Hacking WordPress – Ein Blick hinter die Kulissen

Angreifen und Sichern von WordPress

Hacking WordPress – Ein Blick hinter die Kulissen

Wie Angreifer WordPress-Installationen hacken bzw. Schwachstellen in Plugins, Themes oder Konfigurationen ausnutzen.

1. Hilfe ich wurde gehackt!

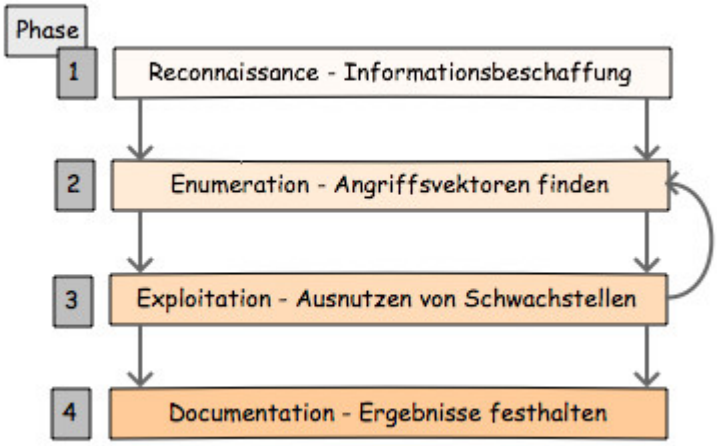
Allein in Deutschland ist der **Verbreitungsgrad** von WordPress enorm – Tendenz weiter steigend. Im weltweiten Vergleich mit anderen Content Management Systemen (CMS) hält WordPress einen Anteil von bis zu 51% (Top Million Sites). Beeindruckende Zahlen also, die WordPress immer stärker in den Fokus von **professionellen Angreifern** rückt. So attackierte ein Botnet im April diesen Jahres WordPress-Installationen weltweit.

Zum Schutz und Absicherung von Installationen wurden bereits zahlreiche Anleitungen veröffentlicht. Empfehlenswert sind »Hardnening WordPress« oder auch meine Artikelserie »WordPress absichern«.

In diesem Beitrag werden allerdings keine weiteren **Schutzmaßnahmen** vorgestellt, sondern wie Angreifer vorgehen, um WordPress-Installationen zu hacken. Es soll ein kleiner Einblick hinter die **Kulissen** sein – in der Realität existieren weitaus mehr Möglichkeiten und Varianten.

2. Hinweis zu »Hacking WordPress«

Der Angriff von WordPress-Installationen oder Systemen ohne Erlaubnis bzw. Einverständniserklärung stellt eine **strafbare Handlung** dar. Wer ohne vertragliche Grundlage fremde Systeme angreift begibt sich auf sehr dünnes Eis. Die nachfolgenden Informationen dienen der Aufklärung und sollten lediglich im Rahmen eines Penetrationstests Verwendung finden. Im Gegensatz zu illegalen Hacking-Angriffen stellt ein Penetrationstest ein **auftragsgesteuerter** Einbruch in ein oder mehrere Systeme dar. Das Vorgehen dient im Grunde der »Qualitätskontrolle« der aktuell umgesetzten IT-Sicherheit im Unternehmensumfeld.



Ein Angriff / Penetrationstest lässt sich in unterschiedlichen **Phasen** unterteilen, von denen ein Teil sequentiell wiederholt wird. Phase 1 dient zunächst der **Informationsgewinnung** über das Ziel. Während ein Penetrationstester in Phase 4 die Ergebnisse festhält, wird sich ein Angreifer diesen Schritt wohl eher sparen…

3. Informationsgewinnung – Phase 1

Im ersten Schritt wird ein Angreifer möglichst viele **Informationen** über sein Ziel sammeln, die für den weiteren Verlauf von Interesse sein können. Zu diesem Zweck werden verschiedene öffentlich verfügbare Informationsquellen durchsucht. Diese werden im Anschluss ausgewertet und sollen Aufschluss darüber geben, über welchen Weg ein Einbruch am **einfachsten** realisiert werden kann. Für diesen Zweck stehen unterschiedliche Tools zur Verfügung – die meisten davon befinden sich auf der Linux Distribution Kali. Die Distribution wird sowohl von **Hackern**, als auch von **Penetrationstestern** zur Auffindung von Schwachstellen / Sicherheitsanalysen eingesetzt.

Dabei helfen Tools die unter »Information Gathering« zusammengefasst sind. Letztendlich werden in der ersten Phase folgende Ziele verfolgt:

- Ziel identifizieren
- System / Anwendungsversion bestimmen
- Verfügbare Netzwerk-Ports
- Laufende Services
- Verteidigungsstrategien erkennen
- […]

Du kannst den Blog aktiv unterstützen!

No Tracking. No Paywall. No Bullshit.

Die Arbeit von kuketz-blog.de finanziert sich zu 100% aus den Spenden unserer Leserinnen und Leser. Werde Teil dieser Community und unterstütze auch du unsere Arbeit mit deiner Spende.

Mitmachen →

3.1 Beispiel: WordPress Identifikation

Das Verstecken der **WordPress-Versionsnummer** oder sonstigen **Meta-Daten** wird bei Laien oftmals mit dem Schutz gegen Spambots oder Sicherheitslücken in Verbindung gebracht. In der Tat lassen sich damit die besonders »dämlichen« Bots an der Nase herumführen, aber bereits semi-professionelle Varianten lassen sich von den Security by Obscurity Maßnahmen nicht beirren. Sie benutzen ausgeklügelte Methoden zur Feststellung ob eine Seite mit WordPress betrieben wird.


```
[*] Num of checks set to: 100

------------------------------------------------------
[*] Input plugin list set to: wp_plugin_list_2013_feb.txt
[*] Num of threats set to: 10
------------------------------------------------------

==> Results for: http://[redacted] <==

[i] Wordpress version found:  3.5.2
[i] Wordpress last public version: 3.5.2


[*] Search for installed plugins


[i] Plugin found: google-sitemap-generator
    |_Latest version:  3.2.9
    |_ Installed version: 3.2.8

[i] Plugin found: jetpack
    |_Latest version:  2.1.2
    |_ Installed version: 2.3.1
    |_CVE list:
    |___CVE-2011-4673: (http://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2011-4673)

[i] Plugin found: si-contact-form
    |_Latest version:  3.1.8.1
    |_ Installed version: trunk

[i] Plugin found: wp-pagenavi
    |_Latest version:  2.83
    |_ Installed version: 2.83
```

Wer selbst mal schauen möchte ob seine WordPress-Installation als solche erkannt wird kann folgende Webseite nutzen: [Is it WordPress?]

Mehr Informationen benötigt? Beispielsweise alle installierten Plugins? Auch gar kein Problem mit dem Tool [plecost]. Hier ein Fingerprint einer WordPress-Installation:

Mit Hilfe der gesammelten Informationen lässt sich WordPress bzw. eines der installierten Plugins gezielt angreifen. Details zu Schwachstellen für bestimmte Versionen stellt beispielsweise CVE-Details zur Verfügung.

3.2 Beispiel: System identifizieren

Nmap ist ein Werkzeug zum **Scannen** und **Auswerten** von Hosts in einem Netzwerk und fällt in die Kategorie der Portscanner. Der Name steht für Network Mapper. Nmap wird in erster Linie für Portscanning eingesetzt. Daneben verfügt es über weitere Techniken, wie beispielsweise die Erkennung des eingesetzten Betriebssystems (OS-Fingerprinting).

Letztendlich dienen solche Informationen wiederum als **Ausgangspunkt** für die weiteren Phasen, in denen Schwachstellen aktiv ausgenutzt werden.

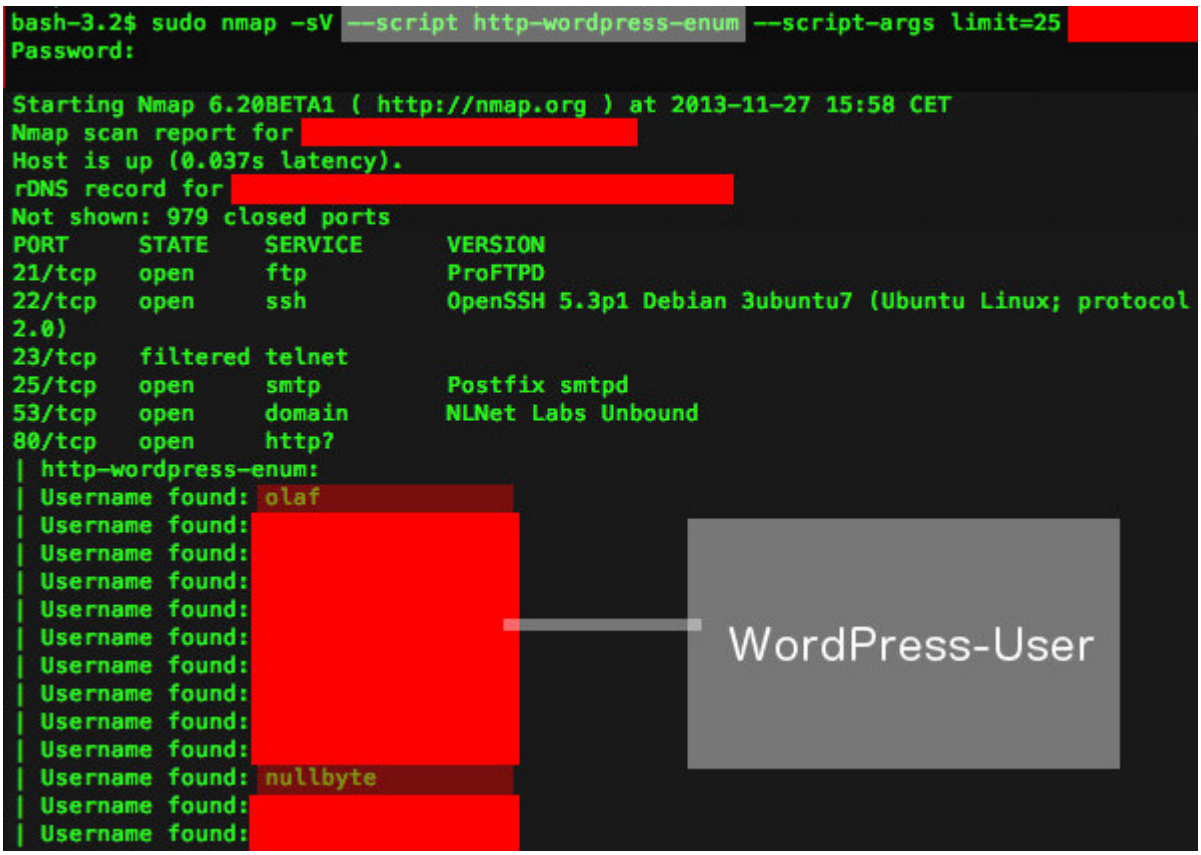
3.3 Beispiel: Erkennung von Benutzer-Accounts

Um sich in den **Administrationsbereich** von WordPress einzuloggen ist die Kombination aus einem Benutzernamen und Passwort erforderlich. Falls ein Angreifer im Vorfeld den Benutzernamen »erraten« kann, benötigt er im Anschluss lediglich das korrekte Passwort. Insgesamt erleichtert das ein erfolgreiches Eindringen in den sensiblen Administrationsbereich.

Oft genügt dazu die Eingabe von

wordpress-blog-adress.de/?author=1

in die Browser-Zeile. In der Standard-Installation bekommt ein Administrator / Nutzer eine eindeutige **Identifikationsnummer** zugewiesen. Meist endet diese auf **author=1** bzw. kann durch den Austausch der 1 am Ende leicht durchprobiert werden.



Falls der WordPress-Betreiber dies manuell geändert hat hilft ein Skript für nmap – wer probiert schon gerne alle

Kombinationen durch:

4. Angriffsvektoren finden – Phase 2

Ausgehend von den in Schritt eins gesammelten Informationen werden anschließend mögliche **Einstiegspunkte** in das System identifiziert. Mit Hilfe von Tools und manuellen Abfragen wird konkrekt nach Schwachstellen und Lücken gesucht, die einen Einbruch ermöglichen. Unter »Vulnerability Analysis« sind die benötigten Tools zusammengefasst und dienen folgenden Zielen:

- Schwachstellen identifizieren
- Identifizieren und priorisieren von System Zugangspunkten
- Risiken einschätzen
- […]

WordPress auf Schwachstellen und Konfigurationsfehler prüfen

Für Deine WordPress-Installation habe ich ein **spezielles** Leistungspaket im Angebot:

- Scan Deiner WordPress-Installation auf Schwachstellen
- Auswertung und Beurteilung der gefundenen Schwachstellen
- Auf Basis der Ergebnisse erhälst Du von mir individuelle Maßnahmenempfehlungen zur Behebung und Absicherung

Wenn du Deine WordPress-Installation **nachhaltig** absichern möchtest, kannst Du mich gerne kontaktieren.

Gut zu wissen: Sicherheit erlangst Du nicht durch die Installation unzähliger Security-Plugins, sondern durch eine saubere Konfiguration, stetige Updates und proaktive Maßnahmen

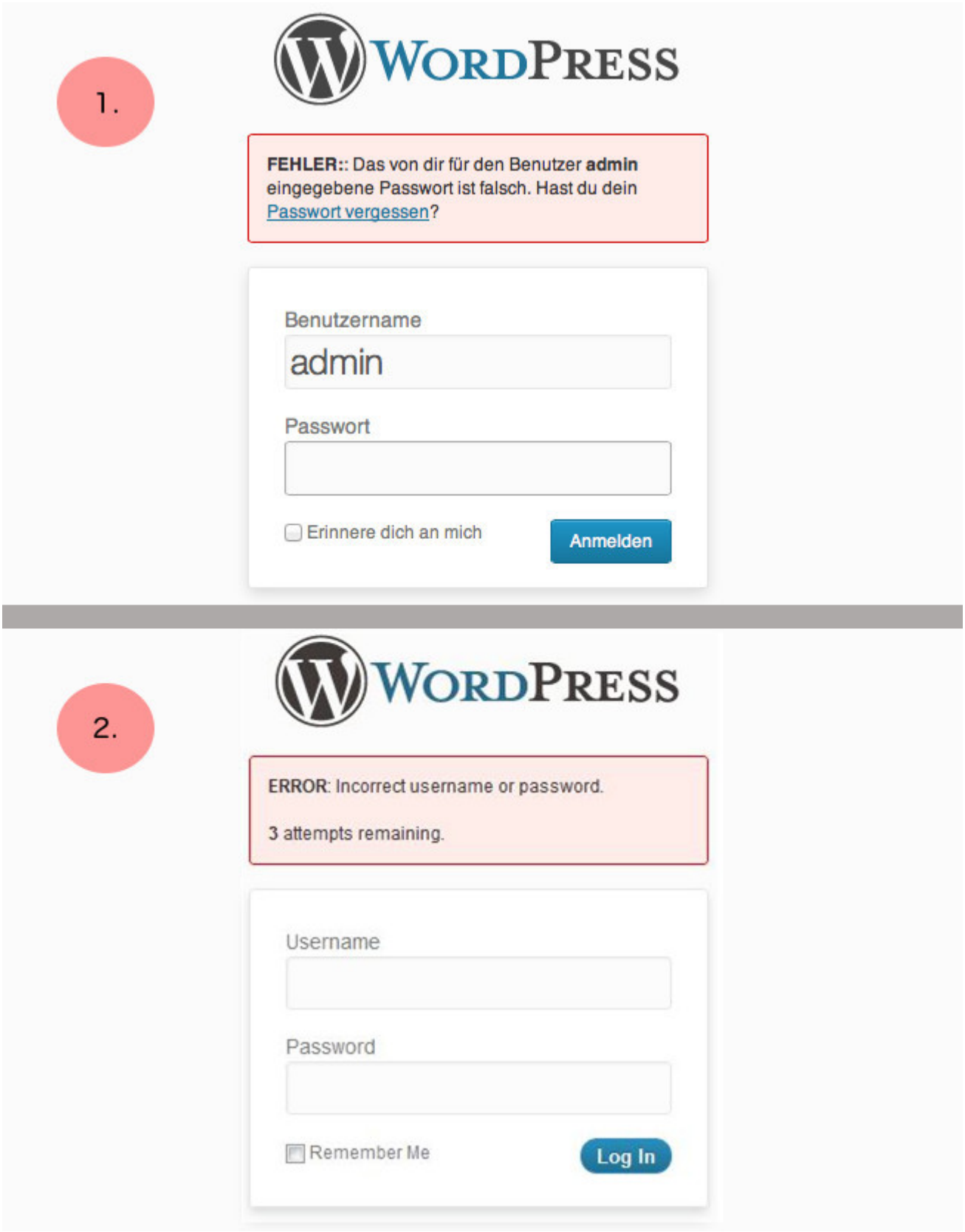
zur Absicherung. Kontakt aufnehmen

4.1 Administrationsbereich

Äußert beliebt als Einstiegspunkt ist der Login zum **Administrationsbereich** von WordPress – nicht zuletzt deswegen, weil sich ein Angriff bei vielen Installationen mit einfachen Mitteln bewerkstelligen lässt.

Über den Browser lässt sich prüfen, ob der Administrationsbereich generell für jeden erreichbar ist:

wordpress-blog-adress.de/wp-admin



Nach der Eingabe eines Benutzernamens und Passwort kann zunächst die Reaktion von WordPress erkundet werden.

Im ersten Beispiel wird der Account »**admin**« bestätigt. Dieser ist also vorhanden und dient vermutlich der Administration der

WordPress-Installation.

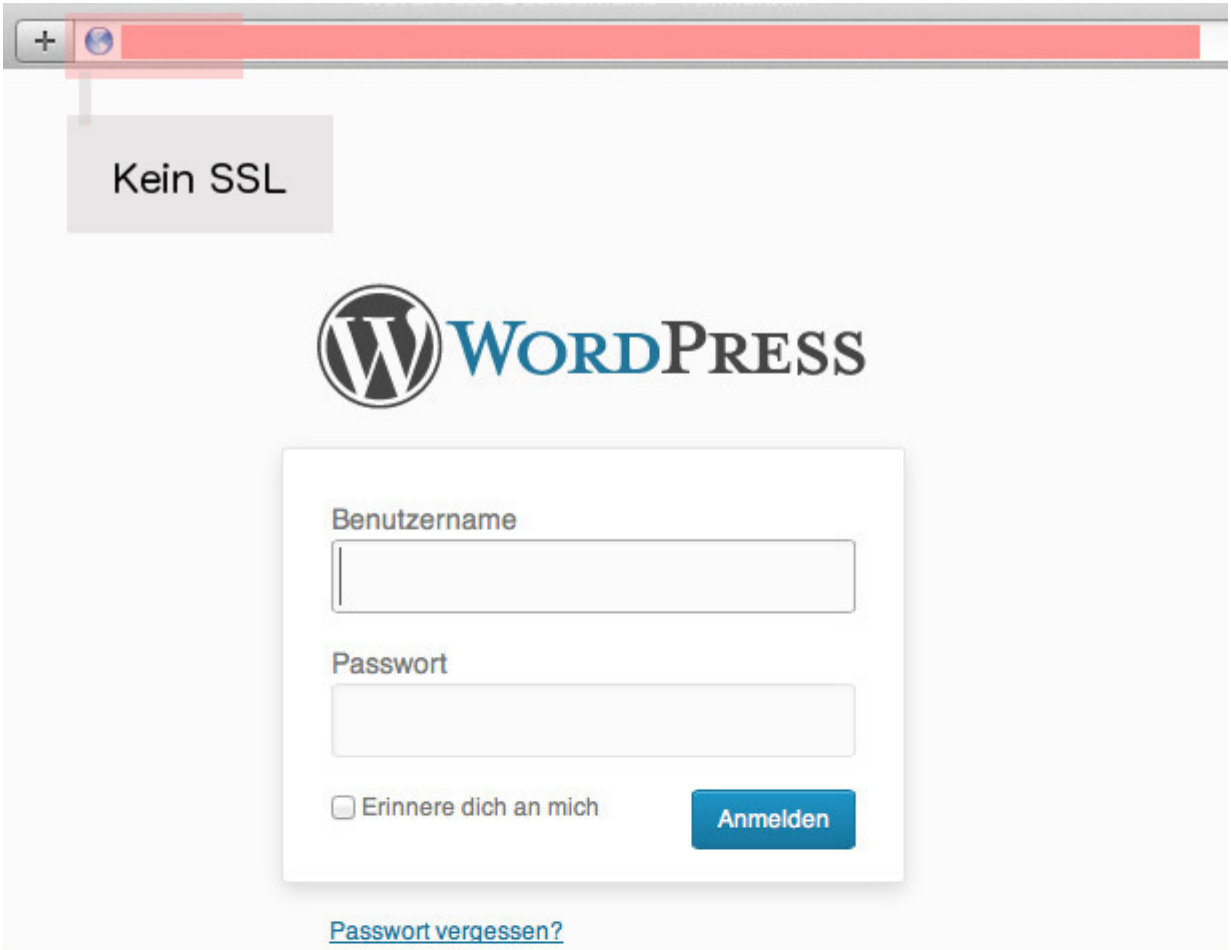
Aus Beispiel zwei lässt sich die Verwendung eines Security-Plugins ableiten. Vermutlich kommt hier Login LockDown / Limit Login Attempts oder ein ähnliches Plugin zum Einsatz. Diese protokollieren fehlgeschlagene **Login-Versuche**. Falls ein Anmeldeversuch innerhalb von 5 Minuten dreimal hintereinander fehlschlägt blockiert das Plugin die anfragende IP-Adresse beispielsweise für eine Stunde. Script-Kiddies und dämliche Bots lassen sich von solchen Maßnahmen abschrecken – professionelle Angreifer hingegen weniger.

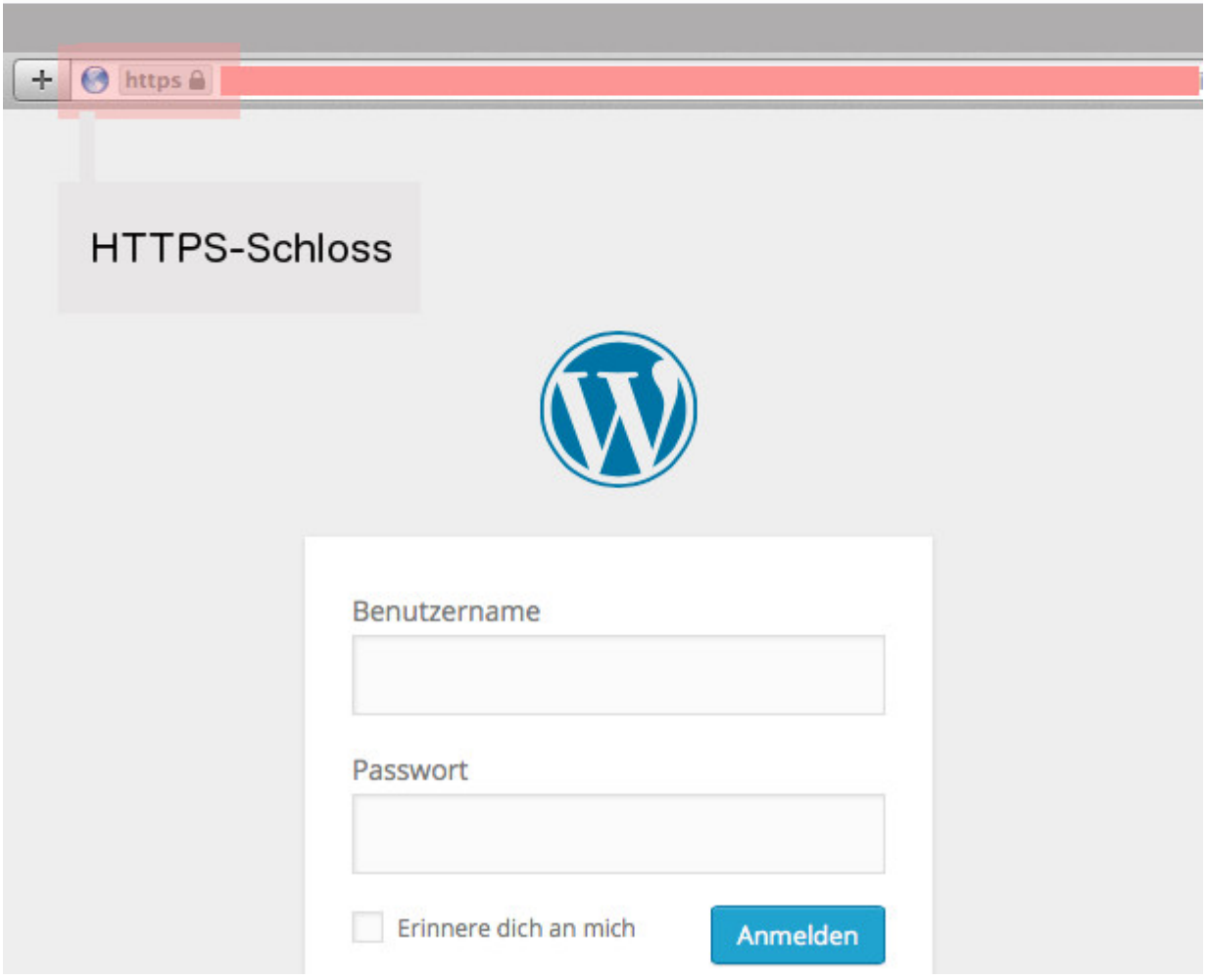
4.2 Fehlende SSL-Verschlüsselung

Hauptsächlich wird SSL für die **Absicherung** zwischen Webbrowser und Webserver eingesetzt – also immer dann, wenn sensible Informationen über das **unsichere** Internet ausgetauscht werden sollen.

Über den Browser wird abermals der Login zum Administrationsbereich aufgerufen:

wordpress-blog-adress.de/wp-admin


Kein SSL
WordPress
Benutzername
Passwort
Erinnere dich an mich
Anmelden
Passwort vergessen?


https
HTTPS-Schloss
Benutzername
Passwort
Erinnere dich an mich
Anmelden

Falls zwischen Browser und Server keine verschlüsselte SSL-Verbindung ausgehandelt wird, können die **Anmeldedaten** mitgeschnitten werden. Ganz konkret: Ein WordPress-Blogger nutzt das kostenlose **WLAN** in seinem Lieblingskaffee und loggt sich in den Administrationsbereich ein. Da die Verbindung nicht über SSL abgesichert wird, kann einer Angreifer die Anmeldedaten im Klartext bzw. unverschlüsselt mitlesen. Solch ein Angriff ist mit einfachen Mitteln bereits von Anfängern durchführbar.

5. Ausnutzen von Schwachstellen – Phase 3

Gefundene Schwachstellen gilt es in Phase 3 gezielt auszunutzen. Dafür werden vorhandene Exploits verwendet oder neue entwickelt, die es ermöglichen Systeme zu **kompromittieren**. Falls in ein System eingerungen werden kann, ergeben sich aus dem Zugriff oftmals weitere mögliche **Angriffsziele**, die vorher nicht erreichbar waren. Mit der Toolkiste aus »Exploitation Tools« oder »Privilige Escalation« stehen in Kali genügend Mittel zur Verfügung. Verfolgt wird damit:

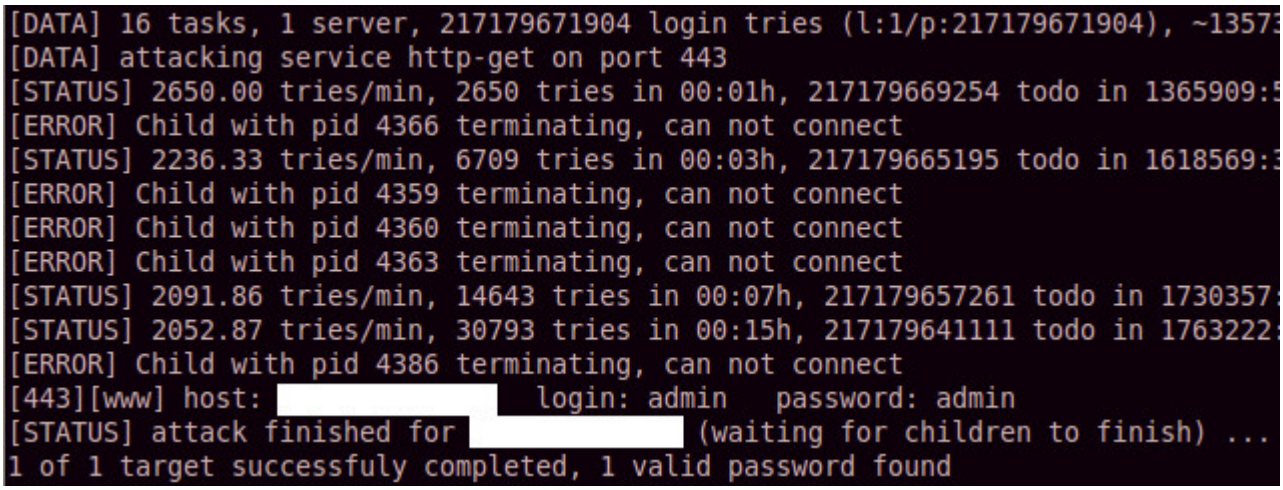
- Schwachstellen in Systemen / Anwendungen ausnutzen
- Systemzugriff erhalten
- Zugang zu geschützten Web-Bereichen
- Erfassen von sensiblen Daten
- […]

5.1 Brute-Force WP-Login

Da Administratoren über die weitreichendsten **Berechtigungen** verfügen, stellen sie ein beliebtes Ziel für Angreifer dar. Einmal eingeloggt erlauben Sie beispielsweise das Hinzufügen von schädlichen PHP- oder Javascript-Befehlen direkt über das

Dashboard. In der Informationsphase wurden bereits Anmeldeinformationen gesammelt, die gezielt für den Einbruch in das Backend genutzt werden können.

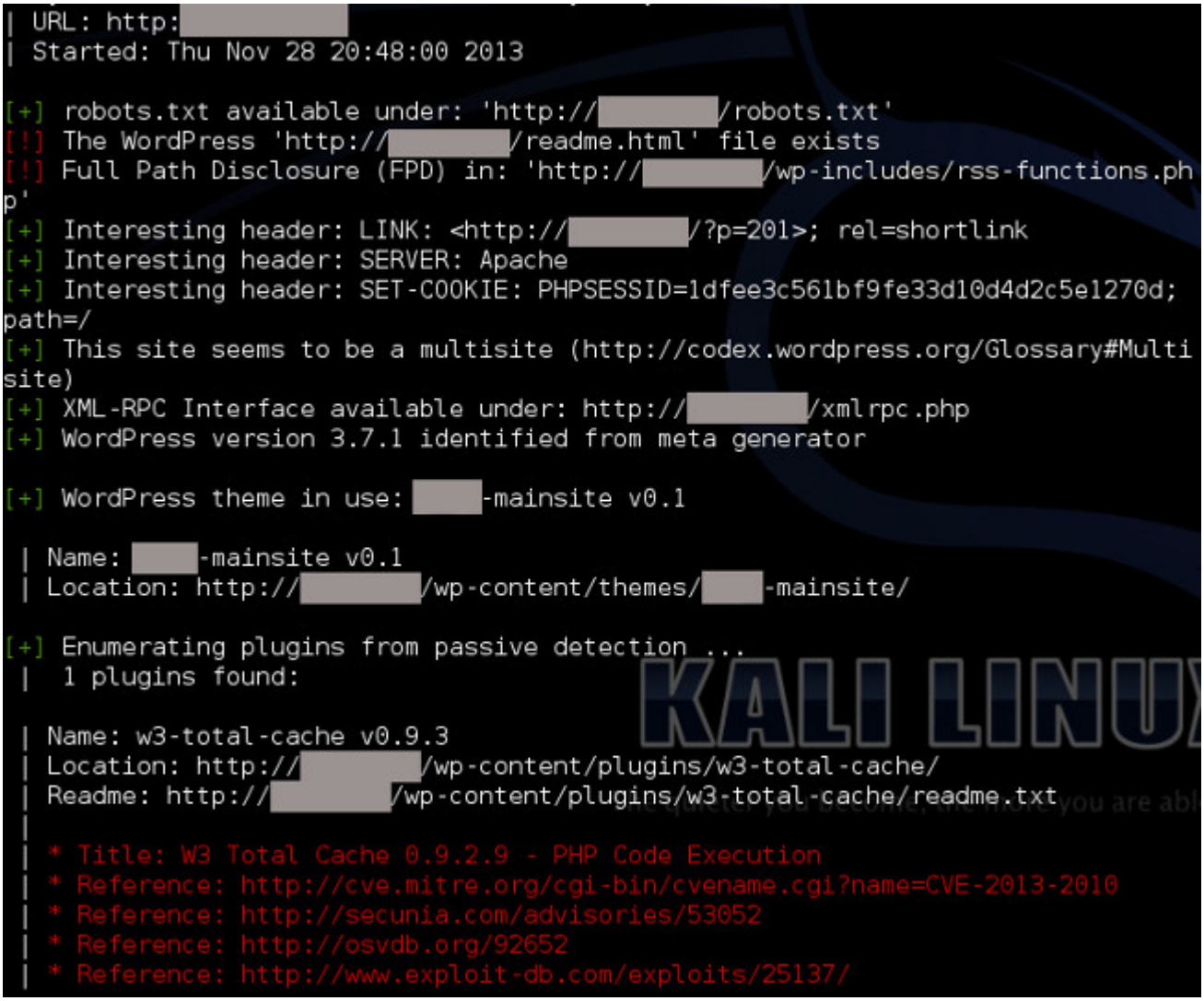
Geschützt wird der Administrationsbereich aus einer **Kombination** von Benutzername und Passwort. Falls ein Angreifer bereits über den Benutzernamen verfügt, so muss er im nächsten Schritt das Passwort »erraten«. Mittels einem Brute-Force-Angriff wird durch Ausprobieren das passende Passwort ermittelt. In freier Wildbahn führt dieser Angriff oft zum Erfolg, da viele Anwender noch immer unsichere Passwörter verwenden.



Speziell für diesen Zweck steht Hydra zur Verfügung. Neben WordPress-Installationen kann damit eine breite Palette von Systemen und Anwendungen angegriffen werden.

5.2 Das Tool WPScan

WPScan ist speziell auf WordPress zugeschnitten. Es bietet zahlreiche Funktionen, wie beispielsweise die **Erkennung** der installierten Plugins, Themes und WordPress-Versionen. Des Weiteren ist es in der Lage Benutzer-Accounts für Brute-Force-Angriffe zu »erraten« und verweist direkt auf **Schwachstellen-Datenbanken**, falls während des Scans auffällige Plugins gefunden werden. Im Beispiel wird eine Lücke im Plugin W3 Total Cache (Version 0.9.3) detektiert.



5.3 Metasploit

Metasploit ist eine Art **Allzweckwaffe** bzw. große Toolbox für Penetrationstests und Sicherheitsanalysen. Es besteht aus unterschiedlichen Teilbereichen, Teilprojekten und Modulen – der Umfang erlaubt den Einsatz in allen **Phasen** eines Penetrationstests. Auch Angreifer machen sich Metasploit zu Nutze, um in fremde Systeme einzudringen. Hier lediglich ein kurzer Einblick in das Metasploit Universum.

Das Metasploit Modul »**wordpress_login_enum**« dient zur Feststellung von gültigen Benutzer-Accounts und kann im Anschluss einen Passwort-Rate-Angriff durchführen.

```
msf > use auxiliary/scanner/http/wordpress_login_enum
msf auxiliary(wordpress_login_enum) > show options

Module options (auxiliary/scanner/http/wordpress_login_enum):

   Name               Current Setting  Required  Description
   ----               ---------------  --------  -----------
   BLANK_PASSWORDS    true             yes       Try blank passwords for all users
   BRUTEFORCE         true             yes       Perform brute force authentication
   BRUTEFORCE_SPEED   5                yes       How fast to bruteforce, from 0 to 5
   PASSWORD                            no        A specific password to authenticate
   PASS_FILE                           no        File containing passwords, one per
   Proxies                             no        Use a proxy chain
   RHOSTS                              yes       The target address range or CIDR id
   RPORT              80               yes       The target port
   STOP_ON_SUCCESS    false            yes       Stop guessing when a credential wor
   THREADS            1                yes       The number of concurrent threads
   URI                /wp-login.php    no        Define the path to the wp-login.php
   USERNAME                            no        A specific username to authenticate
   USERPASS_FILE                       no        File containing users and passwords
   USER_FILE                           no        File containing usernames, one per
   VALIDATE_USERS     true             yes       Enumerate usernames
   VERBOSE            true             yes       Whether to print output for all att
   VHOST                               no        HTTP server virtual host
```

```
msf auxiliary(wordpress_login_enum) > set URI /wordpress/wp-login.php
URI => /wordpress/wp-login.php
msf auxiliary(wordpress_login_enum) > set PASS_FILE /tmp/passes.txt
PASS_FILE => /tmp/passes.txt
msf auxiliary(wordpress_login_enum) > set USER_FILE /tmp/users.txt
USER_FILE => /tmp/users.txt
msf auxiliary(wordpress_login_enum) > set RHOSTS 192.168.1.201
RHOSTS => 192.168.1.201
msf auxiliary(wordpress_login_enum) > run

[*] http://192.168.1.201:80/wordpress/wp-login.php - WordPress Enumeration - Runnin
[*] http://192.168.1.201:80/wordpress/wp-login.php - WordPress Enumeration - Checki
[-] http://192.168.1.201:80/wordpress/wp-login.php - WordPress Enumeration - Invali
[*] http://192.168.1.201:80/wordpress/wp-login.php - WordPress Enumeration - Checki
[+] http://192.168.1.201:80/wordpress/wp-login.php - WordPress Enumeration- Usernam
[*] http://192.168.1.201:80/wordpress/wp-login.php - WordPress Enumeration - Checki
[-] http://192.168.1.201:80/wordpress/wp-login.php - WordPress Enumeration - Invali
[*] http://192.168.1.201:80/wordpress/wp-login.php - WordPress Enumeration - Checki
[-] http://192.168.1.201:80/wordpress/wp-login.php - WordPress Enumeration - Invali
[+] http://192.168.1.201:80/wordpress/wp-login.php - WordPress Enumeration - Found
[*] http://192.168.1.201:80/wordpress/wp-login.php - WordPress Brute Force - Runnin
[*] http://192.168.1.201:80/wordpress/wp-login.php - WordPress Brute Force - Skippi
[*] http://192.168.1.201:80/wordpress/wp-login.php - WordPress Brute Force - Trying
[-] http://192.168.1.201:80/wordpress/wp-login.php - WordPress Brute Force - Failed
[*] http://192.168.1.201:80/wordpress/wp-login.php - WordPress Brute Force - Trying
[-] http://192.168.1.201:80/wordpress/wp-login.php - WordPress Brute Force - Failed
[*] http://192.168.1.201:80/wordpress/wp-login.php - WordPress Brute Force - Trying
[-] http://192.168.1.201:80/wordpress/wp-login.php - WordPress Brute Force - Failed
[*] http://192.168.1.201:80/wordpress/wp-login.php - WordPress Brute Force - Trying
[-] http://192.168.1.201:80/wordpress/wp-login.php - WordPress Brute Force - Failed
[*] http://192.168.1.201:80/wordpress/wp-login.php - WordPress Brute Force - Trying
[+] http://192.168.1.201:80/wordpress/wp-login.php - WordPress Brute Force - SUCCES
[*] Scanned 1 of 1 hosts (100% complete)
[*] Auxiliary module execution completed
msf auxiliary(wordpress_login_enum) >
```

6. Weitere Möglichkeiten

Die oben dargestellten Tools und Möglichkeiten stellen lediglich eine Mini-Auswahl dar. In der **Praxis** existieren unzählige Tools und Varianten, Webanwendungen und deren Host-Systeme zu hacken. Allein in den Datenbanken von Metasploit und exploit-db.com sind hunderte von **Schwachstellen** erfasst und beschrieben. Immer wieder Ziel sind Plugins, Themes und der WordPress-Kern selbst.

Hinweis

Auch von deaktivierten Plugins oder Themes geht eine Gefahr aus. Selbst wenn sie nicht aktiv verwendet werden, so sind sie im Normalfall dennoch erreichbar. Beispielsweise erlaubt das Plugin Asset-Manager (Version <= 2.0) einen Datei-Upload in ein temporäres Verzeichnis – anschließend kann darüber Schadcode ausgeführt werden. Für diesen Einbruch muss das Plugin nicht aktiv sein, sondern lediglich auf dem Webspace vorhanden.**Lücke**: WordPress Asset-Manager PHP File Upload Vulnerability.

6.1 Angriffe auf Systemebene

Allein für Phase 1 (**Informationsbeschaffung**) wird ein Angreifer viel Zeit aufwenden, um an Daten / Informationen zu gelangen, die ihm später nützlich sein können. Immerhin hängt davon indirekt der Erfolg für den späteren Einbruch ab. Bereits einfache Wege wie, Google-Hacking (Dorks), DNS-Informationen und soziale Netzwerke stellen wichtige Informationsquellen dar. Daraus lassen sich oftmals Informationen ableiten, die entscheidende Hinweise für einen erfolgreichen Angriff bieten. Womöglich bietet eine WordPress-Installation selbst keinen **Angriffspunkt**, was den Fokus auf das Host-System richtet. Als Beispiel:

- MySQL-Datenbank

- FTP / SSH Service
- CPanel oder andere Tools für die web-basierte Administration
- phpMyAdmin Zugänge
- […]

Für die Sicherheit von WordPress müssen alle **Zahnräder** ineinandergreifen – letztendlich hat ein Angreifer immer das Ziel das schwächste Zahnrad auszumachen.

7. Fazit

Der Artikel WordPress-Hacking soll einen **Eindruck** über den Ablauf eines Angriffs vermitteln – auch wenn die Phasen leicht vermischt sind. Angreifer verfolgen damit meist unterschiedliche Ziele. Oftmals dienen infizierte WordPress-Installationen als **Ausgangspunkt** für weitere Angriffe oder zum Versenden von Spam-Mails. Neben Vandalismus und Rachegelüste sind praktisch unzählige Absichten denkbar.

Wenn eure WordPress-Installation selbst schon gehackt wurde oder ihr die Sicherheit im Vorfeld verbessern wollt, dann empfehle ich nochmals folgende Anleitungen: »Hardnening WordPress« und meine Artikelserie »WordPress absichern«.

Bildquellen:

Skull: „#9358035“, https://de.fotolia.com/id/9358035

Über den Autor | Kuketz

In meiner freiberuflichen Tätigkeit als Pentester / Sicherheitsforscher (Kuketz IT-Security) schlüpfe ich in die Rolle eines »Hackers« und suche Schwachstellen in IT-Systemen, Webanwendungen und Apps (Android, iOS). Des Weiteren bin ich **Lehrbeauftragter** für IT-Sicherheit an der dualen Hochschule Karlsruhe, schärfe durch Workshops und Schulungen das

Sicherheits- und **Datenschutzbewusstsein** von Personen und bin unter anderem auch als Autor für die Computerzeitschrift c’t tätig.

Der Kuketz-Blog bzw. meine Person ist regelmäßig in den Medien (heise online, Spiegel Online, Süddeutsche Zeitung etc.) vertreten.

Mehr Erfahren →