'real' system (Hosting, 2013; Stevens & Pohl, 2004). The Kippo honeypot is designed to effectively mimic a SSH server. The SSH protocol is designed to securely transmit data using a point to point encryption tunnel (Ylonen & Lonvick, 2006). The protocol provides high grade encryption and is a secure replacement for plaintext terminal programs such as telnet or rsh. Most network connected UNIX or UNIX-like operating systems have SSH installed as a client and often included as a server (daemon).

Kippo honeypots are designed to collect data from attacker interaction with an emulated SSH. The emulated SSH service is provided by an open-source, Python based event-driven program called Twisted (TwistedMatrixLabs, 2013). Twisted provides the libraries that are utilised and deployed by Kippo honeypot to imitate a valid "encrypted" SSH session to an entity. The honeypot also emulates a fake file system to present to the user. The system also presents false system reporting and allows interaction with artefacts such as /proc/cpuinfo or .bash_history logfile. The level of deception in the default setting is limited however this functionality is able to be expanded and modified at will. For this experiment key elements were modified such as /proc entries and different bash entries.

The Kippo SSH honeypots are written in Python with a simple installation process. Source code was obtained from the kippo.googlecode.com Wiki (Code.google.com, 2012).(Labs, 2011). The setup for these particular systems used in the data collection was conducted as specified by the BruteForce Lab Guide (Labs, 2011). This deviates from the original Kippo SSH documentation and uses the authbind daemon instead of twistd as the initial connecting daemon for the service. The configuration lets authbind handle the binding of the twistd as a non-root user to a low numbered TCP port and passes this to the twistd daemon. This configuration has been found to be more consistent, reliable and secure during the conduct of the research project.

During the installation process a local MySQL database was configured securely to record all the interactions with the Kippo honeypots. Figure 1, is a table from (Valli, 2012) which was sourced from the Kippo documentation. It shows the MySQL database structure used in the Kippo honeypots to record all the interaction data.

TABLE auth id int(11) PK, session char(32) NOT NULL, success tinyint(1) NOT NULL, username varchar(100) NOT NULL, password varchar(100) NOT NULL, timestamp datetime NOT NULL,	TABLE input id int(11)NOT NULL PK session char(32) NOT NULL, timestamp datetime NOT NULL, realm varchar(50) default NULL, success tinyint(1) default NULL, input text NOT NULL, KEY session (session,timestamp,realm)
TABLE clients id int(4) PK version varchar(50) NOT NULL	TABLE sensors id int(11) NOT NULL (PK) ip varchar(15) NOT NULL
TABLE sessions id char(32) NOT NULL PK starttime datetime NOT NULL, endtime datetime default NULL, sensor int(4) NOT NULL, ip varchar(15) NOT NULL default '', termsize varchar(7) default NULL, client int(4) default NULL, KEY starttime (starttime,sensor)	TABLE ttylog id int(11) NOT NULL PK session char(32) NOT NULL ttylog mediumblob NOT NULL

Figure 1 - MySQL database structure for Kippo honeypot

After recording to the local MySQL database, these data are then transmitted to a centralised PostgreSQL SQL server that is running a Debian-Linux 5.1.49 operating system (Valli et al., 2013). Communication is achieved using a Python extension that uses a PostgreSQL driver to connect to the SURFIDS system IDS logging server (IDS, 2013). The centralised logging server utilises the SURFIDS system for storing the data from the honeypots into an aggregated PostgreSQL database. The database has functions and tables specifically for the Kippo honeypots data. In addition on the honeypots that run Kippo the researchers also operate Dionaea and Glastopf which also report to the SURFIDS instance, however these data are not used in this analysis.

GAINING ACCESS

To gain access to these honeypot systems, as in a real system the correct username and password must be entered at the emulated login screen. While general user accounts on well administered systems may have lockout of the account for unsuccessful attempts it is not a feature that is enabled on administrative and root accounts at any time. The reason being that simple denial of service can occur by locking administrative or root accounts. This Achilles heel of system availability of administrative accounts or system accounts can be exploited by the use of automated attack tools. The generic tool used for this type of activity is called colloquially a “password cracker”.

Passwords crackers can be deployed to identify the correct password by trying different passwords with contemporary speeds reaching billions of passwords a second when using multi CPU (Central Processing Unit)- or GPU (Graphic Processing Unit)-enabled password crackers. There is a finite number of passwords for any given system often referred to as a key space. While finite these key spaces still can be computationally large for example, the standard Windows LM (LAN Manager) password key space for all possible passwords is 2^{43}. While it is relatively infeasible for a single conventional computer to "crack" these passwords in a timely fashion, this does not hold for advanced techniques using compute clustering or GPU technology. Furthermore, techniques such as pre-computed rainbow tables can greatly increase speed as the key space is computed once and stored as a hash within a database table for easy reuse.

Passwords are typically stored as a cryptographic hash where the password has applied to it a cryptographic process that protects the actual password on the file system from compromise by storing it as a hash or set length ciphered text. A common method employed to achieve hashing is the MD5 hash format (Rivest, 1992). Without the use of hashing and cryptography the compromise is at its most simple. When stored in plaintext form, simply opening the file that contains the password and reading it. By applying cryptographic techniques to the password and storing this on the file system as a "hash" it becomes a relatively implausible for an attacker to obtain the password on a first guess (Preneel, 1999).

Breaking passwords

There are many different techniques that can be used. A brute force attack uses a systematic method of guessing the password by enumerating through a combination of characters, symbols and numbers of the allowable characters. Typically starting at one character and moving through the key space until all possibilities have been attempted (Preneel, 1999).

A dictionary attack uses hashes of words that appear in a dictionary and compares them to the stored password or feeds the hash as input to the login mechanism. This process of comparison continues until a match is found or the process is terminated through user input or poor programming (Chakrabarti & Singhal, 2007).

Rainbow tables also use hash value to identify the correct password. Rainbow tables are databases comprised of various character combinations that have been pre-computed and stored typically in an efficient binary structure, allowing fast retrieval (Kim, Seo, Hong, Park, & Kim, 2014).

Password techniques that utilise plaintext wordlists can also be deployed. These are lists of possible passwords typically stored as text strings a system could use. Depending on the application chosen the order of the words in the list could be in sequential or non-sequential order. Also some advanced techniques will rearrange the word into a series of patterns derived from a given word or string (i.e., a mask).

These types of attack tend to utilise social engineering techniques and deductive reasoning to pick viable candidate passwords. As an example if the password being probed is from a provincial business who is a football supporter that has a team called the Dunders, sponsors the team and the year is 2014. Plausible passwords would of course be GoDunders, Dunders2014, GoDunders14 and the ever cryptic "sredund". Likewise within any culture there are often key phrases or vernacular that they use to identify and belong to the group. For example in the "IT culture" there are passwords that are entirely plausible guesses take from the vernacular e.g., "iamr00t", "p0wned" or "hard scan". Intelligent attackers know this and there are "customised" wordlists available for download and use. In some cases these are provided as defaults with the security software distribution or attack utilities used in, for example, Kali. Kippo honeypot facilitates the use of these defaults to produce a list of acceptable passwords.

INTELLIGENCE GATHERED FROM THE THREE HONEYPOTS

During the period of coverage for this research there were a total 373,216 login attempts. The distribution of these attempts was uneven, Mopoke recorded 277,248, Lair recorded 80,159 and Quokka 15,809 attempts, respectively. This occurrence in of itself would bring serious question to the widespread, systematic scanning of the Internet by cyber criminals claimed by researchers, governments and the media.

The successful login percentage per honeypot over the data collection period was; Mopoke 954 or 0.344%, Lair 201 or 0.251% and Quokka with 111 or 0.702%. While Quokka was the least attacked of the three honeypots the success rate is the greatest, with an overall success rate of 1266 or 0.339%.

The methodology of the investigation begins with identifying the top 20 password attempts for the combined data set. Once a list had been compiled, it was compared with a set of known wordlists. When the wordlist had been identified; the top 20 passwords from each of the host was compiled and compared to the identified wordlists. If similarity were found further investigation was conducted.

Furthermore, the top 20 password attempts for each of the individual honeypots are markedly different; across the combined data from all three honeypots there is a contradictory set of top 20 passwords attempts. The purpose of this investigation is to find evidence of the consistent use of default password wordlists to gain access to the systems.

The Kippo honeypots have front loaded passwords derived from numerous known bad databases that will allow honeypot logins to interact with a fake command shell. This front-loading is a primary deceptive function of a honeypot. By allowing a login, an attacker believes that s/he has gained access to the system as a result of poor password practices by the system custodians. There is a caveat on use in that a system must not be too readily comprisable otherwise the deception advantage may be lost.

Combined Top 20 password login attempts

Figure 2 shows the top 20 passwords attempts for the combined data set from all three hosts. The most used login attempt was *123456* with 9548 attempts; closely followed *password* with 7169 attempts. *123123* recorded only 910 attempts. The passwords identified in figure 2 are passwords that should appear in a wordlist.

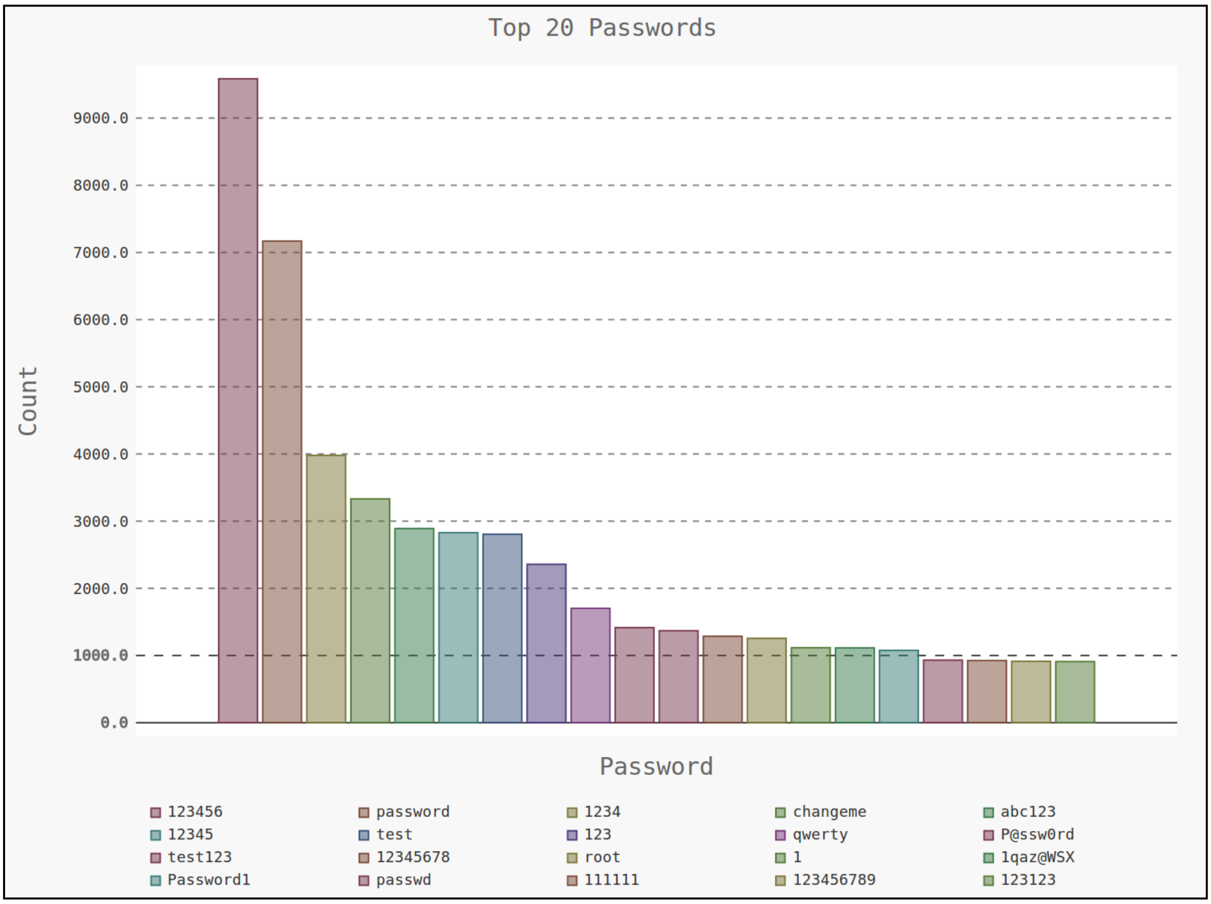

Figure 2: Combined Top 20 password attempts

Identifying wordlists

Two collections of open-source wordlists were compiled. The first, collection was from a simple Internet search for wordlists without a specified topic, 10 lists were found. The second collection was compiled from an open-source Linux distribution used for penetration testing, Kali. 17 wordlists were found (OffensiveSecurity, 2013). Table 2, shows a table of the two collections and the wordlist within each collection, some lists belong to more than one collection.

Internet search of wordlists	References	Wordlist found on the Kali distribution
500-worst-passwords.txt	(Warzone, 2011)	best11.txt
cain.txt	(Security, 2011; Warzone, 2011)	best15.txt
conflicker.txt	(Security, 2011)	best1050.txt
elitehacker.txt	(Warzone, 2011)	big.txt
honeynet.txt	(Warzone, 2011)	burnett_top_500.txt
john.txt	(Openwall, 2014; Security, 2011; Warzone, 2011)	burnett_top_1021.txt
password.lst	(Warzone, 2011)	common.txt
rockyou.txt	(Security, 2011; Warzone, 2011)	idrac_default_pass.txt
twitter-banner.txt	(Security, 2011)	ipmi_password.txt
world-tech-most_common_password.txt	(Warzone, 2011)	phpbb.txt
		rockyou.txt
		twitter.txt
		unix.passwords.txt

Table 1: Table of the two collections and the wordlist used

A *grep bash script* was used to parse each wordlist to identify which top 20 passwords the list contained. The wordlist *rockyou.txt* was found on both the Internet and the Kali distribution contained all the top 20 passwords. This occurrence indicates that the Kali Distribution and its older siblings in the Backtrack family distributions could be the dominate tool of choice for cyber criminals.

The three hosts

To progress further the top 20 passwords from each of the honeypots was compiled and compared to the *rockyou.txt* word list. The following table, Table 2 represents the top 20 password login attempts for each honeypot with additional columns for the top 20 combined password attempts. While each of the data sets have recorded different top 20 password attempts. It shows the top passwords attempt *123456* was the same for all three data sets.

Mopoke		Lair		Quokka		Combined	
123456	7350	123456	1637	123456	597	123456	9584
password	6204	password	808	password	157	password	7169
1234	3481	changeme	540	1234	125	1234	3979
changeme	2757	Test	422	12345	101	changeme	3332
abc123	2592	1234	373	1	90	abc123	2890
12345	2414	123	365	123	89	12345	2829
test	2300	12345	314	test	84	test	2806
123	1904	Qwerty	293	root	80	123	2358
qwerty	1342	Admin	281	abc123	70	qwerty	1703
p@ssw0rd	1102	test123	278	qwerty	68	P@ssw0rd	1416
test123	1056	p@ssw0rd	254		63	test123	1369
12345678	1032	Passwd	243	admin	62	12345678	1287
root	939	Root	237	p@ssw0rd	60	root	1256
1qaz@WSX	926	abc123	228	oracle	60	1	1117
password1	910	12345678	219	111111	54	1qaz@WSX	1114
1	823	1	204	123123	49	password1	1077
111111	726	123456789	198	1q2w3e4r	48	passwd	932
123123	715	admin123	191	123drag0s123	46	111111	925
P4ssw0rd	713	Letmein	179	123dragos123	45	123456789	914
654321	688	1qaz@WSX	174	1qaz2wsx	45	123123	910

Table 2: Top 20 passwords from the four data sets.

The top 20 attempts from Mopoke and Lair were also found in the *rockyou.txt* wordlist. However, 18 out 20 passwords were found for Quokka excluding, *123drag0s123* and *123dragos123*. Further, investigation showed the attempted time recording for both passwords where between the same hour time period; 2013-03-26 08:15:13 until 2013-03-26 09:12:12. Both of the passwords used the same username list to gain assesses in addition the attacks originated from the same IP address 94.127.XX.XX [anonymised] located in Serbia.

DISCUSSION AND CONCLUSION

All three honeypots have recorded passwords that have appeared in the *rockyou.txt* wordlist. The wordlist *rockyou.txt* is widely available as it is an open-source list found on the Internet additional it is one of pre-loaded wordlists found on the Kali distribution.

Quokka has two of the top 20 passwords not found in the list; *123drag0s123* and *123dragos123,* with 46 and 45 counts repeatedly. Further, investigation has shown the passwords had orientated from the same IP address 94.127.XX.XX in Serbia. A reasonable extrapolation here is that Dragan and Drago are common Slavic masculine names. Despite the two passwords not appearing in the rockyou.txt wordlist, the evidence still suggests the uses of the wordlist to gain access to a system.

As the reminder of the top 20 passwords from the three data sets appear in this list. The two outliers could suggest alternate forms of gaining access or simply ego on the part of the attacker or targeting systems owned by an intended victim with the name Drago or Dragan. The research has proven extensive use of the rockyou.txt dictionary in attacks on these particular Kippo based honeypots. The *rockyou.txt* dictionary is a default on Kali and its predecessor (BackTrack). Furthermore, the Kippo honeypot gathers the SSH banners from the attacking entities and again the banners are consistent with those that would be presented from a default Kali or BackTrack installation. These factors would tend to indicate possible default usage of a tool as a result of using a “guide” from the Internet or default mode use of a security tool.

This practise is referred to in derogatory cyber security vernacular as “script kiddie” activity. This mocks the activity as somehow sub-normal or lacking intellectual engagement, but misses the point about the dangerous nature and potential devastation this use can evince. This type activity is akin to thoughtless and criminal use of a weapon without consideration of second- and third- order consequences of action.

In conclusion this evidence starts to suggest it maybe time we start considering the possession of these cyber security weapons in the same way that we consider the possession of firearms, fissionable material or biologicals. The type of use of cyber security tools as evinced by the data collected in this project would support this consideration. Further research and analysis is required on the varying number of attacks recorded on the three identical honeypots that are currently collecting attacker data.

REFERENCES

Chakrabarti, S., & Singhal, M. (2007). Password-Based Authentication: Preventing Dictionary Attacks. *Computer, 40*(6), 68-74. doi: 10.1109/MC.2007.216

Code.google.com. (2012). Kippo shows up in Metasploit. *SSH Honeypot* Retrieved 23.09.2013, from https://code.google.com/p/kippo/issues/detail?id=48

Hosting, G. P. (2013). Kippo. *Kippo SSH Honeypot.* Retrieved 09.10.2013, from http://code.google.com/p/kippo/

IDS, S. (2013). SURFcert IDS. Retrieved 20/10/2013, from http://ids.surfnet.nl/wiki/doku.php

Kim, J. W., Seo, J., Hong, J., Park, K., & Kim, S.-R. (2014). High-speed parallel implementations of the rainbow method based on perfect tables in a heterogeneous system. *Software: Practice and Experience,* n/a-n/a. doi: 10.1002/spe.2257

Labs, B. (2011). Installing Kippo SSH Honeypot on Ubuntu. Retrieved 27.09.2013, from http://bruteforce.gr/installing-kippo-ssh-honeypot-on-ubuntu.html

OffensiveSecurity. (2013). Download your flavour of Kali Linux. 2013, from http://www.kali.org/downloads/

Openwall. (2014). Openwall wordlists collection. 2014, from http://www.openwall.com/wordlists/

Preneel, B. (1999). State-of-the-art ciphers for commercial applications. *Computers & Security, 18*(1), 67-74. doi: http://dx.doi.org/10.1016/S0167-4048(99)80009-1

Rivest, R. (1992). The MD5 Message-Digest Algorithm. from https://www.ietf.org/rfc/rfc1321.txt

Security, S. (2011). Leaked passwords. *Passwords.* 2014, from https://wiki.skullsecurity.org/Passwords

Stevens, R., & Pohl, H. (2004). Honeypots und Honeynets. *Informatik-Spektrum, 27*(3), 260-264. doi: 10.1007/s00287-004-0404-y

TwistedMatrixLabs. (2013). What is Twisted? Retrieved 23.09.2013, from http://twistedmatrix.com/trac/

Valli, C. (2012). *SSH: Somewhat Secure Host*. Paper presented at the Cycberspace Safety and Security, Melbourne Australia.

Valli, C., Rabadia, P., & Woodward, A. (2013). *Patterns and Patter - An Investigation into SSH Activity Using Kippo Honeypots*. Paper presented at the Australian Digital Forensics Conference, Edith Cowan University.

Warzone, C. (2011). Outpost9 Word List. *Password cracking: A mega collection of password cracking word lists.* 2014, from http://www.cyberwarzone.com/cyberwarfare/password-cracking-mega-collection-password-cracking-word-lists

Ylonen, T., & Lonvick, C. (2006). The Secure Shell (SSH) Connection Protocol. from https://tools.ietf.org/html/rfc4251