

How to keep on top of your hard drives, so you won't be in for a nasty surprise when one of them fails.

THE SMART WAY TO MONITOR YOUR HARD DRIVE

Don't get caught out by a failing hard drive.

JOHN LANE

The most valuable part of your computer is the data stored within. That's why you take regular backups and hedge against failures by running RAID arrays. Or perhaps you don't think about the worst until after it has happened? Perhaps it would help if you could predict when that might be – that would be smart. It would also be Smart.

Smart stands for Self-monitoring, Analysis and Reporting Technology, and is a monitoring system for hard and solid-state drives that detects and reports on various indicators of reliability with the aim of anticipating failures. Smart is built into the drive's firmware, and maintains various attributes that measure its performance and reliability.

The tools that you use to interact with Smart are collectively known as **Smartmontools**, and should be accessible in your distribution's package manager; for example, in Debian-based systems you'd install it with:

```
apt-get install smartmontools
```

It provides two things: a tool called **smartctl**, which enables you to interact with your Smart drives, and **smartd**, which is a daemon that can be used to perform monitoring in the background.

To begin, you can scan for Smart drives and establish whether they have Smart enabled:

Use **smartctl --info** to see details of your drive, including whether Smart is enabled.

```
sudo smartctl --scan
```

```
sudo smartctl --info /dev/sda
```

If Smart isn't enabled, the command to enable it is

```
sudo smartctl -s on /dev/sda
```

To view a drive's health self-assessment test result:

```
sudo smartctl --health /dev/sda
```

which will tell you briefly if the drive is about to fail. If your drive reports that it is failing then it would be a good idea to think about backing up and replacing it as soon as possible.

You can delve further into the Smart data:

--capabilities shows the Smart features that the drive has, and **--attributes** lists the Smart attributes that the drive monitors (this is probably the useful information that you would like to see).

What's going on with your drive?

The **attributes** display shows a raw value that's vendor-specific but usually sensible – a temperature's raw value is usually represented by a Celsius value, for example – but, to remove vendor-specifics, the firmware normalises them as a value between 1 and 254. The report displays the normalised value in a column called **VALUE** and the raw value in one called **RAW VALUE**. The third important value is in the **THRESH** column, which is a threshold that indicates failure when the normalised value is less than this value. The thresholds are chosen so they indicate imminent failure within 24 hours.

Other columns include **WORST**, which shows the closest to failure value that the attribute's normalised value has held since Smart was enabled, and **TYPE**, which shows a value of either Pre-failure or Old age and describes whether an attribute indicates imminent failure or end-of-life due to expected wear and aging. Don't be alarmed when you see many attributes with Pre-fail against them – that alone does not mean your drive is about to die!

The **WHEN_FAILED** column is the one that warns you when things are not good. It will show "failing

```

[john@testbox]$ sudo smartctl --info /dev/sdd
smartctl 6.2 2013-07-26 r3841 [x86_64-linux-3.14.6-1-ARCH] (local build)
Copyright (C) 2002-13, Bruce Allen, Christian Franke, www.smartmontools.org

=== START OF INFORMATION SECTION ===
Model Family:      Seagate Barracuda 7200.11
Device Model:     ST31500341AS
Serial Number:    9WS1E026
LU MNW Device Id: 5 000c50 0111f615c
Firmware Version: CC1H
User Capacity:    1,500,301,910,016 bytes [1.50 TB]
Sector Size:     512 bytes logical/physical
Rotation Rate:   7200 rpm
Device is:       In smartctl database [for details use: -P show]
ATA Version is:  ATA8-ACS T13/1699-D revision 4
SATA Version is:  SATA 2.6, 3.0 Gb/s
Local Time is:   Tue Jun 17 14:36:34 2014 BST
SMART support is: Available - device has SMART capability.
SMART support is: Enabled

[john@testbox]$

```

now” when the normalised value is less then its threshold. If it isn't failing now but has in the past it'll show “in the past” and the **WORST** value will be less then the threshold.

Beware of vendors

Something to be aware of is that none of the attribute's names and meanings are defined by the standard and, while they are generally used sensibly, vendors can, and sometimes do, use them for differing purposes. If you have a drive that does this then you can give the **--vendorattribute** argument to alter how **smartctl** displays such attributes.

Smart attributes are updated by tests run within the drive's firmware. The tests don't affect a disk's data and can operate when it is mounted and in use. There are three kinds of tests.

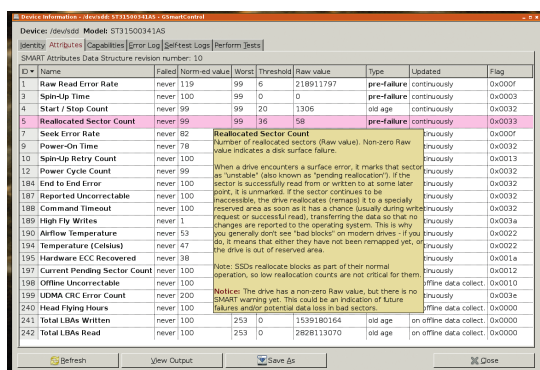
- **Online testing** happens transparently when Smart is enabled and it doesn't impede the drive's performance.
- **Offline testing** can impede performance. It can either be performed on-demand or automatically at regular intervals. They are suspended during disk access and, therefore, have little real impact.
- **Self-testing** only happens on demand. These are more resource intensive and can impede drive performance while running.

The online and offline tests update the Smart attributes (the **UPDATED** column in the attributes listing shows whether they are updated by online or offline testing).

Most drives perform offline testing automatically every four hours but you can enable or disable this with **smartctl --offlineauto**. If they are disabled, they can still be run on demand:

```
sudo smartctl --test=offline /dev/sda
```

Self-tests are different. These are only run on demand and, although they don't affect the disk's operation, they can be resource intensive. There are short tests that take a few minutes to complete and longer ones that can take several hours. A third kind of test is the conveyance test that's designed to detect problems caused when the drive is in transit. The results of these tests can be viewed in the Smart logs (see the boxout above).



ID	Name	Failed	Norm-ed value	Worst	Threshold	Raw value	Type	Updated	Flag
1	Raw Read Error Rate	never	119	99	6	218911797	pre-failure	continuously	0x0007
9	Spin-Up Time	never	100	99	0	0	pre-failure	continuously	0x0003
8	Start/Stop Count	never	99	99	1306	1306	pre-age	continuously	0x0032
5	Reallocated Sector Count	never	99	99	36	58	pre-failure	continuously	0x0033
7	Seek Error Rate	never	82	Reallocated Sector Count			inuously	0x0007	
9	Power-On Time	never	78	Number of reallocated sectors (Raw value). Non-zero Raw value indicates a disk surface failure.			inuously	0x0032	
10	Spin-Up Retry Count	never	100				inuously	0x0013	
12	Power Cycle Count	never	99				inuously	0x0032	
184	End to End Error	never	100				inuously	0x0032	
187	Reported Uncorrectable	never	100				inuously	0x0032	
188	Command Timeout	never	100				inuously	0x0032	
189	High Fly Writes	never	1				inuously	0x003a	
190	Airflow Temperature	never	53				inuously	0x0022	
194	Temperature (Celsius)	never	47				inuously	0x0022	
195	Hardware ECC Recovered	never	38				inuously	0x001a	
197	Current Pending Sector Count	never	100				inuously	0x0012	
198	Offline Uncorrectable	never	100				inuously	0x0010	
199	UDMA CRC Error Count	never	200				inuously	0x003e	
240	Head Flying Hours	never	100				inuously	0x0000	
241	Total LBAs Written	never	100				inuously	0x0000	
242	Total LBAs Read	never	100				inuously	0x0000	

The Smart attributes are a measure of the drive's health.

Smart logs

Smart maintains various logs – here are the one's you're most likely to find useful. Refer to the **smartctl** man page for a full list of logs and their uses.

- **error** Summary SMART error log
- **xerror** Extended Comprehensive SMART error log
- **selftest** SMART self-test log
- **xselftest** Extended SMART self-test log

To view a log, use the command

```
sudo smartctl --log=TYPE /dev/sda
```

where **TYPE** is one of those listed above.

Tests are only useful if they are run and if their output is monitored. To help with this, the other part of **smartmontools** is the **smartd** daemon. It monitors Smart attributes for signs of problems, runs self-tests and reports to the system log. It can also use email to alert administrators of potential problems.

smartd

To use **smartd**, first check its config file, **/etc/smartd.conf** and then enable it in the appropriate manner for your system. For example, if you're using systemd:

```
sudo systemctl enable smartd
```

```
sudo systemctl start smartd
```

The default configuration contains a single entry, **DEVICSCAN**, which instructs the daemon to scan for Smart devices to monitor. You can perform this scan yourself to see what would be monitored in this default scenario

```
sudo smartctl --scan
```

Should you wish to effect more control, you can replace this default configuration with explicit entries for each device. Any entry can be supplemented with directives that enable you to control which tests are performed and where issues get emailed to. For an explanation of these directives, see the man page for **smartd.conf** or do:

smartd -D

The directive that launches self-tests is one that you will probably want to use but it is a little complex. It's specified as a regular expression of the form **T/MM/DD/d/HH** to represent a test and the date and time when it should be run. The **T** character defines the test type – either **S** for a short test or **L** for a long one.

Coding regular expressions is beyond what we can cover here but, as an example, running a daily short test at 2AM and a weekly long test on Saturdays at 2AM could be achieved with an expression like this **(S/..../02L/..../6/02)**

Here's an example **smartd.conf** configuration:

```
DEVICSCAN -a -s (S/..../02L/..../6/02) -m me@example.com
```

Hopefully you'll never experience a disk failure but, by spending some time getting to know **smartmontools** and configuring a **smartd** daemon to monitor your drives, at least you can be forewarned if the worst is about to happen.

A SMART GUI SOLUTION

GSmartControl makes checking your hard drives easy...

JOHN LANE

Sometimes a GUI application comes along that nicely supplements a command line tool. GSmartControl is a GUI wrapper around the command-line **smartctl** tool that we covered in the previous pages. It presents its output in a more accessible form and makes it even easier to interact with your Smart devices.

Because Smart is part of the drive's firmware, tests are run inside the drive – you need to issue one command to start a test and another to check its progress or end result. GSmartControl wraps this process up in a graphical shell around **smartctl** and makes it much easier to use Smart interactively. You can use GSmartControl without knowing how to use **smartctl**, but a little knowledge will help you understand some of the features and options.

Your distro's repository should contain the GSmartControl package, so installing should be straightforward:

apt-get install gsmartcontrol

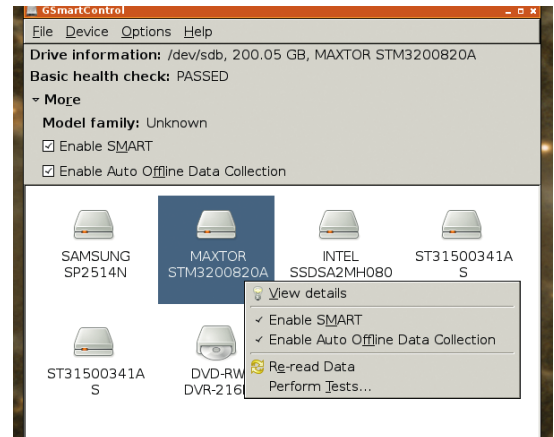
It's a GTK application so its dependencies are minimal. Launching it can be as simple as typing **gsmartcontrol** but you will need root privileges to access some of the Smart functionality. As a convenience, you can launch it like this instead

gsmartcontrol-root

which will automatically perform **su** to gain root privileges (this requires that your environment has a graphical **su** interface, such as gksu). Enter the root password if prompted.

Once it's been appropriately launched, a scan is performed and the available drives are displayed as icons. You may see icons for non-Smart devices such as floppy and optical drives, but you can choose to hide such devices by setting an option in Options > Preferences. If the scan misses a device that you want, use Device > Add Device to add it manually.

The icon's title shows the drive's make and model; you can, in Options > Preferences, add its device name



Click on a device to see a health summary or double-click for detailed device information and to run tests.

and serial number to this view. Click on an icon to select it and a summary is displayed at the top of the window. Expanding the More option shows the drive's model family and offers options that enable/disable Smart and, if the drive supports them, its automatic offline tests.

Health check for drives

You work in GSmartControl one drive at a time. Double-click on a drive to open its device information window. This displays Smart data in tabs for identity, attributes, capabilities and logs. Each tab displays output from **smartctl** that you could get like this:

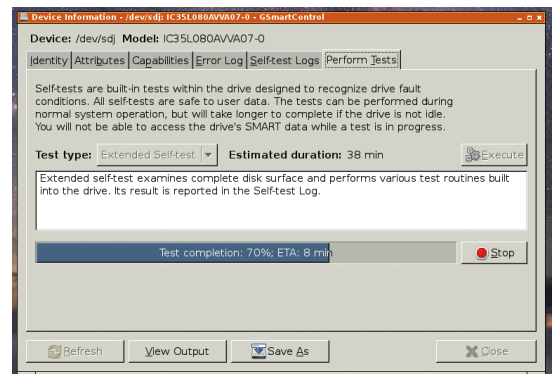
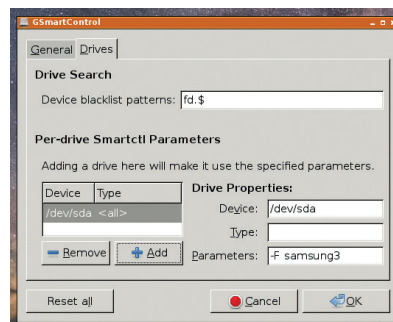
sudo smartctl --health --info --capabilities --attributes /dev/sda

The tabbed display makes the information more readable and provides plentiful help text that goes a long way to explain what the information means. Hover your mouse pointer over most things and a pop-up will explain what it is. This is especially useful when viewing attributes and capabilities whose meanings may be less-than obvious. This online

Preferences

You can customise the behaviour of GSmartControl through Options > Preferences on the main window's menu.

It has two tabs; the first allows application-wide general preferences. The second tab (shown right) enables you to configure how specific drives are handled. Some drives with firmware bugs need additional options passed to **smartctl** and this is where you can specify them. You can also blacklist irrelevant devices, like floppy disk drives, from the device scan so they aren't displayed.



You can see the progress of a running test and when it should finish.

```

[john@testbox]$ sudo smartctl --attributes /dev/sdd
smartctl 6.2 2013-07-26 r3841 [x86_64-linux-3.14.6-1-ARCH] (local build)
Copyright (C) 2002-13, Bruce Allen, Christian Franke, www.smartmontools.org

=== START OF READ SMART DATA SECTION ===
SMART Attributes Data Structure revision number: 10
Vendor Specific SMART Attributes with Thresholds:
ID# ATTRIBUTE_NAME          FLAG     VALUE WORST THRESH TYPE      UPDATED  WHEN_FAILED RAW_VALUE
  1 Raw_Read_Error_Rate      0x000f   118   099   006   Pre-fail Always    -         196087878
  3 Spin_Up_Time              0x0003   100   099   000   Pre-fail Always    -           0
  4 Start_Stop_Count         0x0032   099   099   020   Old_age  Always    -        1304
  5 Reallocated_Sector_Ct    0x0033   099   099   036   Pre-fail Always    -           58
  7 Seek_Error_Rate          0x000f   082   060   030   Pre-fail Always    -       194824787
  9 Power_On_Hours           0x0032   078   078   000   Old_age  Always    -       19413
 10 Spin_Retry_Count         0x0013   100   100   097   Pre-fail Always    -           0
 12 Power_Cycle_Count        0x0032   099   099   020   Old_age  Always    -        1303
184 End-to-End_Error         0x0032   100   100   099   Old_age  Always    -           0
187 Reported_Uncorrect      0x0032   100   100   000   Old_age  Always    -           0
188 Command_Timeout         0x0032   100   098   000   Old_age  Always    -       393241
189 High_Fly_Writes          0x003a   001   001   000   Old_age  Always    -        133
190 Airflow_Temperature_Cel 0x0022   053   049   045   Old_age  Always    -         47 (Min/Max 23/47)
194 Temperature_Celsius     0x0022   047   051   000   Old_age  Always    -         47 (0 14 0 0 0)
195 Hardware_ECC_Recovered  0x001a   040   024   000   Old_age  Always    -       196087878
197 Current_Pending_Sector  0x0012   100   100   000   Old_age  Always    -           0
198 Offline_Uncorrectable    0x0010   100   100   000   Old_age  Offline   -           0
199 UDMA_CRC_Error_Count    0x003e   200   200   000   Old_age  Always    -           0
240 Head_Flying_Hours       0x0000   100   253   000   Old_age  Offline   -       64046552337358
241 Total_LBAs_Written      0x0000   100   253   000   Old_age  Offline   -       1535687354
242 Total_LBAs_Read         0x0000   100   253   000   Old_age  Offline   -       2825309590

[john@testbox]$

```

The tabbed device information presents the Smart data clearly and with additional helpful information. You can View Output to see the raw **smartctl** output if you really want to, or you can save it to a file.

documentation is value added by GSmartControl that is not present in **smartctl**.

Another tab enables you to perform the self-tests that the drive supports: the short, long and conveyance tests. It gives an estimate for how long the test will take. Press the Execute button to start the test. A progress bar reassures you that the test is running and estimates the remaining time, which is a nice feature that the command line **smartctl** lacks. Behind the scenes, GSmartControl is launching the test

```
smartctl -t short /dev/sda
```

and then polling for status until it completes

```
smartctl -c /dev/sda
```

Once the test completes, the result is displayed. "Completed without error" is what you want to see!

Oh no, I have an error!

To help draw your attention to problems, the device information screen highlights potential problems. It highlights both the error and the tab containing it, so

Virtual devices

You can load **smartctl** data from a file as a virtual device. You could, for example, extract data on a remote server using the command line with something like this:

```
ssh root@remoteserver smartctl -a /dev/sda > remoteserver_sda
```

You can then use Device > Load SmartCtl Output As Virtual Device to load that data into GSmartControl. It's presented as another device on the main screen, and you can interact with it in exactly the same way as real devices. The only exception is that you can't run tests.

There is also a Save As function to save a device's **smartctl** data. This may be useful to keep historic records so that you can compare how a device changes over time. You would have to do such comparisons yourself, however, because GSmartControl doesn't offer that feature.


you can go straight there. It also augments the pop-up help with further information specific to the problem.

There are three levels: notices are displayed in a light pink colour; warnings in a darker pink; and alerts in red. Notices convey information that you should be aware of – they are not Smart errors reported by **smartctl**, but GSmartControl interprets the Smart data to provide another level of reporting. Warnings are raised by old-age attributes and alerts by pre-fail attributes.

A pinch of salt

Having Smart is better than not having it, but you shouldn't become complacent. It is prudent to have a backup (and recovery) strategy for your data. Smart does have its issues – it can't possibly detect every kind of failure, and may show errors that don't necessarily impact a drive's stability.

Smart is implemented by the drive

manufacturers, and they don't always adhere to the correct standard. Some drives use Smart data fields for different values, and other firmware idiosyncrasies may introduce bugs or other undesirable features. They sometimes provide proprietary utilities and promote these instead of Smart. However, Smart is a useful addition to your toolbox and may provide timely warnings of potential data loss, giving you time to act before it happens. 

"Having Smart is better than not having it, but you shouldn't become complacent."

PRO TIP

Launch **gsmartcontrol** **--verbose** to see the underlying **smartctl** commands.

John Lane is a technology consultant with a penchant for Linux. He helps new businesses and start-ups make the most of open source software.