Mitigate complexity in cloud microservices analysis

Swiss army knife tool for live container analysis

Cisco CSIRT
Fabio Nigi
FIRST TC April 20, 2015
Last login: Thu Apr 20 16:30:00 on ttyfirst2016
$
$whoami
Fabio Nigi / @fnigi / fnigi@cisco.com
$sudo su –
Cisco csirt
$finger
Member of the winner pool of global Docker hack 2015
“Never upgrade a server again. Never update your code. Instead, create new servers, and throw away” Jerome Petazzo, Docker Tinkerer Extraordinaire

```
FROM ubuntu:14.04

RUN apt-get update && 
    apt-get install -y curl openjdk-7-jre-headless
```
Introduction to the technology Docker from the OS stack to minimal infrastructure design

Image source: https://docs.docker.com/engine/understanding-docker/


Image source: https://cloud.docker.com/engines/dn/ending-docker
Serve a website: architectural approach: with or without micro-services

- Linux - OS
- Apache - Web
- MySQL - DB
- PHP - Script

Orchestration at scale w/Docker
**Overriding Dockerfile image defaults**

When a developer builds an image from a `Dockerfile` or when she commits it, the developer can set a number of default parameters that take affect when the image starts up as a container.

Four of the Dockerfile commands cannot be overridden at runtime: `FROM MAINTAINER`, `RUN`, and `ADD`. Everything else has a corresponding override in `Docker run`. We’ll go through what the developer might have set in each Dockerfile instruction and how the operator can override that setting.

1. **CMD (Default Command or Options)**
2. **ENTRYPOINT (Default Command to Execute at Runtime)**
3. **EXPOSE (Incoming Ports)**
4. **ENV (Environment Variables)**
5. **VOLUME (Shared Filesystems)**
6. **USER**
7. **WORKDIR**

---

**Bypass security without warning**

*Overriding Dockerfile image defaults*
Design and feature

- Injecting tools into running container
- Network transparent or 9pfs client server
- Browsing container FS
- Scriptable - easy to use in bash scripts
- Live analysis without trace (removing all after)
How to run and execute the cloud micro services analysis:

1) setup a tools container

$docker build -t csk-tools Dockerfiles/debian-tools

2) setup a csk container

docker build -t csk.

3) enhance desired container with your tool and follow the instruction to use it!

docker run -ti -v /var/run/docker.sock:/var/run/docker.sock csk <container to enhance>

4) hack it!

5) remove tools from a container

docker run csk --remove <container to enhance>
Usage & cases

System debug
File comparison (hashing)
Compromised applications testing
The tool: https://github.com/docker/global-hack-day-3/tree/master/container-swiss-knife