2015

Hacked! Lessons Learned from an URL Injection

Ayyoub Ajmi
University of Missouri - Kansas City, School of Law

HACKED!

LESSONS LEARNED

BEFORE WE KNEW IT, WE HAD SEVERAL CORE WORDPRESS FILES COMPROMISED, AND ADDITIONAL FILES WERE INJECTED THROUGHOUT THE SERVER.

Starting in 2014, we've seen an increasing number of cyberattacks targeting all sorts of organizations. These have cost the U.S. economy billions of dollars and exposed the private data of millions of individuals.

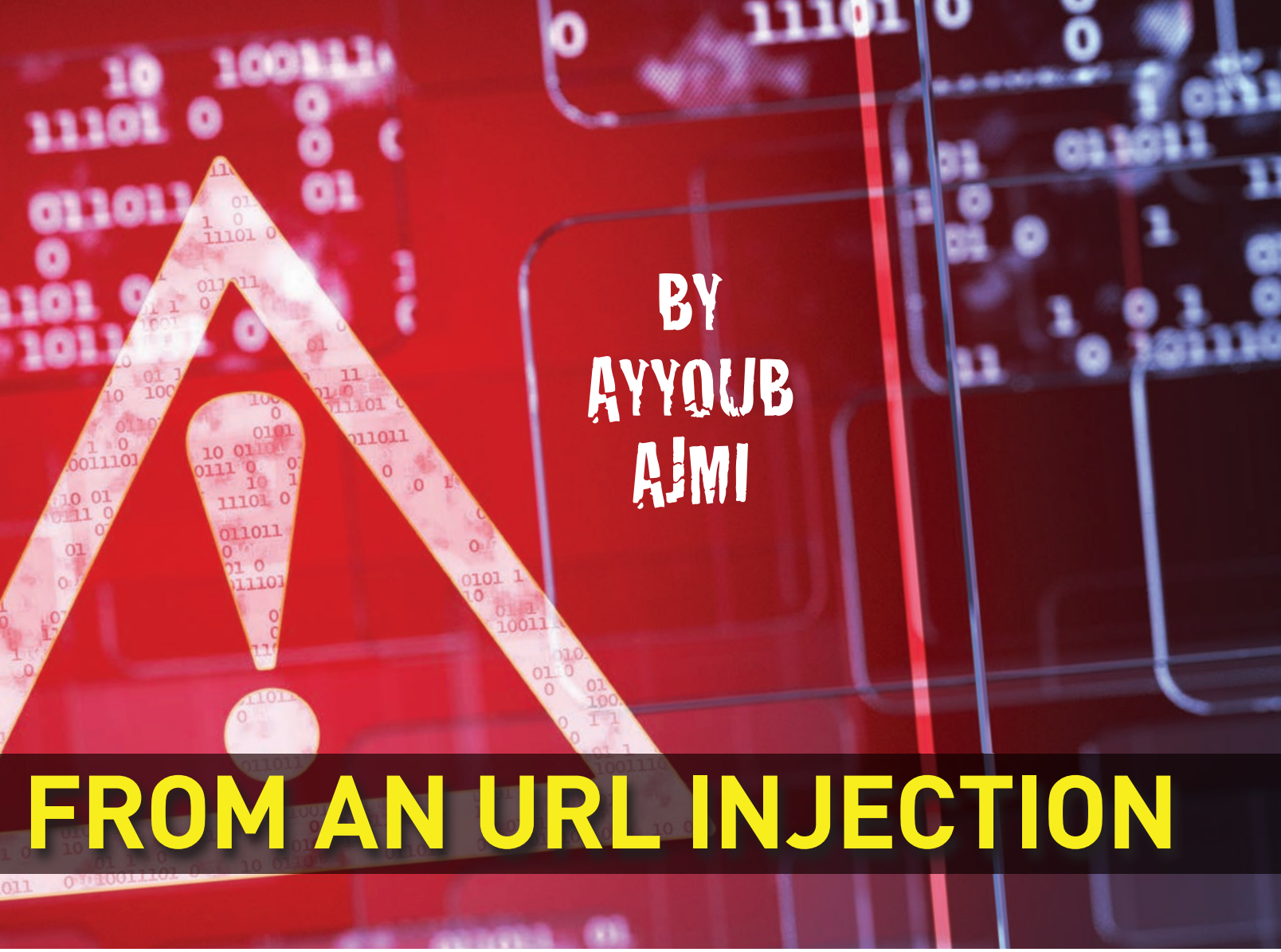
While most of them are designed to steal trade secrets, credit card information, or even celebrities' personal information, there are still other attacks targeting individuals and small organizations with the sole goal of spreading malware and promoting shady businesses. This is URL injection.

The experience of the University of Missouri–Kansas City's School of Law falls under the latter scenario. Recently, the school experienced a cyberattack, and its website was breached. No private or critical information was exposed. Nevertheless, our newly designed website was compromised, and our search engine ranking was nearly damaged. Thanks to our security team's quick response, we avoided a disaster and a public embarrassment.

The purpose of this article is to share our experience with this type of hacking, to describe its scope, to suggest how to avoid it, and—if you fall victim to it—how to clean up the mess it leaves behind in your server and in search engines.

About the UMKC School of Law

The School of Law works closely with and abides by the governing rules of both the University of Missouri–Kansas City (UMKC) and the University of Missouri System. When it comes to technology and infrastructure, we are required to get the approval of the main campus IT services department for any connected devices to make sure they successfully pass the school's security requirements.



BY AYYOUB AJMI

FROM AN URL INJECTION

Additionally, our law library's IT staffers support all the technology needs of the School of Law, including various devices and programs used by faculty members, staffers, and students, as well as our storage, intranet, and web servers.

The School of Law's previous website, designed and built by the main campus' communications department, was rather new. It featured a clean design and organization. However, what started as a marketing tool to promote the school and its services became more of a repository for current students' data and information. Hundreds of static pages went out of control, as new content was being added daily, making the navigation harder for us and our visitors.

In 2013, the school adopted a new intranet system as its main internal communication channel as well as the school's learning management system. This initiative was soon followed by a decision to remove all the active students' related material, and information hosted on our website was to be made available exclusively in our intranet system.

The New Website

In late 2013, we started planning for a new website. After consulting with the campus' communications department and IT services, we decided to use WordPress as our CMS to build the new website. WordPress was supported by the university, and a couple of other schools had already migrated their static websites to it.

While most of these existing websites were hosted on the university's Linux servers, ours was going to be hosted on our virtual server running Windows' operating system. Although WordPress is fully supported by Windows, the installation and configuration of the server are noticeably different and require more resources.

Our new WordPress website was first built in a development server while the law school's communication director was working on a new design and gathering new and updated content from the different stakeholders. When we approached our launch date, we decided to move the new website to the live server in a subfolder, while keeping the old website active. By doing so, we had enough time to set up the server and switch to the new website without having any down time. However, the launch date was delayed as more content needed to be added and a few design elements needed to be changed. On January 5, we launched the website. Two weeks later, we received an email through Google Webmaster

Tools, informing us that our newly designed website was hacked and Google might label our site's pages as hacked.

The URL Injection

URL injection (aka SEO poisoning) is a technique used to drive traffic to malicious sites by injecting poisoned URLs and redirections into legitimate websites.

Most search engines have guidelines for acceptable methods to help developers better optimize their websites to elevate the ranking of their pages in search results. However, some search engine optimizers may game the system by using different techniques (aka Blackhat SEO), such as cross-linking pages of the same website, "stuffing" webpages with popular keywords, posting links to a site in the comments section of webpages, and "link farming," which consists of a group of websites that all link to each other (Shinder, 2010). When the goal is to drive traffic to a malicious site, it becomes SEO poisoning.

SEO-driven attacks use scripting languages (typically, PHP) to dynamically generate pages stuffed with popular keywords and links to be eventually crawled and indexed in search engines (Howard and Komili, 2010). When visitors click on a poisoned link, they are redirected to malicious sites, exposing them to malware and viruses.

In its "Web Application Attack Report #5," Imperva (a cyber and data security products provider) found that websites running WordPress were attacked 24.1% more than websites running on all other CMS platforms combined (2014). The report also suggests that it is mainly due to WordPress' increasing popularity in recent years, which has led to more attackers researching and exploiting its vulnerabilities.

In our case, we found similar types of attacks used to poison our WordPress installation, caused by an incorrect configuration of our file structure permission. Before we knew it, we had several core WordPress files compromised, and additional files were injected throughout the server. We also found the abovementioned link farming. All of the injected links were using popular brand names and popular keywords to drive traffic to malicious websites.

Figure 1

Figure 2

For example, the link in Figure 1—which was placed in the index.php file of our theme—will not be displayed when rendered in a browser, as it does not have a link text element. However, search engine crawlers will read the title element part, which holds the keywords used by the attackers, "Replica Rolex." This simple code, when injected in many other pages, will elevate the ranking of the targeted URL in search engines.

Figure 2 shows an example of link farming in which links to other compromised websites were injected in core WordPress files and placed inside a hidden div. Hidden div is a common Blackhat SEO technique that hides keywords—or, as in this case, links—from the visitors to a webpage while they remain visible to search engine crawlers.

In other instances, we discovered new files in our server named like WordPress core files, but whose function is to generate hundreds of other malicious pages that, in turn, will be indexed in search engines. By limiting the search results using the keyword "Vuitton" to our website only in Bing's search engine, Bing returned the results shown in Figure 3. This result is linking to a page that has been automatically generated by the code placed in the umk.php file.

Figure 3

Figure 4

And by limiting the search results using the keyword "replica" to our website only in Google's search engine, Goolge returned the results shown in Figure 4. In addition, the malicious links were not only limited to web results but also were found in image results on popular search engines. Figure 5 shows the results of a search within our website in Google Images' search engine.

Figure 5

The Cleanup

When we first received the note informing us that our website was hacked, we immediately contacted the CIO's office at the university. The office quickly had a team working on our server. The initial action taken by the information security officer was to find and close the breach in our server that allowed the attack. We then spent several hours figuring out the damage and cleaning up the files.

ONE MONTH AFTER THE INCIDENT, WE HAD THE WEBSITE RUNNING AND SECURE AGAIN.

The assessment from the security team was that the attack seemed to have happened months before the official launch of the website. It coincided with when the website moved from the development server to the live server. Due to the nature of the attack, it was highly probable that it would have left some untraceable malicious codes that would allow it to happen again. The solution suggested by our security team was to take down the entire server and have a clean installation of the website in a fresh WordPress environment. However, taking the website down for an extended period of time was out of question. We were in the middle of the school year and getting ready to kick off the enrollment process. After discussing it with all stakeholders, we decided to stabilize the damage while we figured out what to do with the server.

In our case, finding suspicious files and injected codes wasn't difficult. Once we knew the type of attack we were dealing with, it was evident that the goal was to promote web links. We scanned the entire server looking for links in places they shouldn't be, such as the upload folder. We also scanned and verified every single link in our custom files. Another action we took was to replace all core WordPress files with new ones, which left us with the plug-ins and the theme files to scan.

Overall, it took almost a week to clean up all the files in our server. But we were not done. The compromised pages had already been crawled and indexed in search engines. Although these links will eventually be dropped—since the pages have been removed from our server—we needed to clean up our search results. Fortunately, we discovered the right tools to expedite the process.

During this phase, we limited our task to Google and Bing since they represent most, if not all, the traffic generated to our website through search engines. Google Webmaster Tools and Google Analytics were particularly useful as they helped us figure out where the traffic to our website was coming from and what pages were being indexed. Knowing the keywords that generate traffic to our website helped us identify the pages infected that needed to be removed from our server. Filtering search engine results based on these keywords in relation to our website helped us identify the already indexed pages and images that needed to be removed from search engines. Google and Bing have their own page removal request forms, and the process can be completed in few hours. By following this strategy, we were able to clean up most of the compromised links related to our website in Google and Bing.

During this time, we were closely monitoring the performance of our server in order to identify any anomaly. We were mostly looking for any suspicious connection, files, or folders that can be re-injected.

Conclusion

The big lesson we learned from this experience is the need for a healthy cross-departmental collaboration. Teaming up with the main IT and security departments should have been requested from the beginning, even if it meant delaying the project a few more weeks or even months. But thanks to their alertness and expertise, we avoided the worst. One month after the incident, we had the website running and secure again. Another lesson learned is the necessity to ensure that the test environment reflects the live environment. In our case, our development server did not meet the ideal requirements to run a dynamic website, which led us to resume the development in the live environment. No website should go to a live server during development or before being fully tested and audited by personnel different from those who built it. Testing the website in a development environment would have helped us identify the permission errors at an early stage and avoid the attack.

At the same time, hackers are always searching for vulnerabilities in websites and reinventing new ways to compromise them. Therefore, it is very important to have a fallback plan for when such misfortune happens. Having a clean backup of our WordPress installation and database helped us restore the website in a short amount of time. Keeping a change log was a great way to figure out the updates and changes we made since the last backup, so it was easy to re-create them.

In the next few weeks, we will start the process of migrating the website from our Windows platform to a Linux server dedicated exclusively to WordPress, to be run and maintained by the UMKC's information services department. Our task will then be limited to updating our content, WordPress installation, and any third-party plug-in we use.

References

Howard, Fraser and Komili, Onur (2010). "Poisoned Search Results: How Hackers Have Automated Search Engine Poisoning Attacks to Distribute Malware." SophosLabs.

Imperva (2014). "Web Application Attack Report #5."

Shinder, Deb (July 28, 2010). "SEO Poisoning: What It Is and What You Can Do About It." Retrieved from windowsecurity.com/articles-tutorials/Web_Application_Security/SEO-Poisoning-What-it-is-what-you-can-do-about-it.html.

Ayyoub Ajmi (ajmia@umkc.edu; ayyoubajmi.com) is a digital communications and learning initiatives librarian at the University of Missouri–Kansas City's School of Law. He is building and managing an integrated digital communications platform, which provides access to the law school's library and its digital resources, supports law faculty members' effective use of technology to enhance student learning, and facilitates information and communication among various constituencies of the law school.