

Silent Corruptions

KELEMEN Péter
IT-FIO

After C5
CERN



Corruptions

- Corruption: data is changed unintentionally
- mechanisms to detect/correct
 - parity, checksum (CRC32, MD5, SHA1, ...)
 - ECC
 - multiple copies with quorum
- detection: SW/HW-level with error messages
- correction: SW/HW-level with warnings
- Silent corruption: data is changed unintentionally **without** any errors/warnings!



Corruption Sources

- hardware errors (memory, CPU, disk, NIC)
- data transfer noise (UTP, SATA, FC, wireless)
- firmware bugs (RAID controller, disk, NIC)
- software bugs
 - kernel
 - VM
 - filesystems
 - block layer
 - application: crash, etc. (not discussed)



Expected Bit Error Rate (BER)

- NIC/link: 10^{-10} (1 bit in ~ 1.1 GiB)
 - checksummed, retransmit if necessary
- memory: 10^{-12} (1 bit in ~ 116 GiB)
 - ECC
- desktop disk: 10^{-14} (1 bit in ~ 11.3 TiB)
 - various error correction codes
- enterprise disk: 10^{-15} (1 bit in ~ 113 TiB)
 - various error correction codes
- quotes from standards/specifications



It Already Happened to Us

- “DON'T PANIC!”
- acknowledge user observation (if any)
- assess the problem
 - develop/deploy tools for data collection
- estimate the scale of the problem
- research the cause (correlation) and impact
- evaluate possible solutions
- deploy possible solutions

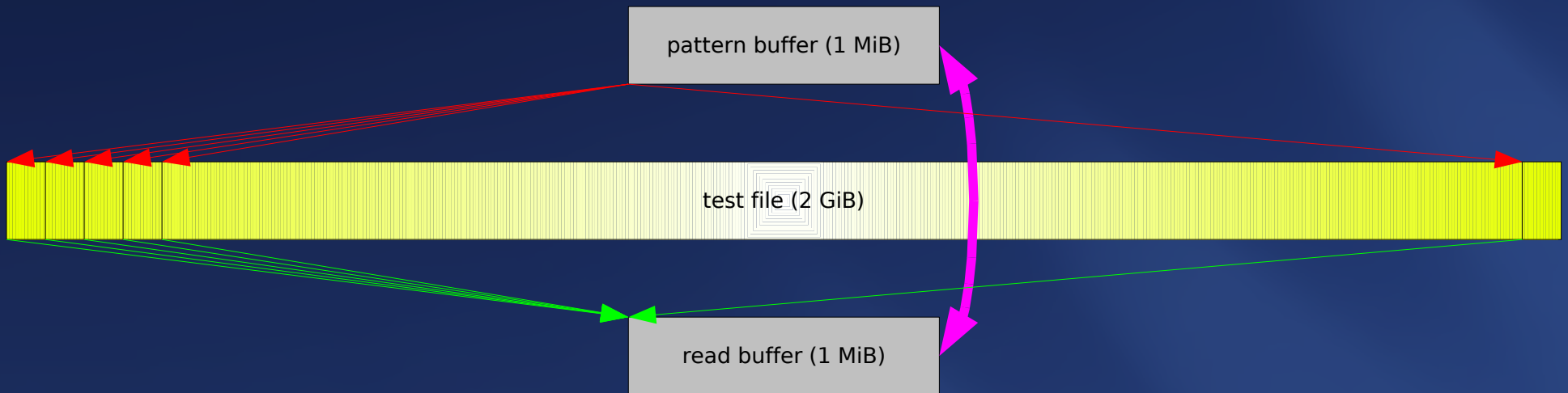


fsprobe(8)

- probabilistic storage integrity check
 - write known bit pattern
 - read it back
 - compare and alert when mismatch found
- low I/O footprint for background operation
- keep complexity to the minimum
- use static buffers
- attempt to preserve details about detected corruptions for further analysis



fsprobe cycles



ONE cycle completes in $2 \cdot 2048 \text{ sec} = 1 \text{ hour } 8 \text{ minutes}$

SIX cycles in $(2 \cdot 2048 + 300) \cdot 6 \text{ sec} = 7 \text{ hours } 20 \text{ minutes}$

0x55	[01010101]
0xAA	[10101010]
0x33	[00110011]
0xCC	[11001100]
0x0F	[00001111]
0xF0	[11110000]



Investigation

- fsprobe deployed on 4000 nodes
- 1400 incidents reported (total ~50PB traffic)
- 230 nodes affected (26 HW types)
- incidents tracked in SQLite
- steady flow of 1-3 incidents per day
- multiple types of corruptions observed
- affected systems are very diverse
 - SLC3/SLC4/RHEL4, XFS/ext3, 3ware/ARECA, ...
- some corruptions are transient



Corruption Types

- Type I
 - single/double bit errors
 - usually bad memory (RAM, cache, etc.)
- Type II
 - small, 2^n -sized chunks (128-512 bytes)
 - of unknown origin
- Type III
 - multiple large chunks of 64K, “old file data”
- Type IV
 - various sized chunks of zeros



Type I

- usually persistent
- bit(s) have flipped in a byte
- Single Bit Error (SBE)
- Double Bit Error (DBE)
 - DBEs are 3x more common than SBEs
 - a single case of a triple bit error was observed
- 1→0 transition more frequent than 0→1
- strong correlation with bad memory (verified)
- happens with expensive ECC-memory too



Type I Example

```
00000000 33 33 33 33 33 33 33 33 33 33 33 33 33 33 |33333333333333333333|
*
35285650 33 33 33 33 33 33 33 33 33 33 33 33 22 33 |3333333333333333"3|
35285660 33 33 33 33 33 33 33 33 33 33 33 33 33 33 |33333333333333333333|
*
80000000
```

0x33 = 00110011b
 ↓ ↓
0x22 = 00100010b



Type II

- usually transient
- small chunks of “random” looking data
 - ...but can go up to 128K
- sometimes identifiable user data
- observed in vicinity of OOM situations
- possible SLAB corruption?



Type II Example

```
00000000 cc cc cc cc cc cc cc cc | .....|
*
000def00 f1 2b f8 2b cd 43 38 38 e3 43 bd 8d b0 01 12 0a |.+.+.C88.C.....|
000def10 af 6e c3 b6 57 3e 5f fa e3 d6 e8 7b ef 5b 6f 3c |.n..W>_....{.[o<|
000def20 e4 42 42 0c e9 22 2e f1 d0 c6 a5 55 f2 f3 a7 38 |.BB..".....U...8|
000def30 b5 43 77 c9 5d 4e 16 a2 39 79 5f 31 10 65 b8 e4 |.Cw.]N..9y_1.e..|
000def40 9c 8a 94 a0 73 2d f7 ad d9 12 31 2b f5 db b4 18 |....s-....1+....|
000def50 e4 ff c6 14 ee 00 d6 c0 7a c8 8e c0 3f 73 32 73 |.....z...?s2s|
000def60 79 b3 63 d8 4c 8f 5d d8 c5 1e e4 5f 0e 2a 1d 94 |y.c.L.]....._*..|
000def70 f8 2d 64 e7 10 e1 6b 89 da b3 fb 0b 48 59 d4 df |.-d...k.....HY..|
000def80 9a 0c a4 ae 18 c5 40 a5 70 6b 19 d3 b9 f7 a2 b3 |.....@.pk.....|
000def90 44 df 2a 50 a1 55 31 02 57 d6 19 43 80 b0 0a 89 |D.*P.U1.W..C....|
000defa0 fe c2 34 ed cc 73 2e 64 38 89 6e 5a be d7 3b c4 |..4..s.d8.nZ...;.|
000defb0 db dd 58 42 a1 62 2f 6d 92 6a ed 9b 23 6e 1e 79 |..XB.b/m.j..#n.y|
000defc0 d8 88 38 86 92 2f af 29 a1 0f c0 21 46 fc 3a e3 |..8../.)...!F...|
000defd0 ac 17 0a f1 c3 31 89 82 59 e5 89 b9 9e fa 45 b9 |.....1..Y.....E.|
000defe0 54 d6 6a 72 b9 6a d6 1f ff cb 6a 10 1e bd 66 87 |T.jr.j....j...f.|
000defff0 68 80 64 b0 53 97 74 72 ee f6 87 9f 23 47 cb 48 |h.d.S.tr....#G.H|
000df000 cc cc cc cc cc cc cc cc | .....|
*
00100000
```



Type III

- usually persistent, comes in bursts
- strong correlation: I/O command timeouts
 - 3ware hides timeouts, look at extended diag
 - also observed on plain SATA systems
 - ...sometimes with failed READ commands!
- “previous” data from earlier cycles (sometimes multiple cycles old!) or from another location on disk
- seems to match RAID stripe size (64K)
 - observed on 16K chunk RAID arrays as well



Type III Example

```
00000000  cc cc cc cc cc cc cc cc  cc cc cc cc cc cc cc cc  |.....|
*
34205200  33 33 33 33 33 33 33 33  33 33 33 33 33 33 33 33  |3333333333333333|
*
34215200  cc cc cc cc cc cc cc cc  cc cc cc cc cc cc cc cc  |.....|
*
34265200  33 33 33 33 33 33 33 33  33 33 33 33 33 33 33 33  |3333333333333333|
*
34275200  cc cc cc cc cc cc cc cc  cc cc cc cc cc cc cc cc  |.....|
*
342c5200  33 33 33 33 33 33 33 33  33 33 33 33 33 33 33 33  |3333333333333333|
*
342d5200  cc cc cc cc cc cc cc cc  cc cc cc cc cc cc cc cc  |.....|
*
34325200  33 33 33 33 33 33 33 33  33 33 33 33 33 33 33 33  |3333333333333333|
*
34335200  cc cc cc cc cc cc cc cc  cc cc cc cc cc cc cc cc  |.....|
*
34385200  33 33 33 33 33 33 33 33  33 33 33 33 33 33 33 33  |3333333333333333|
*
34395200  cc cc cc cc cc cc cc cc  cc cc cc cc cc cc cc cc  |.....|
*
80000000
```



Type IV

- usually persistent
- relatively recent observations (since April)
- ...not sure yet this warrants another category

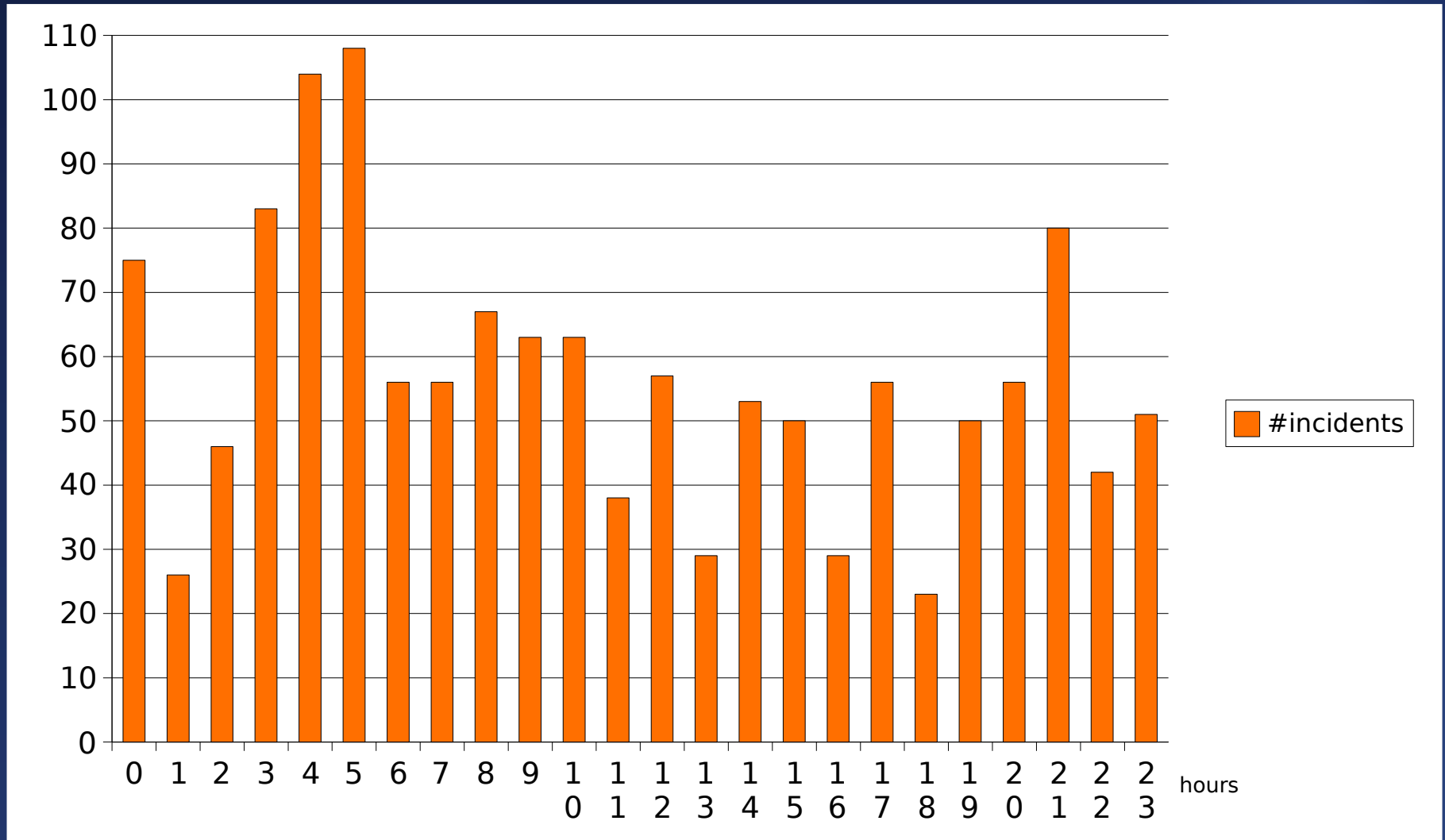


Type IV Example

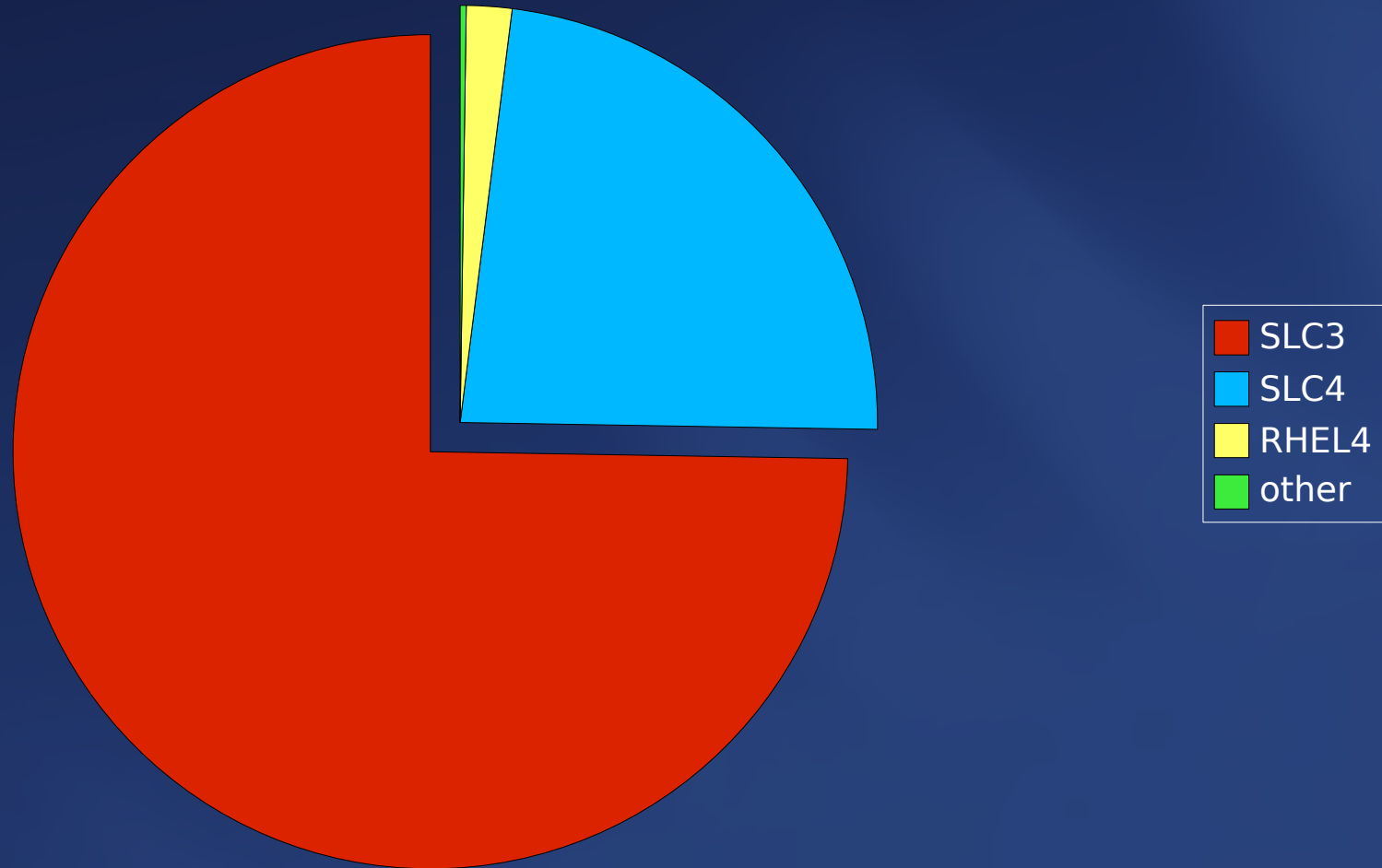
```
00000000 aa aa aa aa aa aa aa aa aa aa aa aa aa aa |.....|
*
00052980 00 00 00 00 00 00 00 00 00 00 00 00 00 00 |.....|
*
00053000 aa aa aa aa aa aa aa aa aa aa aa aa aa aa |.....|
*
80000000
```



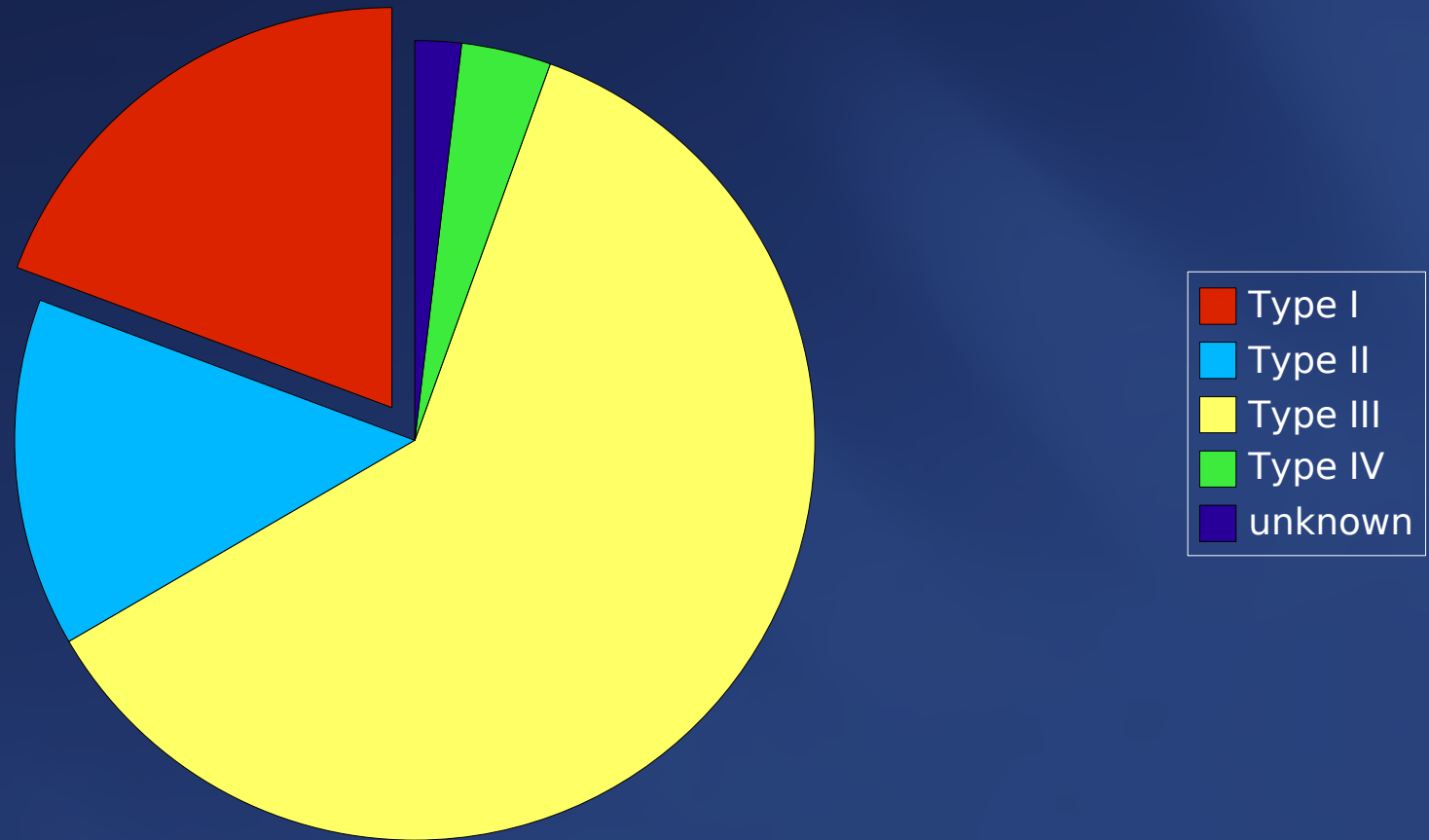
Corruption Time Distribution



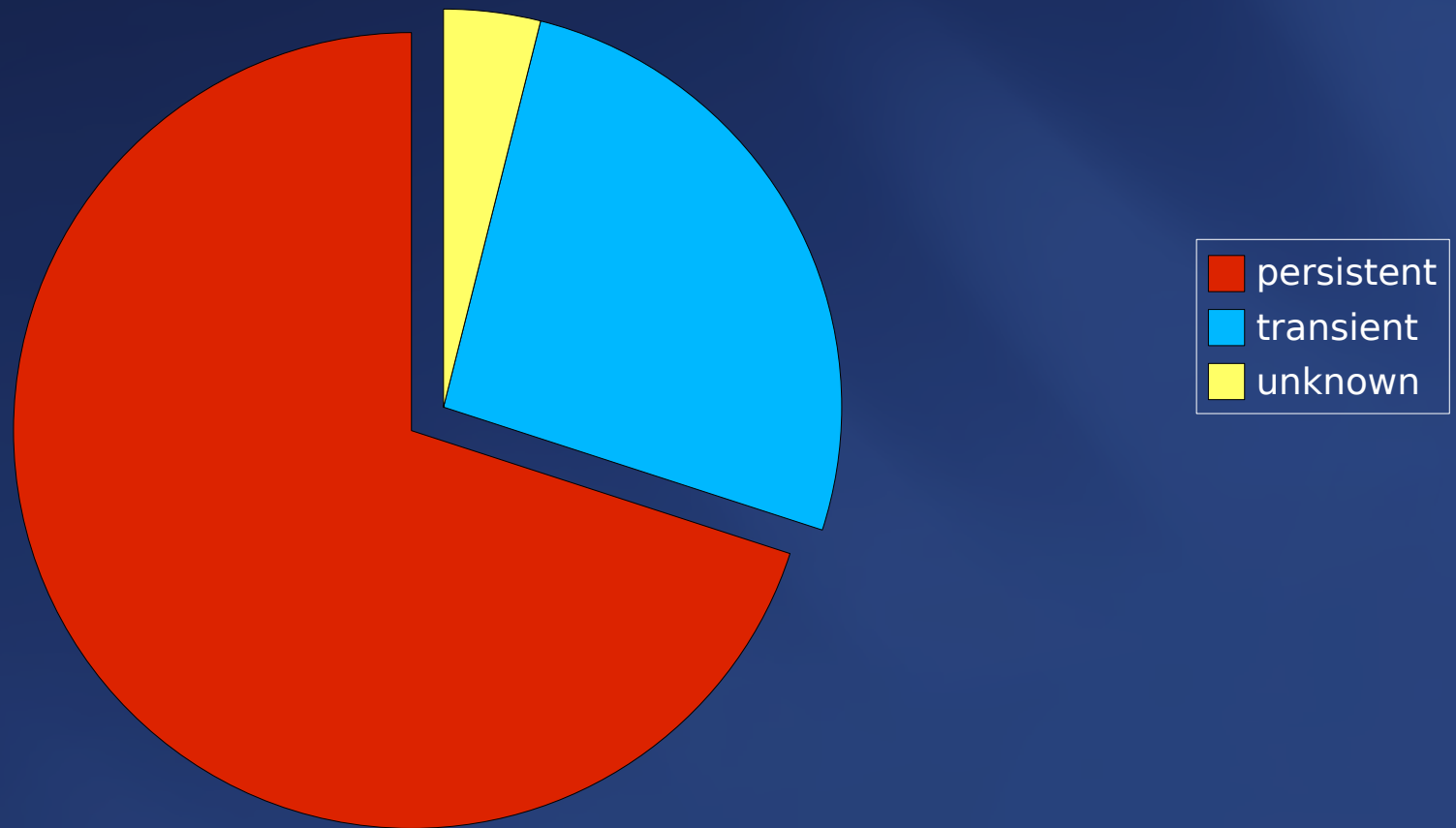
Operating Systems



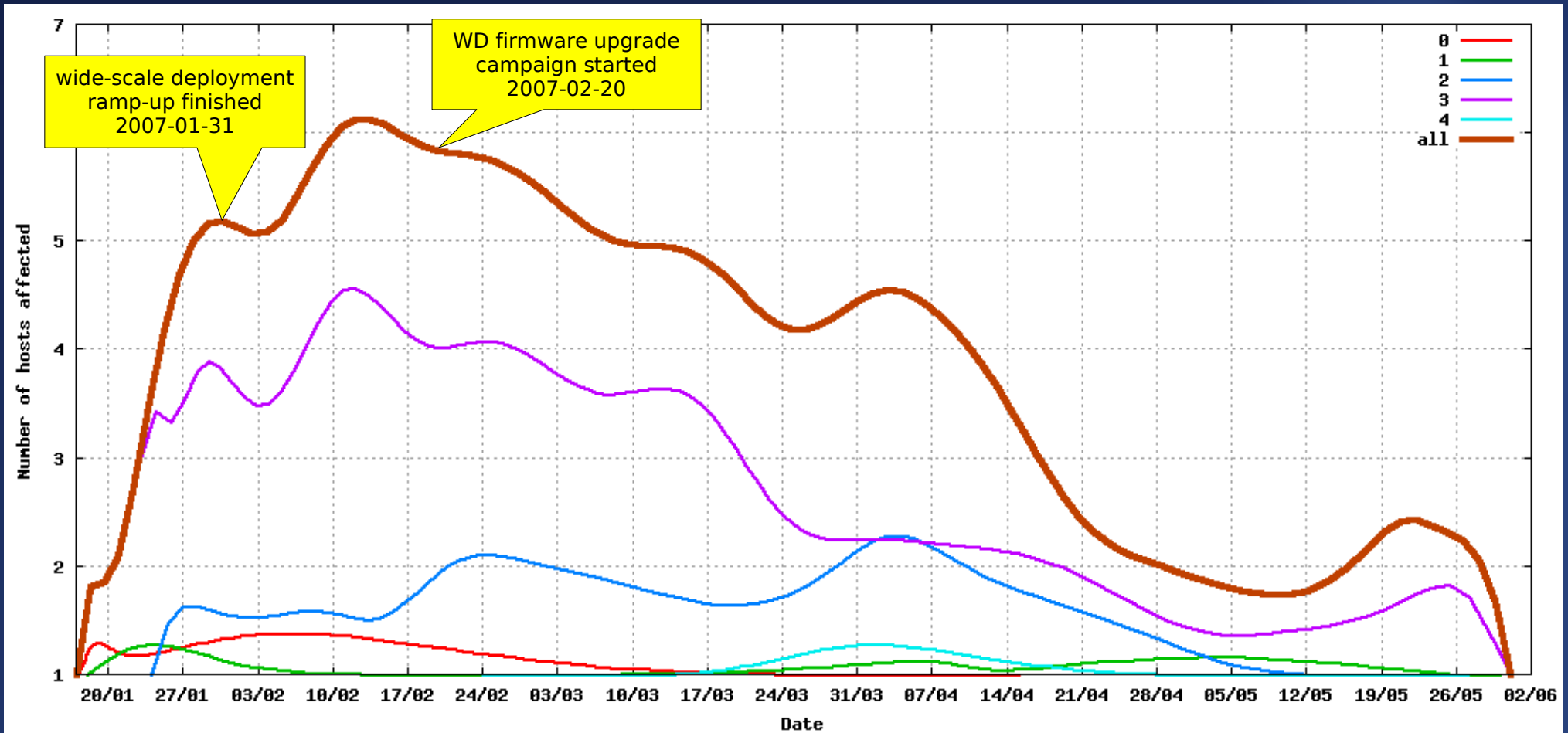
Corruption Types



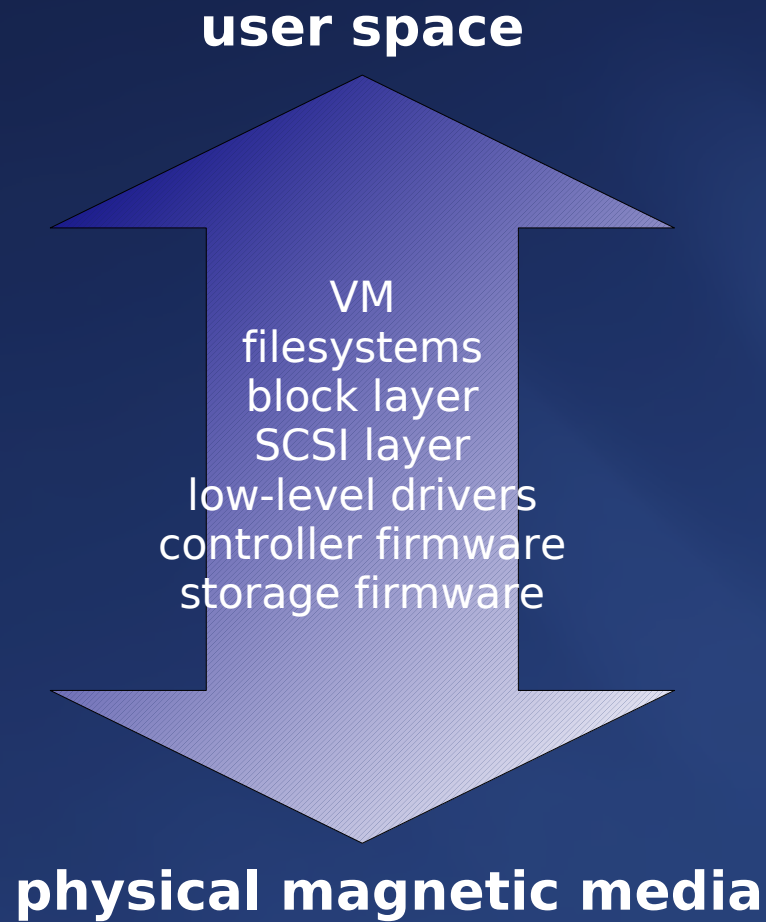
Corruption Persistence



Daily Distribution



Where From?



What Can Be Done?

- self-examining/healing hardware (?)
- WRITE-READ cycles before ACK
- checksumming? → not necessarily enough
- end-to-end checksumming (ZFS has a point)
- store multiple copies
- regular scrubbing of RAID arrays
- “data refresh” re-read cycles on tapes
- ...generally accept and prepare for corruptions



Conclusions

- silent corruptions are a fact of life
- first step towards a solution is detection
- elimination seems impossible
- existing datasets are at the mercy of Murphy
- correction will cost time AND money
- effort has to start now (if not started already)
- multiple cost-schemes exist
 - trade time and storage space (à la Google)
 - trade time and CPU power (correction codes)



Departing Words

“Trust, but verify”

— Ronald Reagan



Questions?

Thank you and have a nice
filesystem (without corruptions)!



Further Reading

- Bernd Panzer-Steindel: Data Integrity v3

<http://indico.cern.ch/getFile.py/access?contribId=3&sessionId=0&resId=1&materialId=paper&confId=13797>

