RAM Disk voll - No space left on device

Wenn die RAM-Disk voll ist, funktioniert das Webinterface von LoxBerry nicht mehr. Außerdem funktionieren wahrscheinlich auch Plugins nicht mehr korrekt. Dieser Artikel beschreibt die Vorgehensweise, um herauszufinden, was verursacht hat, dass die RAM-Disk voll ist.

Allgemeines

So sieht eine der Fehlermeldungen aus, wenn die RAM-Disk voll ist:

```
Software error:
HTML::Template::new() - Problem writing cache file
/tmp/templatecache/37/ad42418549121b272142da1fb5b797 (file_cache => 1) : No
space left on device at
/opt/loxberry/webfrontend/htmlauth/system/plugininstall.cgi line 284.
Depending of what you have done, report this error to the plugin developer
or the LoxBerry-Core team.
Further information you may find in the error logs.
```

Screenshot


Software error:

HTML::Template::new() - Problem writing cache file /tmp/templatecache/37/ad42418549121b272142da1fb5b797 (file_cache => 1) : No space left on device at /opt/loxberry/webfrontend/htmlauth/system/plugininstall.cgi line 284.

Depending of what you have done, report this error to the plugin developer or the LoxBerry-Core team.
Further information you may find in the error logs.

Die Ordner und Dateien müssen nicht identisch sein - Indikator ist das **`No space left on device`**.

Kein Fehler eines Plugins

Eine derartige Fehlermeldung ist **kein Fehler des verwendeten Plugins**, bzw. **kein Fehler der aufgerufenen Funktion von LoxBerry**. Wenn die RAM-Disk voll ist, funktionieren Dinge einfach nicht mehr. Das ist unabhängig vom Plugin oder der aufgerufenen Funktion.

Ursachen für eine volle RAM-Disk

Um zu verstehen, wie die RAM-Disk voll werden kann, muss man wissen, wie LoxBerry die RAM-Disk verwendet:

- Alle Log-Dateien von Plugins verwenden die RAM-Disk
- Das Template-System von LoxBerry verwendet die RAM-Disk (vom Template-System kommt die Fehlermeldung oben)
- Auch die normalen Linux-Logfiles werden am LoxBerry in der RAM-Disk abgelegt.
- Die Betriebssystem-Ordner `/tmp`, `/dev/shm` und auch `/var/log` werden von LoxBerry in die RAM-Disk umgeleitet.

Die **Log-Dateien der Plugins** werden von LoxBerry anhand eines Algorithmus regelmäßig

automatisch aufgeräumt.

Der Template-Cache des Template-Systems ist so klein, dass er die RAM-Disk nicht belastet.

Hauptursache für volle RAM-Disk sind Logfiles in den System-Temp- und Log-Ordnern (/tmp und /var/log). Diese müssen nicht unmittelbar von LoxBerry oder den Plugins erzeugt sein!

Aus unserer Erfahrung sind die Haupturache für große Dateien in der RAM-Disk:

- Ein Plugin installiert eine Drittapplikation (z.B. FHEM). Die Drittapplikation loggt sehr viel in die RAM-Disk und räumt das Log nie auf.
- Ein Benutzer installiert selbst eine Applikation. Diese Applikation loggt in die RAM-Disk und räumt nie auf.
- Dateien wurden zwar gelöscht, aber von der Applikation noch nicht freigegeben

Schnelle Abhilfe

Schnelle Abhilfe bei diesem Fehler schafft ein Reboot von LoxBerry. Die RAM-Disk wird durch den Reboot geleert, sodass danach wieder alles funktioniert.

Allerdings ist das nur Symptombekämpfung. Der Verursacher wird die RAM-Disk mit der Zeit wieder voll schreiben, bis LoxBerry nicht mehr funktioniert.

Deswegen ist ein Reboot nur sinnvoll, wenn es schnell gehen muss.

Richtige Lösung - Verursacher suchen

Um den Verursacher zu finden, darf LoxBerry **nicht** unmittelbar vorher neu gestartet worden sein, sonst findet man die Dateien nicht.

Bitte melde dich zuerst mit Putty an der Shell an → Eine SSH-Verbindung mit putty aufbauen / Shell-Zugriff.

Mit `su` und der Passworteingabe machst du dich zum root-Benutzer (Administrator).

Prüfen des freien Speichers in der RAM-Disk

```
df -h /var/log
```

```
loxberry@loxberry-dev:/tmp $ df -h /var/log
Filesystem      Size  Used Avail Use% Mounted on
log2ram         200M  200M     0 100% /var/log
```

Das ist die Bestätigung, dass die RAM-Disk (ältere Pi's verwenden das Dateisystem tmpfs, neuere Pi's verwenden log2ram) voll ist.

Verschiedene Verzeichnisse auf deren Größe prüfen

Wir prüfen nun die Größe verschiedener Verzeichnisse, die die RAM-Disk benutzen:

- `du -h /tmp`
- `du -h /opt/loxberry/log/plugins`
- `du -h /opt/loxberry/log/system_tmpfs`
- `du -h /var/log`

Ausgegeben wird dabei jeweils die Größe (belegter Platz) jedes Unterverzeichnisses, und am Ende die Summe aller Verzeichnisse.

```
root@loxberry-dev:/tmp# du -h /var/log
0       /var/log/watchdog
828K    /var/log/unattended-upgrades
0       /var/log/samba/cores/nmbd
0       /var/log/samba/cores/smbd
0       /var/log/samba/cores
400K    /var/log/samba
103M    /var/log/mosquitto
0       /var/log/lighttpd
0       /var/log/fsck
0       /var/log/cups
68K     /var/log/apt
0       /var/log/apache2
0       /var/log/private
194M    /var/log
```

In diesem Beispiel belegt das Unterverzeichnis `mosquitto/` 103 MB. In Summe liegen im `/var/log` 194 MB. Die Größe der RAM-Disk ist jedoch nur 200 MB.

→ In meinem Beispiel ist ein Verursacher der Mosquitto Broker

→ Nochmal etwa 91 MB liegen offenbar direkt im `/var/log` Verzeichnis (Differenz zwischen den 103 MB von Mosquitto und den 194 MB in Summe).

Um den Verzeichnisinhalt sortiert nach Größe auszugeben, funktioniert dieser Befehl:

- `ls -a -S -h -l /var/log`

```
root@loxberry-dev:/tmp# ls -a -S -h -l /var/log
total 91M
-rw-r-----  1 loxberry  loxberry  48M May 11 09:18 daemon.log
-rw-r-----  1 loxberry  loxberry  37M May  3 00:00 daemon.log.1
-rw-r-----  1 loxberry  loxberry 5.2M May 11 09:19 auth.log
-rw-rw-r--  1 root      utmp     286K May  6 07:27 lastlog
-rw-r-----  1 loxberry  loxberry 267K Apr 26 00:00 auth.log.3.gz
-rw-r-----  1 loxberry  loxberry 261K May  3 00:00 auth.log.2.gz
-rw-r-----  1 loxberry  loxberry 249K Apr 19 00:00 auth.log.4.gz
-rw-r-----  1 loxberry  loxberry 249K Apr 12 00:00 auth.log.5.gz
-rw-r-----  1 loxberry  loxberry 161K Apr 12 00:00 daemon.log.4.gz
-rw-r-----  1 loxberry  loxberry 159K Apr 26 00:00 daemon.log.2.gz
-rw-r-----  1 loxberry  loxberry 159K Apr 19 00:00 daemon.log.3.gz
-rw-r-----  1 loxberry  loxberry  24K May 11 09:19 messages
-rw-r-----  1 loxberry  loxberry  24K May 11 09:19 syslog
-rw-r-----  1 loxberry  loxberry  13K May  5 22:33 kern.log
-rw-r--r--  1 root      root     9.7K Apr 28 04:25 dpkg.log.1
-rw-r-----  1 loxberry  loxberry 6.9K Mar 28 10:17 kern.log.3.gz
drwxr-xr-x 11 root      root     4.0K Mar 28 10:17 ..
-rw-rw-r--  1 root      utmp     3.4K May  6 07:27 wtmp
-rw-r-----  1 loxberry  loxberry 1.5K May  5 22:33 syslog.5.gz
-rw-r-----  1 loxberry  loxberry 1.3K Mar 28 10:17 user.log.1
-rw-r-----  1 loxberry  loxberry 1.2K Mar 28 10:17 debug.1
```

Hier sieht man, dass das daemon.log und daemon.log.1 sehr groß sind. Diese Logs würden normalerweise vom Dienst logrotate geleert, das hat aber (wahrscheinlich weil die RAM-Disk bereits zu voll war) nicht mehr funktioniert.

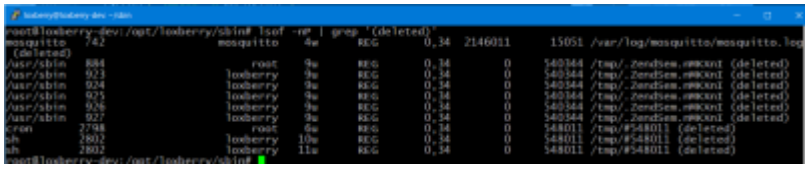
Gelöschte, offene Dateien aufspüren

Linux hat die Eigenheit, dass Dateien gelöscht werden können (und sie sind dann auch nicht mehr sichtbar), wenn aber die gelöschte Datei noch von einem Programm genutzt wird, bleibt die Datei noch vorhanden. Erst, wenn diese Applikation die Datei schließt, wird sie tatsächlich gelöscht.

Verwende diesen Befehl, um offene, gelöschte Dateien zu finden:

```
lsof -nP | grep '(deleted)'
```

Es erscheint eine Liste aller offenen, gelöschten Dateien:



Die vorletzte Zahl ist der belegte Speicher in Bytes (im Bild hat das gelöschte mosquitto.log 2146011 Bytes = ~2 MB). Die erste Spalte ist der Prozess, die letzte Spalte der Pfad der Datei. Damit kannst du meist auf die Applikation, den Dienst oder das Plugin schließen.

Um die Datei endgültig zu entfernen, musst du den entsprechenden Prozess stoppen.

Fehler korrigieren

Mit den Tipps oben bist du jetzt soweit, dass du weißt, wer der Verursacher ist. Nun gilt es, ein erneutes Auftreten zu verhindern:

- Wenn es sich um Logdateien von einer Applikation ist, die durch ein Plugin installiert wurde:
  - Suche nach Einstellungen der Applikation bezüglich Logging, Verbose etc. in dessen Konfigurationsdatei. Schalte den Log-Output auf minimale Ausgabe oder komplett ab.
  - Gib dem Plugin-Autor Bescheid, dass er die Applikation evt. mit einer besseren Logging-Einstellung ausliefert.
- Wenn es sich um Logdateien einer von dir installierten Applikation handelt:
  - Suche nach Einstellungen dieser Applikation, um das Logging zu reduzieren oder zu deaktivieren.
  - Wenn es eine Pfad-Einstellung in der Konfiguration gibt, kannst du evt. den Pfad an einen Ordner verschieben, der auf der SD-Karte liegt (nicht empfohlen).
- Wenn es sich um andere Dateien (nicht Log-Dateien) handelt:
  - Im Falle eines Plugins: Frag beim Plugin-Autor nach, wie du die Größe oder den Pfad ändern kannst.
  - Im Falle einer selbst installierten Anwendung: Prüfe die Webseite der Applikation, ob es dafür Einstellungen gibt.
- Wenn es sich um Dateien direkt von LoxBerry handelt:
  - Bitte mach im Loxforum einem Thread auf und mach einen Screenshot, welche Dateien wo liegen mit deren Größe. Es könnte sich um einen Bug handeln.

Große Files löschen

Wenn es sich um Log-Dateien handelt, und du die Ursache gelöst hast, solltest du zum Löschen folgendermaßen vorgehen:

```
echo "File geleert" > /pfad/zur/datei/datei.log
```

Erst danach:

```
rm /pfad/zur/datei/datei.log
```

Hintergrund ist, dass die Applikationen oft die Logdateien offen halten. Wird das File direkt über `rm` gelöscht, bleibt die ursprüngliche Datei (unsichtbar) vorhanden, bis die Applikation beendet wurde.

Mit dem `echo` hingegen schreiben wir in die bestehende Datei eine einzelne Zeile, was implizit die Datei leert.

Abschließende Kontrolle:

```
df -h /var/log
```

```
root@loxberry-dev:/tmp# df -h /var/log
Filesystem      Size  Used Avail Use% Mounted on
log2ram         200M   98M  103M  49% /var/log
root@loxberry-dev:/tmp#
```

Sollte der Platz dennoch nicht freigegeben werden, starte deinen LoxBerry neu.

Siehe auch

LoxBerry und RAM-Disk

Message The logfile database sends an error and cannot automatically be recovered