Gillware™ Data Recovery

WHITE PAPER

Firmware Corruption and Data Recovery

In this white paper, Gillware's CEO Brian Gill discusses what firmware is, how it fails and how Gillware recovers data in cases of firmware corruption.

What is firmware?

Firmware is the storage device's operating system. Just like you may run a Windows operating system on your computer and run an iOS or Android operating system for your phone, a complicated device needing to store and organize billions of bits onto platters or NAND needs an operating system. I personally prefer the term hard drive operating system or HDD O/S instead of firmware. Either is accurate, but the term firmware tends to have a connotation that it's nothing special or unique. One would expect the firmware for multiple electronic devices coming off a manufacturing line to be identical, but this is not the case in the world of storage.

The firmware on a spinning disk will have all the compiled application code for doing everything the drive needs to do. This baseline firmware will vary slightly from O/S revision to O/S revision. Hard drive manufacturers are always making tweaks to this code for increased performance, security and reliability. Some manufacturers will produce hundreds of versions of their base firmware in a calendar year. For any particular drive-line, like the Western Digital Blue desktop series, they may have ten or twenty revisions a year.

HDD manufacturers will sometimes create custom firmware for different computer companies; Apple likes to have their own firmware as one example. They will code different drive behavior for drives intended for enterprise data centers, consumer desktops, consumer DVR units, etc. A consumer drive like a WD Green will spin down its platters and park the heads during inactivity, as opposed to a WD Enterprise drive that will keeping spinning until a RAID controller tells it to spin down. These behaviors are defined in the firmware.

The firmware zone is also where a lot of the drive's unique calibrations, defect lists, zone tables, unique translation (addressing) information, performance logs and SMART attributes are stored.

How is the firmware produced?

Programmers! Underappreciated, undercompensated and rarely seeing the sun programmers. Firmware is typically written in the programming language C or possibly even lower-level assembly. Today, it's typically compiled for execution on a standard ARM processor. The SOC (system-on-chip) is usually the biggest chip on the printed circuitry; these days it is often manufactured by Marvell. The firmware runs within microprocessors in the SOC which controls all aspects of drive operation.

How does the drive load the firmware?

At power on, the drive executes a small program that is permanently stored in a ROM within the SOC. These instructions will then first load the external ROM which usually contains a head mapping and on-platter firmware addressing. It will then spin up the drive and load the bulk of the firmware from the service area of the platter. The small amount of boot-loader code that lives within the SOC is not changed by the hard drive manufacturer. They don't want to be developing the hardware and the software at the same time. A company like Marvell will develop the SOC first and a company like Western Digital will develop their HDD O/S second.

Why is the firmware important from a data recovery perspective?

It can become corrupted and require repair.

The vast majority of the firmware lives on the platters in a special service area. This area is typically significantly lower density (sectors per track) than the user data area so the drive can load it easily. Even though a manufacturer will typically keep at least one backup copy of this special set of data, unrecoverable corruptions can occur.

Ironically, I believe the majority of these corruptions are directly attributed to the very mechanisms that exist to prolong a drive's lifespan and warn you of imminent failure.

All modern drives implement SMART (Self-Monitoring, Analysis and Reporting Technology). These drives are paying attention to their own behavior and performance, and when it starts deviating outside the norms, logs that information in log files and SMART tables.

During a drive's lifetime sectors go bad. The first time a sector is attempted to be read after it has failed it needs to get put on a list of sectors that we'd like to relocate. The drive can't relocate it right away as the sector is corrupted as the drive does not know what data used to live there. If that sector happened to be in the middle of a payroll database, and the drive just handed back a bunch of random zeros instead of giving an UNC error, you might pay an employee $1000000 instead of $1000. But, at the next write opportunity, the sector will get remapped to a healthy sector that in a reserve area. It's not a great situation when you try to load that database and the operating system says it cannot be loaded because of sector errors, but it is better than pretending everything is fine.

This information about which sectors are pending reallocation and have been successfully reallocated (and where) live on the platters in the firmware zone. Also in the firmware zone are the performance logs, events, and subsequently SMART attributes.

How do these corruptions occur?

So let's imagine a scenario where a headstack is in the early stage of failure. It's taking multiple read attempts to successfully read data, those read events are having unacceptable latency, and lots of sectors need to be added to the growth defect list. The drive needs to use those same heads to save this performance and sector information to the platters! So, one can easily understand how they might write a bunch of gibberish to the firmware zone.

Let's imagine another scenario where a drive is in the middle of doing a bunch of this sector reallocation and subsequently a bunch of performance bookkeeping in the SMART tables. The end user is experiencing I/O lag on the drive and is getting frustrated. The frustrated user decides to do a therapeutic shutdown and cold reboot. The operating system notifies the drive that it wants to perform a shutdown. The drive replies “gimme a minute I'm in the middle of some bookkeeping” so the O/S blocks the event temporarily and is going to wait until the drive tells it “cool I'm done, go ahead and shut down". The human is now having their blood pressure raised as even the shutdown is taking 30 seconds! And they perform a hard shutdown or just yank the power cord rather than wait, while the drive was right in the middle of altering its operating system. Once again, it isn't difficult to comprehend how the HDD O/S can be adversely affected.

What is the effect of corrupted firmware?

When these firmware areas are corrupted it will need to get repaired or the drive cannot boot itself, just like if Windows has corrupted O/S files and cannot boot itself you'll need to grab an O/S disk and troubleshoot. There are many approaches for performing this analysis and repair. Here at Gillware we build our firmware library every single day and attempt to back up the firmware on every drive that enters our doors as part of our standard process. There are tools you can buy to perform the basic operations of reading/writing firmware, the most popular being the PC-3000 toolkit.

Typically the drive will not correctly detect in the BIOS. Instead of this vintage drive detecting as a 6E040L0 model with 40 GB of capacity, a common firmware corruption will cause that drive to detect with 0 GB capacity and as N40P. A more modern example is the Intel 8MB bug. When the firmware has a corruption, instead of this SSD drive detecting as an Intel 320 series with 160GB of capacity, it will detect as BAD_CTX with an error code and 8MB of capacity.

Ten percent of the cases we see here at Gillware have healthy internals, electronics, and the data user area (partition tables, file system meta-data, binary user data) is healthy as well. They show up here because the only problem is they have a firmware bug. 10 percent may not seem like a lot, but there are other uses for firmware manipulation besides repair. And as a company that has spent millions of dollars to increase success rates, I'll tell you the ability to recover data on an extra ten percent of cases is huge.

Anytime we are performing internal part replacement on the guts of a drive, the parts are not quite perfect from a compatibility perspective. Even if we source a headstack from a donor drive that came off the same production line on the same day, it's slightly different. This incompatibility may cause the drive to click and not be detected. This incompatibility may allow the drive to boot but is unable to read data off of one of the surfaces. Advanced data recovery companies will actually use their knowledge of firmware to tweak the drive's behavior to better deal with these circumstances.

For example, you may want to tell the drive not to spam its SMART tables with all kinds of error logs. The drive is already dead, we know it has major problems, so why run the risk of corrupting the SMART tables or the drive dying on the vine and being generally unresponsive because of all of its internal recordkeeping? Similarly you don't want a drive performing its natural day to day behavior of adding bad sectors to defect lists for reallocation. First off, we don't want to take the performance hit. Second, if the growth defect lists overrun it may cause the drive to die again. Third, those sectors really aren't bad; the head just can't read them right now. We plan on replacing that head after we read the other surfaces, and we don't want these good sectors being falsely identified as failed and getting reallocated.

We can also manipulate the basic drive boot sequence. Some drives will run a diagnostic heads check every time you power them up, and if one of the heads is out of a certain tolerance range it will not allow the drive to boot. We will commonly tweak the O/S or manipulate that boot sequence to ignore these checks and give us binary read access to the surfaces with compatible heads.

Recovery Engineering at Gillware

I've been mucking around inside the guts of hard drive operating systems for about 12 years now. My computer science degree from the University of Wisconsin (especially the operating systems course) and my programming background gave me a nice jumpstart. I spend most of my "recovery time" these days building the training programs and running the certification program. The heart of the training is prerequisites that they have ideally learned in school. Understanding hexadecimal math, multithreading, basic computing concepts like XOR or little endian, how checksums work, junk like that.

Young engineers here at Gillware will on average study HDD O/S theory for about six months before passing their internal first level certification exam. This internal certification allows them to participate in entry-level hard drive evaluation on client cases.

I personally work on one or two firmware related cases a month. This typically happens either on a very new hard drive or a super vintage hard drive. Every once in a great while someone will send in a 200MB hard drive manufactured a few decades ago, suspected to have a firmware bug, that most of the staff hasn't yet been exposed to. So they dig me out of mothballs, we all get a good laugh at my expense, and I show the whipper-snappers what's what. About once a week a senior engineer will need my assistance with developing a workaround for a new firmware bug that we just have never seen before. This is usually because the drive was built only a couple months ago, but sometimes it is just a crazy rare bug that isn't a part of our training program. So we reverse engineer the problem together, ideally fix the problem and get the customer their stuff back, and then I add it to the training.

The biggest problem with these rare cases is time. Usually the research will take at least a week and if it is a particularly nasty problem I'll check with some of my peer group at manufacturers or other labs to see if they've ever encountered it before. In general most of the folks within our industry fiercely protect their R&D and take a lot of pride in their unique understanding and solutions. Some, like ACE Labs and their competitors, actually make their money by doing this research and selling solutions to common problems.

It is easy to understand why engineers fight fiercely to protect their hard earned skills. If I've spent three weeks of my life working on a single case, ultimately gathering hard-earned knowledge that leads to a profitable solution set for Gillware, the human instinct is to protect it and keep it a secret. Having said that, when I know another engineer that is just as smart and just as passionate about the topic, I'm happy to discuss how I'd approach their problem. It's extremely easy to do that if they've actually helped us out in the past. There are a handful of souls within our little industry that feel the same way and it's in the best interests of our collective clients. If someone is a junior level tinkerer just getting into the profession, has no programming background and doesn't work for me, I'm going to encourage them to seek paid training like you can receive at Deepspar or ACE and leave me alone. Gillware makes money by performing services, not training people to compete with us.

The truth is that hiring, training and retaining the humans capable of doing this type of work is a huge challenge, not just for Gillware but our whole industry. The folks that do this for me have the degrees and skills to work at Google/Facebook/Apple/Intel. Their ability to do complex troubleshooting and program solutions is amazing. In truth, we've got more than one engineer that run circles around me at this point. That's the way it should be. Employing these humans and keeping them happy financially is the major reason data recovery can be expensive.

Conclusion

Storage device O/S access and manipulation is definitely a key part of any data recovery lab's success. How good would a computer repair technician be if he didn't understand basic Windows Operating System troubleshooting? They, of course, have the benefit of Microsoft supplying operating system disks, troubleshooting software and training. They also have the benefit of just internet searching for 'HAL.DLL missing' and reading 370 troubleshooting tips and exact instructions on what to do. In the small and secretive world of data recovery we don't have as much luck unfortunately.

The challenge of reverse engineering complicated electronic devices, and applying that knowledge to get people out of jams, is actually fun for a certain type of engineer. It would be a lot more fun if the stakes and anxiety wasn't so high for all our clients. If you've got a computer engineering and programming background, and dedicate about three years of your life to it, you'll get pretty decent at it. A scientific background and ten thousand hours will make you a master of this very odd specialty. I'd estimate there are less than 300 humans worldwide that have put in these 10,000+ hours. I've met about 12 of them, and none of us want to do it very much anymore but it's very difficult to get away from it entirely.

Knowledge of the storage device's operating system is of paramount importance to any company serious about data recovery. It is when you use it in conjunction with other skills like electrical or physical rework that the applied knowledge is truly useful.

Gillware™ Data Recovery

Contact Us Gillware Data Recovery
1802 Wright Street
Madison, WI 53704

Tel: 877-624-7206
Email: partners@gillware.com
https://gillware.com