

ISLET: An Attempt to Improve Linux-based Software Training

Jon Schipp, Hack3rcon V5 (2014)

jonschipp@gmail.com, [@Jonschipp](https://twitter.com/Jonschipp), jonschipp.com

About me:

- Security Engineer for the National Center for Supercomputing Applications



1 PETA =

1 quadrillion = 10^{15} = 1,000,000,000,000,000

Project Contributions:

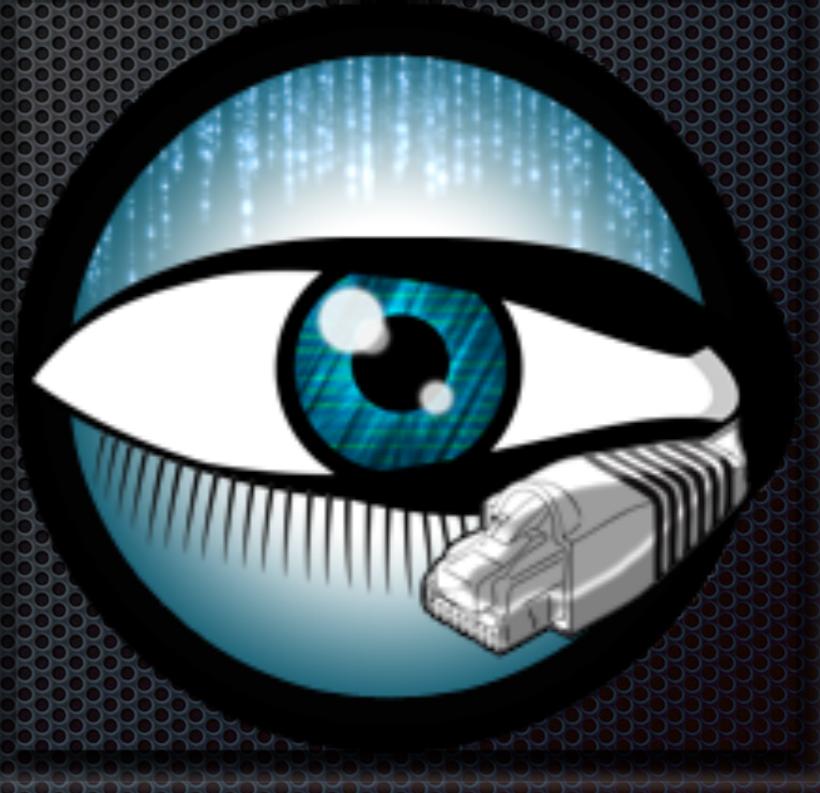
- Netsniff-NG Toolkit

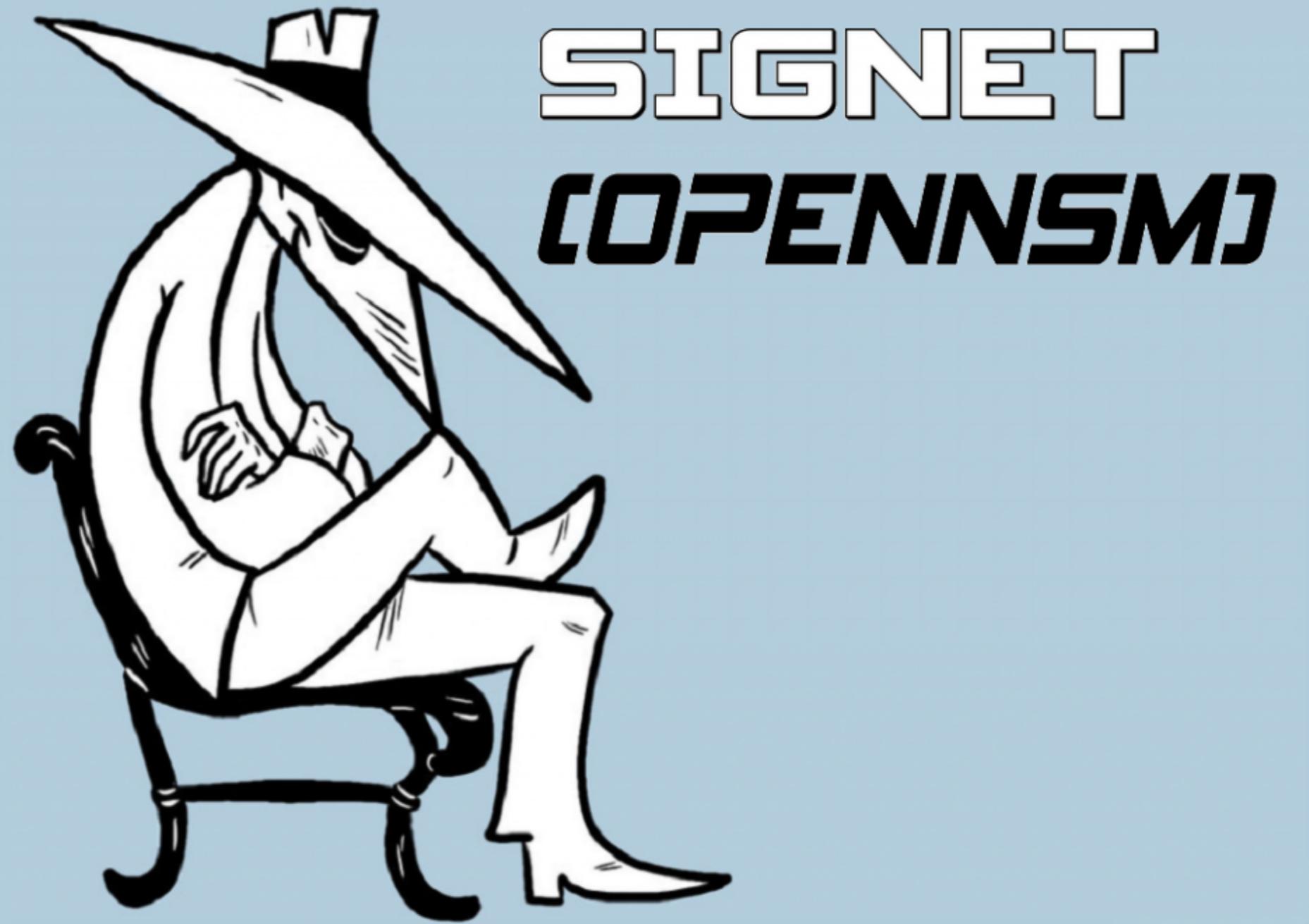


- SecurityOnion



- Bro Team





Open Network Security Monitoring Group
Weekly meetings with presentations, open to all
irrespective of location. <http://opennsm.ncsa.illinois.edu>

The Problem



Problems:

(?) = Who's directly affected?
User/Admin/Everyone

- Too much **time** is spent distributing, downloading, or copying Virtual Machines and other materials.
 - 1.) Conference networks are slow and VM files are big (?)
- **Technical difficulties** can occur and often do that end up putting some behind the group
 - 1.) Hypervisor compatibility e.g. Virtualbox, VMware, etc(?)
 - 2.) VM bus and network configuration
- Account management is repetitive and **time consuming**
- **Changes are not easy**
 - 1.) Insertion of wrong exercises, versions, mistakes, etc.. (?)
How is this handled?

(?)

Solutions:

(?) = Who's directly affected
now?

User/Admin/Everyone

- Avoid passing around or downloading VM's if possible.
Give user's **access to your server**. Big time saver! (?)
- Make barrier to participation as thin as possible
 - 1.) Require only a cross-platform program for access
 - 2.) Opens possibilities to phones, tablets, etc. (?)
- Automated account management (?)
- Changes can be easily completed
 - 1.) Add, remove, or modify exercises during event
 - 2.) Immediately available (?)



Place the burden on the admins
That's where it should go. Users should have a smooth experience. (don't worry, admins will too)

Goal Summary

Admin: make something that is really easy to install, deploy, and configure

User: design an intuitive interface and smooth training experience

ISLET: A Linux-based Software Training System

(I)olated,
(S)calable,
& (L)ightweight (E)nvironment
for (T)raining

Web: <https://github.com/jonschipp/islet>

...Press any key to continue or CTRL+C to exit...

Isolated, Scalable, and Lightweight Environment for Training

A container system for teaching Linux based software with minimal participation effort. The participation barrier is set very low, students only need an SSH client.

ISLET is

- It gets all the components to work together in a way that solves our problems.
- Provides an easy 3-step process for deploying new training environments.



Uses:

Excels at teaching command-line software on Linux

- Event training (intended design)
- Internal software training
- Capture the flag competitions
- Trying out tools in a containerized environment
- Development environments

Use Cases:

- Launched the precursor at BroCon 14.
Used to teach Bro scripting. ~50 users had shell access to Bro and unix tools in a container simultaneously on a lower end (m3.xlarge) EC2 VM, no problem.
- Unnamed consultant teaches NSM software at various conferences across the world with it.
- Used to teach OSSEC at an OpenNSM meeting
- Used to teach Linux tools at UIUC LUG meetings

Feedback loop

Container logs show user's actions e.g. mistakes
which can be used to improve future training

Technical Part

Compare

Contrast

Remote Container Training

Multiple training environments per user

Scalable

Immediate revisions

Low Participation Barrier

Server maintenance

Remote Access (e.g. ssh)

Isolation

Shell
Access

Remote Host Account Training

Account Management

Privileged training limitations

Single training
environments
per user

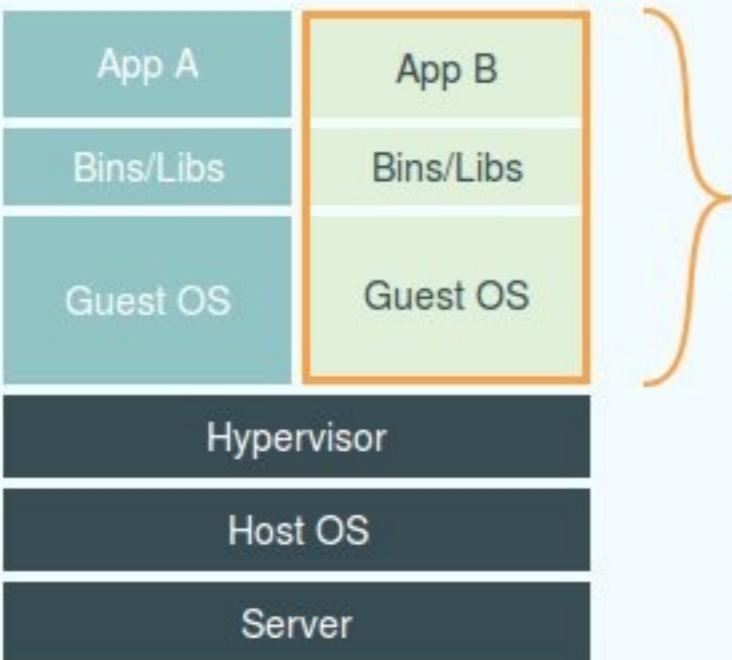
Local Virtual Machine Training

Effortful

Hypervisor configuration and compatibility

Possession of training environment

VM Distribution time



Virtual Machines

Each virtualized application includes not only the application - which may be only 10s of MB - and the necessary binaries and libraries, but also an entire guest operating system - which may weigh 10s of GB.



Docker

The Docker Engine container comprises just the application and its dependencies. It runs as an isolated process in userspace on the host operating system, sharing the kernel with other containers. Thus, it enjoys the resource isolation and allocation benefits of VMs but is much more portable and efficient.

Isolation via kernel namespaces and cgroups The user can't tell the difference.

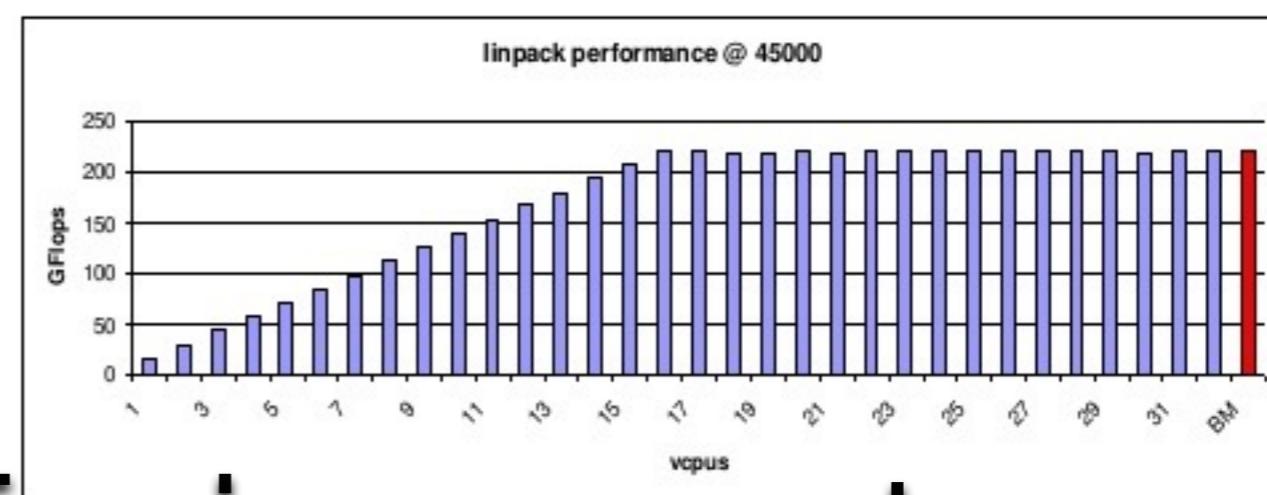
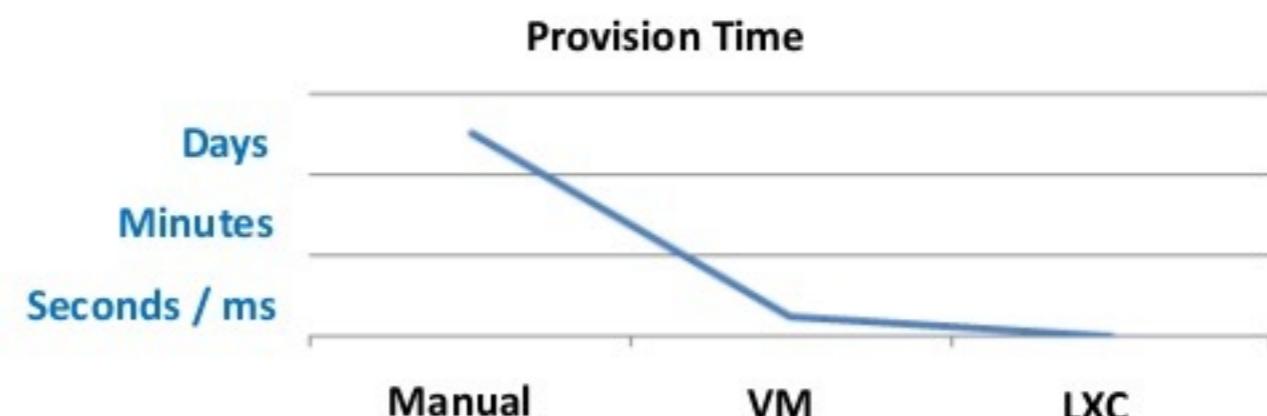


Scalability

Could virtual machines teach all these metal heads?
You be the judge. 1000+ containers is possible

"Linux Containers as poised as the next VM in our modern Cloud era..."

- Provision in seconds / milliseconds
- Near bare metal runtime performance
- VM-like agility – it's still “virtualization”
- Flexibility
 - Containerize a “system”
 - Containerize “application(s)”
- Lightweight
 - Just enough Operating System ([JeOS](#))
 - Minimal per container penalty
- Open source – free – lower TCO
- Supported with OOTB modern Linux kernel
- Growing in popularity



Research Moment

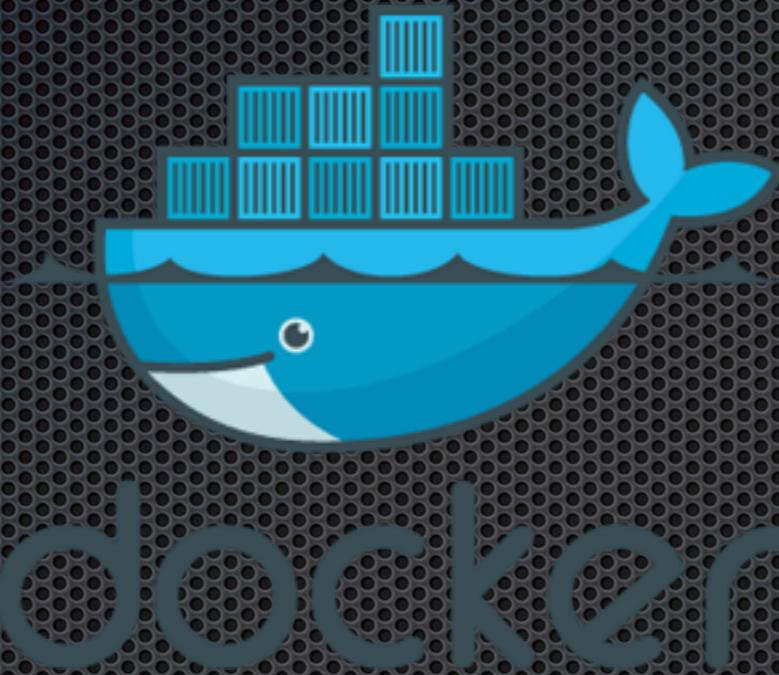
“In general, Docker equals or exceeds KVM performance in every case we tested.”

– *IBM Research Report: An Updated Performance Comparison of Virtual Machines and Linux Containers*

< [http://domino.research.ibm.com/library/cyberdig.nsf/papers/0929052195DD819C85257D2300681E7B/\\$File/rc25482.pdf](http://domino.research.ibm.com/library/cyberdig.nsf/papers/0929052195DD819C85257D2300681E7B/$File/rc25482.pdf) >

Linux Containers?

Docker popularized the technology.
It's actually been around for 7 years.



- Automates the deployment of Linux based container
- Provides layers of abstraction
- Various methods of container creation
- Docker hub and registries for sharing and deployment

Containers

- **Important:** "Linux Based Containers"
There is no container specification
- There are different container (and like) technologies
 - Linux: LXC, OpenVZ, Google containers, etc.
 - Non-Linux: BSD Jails, Solaris Zones, AIX WPAR, etc.
- What do containers do?
Light-weight process virtualization
- What do virtual machines do?
Hardware virtualization

Linux Kernel Stuff

- **Support:** 3.8 introduce the building block for containers
Namespaces: Process isolation
Currently available: *pid, net, ipc, uts, mnt, and user*
Control Groups: Resource management
e.g. `cpu, cpuset, blkio, memory`
- It's not magic, you can create namespaces and cgroups directly from your shell by modifying procfs and sysfs. That's how they were deployed before userland tools like LXC and Docker existed.

Container Security

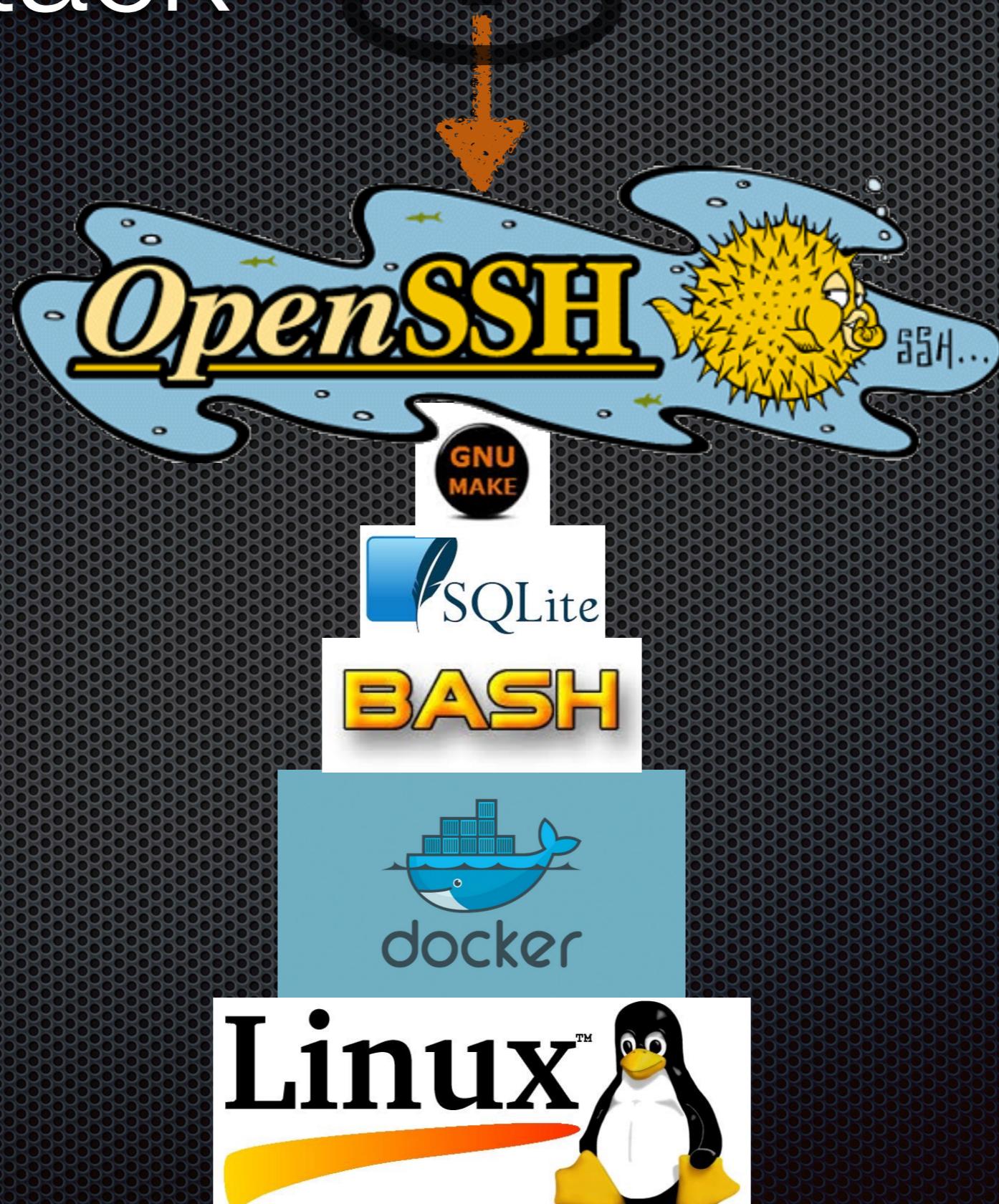
- Networking can be disabled in containers to prevent attacks against other hosts, containers, or self
- Don't run processes as root, drop user to standard user
- System resources are limited per container to prevent selfishness and abuse
- Control CPU and RAM allocation
- Containers and users are automatically removed after a period of time. Typically the length of conference or event
- Containers which get too large are automatically removed to prevent disk space abuse or denial of service. Devicemapper can also limit disk space.
- Finer environment controls via ulimit for Docker processes: fsize, nproc, etc.

Host Security

- grsecurity patches
- sshd hardening - e.g. disable port forwarding, X11 forwarding, etc.
- iptables rate limiting to prevent excessive connections
- permissions and ulimit
- Protect resources

Working with ISLET

The Stack



Platform

- Linux only (currently), 64-bit only (because of Docker)
- Debian family is preferred but will work on others that support Docker. I do nearly all of my testing on Ubuntu so I recommend using that for the best experience.

```
$ git clone  
http://github.com/jonschipp/islet
```

Configuration & Workflow

- Global: configuration file i.e. /etc/islet/islet.conf
- Per-image: configuration files per image /etc/islet/*.conf
- Creating a new training environment (3 steps)
 1. Build Docker image with the software for training
 2. Create configuration file and point to new image
 3. Place in /etc/islet/ e.g. new_image.conf

Other things

- Ability to port forward to do things like web interfaces for training where each container is running a web interface that only the user is accessing. Requires user to have a unique IP address (e.g. no PAT device)
- Direct X11 works if user is running an X server and *trusts* connections from the ISLET server. You could train GUI applications using this method.
`export DISPLAY="192.168.1.100:0" && wireshark`

Easy updates

\$ make update

Demo

“F*** it, we'll do it live”

– *Bill O'Reilly*

Hack3rcon Special:

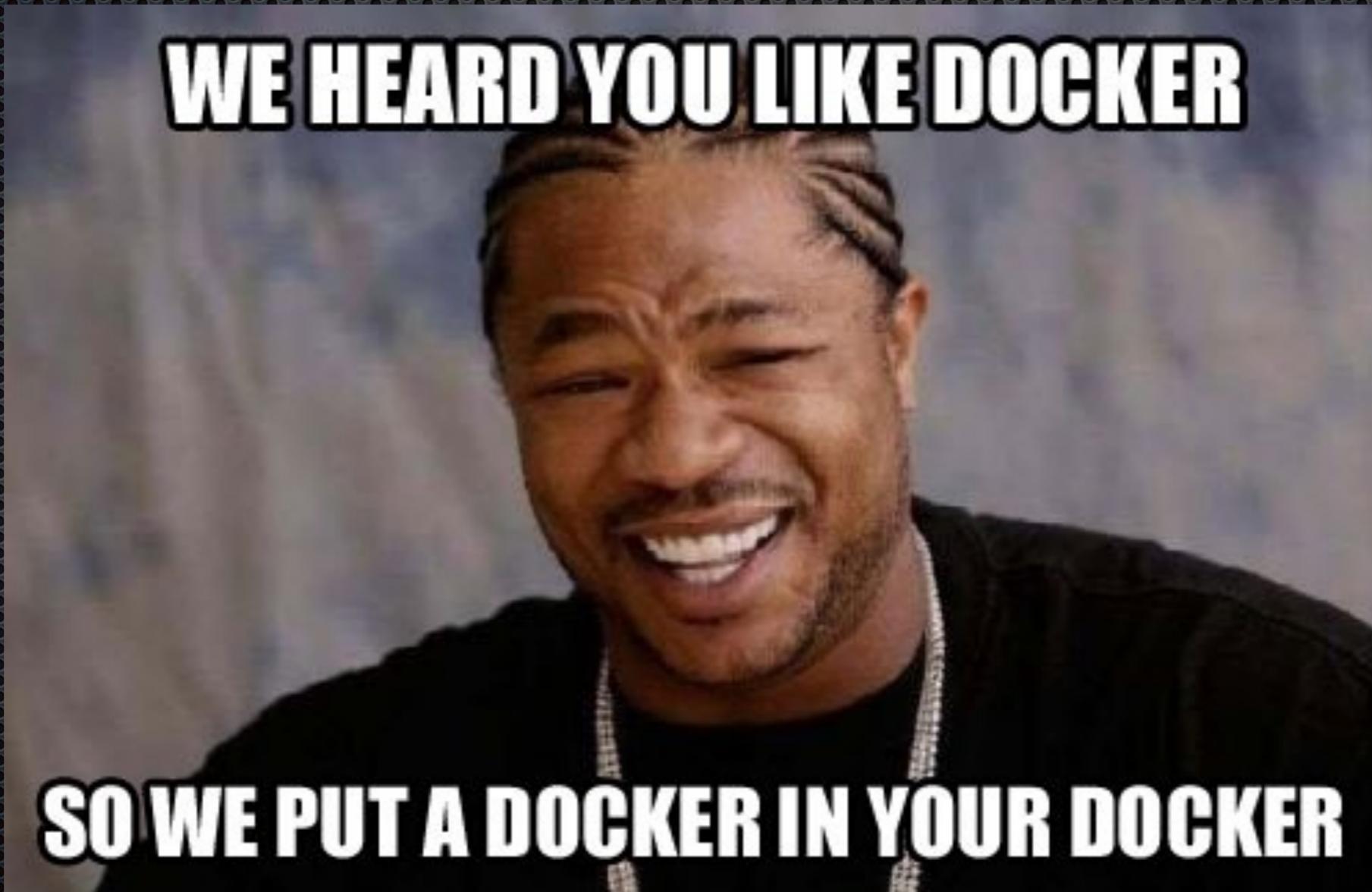
```
$ ssh demo@islet1.jonschipp.com
```

```
$ ssh demo@islet2.jonschipp.com
```

Password: **hack3rcon**

Roadmap:

- Run ISLET from a container
 - 1.) No modification to host system



Roadmap:

- Publish paper
- Port to FreeBSD using jails
- Security/penetration testing
- Autoconf scripts
- POSIX compliance (?)
- Export containers and exercises
 - 1.) Users can save their work or continue at home like they can with VM training
- Scalability testing beyond 1000+ containers, distributed setup

Contributing

Send me pull requests,
patches, feature
requests, and issues.

Use it and tell me what
you think!



Testing & Dev

```
$ git clone http://github.com/jonschipp/vagrant
$ cd vagrant/islet
$ vagrant up
$ ssh -p 2222 demo@127.0.0.1
```



Communication

- jonschipp@gmail.com
- <http://jonschipp.com>
- [@JonSchipp](https://twitter.com/@JonSchipp) #ISLET #NoMoreVMs
- [keisterstash](#) on freenode

References:

- IBM Research Report: An Updated Performance Comparison of Virtual Machines and Linux Containers < [http://domino.research.ibm.com/library/cyberdig.nsf/papers/0929052195DD819C85257D2300681E7B/\\$File/rc25482.pdf](http://domino.research.ibm.com/library/cyberdig.nsf/papers/0929052195DD819C85257D2300681E7B/$File/rc25482.pdf) >
- Realizing Linux Containers (LXC): Building Blocks, Underpinnings, and Motivations < <http://www.slideshare.net/BodenRussell/realizing-linux-containerslxc> >
- Resource management: Linux kernel Namespaces and cgroups. < <http://www.haifux.org/lectures/299/netLec7.pdf> >
- Linux Containers and the Future Cloud. < http://www.haifux.org/lectures/320/netLec8_final.pdf >
- Lightweight Virtualization with Linux Containers (LXC) < <http://www.ciecloud.org/2013/subject/07-track06-Jerome%20Petazzoni.pdf> >
- www.docker.com
- <https://images.google.com>