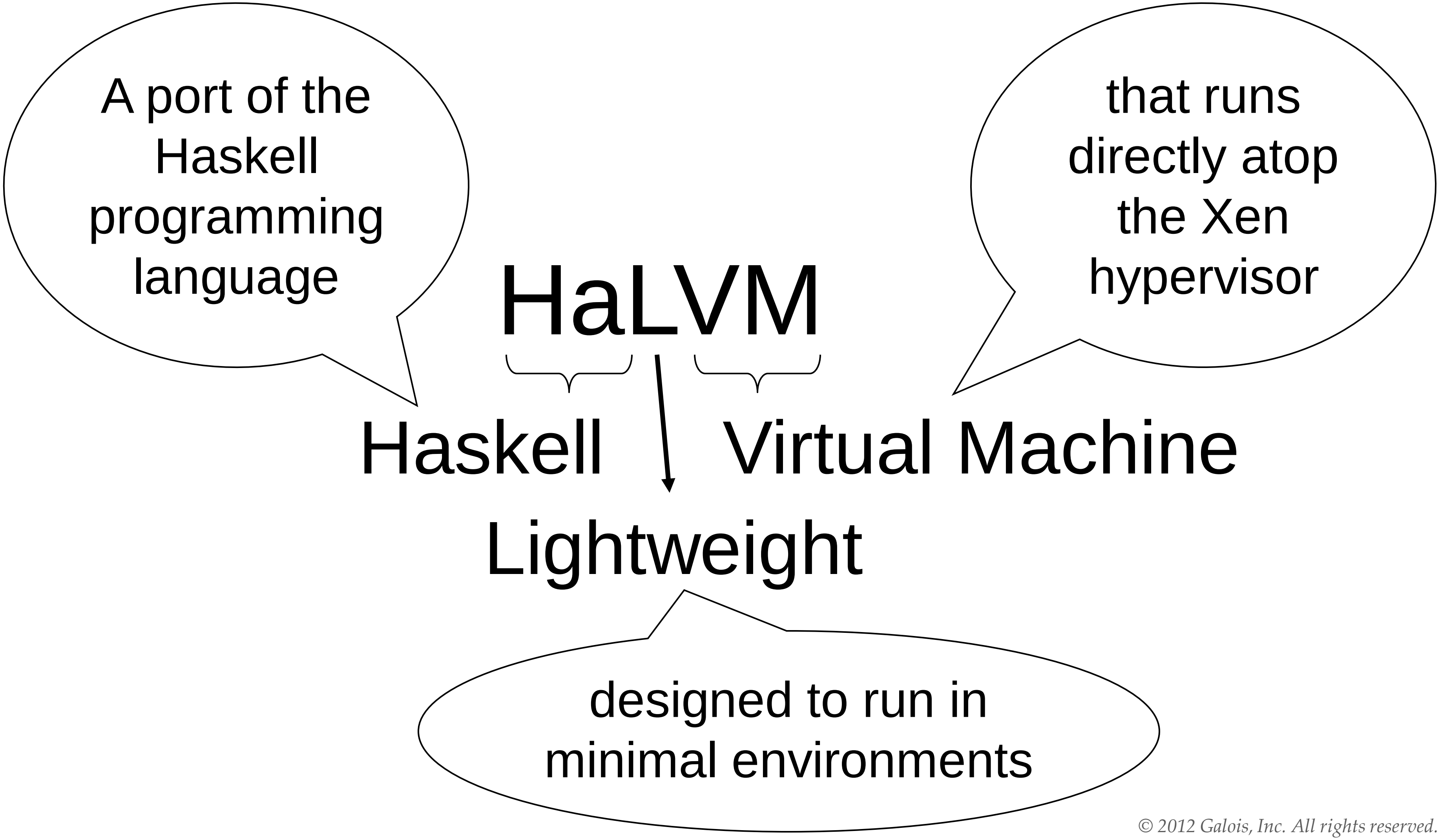
The HaLVM

A Simple Platform for Simple Platforms

Adam Wick | XenSummit | August 27th, 2012

The What?



HaLVM

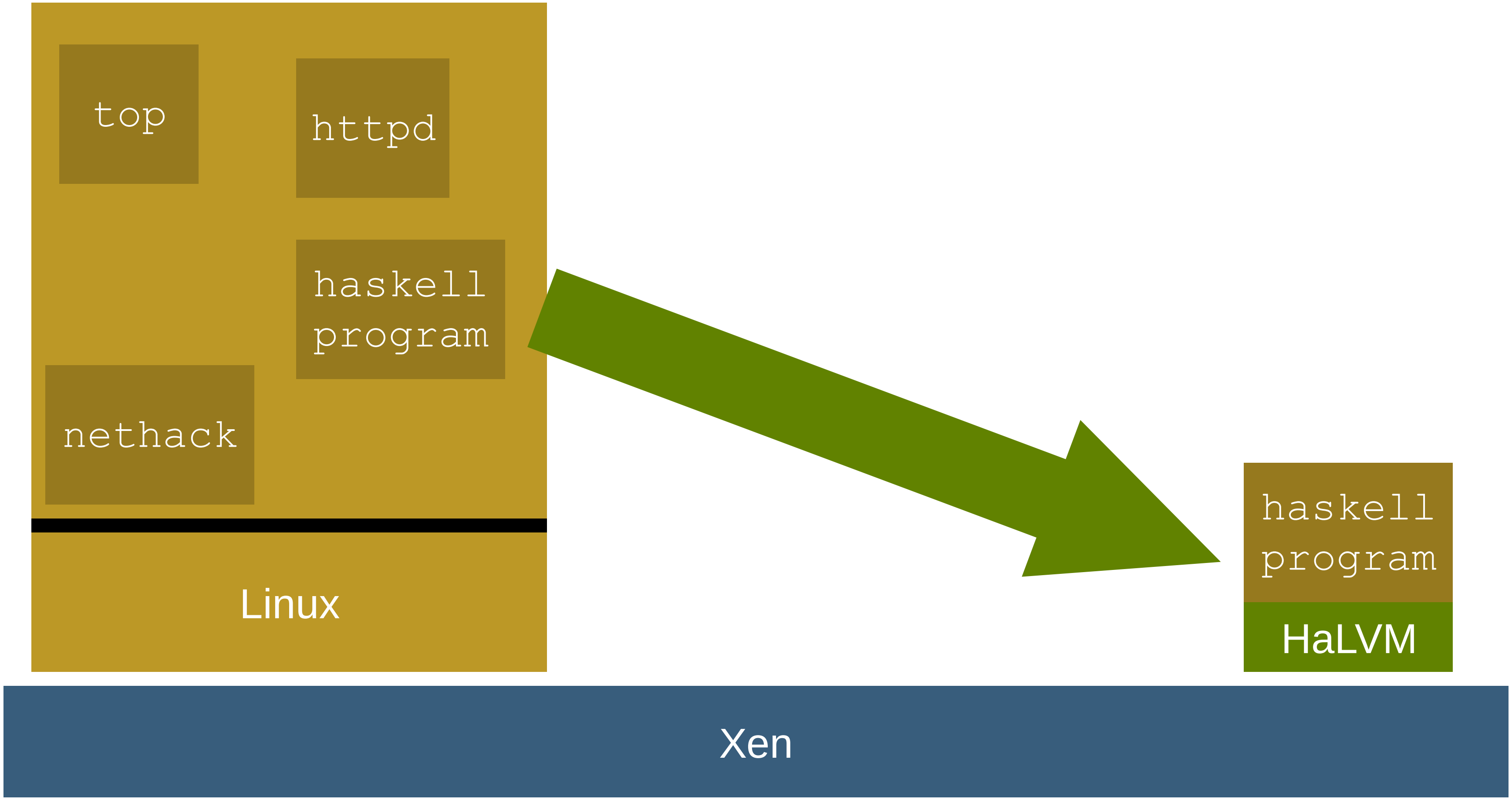
Haskell Lightweight Virtual Machine

A port of the Haskell programming language

that runs directly atop the Xen hypervisor

designed to run in minimal environments

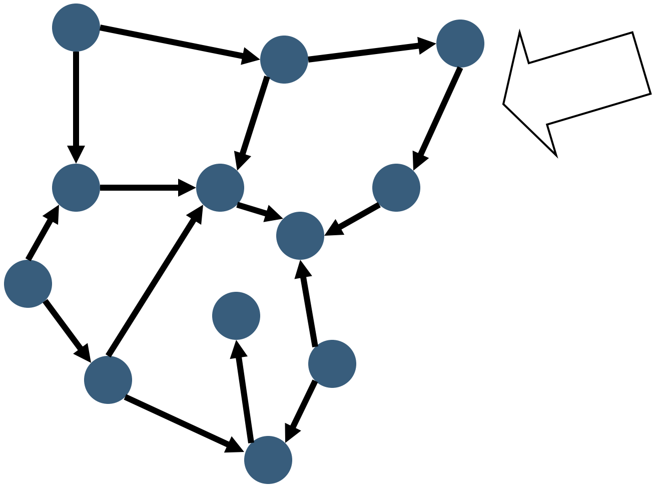
Wait, what?


top
httpd
haskell program
nethack
Linux
haskell program
HaLVM
Xen

| galois |

Why would you do such a thing?

Strangely enough, we did this because we were doing design work in microkernel-based operating systems.

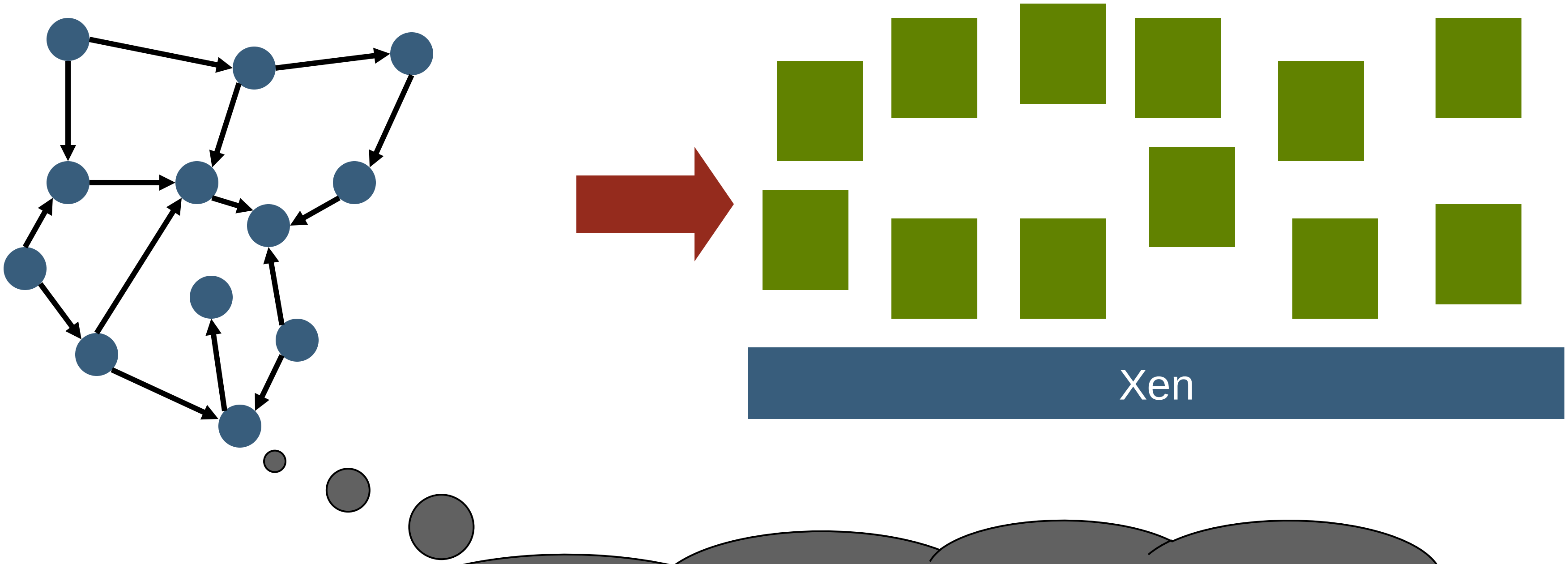


Typical component communication graphs for microkernel-based OSes look something like this.

It is very easy to create designs that look good on a white board, but include issues that make the system impossible to boot.

We designed the HaLVM as a quick way to show feasibility.

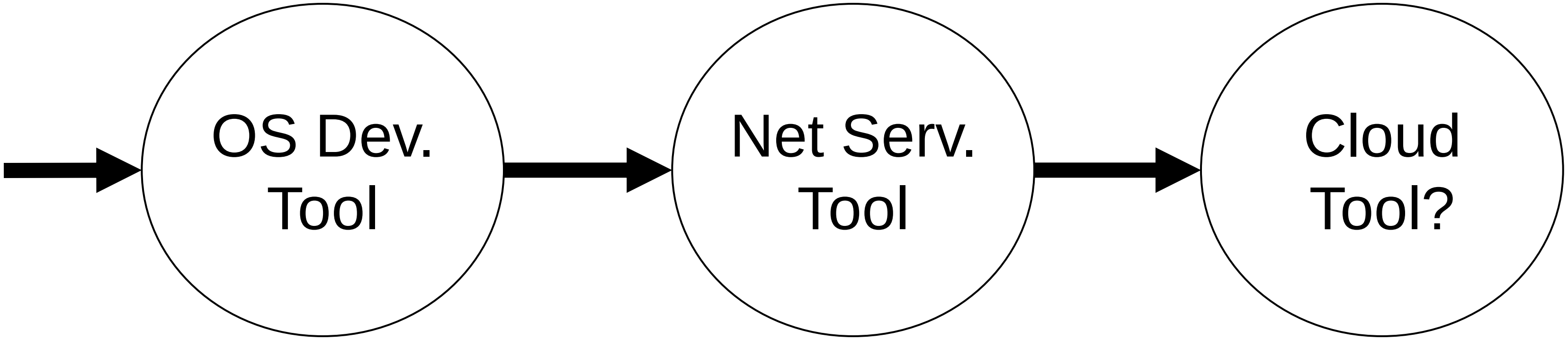
Simple Testing Apparatus



```
main = do
   block <- openBlockDeviceChannel
   dev <- openOutputChannel
   crypto <- openCryptoDeviceChannel
   print “Encrypted Disk Booted!”
```

Interlude

You may be wondering why I’m talking about a tool for doing explorations of component breakdowns for microkernel-based operating systems during a workshop about developing cloud-based services based on Xen.



OS Dev. Tool

Net Serv. Tool

Cloud Tool?

How does this happen?

```
main = do
   block <- openBlockDeviceChannel
   dev <- openOutputChannel
   crypto <- openCryptoDeviceChannel
   print “Encrypted Disk Booted!”
```

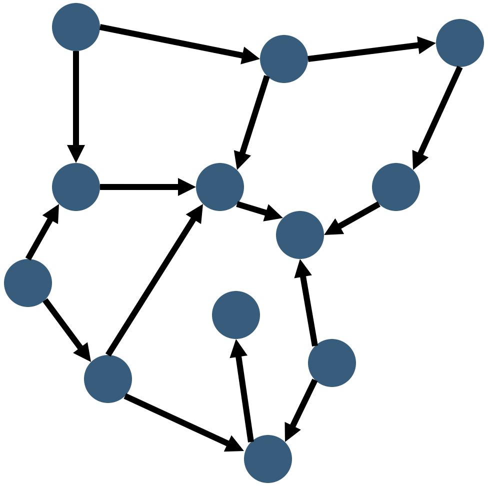
Nifty, but only checks that boot works in our imagined world.

Fails if we forgot any prerequisites in our design, and low-level details of drivers and OS components are really easy to forget.

So we let each component get more and more detailed …

```
main = do
   block <- openBlockDeviceChannel
   dev <- openOutputChannel
   crypto <- openCryptoDevice
   startServiceLoop reqchan crypto block
```

Focus on Small (and Fast)



We are not kidding on the size of the graphs we wanted to test with. One of our early projects involved 15+ domains.

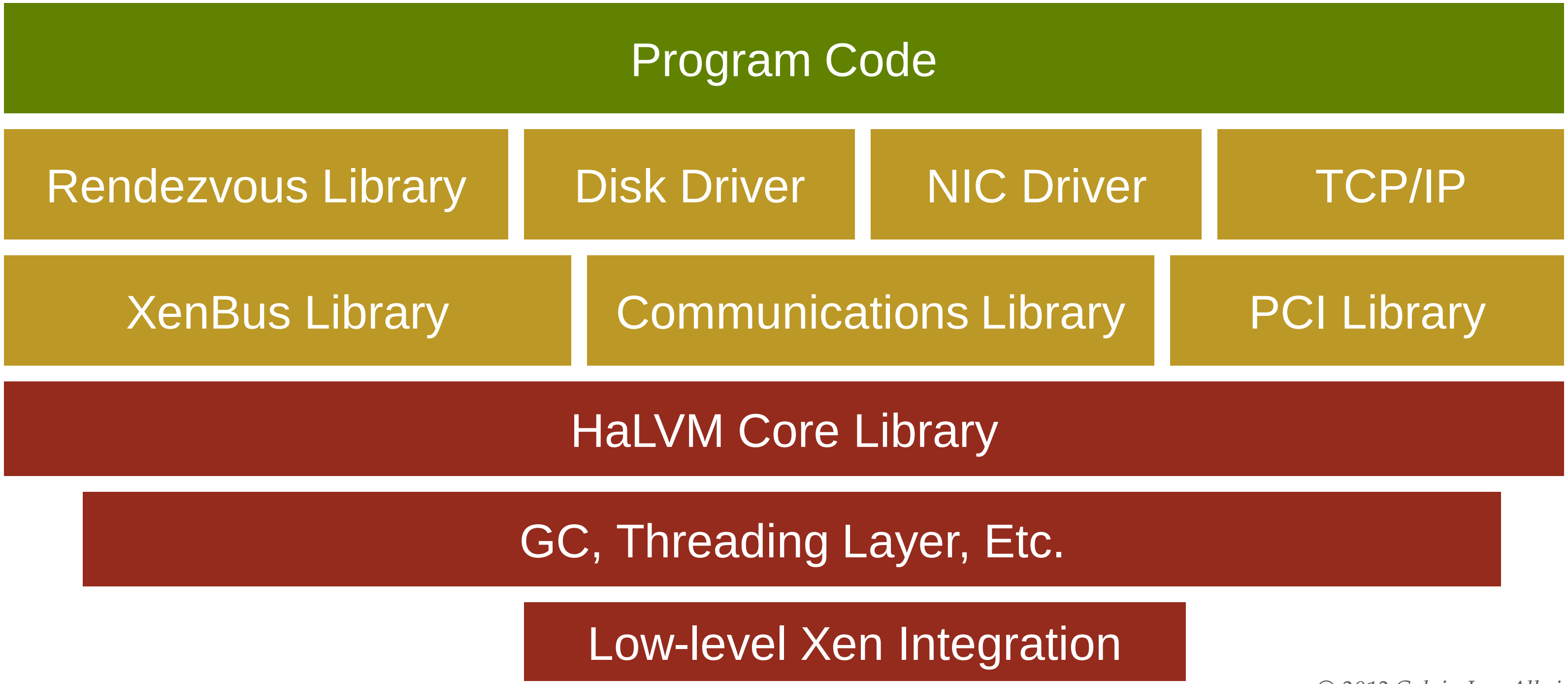
Space usage and load speed quickly became critical constraints.

15 domains * 10 second boot time = 2½ minutes to start
15 domains * 64MB footprint = 960MB basic req.

Focus on quick and fast:
→ The HaLVM boots in milliseconds
→ The HaLVM uses as little as 4MB
We are typically limited by Xen, not by design.

Abstraction Makes The Heart Grow Fonder

We designed the HaLVM to easily support this expansion by starting with the smallest core implementation possible and then providing the facilities for more complex behavior to be implemented as (layered) libraries.



Program Code

Rendezvous Library

Disk Driver

NIC Driver

TCP/IP

XenBus Library

Communications Library

PCI Library

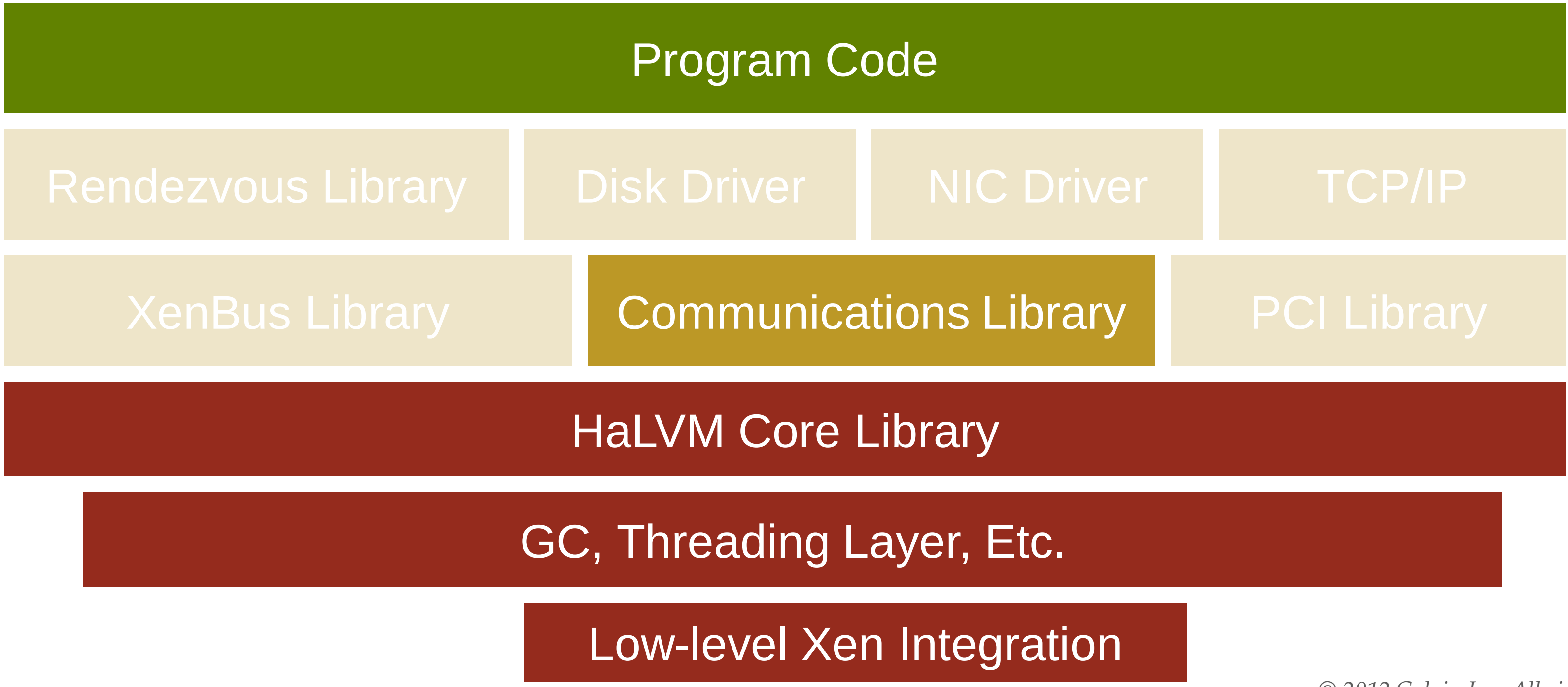
HaLVM Core Library

GC, Threading Layer, Etc.

Low-level Xen Integration

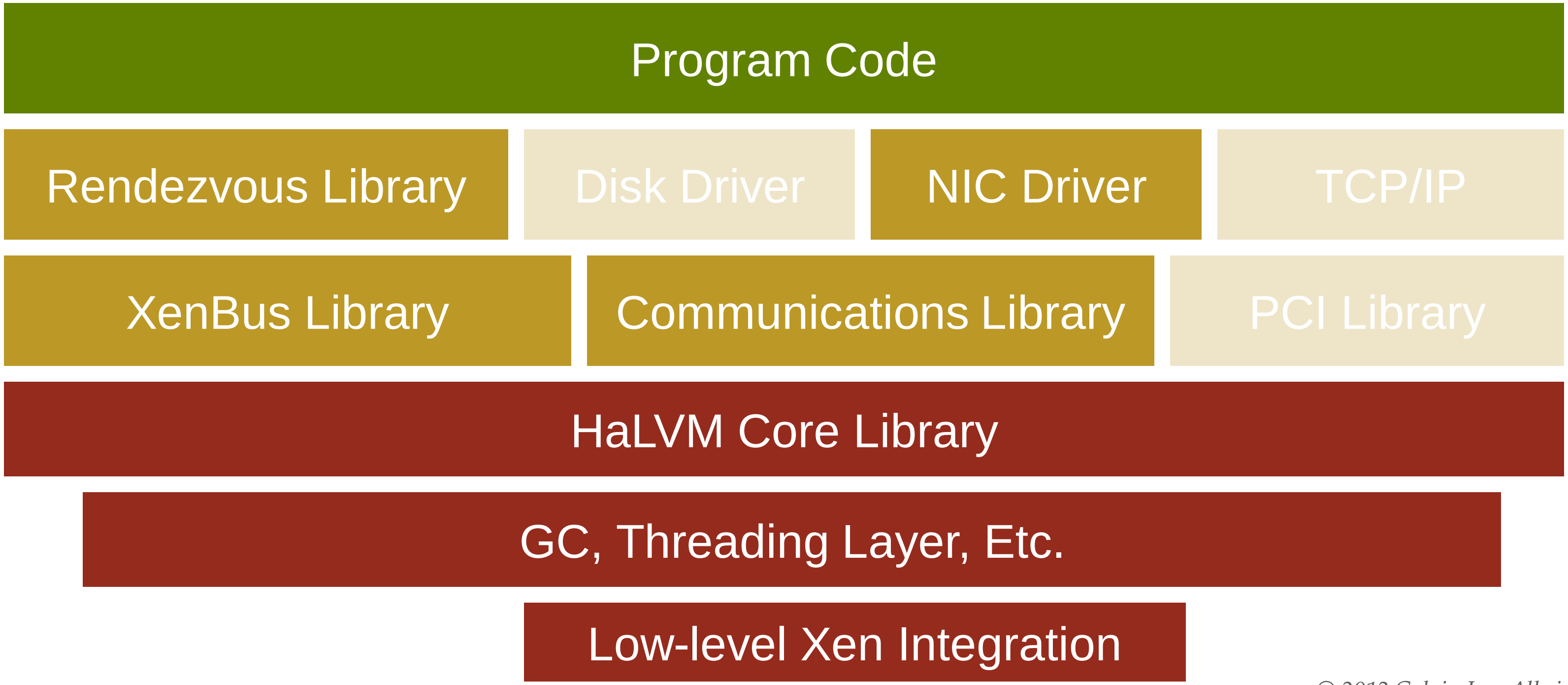
Abstraction Makes The Heart Grow Fonder

The nice thing about this methodology is that programs use only the libraries they need, and thus only pay the space and initialization cost of the libraries they need.



Abstraction Makes The Heart Grow Fonder

The nice thing about this methodology is that programs use only the libraries they need, and thus only pay the space and initialization cost of the libraries they need.



Program Code

Rendezvous Library

Disk Driver

NIC Driver

TCP/IP

XenBus Library

Communications Library

PCI Library

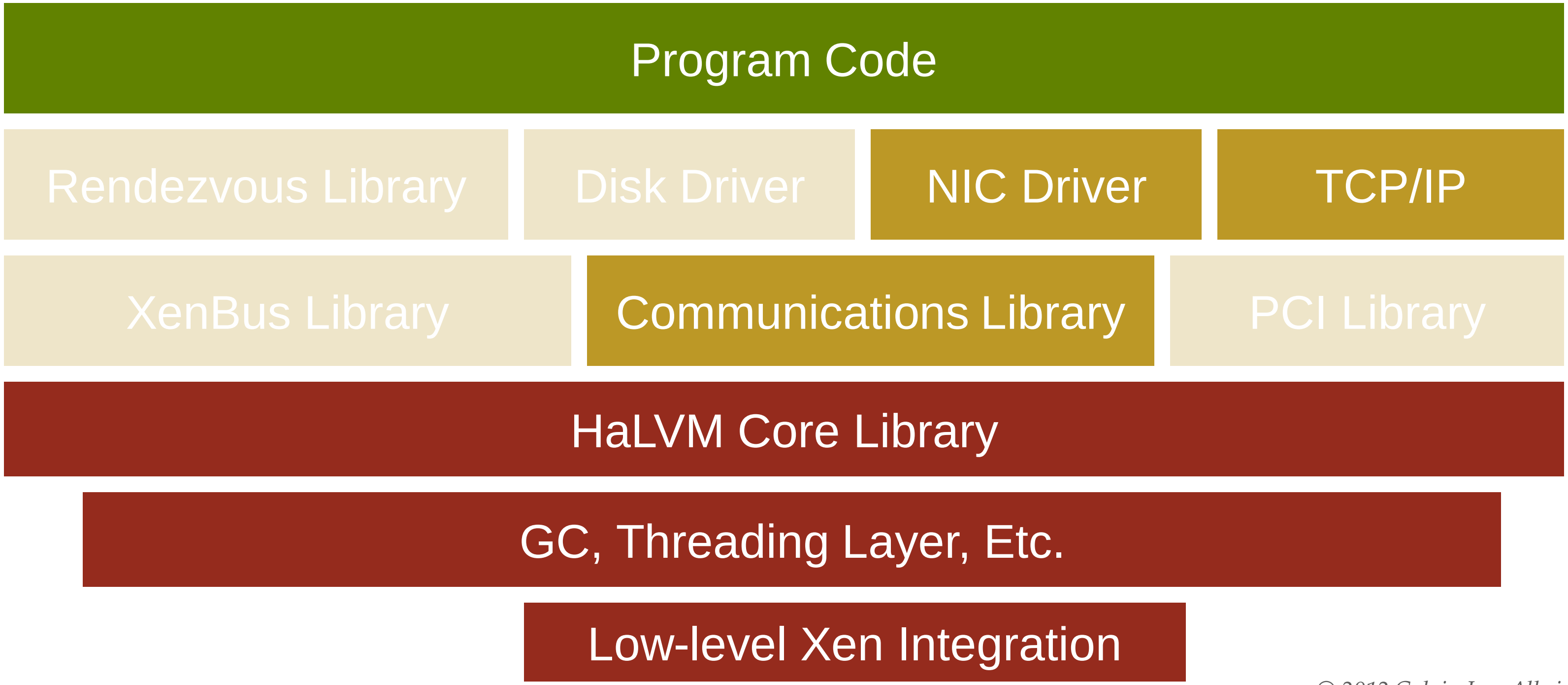
HaLVM Core Library

GC, Threading Layer, Etc.

Low-level Xen Integration

Abstraction Makes The Heart Grow Fonder

The nice thing about this methodology is that programs use only the libraries they need, and thus only pay the space and initialization cost of the libraries they need.



Program Code

Rendezvous Library

Disk Driver

NIC Driver

TCP/IP

XenBus Library

Communications Library

PCI Library

HaLVM Core Library

GC, Threading Layer, Etc.

Low-level Xen Integration

Wait, What Did We Just Do?

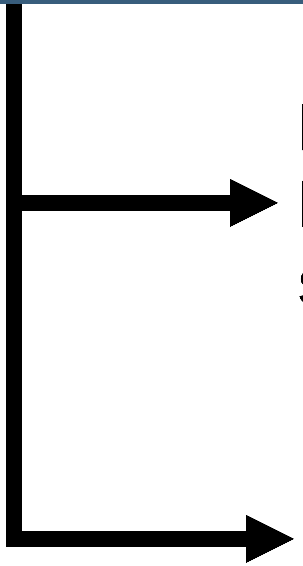
At some point it occurred to us that we had developed something that was actually useful on its own, rather than a tool for OS design work.

> If you can speak Ethernet to a network and write blocks to a disk, then some interesting possibilities emerge for small software network devices:

HaLVM random number generators, HaLVM traffic monitors, HaLVM alert systems, HaLVM chat servers, HaLVM time servers, HaLVM encryption servers, HaLVM storage servers …

If only we had …

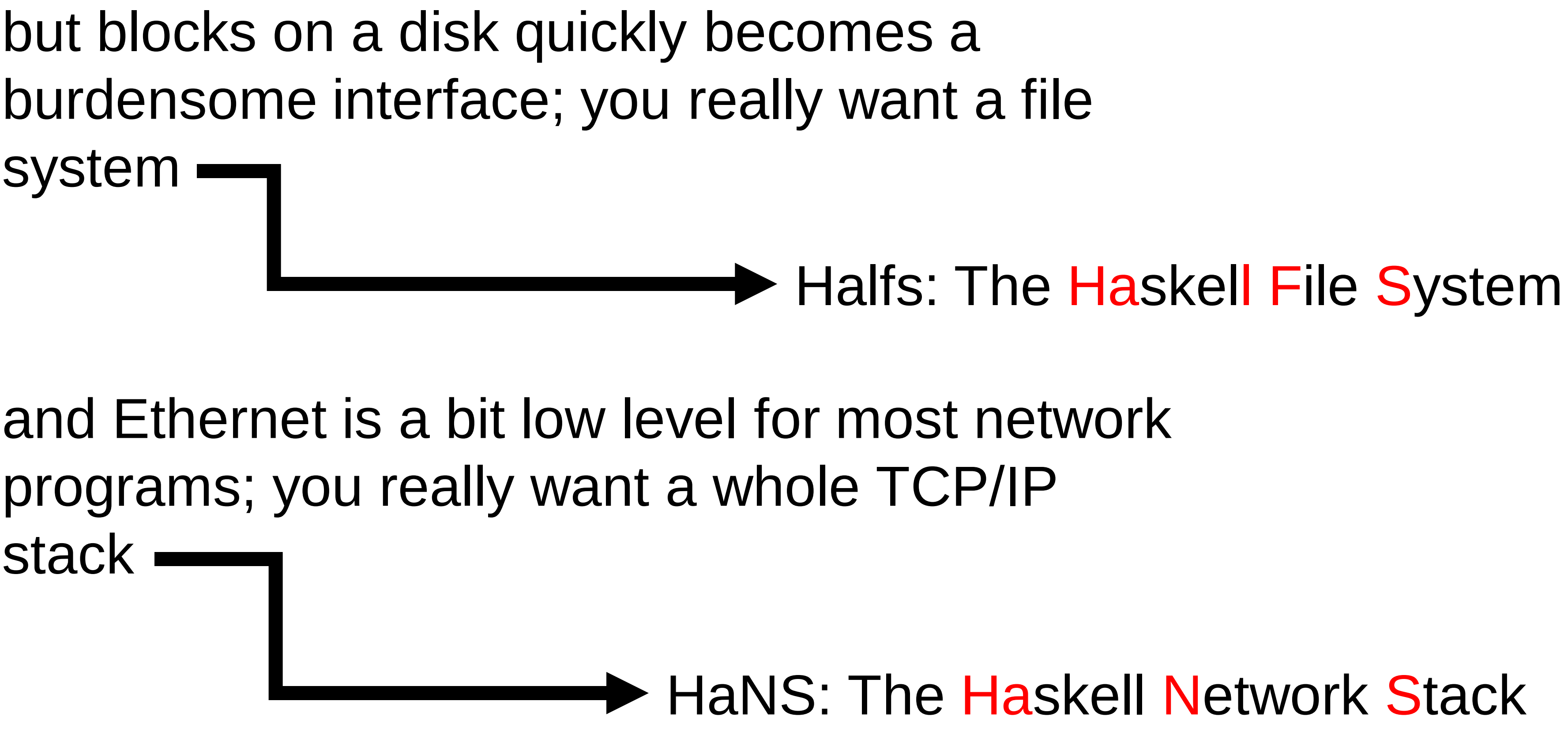
If you can speak Ethernet to a network and write blocks to a disk, then some interesting possibilities emerge for small software network devices …



but blocks on a disk quickly becomes a burdensome interface; you really want a file system

and Ethernet is a bit low level for most network programs; you really want a whole TCP/IP stack

We Really Like Haskell



but blocks on a disk quickly becomes a burdensome interface; you really want a file system

Halfs: The Haskell File System

and Ethernet is a bit low level for most network programs; you really want a whole TCP/IP stack

HaNS: The Haskell Network Stack

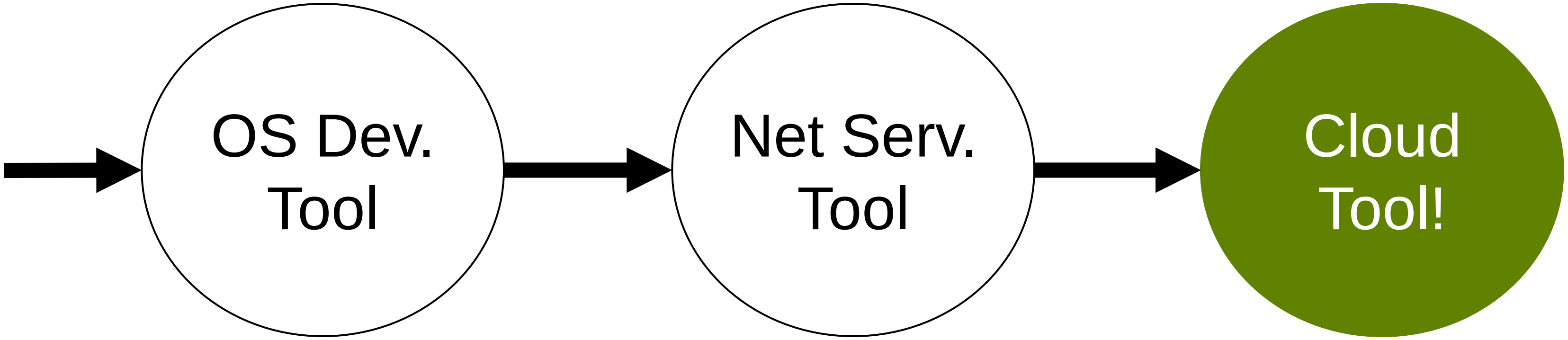
| galois |

Wait, what have we built?!

Now armed with a file system and a network stack, we had an interesting little platform for writing lightweight, single-purpose cloud services.

… which works on Xen

… which explains why I'm talking to you right now



… mostly

Cloud Tool!

This picture made a great little slide addition.

But this picture is probably a bit more honest.

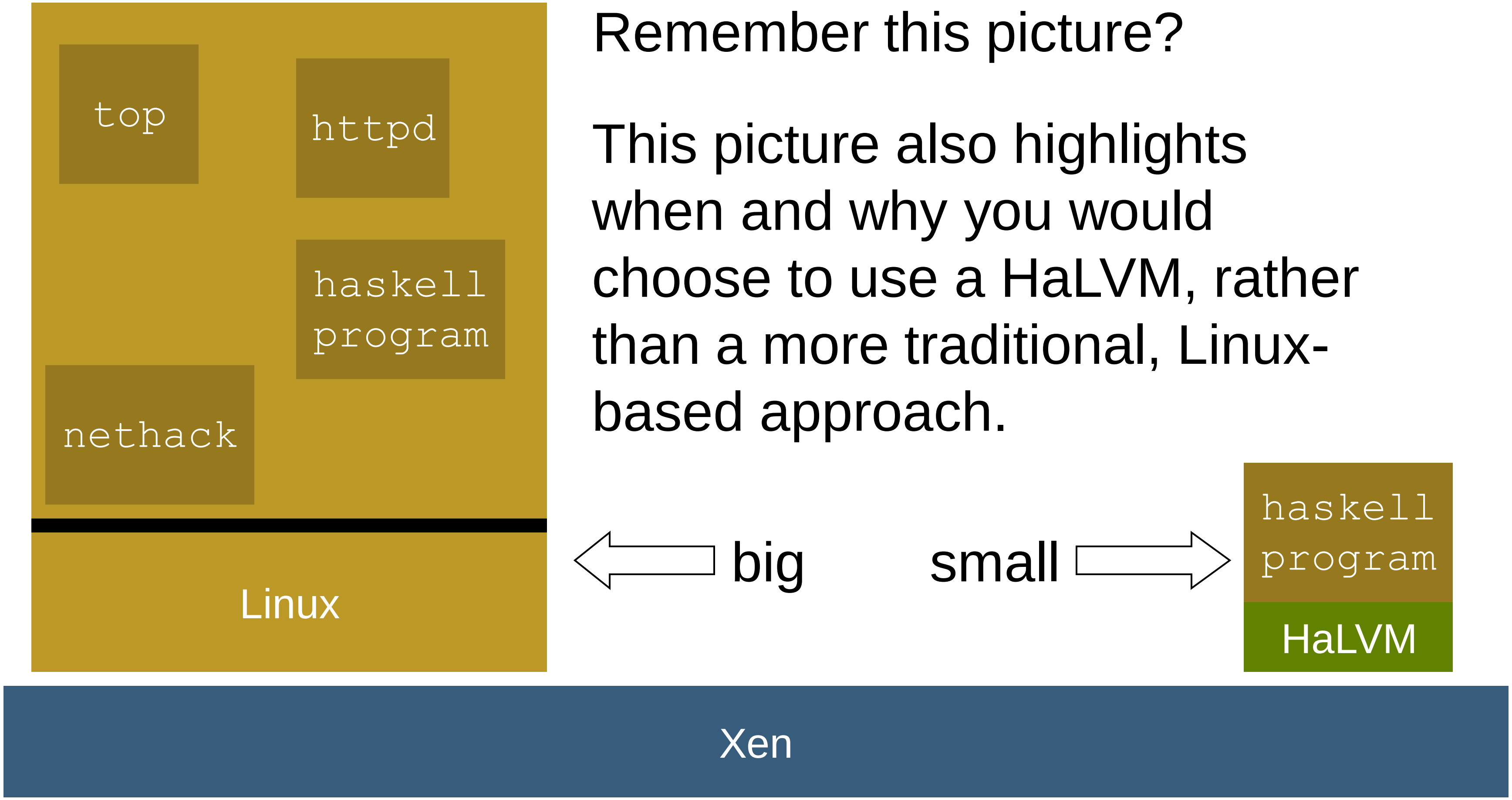
Cloud Tool?

The HaLVM is a work in progress. There are some rough edges – particularly around network stack and file system integration – but the basis is there. It will get better with time, particularly if interested parties contribute.

So, A Cloud Service Framework

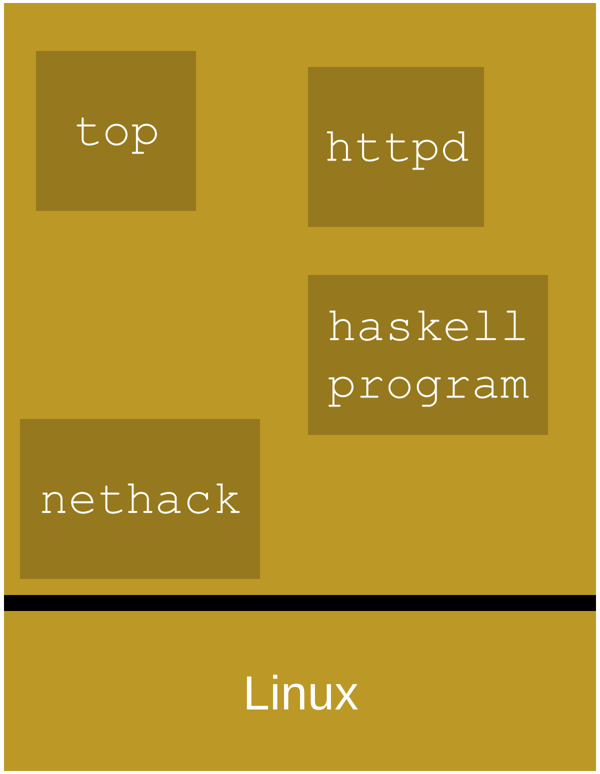
Remember this picture?

This picture also highlights when and why you would choose to use a HaLVM, rather than a more traditional, Linux-based approach.



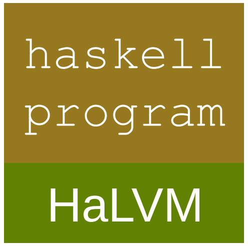
Why and Why Not

The HaLVM offers a very lightweight platform for writing simple cloud services, at the cost of more limited functionality and less access to the very highest-speed utilities.



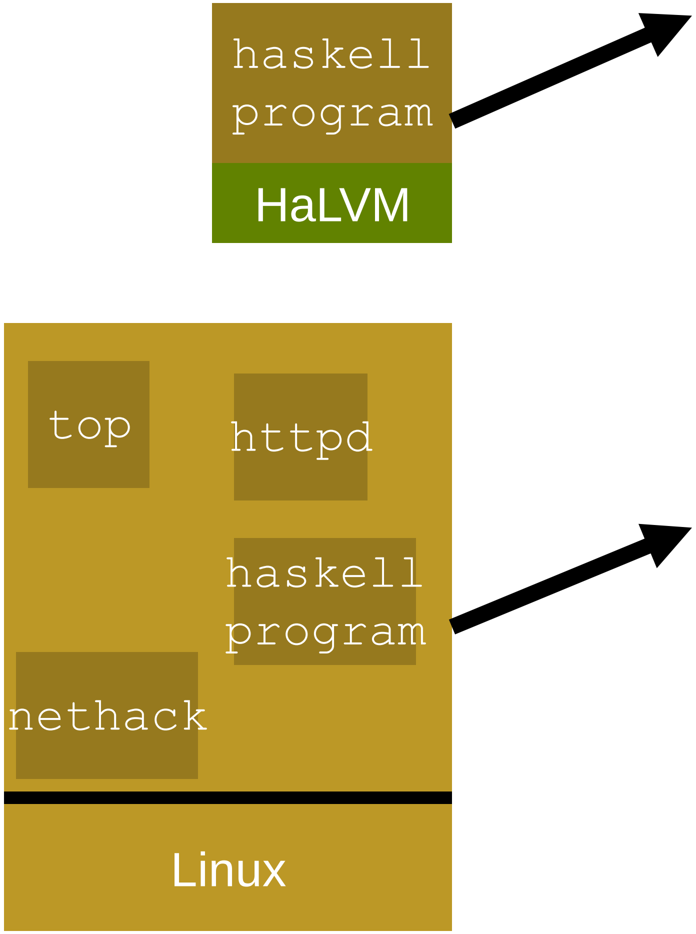
Does your service need a high-speed database server hooked up to fast business logic with a low-latency web server?

Does it need it right now?



| galois |

Why and Why Not



Good for small, single-purpose cloud services with limited dependencies.

Saves money on cloud service costs via lowered memory and disk footprint.

Good for more complicated systems with many system dependencies.

Larger selection of tools and frameworks for developing your application.

That's it …

So that is the HaLVM.

- A nifty platform for developing simple cloud services.
- A nifty platform for creating critical, isolated services.

Oh, and it's free, and available on GitHub:

Base HaLVM	https://github.com/GaloisInc/HaLVM
Network Stack	https://github.com/GaloisInc/HaNS
File System	https://github.com/GaloisInc/Halfs

… I’m done.

Questions, comments, or suggestions:

now or at awick@galois.com

Also see the HaLVM mailing list at

http://community.galois.com/mailman/listinfo/halvm-devel

Base HaLVM	https://github.com/GaloisInc/HaLVM
Network Stack	https://github.com/GaloisInc/HaNS
File System	https://github.com/GaloisInc/Halfs