FUZZING FILE SYSTEM IMPLEMENTATIONS

On BSD based operating systems

~~Lab~~ Lockdown Edition

WHOAMI

- Christopher Krah
  - : @0xricksanchez
  - : 0xricksanchez
  - : ricksanchez
  - : christopher.krah@fkie.fraunhofer.de
- B.Sc - Comp. Sci. – 2017
- M.Sc Comp. Sci. - 2019
- Security Researcher @ Fraunhofer FKIE in Germany – 2019
- Interests: UNIX, IoT, RE, Exploit-Dev. & Fuzzing

OUTLINE

- Whys
- Hows
- Caveats
- Results

WHY…

fuzzing?

*BSD?

file systems?

not use tool X?

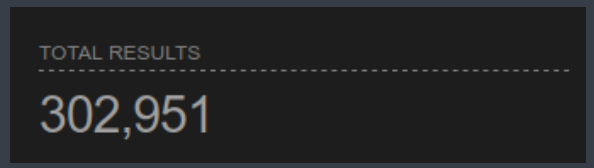
FUZZING

- "Fuzzing renaissance"
- Allows for deeply inspecting a project
- Manual bug hunting not scalable
- Fun

ck@hitb:~/why$

*BSD

NETFLIX



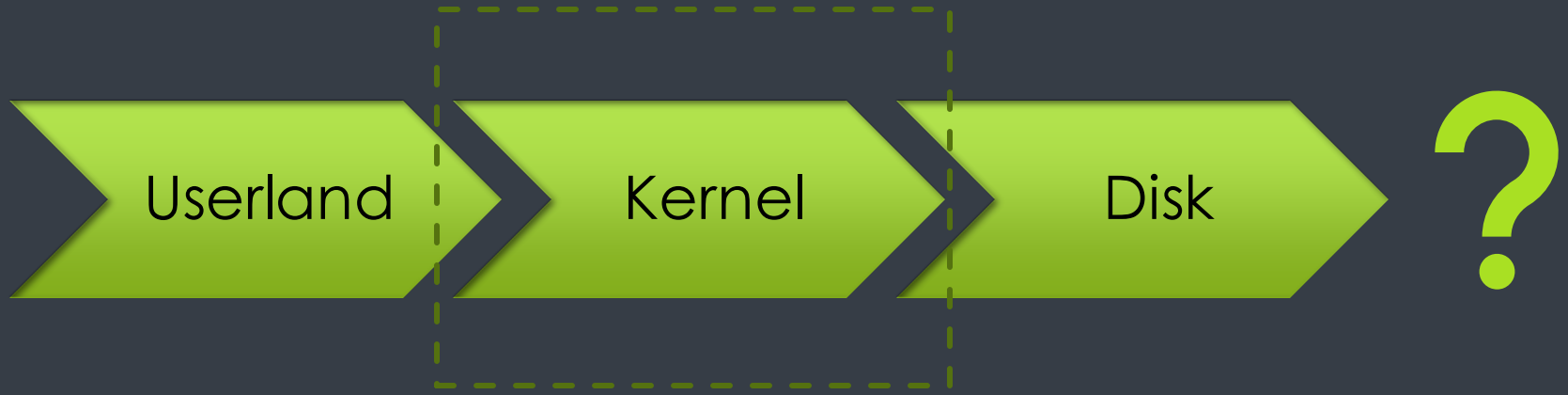
●●●

- Not interested in Windows
- Also everybody does GNU/Linux...

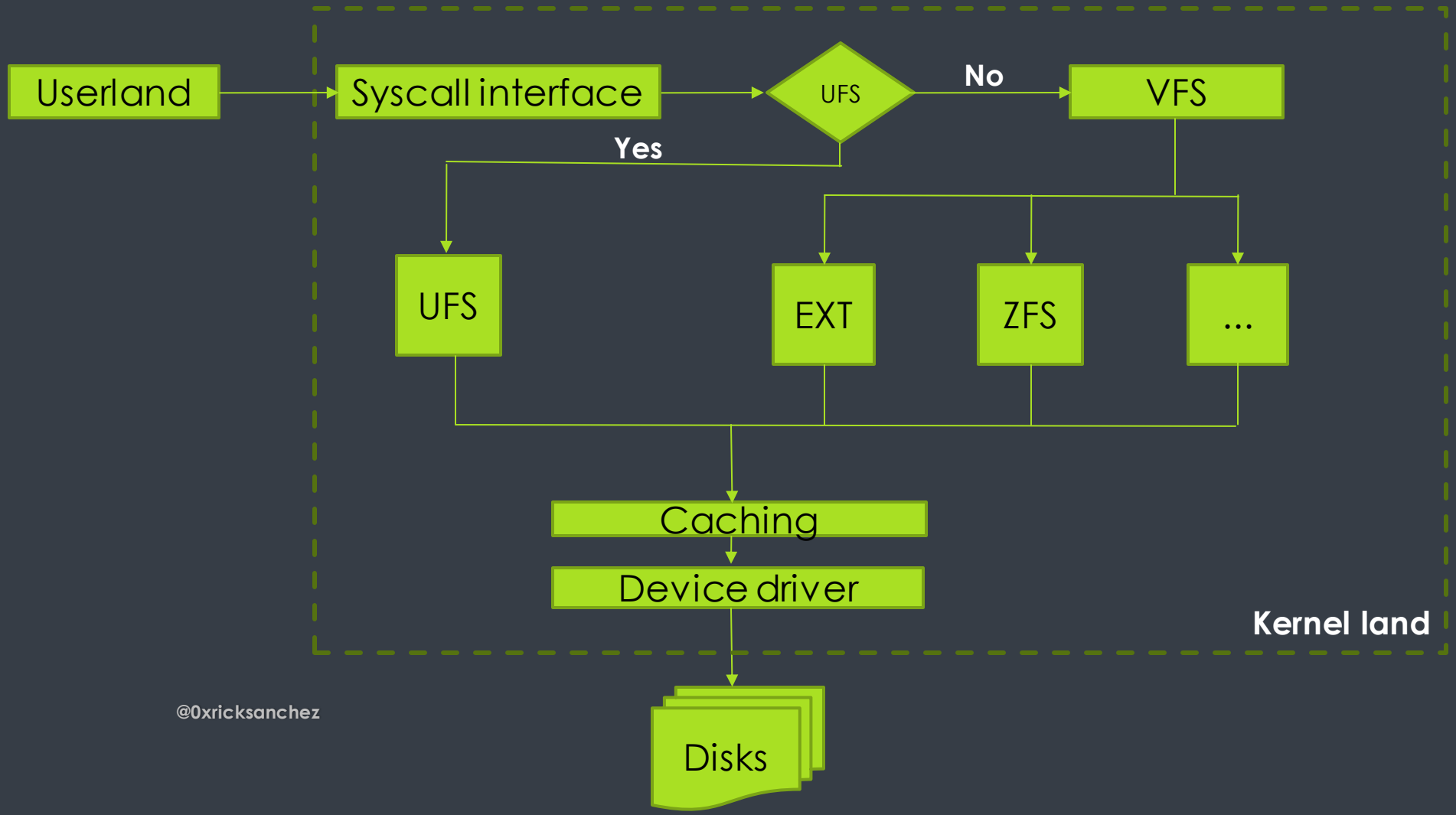
➡ Not all Systems tested equally well

➡ So Why not check out the BSDs!

FILE SYSTEMS?



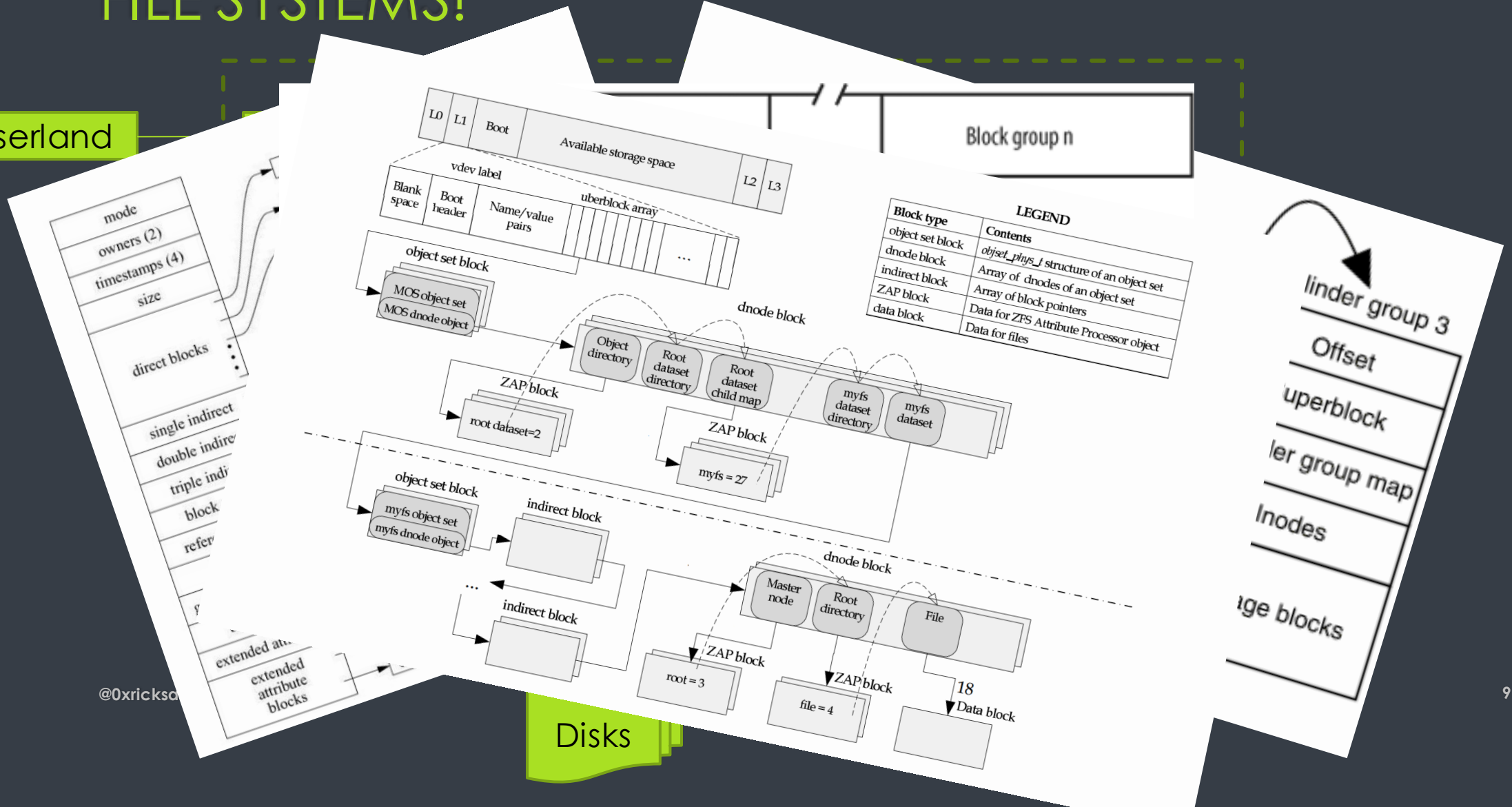
FILE SYSTEMS!



ck@hitb:~/why$

FILE SYSTEMS!

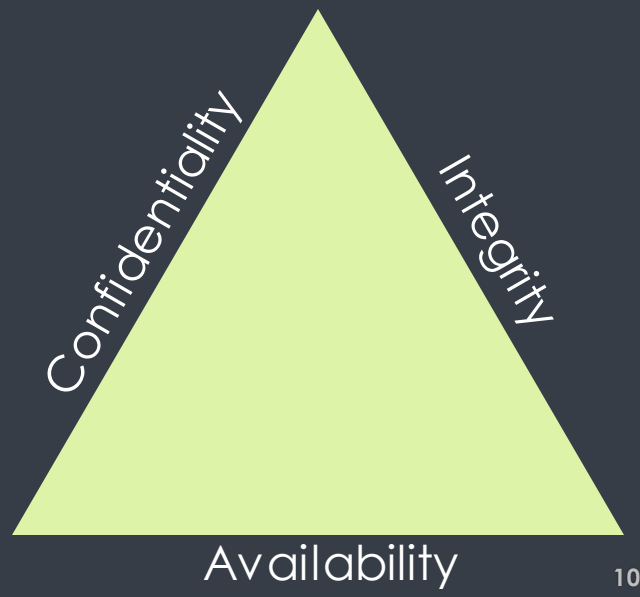
Userland



Disks

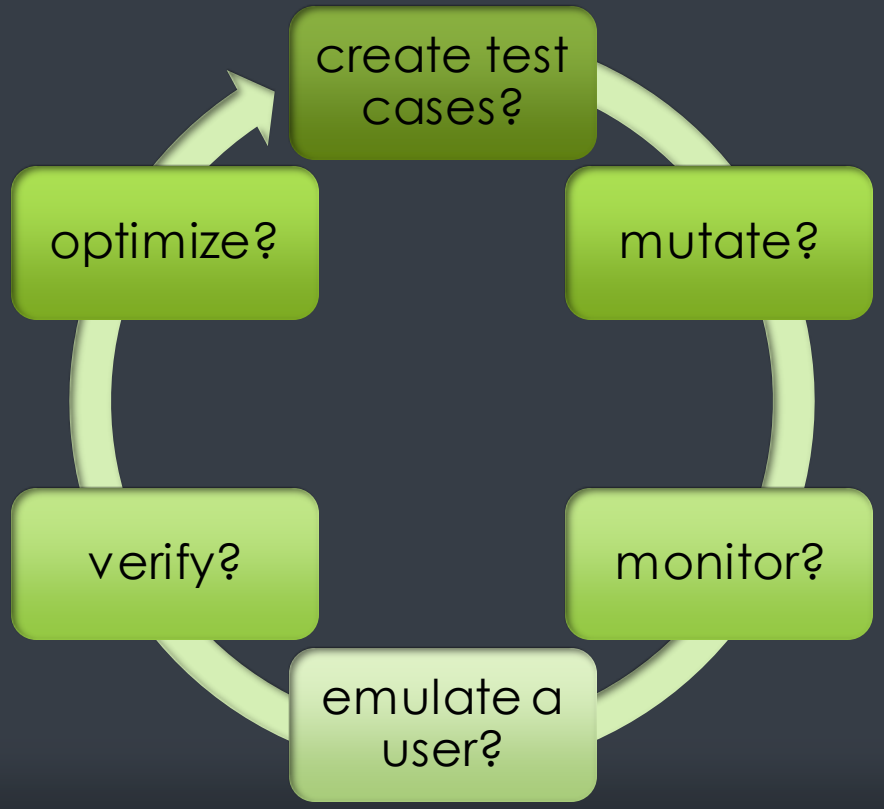
So Why Filesystems afterall?

- Filesystems often overlooked
- However:
  - At least Availablity of data should be ensured/tested for
  - Additionally: Daily Usage of e.g. USB drives
  - Ultimately, filesystems == kernel code execution



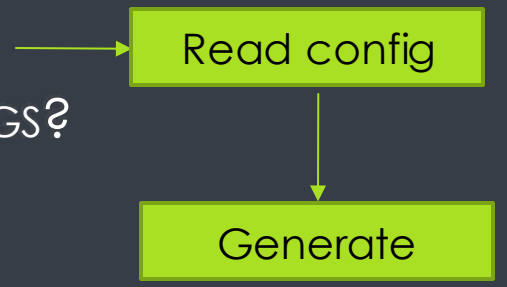
WHY NOT USE 'X' FOR KERNEL FUZZING?

- INTERESTED IN THE COMPLETE EXECUTION CHAIN
  - METADATA PARSING
  - MOUNTING
  - ACCESSING
  - MODIFCATION
  - UNMOUNTING


create test cases?
mutate?
monitor?
emulate a user?
verify?
optimize?

HOW TO…

1. Test case generator



- What's a valid test case when looking for file systems bugs?
  - An actual disk image!
- Automatic generation of
  - (non-) populated FS with variable sizes
  - Currently supported: UFSv1/v2, ZFS, EXT2/3/4, APFS

Observation: Avoid headaches by using the same OS for target and host..

2. Mutation?

- Zero-/FF-out/Randomize superblocks, cylinder groups, single bytes
- Targeted mutations in superblock(s)
- (Deterministic) Full binary mutation via radamsa

☞ *Observation*: 'Dumb mutations' often enough*

2. Mutation?

FS	#good_mounts	#bad_mounts
UFS	~ 20%	~ 80%
EXT	~ 80%	~ 20%
ZFS	~ 7.5%	~ 92.5%

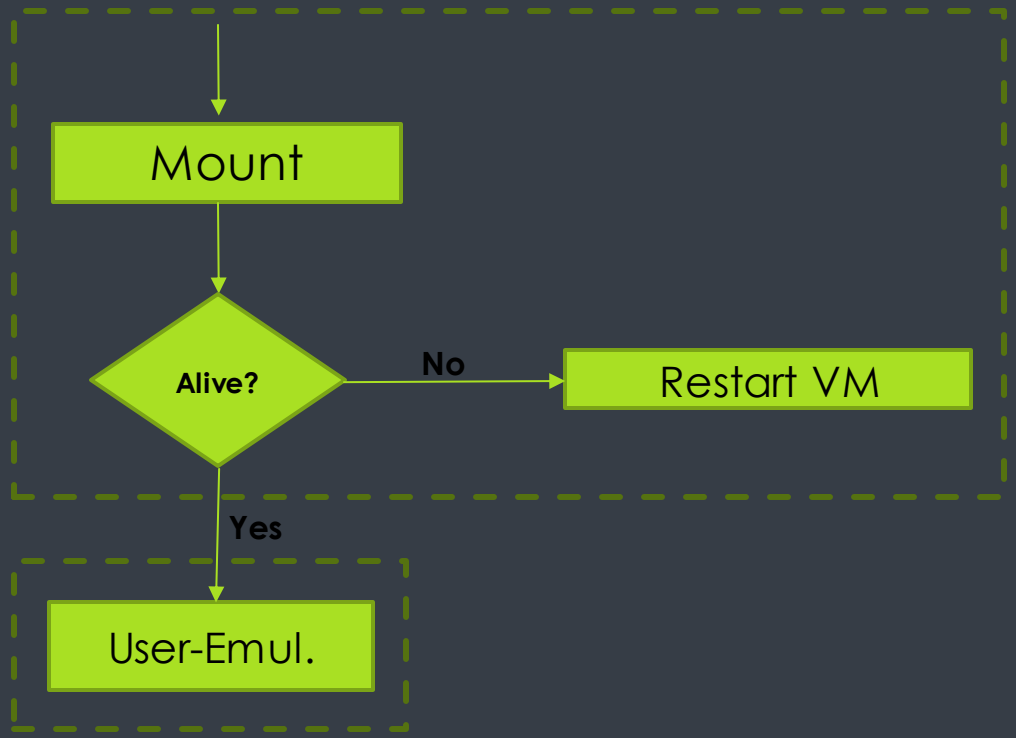
Scenario	Result
Pool not recognized	Not importable
Pool metadata corrupted	Not importable
One or more devices contain corrupted data	Not importable
Valid pool	Importable

2. Mutation?

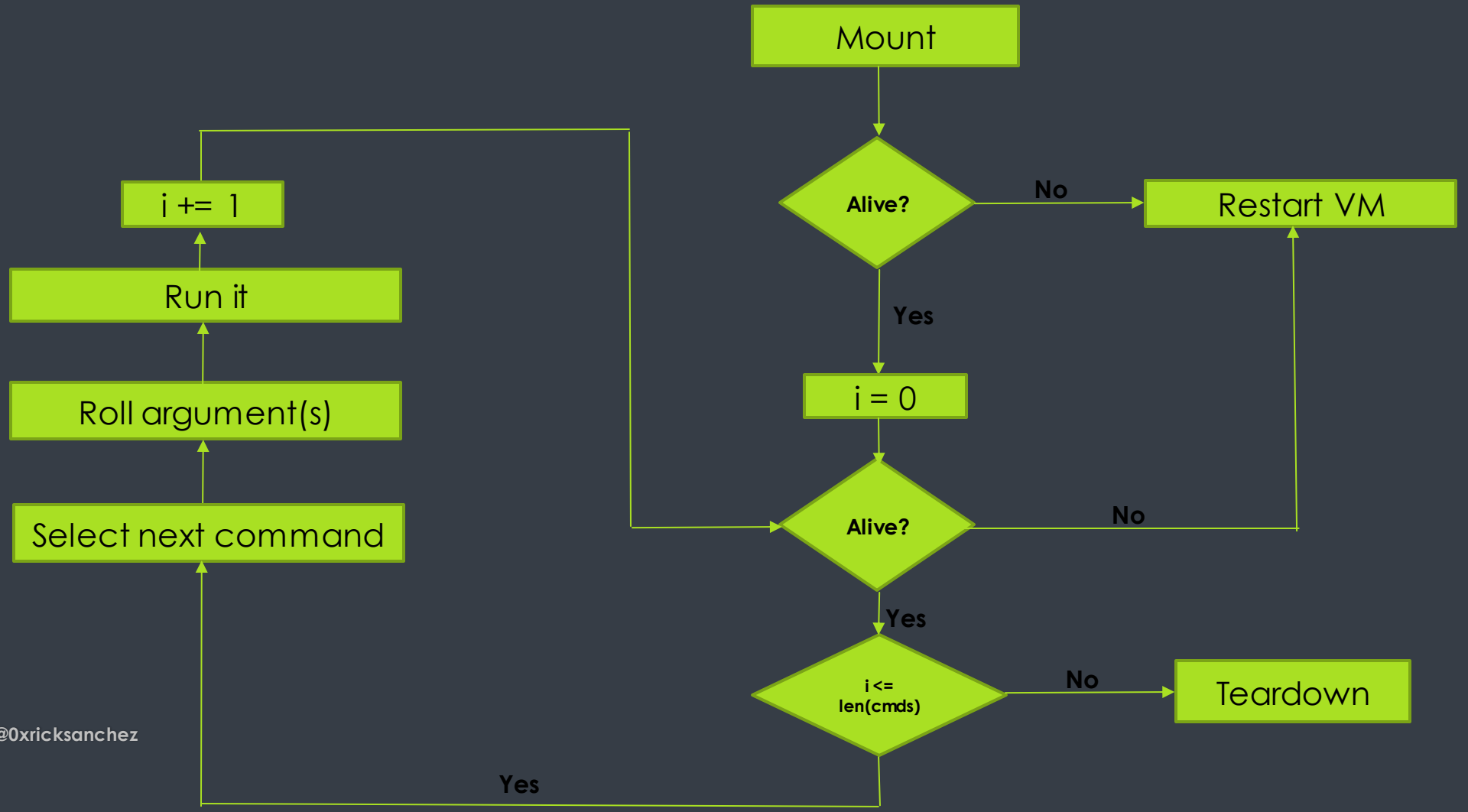
FS	#good_mounts	#bad_mounts
UFS	~ 20%	~ 80%
EXT	~ 80%	~ 20%
ZFS	~ 7.5%	~ 92.5%

- *Observation*: Inverse correlation between UFS and EXT
- *Observation*: Integrity checks of ZFS

3. User Emulation



3. User Emulation



3. User Emulation

Category	Operation
changing geometry	chflags, chgrp, chmod, chown, mv, rm, truncate*
extending geometry	cp, dd, echo, ln, mkdir, mknod, split*, touch
parsing geometry	basename*, chdir, dirname*, du*, file, find, getfacl*, ls, readlink, stat, wc*

- Static vs. Randomized order
- Static vs. Randomized Arguments

3. User Emulation

FS	Static User-Emulation		Random User-Emulation	
	#good	#bad	#good	#bad
UFS	~ 27.5%	~ 72.5%	~ 45%	~ 55%
EXT	~ 20%	~ 80%	~ 40%	~ 60%
ZFS	~ 98 %	~ 2%	~ 98 %	~ 2%

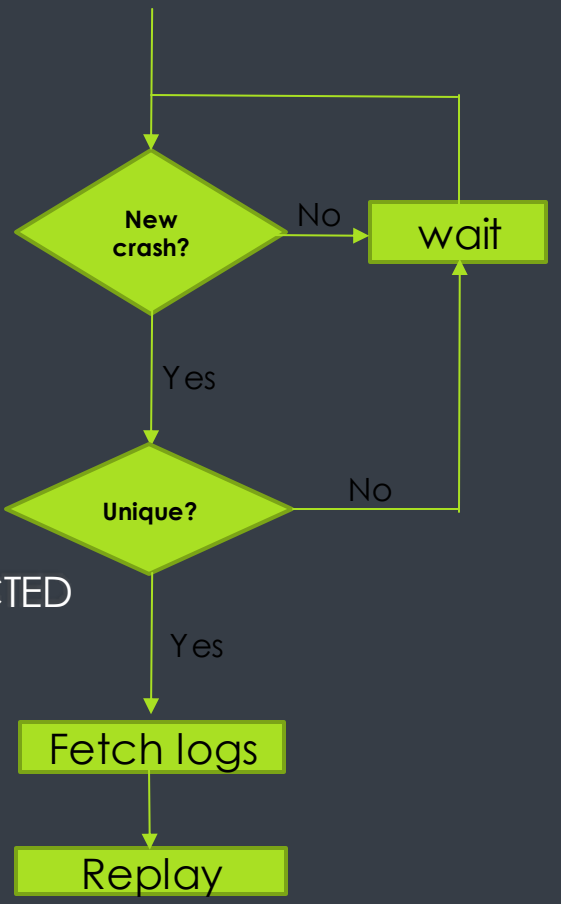
- *Observation*: RNG matters!
- *Observation*: Crashes happen during Mount, User-Emul. & Teardown!

4. MONITORING

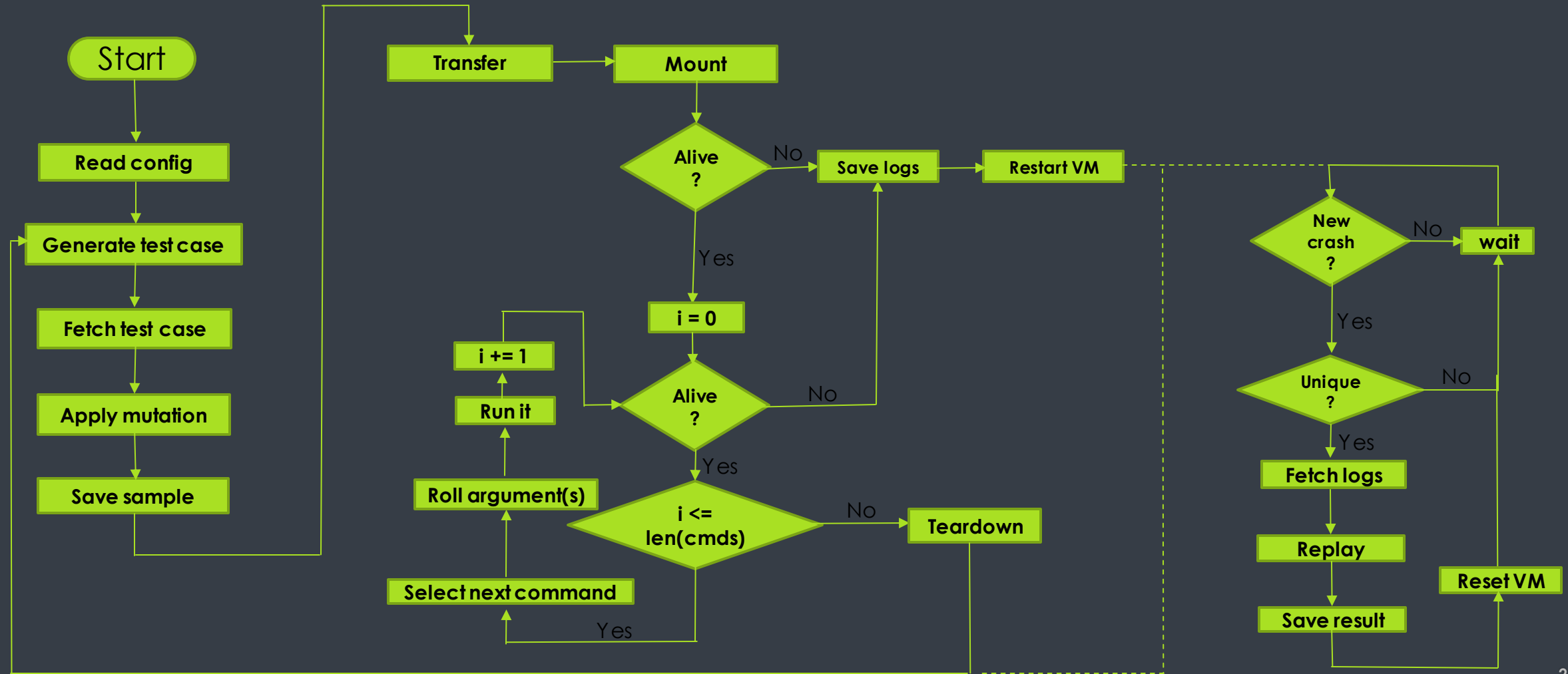
- Permanent alive checks for fuzzers
- Tracking of samples, mutations, seeds, crashes
- Logging of FS structure
- Logging of user emulation

5. VERIFICATION

- AUTOMATIC CONTINUOUS CHECKS FOR NEW CRASHES
  - BASED ON HASH IDENTIFIER
- REPLAY ON SEPARATE, 'ALWAYS FRESH' INSTANCE
- *OBSERVATION* : OS SIDE EFFECTS NOT AS BAD AS EXPECTED



PUTTING THINGS TOGETHER



BUT CAN IT PRODUCE CRASHES?

ck@hitb:~/results$

PUTTING THINGS TOGETHER

```
Start date: 2019-02-21 21:48:31.16 | Runtime: 11 days, 22:40:44.85 | Iteration: 81471 | Last iteration time: 6.51s
Avg. iteration time: 12.52s | #Crashes: 5092 | #New crashes: 32 | Last panic: ufs_dirbad
Last new crash (iter): 77373 | Filesystem type: ufs2 | Filesystem size: 25MB
Successful mounts: 17228 (21.15%) | 106221/267446 (39.72%) Commands executed
```

```
[+] VM status: OK
[+] Mounting successful!
>> Accessing & modifying mounted filesystem: /mnt/radamsa_fuzz1_ufs2_25MB
[*] Successfully completed 15/22 program calls
[+] Unmounted /mnt/radamsa_fuzz1_ufs2_25MB successfully
```

ck@hitb:~/results$

Findings

- >100 unique crashes in UFS/EXT
  - Multiple OOB-R/OOB-W
  - Triple Fault in UFS
  - Double Fault in EXT
  - Bonus: Non-Deterministic crash in UFS with 6 unique core dumps so far
- Overall >82% reproducibility rate
  - Additionally another 5% produced a different crash on verification
- 17 syscalls covered

SYSCALLS

- 26 USERLAND PROGRAMS
- 17 SYSTEM CALLS
  - 2 NEW VIA RANDOMIZING
  - 3 NEW VIA EXTENDED EMULATION

Category	Operation
changing geometry	chflags, chgrp, chmod, chown, mv, rm, truncate*
extending geometry	cp, dd, echo, ln, mkdir, mknod, split*, touch
parsing geometry	basename*, chdir, dirname*, du*, file, find, getfacl*, ls, readlink, stat, wc*

SYSCALLS

- 17 SYSTEM CALLS
  - 2 NEW VIA RANDOMIZING
  - 3 NEW VIA EXTENDED EMULATION

☞ OBSERVATION: RNG MATTERS!

☞ OBSERVATION: FINE TUNING MATTERS!

RESP.DISCLOSURE - FREEBSD

- ~50/>100 DISCLOSED VIA RESPONSIBLE DISCLOSURE
- A BUNCH OF MAILS LATER:
  - 21 CONFIRMED BUG TRACKER NUMBERS
  - 10 CONFIRMED FIXES
  - HOWEVER, NO FEEDBACK/REPLIES FOR MONTHS NOW.. 🤔

Resp.Disclosure - Net-/OpenBSD

- Short evaluation in both of these show similar results (FFS/UFS).
  - NetBSD: "not interested"
    - #Fixes: 0 🤷
  - OpenBSD: "FFS/UFS filesystem has made these design decisions, kernel has no logic to handle inconsistencies, …"
    - #Fixes: 0 🤷

CAVEATS

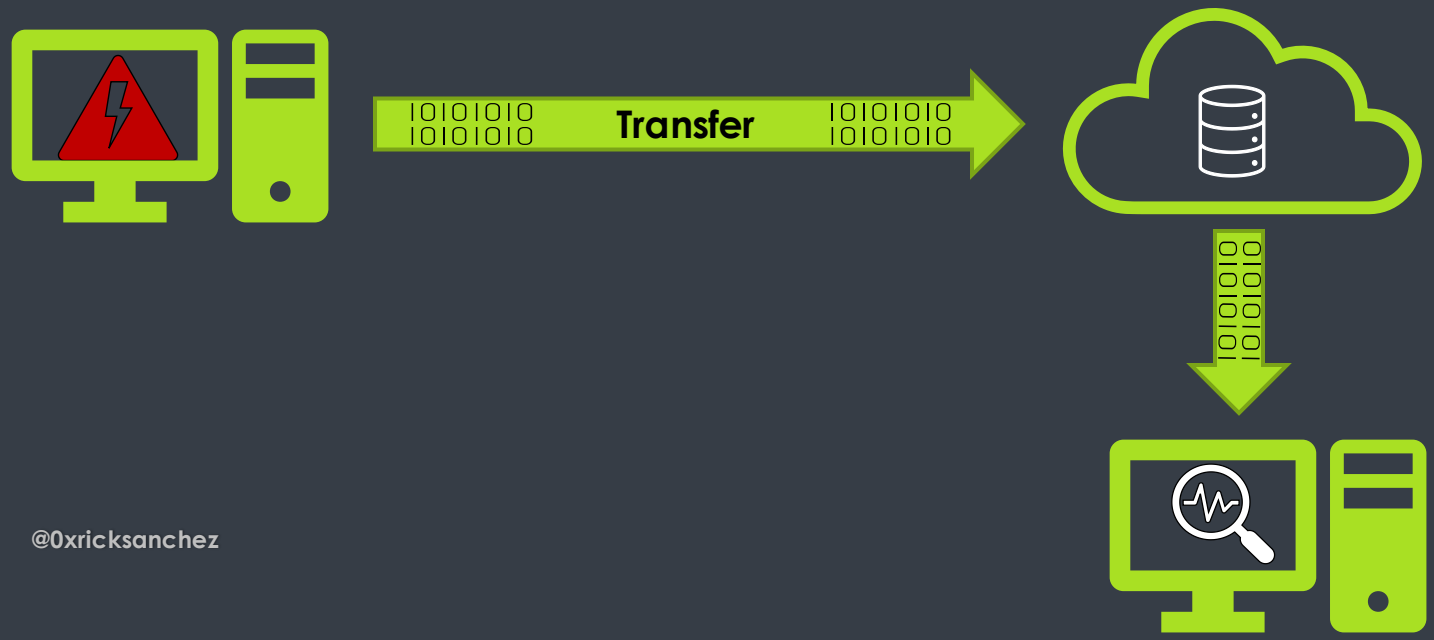
BOOT TIMES

-> % CAT RESULT.TXT

	FAT	FAT_DIAG	SMALL_DIAG	SMALL_DIAG_BOOT_DELAY
RUN 1:	39.23S	41.97S (+7.0%)	37.79S (-10.0%)	26.10S (-38.2%)
RUN 2:	39.77S	40.80S (+2.6%)	36.24S (-12.1%)	27.68S (-32.2%)
RUN 3:	38.15S	40.79S (+6.9%)	37.26S (-8.7%)	27.11S (-34.5%)
RUN 4:	39.12S	38.82S (-0.1%)	36.73S (-5.4%)	26.01S (-33.0%)
RUN 5:	39.76S	41.45S (+4.3%)	36.71S (-11.5%)	25.58S (-38.3%)
AVG:	39.21S	40.77S (+4%)	36.95S (-9.4%)	26,50S (-35.0%)

~~BOOT TIMES~~ MAN NETDUMP

- NETDUMP - PROTOCOL FOR TRANSMITTING KERNEL DUMPS TO A REMOTE SERVER
  - WOULD ELIMINATE NEED TO REBOOT TO FETCH CORE DETAILS
  - HOWEVER: UNRELIABLE IN MY SETUP



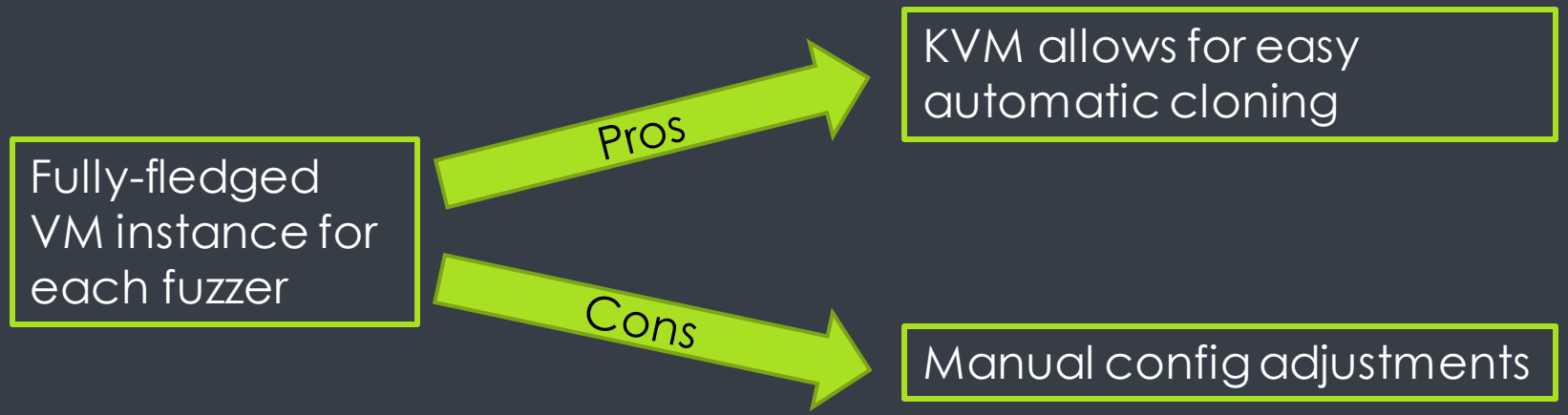
~~BOOT TIMES~~ LIBOS

- INSTEAD OF FULL FLETCHED KVM VM WITH '*OPTIMIZED*' KERNEL
  - ONLY PLUG NECESSARY PARTS TOGETHER…

SMART(ER) MU:A]1ON 

- RIGHT NOW:
  - NO RESTORING OR RE-CALCULATION OF CHECKSUMS/INTEGRITY CHECKS
    - IMPORTANT FOR EXT4, ZFS
  - KERNEL FEEDBACK, KASAN
  - AUTOMATIC DEDUCTION OF METADATA FIELD TYPES/SIZE

SCALABILITY?



EOF

Conclusion

- Write your own fuzzing tools!
  - Kernels still offer lots of bugs that wait to be uncovered
  - Modern FS implementations will need some more considerations
- Responsible Disclosure sometimes frustrating
- File systems allow for deep introspection of userland to kernel land behavior

ck@hitb:~/EOF$

Fin.

- Questions/Suggesstions? Please reach out!
  - : https://twitter.com/0xricksanchez
  - : https://github.com/0xricksanchez
  - : https://0x00sec.org/u/ricksanchez
  - : christopher.krah@fkie.fraunhofer.de
- Slides/Scripts?
  - Will be here: https://github.com/0xricksanchez/fs-fuzzer