Wordpress Administration

Sicherheit, Zuverlässigkeit und Performance: Administration im Dreieck zwischen Erstellung, Betrieb und Hosting.

Administration wird bei Wordpress-Sites oft stiefmütterlich behandelt. Selbst wo der inhaltliche Betrieb, die Vermarktung und die Programmierarbeiten professionellen Ansprüchen genügen wird die Administration von irgendjemand "nebenbei" gemacht.

Ich zeige die Möglichkeiten professioneller, spezialisierter Administration auf, sowie die Gefahren, die lauern, wenn man die Administration vernachlässigt.

Ich

Matthias P. Würfl

Früher PHPler, jetzt WP-Admin

mpw@taquiri.de

http://www.taquiri.de

Stichworte:

- FTP (schlecht)
- GIT/SVN (gut)
- wp-cli (gut)
- DDOS (schlecht)
- Feuer im RZ (schlecht)
- Caching (gut)
- 12factor app (gut)
- Daten ausführen (schlecht)
- Monitoring (gut)
- Backup (gut)
- Restore (besser)
- Puppet/Chef (gut)
- Apache mit PHP-Modul (schlecht)
- fastCGI (gut)
- DevOps (hihi)
- HTTP/2 (gut)
- PHP7 (gut)
- chmod 777 (schlecht)
- Adminstrator

Hier keine fertigen Lösungen (Code), sondern Konzept/Checkliste/Ideen

FTP

- klassisch
- manuelles hin und herschieben von Dateien (Programmierung, auch Core und so)
- unverschlüsselt
- fehleranfällig
- gleichzeitiges (über-)schreiben
- versehentliches überschreiben von Config
- no undo
- chmod 777 *facepalm*
- Zugriffskontrolle? Rechte?
- auch Zugriff auf Uploads-Verzeichnis
- http://www.trullala.de/du-willst-kein-ftp-zugang-zu-deiner-website/

SVN/GIT

- Betreiber gibt Zugang für Entwickler
- Versionskontrolle
- gleichzeitig
- verschlüsselt
- Automatisches Deploy (besonders schön mit eigenem Nameserver)
- Branches / Testing, später Live
- WP-Core im Unterverzeichnis als external (oder Composer), hierzu WP_SITEURL und WP_HOME in Config setzen
- immer ein „Zurück“ möglich
- (WP-)Updates einfach mit „svn up“

Konfiguration

- Konfiguration (Verzeichnisse, Datenbank Zugangsdaten) unterschiedlich von Installation zu Installation
- Problem: wp-config.php Nicht überschreiben!
- Lösung: 12factor-Methode: Konfigurationsdaten werden vom Webserver selbst gestellt https://12factor.net/config & https://roots.io/twelve-factor-03-config/
- URL Rewrite (Daten von mehreren Installationen) bei Umzug (wp-cli) oder Live bei Ausgabe mittels Filter
- WP darf weder wp-config.php noch .htaccess schreiben (Sicherheit)

Daten

- normalerweise „irgendwo mittendrin“ zwischen Config und Code
- besser: getrennt (Code, Dependencies, Daten, Config, siehe 12factor)
- WP darf nur Daten schreiben (in DB und Uploads-Verzeichnis)
- Daten werden nicht ausgeführt (entsprechende Webserver-Config)
- Backup (full, mittels rsync, dedupliziert BTRFS) https://btrfs.wiki.kernel.org/index.php/Incremental_Backup
- Backup-Server macht Pull, nicht WP Push. WP hat keinen Zugriff auf Backups (sicherer)
- Restore (automatisch, getestet, oft benutzt im täglichen Betrieb z.B. für Kopie der Site)
- Verbesserungswürdig: MySQL-Backup mittels mysqldump (nicht dedupliziert)

WP-Admin

- nicht für „Admin“, weil WP (der Webserver) keine Schreibrechte hat.
- Teilweise WP-CLI
- Updates mittels SVN/GIT
- Minor Updates unattended, Major mit Testdeploy

Testdeploys

- Kontrolle des Betreibers nach Programmierarbeiten
- Kontrolle nach Updates
- Einfach, schnell, kein Aufwand
- Automatisch (Puppet, Chef, Ansible oder einfach paar Bash-Scripte)

Live-Deploys

- Genauso wie Test
- Desaster Recovery Plan (wenn das Rechenzentrum abbrennt)
- „Pets vs. Cattle“ (Google this): Webserver sind Cattle
- Irgendwo auf der Welt wird in irgendeinem RZ die Site mittels Klick aus aktuellem Backup wieder automatisch hergestellt.

Performance

Generell immer „Abkürzungen“ nehmen, Aufwand weglassen:

- 64M should be enough for anybody (memory_limit für Öffentlichen Teil der Site, WP-Admin bekommt mehr)
- billige 404er (insbesondere in uploads)
- Nginx & FastCGI Setup (schneller als Apache mit PHP-Modul)
- FastCGI-Cache für Speed und gegen Slashdotting
- GZIP
- Browser Cache
- ObjektCache (Memcache, Redis)

Ansonsten:

- PHP7 (2-5mal so schnell)
- HTTP/2
- keine Schneller-Mach-Plugins und Optimier-Plugins, keine Cache-Plugins
- RAM ist schneller als SSD
- i.d.R. kein CDN notwendig

Testen z.B. mit Google Pagespeed

Monitoring

- High-Level (Codeception, Selenium) vs. Low-Level (Ping)
- Alarm z.B. mit Zabbix

Fazit:

- performant, sicher und zuverlässig
- nachvollziehbar, wiederherstellbar
- aktuelle, sichere Software
- klare Zuständigkeiten und Schnittstellen
- viel automatisiert