HaLVM:
GHC running on Xen

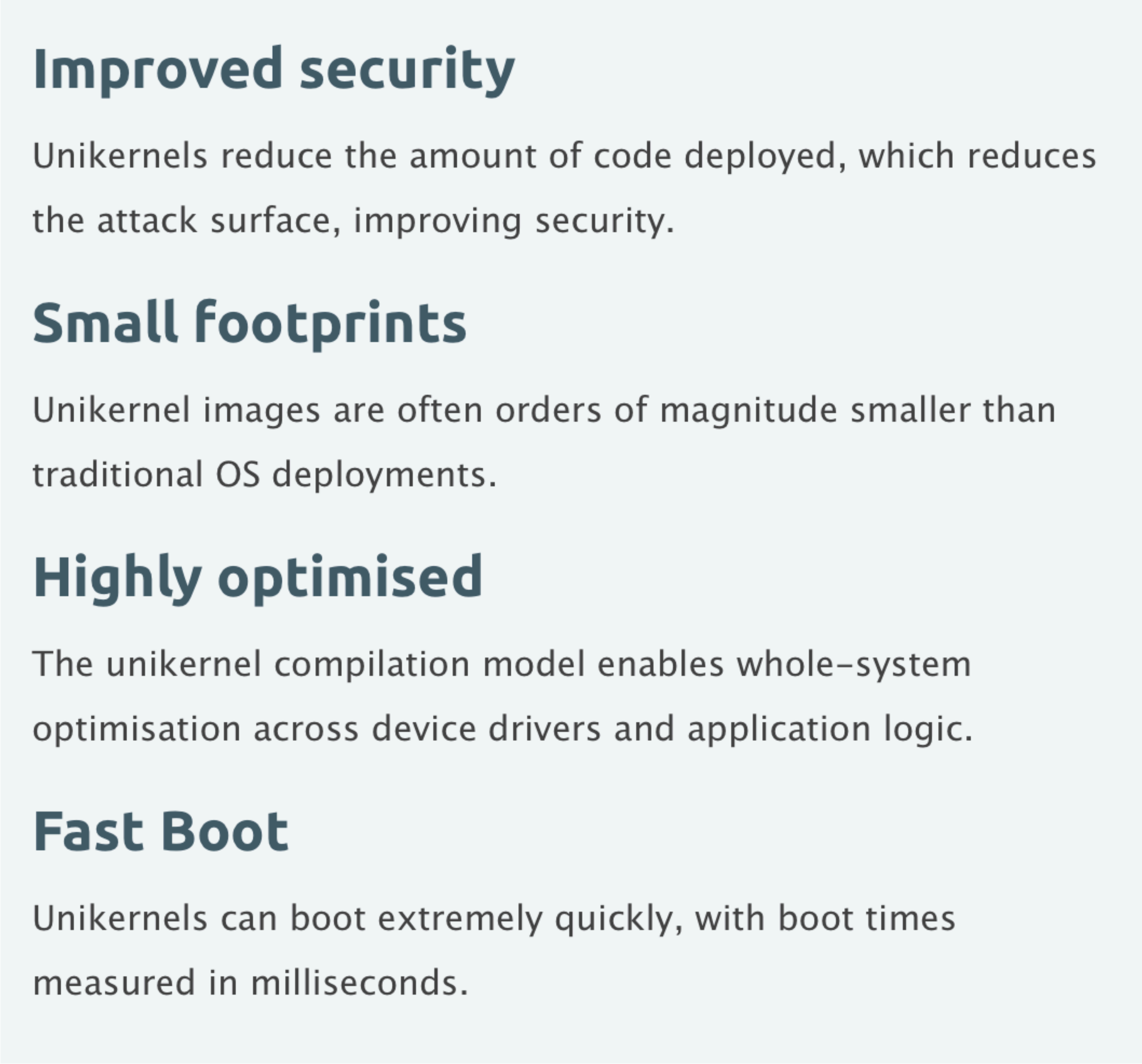
张震 (izgzhen@github, twitter, gmail)

Jan. 2017

Our topics

- Unikernel OS
- GHC Infrastructure
- GNU Autotools
- Paravirtualization with Xen
- HaLVM!

Unikernel OS



Improved security

Unikernels reduce the amount of code deployed, which reduces the attack surface, improving security.

Small footprints

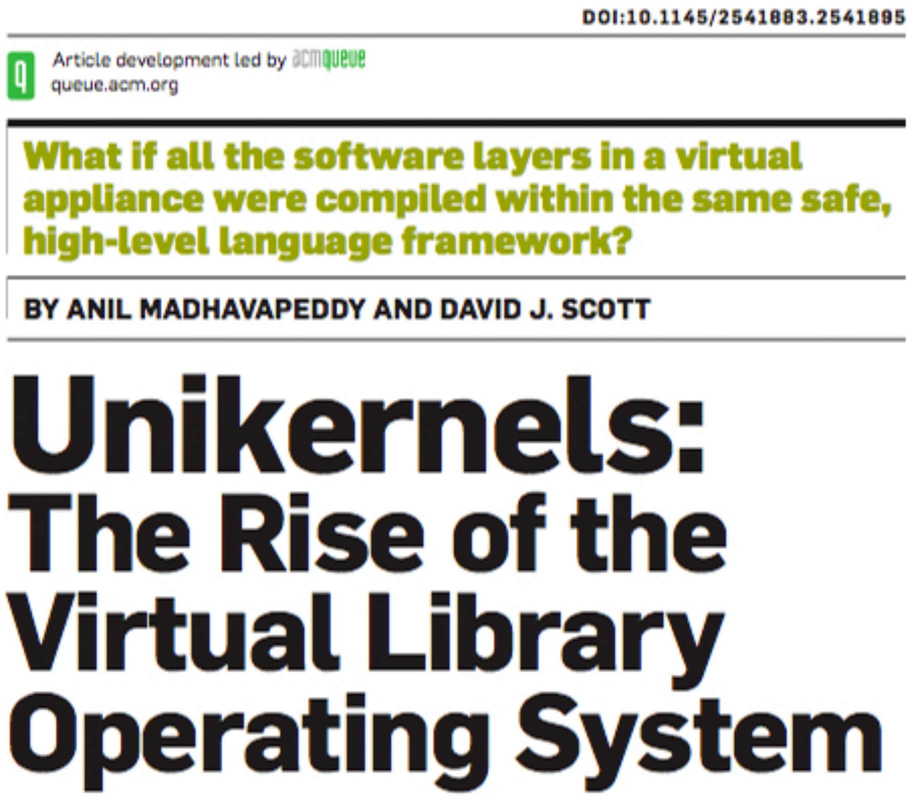
Unikernel images are often orders of magnitude smaller than traditional OS deployments.

Highly optimised

The unikernel compilation model enables whole-system optimisation across device drivers and application logic.

Fast Boot

Unikernels can boot extremely quickly, with boot times measured in milliseconds.



DOI:10.1145/2541883.2541895

Article development led by acmqueue
queue.acm.org

What if all the software layers in a virtual appliance were compiled within the same safe, high-level language framework?

BY ANIL MADHAVAPEDDY AND DAVID J. SCOTT

Unikernels: The Rise of the Virtual Library Operating System

http://unikernel.org

Unikernel OS: Is it a hype?

- “Unikernels are unfit for production” [1]
- “Unikernels will send us back to the DOS era” [2]
- “Unikernels will create more security problems than they solve” [3]

1. https://www.joyent.com/blog/unikernels-are-unfit-for-production
2. http://www.theregister.co.uk/2016/01/27/unikernels_dos_era/
3. http://thenewstack.io/unikernels-will-create-security-problems-solve/

Unikernel OS Debate: PL perspective

- **Unikernel OS**: “Immutable Infrastructure”
  - **PL**: Static & strong typed language
- **UNIX OS**: Text(String)-based, highly configurable
  - **PL**: Dynamic & weakly type language

GHC Infrastructure

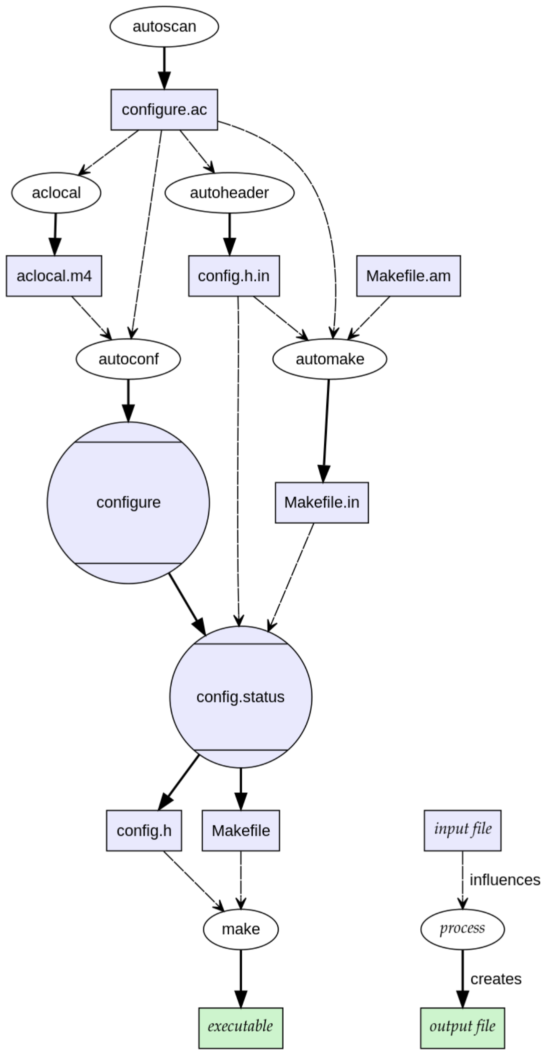
- What is Haskell: Purely Functional Lazy ***Magic.***
- GHC is written in Haskell, C, C--
  - runtime: GC, system interfaces, …
  - compiler: Haskell — Core — STG — Low-level code
  - libraries: “boot-libraries”
- Stages: Stg0 — Stg1 — Stg2 — Stg3
- The build system is largely built on GNU auto-tools (next)

http://www.aosabook.org/en/ghc.html

GNU Build System: Autotools

- User: `./configure; make`
- Developer:
  - Where does `configure` and `Makefile` come from?
    - `autoconf`, `automake`, `libtool`, `.in`, `.ac`, `.am`, `.m4`
  - What do they do?
    - Automatic dependency resolution of `.h` files
    - Portability — conditional compilation

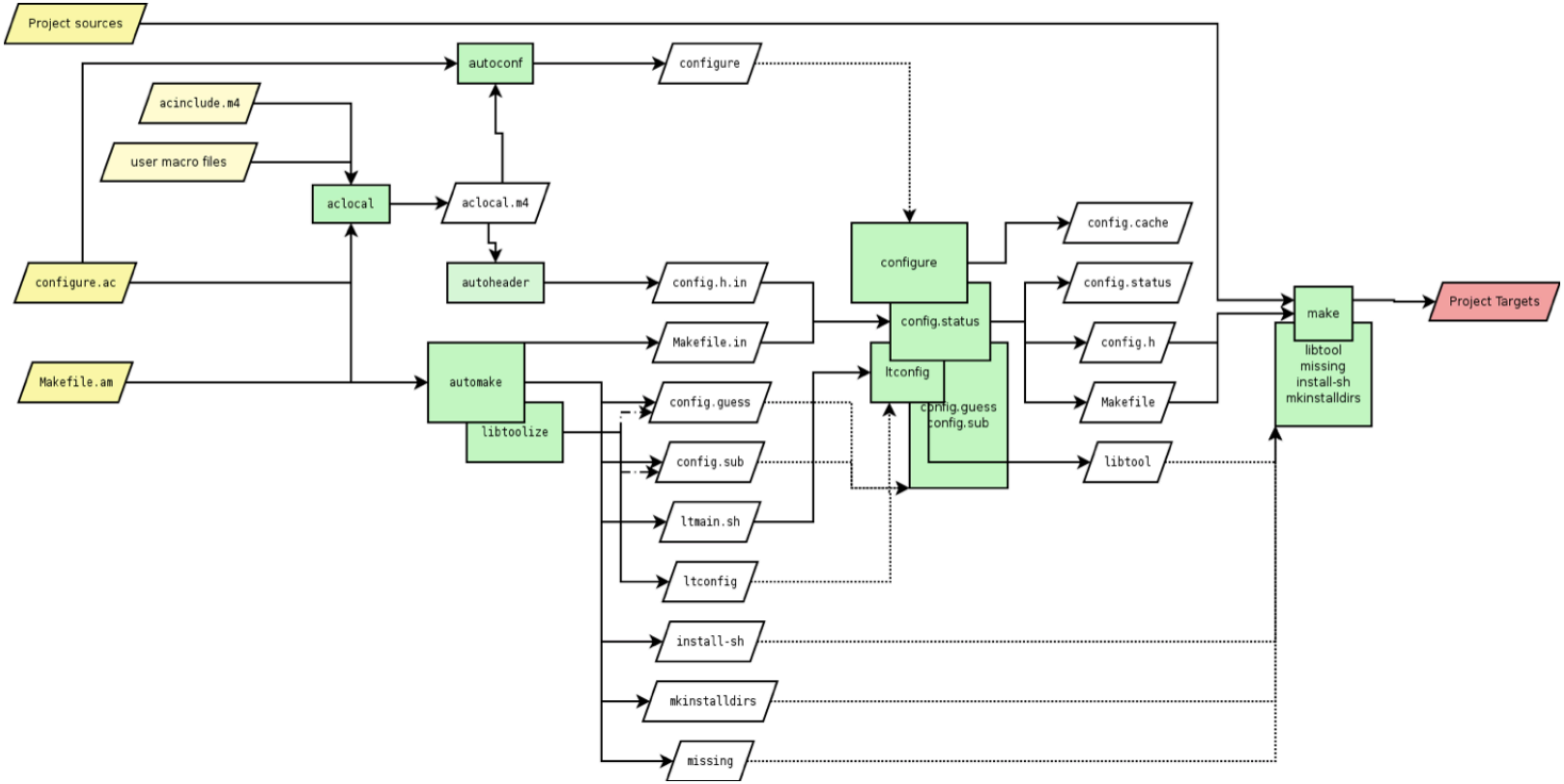
GNU Build System: Autotools



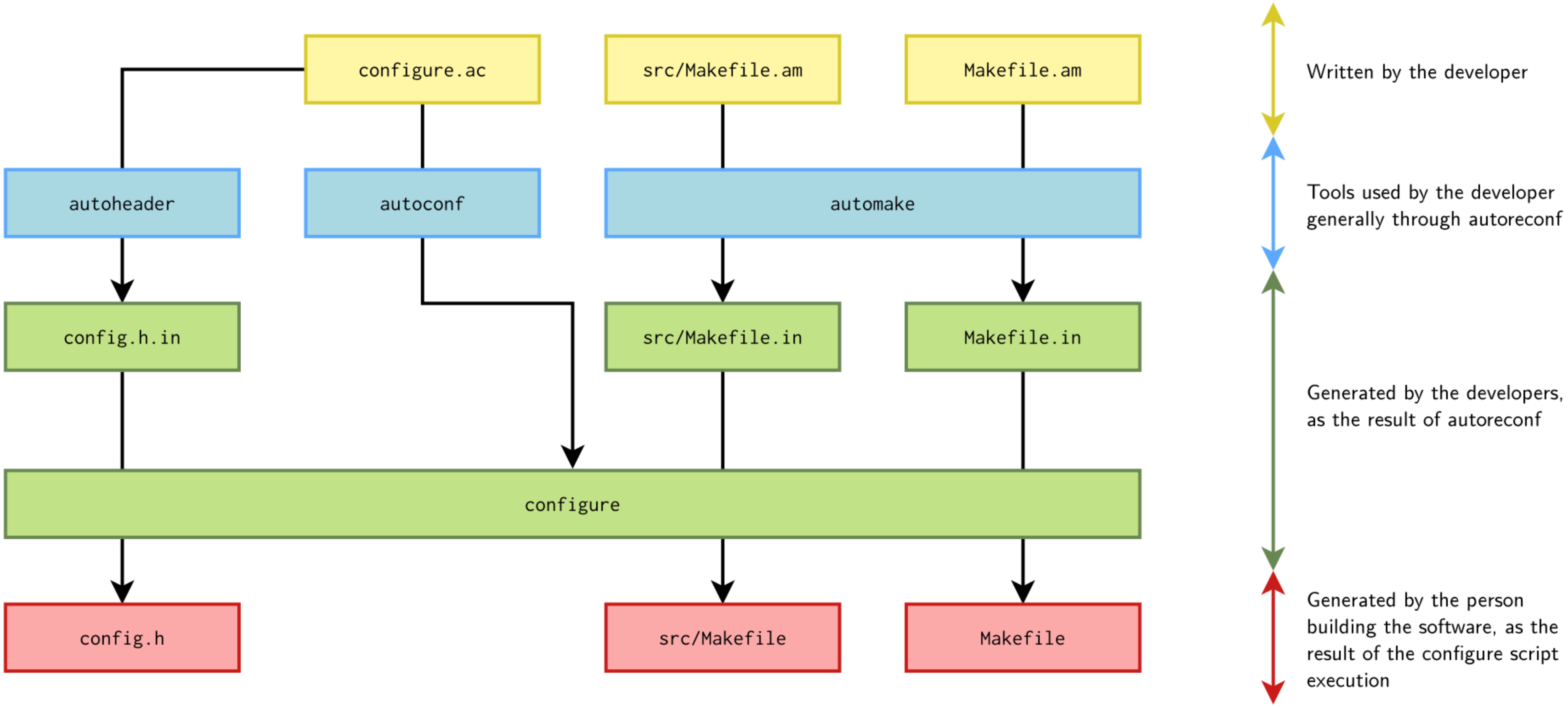
- Take-away:
  - GNU Autotools is **not** a completely arbitrary unreadable hack
- Quiz:

```
./configure --build=powerpc --host=i686 --target=mips
```

What creates what?



Overall organization



configure.ac

src/Makefile.am

Makefile.am

autoheader

autoconf

automake

config.h.in

src/Makefile.in

Makefile.in

configure

config.h

src/Makefile

Makefile

Written by the developer

Tools used by the developer generally through autoreconf

Generated by the developers, as the result of autoreconf

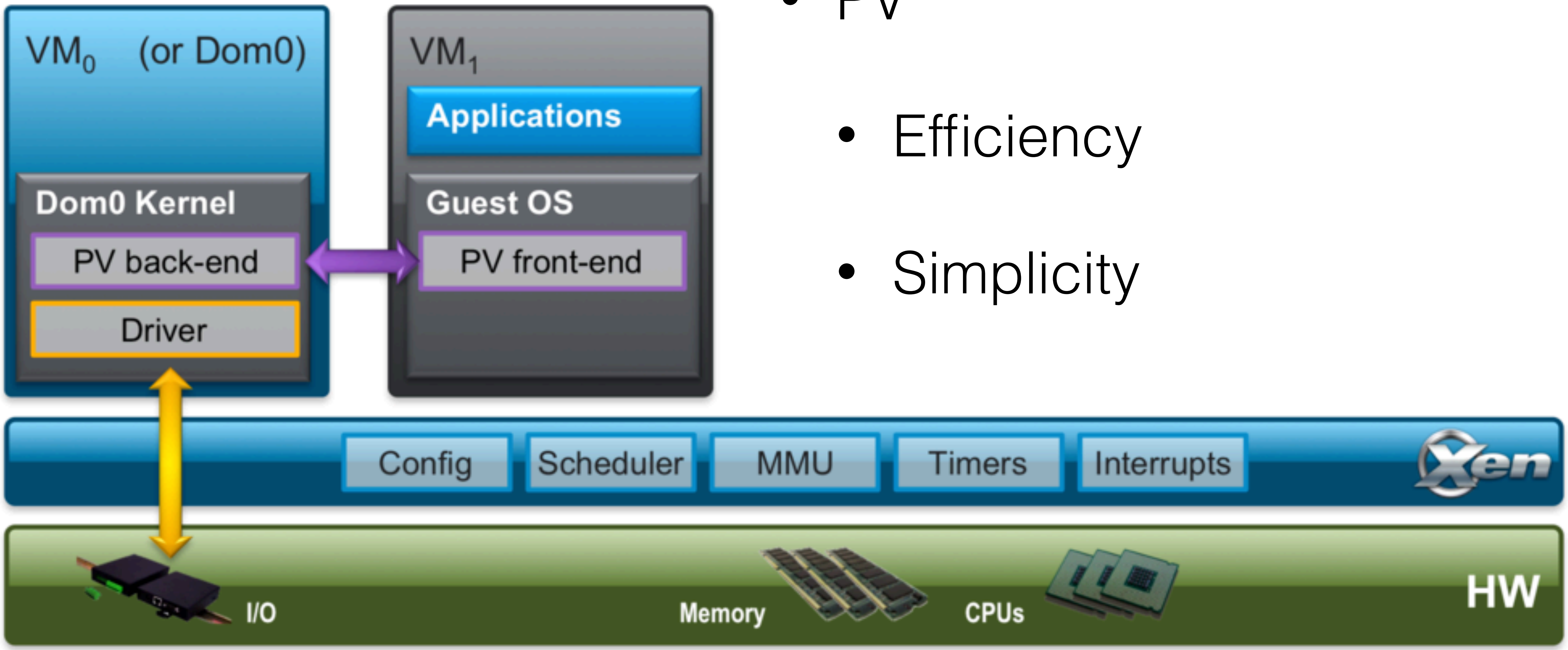
Generated by the person building the software, as the result of the configure script execution

Other build systems

- Multi-language
  - Autotools + `make`
  - CMake
  - SCons
  - …

- Single language
  - Ant (Java)
  - Cargo (Rust)
  - Cabal (Haskell)
  - …

Para-virtualization



- `include/xen/xen.h`
- PV
  - Efficiency
  - Simplicity

HaLVM!

- **Ha**skell **L**ightweight **V**irtual **M**achine
- Run Haskell programs on the OS *written in Haskell*
  - examples: WebServer, K-Means, shell
- HaLVM v3: Galois Inc => Community
  - `HACKING.md` (300 lines of markdown … only setup)
  - Port HaLVM to GHC 8.0.1

http://uhsure.com/halvm3.html

Discussion:
The next-generation OS