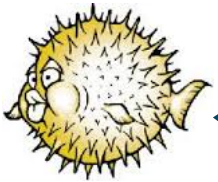
DPB: One protocol to build them all

Marc Espie <espie@openbsd.org>, <marc.espie@epita.fr>



September 27, 2025

A sample configuration for DPB

```
1  STARTUP=%p/cleanup
2  LOG_USER=espie
3  DEFAULT chroot=/vide/build
4  localhost
5  verycloudy1
6  verycloudy2
7  verycloudy3
8  verycloudy4
9  COLOR=1
10 FETCH_JOBS=16
```

RTFM **bulk(8)**

Demo 0

Demo !

Everybody cheats

`DPB_PROPERTIES=parallel` is a lie.

What it does

- It starts the jobs with `MAKE_JOBS=4`
- Then it hopes to have enough cores to run it.

The model

- For instance, each machine in the build has 8 cores
- Each core gets affected to a given build
- ... so we have 8 cores $\rightarrow$ 8 builds
- When we start parallel
- $\rightarrow$ we hope that by the time we get to it, we will have enough cores (they get "swallowed" as the DPB code says)

Good / bad

- Ideally, by the time we end `configure` we have swallowed enough cores
- $\rightarrow$ so concurrency doesn't go $C \geq 8$
- If we're unlucky, we haven't got any cores
- $\rightarrow$ so concurrency gets as high as $C = 11$
- that's the reason why by default,

$$\text{parallelism} = \frac{\text{cores}}{2}$$

Demo one

Sidebar: OpenBSD's `make` *does the same thing*

- in recursive mode, if you're not careful, you end up with n^p jobs with p the depth of recursion
- $\rightarrow$ so make detects recursion, and holds on creating new jobs while it's going deeper, hence we end up with at most $n.p$ (and quickly n jobs, assuming each compile is "fast")

When dpb goes bad

We still have monsters like `iridium`, `chromium`, `chromium-ungoogled`
When two of them get scheduled on the same machine, the build won't finish cleanly.

Meanwhile in Vera Cruz...

- I have a colleague who works on pervasive computing
- and I do compiles on my workstation where I browse ~~porn~~the web
- $\rightarrow$ what if I could adjust builds on the fly
- $\rightarrow$ the main idea is that jobs are *short*

Introducing buildcontrol

See `https://github.com/marcespie/build-control` for the code

- Simple protocol with a socket
- You allocate a build from the server
- You get a `BUILDSOCKET` and `BUILDTOKEN` to put in your environment

Demo two

The protocol

- you ask the server for a new build, it gives you the `BUILDTOKEN` to use
- the build client connects to the server and passes back the `BUILDTOKEN`
- $\rightarrow$ because of recursive makes, so several processes may be part of the build
- each time you adjust the number of jobs, each client with the same token receives the new number of jobs
- $\rightarrow$ as most compiles are quick, the adjustment is instantaneous
- the `BUILDTOKEN` is a number + random hash in the default implem
- when the last client disconnects, the build corresponding to the `BUILDTOKEN` is "over". (might be an issue)

Build programs particular

```
1  // initialization
2
3  int maxjobs = parseparallelism_option():
4
5  // main loop
6
7  while (!finished) {
8          while (jobs < maxjobs && work_to_do()) {
9                  start_new_job();
10         }
11         wait_for_job_to_finish();
12         adjust_work_to_do();
13 }
```

Build programs particular

```
1  // initialization
2
3  int maxjobs = parseparallelism_option();
4  bool buildcontrol = try_connecting_to_server();
5
6  // main loop
7
8  while (!finished) {
9          if (buildcontrol)
10                 poll_to_adjust(&max_jobs);
11         while (jobs < maxjobs && work_to_do()) {
12                 start_new_job();
13         }
14         wait_for_job_to_finish();
15         adjust_work_to_do();
16 }
```

The devil

- Beware: most make implementations have special code for parallel $= 1$
- parallel=0 is fun

Show me the code

The code

integration into dpb

warning: this has not been finished yet (because of network trouble at my dayjob)

- just pass `BUILDTOKEN` and `BUILDSOCKET` through `MAKE_ENV`
- set them for `PARALLEL` only
- use the `FullPkgPath` for the hash token
- set up a simple server, similar to external
- make sure to only set them on `make build` proper (.e.g, disable small)
- control lifespan of fds

integration into dpb 2

- server similar to external, that matches connexions with hashes
- need to tie main core with job and fullpkgpath
- if we get a connection, this means we control the jobs

Odds and ends

- no real need for more security
- could pass more info like the initial number of jobs
- or stuff like [200/500]

Thank you

Questions ?