Quality of WordPress Plug-Ins: An Overview of Security and User Ratings

WordPress started in 2003 (WordPress.org. (n.d.)). Today it is among one of the very few platforms where that provide for non-technical user to create their own website. Wordpress was is an open source project, and it leads to cost of product which is zero. It has a community, which consist of millions of contributor working to enhance existing features and to create their own plugins. Plugins are the extended features that makes any particular task to be more flexible and lightweight code. Wordpress is heavily used in blogging, website creating and other web applications. It welcome changes to website without programming skills required (Koskinen, Karavirta, & Ihantola, 2012, pg. 1). Currently more than 2.5 billion pages are visited by approximately 300 million users each month (Koskinen et al. (2012), pg. 1). This traffic of user is main cause of large community working for updating and creating new plugins. Anyone with technical knowledge can create a plugin, which enables to create plugins with good and bad intentions. Web security is a topic of discussion among web developer for data security and privacy. People using Wordpress as their platform uses many kind of online free plugin which include basic management widgets like calendar, recent post, video player etc. However, it also includes some critical plugins like payment or transactions (Koskinen et al. (2012), pg. 1). There can be many types of vulnerabilities which can result malfunction of a plugin and to ensure its safety, Wordpress checks them annually for security (Koskinen et al. (2012), pg. 1). To make searching better plugins easily, WordPress have

introduced user rating which enable more secure and effective plugins to be first option for user (Koskinen et al. (2012), pg. 1).

In our current project of a Wordpress website, for South End Daycare, where selection of a plugin which need to access database in order to extract upcoming events and archive news of day care would be kept. In addition, there would be a whole waiting list that can be managed, and each user have access to their waiting list number. There is currently no plugin available due to which a new plugin would be programmed. However, the vulnerability of that plugin is needed to be assessed. If the plugin is exploited, then penetrator can have direct access to database. A video player requirement can also be fulfilled using plugins like youtube or venom. In case of it is decided to use youtube, a video is directly access from youtube server. Which have no direct access to database. In result it can effect crashing the plugin and pausing video. It will have

User feedback plays an important role when shopping is done online, where user share their feedback about product and its efficiency. There can be many ways to provide feedback, like description or marks, and stars which is another way of giving marks with minimum description. However, a study was made in 2006 by Chevalier J. and Mayzlin D. on impact of feedback on sale, and it was determined that verbal feedback have more impact on sale than star (or marks) feedback does (Koskinen et al. (2012), pg. 2). There are different kind of threats or possible vulnerabilities in plugins, a project called Open Web Application Security Project (OWASP) classified them in six major categories

(Koskinen et al. (2012), pg. 1). These vulnerabilities include SQL injection, remote code execution, PHP configuration and system attack (Koskinen et al. (2012), pg. 1). Among these XXS and SQL injection are the most common vulnerabilities in recent years. SQL injection is performed by detail examine process of any plugin or functionality and later on injecting it through URL to manipulate the database and its records (Koskinen et al. (2012), pg. 2). Detecting the possible vulnerabilities in any plugin can be done by statistically and by using different automated programs. However, checking statistically is a lengthy but gives more accurate results Koskinen et al. (2012), pg. 2).

For the determination of how a plugin can be vulnerable there are two main approaches, testing and statistical. Using testing the web service is accessed using HTTP protocol interface similar to what user do because it is assumed that attacker is outsider and have minimal knowledge about code and its vulnerability. The failure rate of detecting of test is approach is higher, nearly 60%-90% (Koskinen et al. (2012), pg. 2). On the other hand, for statistical analysis the knowledge of backend development and basic network and code is necessary (Koskinen et al. (2012), pg. 2). Using this knowledge helps to determine any loop hole that can be exploited. There are few commercial analyzers like Fortify Source Code Analyzer, and open source RATs and RIP which can statistically analyze PHP code and plugin. A detail statistical study was done by Koskinen, Karavirta, & Ihantola in 2012 in which nearly 322 sample plugins were downloaded, out of them, 860 vulnerabilities were discovered in 127 plugins, 72% of them were XXS vulnerabilities. When these statistics were compared

number of downloads and feedback of each user, a clear relation between user and their feedback on number of downloads. In a similar research done in 2015 by Trunde, H., & Weippl, E, they examined 448 plugins, and 405 plugins were cleared. As in 2003 when Wordpress ware recently released, it had major bugs in its core which were highly vulnerable.

Other than the statistical analysis, a manual review was also done. A plugin with more than 4000 downloads had average of 77.6. The test method neglected nearly half of the vulnerable plugins which were later discovered in manual review that they can be easily exploited (Koskinen et al. (2012), pg. 2). The plugins are not to be considered as defective, instead it has some blank side which can be altered and behavior can be easily modified. In the same research as a whole, 179393 line of PHP code was analyzed where majority of part was not vulnerable (Koskinen et al. (2012), pg. 3) and in addition majority of vulnerabilities are difficult to trace during any detection phase.

Wordpress is an open source project, there are many people who are contributing in providing features, in form of template or plugins. This reduces efforts from user end and only have to install these plugins. However, behavior of each plugin depend upon its code and usage. If plugin is not secure it can be exploited and database can be easily get manipulated. Wordpress annually checks each plugin in order to check its vulnerability level. Different strategies are used to analyze data like, testing and statistical methods. Testing method uses an analyzer which examines a PHP code and checks for vulnerability, but due to insufficient

parameters and less information about the system is not a reliable method to check if any plugin is secure or not. However, for statistical method also known as white box method is when a programmer check code with perspective of an attacker who have complete knowledge of how system works, which increases to create penetration and detect vulnerabilities. Many surveys have been conducted and it was determined that statistical method have more advantages than testing (black box method), as many plugins were declared safe where as in statistical analysis it was considered as highly vulnerable. The other reason for user not preferring to any risky plugin or any unknown contributor is because user feedback creates trust towards existing plugins. Verbal feedback has improved preference of many plugins. In survey of Koskinen et al. (2012), pg. 4). It was concluded that a nonlinear relation between rating and number of vulnerabilities in a plugin. The time when WordPress was introduced, it had many problems, but by now other security checks for every product launched from their platform or plugin that is needed to be added in official directory.

Reference:

1) Koskinen, T., Karavirta, V., & Ihantola, P. (2012). Quality Of WordPress Plug-Ins: An Overview of Security and User Ratings. 834-837. doi:10.1109/SocialCom-PASSAT.2012.31

2) Trunde, H., & Weippl, E. (2015). WordPress security: An analysis based on publicly available exploits. 1-7. http://dx.doi.org/10.1145/2837185.2837195

3) WordPress.org. (n.d.). Retrieved June 13, 2016, from https://wordpress.org/about/