



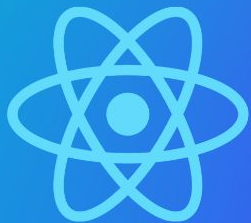
How Canva runs millions of Jest tests a day

Johnny Cheng

Canva's frontend stack



×



Canva

Our Frontend Testing stack



Integration tests



Visual regression tests



Unit tests

Volume of unit tests



18,000 targets!



Target count

Why is running unit
tests hard?



First-party code is small

Canva Product Code



Canva Test Tooling Code



~ 60 files per test target

node_modules crash course



×



≈

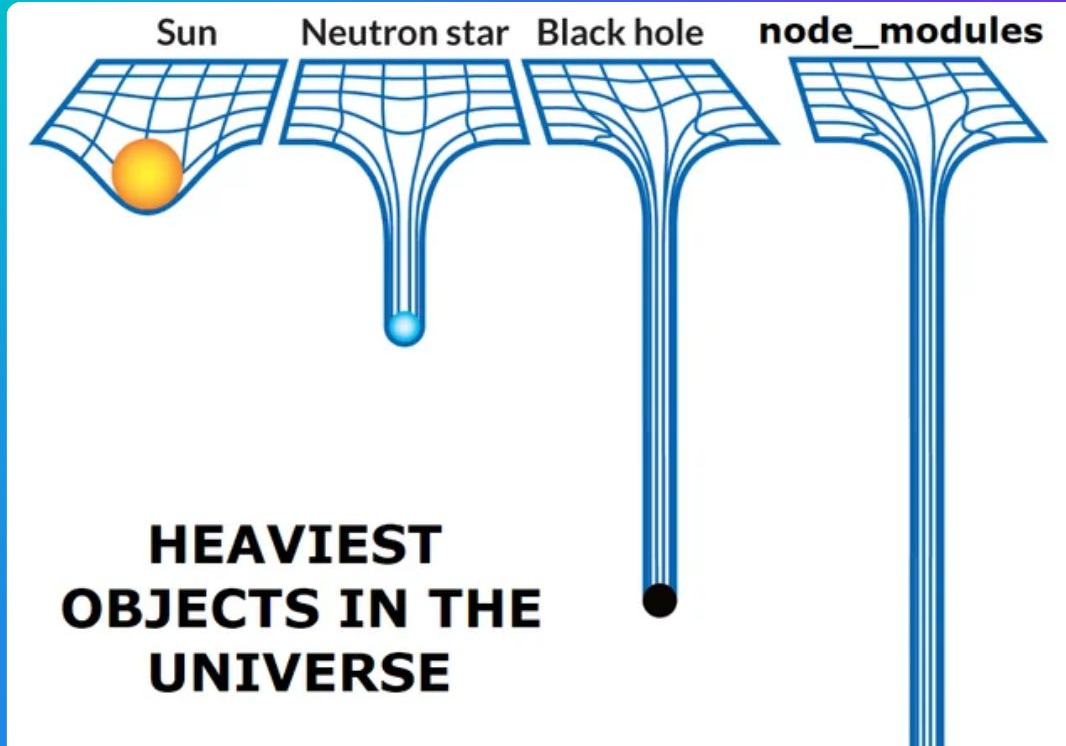


×



```
1  node_modules/
2  └─ @types/
3     └─ react/
4         └─ index.d.ts
5         └─ package.json
6  └─ jest/
7     └─ bin/
8         └─ jest.js
9     └─ build/
10        └─ package.json
11 └─ react/
12    └─ cjs/
13    └─ index.js
14    └─ package.json
15 └─ scheduler/
16    └─ index.js
17    └─ package.json
18 ... 10,000+ more files!
19
```

node_modules is huge!



Test setup outlasts the test

Test Setup - 15s

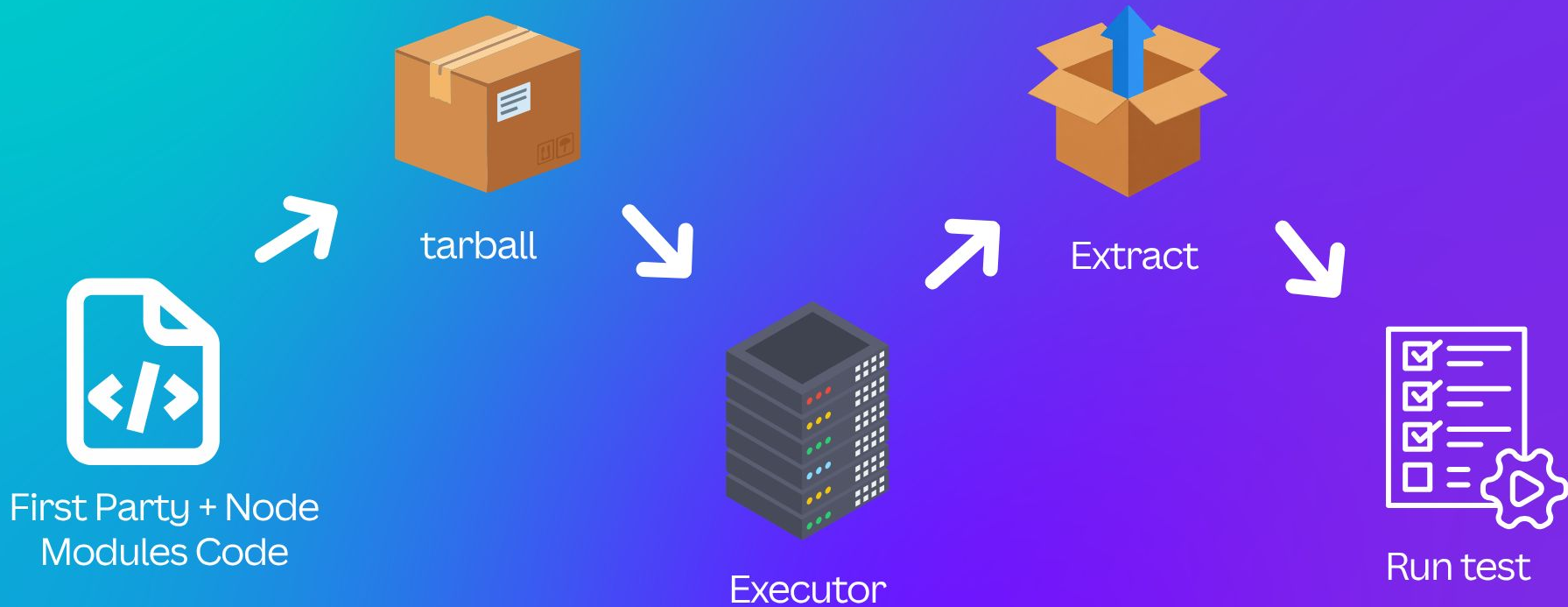
Test execution - 12s



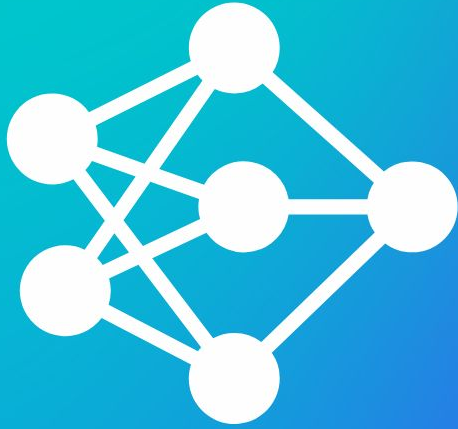
Across millions of targets, everyday!



v1 - Tar every input



v1 - Learnings

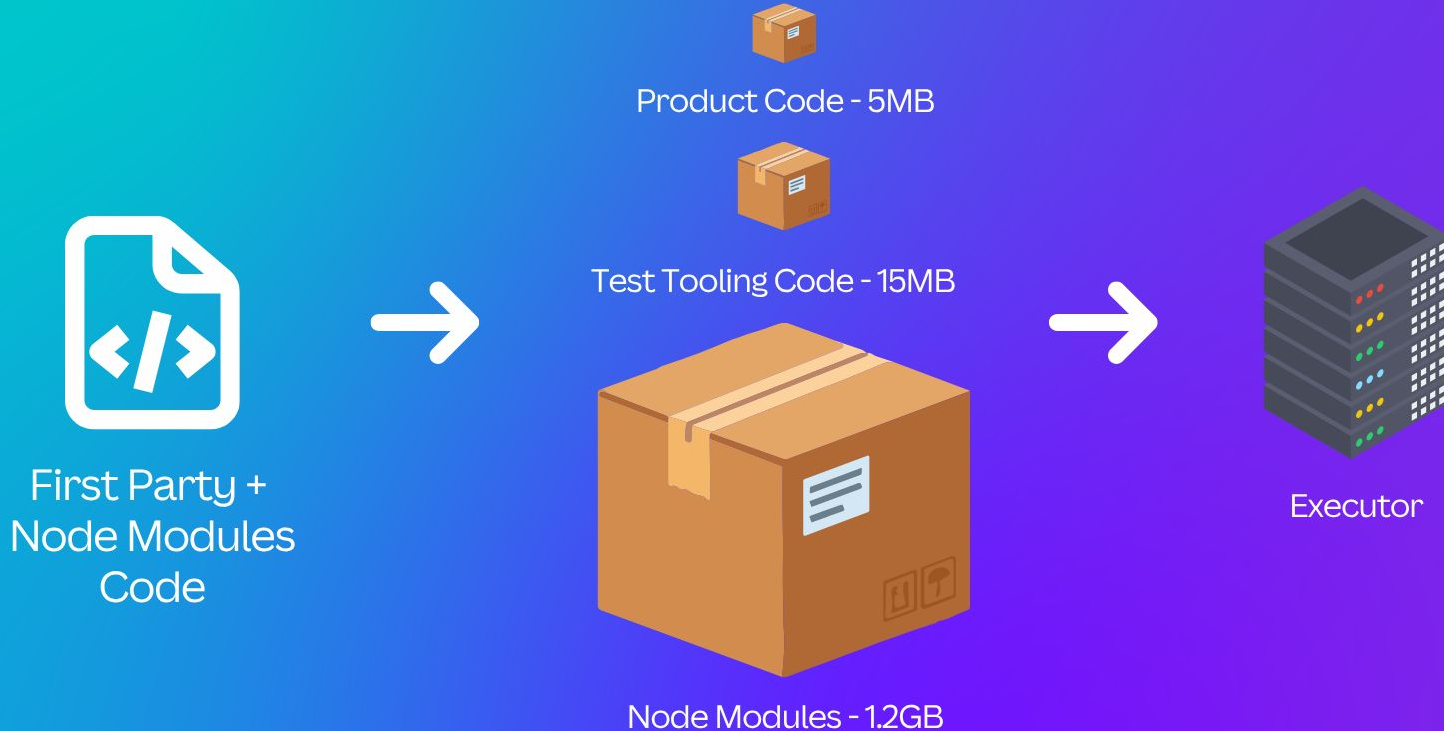


Win for network calls!



Moved the problem to IOPS

v1.5 - Split inputs into layers



v1.5 - Learnings

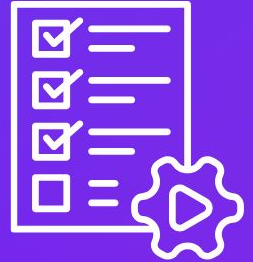


Faster tarball builds!



Extraction still slow

v2 - Mount SquashFS image



Build Node Modules

Executor

Mount SquashFS
with FUSE

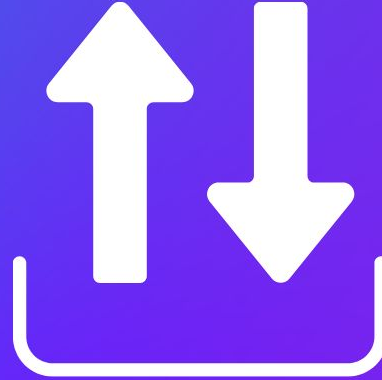
Run test

Instant!

v2 - Wins!



Test preparation optimised!



IOPS no longer an issue!

Are we done?



v2 - Learnings

Test Setup

Test execution



All the gains we made in test preparation were erased!



v2.5 - EROFS instead of Squashfs

Same random reads - Enhanced Read Only File System (EROFS)
uses **6x** less I/O than SquashFS! [1]

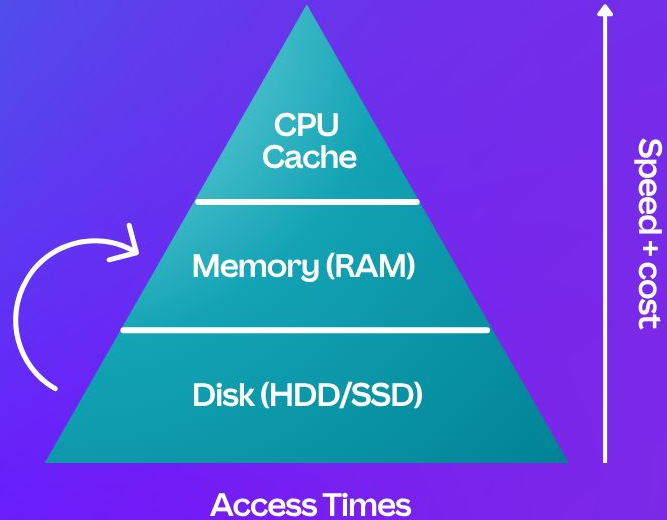


Less I/O → faster reads!



v3 - EROFS mounted in memory with FUSE

Instead of mounting in disk...
mount in memory!

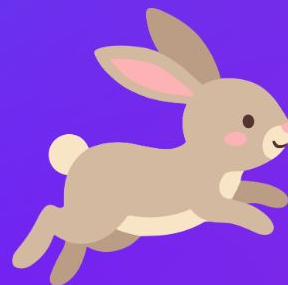
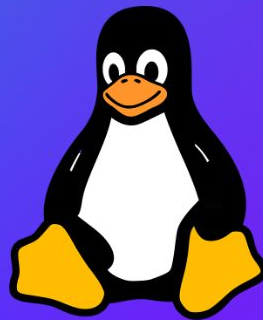


Next steps - direct kernel mount

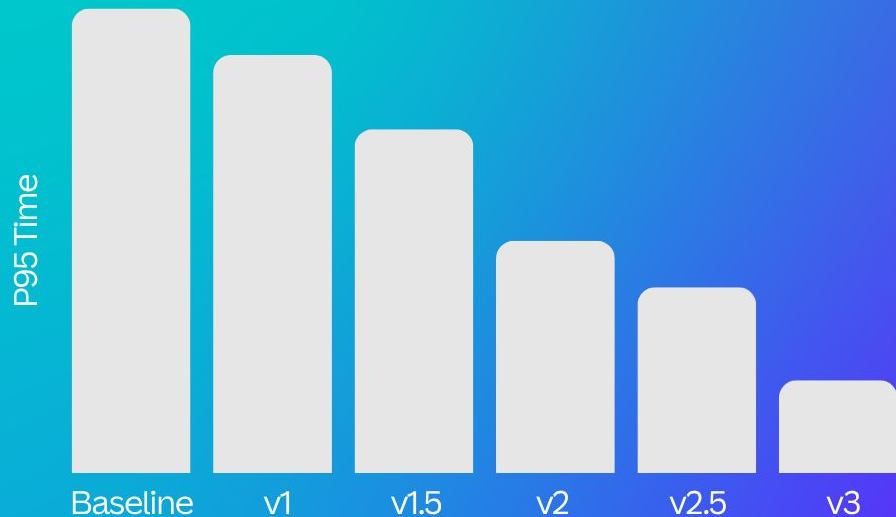
FUSE mount



Kernel mount



Results at Scale



P95 duration fell 80%!

Variance plummeted!



Thank you!

