TLDR; Just read after part 2.

Part 1 (Monday)

Nothing came out here, but I put it just to remember how lost I was when I started.

- vm with 4 devices
- make a partition taking up the whole device for /dev/sdb
- `dd if=/home/baz of=/dev/sdb1`
- tried to m ount unsuccessfully `mkdir /home/pesho; mount /dev/sdb1 /home/pesho`

- got output which means that the fs type is wrong and the superblock is broken, a:

```
mount: wrong fs type, bad option, bad superblock on /dev/
       missing codepage or helper program, or other error

       In some cases useful info is found in syslog - try
       dmesg | tail or so.
```

- as suggested form the output I run `dmesg | tail` and got but could not deduce anything:

```
[ 14.165735] EXT4-fs (sda1): re-mounted. Opts: errors=rem
[ 14.260291] loop: module loaded
[ 14.521401] IPv6: ADDRCONF(NETDEV_UP): eth0: link is not
[ 14.524351] e1000: eth0 NIC Link is Up 1000 Mbps Full Du
[ 14.524616] IPv6: ADDRCONF(NETDEV_CHANGE): eth0: link be
[ 20.510482] IPv6: ADDRCONF(NETDEV_UP): eth1: link is not
[ 20.512302] e1000: eth1 NIC Link is Up 1000 Mbps Full Du
[ 20.512581] IPv6: ADDRCONF(NETDEV_CHANGE): eth1: link be
```

```
[141.535343]  sdb: sdb1
[141.536371]  sdb: sdb1
```

- run `fdisk /dev/sdb1` and got:

```
Device does not contain a recognized partition table.
Created a new DOS disklabel with disk identifier 0x92d9ec

Command (m for help): p
Disk /dev/sdb1: 63 MiB, 66060288 bytes, 129024 sectors
Units: sectors of 1 * 512 = 512 bytes
Sector size (logical/physical): 512 bytes / 512 bytes
I/O size (minimum/optimal): 512 bytes / 512 bytes
Disklabel type: dos
Disk identifier: 0x92d9edc6
```

- installed `gparted` to play with it.

- run `parted -l` and got:

```
Model: ATA VBOX HARDDISK (scsi)
Disk /dev/sdb: 67.1MB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:

Number  Start   End     Size    Type     File system  Fla
 1      1049kB  67.1MB  66.1MB  primary
```

This means that there is no file system and no special flag(e.g. boot) However, We see that the partition table is msdos.

- Read in wikipedia that the partition entries begin at 446(+1BEhex) and that each partition entry is 16 Bytes.
- run `dd if=/home/baz of=/home/MBR_baz bs=1 count=512` to copy the whole MBR
- run `vi /home/PT_baz` and the `:%!xxd`, tried to compare everything with the slide from the lectures nothing
- hex output http://superuser.com/questions/354551/how-do-you-read-a-hex-partition-table
- as wikipedia stated the four expected entries should be grouped in 16 bytes starting at +1BEhex. Here it is 28 is the first from the MBR. Too weird

```
00001b0: e27b 40b3 16a3 6902 f36b f0b5 27a7 2800  .{@...
00001c0: 3165 fd9b 0961 3220 6480 a366 1282 8b90  1e...a
00001d0: 2066 e7ad 420b f0d8 3610 7a85 13a9 1408   f..B.
00001e0: 07f2 08cd fb77 c86e 89a2 e093 3444 a306  .....w
00001f0: 75f6 1d4a 1dd5 544c 07d4 aa8f 6df1 d09d  u..J..
0000200: 08b3
```

- something fishy about 2800 the first byte of the partition table. Probably messed up boot
- make partitioned device /dev/sdc to investigate correct partition table of one element

```
00001b0: 0000 0000 0000 0000 5262 f19f 0000 0020  ....
00001c0: 2100 8328 2008 0008 0000 00f8 0100 0000  !..(
00001d0: 0000 0000 0000 0000 0000 0000 0000 0000  ....
00001e0: 0000 0000 0000 0000 0000 0000 0000 0000  ....
00001f0: 0000 0000 0000 0000 0000 0000 0000 55aa  ....
```

- divide /dev/sdd/ into two partitions to see how everything is organized. open the the partition table

```
00001b0: 0000 0000 0000 0000 49c9 71b0 0000 0020  ....... 
00001c0: 2100 8325 2401 0008 0000 0040 0000 0025  !..%$..
00001d0: 2501 832a 2802 0048 0000 0040 0000 0000  %..*(..
00001e0: 0000 0000 0000 0000 0000 0000 0000 0000  ....... 
00001f0: 0000 0000 0000 0000 0000 0000 0000 55aa  ....... 
0000200: 0a                                       .
```

- divide /dev/sdd/ into three partitions to see how everything is organized

```
00001b0: 0000 0000 0000 0000 49c9 71b0 0000 0020  ....... 
00001c0: 2100 8325 2401 0008 0000 0040 0000 0025  !..%$..
00001d0: 2501 832a 2802 0048 0000 0040 0000 002a  %..*(..
00001e0: 2902 832f 2c03 0088 0000 0040 0000 0000  )../,..
00001f0: 0000 0000 0000 0000 0000 0000 0000 55aa  ....... 
0000200: 0a
```

- weird compared to the partition table of the original file
- erase previous partitions on /dev/sdc and create a new one to load the file
- so vim + xxd didn't work, du on baz and baz_play(a file which was opened with vim and then :!%xxd) showed, menaing i had to convert back to binary

```
du  baz baz_play
4.3M    baz
18M     baz_play
```

- so understood how vim hex editing works, tried fixing the first byte in 0x1be original:

```
00001b0: e27b 40b3 16a3 6902 f36b f0b5 27a7 2800  .{@...i
00001c0: 3165 fd9b 0961 3220 6480 a366 1282 8b90  1e...a2
```

```
00001d0: 2066 e7ad 420b f0d8 3610 7a85 13a9 1408   f..B..
00001e0: 07f2 08cd fb77 c86e 89a2 e093 3444 a306  .....w.
00001f0: 75f6 1d4a 1dd5 544c 07d4 aa8f 6df1 d09d  u..J..T
0000200: 08b3
```

v1:

```
00001b0: e27b 40b3 16a3 6902 f36b f0b5 27a7 8000  .{@...i
00001c0: 3165 fd9b 0961 3220 6480 a366 1282 8b90  1e...a2
00001d0: 2066 e7ad 420b f0d8 3610 7a85 13a9 1408   f..B..
00001e0: 07f2 08cd fb77 c86e 89a2 e093 3444 a306  .....w.
00001f0: 75f6 1d4a 1dd5 544c 07d4 aa8f 6df1 d09d  u..J..T
0000200: 08b3
```

v2:

```
00001b0: e27b 40b3 16a3 6902 f36b f0b5 27a7 8020  .{@...i
00001c0: 3165 fd9b 0961 3220 6480 a366 1282 8b90  1e...a2
00001d0: 2066 e7ad 420b f0d8 3610 7a85 13a9 1408   f..B..
00001e0: 07f2 08cd fb77 c86e 89a2 e093 3444 a306  .....w.
00001f0: 75f6 1d4a 1dd5 544c 07d4 aa8f 6df1 d09d  u..J..T
0000200: 08b3
```

No success

- looked for the sequence 55aa in the hex this is the terminating symbol but could not find it in the disk image of baz
- http://thestarman.pcministry.com/asm/mbr/PartTables.htm

Part 2 (Wednesday)

- run `file baz` and got

```
baz: XZ compressed data
```

- to decompress the file run `xzcat baz1 > uncompressed`
- after decompressing we see something we can work with: There is the MBR partition table terminating bytes 55aa

```
00001b0: 0000 0000 0000 0000 c7ef b104 0000 0020  .......
00001c0: 2100 fd28 2008 0008 0000 00f8 0100 0000  !..( ..
00001d0: 0000 0000 0000 0000 0000 0000 0000 0000  .......
00001e0: 0000 0000 0000 0000 0000 0000 0000 0000  .......
00001f0: 0000 0000 0000 0000 0000 0000 0000 55aa  .......
0000200: 0a                                       .
```

- run `file uncompressed`:

```
 uncompressed: DOS/MBR boot sector; partition 1 : ID=0xfc
```

- run `fdisk /dev/sdc`

```
Device     Boot Start    End Sectors Size Id Type
/dev/sdc1        2048 131071  129024  63M fd Linux raid a
```

- run `fsck uncompressed`

```
fsck from util-linux 2.25.2
e2fsck 1.42.12 (29-Aug-2014)
ext2fs_open2: Bad magic number in super-block
fsck.ext2: Superblock invalid, trying backup blocks...
fsck.ext2: Bad magic number in super-block while trying t

The superblock could not be read or does not describe a v
filesystem.  If the device is valid and it really contain
```

```
filesystem (and not swap or ufs or something else), then
is corrupt, and you might try running e2fsck with an alte
    e2fsck -b 8193 <device>
 or
    e2fsck -b 32768 <device>
```

- make a new device `fdisk /dev/sdb` with identical parameters(64MB, linux raid autodetect):

```
00001b0: 0000 0000 0000 0000 3ee8 6759 0000 0017  ......
00001c0: 0101 fd24 2063 0008 0000 00f8 0100 0000  ...$ c.
00001d0: 0000 0000 0000 0000 0000 0000 0000 0000  ......
00001e0: 0000 0000 0000 0000 0000 0000 0000 0000  ......
00001f0: 0000 0000 0000 0000 0000 0000 0000 55aa  ......
0000200: 0a
```

Part 3(Friday)

- asked marian and boyan on Wednesday about hex editing and they looked me like crazy as if i am doing the last thing I would do.
- start over again by extracting the file `xzcat baz > foo`
- `dd if=foo of=/dev/sdb` the content of the file
- run `mke2fs -n /dev/sdb` and got:

```
Superblock backups stored on blocks:
        8193, 24577, 40961, 57345, 73729
```

- tried and nothing

```
e2fsck -b 73729 /dev/sdb
e2fsck -b 24577 /dev/sdb
e2fsck -b 40961 /dev/sdb
```

```
e2fsck -b 57345 /dev/sdb
e2fsck -b 73729 /dev/sdb
```

- run `dumpe2fs /dev/sdb` and got:

```
dumpe2fs 1.42.12 (29-Aug-2014)
dumpe2fs: Bad magic number in super-block while trying to
Couldn't find valid filesystem superblock.
```

- installed testdisk and it told that there is no bootable partition
- run `fdisk /dev/sdb` and press `a` thus raising the boot flag
- tried to mount on folder `pesho` but got error
- tried to assemble the raid after google the error with `mdadm --assemble /dev/md0 /dev/sdb1`
- run `cat /proc/mdstat` and got:

```
md0 : active (auto-read-only) raid1 sdb1[0]
      64448 blocks super 1.2 [2/1] [U_]
```

- `mount /dev/md0 pesho` prints `mount: unknown filesystem type 'crypto_LUKS'`

- `mdadm --detail /dev/md0`

```
Version : 1.2
  Creation Time : Sat Oct 22 16:14:17 2016
     Raid Level : raid1
     Array Size : 64448 (62.95 MiB 65.99 MB)
  Used Dev Size : 64448 (62.95 MiB 65.99 MB)
   Raid Devices : 2
  Total Devices : 1
    Persistence : Superblock is persistent

    Update Time : Fri Oct 28 22:24:41 2016
```

```
          State : clean, degraded
 Active Devices : 1
Working Devices : 1
 Failed Devices : 0
  Spare Devices : 0

           Name : callisto:0
           UUID : a7be0c5d:a38e8144:1ac2e275:9f2dccec
         Events : 19

    Number   Major   Minor   RaidDevice State
       0       8       17        0      active sync   /de
       2       0        0        2      removed
```

- nice tutorial http://www.ducea.com/2009/03/08/mdadm-cheat-sheet/
- started over because everything went south
- dd the content of the uncompressed image to a new device

```
mdadm --assemble --scan -v
```

- after running `cat /proc/mdstat`

```
md127 : active (auto-read-only) raid1 sdb1[0]
      64448 blocks super 1.2 [2/1] [U_]
```

- run `cryptsetup luksOpen /dev/md127 dido`, used the usual passphrase `asdf` and made a device mapper called `dido`
- run `ls /dev/mapper` and get `control dido vgmaya-turing`
- run pvdisplay and get

```
--- Physical volume ---
 PV Name               /dev/mapper/dido
 VG Name               vgmaya
 PV Size               60.94 MiB / not usable 4.94 MiB
```

```
  Allocatable           yes
  PE Size               4.00 MiB
  Total PE              14
  Free PE               4
  Allocated PE          10
  PV UUID               c3FxLW-aLnR-qwGj-Xxes-jvGn-Sej1-wf
```

- run `vgdisplay`

```
  --- Volume group ---
  VG Name               vgmaya
  System ID
  Format                lvm2
  Metadata Areas        1
  Metadata Sequence No  2
  VG Access             read/write
  VG Status             resizable
  MAX LV                0
  Cur LV                1
  Open LV               0
  Max PV                0
  Cur PV                1
  Act PV                1
  VG Size               56.00 MiB
  PE Size               4.00 MiB
  Total PE              14
  Alloc PE / Size       10 / 40.00 MiB
  Free  PE / Size       4 / 16.00 MiB
  VG UUID               GP2dez-BsjJ-HkjL-hNM0-5Fst-Jyjw-Kc
```

- `run vgscan`

```
  Reading all physical volumes.  This may take a while...
  Found volume group "vgmaya" using metadata type lvm2
```

- run `lvdisplay`

```
  --- Logical volume ---
  LV Path                /dev/vgmaya/turing
  LV Name                turing
  VG Name                vgmaya
  LV UUID                tjSo36-vJlQ-xUOQ-8n2I-qIWV-Mgbm-3
  LV Write Access        read/write
  LV Creation host, time callisto, 2016-10-22 16:18:47 +03
  LV Status              NOT available
  LV Size                40.00 MiB
  Current LE             10
  Segments               1
  Allocation             inherit
  Read ahead sectors     auto
```

- run `lvscan` and got: `inactive '/dev/vgmaya/turing' [40.00 MiB] inherit`
- since it is inactive, activate it with `lvchange -a y /dev/vgmaya/turing`
- mount `mount /dev/vgmaya/turing pesho/`
- `ls pesho/` and get cake.jpg
- we know that "the cake is a lie"

- 
The Cake is a Lie