UT Web - Research - Zenoss and filesystem mount order

The Problem: Full file systems that are not full

Sometimes after a routine reboot, a UT Web system will see quasi-random "disk space threshold" events from Zenoss:

```
disk space threshold: 1820.9% used (-1.6TB free)
```

The "full" file system may be `/boot` , or a database file system. File system capacity remains the same but the usage numbers change to wildly oversized values.

Sometimes another reboot will reset the alert, and sometimes the device has to be remodeled in Zenoss. Some ITS services routinely include a remodel step after rebooting a system so it is sure to go "green".

This issue is observed most frequently on systems with additional file systems mounted - e.g. database servers, file servers, or systems that mount remote file systems.

Possible RCA: Indeterminate file system mount order due to parallel mounts

Debugging data from the SNMP agent shows that when a file system shows incorrect usage numbers in Zenoss after a reboot, its internal index number is not the same as when Zenoss shows the correct data.

This difference in index number is because the SNMP agent indexes file systems as they are found in `/proc/mounts`. Since `systemd` mounts file systems in parallel by default, it is a matter of sheer timing whether a file system is always in the same place in the mount list after a reboot.

When Zenoss retrieves a fresh round of data from the SNMP agent, it does not detect that the file systems have different index numbers, prompting the behavior we observe.

For example, if `/boot` and`/var/lib/mysql` exchange index numbers, /boot will look full because it doesn't have as much capacity as the database filesystem. `/var/lib/mysql` will show a corresponding drop in usage at the same time.

Ideal solution: Zenoss detects index number changes

It seems that Zenoss could verify that file system mount point paths still match the index numbers that were observed during the most recent device modeling run. Then these index number changes would not matter.

In terms of where the responsibility lies for detecting these changes, RFC 2790 states that the file system index numbers only need to be stable between one re-initialization of the agent to the next. The SNMP agent's behavior appears to be standards compliant in that respect.

```
   hrFSIndex OBJECT-TYPE
       SYNTAX     Integer32 (1..2147483647)
       MAX-ACCESS read-only
       STATUS     current
       DESCRIPTION
           "A unique value for each file system local to this
           host.  The value for each file system must remain
           constant at least from one re-initialization of the
           agent to the next re-initialization."
       ::= { hrFSEntry 1 }
```

Workaround: mount in explicit order using x-systemd.requires

Absent an update to Zenoss as suggested above, the `systemd`-specific mount option `x-systemd.requires` lets the system administrator (tediously) order file system mounts.

As documented in `systemd.mount(5)`, local and remote file systems have different dependency targets. Local file systems generally land in the mount table before remote file systems, so a particular system may only exhibit index number swaps for the remote file systems.

For example, on a typical UT Web FUSE client we may only need to assure that `/home/utweb` and `/utweb/common` always mount in that order. So we configure `/home/utweb` to require`/utweb/common` to be mounted by using `x-systemd.requires=/utweb/common` . This approach can be extended to the entire set of file systems if necessary.

```
# See the FSTAB section of systemd.mount(5) for details on the x-systemd.requires option
utweb-fs-z1-p03-gfs:/utweb_home        /home/utweb        glusterfs        defaults,_netdev,backup-volfile-
servers=utweb-fs-z1-p04-gfs,lru-limit=65536,invalidate-limit=131072,x-systemd.requires=/utweb/common
0        0
```

Only about reboots

This workaround is only about annoying incorrect Zenoss disk threshold events after a normal reboot.

If a new file system is mounted or removed, that will affect the index numbers generated from the mount table. In that case the device should be remodeled in Zenoss to account for the changed configuration.

Appendix: File system index debug process

- Modify `/etc/sysconfig/snmpd` to direct debug messages with tokens starting with the string "`fsys`" to `/var/log/snmpd` :

  ```
  OPTIONS="-LS0d -Dfsys -Lf /var/log/snmpd -p /var/run/snmpd.pid"
  ```

- Observe the index numbers assigned to file systems

  ```
  fsys:path: Get filesystem entry (/utweb/common)
  fsys:new: Create filesystem entry (index = 31)
  --
  fsys:path: Get filesystem entry (/home/utweb)
  fsys:new: Create filesystem entry (index = 32)
  ```

- The index numbers (e.g. 32 for `/home/utweb`) conveniently match the output from `grep -n`:

  ```
  $ grep -n /home/utweb /proc/mounts
  32:utweb-fs-z1-p03-gfs:/utweb_home /home/utweb fuse.glusterfs rw,relatime,user_id=0,group_id=0,
  default_permissions,allow_other,max_read=131072 0 0
  ```

- Reboot the system until Zenoss is unhappy with it
- Review `/var/log/snmpd` and `/proc/mounts` to observe the file systems now have different index numbers, e.g.

  ```
  fsys:path: Get filesystem entry (/home/utweb)
  fsys:new: Create filesystem entry (index = 31)
  --
  fsys:path: Get filesystem entry (/utweb/common)
  fsys:new: Create filesystem entry (index = 32)
  ```

- Apply the `x-systemd.requires` mount option to entries in `/etc/fstab` as desired, e.g.

  ```
  /dev/vg_rsnapshot/lv_rsnapshot        /utweb/rsnapshot        xfs        defaults,noatime,x-systemd.
  requires=/home/utweb        1        2
  ```

- Reboot the system to confirm the fstab entries
- Remodel the device in Zenoss
- Reboot the system multiple times
- Confirm that the file system index numbers have become stable
- Confirm that Zenoss shows correct file system usage
- Revert `/etc/systemd/snmpd` to the usual values

  ```
  OPTIONS="-LS0-4d -Lf /dev/null -p /var/run/snmpd.pid"
  ```

Notes

- Example`snmpwalk` command to get the file system list from the local SNMP agent:

```
$ snmpwalk -v3 -l authPriv -u zenoss -a SHA -A 'APASSWD' -x AES -X 'XPASSWD' localhost
hrStorageTable
HOST-RESOURCES-MIB::hrStorageIndex.1 = INTEGER: 1
HOST-RESOURCES-MIB::hrStorageIndex.3 = INTEGER: 3
HOST-RESOURCES-MIB::hrStorageIndex.6 = INTEGER: 6
HOST-RESOURCES-MIB::hrStorageIndex.7 = INTEGER: 7
HOST-RESOURCES-MIB::hrStorageIndex.8 = INTEGER: 8
...
```

References

- `systemd.mount(5)`
- https://github.com/systemd/systemd/commit/3519d230c8bafe834b2dac26ace49fcfba139823
- RFC 2790