

# **Geheime Profi-Tricks für Schnellere Websites!**

**Peter Debik M.A.**

In zahlreichen Abschnitten dieses Buchs werden Beispiele zu Skripten und Serverbefehlen gezeigt. Alle Beispiele stammen aus der Praxis und sind getestet. Manche Befehle sind mächtige Werkzeuge, die mit wenigen Buchstaben die Einstellungen einer Software so verändern können, dass sie nicht mehr fehlerfrei arbeitet. Informieren Sie sich deshalb stets vor der Nutzung eines Befehls, wie der Befehl genau funktioniert, welche Optionen er hat (z.B. bestimmte Parameter, die sein Verhalten steuern) und was er bewirkt. Besonders hilfreich ist dazu der Linux-Befehl `man` (=“Manual“, zu Deutsch „Handbuch“), mit dem Sie sich viele wichtige Befehle erklären lassen können.

Wenn Sie sich nicht sicher sind, welche Auswirkungen eine bestimmte Einstellung hat, führen Sie sie nicht aus oder führen Sie sie nur auf einem Computer aus, auf dem Sie alles risikofrei testen können.

Die Informationen in diesem Buch wurden sorgfältig erarbeitet und überprüft. Dennoch können Fehler nicht vollständig ausgeschlossen werden. Verlag und Autor übernehmen keine juristische Verantwortung oder Haftung für vielleicht verbliebene Fehler und deren Folgen.

Alle im Text genannten Marken sind vielleicht eingetragene Marken ihrer jeweiligen Eigentümer. Sie werden hier nicht im Sinne einer markenmäßigen Nutzung wiedergegeben, sondern zu Referenzzwecken im Sinne einer technisch-wissenschaftlichen Arbeit. Das Werk einschließlich aller seiner Teile ist urheberrechtlich geschützt. Alle Rechte vorbehalten einschließlich der Vervielfältigung, Übersetzung, Speicherung und Verarbeitung in elektronischen Systemen sowie Mikroverfilmung.

Geheime Profi-Tricks für schnellere Websites!  
ISBN 3-98050723-8  
EAN 9783980507233

Copyright © 2014 Bitpalast GmbH. Alle Rechte vorbehalten.  
Hintergrundbild des Umschlags: © alphaspirt - Fotolia.com  
1. Auflage 2014

Lektorat, Satz, Gestaltung: Bitpalast GmbH.

**Bitpalast**  


# Inhalt

---

<b>Vorwort .....</b>	<b>IX</b>
<b>1 Verrückt nach Geschwindigkeit.....</b>	<b>1</b>
<b>1.1 Schnell ist noch zu langsam.....</b>	<b>2</b>
<b>1.2 Was bremst Websites? .....</b>	<b>4</b>
1.2.1 Domain-Namensauflösung.....	6
1.2.2 Transportweg zwischen Webserver und Webbrowser.....	7
1.2.3 Anzahl der Prozesse und CPU-Last .....	9
1.2.3.1 Mehrere Befehle gleichzeitig verarbeiten? .....	9
1.2.3.2 Was ist CPU-Last und wie misst man sie? .....	11
1.2.3.3 CPU-Last Headroom beachten .....	13
1.2.3.4 CPU-Zeit sparen .....	13
1.2.4 Andere Nutzer auf der gleichen Maschine .....	14
1.2.5 Datenbank-Abfragen .....	14
1.2.6 Datenmenge und Website-Programme.....	15
1.2.7 Persönliches Empfinden der Anwender .....	16
<b>2 So geht's schneller .....</b>	<b>19</b>
<b>2.1 Systemumgebung.....</b>	<b>21</b>
2.1.1 Unnötige Dienste (Daemons) abschalten .....	21
2.1.2 RAM vergrößern.....	23
2.1.3 Schnelle Festplatten nutzen.....	24
2.1.4 Gegen DoS und Einbruchversuche schützen .....	25
2.1.4.1 Firewall einsetzen .....	25
2.1.4.2 Angreifende IP-Adressen blockieren .....	26
2.1.4.3 SSH-Zugang auf bestimmte IP-Adressen und Nutzer beschränken.....	28
2.1.5 Leistung auf mehrere Hosts verteilen .....	29
2.1.5.1 Cloud-Computing macht Ihre Website nicht schneller .....	29
2.1.5.2 Webserver-Gruppe einrichten .....	30
2.1.5.3 Lastverteilung per DNS oder Hardware-Lastverteiler verwenden.....	31
2.1.5.4 Mailserver vom Webserver trennen.....	32
2.1.5.5 Datenbankserver vom Webserver trennen.....	34
2.1.5.6 Mehrere Datenbankserver einsetzen .....	35
2.1.5.7 Mehrere Konzepte zur Lastverteilung kombinieren.....	35

<b>2.2</b>	<b>E-Mail-Konfiguration.....</b>	<b>36</b>
2.2.1	local-host-names einschränken.....	36
2.2.2	Sendmail Timeouts verringern.....	37
2.2.2.1	Konfiguration vorübergehend direkt in sendmail.cf ändern.....	39
2.2.2.2	Konfiguration dauerhaft in der m4-Makro-Datei ändern.....	40
2.2.3	Viren-, Spamfilter und procmail abschalten.....	41
<b>2.3</b>	<b>Webserver-Konfiguration.....</b>	<b>44</b>
2.3.1	Nginx statt Apache einsetzen.....	44
2.3.2	Unnütze Module abschalten.....	46
2.3.3	Kommentarzeilen aus httpd.conf entfernen.....	48
2.3.4	Host Name Lookups abschalten.....	49
2.3.5	Test auf symbolische Links abschalten.....	49
2.3.6	.htaccess-Befehle nach httpd.conf verschieben.....	50
2.3.7	Verschlüsselung nur verwenden, wo nötig.....	51
2.3.7.1	Falls TLS verwendet wird.....	51
2.3.7.2	Falls auf TLS verzichtet wird.....	53
2.3.8	Datenkompression nutzen.....	54
2.3.8.1	Texte komprimieren.....	55
2.3.8.2	Bilder und andere Binärdaten nicht komprimieren.....	56
2.3.8.3	Mit „Deflate“ im Webserver den Datenverkehr komprimieren.....	57
2.3.9	Log-Dateien sinnvoll nutzen.....	58
2.3.9.1	Unnütze Log-Dateien abschalten.....	58
2.3.9.2	Log-Dateien häufig rotieren.....	58
<b>2.4</b>	<b>Crontab- und atd-Zeitplanung.....</b>	<b>60</b>
2.4.1	Aufträge ohne Wildcard planen.....	61
2.4.2	Aufträge nur ausführen, wenn es wirklich nötig ist.....	63
2.4.3	Termin-Aufträge nur bei niedriger CPU-Last ausführen.....	64
<b>2.5</b>	<b>MySQL-Datenbank.....</b>	<b>65</b>
2.5.1	Gegen Angriffe schützen.....	65
2.5.1.1	Nur Zugriffe von localhost erlauben.....	65
2.5.1.2	Zugriffe von anderen Hosts erlauben, aber beschränken.....	66
2.5.1.3	Nutzer auf bestimmte Server beschränken.....	66
2.5.2	MyISAM statt InnoDB verwenden.....	67
2.5.3	Konfiguration optimieren.....	69
2.5.3.1	Aktuelle Betriebsdaten ermitteln.....	69
2.5.3.2	Konfigurationswerte setzen.....	71
2.5.4	Indizes verwenden, aber deren Länge begrenzen.....	72
2.5.5	Tabellenstruktur und Abfragen optimieren.....	74
2.5.5.1	Tabellen mit „optimize“ verbessern.....	74
2.5.5.2	Platzsparende Datentypen verwenden, „procedure analyse“ nutzen.....	74
2.5.5.3	Datentyp „ENUM“ häufig verwenden.....	75
2.5.5.4	Datenfelder und Zeichensätze so kurz wie möglich halten.....	76
2.5.5.5	Nur die Felder abrufen, die benötigt werden.....	78
2.5.5.6	"Order by" sparsam verwenden.....	78
2.5.5.7	"Limit" häufig verwenden.....	79
2.5.5.8	"Inner Join" statt anderer Joins verwenden.....	80

2.5.5.9	„IN“-Mengenoperator statt „OR“-Verknüpfung verwenden .....	83
2.5.5.10	Mehrere Transaktionen zusammenfassen .....	84
<b>2.6</b>	<b>Gestaltung und Programmierung .....</b>	<b>85</b>
2.6.1	Bild-, Audio- und Videodateien platzsparend speichern .....	85
2.6.1.1	Das richtige Dateiformat für Bilder wählen .....	85
2.6.1.2	Die richtige Kompressionsstufe für Bilder wählen .....	86
2.6.1.3	Bilddateien auf die Anzeigegröße herunterrechnen .....	87
2.6.1.4	Video- und Audio-Dateien vielleicht von externen Websites einbetten ...	88
2.6.2	Fehler vermeiden, Logs abschalten .....	89
2.6.3	Cache und Repositories nutzen .....	90
2.6.3.1	Webseiten-Formatierung in CSS-Stylesheets auslagern .....	90
2.6.3.2	JavaScript in .js-Dateien auslagern .....	92
2.6.3.3	.js-Dateien aus Repositories laden .....	93
2.6.3.4	Falls Caching unerwünscht ist .....	94
2.6.4	HTML-Code kürzen .....	96
2.6.4.1	Formatierungsnamen kürzen .....	96
2.6.4.2	Unnötige Zeichen entfernen, Dateinamen kürzen .....	97
2.6.4.3	Unnötige Eigenschaften weglassen .....	98
2.6.4.4	Wiederkehrende JavaScript-Ereignisse aus HTML-Tags verbannen .....	99
2.6.5	Erst HTML-, danach JavaScript-Code .....	101
2.6.6	Dem Anwender zeigen, wenn etwas passiert .....	102
2.6.7	Asynchrone Kommunikation nutzen .....	103
2.6.7.1	Eigene AJAX-Funktion .....	104
2.6.7.2	AJAX mit JQuery .....	108
2.6.7.3	Anzahl der Serveranfragen begrenzen .....	109
2.6.8	RAM sparsam verwenden .....	111
2.6.9	Datenbanken überlegt verwenden .....	112
2.6.9.1	Zugriffe sparen .....	112
2.6.9.2	PHP Baustein MySQLi statt MySQL verwenden .....	113
2.6.9.3	Persistente Verbindungen nutzen .....	114
2.6.10	Includes selten nutzen .....	114
2.6.11	Verarbeitung frühzeitig abbrechen .....	116
2.6.11.1	Schleifen frühzeitig beenden .....	116
2.6.11.2	Programme vorzeitig beenden – manchmal aber nicht .....	117
2.6.12	Schleifen optimieren .....	119
2.6.12.1	Unnötiges aus Schleifen verbannen .....	120
2.6.12.2	Steuerparameter für For-Next-Schleifen vorab berechnen .....	120
2.6.13	Programm-Befehle optimieren .....	121
2.6.13.1	Fehler abfangen, statt die Fehlerausgabe zu unterdrücken .....	122
2.6.13.2	Einfache statt doppelte Anführungszeichen .....	122
2.6.13.3	Identität statt Gleichheit prüfen .....	123
2.6.13.4	Trinitätsoperator statt If-Then verwenden .....	124
2.6.13.5	Reihenfolge verknüpfter If-Then-Bedingungen sortieren .....	124
2.6.13.6	Unnötige Klammern weglassen .....	125
2.6.13.7	Optionale Parameter weglassen .....	126
2.6.13.8	Variablen als Zeiger an Funktionen übergeben .....	126
2.6.13.9	Reguläre Ausdrücke vermeiden, Arrays beim Ersetzen nutzen .....	128
2.6.13.10	++\$i statt \$i++ oder \$i+=1 verwenden .....	129
2.6.13.11	array_map() statt foreach() verwenden .....	129

2.6.13.12	htmlspecialchars() statt htmlentities() verwenden.....	130
2.6.13.13	echo() statt print verwenden .....	131
2.6.14	DOM-Zugriffe optimieren .....	131
2.6.15	Warteschlangen bilden .....	132
2.6.16	Code minimieren.....	135
2.6.16.1	HTML-Dateien minimieren .....	136
2.6.16.2	CSS-Stylesheets minimieren .....	136
2.6.16.3	JavaScript-Bibliotheken minimieren.....	137
2.6.16.4	PHP-Programme minimieren .....	138
2.6.16.5	Compressor-Programme vereinfachen die Minimierung .....	139
2.6.17	Dynamische Webseiten statisch speichern .....	140
2.6.17.1	Seiten auf dem Server cachen.....	140
2.6.17.2	Varnish-Cache nutzen.....	141
2.6.18	Vor Hackern schützen .....	142
2.6.18.1	Denial-of-Service Angriffe abwehren .....	142
2.6.18.2	Data-Mining abwehren .....	143
<b>3</b>	<b>Anhang .....</b>	<b>145</b>
<b>3.1</b>	<b>Glossar .....</b>	<b>146</b>
<b>3.2</b>	<b>Über den Autor .....</b>	<b>155</b>

# Abbildungen

---

Abbildung 1: CPU-Last Analogie zu Autos, die eine Brücke passieren müssen. ....	11
Abbildung 2: Cloud-Computing Schema.....	30
Abbildung 3: Lastverteilung per DNS oder Hardware-Lastverteiler.....	32
Abbildung 4: Lastverteilung durch Verteilung der Dienste auf mehrere Hosts Ihrer Domain	33
Abbildung 5: Hohe Rechenlast durch Viren- und Spam-Filter .....	42
Abbildung 6: Ein TLS Handshake braucht viele Rechenschritte .....	52
Abbildung 7: Morse-Code nutzt Datenkompression .....	55
Abbildung 8: Gleichzeitig startende Programme erzeugen plötzlich eine hohe CPU-Last....	62
Abbildung 9: Unterschiedliche Skript-Startzeiten entlasten die CPU.....	63
Abbildung 10: phpMyAdmin markiert problematische Datenbank-Werte .....	70
Abbildung 11: Vergleich verlustbehafteter und verlustfreier Bilddaten-Kompression.....	86
Abbildung 12: JPEG-Kompressionsartefakte, Informationsverlust in JPEG-Bildern .....	87
Abbildung 13: Vergleich serieller mit paralleler Datenverarbeitung mit Warteschlangen ....	133