Replacing a failed hard drive in a Barracuda Spam & Virus Firewall 400

Symptoms of a disk failure: On the main admin page, under "Performance Statistics," the Barracuda displays a message like "RAID Unit 0: Drive 1 has failed. Contact Barracuda Networks Technical Support."

The "correct" course of action is to have purchased an "instant replacement" support plan from Barracuda Networks; Barracuda will then presumably ship you a replacement unit and migrate all of your old unit's functions and configuration over to it transparently, then ask you to send the old one back.

Some newer Barracuda models probably have easily swappable disks (and in that case, Barracuda Networks could send you a disk, rather than a whole unit), but our model 400's drive bays are not accessible from the outside at all.

What to do if you don't have an Instant Replacement plan

The Barracuda 400 has a dual hard drive RAID-1 configuration. When a disk has failed, it needs to be replaced immediately, because the unit may refuse to boot if only one drive is operational (the initramfs configuration is a bit dodgy; more on that later). So, leave the unit running until you can afford some downtime. Replacing the disk will require 1 to 5 hours in total, during which the unit will be offline.

The RAID is Linux software RAID, using mdadm. There is a configuration file at /etc/mdadm.conf, and there is another copy of this configuration inside the initramfs image that runs at boot time. This means that unless you can work out how to regenerate initramfs, changing /etc/mdadm.conf will not affect the boot process.

To replace the disk, you will need:

- A new/clean/tested replacement SATA disk at least 250 GB in size (you can find out exactly what size your unit's disks are from Barracuda support). I strongly recommend using a disk smaller than 2 TB if possible, as the Barracuda's kernel or SATA subsystem may not support large drives.
- Various sizes of cross-head screwdrivers (Phillips, if you're in the habit of using the screwdriver words that don't actually describe the driver/head at all...)
- A method of keeping lots of small screws organized, and labelling where they go
- A PS/2 keyboard
- A VGA monitor (i.e. one with a D-sub/standard VGA)
- A second computer with at least two SATA ports that will boot to Linux (I recommend using a USB boot stick with no other disks connected to any SATA/IDE ports, to reduce the chance of screwing something up). smartmontools (smartctl) and mdadm should be available.
- Good working knowledge of command line Linux and software RAID (mdadm) operations, i.e. you should be familiar with commands like "mdadm --assemble /dev/md0 /dev/sda2"

Procedure:

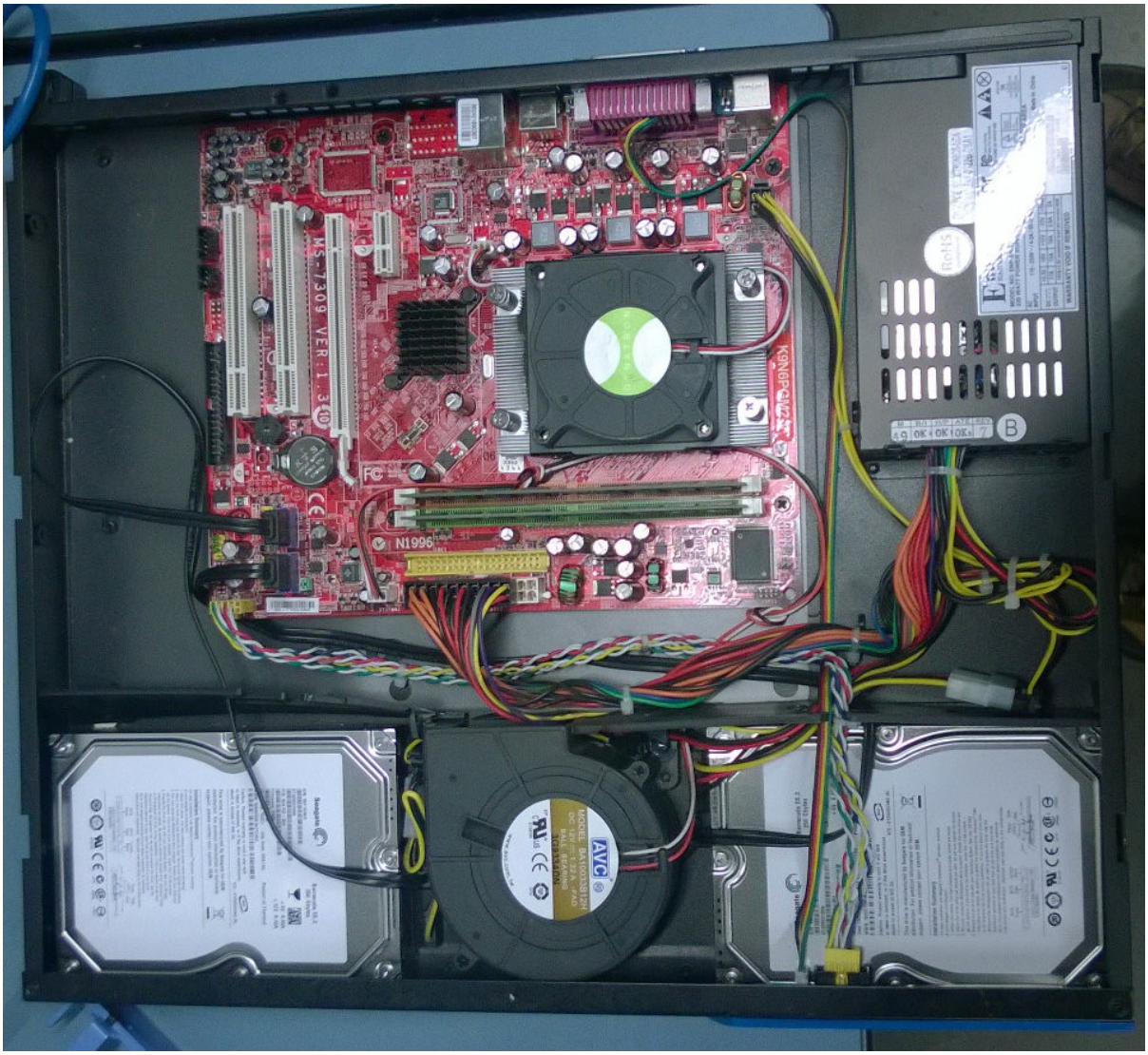
Overview: We will be removing both disks from the Barracuda, setting the bad one aside, and moving the good one to our additional computer. We'll then add the new/replacement/blank disk to the

additional computer as well, then resynch the RAID arrays. Finally, we'll move the disks back to the Barracuda, then see whether it still boots.

Warning/disclaimer: You should read through all the directions before you begin. You should only proceed if:

- You're OK with voiding the warranty on your unit, because this *will* void the warranty.
- You are comfortable/knowledgeable enough to do each of these steps.
- You are OK with the notion that sometimes these steps might not go as described, and then you'll have to improvise to fix things.
- You're the sort of person who thinks it goes without saying that you have backups. There are a few places in this procedure where I've left out "and make a backup in case you fuck this up" instructions, like when I'm telling you to blow away a partition. You should make complete images of your Barracuda if you can afford the downtime.
- You know enough about MX records, mail servers, and routing to get your e-mail working again even if you completely destroy your Barracuda and they can't ship you a new one for several days.

1. Call Barracuda Networks support and tell them you have a disk failure message. If possible, see if you can get a tech to log in to your unit remotely and tell you the serial number of which disk has failed, and which disk is still good. (In my experience Barracuda technicians are quite knowledgeable and will do this for you, even if you don't have a replacement plan.)
2. Make a backup of your Barracuda's configuration.
3. Shut down the Barracuda from the web admin interface (Basic > Administration > Shutdown button at the bottom).
4. Remove the Barracuda from your rack, as you will need to remove screws from all surfaces. Also remove any rack mount brackets from the sides/corners of the unit.
5. Unscrew and remove the top panel. One of the screws is under a "Warranty void if removed" sticker. The unit I worked on looked like this:



As you can see, the disks are partly covered by the lip of the front plate, some wiring, and are wedged in with black pieces of plastic. The black plastic pieces are glued onto the faring of that cooling fan in between the disks, but they are not glued at the other end.

6. Take note of how the two connectors (one yellow, one small white) are plugged into the front plate button and LED panel -- this is the panel with the power button on it. Then unplug these, so that there is no wiring connected to the front plate or anything attached to the front plate.
7. Remove the front plate (this is the plate with the Barracuda logo on it, at the bottom of the above picture). There are screws on the sides, top, and bottom. Note that four of the bottom screws are actually disk screws, and the side screws are a different type to the top screws, so you should keep track of which screws go where.
8. Pull the outer ends of the black plastic pieces (between the hard drives and the mainboard) away from the disks, as much as you can without straining or breaking anything.
9. Carefully undo the other two screws holding each disk in place (you'll have to get at the bottom of the case to do this, so if you prop it up, be careful that the disk doesn't flop loose and hit something).
10. If you know the serial numbers of the good & bad disks, label them "probably good" and

"probably bad" now. Also label which SATA connector each disk was connected to. Lift each disk far enough away that you can undo its power and SATA connectors, then pull it out.

11. Make sure the secondary computer will definitely NOT try to boot from its SATA ports before you connect any of the Barracuda's disks to it. If you can't guarantee this, boot the secondary computer with no SATA disks connected. If the secondary computer tries to boot the Barracuda disk, hit Ctrl+Alt+Del or the reset button before it gets past the bootloader screen. Also do NOT use a Linux distribution that will auto-mount partitions, and especially do not use one that will auto-start and attempt to auto-repair a RAID array (because there is a danger that it will attempt to auto-synch the bad drive onto the good one, thus destroying everything).

(Hint: If your secondary computer is already running (because you couldn't otherwise guarantee that it would not attempt to boot from the Barracuda disks), the command to make Linux see the Barracuda disks after you hot-connect them is: `echo "- - -" > /sys/class/scsi_host/hostX/scan` (where X is a number -- just try all the host entries under scsi_host until your disk appears in `dmesg | tail`))

12. If you don't know which drive is the bad one, put both disks in your secondary computer and use smartctl to see whether one disk reports itself as dead/dying in SMART. If one disk is very clearly bad (won't even spin up, computer won't even recognize it, it makes bad noises, or SMART has a lot of error log entries and bad sector counters are high), label it as the bad disk. If both disks spin up and SMART info looks OK on both of them, or both have bad sectors or errors logged, try assembling the RAID array that was the swap partition (you'll have to identify this by taking a read-only look at its header); mdadm should say "kicking non-fresh disk from array" about one of the disks; the disk it kicks out is the bad one.
13. Set the "probably bad" disk aside and put the "probably good" disk in your secondary computer.
14. Dump the partition table on the good disk, e.g. with `sfdisk --list` and save it somewhere (e.g. /tmp). I also recommend dumping the bootloader sectors, so that you can make the new disk bootable, but this is not strictly required if the failed disk was not the boot disk.
15. Spin down and remove (or disconnect) the good disk (hint: `echo 1 > /sys/class/block/sdX/device/delete` will gracefully spin down the disk for removal)
16. Put the new/blank/replacement disk in your secondary computer and partition it to match *exactly* with the partition table from the good disk. That is, when you dump this partition table, you should get an identical result to the old one (except the total physical size of the disk). This includes partition types: if the type was "Linux software raid member (fd)" on the old one, set the same thing on the new one.
17. Reconnect the good disk to the secondary computer.
18. Now attempt to start the RAID arrays and then add the matching partitions from the new disk.
19. Wait for all the arrays to rebuild.
20. Once all the arrays have rebuilt, mount the Barracuda's root partition somewhere, and make the following changes:
    - make a backup of the Barracuda's /etc/shadow
    - blank out the root password in the Barracuda's /etc/shadow
    - comment out the grub boot password in the Barracuda's /boot/grub/menu.lst
21. Copy the Barracuda's /etc/mdadm.conf file somewhere outside of the Barracuda disks.
22. The Barracuda uses RAID device names like /dev/md0, /dev/md1, etc. The digit in the device name matches the device's preferred minor device number (look up "major/minor device

numbers" if you're not familiar with this — these are from before the days of udev). For each RAID array, do this:
mdadm --detail /dev/mdX
in the detail output, get the arrays UUID
look up this UUID in the mdadm.conf file
then compare the /dev/mdX name in the .conf file with the "preferred minor" field in the detail output

23. For any RAID arrays that have become assembled with minor device numbers that no longer match what's in the Barracuda's mdadm.conf, you will need to fix these. To do that for an array:
stop the array
reassemble the array using the `--update=super-minor` parameter (see the mdadm man page for details)
24. Once all the RAID arrays have preferred minor numbers that match what they were on the Barracuda, shut down your secondary computer and move the good disks back to the Barracuda, connecting the good original disk back to the same SATA connector it was on before, and the new disk to the other SATA connector.
25. Reconnect the faceplate power button and LED panel (don't screw anything back in yet unless you want to; we're just going to see if it boots first).
26. Connect keyboard, monitor, and power to the Barracuda. Make sure network is NOT connected.
27. Press the power button and see if it boots.
28. If it does boot, log in as root (you will be able to do this with no password, since we removed it from /etc/shadow) and check the RAID arrays.
29. If everything looks good, shut down the Barracuda (`shutdown -h 0` should work), reassemble everything, put it back in the rack, and boot it back up for real. Don't forget to set a new root password (or put the old one back).

Closing notes/troubleshooting:

- If the boot process dies partway through: remember that we removed the grub password, so you can edit the boot line and remove the "quiet" parameter, to see what's really going on.

- Why do you need to fiddle with the preferred/super minor numbers in the RAID arrays? Because Barracuda is using a mixture of "the new way" and "the old way" of doing software RAID. There are a few options for ensuring that your RAID arrays always boot up with the right names:

A. Use explicit assembly only. No autodetecting at all, you tell mdadm, "I want you to assemble an array called /dev/md0 and I want you to use the first partition on my first IDE disk, and the first partition on my second IDE disk." In your fstab and bootloader/initramfs config you can use /dev/mdX style names, because these are fairly guaranteed to be stable. This is "the old way", because this was stable with IDE device names (/dev/hda was primary master, hdb primary slave, etc.), but is harder to do with SATA because SATA drives can spin up in any order and /dev/sda is not guaranteed to be sda next time. (But you can still do it via /dev/disk/by-path if you want.)

B. Use auto-assembly with /dev/mdX style names, based on the superblock "preferred minor" number. These are stable only as long as the disks are never run from another computer, and all updates to the boot configuration are only done on the disks' "home" operating system. If you attach the disks to another computer or boot to another device in the same computer, the minor numbers might

change. Sometimes they might change for no good reason during a kernel update, and it's also easy to make them change by running the wrong mdadm command. In this case you should use UUIDs in fstab and bootloader/initramfs -- using /dev/mdX names is a bad idea because if a minor number changes, your system becomes unbootable (or worse, filesystems get mounted in the wrong place).

C. Use auto-assembly with explicitly named arrays. You can tell mdadm you want to name your array "george", and then whenever mdadm assembles that array, it always gets that name. Then you can use /dev/md/george in your fstab.

Barracuda has chosen to use method B, and to also do exactly what I said is a bad idea for method B.

- Be careful about the partition layout. At least on our particular unit, the partition numbering went something like 1, 2, 5, 6, 7, 8, 9, 3. It's as if someone was partitioning it, then at the end forgot they needed one more primary partition, and said, "meh, ship it like this anyway". This kind of thing makes reassembling RAID arrays trickier and more dangerous.

- I have not tried to run the Barracuda's grub-install to get the bootloader onto the new disk. I did this with dd instead:

1. before partitioning, copy the first few MB of raw sectors from the old disk onto the new one
2. do the partitioning
3. when the partition tables match, wipe the first partition on the new disk with zeros, to prevent it being autodetected by mdadm