

HOWTO zum Kommando "find"

(C) 2006-2020 T.Birnthaler/H.Gottschalk <howtos(at)ostc.de>
OSTC Open Source Training and Consulting GmbH
http://www.ostc.de

\$Id: find-HOWTO.txt,v 1.43 2020/05/06 06:43:25 tsbirn Exp \$

Dieses Dokument beschreibt die Kommandos "find" und "locate".

INHALTSVERZEICHNIS

- 1) Beschreibung
 - 1.1) Die wichtigsten Bedingungen CONDITION
 - 1.2) Weitere wichtige Bedingungen CONDITION
 - 1.3) Bezugszeitpunkt von Datumvergleichen
 - 1.4) M gliche Aktionen ACTION
 - 1.5) Verkn pfung von Bedingungen CONDITION
 - 1.6) N tzliche "find"-Beispiele
 - 1.7) GNU-"find" kennt weitere Bedingungen CONDITION
 - 1.8) GNU-"find" kennt weitere Aktionen ACTION
 - 1.9) GNU-"find" kennt weitere Verkn pfungen
- 2) Performance-Problem von "find"
- 3) "find"-Ersatz "locate"

1) Beschreibung

Das `find`-Kommando durchsucht Verzeichnisse rekursiv nach Dateien mit bestimmten Eigenschaften. Suchkriterien k nnen neben dem Dateinamen alle (im Inode vorhandenen) Dateiattribute (Metadaten) wie Dateityp, Besitzer, Besitzergruppe, Zugriffsrechte, Dateil nge, Zeitstempel, ... sein. Diese Suchkriterien sind beliebig  ber AND, OR, NOT und Klammerung kombinierbar.

"find" listet entweder alle zu den Bedingungen CONDITION passenden Dateinamen auf (Standardverhalten) oder f hrt beliebige Kommandos ACTION auf ihnen durch. Dazu wird der Verzeichnisbaum ab den angegebenen Verzeichnissen PATH rekursiv durchlaufen (das kann einige Zeit dauern). Die allgemeine Aufrufsyntax lautet:

```
find [PATH...] [CONDITION...] [ACTION...]
```

HINWEIS: Bei fehlendem Suchpfad PATH beginnt die Suche im aktuellen Verz. ".". Bedingungen CONDITION sind per Default UND-verkn pft (d.h. m ssen ALLE erf llt sein), fehlt CONDITION gilt sie als erf llt. Die Standard-Aktion ACTION ist "-print" (d.h. alle ab Verz. PATH zu den Bedingungen CONDITION passenden Dateinamen werden ausgegeben). D.h. ein Aufruf der Form

```
find
```

f hrt zum Auflisten ALLER Verz.- und Dateinamen ab dem aktuellen Verzeichnis.

1.1) Die wichtigsten Bedingungen CONDITION

Bedingung	Bedeutung
-mtime [+] -name "PATTERN" -perm [+] -size [+] ...[GMkwcB] -type TYPE	Inhalt-�nderung vor +mehr/-weniger/genau Tagen [modify] Dateiname passt zu Pattern (* ? [...] ... sch�tzen!) Oktale Rechte (-000=mind, +000 bzw. /000=any, 000=exakt) 1=x=Execute, 2=w=Write, 4=r=Read [permission] Dateigr��e (+mehr/-weniger/genau) in 512-Byte Bl�cken (bzw. G=Giga M=Mega k=Kilo w=2-byte c=byte b=512-Byte) Dateityp (f=file d=directory l=symboliclink D=door c=characterdev. b=blockdev. p=named pipe s=socket)
-group GNAME/GID -user UNAME/UID	Besitzergruppe (Name oder GID) Besitzer (Name oder UID)

Bei Zahlenwerten (DAY, NUM) bedeutet die Zahl:

- +N = Gr  er/mehr als angegebener Wert (gleich nicht enthalten!)
- N = Kleiner/weniger als angegebener Wert (gleich nicht enthalten!)
- N = GENAU der angegebene Wert

Bei "-perm" (RIGHT) bedeutet der Präfix:

- /NNNN = Mind. EINES der angegebenen Rechte NNNN vorhanden (modern)
- +NNNN = Mind. EINES der angegebenen Rechte NNNN vorhanden (veraltet)
- NNNN = Mind. ALLE der angegebenen Rechte NNNN vorhanden
- NNNN = Genau ALLE der angegebenen Rechte NNNN vorhanden

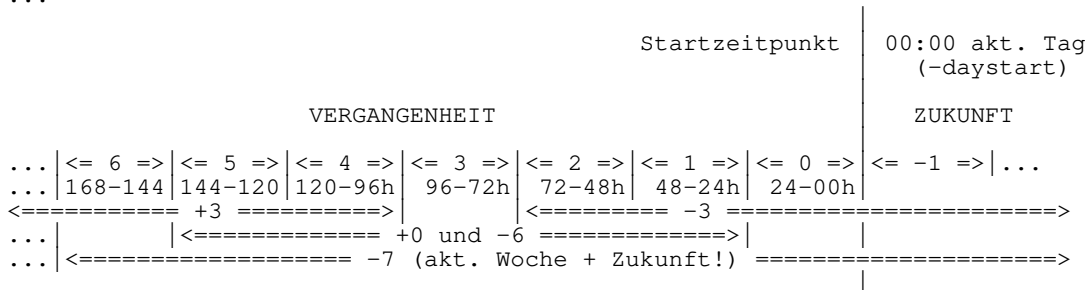
1.2) Weitere wichtige Bedingungen CONDITION

Bedingung	Bedeutung
-atime [+]DAY	Lesender Zugr. vor +mehr/-weniger/genau Tagen [access]
-ctime [+]DAY	Status-Änderung (Inode) vor +mehr/... Tagen [change]
-iname "PATTERN"	Dateinamen-Muster (* ? [...] ... vor Shell schützen!), Groß-/Kleinschreibung egal [ignorecase]
-path "PATTERN"	Pfadnamen-Muster (* ? [...] ... vor Shell schützen!)
-ipath "PATTERN"	Analog, aber Groß-/Kleinschreibung egal [ignorecase]
-inum NUM	Inode-Nummer NUM
-links [+]NUM	Anzahl Hardlinks +mehr/-weniger/genau NUM (Anzahl Namen)
-nogroup	KEINER Gruppe in "/etc/groups" zugeordnet (GID ohne Name)
-nouser	KEINEM Benutzer in "/etc/passwd" zugeord. (UID ohne Name)

1.3) Bezugszeitpunkt von Datumvergleichen

Beim Datumvergleich per -mtime/-atime/-ctime DAY (bzw. -mmin/-amin/-cmin MIN) ist die Basis für die Tages/Minutenanzahl der STARTZEITPUNKT des Kommandos "find" (d.h. eine DYNAMISCHE Definition!). Die Bedeutung der "Tage" DAY bzw. der Minuten MIN ist daher:

- 0 = 0:00:00h-23:59:59h VOR Startzeitpunkt (bzw. vor 00:00 bei -daystart)
- 1 = 24:00:00h-47:59:59h VOR Startzeitpunkt (bzw. vor 00:00 bei -daystart)
- 2 = 48:00:00h-71:59:59h VOR Startzeitpunkt (bzw. vor 00:00 bei -daystart)
- 3 = 72:00:00h-95:59:59h VOR Startzeitpunkt (bzw. vor 00:00 bei -daystart)
- ...



HINWEIS: Option "-daystart" legt die Basis des Zeitvergleichs auf den Beginn 00:00 des aktuellen Tages fest (statt auf den Startzeitpunkt von "find").

1.4) Mögliche Aktionen ACTION

Aktion	Beschreibung
-print	Ausgabe gefundener Dateinamen (Standard!)
-ls	Ausgabe analog "ls -dils" (directory inode long size)
-exec CMD {} \;	Kommando CMD auf gefundenen Dateien ausführen ({} = gef. Dateiname, \; = Kommando-Ende, quotieren wg. Shell)
-ok CMD {} \;	Analog "-exec", verlangt vorher Bestätigung mit "y" (es)
-delete	Datei löschen (Vorsicht!)

1.5) Verknüpfung von Bedingungen CONDITION

Verknüpfung	Beschreibung
\ (... \)	Klammerung (quotieren wg. Shell!)

Apr 25, 24 3:00

find-HOWTO.txt

Page 3/6

\!	Negation (quotieren wg. Shell!)
-a	UND-Verknüpfung (Standard falls keine Verknüpfung angegeben!)
-o	ODER-Verknüpfung

HINWEIS: Die Verknüpfungen sind nach fallender Vorrang angegeben, durch Klammerung lässt sich dieser Vorrang durchbrechen.

1.6) Nützliche "find"-Beispiele

Beispiel	Beschreibung (Std: Ergebnis ausgeben)
<pre>find -name "*.c" find . -name "*.c" find .. -name "*.c" find \! -name "*.c" find ~ -name "*.c" find / -name "*.c" find / -iname "*.c" find / -regex ".*\.c" find / -iregex ".*\.c" find /usr -type d -name "man*" find /usr -type d -regex ".*man.*" find /usr /lib /etc -name "*.o"</pre>	<p>C-Dateien "*.c" ab akt. Verz. "." (analog) C-Dateien "*.c" ab Eltern-Verz. ".." Alle außer-^_er C-Dateien "*.c" ab "." C-Dateien "*.c" ab Heimat-Verz. "~" C-Dateien "*.c" ab Root-Verz. "/" analog + GROSS/kleinschreibung ignor. analog per Regulärem Ausdruck analog + GROSS/kleinschreibung ignor. Verz. in "/usr" mit "man" im Namen analog per Regulärem Ausdruck Objekt-Dateien "*.o" in 3 angeg. Verz.</p>
<pre>find / -mtime 0 find / -mtime 1 find / -mtime -7 find / -mtime +6 -mtime -14 find / \(-mtime 7 -o -mtime 8 \ -o -mtime 9 -o -mtime 10 \ -o -mtime 11 -o -mtime 12 \ -o -mtime 13 \) find / -atime +365 find / -atime +365 -daystart find / -newer FILE</pre>	<p>Heute geänderte Dateien Gestern geänderte Dateien Letzte Woche (Tage 0-6) geänd. Dateien Vorletzte Woche (Tage 7-13) geänd. Dat. (analog: Tage einzeln aufzählen und per ODER verknüpfen) Letzter Zugriff vor mehr als 1 Jahr (analog, aber Startzeitpunkt 00:00) Neuer als Datei FILE</p>
<pre>find / -user tsbirn find / -user root -perm -002 find / \(-nouser -o -nogroup \) find / -uid +99 -uid -201 find / -gid +0 -gid -100 find / \(-uid 1000 -o -gid 1000 \) -print0 xargs -0 chown 1005:1005</pre>	<p>Dateien des Anwenders "tsbirn" Gehören root + für andere schreibbar Gehören keinem User oder keiner Gruppe Gehören User mit 100 <= UID <= 200 Gehören Gruppe mit 1 <= GID <= 99 UID/GID 1000 --> 1005 konvertieren</p>
<pre>find / -type f \(-perm -100 -o -perm -010 -o -perm -001 \) find / -type f -perm -4000 find / -type f -perm -2000 find / -type d -perm -2000 find / -type f -perm -1000</pre>	<p>Ausführbare Dateien (Programme) Dateien mit SetUID-Recht (mindestens) Dateien mit SetGID-Recht (mindestens) Verz. mit SetGID-Recht (mindestens) Dateien mit Sticky-Recht (mindestens)</p>
<pre>find / -type f -inum 123 find / -type f \! -type l -links +1</pre>	<p>Dateien zu Inode 123 Dateien mit Anz. Hardlinks > 1</p>
<pre>find / -type f -exec grep -il "made" {} \; find / -type f xargs grep -il "made" grep -ilR "made" /</pre>	<p>Dateien die Text "made" enthalten (analog) (analog per grep "rekursiv" aufrufen)</p>
<pre>find / -type f -size 0 -ok rm {} \; find / -type f -empty -ok rm {} \; find / -type d -exec rmdir -p {} \; find / -type d -empty -exec ... find -L / -type l -exec rm {} \; find / -xtype l -delete</pre>	<p>Leere Dateien löschen (mit Abfrage) (analog) Leere Verz. löschen (kein Inhalt) Leere Verz. löschen (besser!) Broken Symlinks löschen (Ziel ex. nicht) (analog)</p>

TIPP: Alle Dateien im eigenen Heimatverz. auflisten, die in der vorletzten Woche (Tag 7-13) verändert wurden und den Text "muster" enthalten (GROSS/kleinschreibung egal):

```
find ~ -type f -mtime +6 -mtime -14 \
-exec grep -li "muster" {} \; 2> /dev/null # f=file, m=modify
# -l=list -i=ignorecase
```

1.7) GNU-"find" kennt weitere Bedingungen CONDITION

Apr 25, 24 3:00

find-HOWTO.txt

Page 4/6

Bedingung	Bedeutung
-P -L / -follow -H	Symbol. Links NIE verfolgen (Standard!) Symbol. Links IMMER verfolgen [links, dereference] Symbol. Links NICHT verfolgen AUSSER Kmdo-Argumente
-d / -depth	Erst. Verz.inhalt., dann Verz. selbst bearbeiten (Standard: Erst Verz. selbst, dann Verz.inhalt)
-mount / -xdev -prune -noleaf	Nur Verz. auf GLEICHEM Dateisystem betrachten NICHT in Verz. absteigen NICHT optimieren dass "." + ".." vorhanden bei -name...
-maxdepth LEV -mindepth LEV	Max. LEV Verz. + Dateien absteigen (0=nur Kmdo-Argumente) Verz. + Dateien bis LEV ignorieren (1=Kmdo-Argument ign.)
-fstype TYPE	Filesystemtyp (mögliche Werte per -printf %F, z.B. "ext3")
-mmin [+]-MIN -amin [+]-MIN -cmin [+]-MIN	Änderung vor +mehr/-weniger/genau MIN Minuten [modify] Lesender Zugriff vor +mehr/... MIN Minuten [access] Status-Änderung (Inode) vor +mehr/... MIN Minuten [change]
-newer FILE -anewer FILE -cnewer FILE -newerXY FILE	Inhalt-Änderung VOR Änderung an FILE [modify] Lesender Zugriff VOR lesendem Zugriff auf FILE [access] Status-Änderung (Inode) VOR Status-Änd. an FILE [change] X: a=access, B=birth, c=inode change, m=modification time Y: t=FILE als Zeitangabe im "date -d" Format interpretiert
-used [+]-DAY	Zugriff +mehr/-weniger/genau DAY Tage seit Änderung
-uid [+]-UID -gid [+]-GID	Analog "-user" aber per Nummer (Bereich von/bis möglich) Analog "-group" aber per Nummer (Bereich von/bis möglich)
-empty	Leere Datei/Verz. (leeres Verz. sonst schwer zu prüfen)
-regex "PATTERN" -iregex "PATTERN"	Regulärer Ausdruck passt [regular expression] Analog, aber Groß-/Kleinschr. egal [ignorecase]
-wholename "PAT" -iwholename "PAT"	Ganzer Pfadnamen gemäß Shell-Muster "PATTERN" passt Analog, aber Groß-/Kleinschr. egal [ignorecase]
-samefile "FILE" -daystart	Gleiche Inode-Nummer wie FILE (also Hardlink auf FILE) Tagesbeginn 00:00 statt "find"-Startzeitpunkt als Basis!

HINWEIS: -regex ... muss VOLLSTÄNDIG auf Pfad passen (^ und \$ sind wegzulassen!)
-E schaltet extended Regex (ERE) ein \(\ \) -> () ? + {...}
Teilmatch daher so zu schreiben: -regex ".*REGEX.*"

1.8) GNU-"find" kennt weitere Aktionen ACTION

Bedingung	Bedeutung
-print0	Namen in gen. Liste durch "\0" (NUL-Bytes) trennen (Standard: durch "\n", für "xargs -0" sinnvoll)
-printf FMT	Benutzerdef. Ausgabe mit %-Formatelementen in FMT
-fls FILE -fprint FILE -fprint0 FILE -fprintf FILE FMT	Analog "-ls", aber auf Datei FILE ausgeben Analog "-print", aber auf Datei FILE ausgeben Analog "-print0", aber auf Datei FILE ausgeben Analog "-printf FMT", aber auf Datei FILE ausgeben

Bei "-printf" und "-fprintf" sind folgende Formatbezeichner in FMT angebar:

Fmt	Bedeutung
%a	Letzter Zugriffszeitpunkt [access]
%b	Größe in 512-Byte Blöcken [block]
%c	Letzte Statusänderung [change]
%d	Tiefe im Verz.baum (0=Kommando-Argument) [depth]
%D	Geräte Nummer [Device]
%f	Reiner Dateiname (letzte Komponenten) ohne Pfadanteil [filename]
%F	Dateisystemtyp [File]
%g	Besitzergruppenname (oder GID) [group]
%G	Besitzergruppen-ID (GID) [GID]
%h	Pfadname ohne Dateiname (letzte Komponente) [dirpath]
%H	Kommando-Argument das Fund der Datei ausgelöst hat

%i	Inode-Nummer	[inode]	
%k	Größe in 1K Blöcken	[kilo]	
%l	Ziel eines Symb. Links (leer falls keiner)	[link]	
%m	Zugriffsrechte in Oktalnotation	[permission]	
%M	Zugriffsrechte in symbolischer Form	[Permission]	
%n	Anzahl Hardlinks auf Datei	[hardlink]	
%p	Vollständiger Dateiname	[path]	
%P	Dateinamen ohne Kommando-Argument das Fund der Datei auslöst	[Path]	
%s	Dateigröße in Byte	[size]	
%S	Leerheit von Dateien (1.0=keine, <=1.0)	[Sparse]	
%t	Letzte Inhalt-Änderung	[change time]	
%u	Besitzernamen (oder UID)	[user]	
%U	Besitzer-ID (UID)	[UID]	
%y	Dateityp "fdlcbpxD" (analog "ls -l")	[type]	
%Y	Dateityp mit Verfolgung symb. Links (L=Loop, N=Nichtexistent)	[type]	

%%	Prozentzeichen "%"		
\n	Zeilenvorschub (analog \r \t \v \f \\)		

1.9) GNU-"find" kennt weitere Verknüpfungen

Verkn	Bedeutung
-and	Analog "-a" (Standard falls keine andere Verknüpfung angegeben!)
-or	Analog "-o"
-not	Analog "!" ("!" ist normalerweise zu schützen: "\!")
,	Erst linke Seite (Ergebnis verwerfen) dann rechte Seite (Ergebnis)

2) Performance-Problem von "find"

"find" startet das nach "-exec/-ok" angegebene Kommando für JEDE gefundene Datei neu. Dies ist bei sehr vielen gefundenen Dateien ineffektiv, da jedesmal ein neuer Prozess erzeugt wird.

```
find $HOME -type f -size +200 -exec gzip {} \;
```

Effektiver sind die folgenden Aufrufe (einfügen des Ergebnisses von "find" auf der Kommandozeile per Kommandosubstitution `...` oder `\${...}` bzw. sammeln der MAXIMAL möglichen Menge an Argumenten vor einem Aufruf von "gzip" per "xargs"):

```
gzip `find $HOME -type f -size +200 -print` # 1) sh
gzip $(find $HOME -type f -size +200 -print) # 2) bash, ksh
find $HOME -type f -size +200 -print | xargs gzip # 3) xargs
```

Allerdings kann bei den Varianten 1)+2) ein "Überlaufproblem" eintreten, wenn die einzufügende Liste zu lang ist (abhängig von der Shell ab 1-2 Mio Zeichen oder mehreren 1000 Namen).

Alle obigen Varianten 1)-3) haben den Nachteil, dass bei bestimmten Sonderzeichen in Dateinamen (Whitespace, ...) die erzeugte Liste falsch interpretiert wird. Bei GNU-"find" und GNU-"xargs" gibt es dafür eine Lösung, die als Trennzeichen das NUL-Byte "\0" verwendet (statt Zeilenvorschub "\n"), das PRINZIPIELL nicht in einem Dateinamen vorkommen kann (neben dem "/"):

```
find $HOME -type f -size +200 -print0 | xargs -0 gzip # oder
find $HOME -type f -size +200 -print0 | xargs --null gzip #
```

HINWEIS: Viele weitere Kommandos, die Listen von Dateinamen verarbeiten, kennen inzwischen diesen Schalter -0/--null (z.B. tar, cpio, grep, sed, ...).

3) "find"-Ersatz "locate"

"locate" ist eine schnellere, aber nicht so leistungsfähige Variante von "find" (kann nur nach DATEINAMEN suchen, nicht aber nach sonstigen Dateiattributen). Sucht nicht im Dateibaum direkt, sondern in Indexdatei "/var/lib/locatedb".

```
locate "PATTERN" # Datei gemäß Muster PATTERN suchen (* ? [...] ...)
```

Diese Indexdatei muss vorher mit "updatedb" erzeugt und von Zeit zu Zeit aktualisiert werden (da Änderungen am Dateisystem darin nicht vermerkt werden). Dies kann einige Zeit dauern und wird meist täglich um Mitternacht über einen "crontab"-Job erledigt (wenn der Rechner zu diesem Zeitpunkt läuft). "updatedb" kann aber auch per Hand aufgerufen werden (mit root-Rechten!):

```
sudo updatedb      # locate-Datenbank updaten
```