Running Docker applications on Linux on the Mainframe

Jay Brenneman - rjbrenn@us.ibm.com
10 August, 2015

Trademarks

The following are trademarks of the International Business Machines Corporation in the United States and/or other countries.

AIX*	FlashSystem	Storwize*	Tivoli*
DB2*	IBM*	Spectrum Scale*	WebSphere*
DS8000*	IBM (logo)*	System p*	XIV*
ECKD	MQSeries*	System x*	z/VM*
		System z*	Z Systems*

* Registered trademarks of IBM Corporation

The following are trademarks or registered trademarks of other companies.

Adobe, the Adobe logo, PostScript, and the PostScript logo are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States, and/or other countries.
Cell Broadband Engine is a trademark of Sony Computer Entertainment, Inc. in the United States, other countries, or both and is used under license therefrom.
Intel, Intel logo, Intel Inside, Intel Inside logo, Intel Centrino, Intel Centrino logo, Celeron, Intel Xeon, Intel SpeedStep, Itanium, and Pentium are trademarks or registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.
IT Infrastructure Library is a registered trademark of the Central Computer and Telecommunications Agency which is now part of the Office of Government Commerce.
ITIL is a registered trademark, and a registered community trademark of the Office of Government Commerce, and is registered in the U.S. Patent and Trademark Office.
Java and all Java based trademarks and logos are trademarks or registered trademarks of Oracle and/or its affiliates.
Linear Tape-Open, LTO, the LTO Logo, Ultrium, and the Ultrium logo are trademarks of HP, IBM Corp. and Quantum in the U.S. and
Linux is a registered trademark of Linus Torvalds in the United States, other countries, or both.
Microsoft, Windows, Windows NT, and the Windows logo are trademarks of Microsoft Corporation in the United States, other countries, or both.
OpenStack is a trademark of OpenStack LLC. The OpenStack trademark policy is available on the OpenStack website.
TEALEAF is a registered trademark of Tealeaf, an IBM Company.
Windows Server and the Windows logo are trademarks of the Microsoft group of countries.
Worklight is a trademark or registered trademark of Worklight, an IBM Company.
UNIX is a registered trademark of The Open Group in the United States and other countries.

* Other product and service names might be trademarks of IBM or other companies.

Notes:

Performance is in Internal Throughput Rate (ITR) ratio based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput that any user will experience will vary depending upon considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve throughput improvements equivalent to the performance ratios stated here.

IBM hardware products are manufactured from new parts, or new and serviceable used parts. Regardless, our warranty terms apply.

All customer examples cited or described in this presentation are presented as illustrations of the manner in which some customers have used IBM products and the results they may have achieved. Actual environmental costs and performance characteristics will vary depending on individual customer configurations and conditions.

This publication was produced in the United States. IBM may not offer the products, services or features discussed in this document in other countries, and the information may be subject to change without notice. Consult your local IBM business contact for information on the product or services available in your area.

All statements regarding IBM's future direction and intent are subject to change or withdrawal without notice, and represent goals and objectives only.

Information about non-IBM products is obtained from the manufacturers of those products or their published announcements. IBM has not tested those products and cannot confirm the performance, compatibility, or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products.

Prices subject to change without notice. Contact your IBM representative or Business Partner for the most current pricing in your geography.

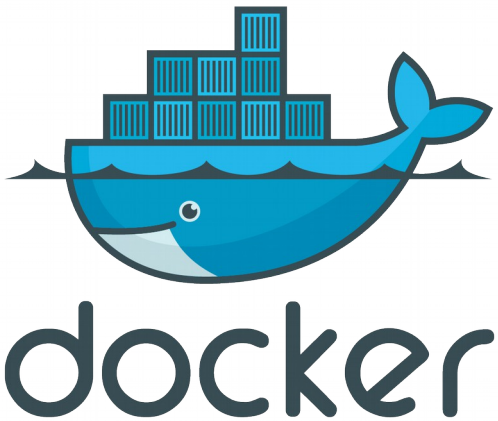
This information provides only general descriptions of the types and portions of workloads that are eligible for execution on Specialty Engines (e.g, zIIPs, zAAPs, and IFLs) ("SEs"). IBM authorizes customers to use IBM SE only to execute the processing of Eligible Workloads of specific Programs expressly authorized by IBM as specified in the "Authorized Use Table for IBM Machines" provided at www.ibm.com/systems/support/machine_warranties/machine_code/aut.html ("AUT"). No other workload processing is authorized for execution on an SE. IBM offers SE at a lower price than General Processors/Central Processors because customers are authorized to use SEs only to process certain types and/or amounts of workloads as specified by IBM in the AUT.

Agenda

What is Docker?

Why are we doing Docker now ?

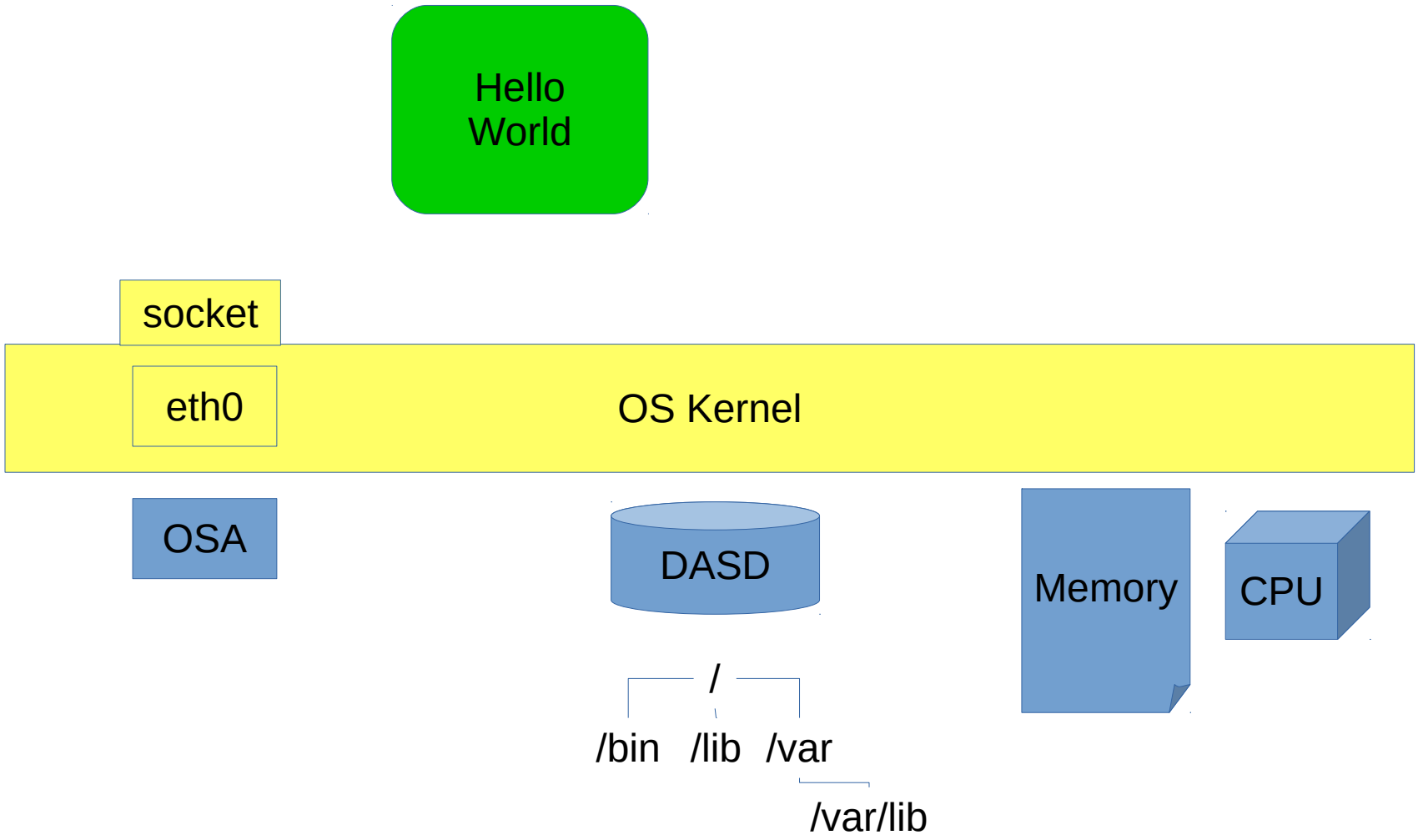
Ok – so how does one run Docker on Linux on Z ?



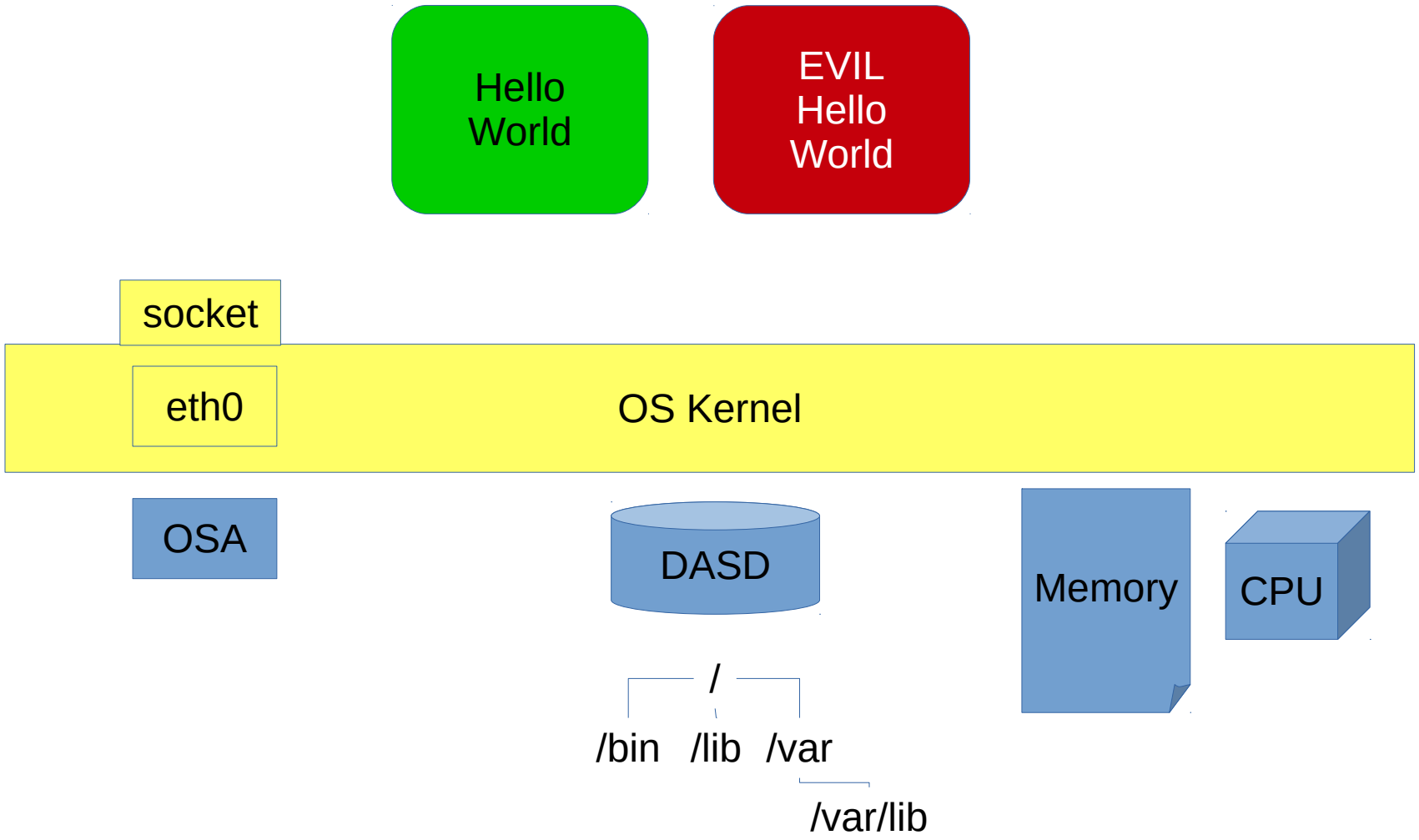
Docker Overview

- Docker is a management tool that handles the construction of a container which provides for all the runtime requirements of an application
- A container is
  - a file system image that contains all the libraries needed to run an application
  - the application itself, which is included in the file system image
  - a union filesystem layer which contains all the writes made to the file system image
  - the specification of what network connectivity the application requires
  - the specification of processor and memory resources that the application requires
- An Image is
  - just a file system inside a file (sorta like an .iso) which can be loopback mounted

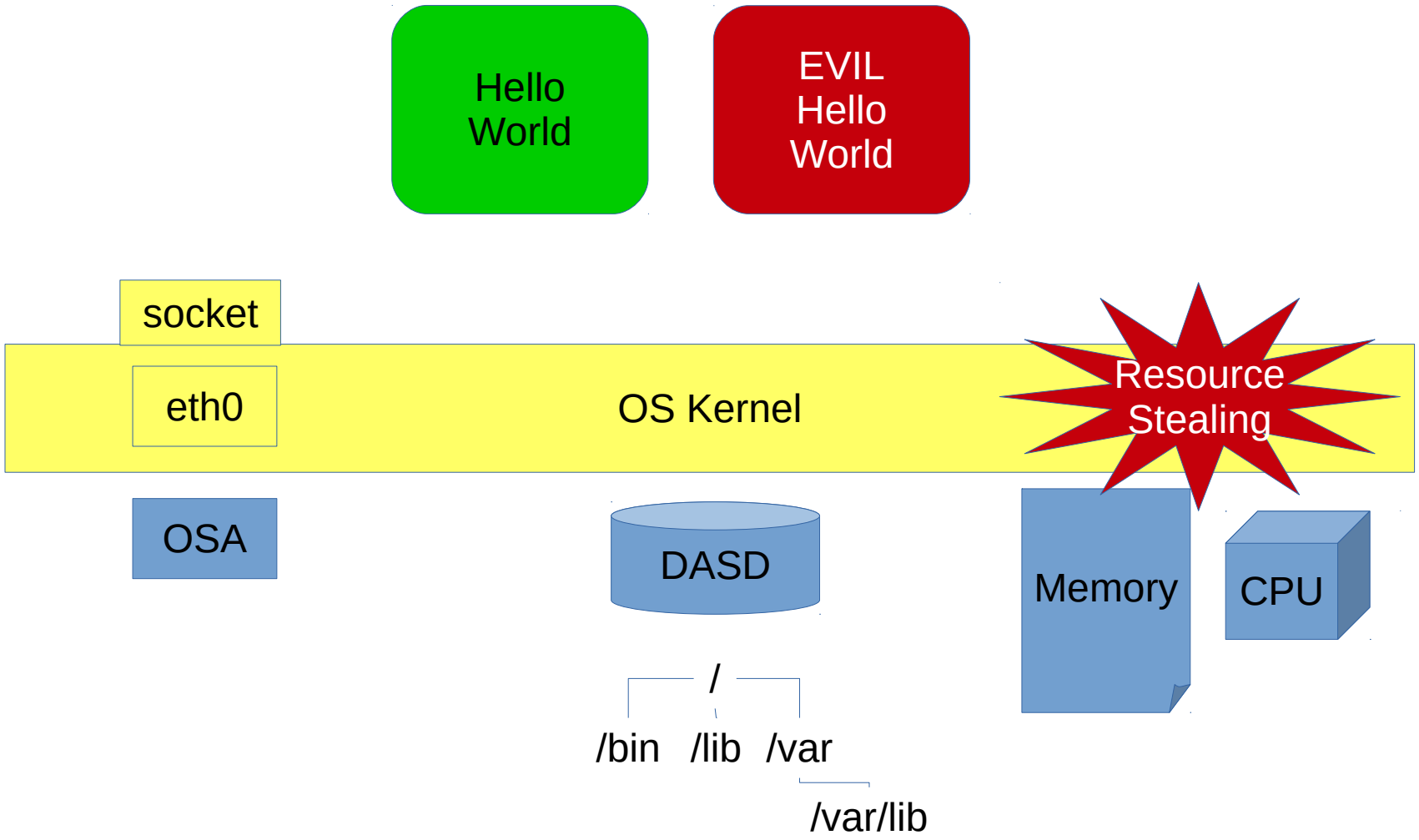
Docker Overview



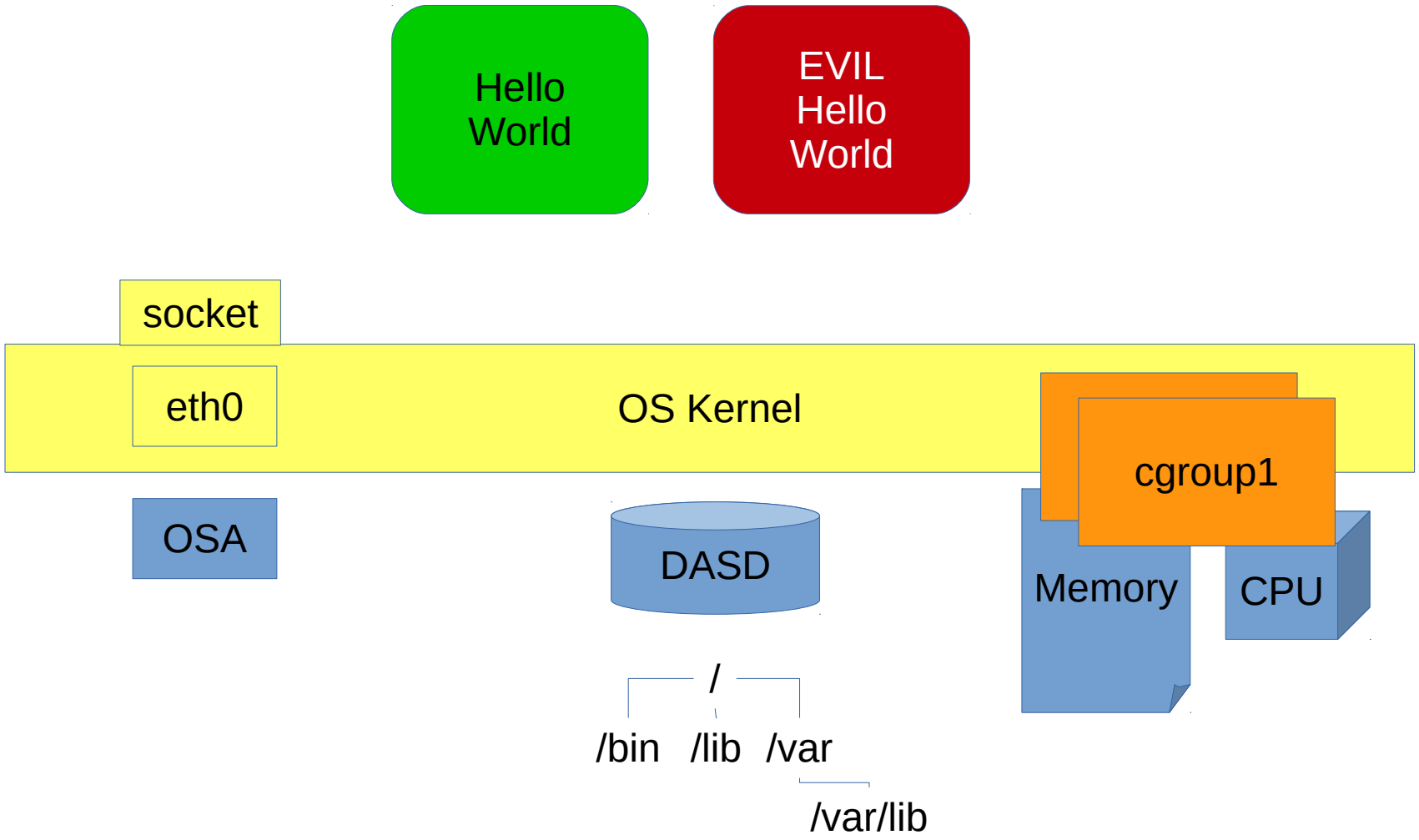
Docker Overview



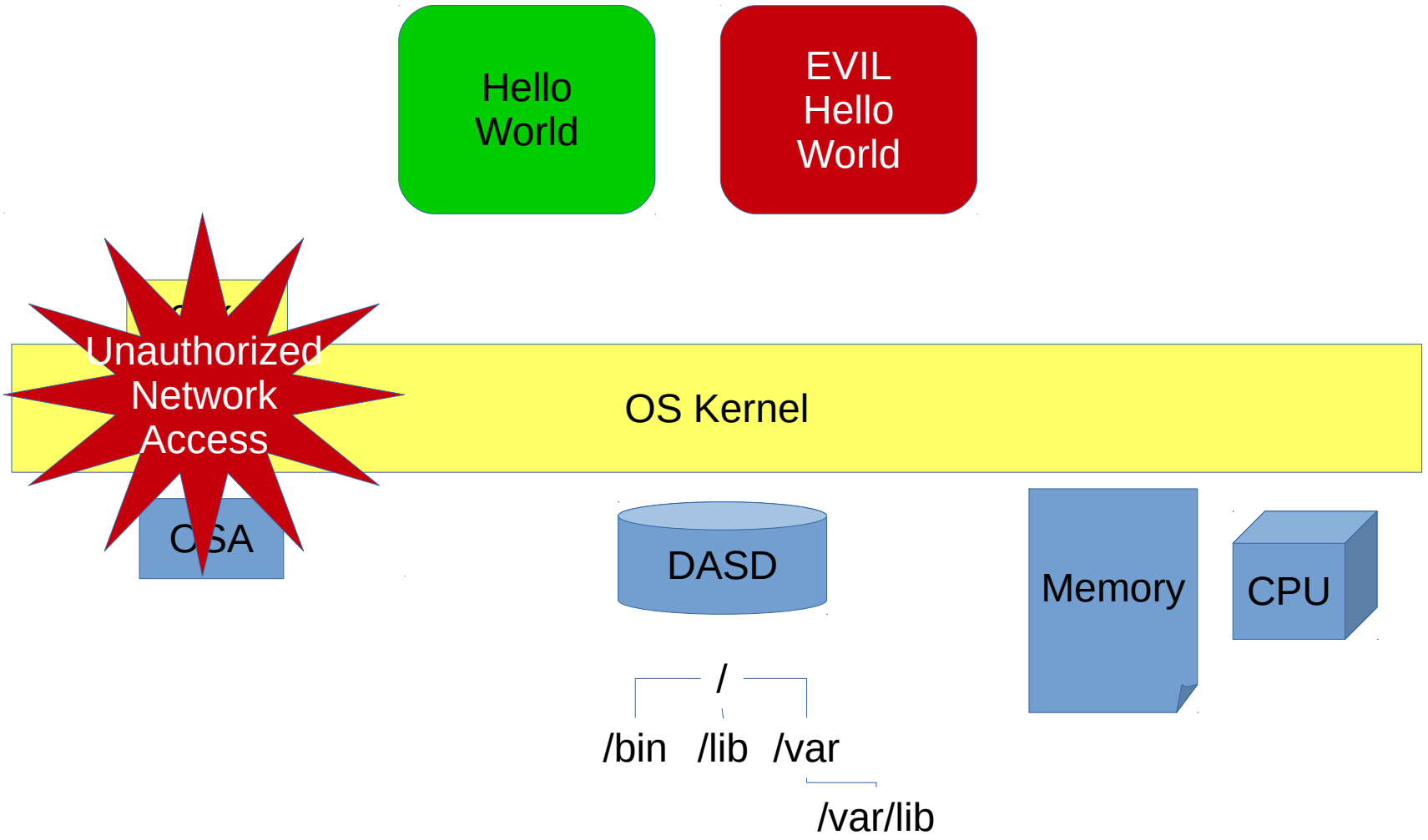
Docker Overview



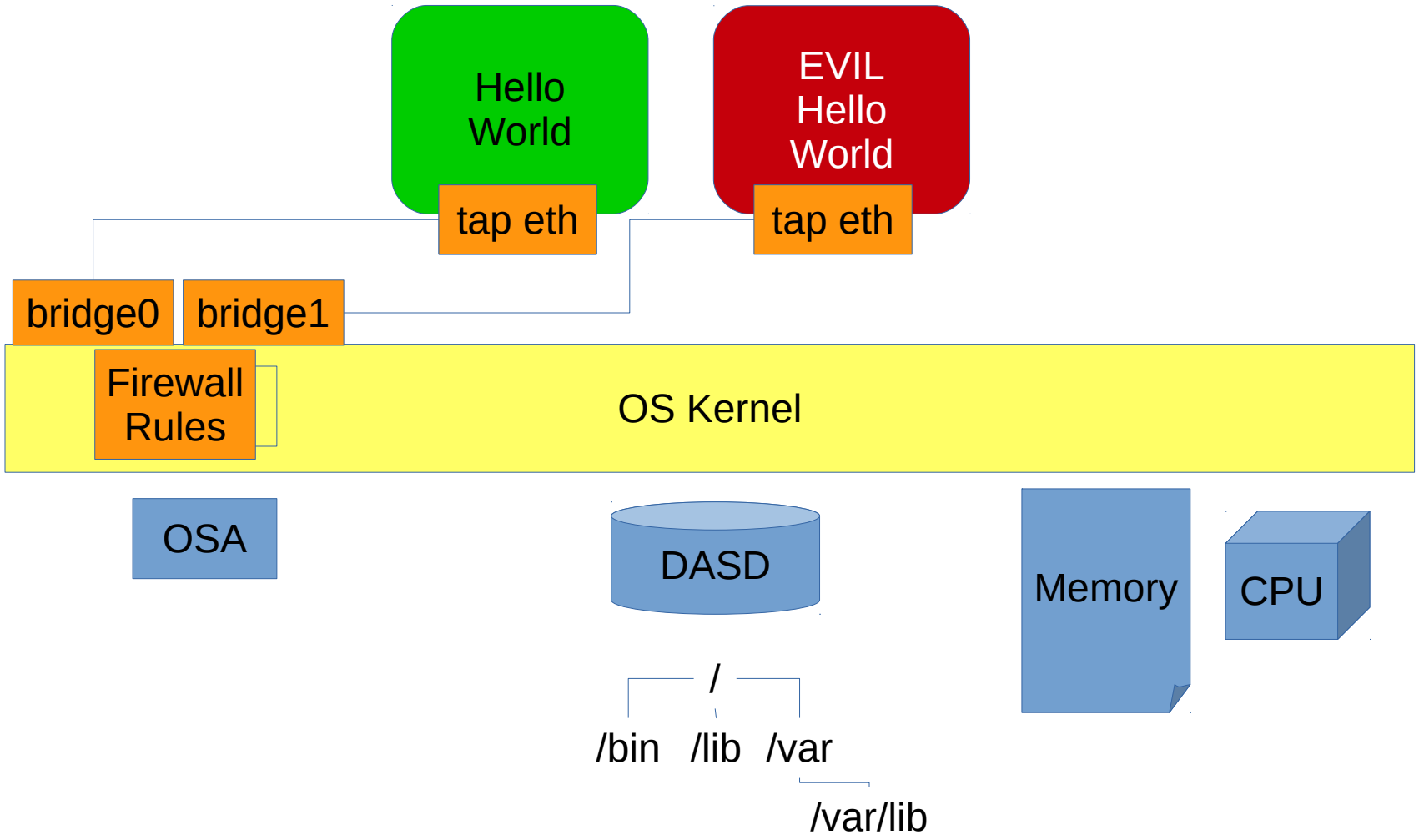
Docker Overview



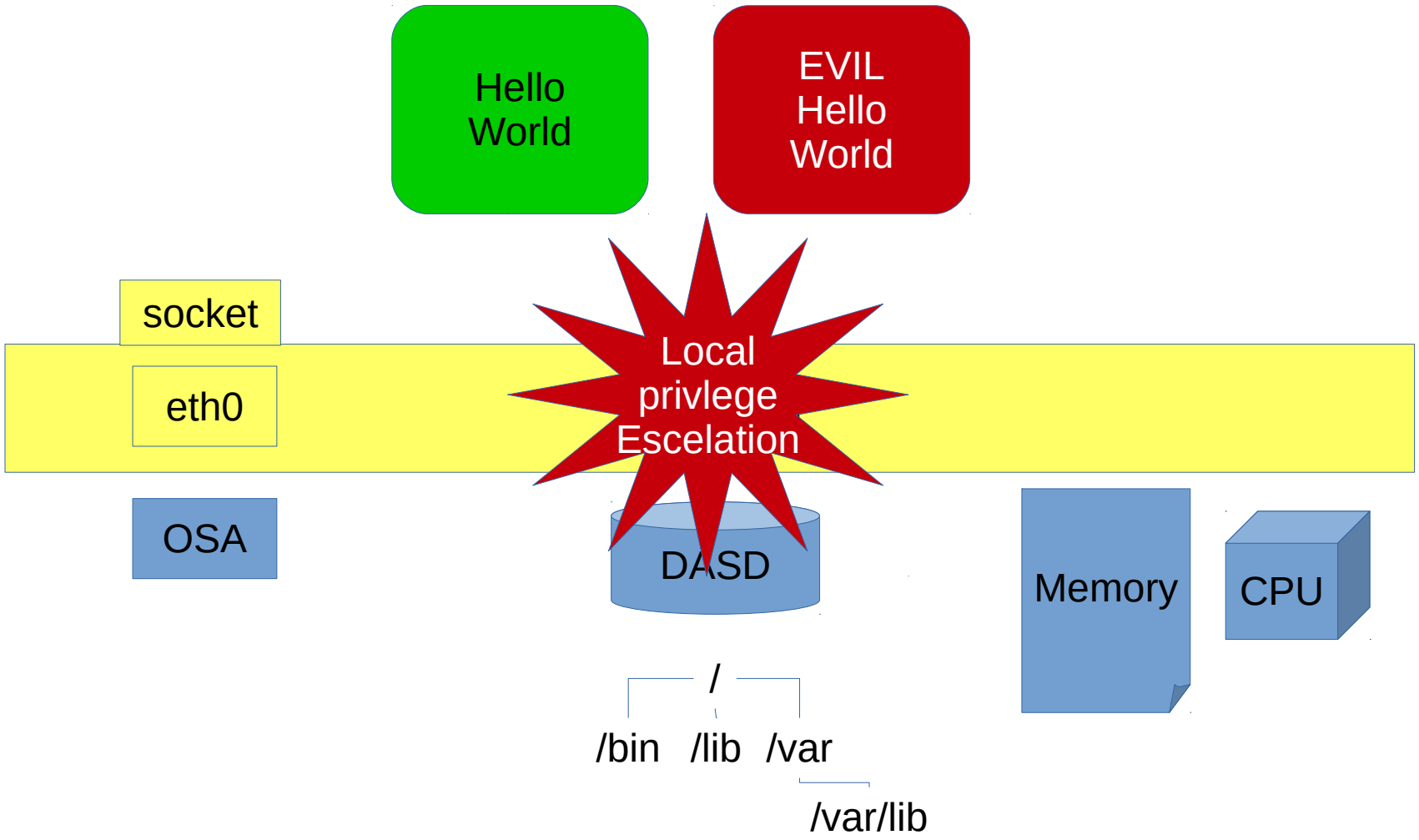
Docker Overview



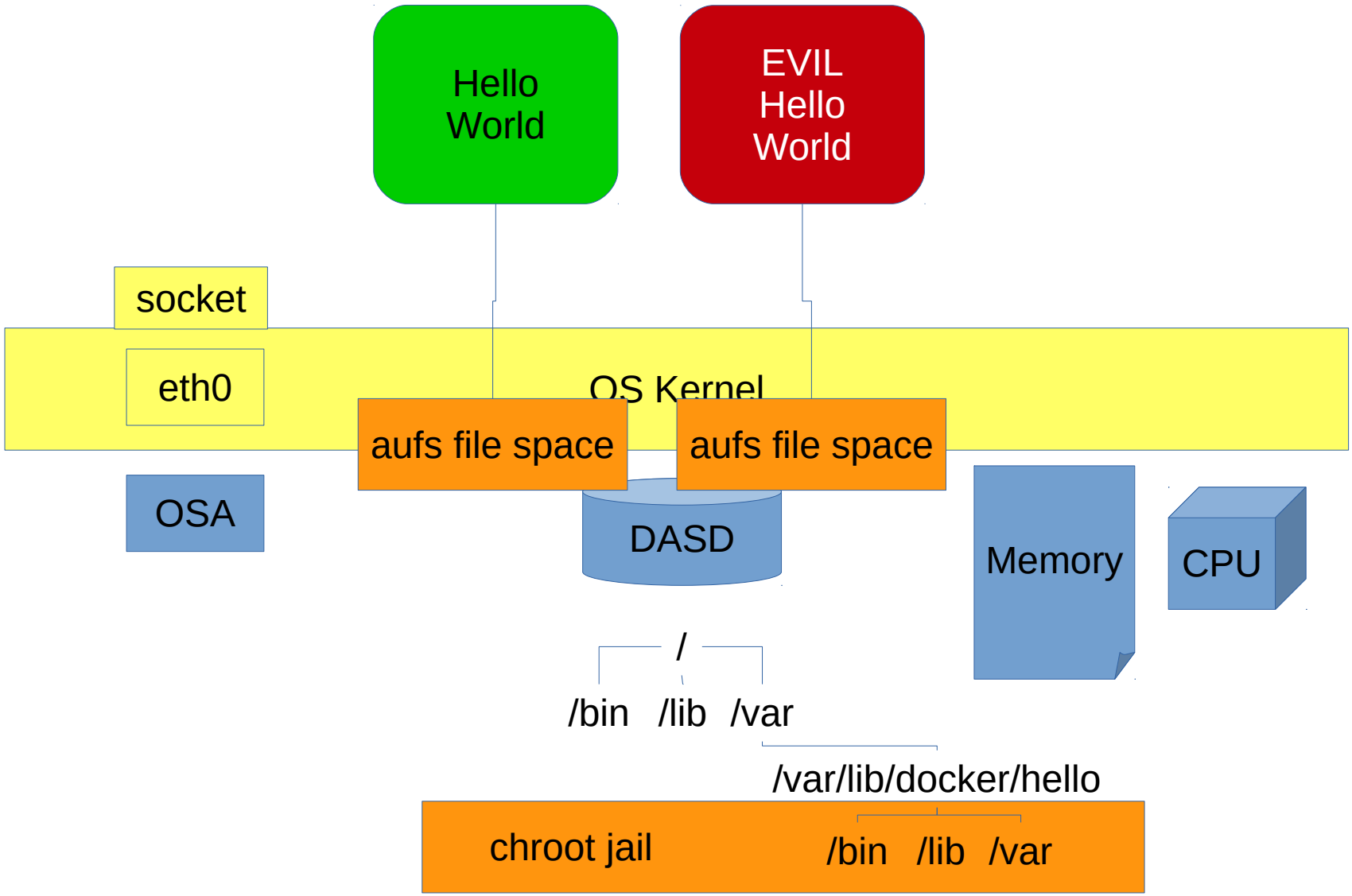
Docker Overview



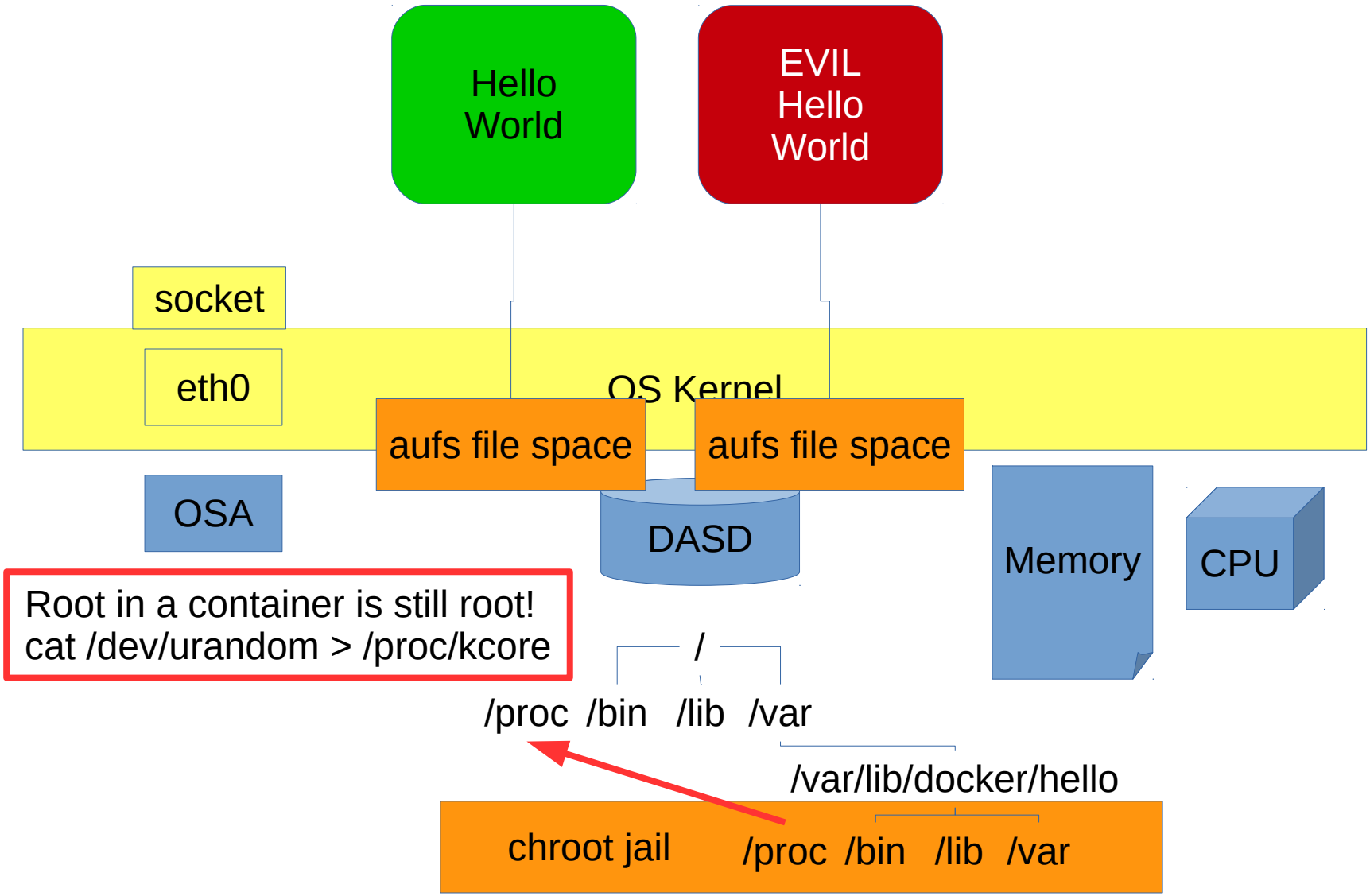
Docker Overview



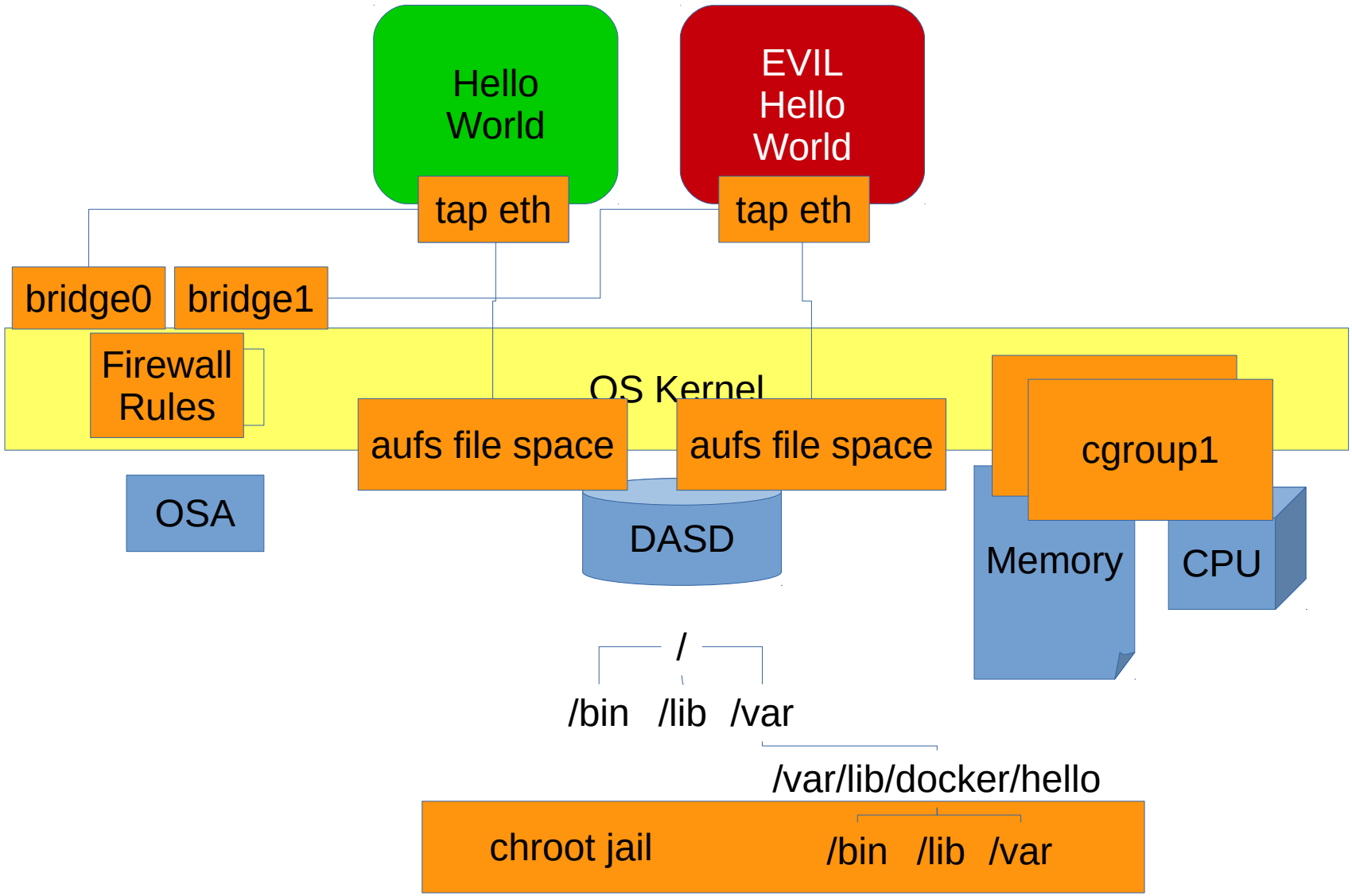
Docker Overview



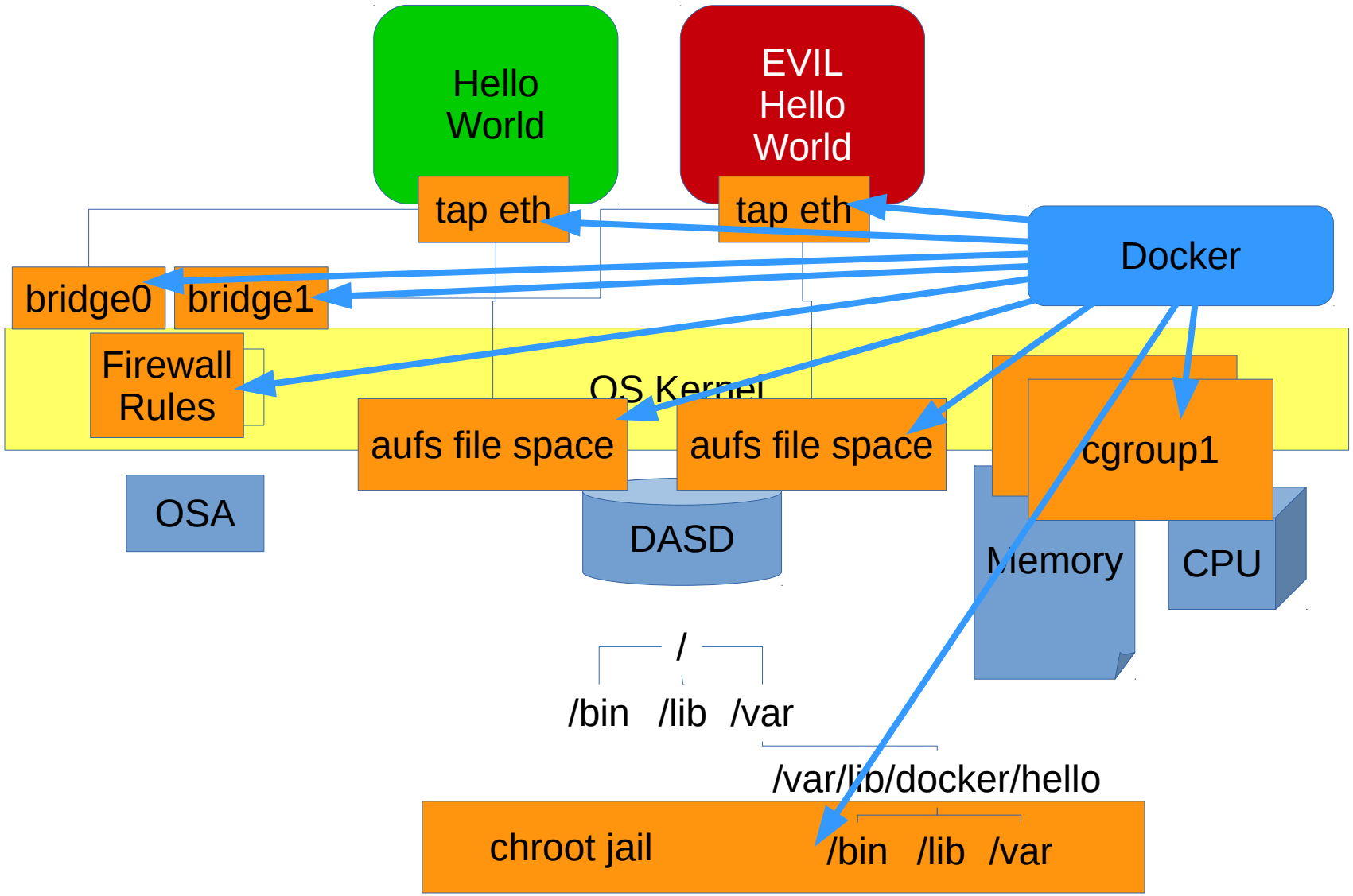
Docker Overview **



Docker Overview

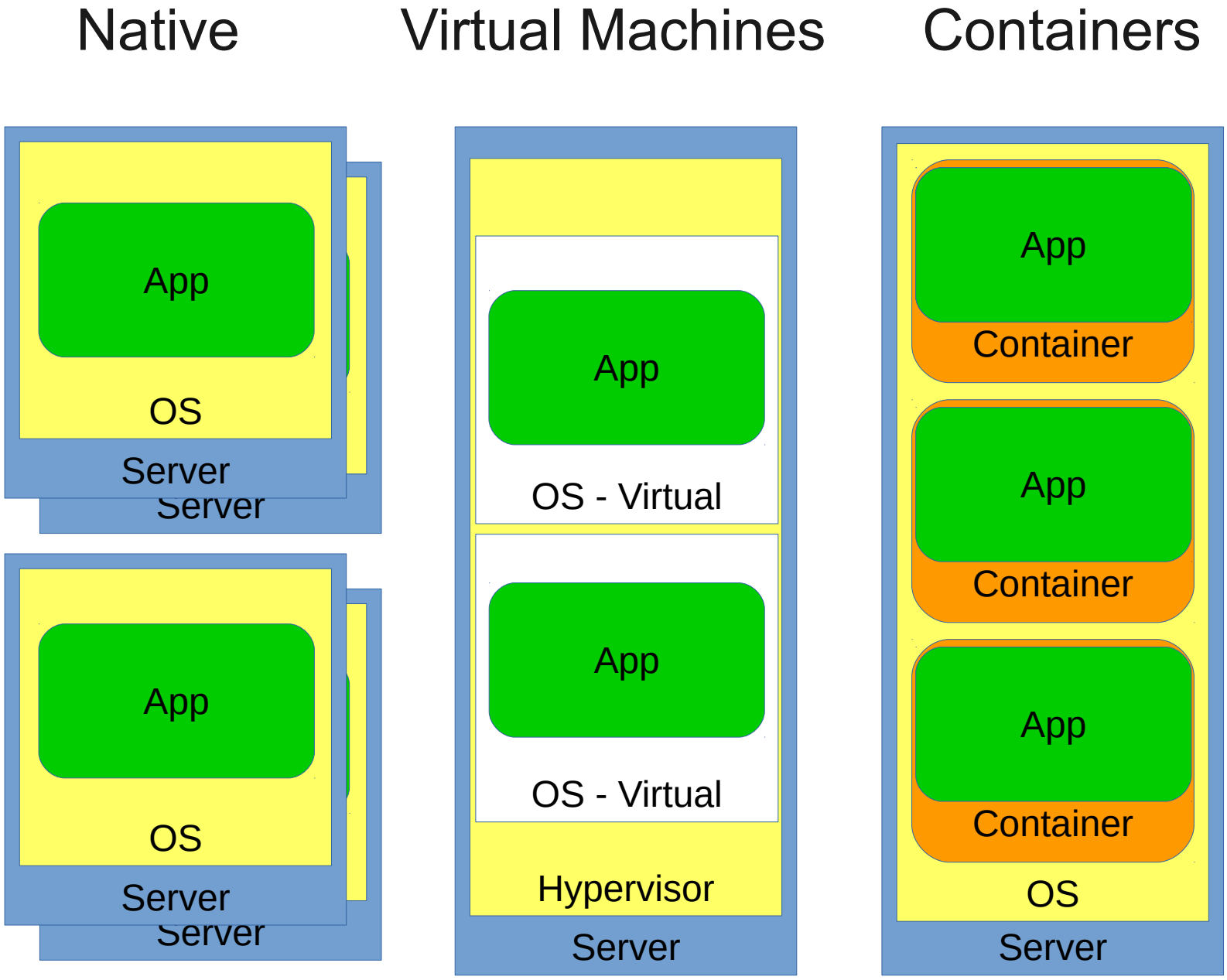


Docker Overview



Docker will perform all the required setup for you

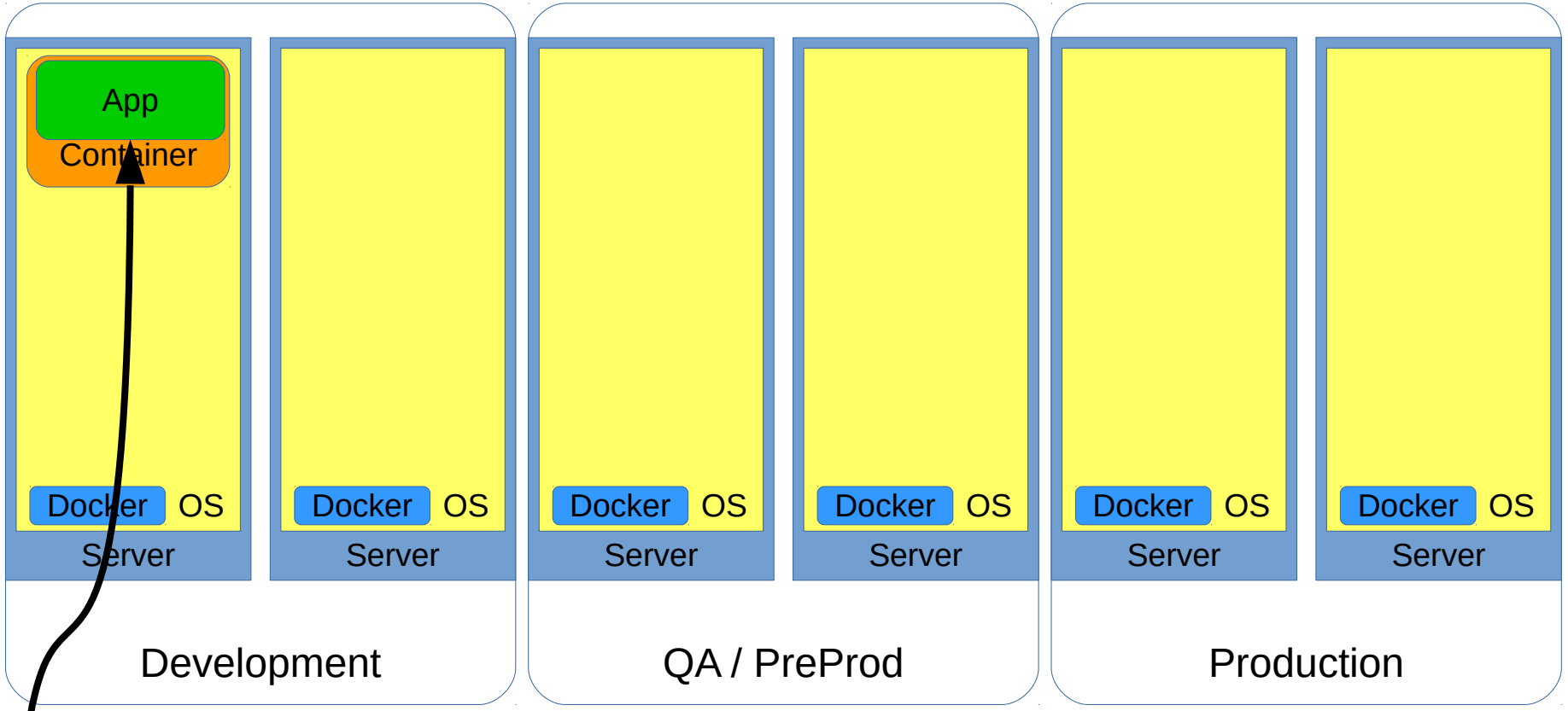
- `> docker pull debian-s390x`
  - Downloads a debian s390x image from docker hub
- `> docker run -it debian-s390x bash`
  - Starts a bash shell within a new container using the debian image
- This new bash shell is contained in a chroot jail
- This new bash shell's writes go to a container specific AUFS layer
- This new bash shell can be limited to a subset of CPU or memory resources using cgroups
- This new bash shell is only permitted outbound NAT access to the network



So what does Docker enable us to do now ?

- Application Development is more automate-able
  - Well suited to a DevOps managed Agile process
- Placing applications is relatively straightforward. There are fewer dependancies between the application and the host that runs it.
  - You can deploy a container to any Docker host of the same processor and operating system family. An application in an image with Debian libraries will run just fine on a Suse host.
  - You still need the right processor architecture though – no deploying x86_64 images on a s390x host.
- Once an image is tested – there is more confidence that the application will work as expected since it brings all its software dependancies along with it.
- Multiple containers can be deployed to a single host, increasing system utilization without the overhead of an additional OS instance to manage.
  - Like Virtualization, but easier
  - Plays to the mainframe's strengths

Docker based development and management



DockerHub
Or
Local Registry

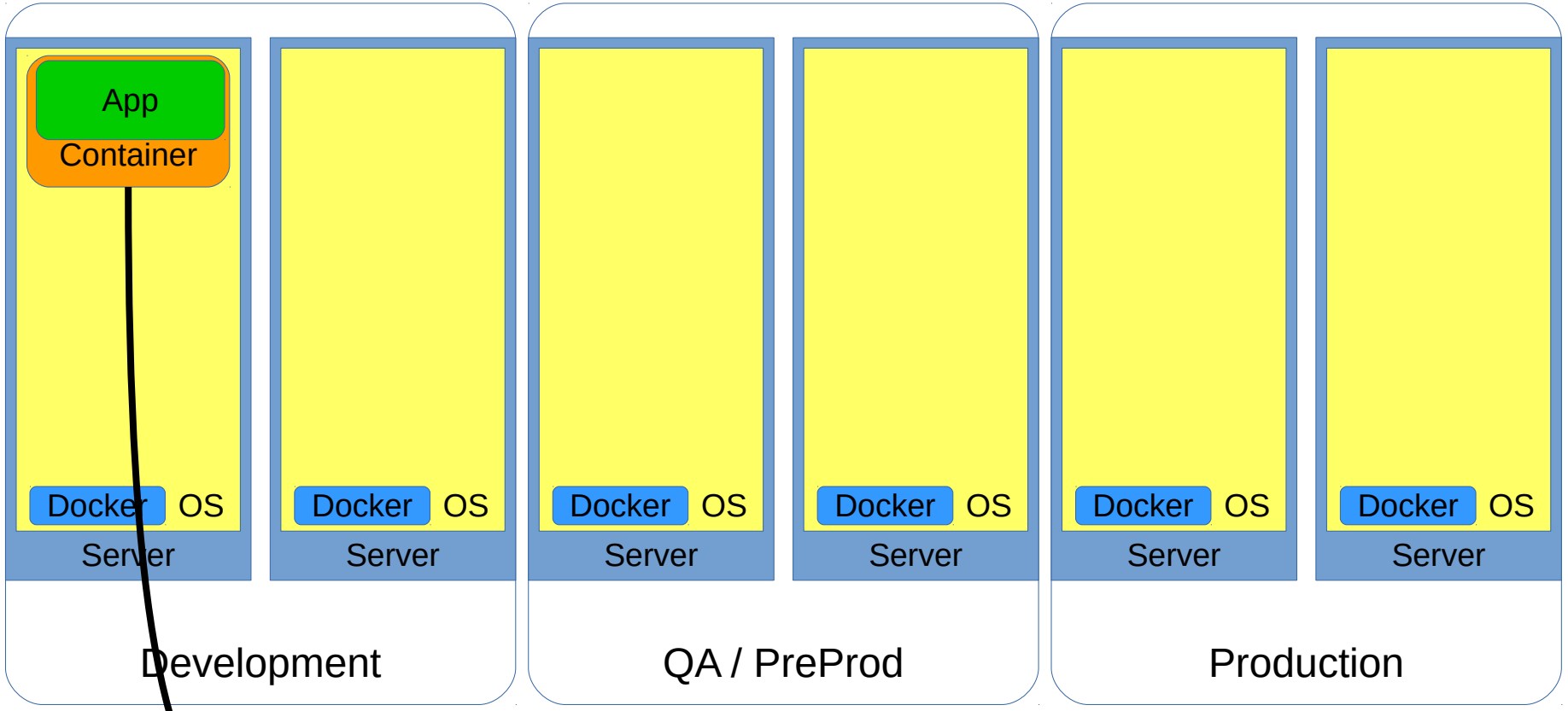
Docker Machine — Install and Manage docker hosts

Docker Swarm — Host clustering and scheduling

Docker Compose — Multi Container orchestration

Developer Develops & Runs Unit Tests

Docker based development and management



DockerHub
Or
Local Registry

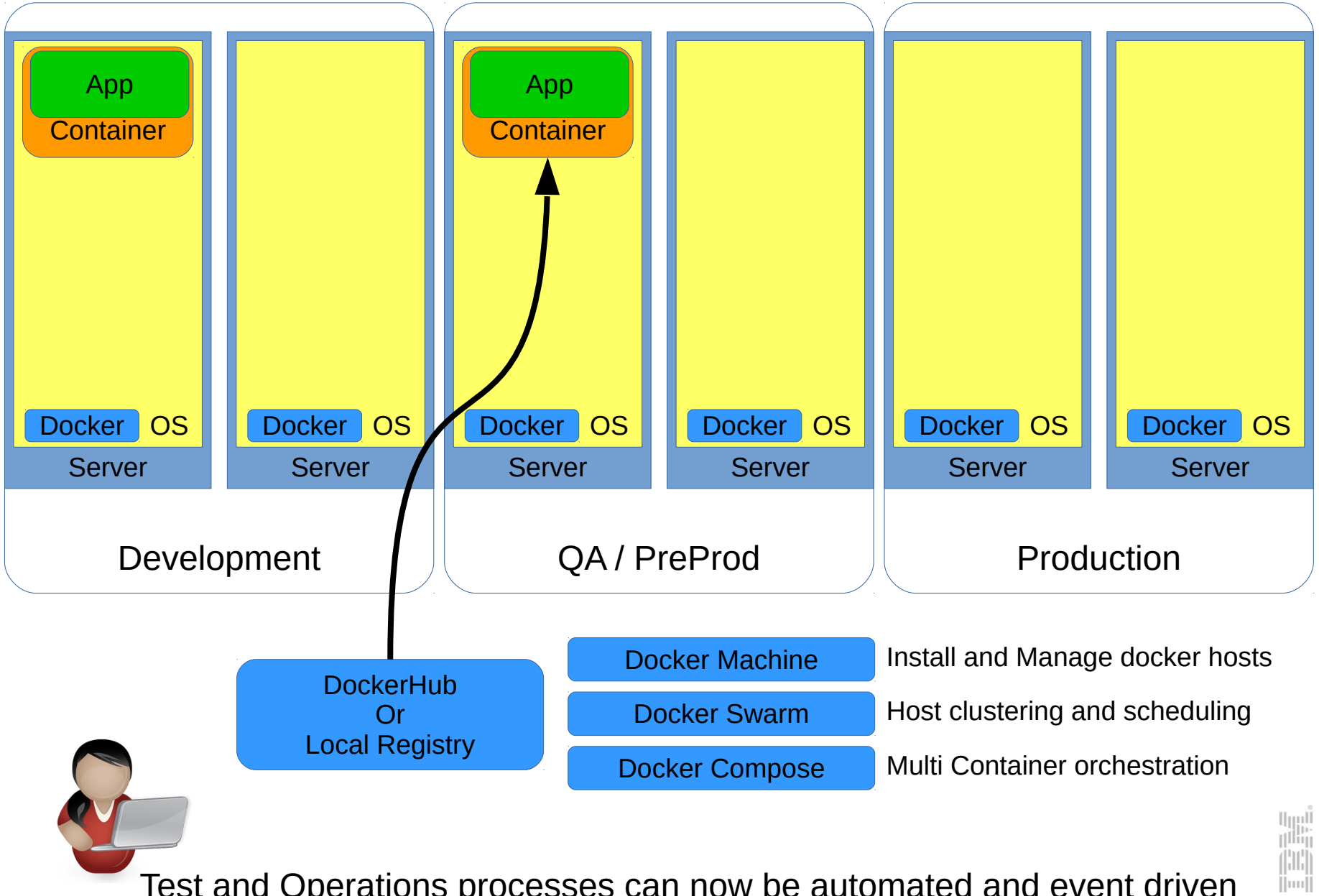
Docker Machine — Install and Manage docker hosts

Docker Swarm — Host clustering and scheduling

Docker Compose — Multi Container orchestration

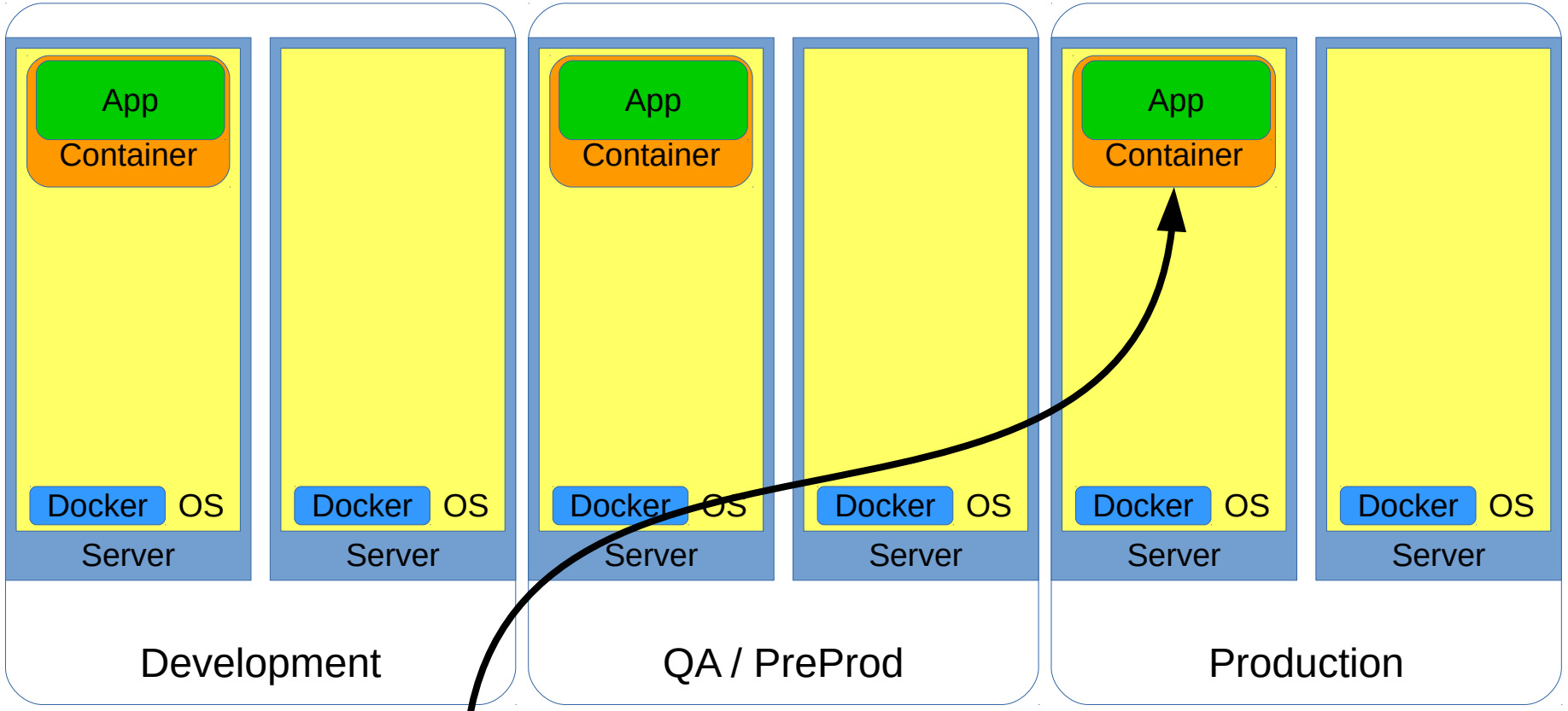
Developer Tags the image & Pushes to Registry

Docker based development and management



Test and Operations processes can now be automated and event driven

Docker based development and management



DockerHub
Or
Local Registry

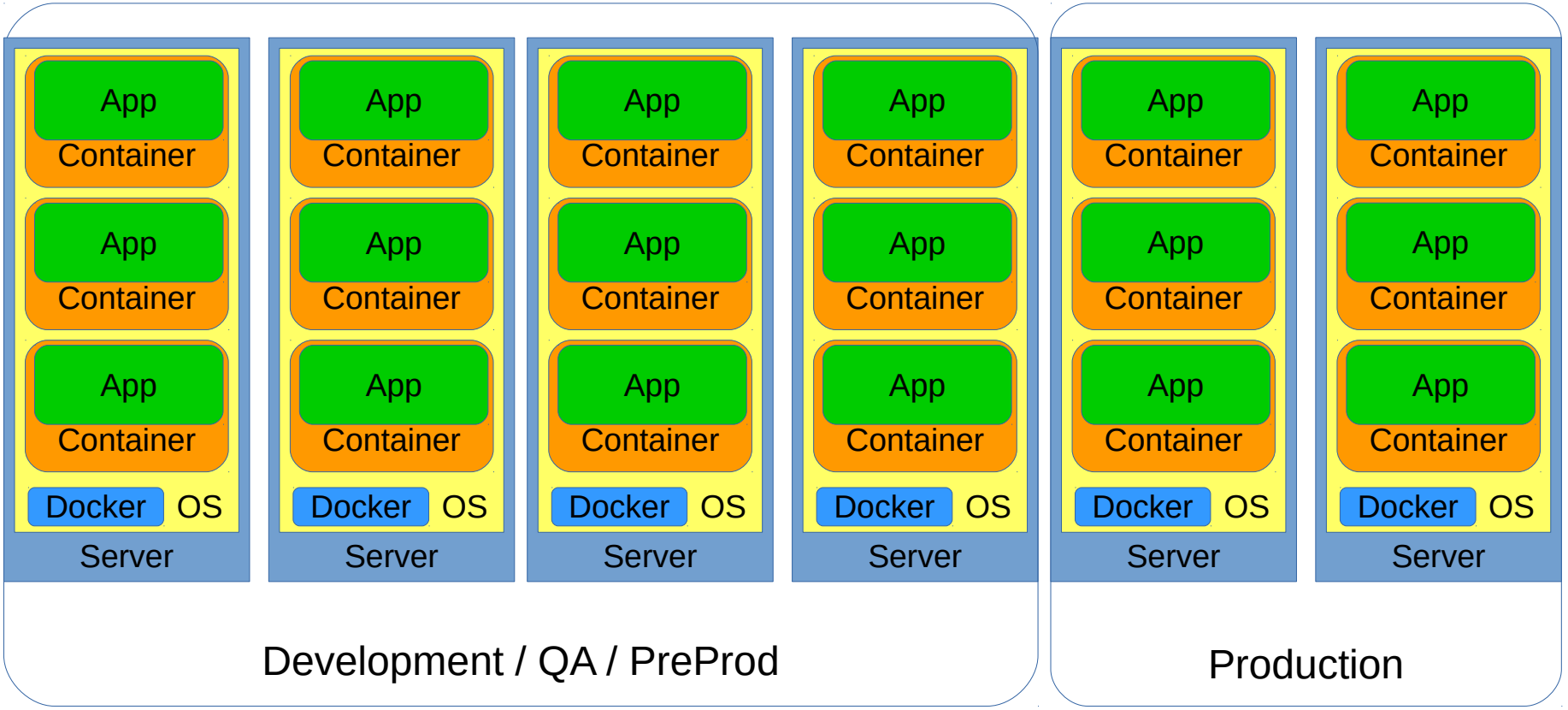
Docker Machine — Install and Manage docker hosts

Docker Swarm — Host clustering and scheduling

Docker Compose — Multi Container orchestration

Applications are deployed to production along with their requisite libraries and and config files

Docker based development and management



DockerHub
Or
Local Registry

Docker Machine — Install and Manage docker hosts

Docker Swarm — Host clustering and scheduling

Docker Compose — Multi Container orchestration

So how do we run Docker on Linux on Z ?

- Download the docker package
  - http://www.ibm.com/developerworks/linux/linux390/docker.html
  - Follow the instructions & make sure you have more than 500MB available in /var
- If you want a local registry, download and install docker on a local x86 system, then start the local registry container
  - `some_x86_machine# docker run -d -p 5000:5000 registry:2`
- Use the mkimage.sh script from a machine with a workable yum / zypper / apt config to create a usable image
  - http://containerz.blogspot.com/2015/03/creating-base-images.html
  - You can add more packages to the image by adding what you're looking for to the yum or zypper install command in the script
  - `imagedir# tar -cf - . | docker import - mynewimage`
- Tag your new image and push it to your local repository for safe keeping
  - `# docker tag mynewimage wherever.the.registry.is.com:5000/mynewimage`
  - `# docker push wherever.the.registry.is.com:5000/mynewimage`
- Start your container adventure!
  - `# docker run -it wherever.the.registry.is.com:5000/mynewimage bash`

Test Lab Notebook

- CPU / Memory performance is more or less native
- Disk Throughput into the image is lower for equivalent CPU consumption
  - ~60% of native speed for Writes: lots of additional codepath, but it is providing value
  - ~85% of native speed for Reads: some additional codepath, but it is providing value
- Disk Throughput into an mapped volume is more or less native
  - Use the -v switch on the docker run command to map a host directory into a container. For example:
    ```
    docker run –it –v /home/bobby:/home ubuntu bash
    ```
- Most Z shops will likely want to use a local image repository so they are not publishing their applications to the public DockerHub portal. Docker also offers private hub cloud services if you want images to be available to multiple orgs or sites.
- You cannot search the local image repository, that support is missing at the moment.
- The Docker daemon will use http_proxy and https_proxy environment variables to get access to the outside world if required, but the squid proxy may not support all the RESTful calls it may make.

Questions?