logstory

Jacques Gélinas
jacquesgelinas2407 at gmail.com

January 15, 2018

Abstract

Tracking usage of hardware components such as disk drives and servers is tedious and seldom done properly. When we need to figure out the **story** of a server (what went in, what failed), our memory is not enough. The **logstory** tool is an automated solution to maintain an inventory of your hardware.

Operation

To get going, you do:

- You install the **logstory** package on a management server which has **ssh** access to all servers , as root, without passwords.
- Then you create the script /root/logstory-listservers. This script just prints the list of servers to probe.
- You run once a day in a cron the command **/usr/sbin/logstory-scanservers –all**.
- You end up with various files in **/var/log/logstory**.
  - One **story** file per server.
  - One **deadstuff** file.
  - One **unused** file.

The last two will be empty (for a while).

The logstory command

This is the main command managing story files. It receives keys identifying hardware components and record them in the story files. There is one story file per server.

This is called a story file because it maintains a list of all components found in a server, including stuff that was removed from it.

The logstory-scan command

This command is copied (on the fly, nothing to install) on each server and executed. It just scans the various hardware components and prints a list of keys. It mostly uses the scsi_id command (part of the **udev** project) and the dmidecode (spits everything it finds in the bios memory).

The logstory-scanservers command

For each server named by the **/root/logstory-listservers**, it executes the following sequence:

```
cd /var/log/logstory
scp /usr/sbin/logstory-scan server:/root
ssh server /root/logstory-scan \
| logstory server.story - >server.story.new
mv server.story.new server.story
```

The story files

For each server, a **story** file is maintained in **/var/log/logstory**. The format of the file is trivial. All lines are made of 3 parts:

- The key identifying a component or part.
- The date when this key was first seen in this server (The start of a period).
- The date when this key was first not seen in this server. If the **logstory-scanservers** is executed once a day, we can say this is the date it was removed from the server. At first, a simple dash (-) is written, showing that the server still contains this resource.

So it looks like this (slightly modified to fit this screen):

```
board/MANUF/Dual_AMD_Opteron,S2882/O.E.M./O.E.M. 2013/10/01 -
cpu/CPU_1/AMD//Opteron_244/1800_MHz/1800_MHz// 2013/10/01 -
disk/sda/ATA/ST320/3.01/ST320_3LJ0XVFY// 2013/10/01 2013/10/06
disk/sda/ATA/ST320/3.01/ST320_3LJ3TZAM// 2013/10/06 -
kernel/2.6.22.19-vs2.2.0.7 2013/10/01 -
mem/8G 2013/10/01 -
ram/B1/D1A/manuf/prod/DDR3//1333_MHz/4096/serial 2013/10/10 -
net/eth0/xx:xx:xx:xx:xx:xx
net/eth1/yy:yy:yy:yy:yy:yy
net/ipmi/zz:zz:zz:zz:zz:zz
```

In this example, we see that the disk sda was changed.

keys

Keys are assembled from various informations produced by tools such as **scsi_id** and **dmidecode**. We are packing useful informations to make the key useful and yet, not quite unique: We combine stuff like serial numbers, unique identifiers (UUID) and some logical informations. This makes the key not perfectly unique but helps us track the evolution of the device. For example, we add:

- Firmware revision number.
- Disk logical name (sda, sdb).

This applies mostly to disk. So during the complete life of a disk unit, there may be many keys representing it. From the story file stand point, they will represent different disks. This was done to more easily track the history of a disk drive and correlate that with other informations. For example, in the kernel logs we may find a hardware failure on disk sdh on a given day. Using the story file, one can associate easily the disk sdh on that day with the proper disk (which btw is not even in the same server anymore because it was tested and finally reused elsewhere).

The firmware revision was added as well because it is an important data point in the life of a disk drive.

Other keys are added to the list. They do not represent hardware components per se, but are helpful to track the evolution of a server:

- kernel/kernel-version
- mem/memory-size

Managing the inventory

When disk drives are removed from servers, they are either **defective** or simply not fit. In the second case, they end up in a drawer in the datacenter, waiting to be reused (if the policy allows it). Keeping track of available disks is seldom done properly. The **logstory** offers a way to automate that. There is the –**unused** command line option. Check the following command line:

```
cd /var/log/story
logstory --unused --deadstuff deadstull.file *.story \
| logstory unused.file - >unused.file.new
mv -f unused.file.new unused.file
```

The –unused command line option

This tells logstory to extract the list of disk units which are not in use in any server. When parsing the story files, only keys associated with disk drives are kept. Also, the logical disk name (sda, sdb) and the firmware revision are removed from the key. It prints any key not associated with an open end date. Those keys are not part of any server.

Instead of just placing this list of keys in a file, it runs those keys through the **logstory** command. We end up with the **unused.file** story file. A different extension was used so *.story may be used on normal server files. So the **unused.file** story file tracks the first appearance and disappearance of keys, like in a server. During the life of a disk drive, it may appear (and disappear) several time from the unused file.

The unused file

The **unused.file** file has the same format as the story files. As such it associates components (disk drives) to period (start and end date). If the disk is not used in any server, the end date is missing (a dash).

The deadstuff file

The **deadstuff.file** file has the same format as the story files. It contains list of retired (defective ?) disk drives. This file has to be maintained by hand. Normally, the sequence to maintain this file looks like this:

- Fix a server: remove a drive and put a new one.
- Run the **logstory-scanservers** command.

- The removed drive is not used any more in any server, so it will appear in the **unused.file** file. It will have **today** as the start date and a dash as the end date. It will probably be the only entry in this file appearing today.
- If the disk was working fine, just store it appropriatly. And you are done.
- If the disk was defective, then continue.
- Remove the line from the **unused.file** file and append it to the **deadstuff.file** file.

The disk won't appear again in the **unused.file** file since it is part of the **deadstuff.file**.

Key formatting

Keys are formatted using various information concatenated using slashes. This format allows easy search in the story files.

Cpu

- Keyword **cpu**
- Socket location
- Manufacturer
- Cpu ID
- Version
- Max Speed
- Current speed
- Core count
- Core enabled

Disk drive

- Keyword **disk**
- Vendor
- Model

- Revision (Firmware)
- Serial
- Wwn
- Scsi serial

Memory

- Keyword **ram**
- Bank number
- Locator
- Manufacturer
- Part number
- Type
- Type detail
- Speed
- size
- serial
- Asset tag
- Rank

System

It represents the enclosure.

- Keyword **system**
- Manufacturer
- Product
- Version
- Serial
- Uuid

System board

- Keyword **board**
- Manufacturer
- Product
- Version
- Serial

Network devices

- Keyword **net**
- network device name
- MAC address