Missive: Fast Application Launch from an Untrusted Buffer Cache

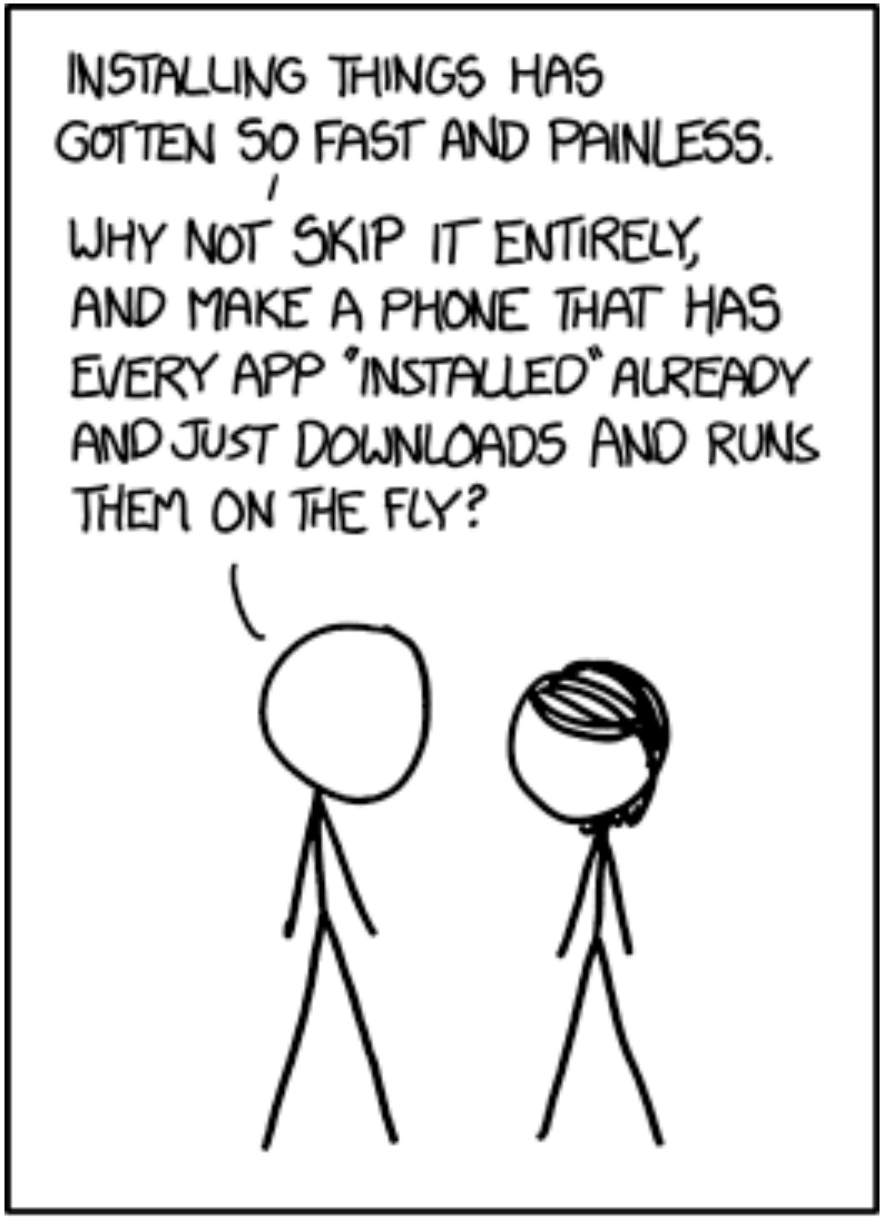
Jon Howell

Jeremy Elson

Bryan Parno

John R. Douceur

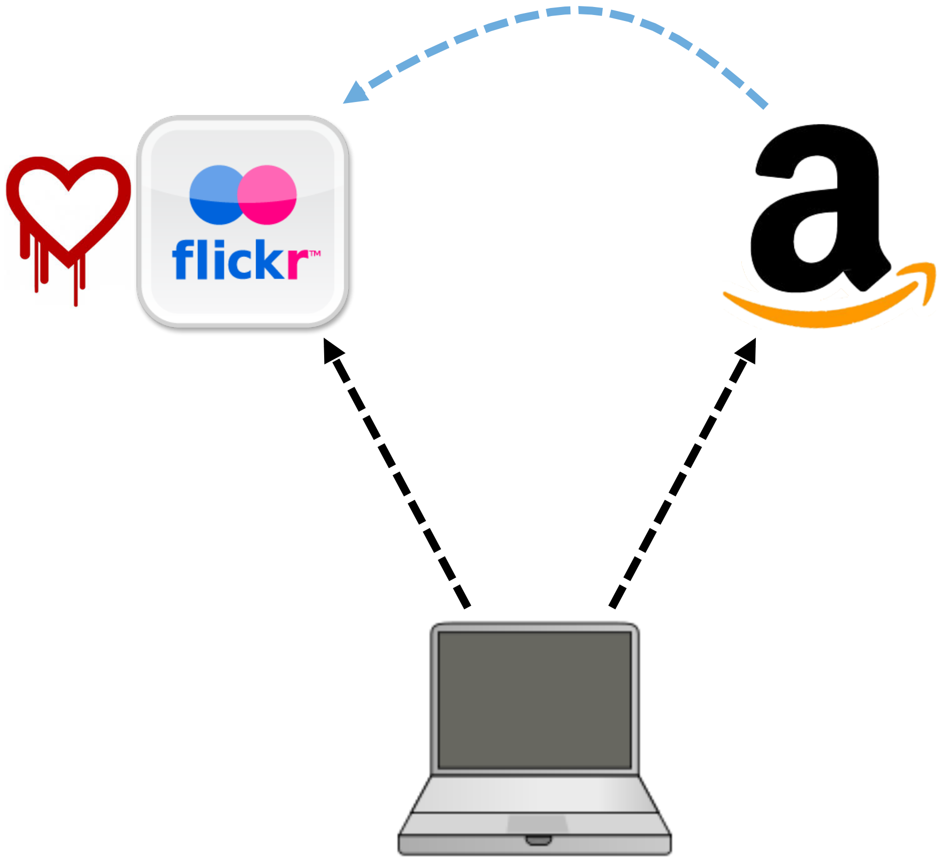
Microsoft Research


INSTALLING THINGS HAS GOTTEN SO FAST AND PAINLESS.
WHY NOT SKIP IT ENTIRELY, AND MAKE A PHONE THAT HAS EVERY APP "INSTALLED" ALREADY AND JUST DOWNLOADS AND RUNS THEM ON THE FLY?

I FELT PRETTY CLEVER UNTIL I REALIZED I'D INVENTED WEBPAGES.

http://xkcd.com/1367/

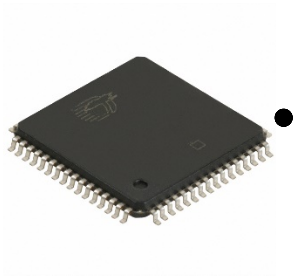
Autonomy leads to painlessness



Client installations are not autonomous

Embassies: autonomy on the client

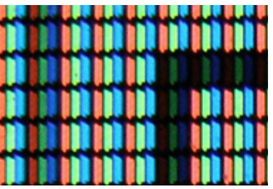
Minimality leads to autonomy



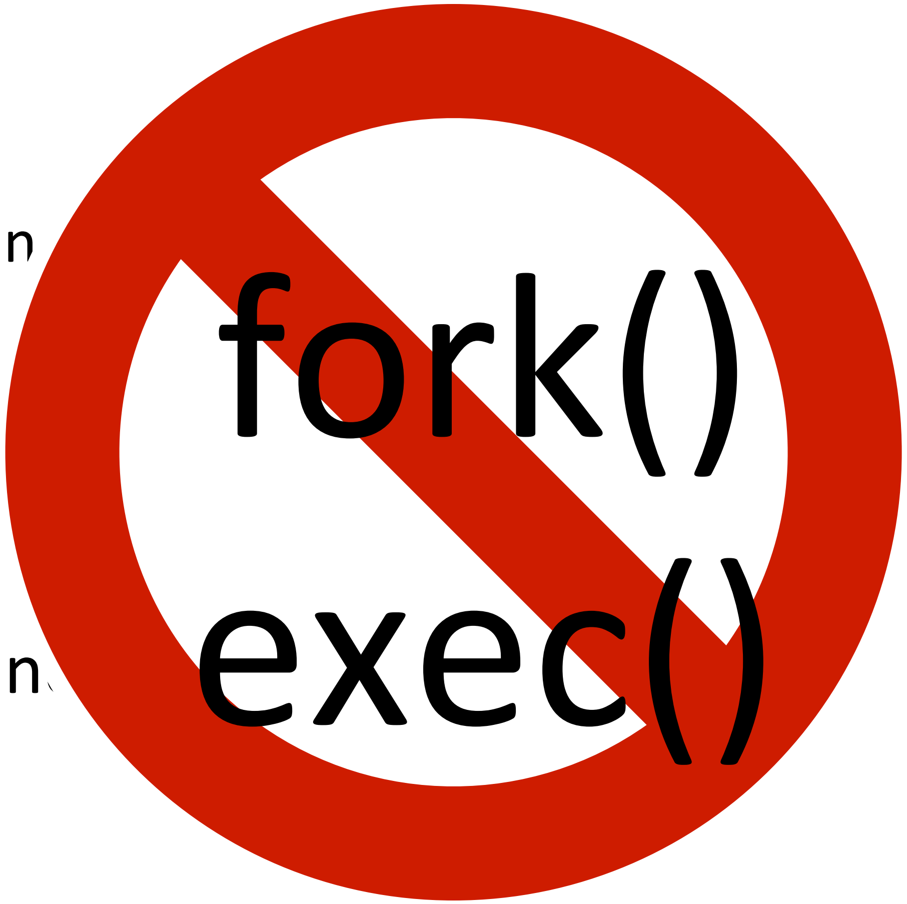
- picoprocess:
  - memory allocation in
    and scheduling

- all communications

- crypto primitives (ran

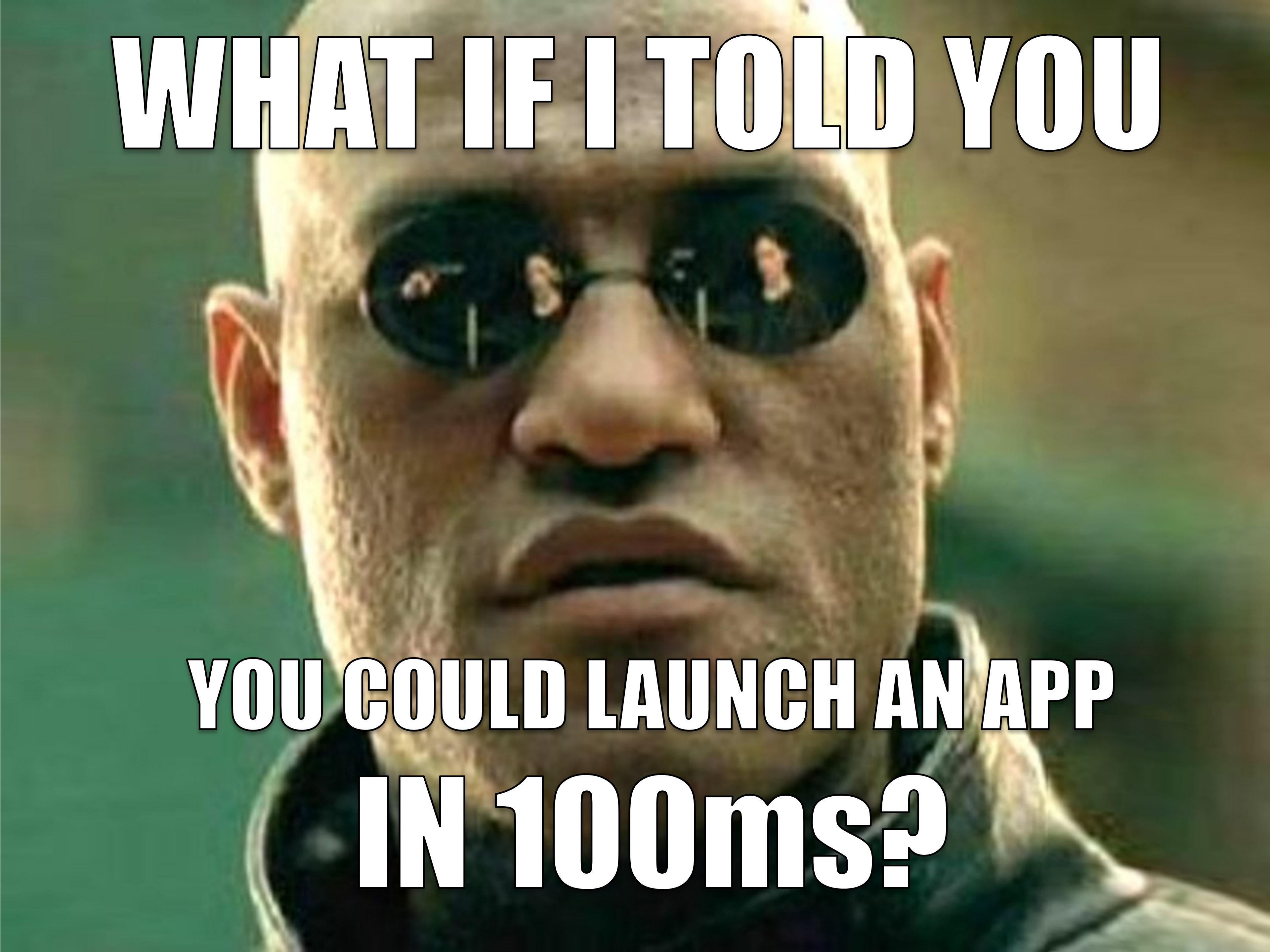


- UI is pixel blitting



Consequences of minimality

- Apps are big.
  - browser
  - rendering libraries
  - OS, filesystem
- No shared, trusted buffer cache


WHAT IF I TOLD YOU
YOU COULD LAUNCH AN APP
IN 100ms?

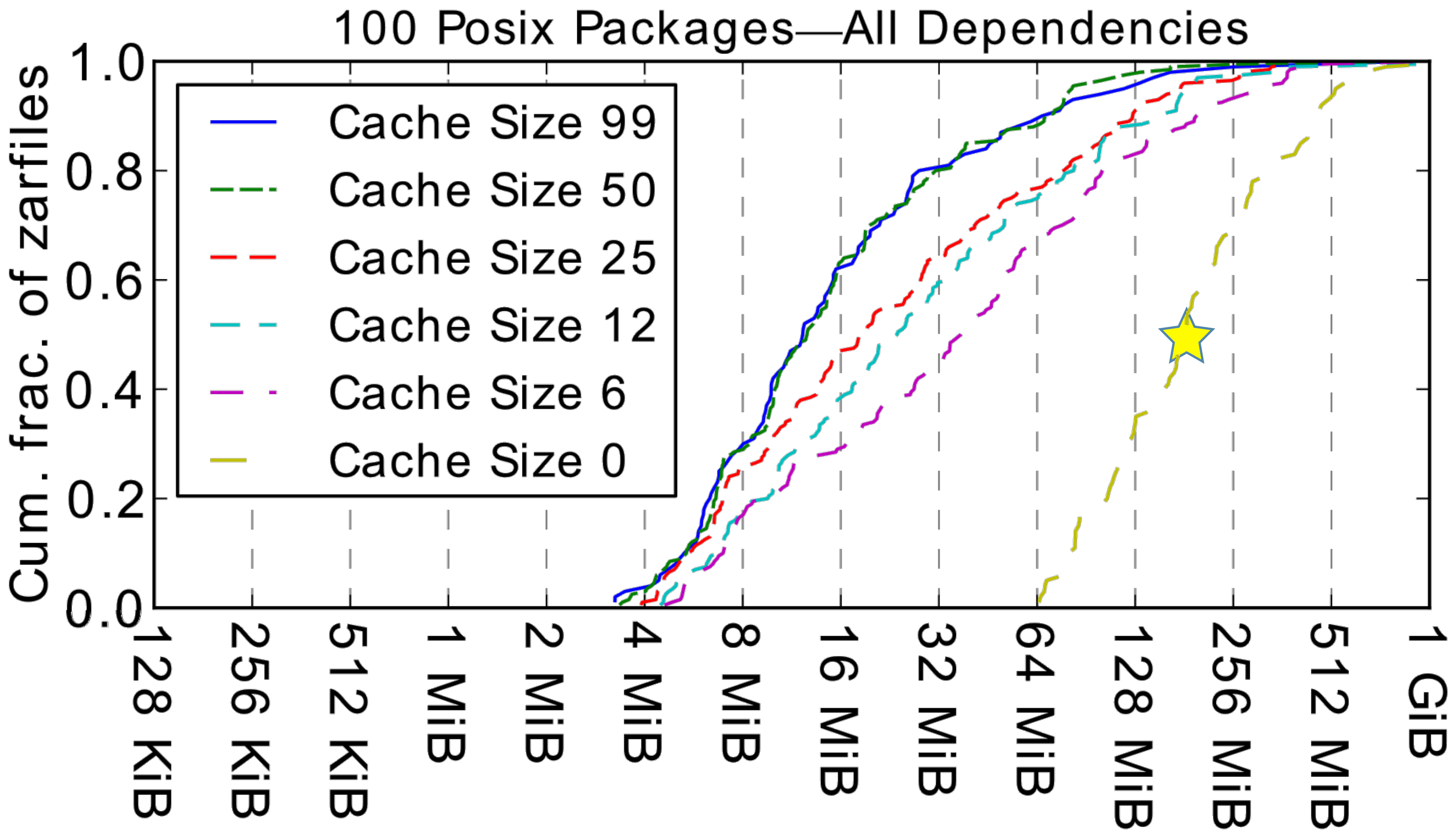
Why will it work?

- Commonality is available
- We can exploit it
    - to reduce network costs
    - to minimize local startup latency
- Proof of concept: cost is around ~100ms

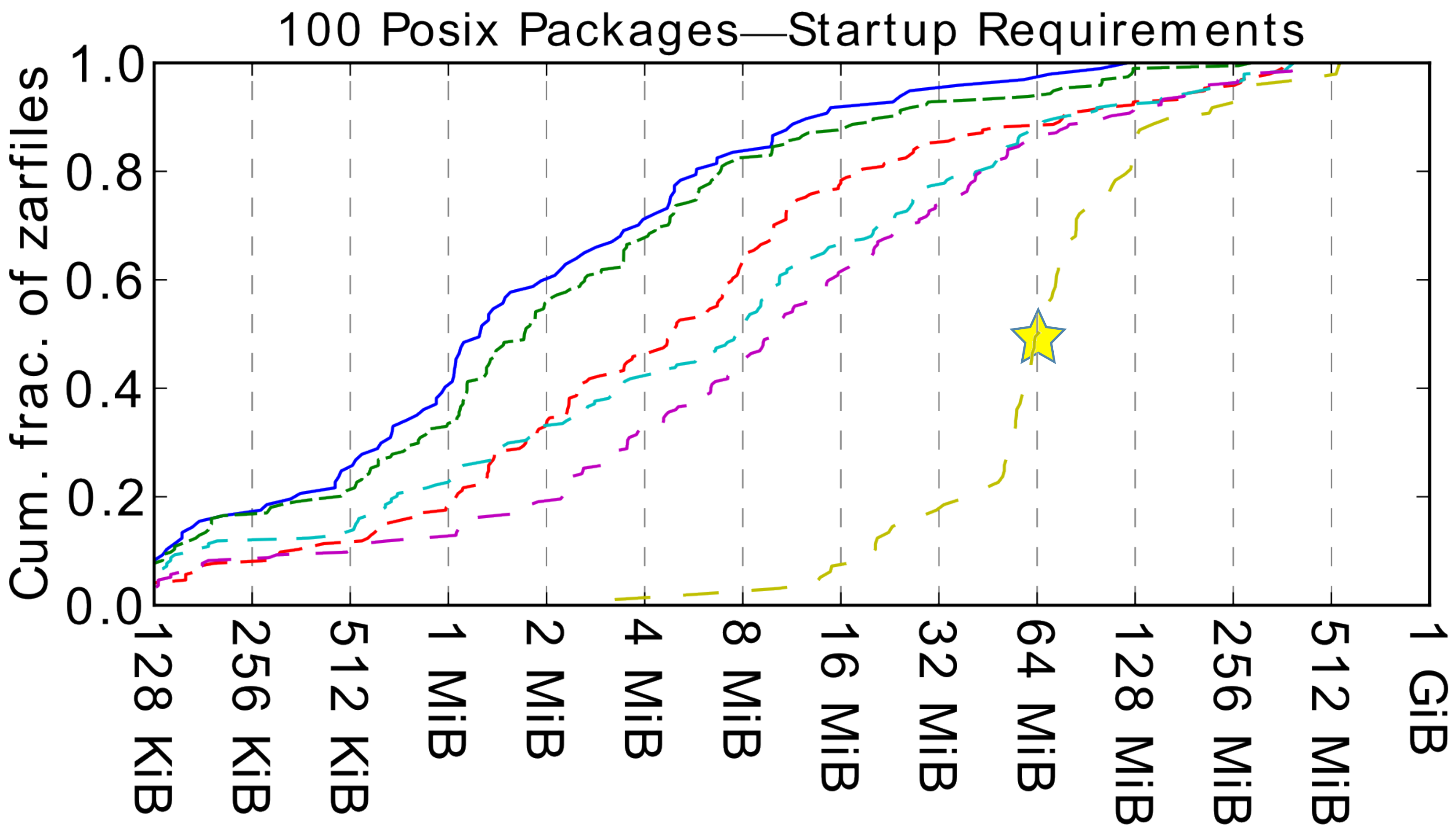
100MiB apps have commonality

- Servers *could* run anything…
  but a few programs serve each function
    - OpenSSL, PolarSSL, Windows SSL
    - postfix, qmail, exim
- Embassies clients *could* run anything…
    - 100 “best-of” interactive desktop apps

100MiB apps have commonality



100MiB apps have commonality



Fast app launch

- Exploiting commonality to save network bandwidth
- Exploiting local commonality at low latency

App launch in Embassies

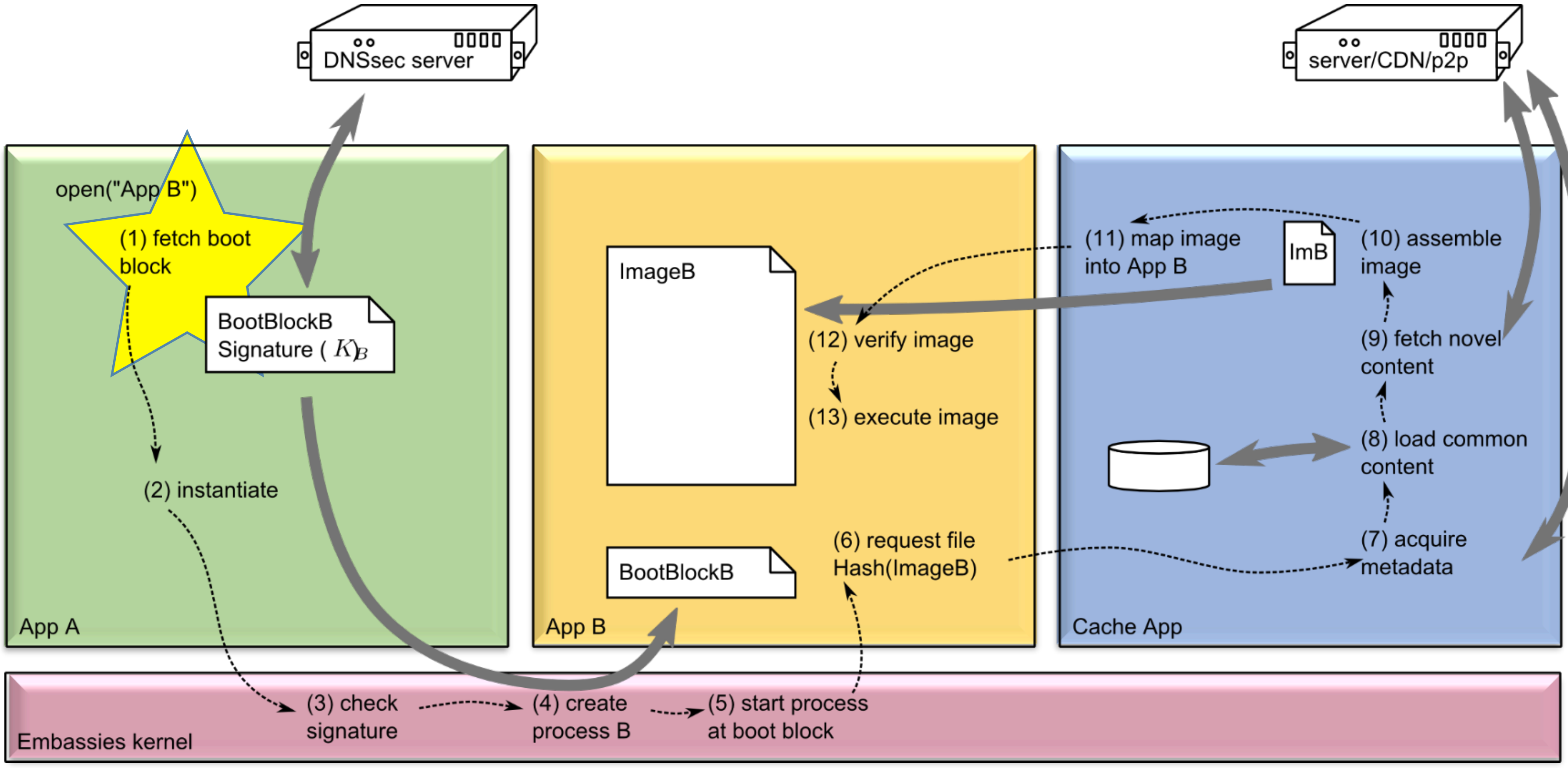
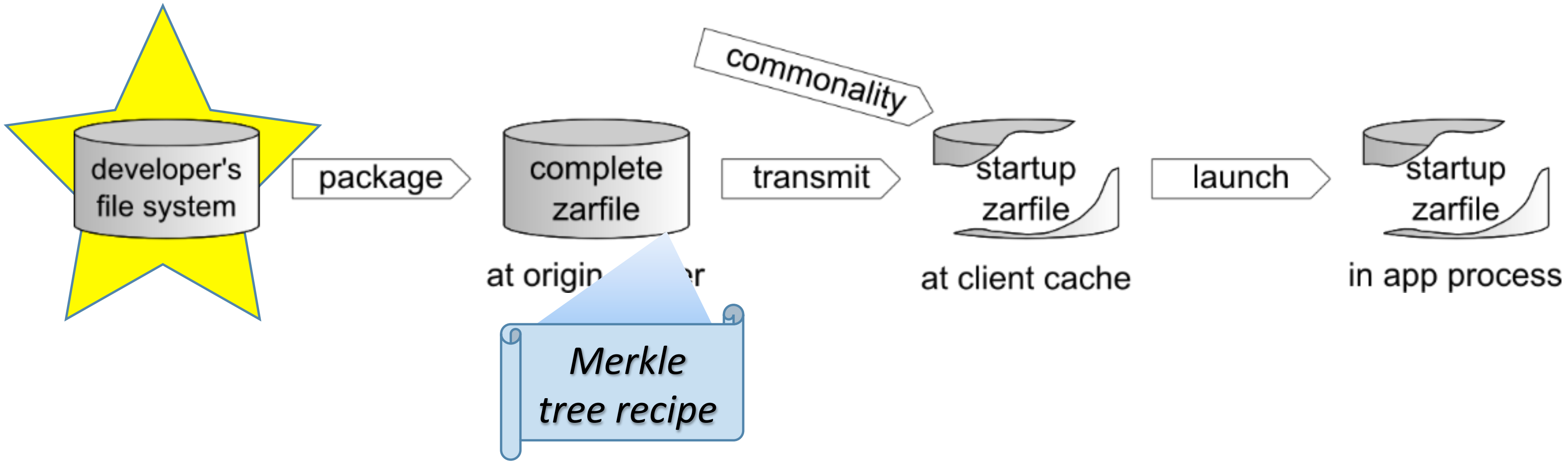
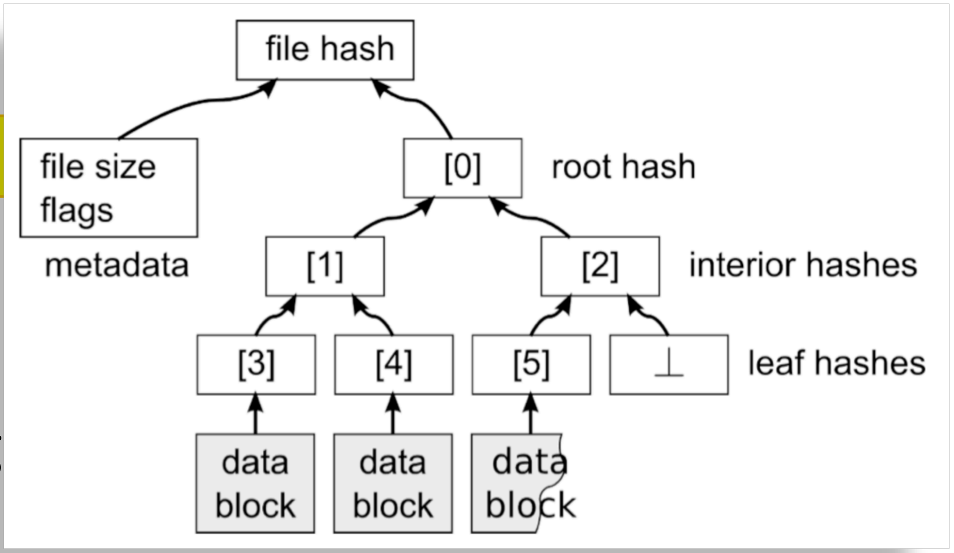


Image provenance

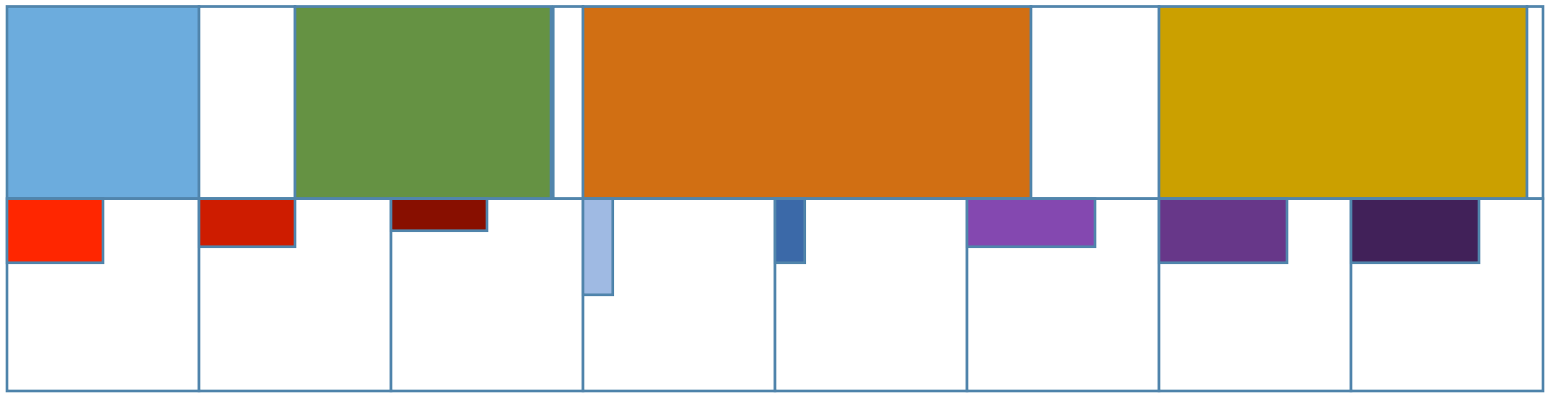


Zarfile structure

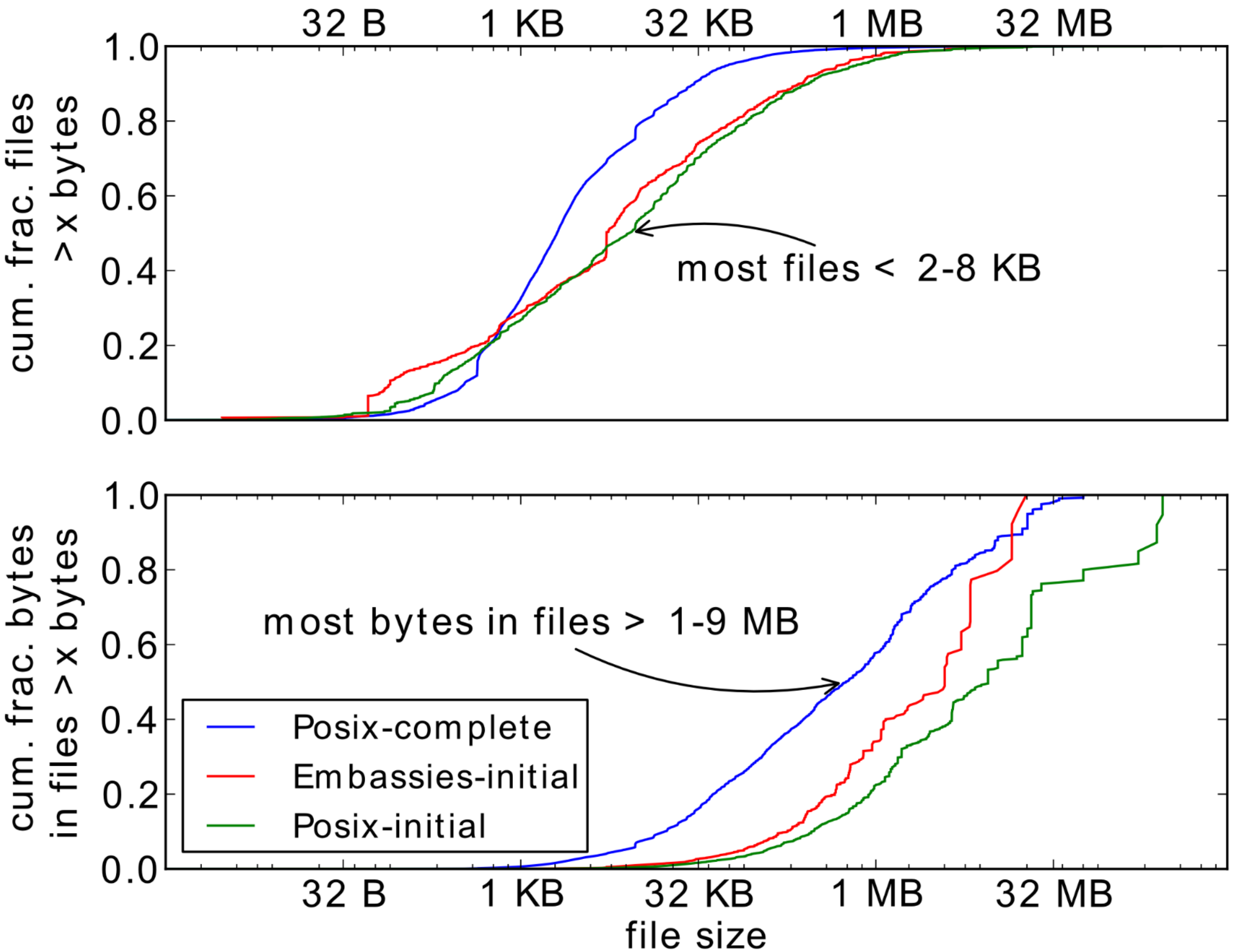
- Blocks vs. strings
- Block size
    - Small blocks: big recipes
    - Big blocks: extra padding
- Small-file packing
- Merkle tree degree
- File placement with consistent hashing



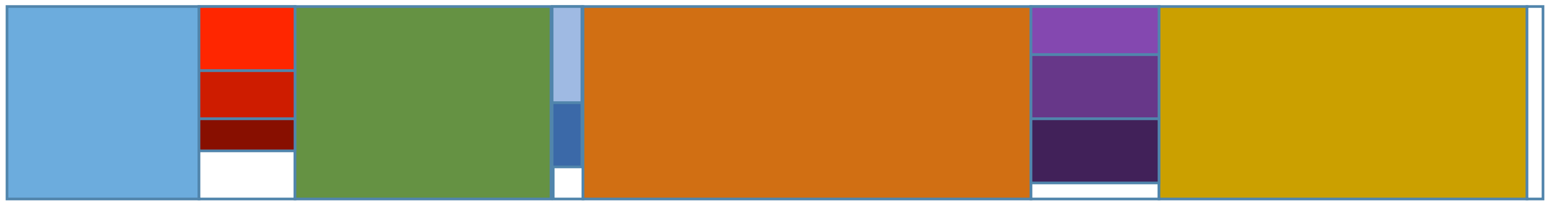
Small file packing



Small file packing



Small file packing



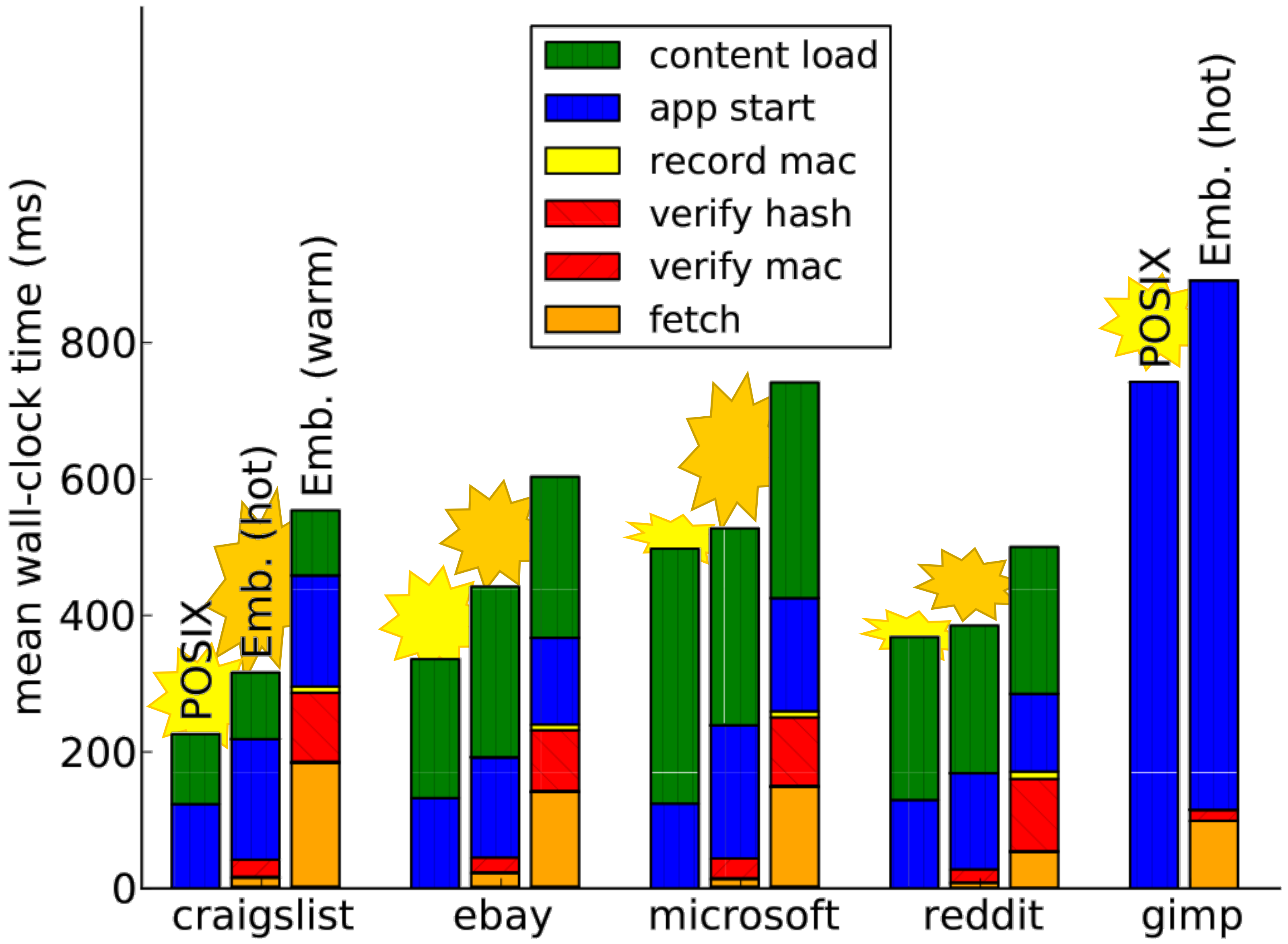
Zarfile structure

- Blocks vs. strings
- Block size
  - Small blocks: big recipes
  - Big blocks: extra padding
- Small-file packing
- Merkle tree degree
- File placement with consistent hashing

Fast verification

- SHA-1: 390ms   (100MiB)
- VMAC: 29ms
    - depends on a secret
    - how can boot block have a secret?

Launch times



Summary

- Client apps exhibit commonality
- Untrusted cache costs ~100ms
- We really can deliver 100MB apps