# references

WP Security Whitepaper [https://wordpress.org/about/security/](https://wordpress.org/about/security/) (how WordPress approaches security)

WP Codex
- [http://codex.wordpress.org/Brute_Force_Attacks](http://codex.wordpress.org/Brute_Force_Attacks)

Blog.Sucuri.net [http://blog.sucuri.net/category/wordpress-security](http://blog.sucuri.net/category/wordpress-security)
WPSecure [http://wpsecure.net/basics/](http://wpsecure.net/basics/)
WPVulnDB.com [http://WPVulnDB.com/](http://WPVulnDB.com/)

healthy dose of paranoia

# what’s new?
- WP 4.1 -> 4.5, 9 minor point (primarily security) releases
- BruteProtect -> Jetpack Protect
- ImageMagick vulnerability (5/4/16)
- Panama Papers
  - Mossack Fonseca
    - 4.8 million emails, in part through vulnerability in Revolution Slider; could read wp-config.php, meaning MySQL access
  - also, Drupal
    - [https://www.wordfence.com/blog/2016/04/panama-papers-wordpress-email-connection/](https://www.wordfence.com/blog/2016/04/panama-papers-wordpress-email-connection/)
    - [http://www.theregister.co.uk/2016/04/07/panama_papers_unpatched_wordpress_drupal/](http://www.theregister.co.uk/2016/04/07/panama_papers_unpatched_wordpress_drupal/)

98% of vulnerabilities from exploited plugins and themes

“A look at the OSVDB (Open Source Vulnerability Database) WordPress vulnerability list shows that 554 out of 562 vulnerabilities reported in 2015 are from a third party theme or plugin. That’s 98.6% of all WordPress vulnerabilities.” [https://blog.osvdb.org](https://blog.osvdb.org) (DB shut down as of April 5th)

# why security? other benefits / get better security by doing other best practices
- reputation (avoid blacklist by proxy)
- optimization (from protecting vs bulk attacks/DDoS)
- uptime/availability (both from results of attack + time to restore)
- server performance (sometimes a server dragging is the first indication of infection)
  - e.g. Apache Status, WHM, top, exim -bpc
- Better site awareness: state of files/changes, baseline performance
- Better development process (cf. code review/source control)
- Earning potential! = freelancers/developers, be sure to include maintenance in your contract
- Good Netizen* (if anyone still uses that term)

# why me?
conscription (spam host, server b/w)
content manipulation (spam links for black-hat SEO, IFRAME injection)
steal user data

# prevent (before getting to WP)

## defense (vulnerability) in depth
Theme
  - child themes
  - embedded plugins
Plugins
  - woocommerce extensions
  - WAF plugins
Core WordPress
MySQL DB
Web Server Daemon
Server Firewall
Server OS
Network Firewall
Network
DNS

Primacy of Defense: The lower in the stack you can intercept, the better

network -> LAMP server -> site

## firewall / breakpoints
- DNS-level (Cloudflare, Incapsula, SiteLock, Sucuri proxy)
  <https://www.cloudflare.com/waf/> ($20/mo)
  <https://www.incapsula.com> ($60/mo+)
  <https://www.sitelock.com> (ask?)
  <https://sucuri.net/website-security/ddos-protection> ($20/mo)
- network-level (Cisco, Sonicwall, Watchguard, pfSense)
- machine-level (ipfw, iptables, CSF, APF)
- service-level (Apache mod_security + OWASP / ComodoWAF rules, fail2ban,
  BFD, mod_evasive)
- application-level (Wordfence WAF, Akismet)
  <https://www.elegantthemes.com/blog/tips-tricks/website-firewalls-what-they-are-how-to-set-one-up-for-wordpress>

## managed WP providers?
  - dedicated vs. shared
specialized vs. general
   cost (dedicated/higher-function packages cost more) / convenience (may already have existing hosting) / flexibility (specialized hosts may set controls)

DreamPress <https://www.dreamhost.com/hosting/wordpress/>
Flywheel.com
Siteground.com
WPEngine.com

## bulk
- proxy (Cloudflare, Sucuri) [around $20/mo]
- Brute Force Detection (e.g. CPHulk in cPanel)
  <https://documentation.cpanel.net/display/ALD/cPHulk+Brute+Force+Protection>
- JetPack Protect <https://jetpack.com/features/>

## credentials
   SSH
   cPanel / Plesk / phpMyAdmin
   MySQL
   (S)FTP
   WP-login
   least privilege assignment

## staging / version control
   staging push
   limit commit rights
   git / subversion
   rollback support
   backup backups (always have an escape route)

# protect WP itself
## security 101 (cf. Michele Butcher)
1. Acquire software only from trusted sources (WP core, plugins, theme)
2. Minimize vulnerabilities by avoiding & removing unnecessary plugins & themes
3. Stay up to date (WP core, plugins, theme)
4. Regular backups
5. Strong passwords (WP admin, MySQL, FTP)
6. Rotate keys & salts <https://api.wordpress.org/secret-key/1.1/salt>
7. No 'admin' account
8. Different DB prefix (not wp_*)
9. Secure access (SSL, SFTP)
10. Consider security plugins (but watch for conflicts & overhead)

   - most vulnerabilities through plugins & themes
## layered permissions (in case of suPHP)

- owner-only write (means manual updates or permission swap before auto-update)
  - group-only execute (with suexec in group)
  - everyone read-only/none (depending on web process owner)

```bash
chmod -R 770 public_html
chmod -R 750 public_html
```

## multi-tenant

- [http://jason.pureconcepts.net/2013/04/updated-wordpress-multitenancy/](
- [http://goodguyry.me/notes/multi-tenant-wordpress.html](http://goodguyry.me/notes/multi-tenant-wordpress.html)

1. Install WP into subdirectory (e.g. /core)
2. Follow Codex instructions for 'Giving WordPress Its Own Directory'
3. Copy wp-config.php to site root (/)
4. Edit subdirectory wp-config.php to include via `$_SERVER['DOCUMENT_ROOT']`
5. Move subdirectory to core path (e.g. /usr/local/wordpress/4.0)
6. Symlink subdirectory to new core path
   ```bash
   ln -s /usr/local/wordpress/4.0 core
   ```
7. Site now loads index.php, which looks to /core/ through symlink, which references back to originating site's wp-config via `$_SERVER['DOCUMENT_ROOT']`

   'Update' of core WordPress is now the same as 'replace symlink with pointer to different version'
   ```bash
   rm core; ln -s /usr/local/wordpress/4.1 core
   ```

## limit/disable

Disable php.ini functions

```bash
disable_functions=exec,passthru,shell_exec,system,proc_open,popen,curl_exec,curl_multi_exec,parse_ini_file,show_source
allow_url_fopen=Off
allow_url_include=Off
```

Set database restrictions (SELECT, INSERT, UPDATE, DELETE, ALTER)

Basic auth on /wp-admin <http://codex.wordpress.org/Brute_Force_Attacks#Password_Protect_wp-login.php>

- allow exception for admin-ajax.php via /wp-admin/.htaccess
- <Files admin-ajax.php>
- Order allow,deny
- Allow from all
Satisfy any
</Files>

Limit logins by IP <http://codex.wordpress.org/Brute_Force_Attacks#Limit_Access_to_wp-admin_by_IP>

Disable file editing in wp-config.php
define('DISALLOW_FILE_EDIT', true);

.htaccess rules
wp-login
  # Stop spam attack logins and comments
  <IfModule mod_rewrite.c>
  RewriteEngine On
  RewriteCond %{REQUEST_METHOD} POST
  RewriteCond %{REQUEST_URI} .(wp-comments-post|wp-login).php*
  RewriteCond %{HTTP_REFERER} !.*yourwebsitehere.com.* [OR]
  RewriteCond %{HTTP_USER_AGENT} ^$
  RewriteRule (.*) http://%{REMOTE_ADDR}/$ [R=301,L]
  </IfModule>

xmlrpc.php
  [Settings > Discussion > Default Article Settings, and uncheck
  “Allow link notifications from other blogs (pingbacks and trackbacks)”]

  # Block WordPress xmlrpc.php requests
  <Files xmlrpc.php>
  order deny,allow
  deny from all
  </Files>

wp-config.php
  <files wp-config.php>
  order allow,deny
  deny from all
  </files>


XST
  # Disable HTTP Trace attack
  RewriteEngine On
  RewriteCond %{REQUEST_METHOD} ^TRACE
  RewriteRule .* - [F]
Block the include-only files.

```html
<IfModule mod_rewrite.c>
RewriteEngine On
RewriteBase /
RewriteRule ^wp-admin/includes/ - [F,L]
RewriteRule !^wp-includes/ - [S=3]
RewriteRule ^wp-includes/[\^\/]+.php$ - [F,L]
RewriteRule ^wp-includes/js/tinymce/langs/.+\.php - [F,L]
RewriteRule ^wp-includes/theme-compat/ - [F,L]
</IfModule>
```

In theme’s `functions.php`:
```php
// remove version info from head and feeds
function complete_version_removal() {
    return "";
}
add_filter('the_generator', 'complete_version_removal');
```

Hide WP version

Block in `robots.txt`
```text
User-agent: *
Disallow: /wp-content/plugins/
Disallow: /wp-admin/
Disallow: /wp-content/
Disallow: /wp-
Disallow: /xmlrpc.php
```

## plugins
- iThemes Security
- Sucuri
- Wordfence

## code sanitation
- review before deployment
  - safe: eliminate XSS/unescaped/unsanitized
  - scalable: smart queries, cached functions, DRY code
```

<https://en.wikipedia.org/wiki/Don%27t_repeat_yourself>
readable: coding standard

WP Coding Standards
<https://tommcfarlin.com/php-codesniffer/>
VIP Quickstart/VIP Scanner (public Vagrant)
<https://github.com/Automattic/vip-quickstart>
<https://wordpress.org/plugins/vip-scanner/>
<https://github.com/Automattic/vip-scanner>
continuous integration testing (Travis)
<https://travis-ci.org>
WP Enforcer
<https://github.com/stevegrunwell/wp-enforcer>

unit tests

code review

# detect
(AV modes: real-time intercept vs scan vs “my computer seems slow”)

## scan
ClamAV
Linux Malware Detect: maldet --monitor /path/to/wordpress/
OSSEC <http://ossec.github.io>
Sucuri scheduled scans
WP CLI: php wp-cli.phar --path=/var/www/bob/ core verify-checksums | mail -s "WP change check" your@email.com

## server anomalies
  high CPU/load
  web activity (Apache Status)
  unusual traffic pattern (analytics)
  mail queue backlog (mailq / exim -bpc)
  review logs

## notify
WP management (InfiniteWP, MainWP)
uptime monitoring (e.g. Jetpack, MainWP extension)
Google Webmaster Tools (for Google Safe Browsing)

don’t let your visitors be the first to know!
## hack

**pen test**

- Flunym0us <http://code.google.com/p/flunym0us/>
- Kali <https://www.kali.org>
- Nikto <https://cirt.net/Nikto2>
- WPScan <http://wpscan.org/>
- WP Sploit Framework <https://github.com/0pc0deFR/wordpress-sploit-framework>

## recover

- [https://codex.wordpress.org/FAQ_My_site_was_hacked](https://codex.wordpress.org/FAQ_My_site_was_hacked)

### what to look for

- check admin accounts
- check logs/analytics
- mismatched modification dates
- base64 encoding
- injected eval( ) code

```
<?php
eval(gzinflate(base64_decode('y0zTyCwuTi3RUIkPcg0MdQ0OiVZPzlCP1VRQU1PQyE0xxZSwtVVQN0szt0xKtDRONTCxTDazNLIwNz0NTU1NzUxMTUxNE1RB+vHMLkgoyA+OT8lFWiMpkK1QmpZYg4OaWuF1rMEg0gXQsA')));
```

- compare folder counts vs staging
- diff vs source

[hackers are lazy too: injections usually in top line because that’s easier to script]
and not break
    [redundancy is easy to program, so cleaning one file is often not enough]
[like worms, they love to burrow into dark sub-sub-directories, like /wp-includes/
SimplePie, /uploads/2012/03]
[check default themes, even if inactive]

- scan & repair
  LMD/ClamAV

  WP plugins <https://wordpress.org/plugins/search.php?q=malware+scanner>
  Sucuri/Wordfence signature comparison
  Exploit Scanner <https://wordpress.org/plugins/exploit-scanner/>
  (out of date) Theme Authenticity Checker <https://wordpress.org/plugins/tac/>

  quarantine out of site root

- nuke & pave
  put up placeholder home page
  reinstall clean WP+theme+plugins from source
  restore content from backup (DB, /uploads)
  test on staging

- forensic postmortem
  how did they get in? (and did you fix it?)
  what did the code allow them to do? (and have you corrected it?)
  is this kind of attack new? (should you share with a security service?)

- reset the locks
  change salts in wp-config
  change passwords
  reapply base permissions
  up vigilance (retribution, re-assertion)