Disk Drive Sanitization

How to safely remove data from various devices

- Applies to all: Encryption!
- Dumb but effective: overwrite data (HDD's and some SSD's)
- SATA SSD's: ATA Secure Erase
- NVME SSD's: NVME Secure Erase

Applies to all: Encryption!

The universal method for securely removing all data from a drive is to encrypt all the data in the first place!

Erasing encrypted drives with the "Dumb" method MAY NOT always be complete, however. Encrypted drives need a section of data holding the decryption keys, which are unlocked at boot with your password. This data block WILL be erased when an encrypted partition table is erased, but over time, the block/sector the data is stored on may fail. If this happens, the drive copies the data to a new location and abandons the bad block/sector, with the data intact. This block is not acessible from the OS, but could theoretically could still be acessed and read.

Utilizing encryption from the start ensures that, no matter what formatting method is used (even the "Dumb method"), any data remaining on dead secotrs/blocks is properly useless.

Self-Encrypting Drives

Some SSD's support self-encrypting, making secure erase as simple as scrambling the stored the keys saved on disk. More info here (Arch only).

Windows: BitLocker

On Windows, set up Bitlocker on the C: drive (requires Windows Pro). This is done AFTER installing WIndows, simply right click on C: and select "Turn on BitLocker". You will have to set up either a USB key or password/PIN to unlock the drive when booting.

To erase securely, follow the Windows directions in the other sections depending on drive type and secure formatting features.

Linux: LVM + LUKS

For non-encrypting SSD's and HDD's, Arch supports encrypted LUKS volumes, which can either act directly as a partition, or can be a container to house multiple partitions. a single LUKS volume with multiple partitions is the easiest to manage, as you can use the same key to encrypt root, /home, swap, etc.

Only the boot partition remains unencrypted. Same as self-encrypting drives, securely erasing a LUKS encrypted drive is as easy as erasing the LUKS header data from the volume (which stores the keys required to decrypt.)

More info here (Arch) on setting up an enecrypted volume.

More info (also Arch) on preparing/wiping an encrypted disk.

Dumb but effective: overwrite data (HDD's and some SSD's)

For Spinning disk drives, and SSD's that don't support secure erase/sanitize commands, the only other option is to manually overwrite all the data on the drive.

Why this is the least secure method of erasure:

HDD's and SSD's self-manage bad sectors/blocks. When a bad sector/block is detected, the data is copied to a new location, and the bad memory is left as-is and marked to be skipped. The OS cannot see or write to these sectors/blocks, so sensitive data left in them will remain even after overwrites. Risk is mitigated by Encrypting the drive and data from the start, which ensures that any data left on dead sectors/blocks is approriately scrambled.

Windows:

Use the built-in `format` command to erase drives. This takes a while and has NO FEEDBACK, so it's a "let it run until it's done" thing. This only works on drives that are NOT the boot drive. Steps:

1. Using the disk manager, "quick format" the drive to have one single partition that spans the ENTIRE drive (with no space before or after the partition) and assign a drive letter and name (ex, drive K:)
2. In a cmd window, run the command `format k: /X /P:1` [format drive k:, unmount before formatting, zero all sectors, then 1 pass of pseudorandom data]

Linux

1. Using `gparted` or another GUI/CLI partition manager, "quick format" the drive by deleting the partition table and creating a new one with no encryption. This step is optional, but is an additional step to ensure that any encrypted LVM headers are at least zeroed out. note the drive name, eg `/dev/sdX` or `/dev/nvmeX`
2. run the command `dd if=\dev\urandom of=\dev\nvmeX bs=4M`

Physical Destruction:

Drives that are being recycled or otherwise disposed of should be physically destroyed. After overwriting the data as described in the other sections here, dissassemble the drives and destroy

them mechanically:

For HDD's, smash the platters with a hammer.

For SSD's, destroy the memory chips/circuit board, either a hammer or blender (not used for food) is recommended.

SATA SSD's: ATA Secure Erase

MOST SATA-based SSD's support some type of secure-erase. This is a much faster and more effective method of erasing data than the one as described in Dumb but Effective.

Instead of overwriting with pseudorandom data at the OS level, which may leave some bad blocks in tact, an SSD that supports ATA Secure Erase will use various methods of erasure on ALL BLOCKS on the drive, including the damaged/unavailable ones. This has the same effect as overwriting the drive with data, but with the added benifit of also overwriting the bad blocks.

Note that supported drives may be "frozen" by the BIOS, usually because of Secure Boot. Disable Secure Boot if SSD's refuse to "unfreeze"

Linux:

Best: Sanitize on tinyapps.org

1. install `hdparm`
2. check drive supports sanitize with `hdparm --sanitize-status /dev/sdx`. Supported sanitize features will be displayed for supported drives, options are block erase, crypto scramble, and overwrite.
3. run a supported sanatize command:
   a) `hdparm --yes-i-know-what-i-am-doing --sanitize-block-erase /dev/sdx`
   b) `hdparm --yes-i-know-what-i-am-doing --sanitize-crypto-scramble /dev/sdx`
   c) `hdparm --yes-i-know-what-i-am-doing --sanitize-overwrite-passes 1 --sanitize-overwrite hex:11111111 /dev/sdx`

Alternate, less secure: Secure Erase on tinyapps.org

1. install `hdparm`
2. check drive supports erase with `hdparm -I /dev/sdx`. it may be frozen, sleep and wake the PC to fix.
3. Set password (arbitrailly set to "p") to enable secure erase: `hdparm --user-master u --security-set-pass p /dev/sdx`
4. erase drive with: `hdparm --user-master u --security-erase p /dev/sdx` (use `--security-erase-enhanced` if supported by the drive).

Windows:

Use the secure erase software from each manufacturer, ie Kingston, Samsung, Crucial, etc:

Ex:

- Samsung Magician Software (May ask to make a bootable USB to secure erase)
- Kingston SSD Manager (all partitions must be deleted first, see instructions.)

NVME SSD's: NVME Secure Erase

ATA and NVME drives use different communication specs, and as such the secure erase/sanitize commands are different. Sanitize overwites all data

Linux:

Best, Sanitize. Source: NVME Sanitize on tinyapps.org

1. Install `nvme-cli`
2. list NVME drives with `nvme list`
3. Check device is supported with `nvme id-ctrl -H /dev/nvmeX`. Check the 'fna' section, if any features have a '0x1'instead of a '0', sanitize is supported. Otherwise use Secure Erase.
4. Run the command `nvme sanitize -a Y /dev/nvmeX` where `Y` is 1, 2, 3, or 4 depending on supported sanitize features:
   a) 1 = exit failure mode
   b) 2 = Block Erase (Does a hi-low pulse on all blocks to reset them all to 0)
   c) 3 = Overwrite (random data overwrite)
   d) 4 = Crypto Erase (delete/change crypto keys, only on encrypted drives)
5. Check Status with `nvme sanitize-log /dev/nvmeX`. Completed when SPROG=65535, and SSTAT=0x101

Alternate, Secure Erase. Source: NVME Secure Erase on tinyapps.org

1. Install `nvme-cli`
2. list NVME drives with `nvme list`
3. Check device is supported with `nvme id-ctrl -H /dev/nvmeX`. if 'oacs' section, option [1:1] is set, Secure erase is supported. If 'fna' section, option [2:2] is set, then cryptographic secure erase is supported as well.
4. trigger the secure erase with `nvme format /dev/nvmeX --ses=Y` where `Y` is 0, 1, or 2, depending on supported features:
   a) 0 = no secure erase
   b) 1 = User Data Erase (random data overwrite)
   c) 2 - Cryptographic Erase (delete/change crypto keys, only on encrypted drives)

Windows:

See the Windows section under the SATA page, method is the same.