Data Science
→ FreeBSD
→ ARM64
Who am I?

Maciej Czekaj, PhD

- ARMv7/ARMv8 embedded software
  - FreeBSD
  - Linux
  - Marvell Armada, ThunderX, Octeon
- Dataplane networking (Telecom/Security)
  - DPDK
    - First ARMv8 40GB/s Ethernet driver
  - ODP
    - Port for ThunderX & Octeon

Lead S/W engineer @ Semihalf

- TCP/IP stacks
- FPGA
  - Kornik 100G Ethernet Smart NIC
- Clang compiler
  - Xtensa HIFI support
- Comp. Sci. PhD
  - AGH University, Kraków, Poland
  - Hardware acceleration of traffic classifiers for high throughput Ethernet
Agenda

- The task
  - Data Science experiment
- The platform
  - ThunderX2
- The stack
  - Data Science software stack on FreeBSD
- The execution
  - Large-scale simulations controlled by simple means
- The aftermath
  - What worked well, what needs to be improved
The task

Massive simulation experiment
- Network device simulation
- Input: pcap files
- Output
  - Event trace
  - Statistics
- Technology:
  - Custom simulator in C++
  - Statistical s/w in Python
- Originally on Linux Desktop
The scale

Massive simulation experiment

- ~ 200 input files spanning 100 GB
- ~ 200 GB of output data
  - Custom binary format
- ~ 1000 simulation experiments
- 1 experiment takes up to:
  - 1 hour on 1 CPU core
  - 30 GB of RAM
- How to make it scale?
  - Simplify I/O
  - Coarse-grained parallelism
The platform

Workload characteristics
- 1 Core Bottleneck: memory latency
  - Large data structures: hash tables, trees
- System bottleneck: memory bandwidth

Wish list
- Lots of RAM
- Many CPUs
- Large L3 cache
ThunderX2

- 2 x 28-core ARMv8.2
- 4-way SMT (turned off)
- 2 x 28 MiB L3$
- L3 cross section B/W: 6TB/s
- 8-channel DRAM controller
- DRAM B/W: 200 GB/s
- CCPIv2 B/W: 600 Gb/s
- FreeBSD 12.2
- ZFS SSD pool
The stack

Python data science toolkit
- Numpy
- Scipy
- Scikit-learn
- Matplotlib
- Pandas
- Jupyter Notebook
- Dependencies…
  - 10s of packages
  - Each requires specific version

 ➔  https://morioh.com/p/a42cb68ff2b5
Deployment dilemma

Deployment options
- OS package manager (FreeBSD ports)
  - bad idea (e.g. wrong Python version)
- pip - Python package manager
- pip + virtualenv ("jail" for Python)
- Anaconda - Python distro
  - only Linux & Windows & MacOS :
- Containers?

➔ https://xkcd.com/1987
Deployment dilemma

**Choice: Python + virtualenv**

- **Pros**
  - It worked! (eventually)

- **Cons**
  - Source packages
    - Python packages are often C wrappers
    - Two compilers: Clang + GCC
    - Tweaks due to Linux/x86 assumptions
  - Not totally isolated
    - Some packages provided by ports
    - System upgrade breaks dependencies

**Conclusions:**

- Jail is a must
- Keep your dev env in a container
- Upgrade when you must
- Keep the backup on the old container
- Don't chase the latest API (unless you must)
Porting C++

Ingredients

- POSIX compliance
  - minor API tweaks
- Clang compliance
  - compiler warnings
- Performance differences
  - C++ iostream is slow
  - … but it is not designed to be fast!

Conclusions:

- Standard compliance pays off
- Portability = code quality
  - Found few bugs in the process
  - Eliminated undefined behavior
- Porting was simpler than installing the Python stack!
- Use low-level I/O
  - fread/fwrite
  - Design C++ structures as C structures (POD)
What about ARM64?

**ARM64 becomes "invisible"**

- Portable code = no issues
- Frictionless recompilation
- Little-endian helps
  - Similar ABI
- FreeBSD/ARM64 is "invisible" too!
  - True Tier 1 platform!
  - Just works
The execution

- Principal guideline: no more software!
- Try standard tools
- 1 experiment takes:
  - GNU Make job
  - 2 processes joined by a pipe
  - Followed by a python script
  - Up to 30GB of RAM!
- Issues with Make job server
  - -j option not flexible enough
  - CPU load is a bad metric
  - No build system monitors RAM pressure
FreeBSD to the rescue

- Embrace the failure
  - Let the OOM killer handle it
  - Make job server would cancel the job
  - Partial artifacts would be deleted
  - ZFS maintains integrity
- OOM killer's strategy:
  - Select the process with the most pages
    - swap & active
    - private & vnodes
  - Controversial but successful!
- see vm_pageout_oom() in vm_pageout.c

From vm_pageout.c:
/* After one round of OOM terror, recall our vote.

- Brutal, but effective
  - System is responsive
  - No page thrashing
  - SSH sessions can be opened
- Blocked by
  - protect
  - procctl()
The final run

A week-long experiment:
- No surprises! (good)
- Let the Make job server do the job
- Occasional supervision & re-spin of the failed tasks
- Scientific article:
The aftermath

The Good:
- A FreeBSD success story
- ARM64 is a premier Tier 1 platform
- A decade long effort!
- Reliable platform for scientific research
- Simple tools do the job

The ugly:
- “If you work for a living, why do you kill yourself working?” – Tuco, The Ugly.
- Lack of binary packages for python
  - Anaconda not interested for now
- Need an image shop for Data Science

The Bad:
- Complex s/w ecosystem needs containers