OPEN SOURCE SERVERS AND WEBSITE PLATFORMS SECURITY

Gabriel Eugen Garais [1*]
Alexandru-Serban Enaceanu [2]

ABSTRACT

Open Source Operating Systems for Servers, together with Website Projects are widely used by network administrators, web developers and web users. Free code access to Operating Systems core files and easy installation and configuration of Open Source Web Site Platforms is a handy but also a risky solution. The use of such platforms is under heavy discussion because the Open Source code is visible to all the potential hackers. This article focuses on the major Server and Open Source Web Platforms attacks and security flaws.

KEYWORDS: *Open Source Websites, Open Source Projects, Open Source Servers, Security, Firewall*

INTRODUCTION

Open Source Operating Sources are widely used among ISPs and companies due to their cost and scalability. However, even experienced network administrators have difficulties maintaining a high security policy on their servers because of the many types of attacks that can appear.

There are many Linux distributions or flavors that can be suitable as web servers, email servers or that can provide other user services.

For example, the CentOS Project is a community-driven free software effort focused on delivering a robust open source ecosystem. For users, it offers a consistent manageable platform that suits a wide variety of deployments. For open source communities, CentOS offers a solid, predictable base to build upon, along with extensive resources to build, test, release, and maintain their code.

Open Source Website Projects are free for download and use, represent a common use for inexperienced users but also for professional web developers. These projects governed by Free Software Foundation are collaborative projects. A standard structure and database design is implemented within these projects. The main standard structure of rules is the core.

This article will address the Open Source CMS Platform called WordPress as this platform is one of the most used free one used by developers.

[1*] corresponding author, Lecturer PhD, Romanian-American University, Bucharest, garais.gabriel.eugen@profesor.rau.ro
2 Lecturer PhD, Romanian-American University, Bucharest, alexandru.enaceanu@profesor.rau.ro

THREAT ON SECURITY SOURCES AND REASONS

The security issue is a very important aspect of Open Source platforms. The identified hacks on these platforms, are in most of the cases automated remotely scheduled applications that relay on the structure of the core platform and search for *open doors* to hack the system. Issues will be addressed in the rest of this article.

The most common identified threat is *brute force attack.* While *brute force* attacks on the server can be restricted by firewall or by configuration of the service (sshd, ftpd, smtp), the hacker gains access to the administration part of the platform by using the *login forms*. Also, another type of attack is the *Denial of Service (DOS)* attack. DOS attack will quickly fill up the server's resources making it unable to respond to legitimate requests.

A distributed denial-of-service (DDoS) attack is a coordinated strike, distributed among different computers, that aims to prevent the authorized use of one or more systems. These Web server DDoS have become a weapon of choice for attackers.

On the other hand, the hack application used to attack Website Platforms, will try to identify the administrator credentials by testing dictionary-based list of usernames and passwords. The main security risk behind this kind of attack comes from Open Source core structure which allows the hacker to discover the vulnerabilities.

Unfortunately, the default and most used username for administration is *admin* so this is the most used username for brute force attacks. Chart presented in figure 1 [1] shows the daily massive brute force attacks during April 2016 on Websites using WordPress.

The numbers were released by the company that developed one of the security plugin for WordPress called *SUCURI*.[1][2]

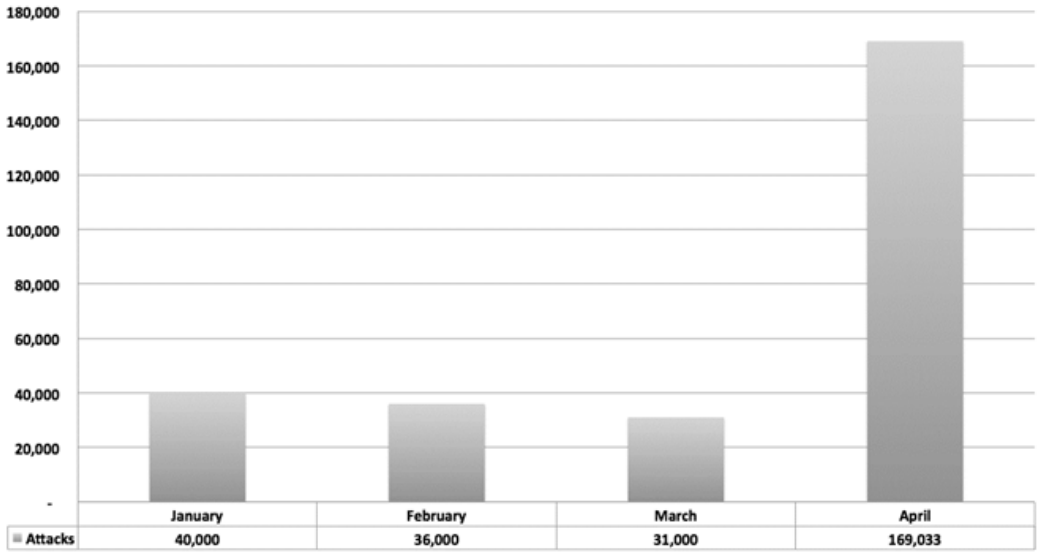

Figure 1 – Daily average number of Brute force attacks on WordPress [1]

The chart in figure 2 [2] presents the number of attacks by usernames during the time span between January and April 2016.

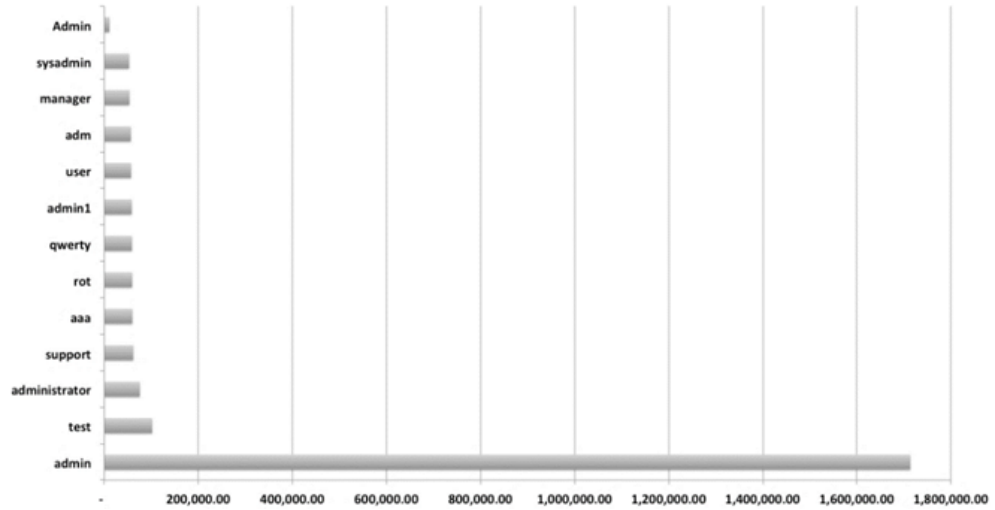

Figure 2 – Username attack distribution [2]

Open Source platform features can be extended by installing plugins, modules and themes. They are made of additional scripts but also with database design adjustments. These added components to the platform, increase the risk of security caveats. Unprotected forms from within the plugins can lead to *SQL Injections or virus code uploads* in the core files structure of the platform.

An interesting graphic [3] presented by the programmers of the *Wordfence* plugin, an important security plugin for WordPress, shows the main reasons of attacking a website. That example suggests that not the website or website information is the primary target, as a defaced or compromised site will be the source of other automatized hacks towards other targeted sites or services.

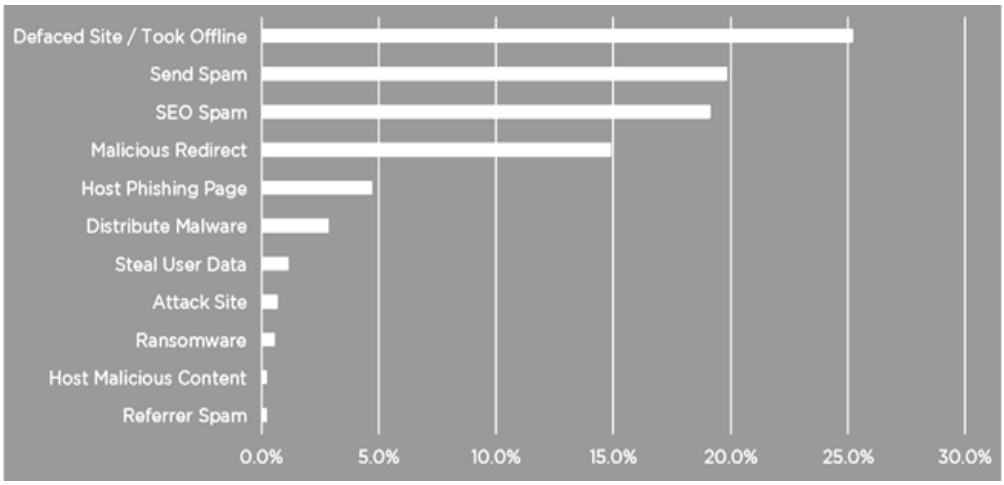

Figure 3 – Purpose for hacking a website by hackers [3]

Although the same techniques were applied to hack the above sites, the targeted actions were different as presented in figure 3.

In order to properly secure a system, developers have to understand the patterns and reasons of hacking so that proper security measures can be implemented.

According to [3], a *defaced site* is the one that after it has been compromised, the content is replaced with new content serving different tasks.

Mail queue investigation is the first sign that a website has been compromised and that is *sending* spam. Long waiting list of emails that can no longer be sent because of the defers transmitted back by the targeted email servers. The immediate result is that the owner's IP and domain will be blacklisted on public Blacklist Servers – RBLs.

Malware scripts hosting is another way of targeting the hacked website. All traffic or partially so that the hacking will be discovered much slower is diverted to the hacker's websites. Malware scripts will eventually be detected by search engines and the website will be tagged as dangerous and a message similar to figure 4 will be displayed.

Figure 4 – Malware warning

SECURITY TIPS FOR DEFENDING OPEN SOURCE SERVERS

SSH IP addresses restrictions, by the use of firewall is a well known practice and together with denying login to the root user provides a basic remote login security policy. The SSH session will be started by logging on with a regular user account, and then escalating (by su or sudo) to root account. SSH service should also run on a different port, rather than the default, but within the 0-1023 port range (privileged ports) to make sure the port is opened by a privileged account. The port range should be the one above mentioned and

under no circumstances the port should be above 1023, to prevent the port being opened by any other user except root.

Ports above 1023 can be opened without the need of a privileged account, meaning that someone can write a script to listen on the SSH port and capture your user and password. This script can easily be done with simple tools commonly available on every Linux server, for example in bash. So running SSH on a non-privileged port is LESS secure, not MORE. There is no way of knowing if your SSH client is communicating to the real SSH server or not. This reason, proves that you should never use a non-privileged port while configuring a SSH server.

SSH daemon configuration file *(/etc/ssh/sshd_config)* should reflect the fact that root user is not allowed to login and also the new port::

PermitRootLogin yes
Port 222

An example of firewall implementation using iptables, to restrict the source IP to the SSH service on the port above mentioned will be:

${IPTABLES} -A INPUT -p tcp --dport 222 –s {trusted_ip} –j ACCEPT
${IPTABLES} -A INPUT -p tcp --dport 222 –j DROP

If you really need SSH access and do not have access to a static IP address, then a VPN solution is needed. VPN servers like OpenVPN [11] will do perfectly in such cases. OpenVPN is a full-featured SSL VPN which implements OSI layer 2 or 3 secure network extension using the industry standard SSL/TLS protocol, supports flexible client authentication methods based on certificates, smart cards, and/or username/password credentials, and allows user or group-specific access control policies using firewall rules applied to the VPN virtual interface.

After installing and configuring OpenVPN you should limit the {trusted_ip} address mentioned above to the IP subclass that the OpenVPN server serves the clients with.

But if you are unable to install a VPN server, then you should at least try the port knocking method which will open the port 222 for a certain number of seconds only if another port or sequence of ports will be knocked before.

There are several ways to implement port-knocking. There are 3rd party tools, which could be way too complex [12] or are implemented in userland so it’s better not to use them. Instead, you can do this with simply using iptables rules, which has got a very useful module called “recent”, which allows you to create simple - yet effective - port knocking sequences, as in the following example:

${IPTABLES} -A INPUT -p tcp --dport {port_to_be_knocked} -m recent --set --name portknock
${IPTABLES} -A INPUT -p tcp --syn --dport 222 -m recent --rcheck --seconds 30 --name portknock -j ACCEPT
${IPTABLES} -A INPUT -p tcp --syn --dport 222 -j DENY

This method should be applied to the rest of services that requires user authentication as well (for example FTP). Also, acces on FTP servers should be configured to allow access only to the user's home directory (chroot ~Home).

To be even more cautious FTP should be replaced by an encrypted file transfer portocol like SFTP or SCP. The SSH File Transfer Protocol (SFTP), also known as the Secure File Transfer Protocol, enables enctypted, secure file transfer between networked hosts. Unlike the Secure Copy Protocol (SCP), SFTP additionally provides remote file system management functionality, allowing applications to resume interrupted file transfers, list the contents of remote directories, and delete remote files. As SFTP runs as a subsystem of SSH it runs on whatever port the SSH daemon is listening on and that is administrator configurable.

Command-line secure file transfer program (sftp) and graphical SFTP clients, such as WinSCP or SFTP plugin for Total Commander file manager, use SSH2 encryption to authenticate and establish secure channels between networked hosts. Although SFTP clients are functionally similar to FTP clients, they employ different protocols; consequently a standard FTP client cannot be used to connect to an SFTP server.

Denial-of-Service (DOS) attacks can be addressed by using a combination of the following techniques:

- Enable SYN COOKIES at the kernel level :

  echo 1 > /proc/sys/net/ipv4/tcp_syncookies

  SYN cookies are a technique used to block SYN flood attacks, by avoiding dropping connections when the SYN queue fills up. Instead, the server behaves as if the SYN queue had been enlarged and sends back the appropriate SYN+ACK response to the client while discarding the SYN queue entry. If the server then receives a subsequent ACK response from the client, the server is able to reconstruct the SYN queue entry using the information encoded in the TCP sequence number.

- Enable and Configure iptables rules to prevent the attack or at least to identify the attack

  ${IPTABLES} -N syn-flood
  ${IPTABLES} -A syn-flood -m limit --limit 100/second --limit-burst 140 -j RETURN
  ${IPTABLES} -A syn-flood -j LOG --log-prefix "SYN-flood: "
  ${IPTABLES} -A syn-flood -j DROP

- Install and configure a thid-party tool like Advanced Policy Firewall (APF firewall) and (D)DosDeflate which allows IP addresses with over a pre-configured number of connections are automatically blocked in the server's firewall (using iptables or APF).

The web server, which is the Apache webserver is usually the primary target of any DoS or DDoS attack. The mod_evasive server side Apache module, also known as mod_dosevasive, acts by protecting against DoS, DDoS (Distributed Denial of Service)

and brute force attacks. It can provide evasive actions during attacks and report abuses via email and syslog facilities.

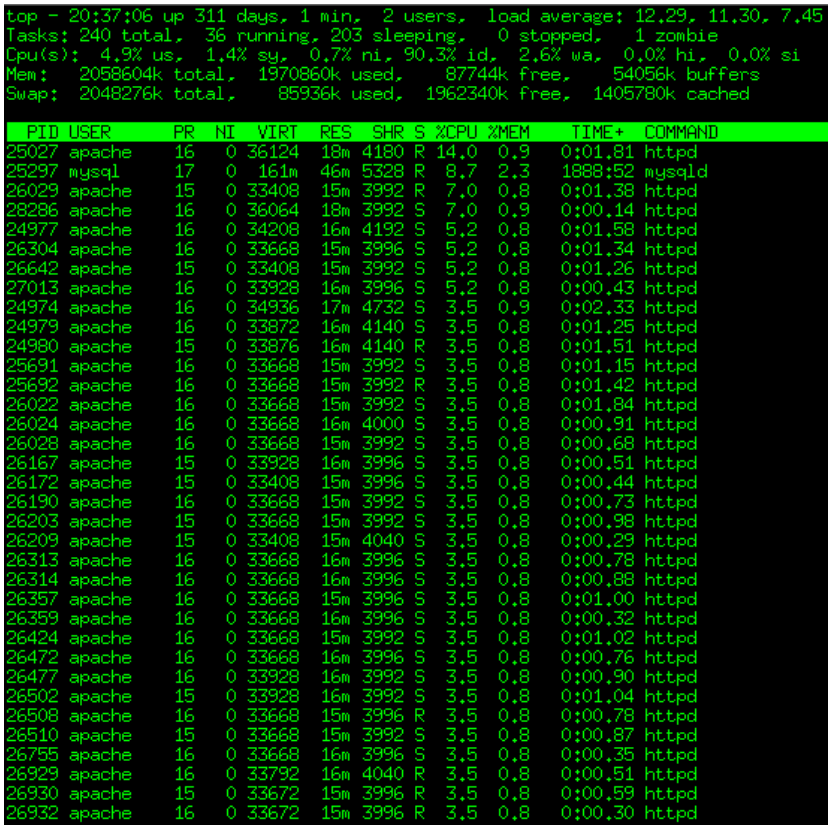

Figure 5 –DDoS attack detected on Apache webserver [14]

The module creates an internal dynamic table of IP addresses and URIs as well as denying any single IP address from any of the following:

- Requesting the same page more than a configurable times per second
- Making more than a preconfigured concurrent requests on the same child per second
- Making any requests while temporarily blacklisted

If any of the above conditions are met, a 403 response is sent and the IP address is logged. Optionally, an email notification can be sent to the server owner or a system command can be run to block the IP address. The *mod_evasive* plugin is available as a preconfigured package for almost any Linux based distribution like: CentOS, Debian, Redhat, Ubuntu, etc.. The installation is pretty forward and configuration files are self explanatory.

Unfortunately, Apache is vulnerable to another type of attack, the DNS Injection, which are attacks that inject fake DNS names into your server's cache. Spam from web forms is

a fast-track method of getting your domain or web server blacklisted by RBL (Real-time Blackhole List) services like Spamhaus [15].

In addition to other Apache2 modules for blocking DDos Attacks, the mod_spamhaus Apache module checks the incoming IP address against the Spamhaus database of blocked IP addresses.

SECURITY TIPS FOR DEFENDING OPEN SOURCE WEB SITE PLATFORMS

In the WordPress terminology for implementing security measures it is referred as *hardening* [4] the system.

The first step to hardening the online platform is to *use longer and complex passwords*, by using upper and lowercase letters, numbers and special signs.

The second step is to be careful about not keeping the core, but also additional features *up-to-date.*

The third step is to to *completely remove any unused functionalities* on the website to prevent having more vulnerable sections in the website.

The following step is to pay attention to file and directory permissions. Directories or files with 777 permissions are to be avoided. Instead *use permissions configurations of 755 or 750 at folders and 600, 640 or 644 for files*, according to the needs.

Login forms can be protected for nonhuman, also called *bots submitting's* by any form of *captcha,* therefore limiting the form submission only to human users and not automatic scripts. By *limiting the number of failed logins* the system must keep tracking the number of same usernames called by one single IP in a certain amount of time.

The *CloudFlare* application filters the entire traffic to the Website Platform by using it's own servers to intermediate all requests. It also delivers a cached but dynamically result in the same time to the end user. DDoS attacks which intermediated and filtered by the CloudFlare servers states [5] that the attackers used 4529 NTP servers and each used an average of 87Mbps of traffic to complete an over 400Gbps DDoS attack which is presented in the figure 6 chart.

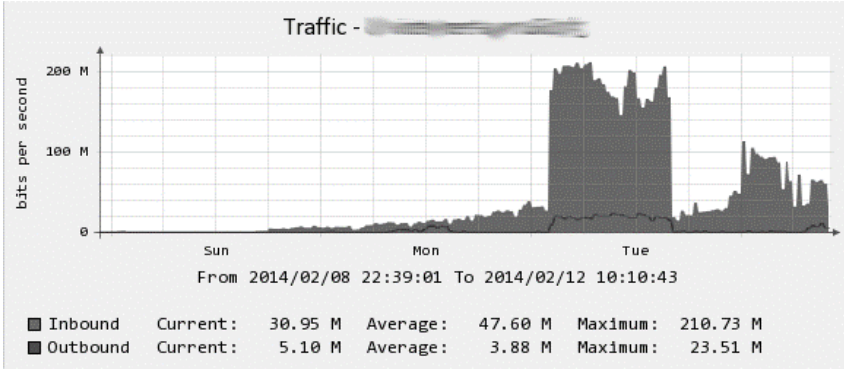

Figure 6 – 400Mbps DDoS attack detected and filtered through CloudFlare system [5]

The advantages of CloudFare powered systems starts with lowering the bandwidth and traffic requests that reaches the web site server and end by filtering the type of traffic which significantly can reduce the amount of attacks to the online website platform.

A *backup syste*m should also be implemented. The backups can be stored locally or remotely on a third party server. There are already a number of free plugins that provide backup directly in a Dropbox account, google drive account, SkyDrive account and so on. The recommended solution is still using a NAS server as the primary backup.

Installing security plugins into the website is the last measure but not the least. The security plugins must be kept up-to-date as well and also heavy tested before relying on them. Wordfence is one example of top rated security plugin for Wordpress based websites. It has over 1 million downloads and high rankings and references. A basic security plugin should be compatible with the core system and should ensure services like: scanning the entire website file system for suspicious scripts; limit the login failures, limit internal plugins to upload other file types as configured; block and log suspicious traffic; send notifications to the website administrator.

Even if the software that powers the server and web platforms are Open Source, in order to increase security measures on the servers, various commercial software products appeared on the market.

Such an example will be ConfigServer eXploit Scanner [13] which costs $60/server and provides active scanning for the following potential issues:

- all modified files within user accounts using the cxs Watch daemon regardless of how they were uploaded;
- PHP upload scripts (via a ModSecurity hook)
- Perl upload scripts (via a ModSecurity hook)
- CGI upload scripts (via a ModSecurity hook)
- Any other web script type that uses the HTML form ENCTYPE multipart/form-data (via a ModSecurity hook)
- Pure-ftpd (FTP server) uploads

The active scanning of files can help prevent exploitation of an account by malware by deleting or moving suspicious files to quarantine before they become active. It can also prevent the uploading of PHP and perl shell scripts, commonly used to launch more malicious attacks and for sending spam.

With over 4000 known current exploit script fingerprint matches and together with the Open Source ClamAV antivirus, ConfigServer eXploit Scanner is a must when it comes to increase server and websites the security measures.

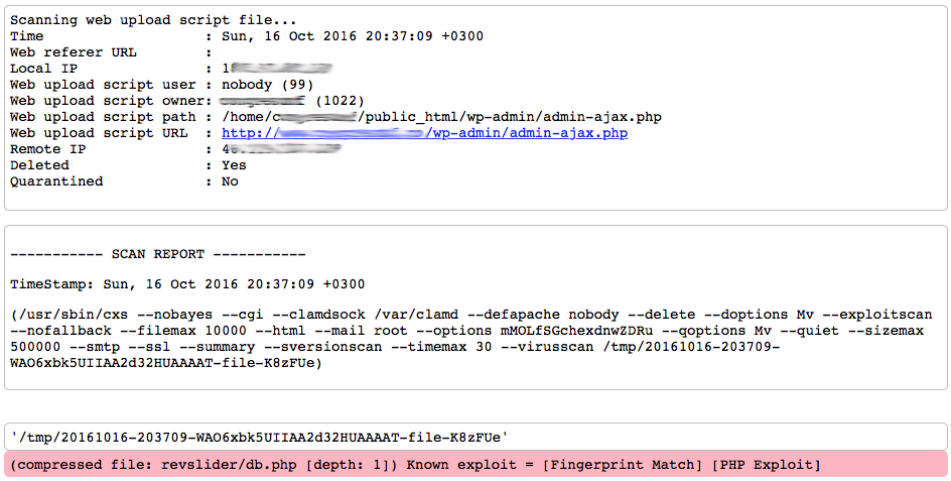

Figure 7 – Sample exploit detection by fingerprint match

The attacker tried to upload a virus by the post method, through wordpress' admin-ajax.php script and was recognized and deleted by ConfigServer eXploit Scanner.

CONCLUSION

Open Source servers and platforms are modern, easy and convenient ways of implementing software solutions for business. However, the risk that comes with those advantages is directly proportional. Fortunately, there are ways to combat the flaws, glitches and deficiencies of such platforms. By following the ideas presented in this article developers and network administrators will successfully succeed preventing attacks and downtimes of the systems.

BIBLIOGRAPHY:

[1] https://blog.sucuri.net/2013/04/the-wordpress-brute-force-attack-timeline.html/screen-shot-2013-04-16-at-11-24-04-am

[2] https://blog.sucuri.net/2013/04/the-wordpress-brute-force-attack-timeline.html/screen-shot-2013-04-16-at-11-15-35-am

[3] https://www.wordfence.com/blog/2016/04/hackers-compromised-wordpress-sites/

[4] http://codex.wordpress.org/Hardening_WordPress

[5] http://thehackernews.com/2014/02/NTP-Distributed-Denial-of-Service-DDoS-attack.html

[6] http://www.infoworld.com/article/2985242/linux/why-is-open-source-software-more-secure.html

[7] http://www.computerweekly.com/feature/Open-source-software-security

[8] https://premium.wpmudev.org/blog/wordpress-security-tips/

[9] http://codex.wordpress.org/Hardening_WordPress

[10] https://hostingfacts.com/how-to-secure-wordpress/

[11] https://openvpn.net

[12] http://www.portknocking.org

[13] http://configserver.com/cp/cxs.html

[14] https://systembash.com/how-to-stop-an-apache-ddos-attack-with-mod_evasive/

[15] https://www.spamhaus.org