Dictionary Attack on Wordpress: Security and Forensic Analysis

Ar Kar Kyaw[1, 2]
Digital Forensic Research Labs[1]
Auckland University of Technology
Auckland, New Zealand

Franco Sioquim
Whitireia Community Polytechnic
Auckland, New Zealand

Justin Joseph
Faculty of Business and IT[2]
Whitireia Community Polytechnic
Auckland, New Zealand

***Abstract* - The effective forensic investigation of a security attack on a web application relies on the forensic readiness of the web application system, supportive forensic tools, and skills of the forensic investigator. Web application forensic readiness incorporates evidence collection by enabling logging and the evidence protection for those log files through techniques such as permission settings in order to retain the integrity. Furthermore, a forensic investigator should have a good comprehension of web application functionality, web server architecture, and web application security issues.**

This paper focuses on a dictionary attack experiment against Wordpress (a web application) administered by a persona named Peter Quill (a fictitious character). The dictionary attack was able to successfully guess the seven-character password used for the persona's user account. A set of techniques and tools are critically analysed to determine whether they can be applicable to the experiment scenario. The techniques mostly focus on retrieving the log files from the web server, the application server, the database server, and the web application itself, while the tools deal with collecting, analysing, and presenting the log file data.

Keywords—web forensics; dictionary attack; wordpress

I. INTRODUCTION

Password cracking attacks can be carried out through different methods such as Dictionary attacks, Brute Force attacks, and Rainbow Tables. All of them are generally a form of password guessing attack wherein the perpetrator attempts to infiltrate the system using sets of words derived from personal knowledge of the user or existing dictionary sets used by password cracking software such as John the Ripper and Cain and Abel [16]. One of the major contributor to their effectiveness is weak passwords utilized by users with their online accounts. An analysis [15] conducted on password strength in Greece shows that the average length is only seven characters, which is relatively low compared to the recommended 16 character passwords [5]. As a result, there are several cases of login credential theft and is still growing [2].

Forensic investigation is highly essential process in determining the evidence of unauthorized or criminal activity [13]. In the domain of Web Applications, there can be several ways to conduct a forensic investigation depending on the type of (web) application framework, platform, database, and web server. This paper specifically deals with Web Application Forensics for a password cracking attack experiment targeting Wordpress. Existing forensic tools and techniques from literature are critically analysed in conjunction with their applicability this experiment/case. In addition, this paper can also provide insights on how to implement a forensic ready Wordpress system and how to carry out the investigation on attacks on web applications.

This paper is structured as follows – the first section provides a review of existing literature to provide a background to concepts including attack types and the taxonomy and requirements for Web Application Forensics. The next section details the experimental setup and the results. Then, the next section discusses techniques that can be used for forensic investigations related to the experiment, which is followed by several countermeasures to mitigate the attacks conducted in the experiment. Lastly, the discussion section is followed by conclusion.

II. BACKGROUND AND RELATED WORKS

A. User Passwords

One factor that contributes to the effectiveness of the online password cracking attack is the easily-guessed passwords of the users. An analysis [15] of 19,000 actual passwords from different datasets come up with factors that generally describe that characteristics of the passwords such as the average length of less than 7 characters and mostly alphanumeric characters are used. Reuse of password across multiple websites which can be of leverage to the perpetrators in password guessing [1]. A survey denotes that 43-51% of users use the same password in their accounts on different websites. Thus, it is imperative for users to have a different password for each website account.

Other studies emphasize more on increasing the password strength to yield a highly complex password for attackers to crack. A context free grammar (CFG) password [14] has a minimum length of 8 characters, and some examples are "123fatherabc", "hyhycyhyh", and "[a*b]-[a-b]". However, there is no actual experiment conducted to test the strength of CFG password. Similar to having an easy memorization of password, [3] suggest a technique that calculates password for many accounts utilizing the strengthened cryptographic hash function. The strength of this approach is that users should only remember a single password for multiple websites through a browser extension. This extension accepts the master password and applies iterated hashing for the actual password input. Password strength is further emphasized by [5] in a simulation using seven password policies that can be imposed by the administrator on the system. An experiment is conducted to analyse the cracking using Weir-algorithm on 12,000

passwords with different applied policies. The results yield that basic16 policy, which is composed of at least 16 character passwords, is the most superior. Password policies are incorporated on some well-known websites such as social, blogging, email, shopping, and financial categories [1]. There are similarities across social, blogging, and email websites wherein majority implements a minimum of 6 and 8 character password policy.

Another approach to password creation is by using mnemonic passwords [6]. A survey is initiated to create a user generated mnemonic passwords then it is compared with control passwords. An example of mnemonic password "SWMtM$$!!" is based from a quote "Show me the money!" of the Jerry Maguirre movie while a control password can be "atreyu09" from the character "Atreyu of the Never Ending Story II" movie. The result indicates that 11% out of 146 control passwords are cracked compared to only 4% of 144 mnemonic passwords. However, it is suggested to avoid creating mnemonic phrases that are well-known in the internet.

B. Acquiring User Passwords

There are different methods to illegitimately acquire user passwords such as phishing websites and online password cracking attacks. Previous researchers [12] point out that passwords can also be gathered by exploitation of the password manager on the browser. An example is given in which a user is directed to a fake login page after connecting to a rogue Wi-Fi router. The login page is configured so that the password manager automatically fills the username and password fields.

On other hand, online authentication attacks are carried out by perpetrators that can be classified into four types based on the information at hand [3].

1. Attackers with stolen username.
2. Attackers with stolen username and password.
3. Attackers with stolen user's cache from a compromised machine.
4. Attackers with stolen cache and the website password.

This report focuses on the first type in which the correct password needs to be guessed through different techniques. Furthermore, attackers can compromise the passwords in four ways [6].

1. By gathering sufficient details about users that can be used to guess their passwords.
2. By social engineering to entice the users in disclosing their usernames and/or passwords.
3. By stealing passwords through shoulder surfing or spyware.
4. By using tools to crack passwords.

In one of the publications, online password cracking attacks can be classified into two forms of attacks, namely brute force and dictionary attacks [16]. In addition, [5] enumerate the factors that affect the effectiveness of password guessing such as the resistance of the password to guessing and the performance of password guessing attack.

C. Brute Force Attacks

This attack makes use of the possible combinations of the supplied characters [14, 16]. For example, a brute force attack with six character alphabets starts with "aaaaaa" and increments to "aaaaab", "aaaaac", "aaaaad" until "zzzzzz". Though these increments may vary depending on the configuration of the attacker. Furthermore, a variety of character sets can be used for this brute force attack such as numeric, lower alphabet, lower alphabet with numeric, and mix alphabet. In some cases, it can be more effective since dictionary attacks use a list of words that only contains a portion of the total words [16].

D. Dictionary Attacks

In conducting a dictionary attack, the perpetrator uses a list of words, referred to as a dictionary that supposed to have the target's password. Some rules can be applied to the input words such as capitalization, numeric prefixes, and replacing character 'a' with '@'. The effectiveness of the dictionary attack relies on the finite words it contains and the rules which are incorporated. A large dictionary may increase the chance of guessing the target password, but this can take more time to process [16].

E. Digital Forensics Model

The aim of computer forensics is to use tools and techniques to gather the evidences from a computer that are related to the crime. The age of digitization continues to grow and cybercrimes that are carried out are not any more limited to computers, rather, they are also prevalent to digital devices. A model is suggested [9] for digital forensic that includes nine components, namely identification, preparation, approach strategy, preservation, collection, examination, analysis, presentation, and the return of evidences. This model creates a standardized structure for the development of digital forensics as well as it can be used by judicial members to relate technology to non-technical individuals. However, the authors also state that the model is too general and there is no simple way to test the effectiveness of their method.

F. Taxonomy of Digital Forensics

Password guessing attacks on websites can be traced using Web Application forensics (WAF). It focuses on tracing back to the origin of the perpetration using the footprints left by the attacker. Web Application Forensics can be conducted through both server-side and client-side depending on the nature of the case. In addition, it is highly reliant on the log files from different sources such as web browser, web server, and database [7]. Web Application Forensics is also related to other forensic investigation areas as shown on Figure 1. As an example, WSF can also go hand-in-hand with Network Forensic since network level equipment logs may also be examined such as IP Routers, Intrusion Detection System, and Firewalls. A taxonomy is proposed [7] wherein WAF and Web Services Forensics (WSF) can be classified under Cloud-Computing Forensics (CCF). Additionally, Operating System

Forensics, Digital Image Forensics, Cloud-Computing Forensics, and Network Forensics fall under Digital Forensics. On the other hand, Web Services Forensics differs from Web Application Forensics since it focuses more with the security attacks on web services.

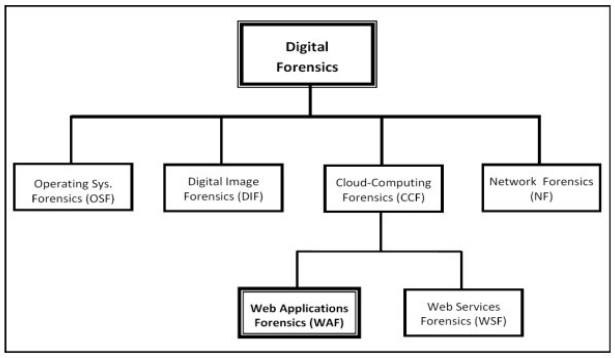

Fig. 1. Digital Forensics Taxonomy [7]

G. Web Application Forensics

The primary aim of Web Application Forensics is to trace a security attack on a web application back to the perpetrator who carried out the cybercrime [7]. However, one important consideration is that there is a higher chance that the compromised system should remain up and running since it is a live website with active users that always access it. Though, it still depends on the nature of the business and the case in deciding whether a necessary downtime for investigation should occur [17].

In addition, a conventional web application system involves four components which are the web server, application server, database server, and web application itself. The web server processes requests and serves the web pages on the user's browser through HTTP while the web application server is where the web application resides, which processes the functions of the web page that are related with the business logic and it also access the database server to read and write information. These four components can be isolated in four different physical servers or can also be in parallel with each other in a single machine [17].

H. Requirements

According to [7], a web application system should be forensically prepared by being capable of evidence collection and evidence protection. For evidence collection, the logs should be enabled and configured properly on the servers. The evidence protection requires that the integrity of these logs should be maintained by setting proper permissions on the log files and keeping them out of reach of the perpetrators. A forensic investigator should also be well equipped in terms of skills to carry out the investigation. The sufficient comprehension on the web application architecture, program flow, functions, and the server components are also important for the forensic investigator. The diversity of the web applications in terms of which type of architecture is used contributes to its high skill requirement for the forensic investigator. The experiment includes a Wordpress application on top of a LAMP stack and this creates a need for the investigator to have knowledge on Apache servers, PHP, Linux, etc.

I. Methodology

A proposed methodology [7] states that:

(a) Web Application should always be protected during a forensic investigation from any manipulation or corruption of data. This also includes the servers where the web application is running.

(b) The forensic investigator should discover all the required files for forensic investigation. These files can be the server logs, server side scripts that are related to web application, server configuration files, third part installed application logs, and operating system logs.

(c) The collected data should be analysed using different forensic tools to recreate the details of the cyber attack.

(d) After the event is successfully recreated, the investigator can then summarize the findings, and make a log of all the data and files extracted from the web application.

III. Experiment

A. Experimental Setup

The attack is carried out in a closed environment using a local web server to host the target web application. The LAMP stack that runs on the server includes Ubuntu Linux 15.04 (x64), Apache HTTP Server 2.4.12, MySQL 5.6.24, and PHP 5.5.24. This web server is targeted by a penetration testing platform - Kali Linux (x64). The IP addresses are in the private IPv4 range - 192.168.1.6 and 192.168.1.10 for the web server and penetration testing system, respectively.

B. Wordpress Application

Wordpress is an open source content management system (CMS) which is primarily used for creating web sites and blogs. Statistics indicate that 4.5% of all websites globally are using Wordpress [4] and this contributes to why it is selected for this demonstration. Wordpress 4.2.2 is installed on the web server with the access uniform resource locator (URL) of http://127.0.0.1/wordpress.

C. Persona

For the purposes of this experiment, a persona with the name of Peter Quill is created. He is currently 32 years old and works as an IT consultant. He likes to listen to music while driving, specifically 1960s - 1970s rock genre. Peter decides to publish his own blog site online using Wordpress since he wants to express his insight on his favourite songs.

D. Passwords

Different sets of passwords are used to have a comparative results based on the success and the process time of the dictionary attack. These passwords are based from the

recommendations of the related publications. In relevance with the persona, the password is based on "I Want You Back" song of *The Jackson 5*.

TABLE I. PASSWORDS BASED FROM THE PROPOSED TECHNIQUES OF THE RELATED PUBLICATIONS

Password	Type
jackson	Default [1]
123jackson45	CFG [14]
iwantyoujackson5	basic16 [5]
IWtYBk!!	mnemonic [6]

E. Dictionary Attack

A dictionary is carried out using three widely used dictionaries namely John the ripper (10,934 bytes) and Cain and Abel (1,069,968 bytes) [11]. This attack is performed using the WPScan application on Kali Linux that is specifically designed to penetrate Wordpress. Moreover, the penetration is initiated by retrieving usernames and Wordpress vulnerability information prior to password cracking. The target username is identified through the name of the persona, Peter Quill, which is provided by the WPScan enumeration function. This username is then supplied to the password cracking parameter with the dictionary list file name.

F. Logging

The Apache server, by default, is generating access logs on an *access logs.txt* file which can be used to trace back the attack. Different parameters from the access logs can be useful, such as the IP Address of the source, time stamp, HTTP request method, and the access URL. After the dictionary attack is performed, the access logs are checked. In addition, a Wordpress plugin called *WP Security Audit Log* is installed in Wordpress application to provide information regarding the login attempts as well as the username being used. In addition, this plugin can also provide details on the activities of a specific user on the Wordpress website.

G. Results

The attack is initiated by identifying the target username prior to the password cracking attack. This is done by using the *wpscan* command found in Figure 2.

```
root@kali:~# wpscan --url http://192.168.1.6/wordpress --enumerate user
```

Fig. 2. WPScan command

```
[+] Enumerating usernames ...
[+] Identified the following 6 user/s:
    +----+----------------+--------------------+
    | Id | Login          | Name               |
    +----+----------------+--------------------+
    | 1  | user           | Peter Peanutbutter |
    | 2  | admin          | Franco Sioquim     |
    | 3  | pquill         | Peter Quill        |
    | 4  | pquill_cfg     | Peter Quill        |
    | 5  | pquill_basic16 | Peter Quill        |
    | 6  | pquill_mnemonic| Peter Quill        |
    +----+----------------+--------------------+

[+] Finished: Sat Jul 18 02:52:09 2015
[+] Memory used: 2.555 MB
[+] Elapsed time: 00:00:03
root@kali:~#
```

Fig. 3. WPScan command to display all the usernames on the Wordpress website

```
root@kali:~# wpscan --url http://192.168.1.6/wordpress --wordlist /root/john.txt --username pquill
```

Fig. 4. WPScan command for dictionary attack using John the Ripper word list

Similarly Figure 3 lists all the usernames available on a Wordpress application, including the administrator account. The usernames are found under the Login column wherein *pquill* is the user with the default password found in Table 1 and the remaining Peter Quill usernames follow the *pquill* password type convention. Furthermore, the ID column is the actual user ID that is used by the Wordpress database.

TABLE II. RESULTS OF DICTIONARY ATTACK USING "JOHN THE RIPPER (JTR)" AND "CAIN AND ABLE (CAA)"

Dictionary	Username	Time	Status
JTR	Pquill	00:01:45	Success
JTR	pquill cfg	00:04:10	Fail
JTR	pquill basic16	00:04:10	Fail
JTR	pquill mnemonic	00:04:11	Fail
CAA	Pquill	03:18:41	Success
CAA	pquill cfg	06:38:55	Fail
CAA	pquill basic16	06:39:00	Fail
CAA	pquill mnemonic	06:39:16	Fail

An example command in Figure 4 is used to execute the dictionary attack on user *pquill* using John the Ripper dictionary. With the knowledge of the usernames used by the persona, remaining attacks are performed using the two dictionaries on different password types as shown in Table 2. It indicates that all the default alphabet password which is "jackson" can be cracked by both dictionaries. On the other hand, the dictionary attacks fail on the other password types that implements password strengthening techniques.

Both the Apache access logs and Wordpress logs from the plugin are checked after performing the dictionary attacks. Figure 5 is the Apache access logs that shows the multiple POST requests sent to the "*wp-login.php*" file. The parameters that are included on the log are client IP address, server timestamp, HTTP request, status code, and the size of the response to the client. The logs from *WP Security Audit Log* in Figure 5 also provide other information, such as the username and the login status that are not present in the Apache access logs.

IV. FORENSIC INVESTIGATION

This section discusses on the techniques that can be used for the forensic investigation on the experiment, and focusing on the log files. In analysing the log files, the following patterns can be indicators of a suspicious activity:

- relatively long HTTP requests,
- files that are located outside of the web server root directory,
- files that are located outside the web application directory, and

- unusual usage of HTTP methods such as HEAD, PUT and PROPFIND.

A. Web Server

a) Apache Logs

For Apache logs, the configuration can be found in the *http.conf* file. It can be used to change the default location and the formatting of the log files. Similarly, it can also point to the actual location of the logs. There are three kinds of logs that can be retrieved for forensic analysis, which are the error logs, module logs, and access logs. The error logs record all error messages on the Apache server. The module logs are records that are specific to the modules installed on the Apache server. On the other hand, the access logs on Figure 5 contains HTTP requests that are going to the server, which also means that brute force and dictionary attacks can be traced using it. Figure 5 illustrates a sequence of multiple POST requests on a Wordpress login file. These requests have very short intervals which contribute to its suspicious characteristics [17].

```
192.168.1.10 - - [18/Jul/2015:18:23:00 +1200] "POST /wordpress/wp-login.php HTTP/1.1" 200 3730
192.168.1.10 - - [18/Jul/2015:18:23:00 +1200] "POST /wordpress/wp-login.php HTTP/1.1" 200 3730
192.168.1.10 - - [18/Jul/2015:18:23:00 +1200] "POST /wordpress/wp-login.php HTTP/1.1" 200 3730
192.168.1.10 - - [18/Jul/2015:18:23:00 +1200] "POST /wordpress/wp-login.php HTTP/1.1" 200 3730
192.168.1.10 - - [18/Jul/2015:18:23:00 +1200] "POST /wordpress/wp-login.php HTTP/1.1" 200 3730
192.168.1.10 - - [18/Jul/2015:18:23:00 +1200] "POST /wordpress/wp-login.php HTTP/1.1" 200 3730
192.168.1.10 - - [18/Jul/2015:18:23:00 +1200] "POST /wordpress/wp-login.php HTTP/1.1" 200 3730
192.168.1.10 - - [18/Jul/2015:18:23:01 +1200] "POST /wordpress/wp-login.php HTTP/1.1" 200 3730
192.168.1.10 - - [18/Jul/2015:18:23:01 +1200] "POST /wordpress/wp-login.php HTTP/1.1" 200 3730
192.168.1.10 - - [18/Jul/2015:18:23:00 +1200] "POST /wordpress/wp-login.php HTTP/1.1" 200 3730
192.168.1.10 - - [18/Jul/2015:18:23:00 +1200] "POST /wordpress/wp-login.php HTTP/1.1" 200 3730
192.168.1.10 - - [18/Jul/2015:18:23:00 +1200] "POST /wordpress/wp-login.php HTTP/1.1" 200 3730
192.168.1.10 - - [18/Jul/2015:18:23:01 +1200] "POST /wordpress/wp-login.php HTTP/1.1" 200 3730
192.168.1.10 - - [18/Jul/2015:18:23:01 +1200] "POST /wordpress/wp-login.php HTTP/1.1" 200 3730
192.168.1.10 - - [18/Jul/2015:18:23:01 +1200] "POST /wordpress/wp-login.php HTTP/1.1" 200 3730
192.168.1.10 - - [18/Jul/2015:18:23:01 +1200] "POST /wordpress/wp-login.php HTTP/1.1" 200 3730
192.168.1.10 - - [18/Jul/2015:18:23:01 +1200] "POST /wordpress/wp-login.php HTTP/1.1" 200 3730
192.168.1.10 - - [18/Jul/2015:18:23:01 +1200] "POST /wordpress/wp-login.php HTTP/1.1" 200 3730
192.168.1.10 - - [18/Jul/2015:18:23:01 +1200] "POST /wordpress/wp-login.php HTTP/1.1" 200 3730
192.168.1.10 - - [18/Jul/2015:18:23:01 +1200] "POST /wordpress/wp-login.php HTTP/1.1" 200 3730
192.168.1.10 - - [18/Jul/2015:18:23:01 +1200] "POST /wordpress/wp-login.php HTTP/1.1" 200 3730
```

Fig. 5. Apache acess logs after the Dictionary attack

B. Application Server

The application servers can include logging mechanism that records events such as application errors, loader problems, unhandled exceptions, and application messages.

a) PHP Logs

The PHP application includes error logs which are stored by default in the */var/log/httpd/php error log* directory. However, this can be configured using the *php.ini* file which is located on the */etc/php.ini* [10]. PHP crash messages and errors can be recorded in the error log. This is also used by the system administrators to monitor the application server for the errors that occur in the background while PHP is running. Another helpful log is the system log are the system logs which includes localised logging for a specific application. The files incorporated on the system log can be *err.log*, *warning.log*, *info.log*, and *debug.log*.

C. Database Server

Databases commonly have limited or even no logging enabled by default, which poses a constraint for forensic investigators. However, a table or data specific logging capabilities can be implemented using database triggers [17].

a) MySQL

MySQL stores five kinds of logs which are the error log, general query log, binary log, relay log, and slow query log. The error log contains details on the issues encountered during the start, run, and stop the operations of the MySQL database. The general query log includes the successful client connection and the SQL statements from the users. On the other hand, the binary log have the records the statements that updates the data and the relay log writes the changes on the data received from a replication server. Finally, the slow query long is useful for the forensic investigator since it includes queries that takes more time to execute, which can pertain to suspicious activities [8].

D. Web Application

Application logs can yield important information for forensic investigation, such as the detailed knowledge of the business logic. The forensic investigator should also seek assistance from developers and administrators who have knowledge on the web application. The information that can be retrieved are the location and format of the application logs, messages on the logs that can be considered as suspicious, and how long the logs are stored [17].

a) Wordpress logs

The Wordpress application can have debug logs when WP DEBUG is set to true. When enabled, all errors are stored in a debug log file within the */wp-content* directory. This can be helpful for the forensic investigator to see all the background notices generated from AJAX requests and *cron jobs* [18]. In addition, Wordpress plugins that offers logging capabilities can also be used to retrieve events within the application. Figure 6 shows an example of login events that are recorded using a Wordpress plugin.

1000	2015-07-18 06:55:52.461 PM	Peter Quill Administrator, Superadmin	192.168.1.10	Successfully logged in
1002	2015-07-18 06:54:26.390 PM	Peter Quill Administrator, Superadmin	192.168.1.10	10+ failed login(s) detected
1002	2015-07-18 06:54:23.843 PM	Peter Quill Administrator, Superadmin	192.168.1.10	10+ failed login(s) detected
1002	2015-07-18 06:54:22.435 PM	Peter Quill Administrator, Superadmin	192.168.1.10	10+ failed login(s) detected
1002	2015-07-18 06:54:15.887 PM	Peter Quill Administrator, Superadmin	192.168.1.10	1 failed login(s) detected

Fig. 6. Wordpress audit logs after the dictionary attack

V. COUNTERMEASURES

The following countermeasures can be applied to mitigate the demonstrated dictionary attack on Wordpress. In some cases, these countermeasures are also applicable for other types of web applications.

A. Lockout system

There can be multiple solutions to protect a website from password cracking attacks. A lockout system allows a fixed

number of password entry mistakes before the website login is locked for a period of time or even permanently [16].

The attacker can only have limited amount of guesses before the system denies his/her login. Furthermore, several existing plugins are available to equip Wordpress with this functionality such as *WordFence*, *Bullet Proof Security*, and *All-in-One WP Security and Firewall*. Some plugins also allow the authorized user to set the maximum failed login attempts before denying access.

B. Multi-factor Authentication

Multi-factor authentication is also another feature that is also offered by Wordpress plugins such as *WordFence* and iThemes Security (formerly Better WP Security). *WordFence* enables Wordpress users to have a two-factor authentication through SMS. This countermeasure is highly effective against dictionary and brute force attacks because in addition to username and password requirement, it adds another element in the authentication which is a code that is only visible to the user's device where the authentication SMS is sent.

C. Strong Passwords

Strengthening the passwords is the fastest countermeasure against password cracking attacks. This solely relies on the user and his/her knowledge of the techniques to attain a highly difficult to crack passwords. Some techniques are used for the dictionary attack demonstration namely CFG, basic16, and mnemonic. These techniques are proven to be effective since the strengthened passwords don't exist in the two dictionaries used in the demonstration. Furthermore, using a longer password approach is recommended such as the basic16 policy. Not only it increases the complexity of the password cracking attack, it also requires a dictionary that contains more than 16 characters which takes a relatively longer time to execute.

D. Associated Names

The demonstration shows that WPScan is able to extract both username and full name associated with the account (Figure 3). Full names can be used by the perpetrator by conducting web searches to gain more information about a specific user. These information can be used to construct a wordlist dedicated only for a specific account. Thus, not using real names on the Wordpress account can reduce the chance of the attacker in acquiring more information about the user.

VI. DISCUSSION

The dictionary attack demonstrates the vulnerability of a fresh Wordpress installation. The usernames can easily be enumerated using the WPScan application. In addition, actual first name and last name are also disclosed in which this information can be used by the perpetrator for a more sophisticated password cracking attack. The John the Ripper and Cain and Abel dictionaries are used to crack the different types of passwords based on the password strengthening technique. The results indicate that the two well-known dictionaries fail to crack the strengthened password, however, they are able to identify the default password which is a lower case alphabet string (”jackson”).

There are other well-known dictionaries that contain bigger word lists and some focus more on specific genres [11]. One factor that needs to be considered is the time it takes to perform the attack. A larger size dictionary can take more time as shown in the demonstration wherein Cain and Abel took almost seven hours to complete while John the Ripper finishes in about four minutes. Well-known dictionaries are useful if the perpetrator have no personal information about the target user. However, if background knowledge exists, some techniques can be applicable such as applying rules to the dictionary or using a word list that are generated from music category. Brute force attack can also be carried out using the phrases ”I Want You Back” and ”Jackson 5” on the word list with a minimum character of six based on the Wordpress password policy.

The demonstration also describes how to trace the attack by analysing the Apache access logs and the logs from *WP Security Audit Log*. The Apache access logs reveal that there are POST requests in the *wordpress/wp-login.php* path which signifies a Wordpress login activity. However, this is considered to be an anomaly because of the consecutive requests with an interval of less than a second. Another factor that contributes to the abnormal login behaviour is that these request are coming from a single client IP address. In addition to the Apache access logs, other necessary information to describe the attack is retrieved from the install Wordpress plugin to generate authentication logs. It shows that the failed login attempts are targeting the user “Peter Quill” that occur in the similar time frame of the consecutive login requests on the Apache access logs.

VII. CONCLUSION

Website authentication is susceptible to different forms of password guessing attacks such as dictionary and brute force. An attack is simulated to further explain how a dictionary attack is carried out to penetrate the Wordpress authentication. The attacker needs to identify the username before attempting to guess the password. Moreover, acquiring personal information may also be helpful for an effective password guessing. The demonstrated dictionary attack proves that using a lower case alphabet password with only seven characters can easily be cracked by well-known dictionaries that are available online. On the other hand, applying password strengthening techniques proposed by the reviewed publications can effectively mitigate the attack. In addition, increasing the number of characters used in the password can be very difficult and time consuming to crack. There are also countermeasures that can be applied on Wordpress such as lockout system, multi-factor authentication, strong passwords, and by avoiding the use of real names. These countermeasures are easily available on Wordpress plugins such as WordFence. Access logs can also be useful in performing forensics because it contains useful details such as timestamp, source IP address of the attack, target username and number of attempts. The comparison of the effectiveness for each password

strengthening technique is not highlighted in this report which can also be a basis for future work. Furthermore, this report only simulates a simple dictionary attack without any specific rules applied. Additional work can be done by attempting more sophisticated dictionary attacks with a larger word list and also a brute force attack using the known phrase of the persona.

To have an effective forensic investigation on the case of a dictionary attack against the Wordpress web application, log files on the server should be properly configured and kept protected. The forensic investigator should also have sufficient skills to conduct the investigation as well as knowledge on the architecture and business logic of the web application. Web forensic mostly focuses on retrieving and analysing logs from web server, application server, database server, and Wordpress application.

REFERENCES

[1] A. Das, J. Bonneau, M. Caesar, N. Borisov and X. Wang, 'The Tangled Web of Password Reuse', *Symposium on Network and Distributed System Security (NDSS),* 2014.

[2] D. Gillman, Y. Lin, B. Maggs and R. K. Sitaraman, 'Protecting Websites from Attack with Secure Delivery Networks', *Computer,* vol. 48, no. 4, pp. 26 - 34, 2015.

[3] J. A. Halderman, B. Waters and E. W. Felten, 'A convenient method for securely managing passwords', *In Proceedings of the 14th international conference on World Wide Web,* pp. 471-479, 2005.

[4] K. K, '[Infographic] WordPress 4.1 Gets Roughly 1 Million Downloads Every Two Days + Other Mesmerizing WordPress Stats', 2015. [Online]. Available: http://www.codeinwp.com/blog/mesmerizing-wordpress-stats/.

[5] P. G. Kelley, S. Komandur, . M. L. Mazurek, . R. Shay, . T. V. L. Bauer, N. Christin, L. F. Cranor and J. Lope, 'Guess Again (and Again and Again): Measuring Password Strength by Simulating Password-Cracking Algorithms', in *Security and Privacy (SP), 2012 IEEE Symposium on*, San Francisco, CA, 2012.

[6] C. Kuo, S. Romanosky and L. F. Cranor, 'Human selection of mnemonic phrase-based passwords', *In Proceedings of the second symposium on Usable privacy and security,* pp. 67-78, 2006.

[7] A. Lazzez and T. Slimani, 'Forensics Investigation of Web Application Security Attacks', *International Journal of Computer Network and Information Security (IJCNIS),* vol. 7, no. 3, p. 10, 2015.

[8] MySQL, '5.2 MySQL Server Logs', 2015. [Online]. Available: https://dev.mysql.com/doc/refman/5.0/en/server-logs.html.

[9] M. Reith, C. Carr and G. Gunsch, 'An examination of digital forensic models', *International Journal of Digital Evidence,* vol. 1, no. 3, p. 1–12, 2002.

[10] I. Sanosyan, 'Getting the Most Out of Your PHP Log Files: A Practical Guide', 2015. [Online]. Available: http://www.toptal.com/php/getting-the-most-out-of-your-log-files-a-practical-guide.

[11] Skullsecurity, 'Passwords', 2015. [Online]. Available: https://wiki.skullsecurity.org/Passwords.

[12] D. Silver, S. Jana, D. Boneh, E. Chen and C. Jackson, 'Password Managers: Attacks and Defenses', *In Proceedings of the 23rd Usenix Security Symposium,* 2014.

[13] M. Taylor, J. Haggerty, D. Gresty and D. Lamb, 'Forensics investigation challenges in cloud computing environments', *Network Security,* vol. 2011, no. 3, p. 4–10, 2011.

[14] S. Vaithyasubramanian and A. Christy, 'An Anaysis of CFG Password Against Brute Force Attack for Web Applications', *Contemporary Engineeri,* vol. 8, no. 9, pp. 367 - 374, 2015.

[15] A. G. Voyiatzis, C. A. Fidas, D. N. Serpanos and N. M. Avouris, 'An Empirical Study on the Web Password Strength in Greece', in *Informatics (PCI), 2011 15th Panhellenic Conference on*, Kastonia, 2011.

[16] C. M. Weir, 'Using probabilistic techniques to aid in password cracking attacks', *Electronic Theses, Treatises and Dissertations*, Tallahassee, Florida, 2010.

[17] C. Willis and R. Belani, 'Web Application Incident Response & Forensics', *Black Hat Briefings USA 2006,* 2006.

[18] Wordpress, 'Debugging in WordPress', 2015. [Online]. Available: https://codex.wordpress.org/Debugging_in_WordPress.