Doesn't Any Presidential Candidate Know How to Secure WordPress?

October 23, 2015

Jonathan Lampe, CISSP - Product Manager, InfoSec Institute
jonathan.lampe@infosecinstitute.com - 920-248-0656 (cell) - http://infosecinstitute.com - @infosecedu

As I researched the security awareness of the top five presidential candidates, Bryan Quigley published a quick list of what every candidate appeared to run on their web site. I took the WordPress and Concrete entries from Quigley's list, added Trump's WordPress site, and performed some quick reconnaissance on all of them. The results were pretty bad.

In all, I harvested one hundred twenty-seven usernames from thirteen candidates' WordPress sites; only two sites refused my request! Three of the WordPress sites still had the default "admin" account from their original WordPress installation. Outdated software with known vulnerabilities was running on four of the thirteen WordPress sites and the only Concrete site in my survey. And directory listings, an issue that would potentially allow people to read hidden files, were available on two sites.

Since I gave a security awareness grade of "C" to both Jeb Bush (R) and Bernie Sanders (D) in my last evaluation and both candidates used WordPress, I used their sites as the standard for all the other candidate sites. If I had hoped I would see better results from other candidates, I was sorely disappointed.

Candidate	Grade	Site	Requires HTTPS	Wordpress or Concrete Version	Vulnerable Software or Plug-In	Exposes Usernames	Exposed Users	Uses Default "Admin"	Exposes Folders	Sign On Page	User Sign Up	Other
Ted Cruz (R)	C	tedcruz.org	Yes	4.3.1	No	Yes	16	No	No	Yes	No	-
Marco Rubio (R)	C-	marcorubio.com	Yes	4.3.1	No	Yes	8	No	Yes	Yes	No	-
Lindsey Graham (R)	C-	lindseygraham.com	No	4.3.1	No	Yes	5	No	No	Yes	No	-
Jeb Bush (R)	C	jeb2016.com	Yes	4.3.1	No	Yes	26	No	No	Yes	No	-
Donald Trump (R)	B+	secure.donaldtrump.com	Yes	4.3.1	Possibly (P)	No	None	No	No	Yes	No	-
Bobby Jindal (R)	C	bobbyjindal.com	Yes	4.2.4	Yes	Yes	6	No	No	No	No	**1**
John Kasich (R)	C-	johnkasich.com	Yes	4.3.1	No	Yes	6	Yes	No	Yes	No	-
Jim Gilmore (R)	D	gilmoreforamerica.com	No	4.3.1	No	Yes	2	Yes	No	Yes	No	-
Bernie Sanders (D)	C	berniesanders.com	Yes	4.3.1	No	Yes	21	No	No	Yes	No	-
Martin O'Malley (D)	C-	martinomalley.com	Yes	4.2.5	Yes (P)	Yes	12	No	No	Yes	No	-
Lincoln Chafee (D)	C	chafee2016.com	No	4.3.1	Yes (P)	Yes	11	No	No	No	No	**2**
Jim Webb (D)	A-	webb2016.com	Yes	Unknown	No	No	None	No	No	Yes	No	**3**
Larry Lessig (D)	D-	lessig2016.us	Yes	4.2.3	Yes	Yes	14	Yes	Yes	Yes	No	-
George Pataki (R)	C-	georgepataki.com	No	5.6.3.2 (C)	Yes	No	Unknown	No	No	Yes	No	-

(C) in the "Version" column indicates this is a Concrete site, not a WordPress Site
(P) in the "Vulnerable" column indicates vulnerability is in a plug-in rather than the core software
"Other" **1** = External access to the sign on and registration pages are actively refused.
"Other" **2** = Sign on and registration pages are both protected with Basic Authentication.
"Other" **3** = Uses WordFence security plug-in (good) but also incorporates non-HTTPS resources in HTTPS pages (bad).

Only one candidate, Jim Webb (R), earned an A-, and his wasn't a perfect grade. Only one other, Donald Trump (R), earned a B+. Three candidates tied Bush and Sanders with a C and

five were right below them with a C-. And down in the bipartisan cellar were Jim Gilmore (R) with a D and Larry Lessig (D) with a D-.

Detailed Analysis

Requiring HTTPS

As I noted in my last research paper, and Quigley notes on his blog post, HTTPS is now not only expected to be used but to be required on all presidential candidates' web sites. However, Lindsey Graham (R), Gilmore, Lincoln Chafee (D) and Lessig didn't seem to get the memo, as all their sites allowed unencrypted communications to occur.

Fortunately, all other eight candidates required HTTPS, as did all the candidates in my previous top five candidate research.

Updated Versions and Vulnerable Software

Most candidates' WordPress sites ran updated software with no vulnerabilities. However the sites of Bobby Jindal (R) and Lessig both ran older versions of WordPress with known vulnerabilities. Jindal's site ran WordPress 4.2.4 which had two cross-site scripting vulnerabilities and one other vulnerability. Lessig's site ran WordPress 4.2.3 which had all the vulnerabilities of Jindal's site plus three more cross-site scripting vulnerabilities, a SQL injection vulnerability and one other vulnerability.

The sites of Martin O'Malley (D) and Lincoln Chafee (D) used safe versions of WordPress but both used an old WordPress plug-in with a known vulnerability. Specifically, both O'Malley and Chafee used a plug-in called WordPress SEO 2.1.1 that contained a cross-site scripting vulnerability. (Martin O'Malley ran an old version - 4.2.5 - of WordPress, but his version was not old enough at the time of my research to contain known vulnerabilities.)

```
[+] WordPress version 4.2.3 identified from meta generator
[!] 8 vulnerabilities identified from the version number

[!] Title: WordPress <= 4.2.3 - wp_untrash_post_comments SQL Injection
    Reference: https://wpvulndb.com/vulnerabilities/8126
    Reference: https://github.com/WordPress/WordPress/commit/70128fe7605cb963a46
815cf91b0a5934f70eff5
    Reference: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2015-2213
[i] Fixed in: 4.2.4

[!] Title: WordPress <= 4.2.3 - Timing Side Channel Attack
    Reference: https://wpvulndb.com/vulnerabilities/8130
    Reference: https://core.trac.wordpress.org/changeset/33536
    Reference: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2015-5730
[i] Fixed in: 4.2.4

[!] Title: WordPress <= 4.2.3 - Widgets Title Cross-Site Scripting (XSS)
    Reference: https://wpvulndb.com/vulnerabilities/8131
    Reference: https://core.trac.wordpress.org/changeset/33529
    Reference: https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2015-5732
[i] Fixed in: 4.2.4
```

A partial list of vulnerabilities found in the version of WordPress that Bobby Jindal's site ran from wpvulndb.com and wpscan.

Also, as I noted in my last research paper, Trump's website may have been running an old version of a donation plug-in from Targeted Victory, but I did not have enough information to conclude that. (If Trump's site was clean, he may have earned an outright "A".)

Donald Trump's donation plug-in from Targeted Victory appears to be on version 1.2.3b, but the plug-in has been removed from github.com, a public source code site (i.e., I could not review the plug-in's version history), and the plug-in is targeted at versions of WordPress released in early 2014 (3.8.1-3.9.0).

George Pataki (R) used a site built on the Concrete content management system rather than WordPress. Like WordPress, Concrete has enough of a following to have its vulnerabilities and fixes published when they occur. Since Concrete vulnerability information is readily available and since Concrete's sites readily reveal their version number, it was easy to see that Pataki's site appeared to be running an old version (5.6.3.2) that contained multiple vulnerabilities.

Source HTML from George Pataki's site suggest he was running Concrete version 5.6.3.2.

#	CVE ID	CWE ID	# of Exploits	Vulnerability Type(s)	Publish Date	Update Date	Score	Gained Access Level	Access	Complexity	Authentication	Conf.	Integ.	Avail.
1	CVE-2015-3989	79		XSS	2015-05-15	2015-05-18	4.3	None	Remote	Medium	Not required	None	Partial	None
Multiple cross-site scripting (XSS) vulnerabilities in concrete5 before 5.7.4 allow remote attackers to inject arbitrary web script or HTML via vectors related to private messages or other unspecified vectors.														
2	CVE-2015-2250	79		XSS	2015-05-15	2015-05-18	4.3	None	Remote	Medium	Not required	None	Partial	None
Multiple cross-site scripting (XSS) vulnerabilities in concrete5 before 5.7.4 allow remote attackers to inject arbitrary web script or HTML via the (1) banned_word[] parameter to index.php/dashboard/system/conversations/bannedwords/success, (2) channel parameter to index.php/dashboard/reports/logs/view, (3) accessType parameter to index.php/tools/required/permissions/access_entity, (4) msCountry parameter to index.php/dashboard/system/multilingual/setup/load_icon, arHandle parameter to (5) design/submit or (6) design in index.php/ccm/system/dialogs/area/design/submit, (7) pageURL to index.php/dashboard/pages/single, (8) SEARCH_INDEX_AREA_METHOD parameter to index.php/dashboard/system/seo/searchindex/updated, (9) unit parameter to index.php/dashboard/system/optimization/jobs/job_scheduled, (10) register_notification_email parameter to index.php/dashboard/system/registration/open/1, or (11) PATH_INFO to ard/extend/connect/.													
wse CWE Definitions														
3	CVE-2014-9526	79		XSS	2015-01-05	2015-01-06	4.3	None	Remote	Medium	Not required	None	Partial	None
Multiple cross-site scripting (XSS) vulnerabilities in concrete5 5.7.2.1, 5.7.2, and earlier allow remote attackers to inject arbitrary web script or HTML via the (1) gName parameter in single_pages/dashboard/users/groups/bulkupdate.php or (2) instance_id parameter in tools/dashboard/sitemap_drag_request.php.														

A list of the cross-site scripting (XSS) vulnerabilities that may affect the old version of Concrete George Pataki's web site ran from cvedetails.com

Exposing Usernames

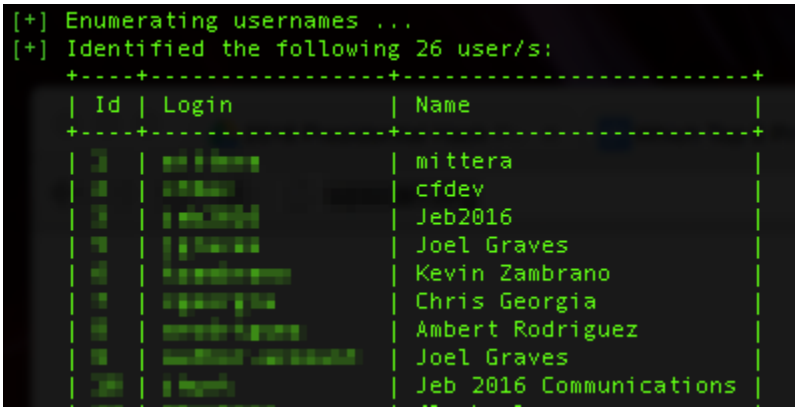
WordPress makes it frightfully easy to download a complete list of usernames and full names of a site's users. By design, WordPress supports a "list all articles by author number X" function, and numbers its users sequentially starting at 1 (for "admin"). This makes it easy for anyone to ask WordPress for "all the articles by user number X." Unfortunately, WordPress's usual answer this question is not only the list of articles by user number X, but the username and full name of the person who authored them too.

This is an example of using WordPress's built-in "articles by author" function to obtain the username and fullname of a particular author. In this case, the contents of the "title" tag from the WordPress page at this URL tell us that "author" (user) number 2 has a username of "eddiev" and a full name of "Edward Vincent".

When multiple requests are made like this, such as “list all the articles by users numbered 1 through 30”, the usernames and fullnames of multiple users can be obtained. In IT security, a technique like this is known as “user enumeration” and can be used to obtain many if not all the usernames on a particular system.

While having all the usernames of a particular system is nowhere nearly as serious as having all the usernames and passwords, it still invites mischief. For example, if I wanted to try to guess the password of a WordPress administrator, I would tune my “brute force” password-guessing script to only try the username of the lowest-numbered WordPress user, who is usually an administrator. Alternatively, I could try a social engineering attack: I could contact one of the people named in the list, pretend like I was someone else on the list, and convince my target to reset “my” password.

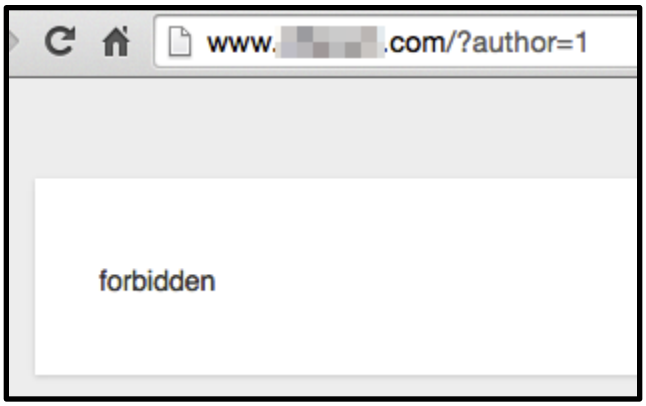
Unfortunately, all but two of the thirteen candidates allowed anyone to list all the user on their sites. The eleven candidate sites that allowed user enumeration revealed one hundred twenty-seven usernames, with individual sites revealing between two users (Gilmore’s site) and twenty-six users (Bush’s site). Only Trump’s and Webb’s sites refused my inquiries.



Partial list of usernames enumerated from Jeb Bush’s WordPress site.

Three of the sites (Kasich’s, Gilmore’s and Lessig’s) revealed a username of “admin”, which is the default administrator created when installing WordPress. Having a user named “admin” active on WordPress is especially dangerous because many “script kiddie” tools (run by unsophisticated hackers) attempt to crack into accounts with this specific username whenever they see a WordPress installation.

Despite the potential for misuse, user enumeration options are turned on by default on almost all WordPress installations, and there is no box to check to turn it off. However, there is a free WordPress plug-in called Stop User Enumeration that prevents this issue. Other WordPress security plug-ins, such as WordFence, also prevent user enumeration. User enumeration can also be prevented by tuning certain web site configurations. (Fortunately, Concrete installations do not seem to support user enumeration by default.)



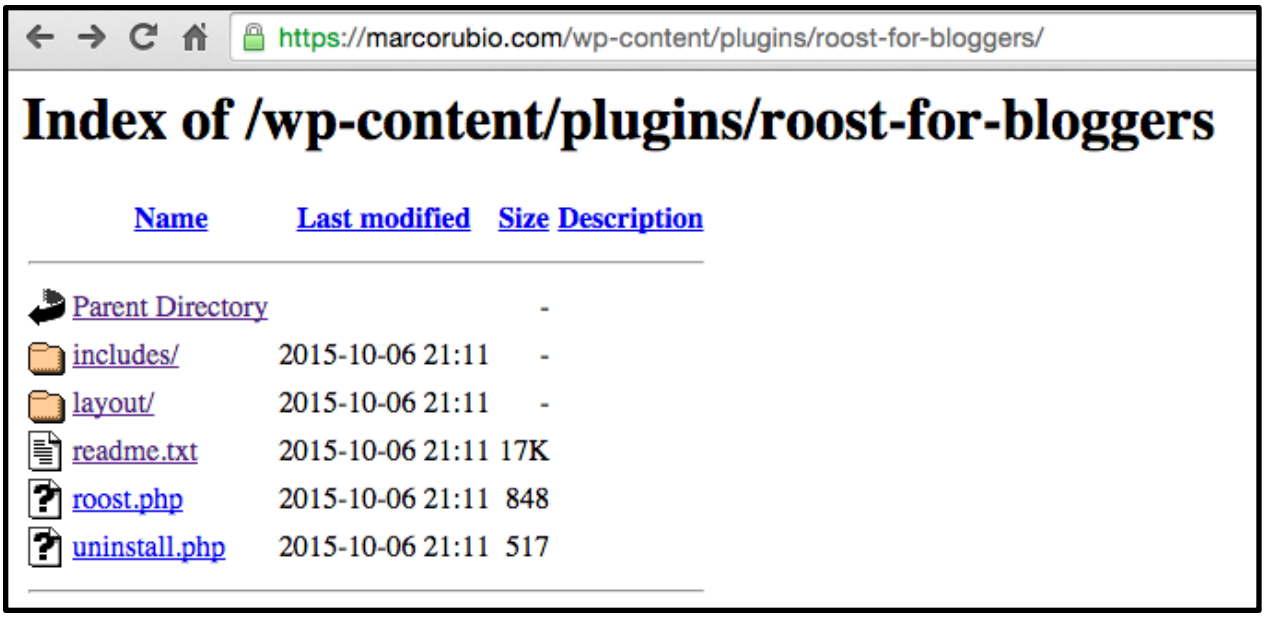
Typical response to a user enumeration request from a WordPress site protected by the Stop User Enumeration plug-in.

Exposing Folders

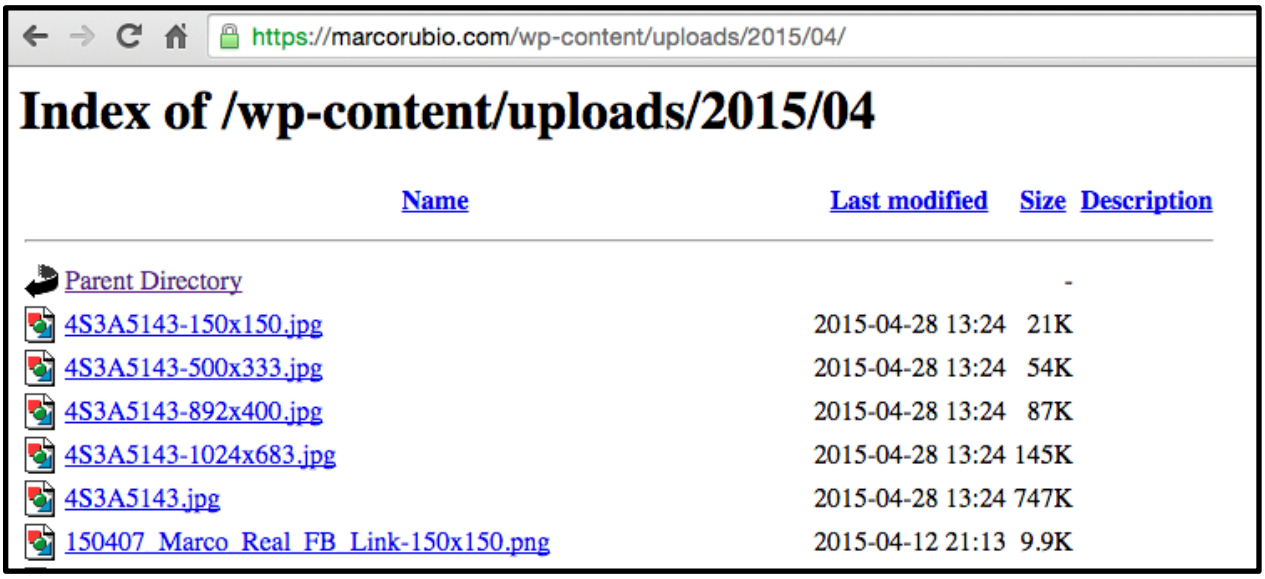
While it is always a bad idea to leave old, unused material behind in web-accessible folders, people still do it because they feel safe knowing that there are no web-accessible links to those files. Examples of materials frequently left where they should not be include source code, design files, and backups of configurations (that often contain internal addresses and passwords).

However, the protection of "no links" becomes irrelevant when a web server allows people to browse folders full of old files by enabling "directory listing." It is no longer common to allow directory listing to an entire web server, but it is still common to see directory listing misconfigurations on individual folders used to support WordPress.

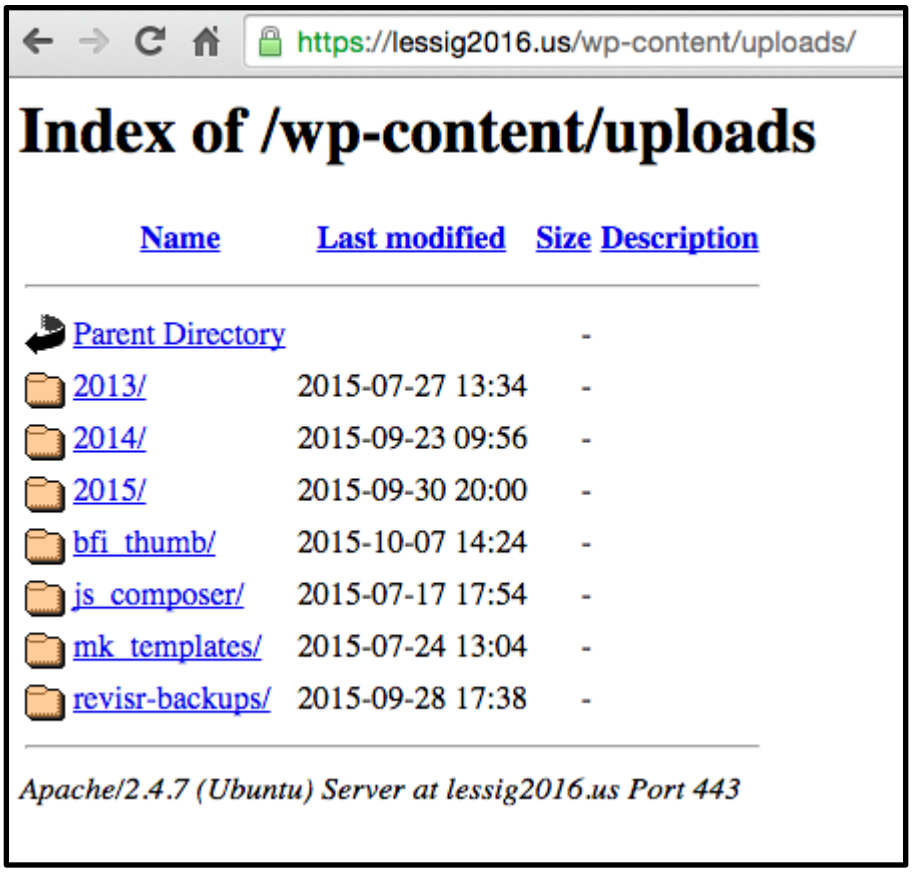
Two candidates' web sites suffered from these kind of directory listing problems. Rubio's site had a directory listing problem in both his "uploads" and "plug-ins" folders, and Lessig's site had a directory listing problem in his "uploads" folders. I did not look closely at either site for improperly exposed material; fortunately the root folders of each exposure seemed to display only harmless material.



Marco Rubio's web site permissions allowed anyone to list and read the contents of his WordPress "plug-ins" folder.



Marco Rubio's web site permissions allowed anyone to list and read the contents of his WordPress "uploads" folder.



Larry Lessig's web site permissions also allowed anyone to list and read the contents of his WordPress "uploads" folder.

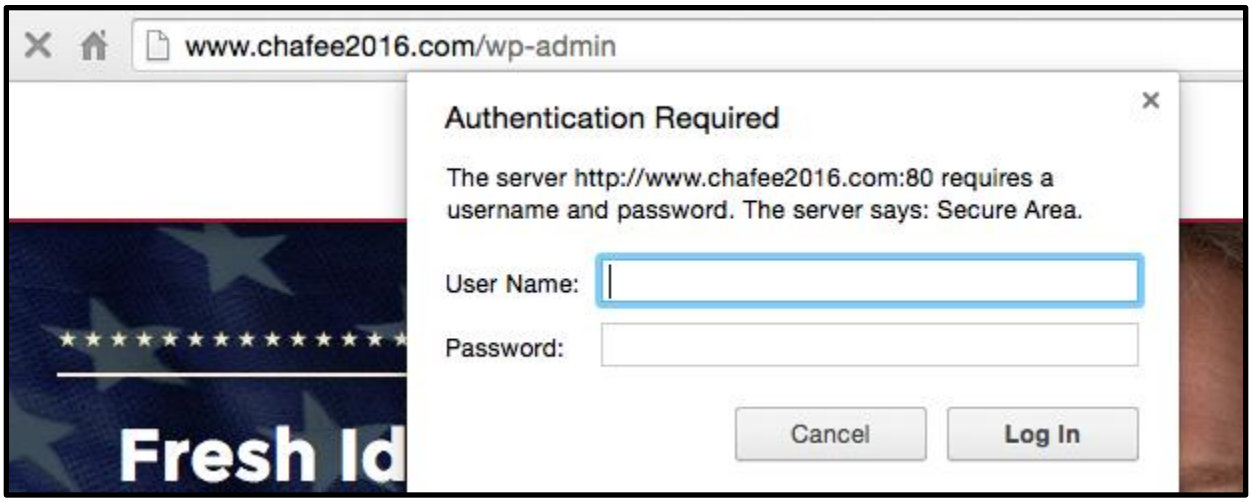
Self-Registration

If your goal is to take over another site, it often helps to sign on as a restricted or normal user and then "elevate" your privileges to those of an administrator. Many WordPress installations allow people to self-register for restricted accounts so they can comment on articles or contribute other content, but this capability is not needed for campaign websites.

I was happy to see that none of the candidates' web sites allowed people to self-register. Unlike user enumeration, self-registration is a checkbox option in WordPress and all candidates appeared to have that box checked correctly.

Other Issues

Three campaign sites offered some surprising but effective defenses against certain WordPress attacks. Jindal's site refused connections to his sign on and registration pages. Chafee's site protected his sign on and registration pages with an extra layer of usernames and passwords called "Basic Authentication." And Webb's site used a whole-site plug-in to prevent hackers from gaining information about the site.



Lincoln Chafee's site required an additional sign-on (using "Basic Authentication") before access to its sign on page was allowed. However, any credentials entered here would be sent without the protection of encryption since HTTPS is not in use.

Unfortunately, there was something less-than-optimal about the implementations of all these protections on the three sites. In Jindal's case, the redirection request that occured when access was denied appeared to reveal information about an internal server. In Chaffee's case, his Basic Authentication request could occur over (unencrypted) HTTP, leaving those credentials open to theft. And in Webb's case a third-party HTTP resource was used on his site, which switched the icon next his site's URL from the usual green lock to an alarming-looking "warning lock."

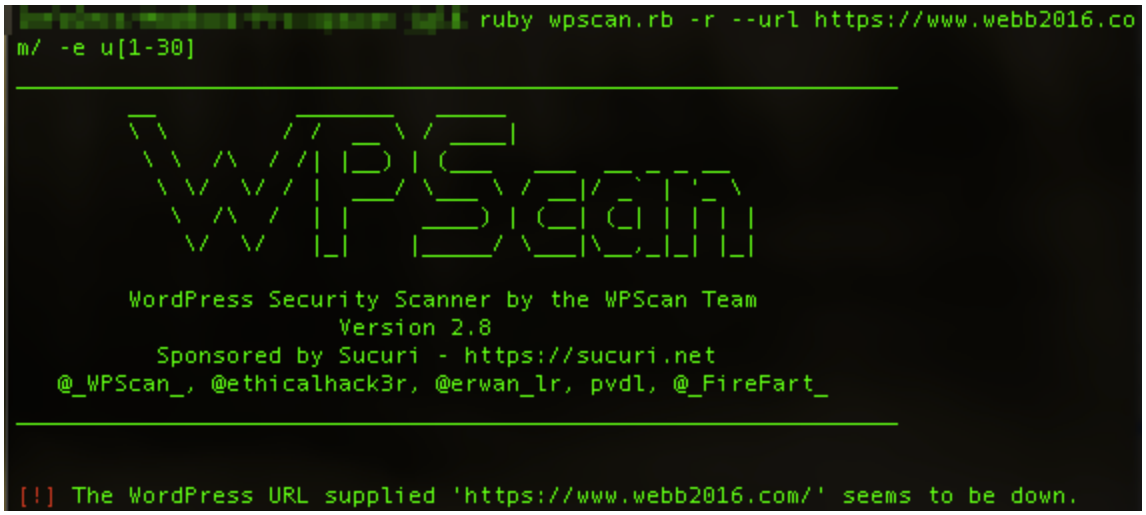
Jim Webb's use of an HTTP-only resource on his otherwise secure web site causes web browsers to display a "warning lock" next to his URL.

How Did Webb Get The Only A?

As I noted above, Webb was the only candidate to earn an A (actually an A-) because his site resisted my probes and required HTTPS. To resist WordPress probes like the one I used on all other sites, Webb uses a WordPress plug-in called WordFence that is specifically designed to secure WordPress sites from people like me.

```
<script type="text/rocketscript">
(function(url){if(/(?:Chrome\/26\.0\.1410\.63
Safari\/537\.31|WordfenceTestMonBot)/.test(navigator.userAgent)){ return; }
```

HTML code from Jim Webb's home page bearing the signature of a WordFence-protected site.



wpscan had a hard time locking on to Jim Webb's WordPress site.

Even when I forced it to look, wpscan still couldn't glean any information from Jim Webb's WordPress site.

Donald Trump would surely be pleased to know that he almost earned an "A" because his site required HTTPS, ran a current version, refused to list his users, had no directory listing issues and refused user sign ups. My only hesitation concerned his closed-source donation plug-in, which might or might not be outdated. *(I asked Targeted Victory for some clarification and will publish their response if I receive one.)*

Candidate	Grade	Site	Requires HTTPS	Wordpress or Concrete Version	Vulnerable Software or Plug-In	Exposes Usernames	Exposed Users	Uses Default "Admin"	Exposes Folders	Sign On Page	User Sign Up	Other
Donald Trump (R)	B+	secure.donaldtrump.com	Yes	4.3.1	Possibly (P)	No	None	No	No	Yes	No	-
Jim Webb (D)	A-	webb2016.com	Yes	Unknown	No	No	None	No	No	Yes	No	**3**

(P) in the "Vulnerable" column indicates vulnerability is in a plug-in rather than the core software

"Other" **3** = Uses WordFence security plug-in (good) but also incorporates non-HTTPS resources in HTTPS pages (bad).

Should Quigley Join the Sanders Campaign?

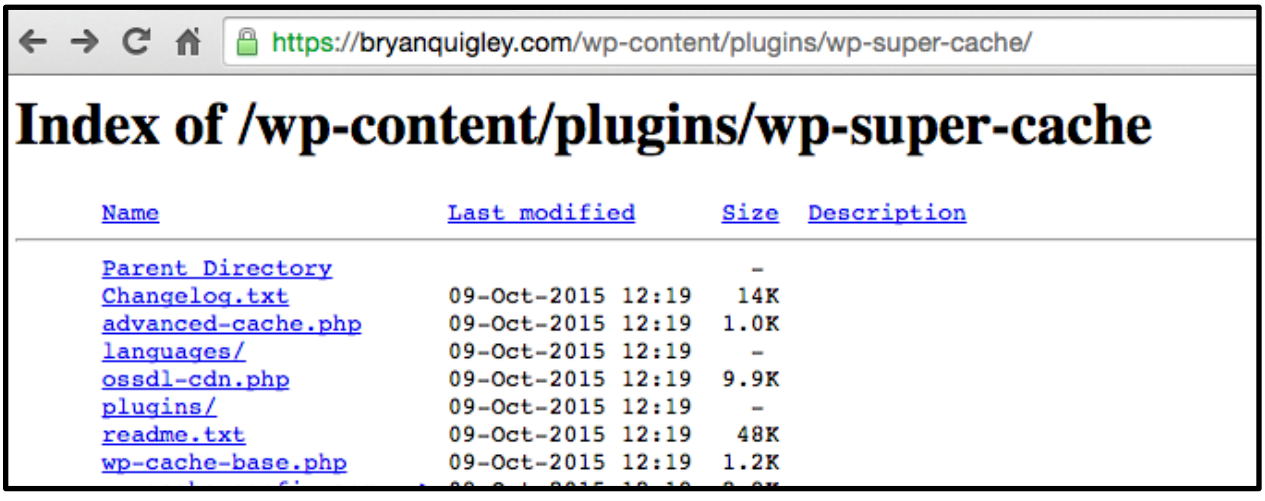
As I mentioned in the introduction, my research was inspired by Bryan Quigley, a WordPress trainer who used some free online tools to look for IPv6 support, HTTPS requirements and stated type of server on the official sites of all Democratic and Republican presidential candidates. At the end of his research he also wrote that, "I'm supporting Bernie Sanders and have volunteered to help them tech wise. "

Now, is that an offer that Sanders ought to take? Just for fun, I reviewed Quigley's site and some of the of the WordPress sites I administered over the years using the same criteria I used to judge presidential candidates.

Candidate	Grade	Site	Requires HTTPS	Wordpress or Concrete Version	Vulnerable Software or Plug-In	Exposes Usernames	Exposed Users	Uses Default "Admin"	Exposes Folders	Sign On Page	User Sign Up	Other
Bernie Sanders (D)	C	berniesanders.com	Yes	4.3.1	No	Yes	21	No	No	Yes	No	-
Bryan Quigley	C-	bryanquigley.com	Yes	4.3.1	No	Yes	2	No	Yes	Yes	No	**4**
Jonathan Lampe	B	(various)	Varies	4.3.1	No	No	None	No	No	Yes	Varies	-

"Other" **4** = Contains a full local path disclosure in rss-functions.php

It turns out that Quigley's WordPress site is strong in a couple of areas, including requiring the use of HTTPS, keeping up with current WordPress versions and plug-ins, disabling registration and avoiding the default "admin" account. However, when I looked at it in mid-October Quigley's WordPress site suffered from the same user enumeration problems that plagued Sanders's site. It also allowed anyone to list and read the complete contents of certain folders (a.k.a. "directory listing enabled") and included a "full path disclosure" vulnerability that exposed the location of Quigley's site on its local web server.



Bryan Quigley's WordPress site allowed listings of certain directories.

With those two extra (but minor) issues on the table, I would probably give Quigley's site a partial grade below Sander's site: C-.

To be fair, I also scanned a couple of the WordPress sites I worked on in the past few years as part of my evaluation. Not all of them require HTTPS, a few allow users to register on the site and many allow universal access to the sign on page. However, my WordPress sites block user enumeration, have been tuned to prevent directory listing and do not allow access to legacy files that could disclose internal information. Since my sites have a wider attack surface because I often use HTTP, allow user registration and permit logins from the Internet, I could not in good conscience give myself an A. However, what I did leave behind is probably clean enough to earn a solid B.

Conclusion

All but two of the fourteen candidates I examined had configurations or old software that exposed too much information or vulnerabilities to the general public. Since I spent no more than a half hour investigating any particular site and made heavy use of a popular "script kiddie" tool (wpscan) to obtain my findings I have to conclude that almost anyone with an IT background could have uncovered these issues on their own.

Unfortunately, that conclusion leads me to another one: that most candidates and their staff, even their so-called "digital experts," are clueless on security awareness.

Simple Defenses Are Necessary And Easily Applied

To protect against the types of probes I used in my research and "script kiddie" hacks, anyone using the WordPress or Concrete content management systems should consider the following precautions. Applying them all would have been good enough to earn an "A" grade in my research.

- **Require HTTPS** - Popular web servers like Apache, IIS and nginx can be configured to automatically redirect insecure HTTP requests to HTTPS.
- **Set Up HTTPS Correctly** - Using a commercially-signed X.509 "web server" certificate for your site is a good start, but you should also tune your web server to turn off SSL (2.0-3.0) requests and turn on TLS (1.0-1.2) requests since TLS has replaced SSL as the secure protocol modern HTTPS sites run. You may also eventually want to tune individual encryption ciphers to strengthen your deployment, but get the other HTTPS basics in place first.
- **Update ALL Your Software** - The easiest way to prevent people from exploiting vulnerabilities on your site is to keep all your software, including your web server, your "content management system" (e.g., WordPress or Concrete), and all your plug-ins up to date. There are automatic notifications and updates built into WordPress and other content management systems - use them!
- **Prevent "User Enumeration"** - "User enumeration" exposes the usernames that can access your content management system, and user enumeration is allowed by default

on most WordPress installations. Unfortunately there is no built-in WordPress option to turn user enumeration off (yet?) but there is a free plug-in called Stop User Enumeration that prevents this issue. Other WordPress security plug-ins, such as WordFence, also prevent user enumeration, and it can also be prevented by tuning certain web filters. (Concrete installations do not seem to support user enumeration by default.)

- **Delete or Rename Your Default "admin" Account** - It has long been a best practice to rename, delete or restrict the default administrator account on any system, such as "Administrator" on Windows or "root" on Unix. Likewise, you should rename or delete the default "admin" account on your WordPress and Concrete systems.
- **Turn Off Directory Browsing** - If directory browsing has been turned on across your entire web site, it is easy to reconfigure your web site to turn it off. Unfortunately, most WordPress directory browsing issues crop up on individual directories, so a little web site expertise is often needed to correct them. Specific fixes like modifying the .htaccess file on Apache or checking different IIS "directory browsing" settings are often used to address these problems.
- **Consider Hiding Sign On and Registration Pages** - Some sites hide their sign on and registration pages from the Internet by restricting access to them from certain trusted IP addresses (such as a campaign or marketing office), or by requiring an extra level of authentication (usually "Basic Authentication") to access them. This provides an extra layer of defense against vulnerabilities that may be present but as-of-yet-undetected in your content management software.
- **Turn Off Self-Registration** - Unless you want to allow comments on your content (generally a bad idea for political campaigns) this option should always be turned off to prevent people from gaining a toehold from which they could "escalate privileges" and take over a site. Fortunately, self-registration is usually a "yes/no" option in content management systems, and all the presidential campaigns I've analyzed so far have made the right choice.
- **Scan Your Site** - Use or get a qualified IT person to use a tool to scan your site for easy-to-find vulnerabilities just like I did. wpscan, which runs on Linux or Mac machines, is an example of such a tool.
- **Get Security Awareness Training** - Even if you are "only" a web designer, an IT intern, or a campaign staffer, you need to understand that the actions you take online can embarrass or distract your candidate. Fortunately there are security awareness programs designed for everyone. The best ones use interactive exercises to help you learn and then use important security concepts in an hour or less.