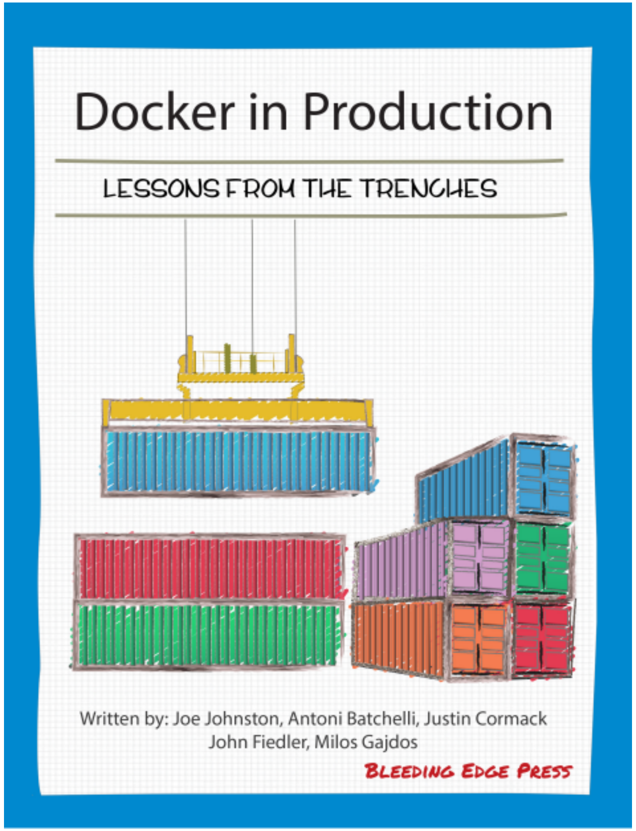
DevOpsDays London: Let's talk about Security

Justin Cormack

Justin Cormack

Cambridge based developer at Docker @justincormack

Co-author of Docker in the Trenches: Successful Production Deployment



Docker in Production

LESSONS FROM THE TRENCHES

Written by: Joe Johnston, Antoni Batchelli, Justin Cormack
John Fiedler, Milos Gajdos

BLEEDING EDGE PRESS

Let’s talk about Security

Security “NO!”

A Conversation

Ops “please, developers, can you write secure code?”

Devs “please, ops, can you secure the environment for our code?”

DevOps “This service needs to do these things and access these other services”

DevOps “Ok, I will restrict its access in test and production to those actions”

To get to this we need a domain specific manifest of types of actions a program can do, and a way to restrict it to just these.

We want defence in depth – a single way of imposing restrictions only needs a single circumvention.

Examples

Android permissions and intents were a good early model

- Certainly conversational...
- Good apart from the bit where the user clicks “Allow”

`pledge(2)`

- System call to reduce ability to do things, grouped into different classes
- stdio rpath wpath cpath dpath tmppath fattr flock inet dns unix sendfd recvfd proc getpw tty ioctl prot_exec exec settime ps vminfo id pf audio

```
if (pledge("stdio rpath wpath cpath", NULL) == -1) {
  perror("pledge");
  exit(2);
}
```

Usability

- Within 6 months it had been introduced to over 400 programs
- Not a typical coding community, true
- Not the sole means of defence, adds defence in depth.
- There are only 8000 SELinux profiles on github after 18 years, and most are the same ones.

Doesn’t Apply to Me

- Probably you are not writing Unix commands for OpenBSD
- Very domain specific rules eg exactly which files can be read
- Some of the specifics are less of a concern
- However, microservices are modelled on the Unix process model

Content Security Policy for Web Applications

- Content headers for browsers limiting actions, defines none, urls or local only, or similar
- default-src script-src object-src style-src img-src media-src frame-src font-src connect-src form-action sandbox script-nonce plugin-types reflected-xss report-uri
- http://w3c.github.io/webappsec-csp/
- Creating a CSP Policy from Scratch

```
Content-Security-Policy
    "default-src 'none';
    script-src 'self' https://www.google-analytics.com/;
    style-src 'self' https://fonts.googleapis.com;
    font-src 'self' https://fonts.googleapis.com https://fonts.gstatic.com;
    frame-src 'self' https://www.slideshare.net;
    upgrade-insecure-requests; block-all-mixed-content;
    reflected-xss block; referrer no-referrer-when-downgrade;
    frame-ancestors 'none'; form-action 'none';
    base-uri diogomonica.com www.diogomonica.com;
    report-uri https://report-uri.io/report/59e303e8e117668e8e166508913a6d1d;"
```

Containers

Docker supports lots of security mechanisms

- Namespaces, capabilities, SELinux, Apparmor, seccomp, iptables, networks (Linux likes different security subsystems)
- The defaults are really good, and work for almost everyone
- Containers are a very secure environment to run code.

Not so friendly

```
{"name": "accept4", "action": "SCMP_ACT_ALLOW", "args": []},
deny @{PROC}/sys/kernel/{?,??,[^s][^h][^m]**} w,
docker run --cap-drop=sys_admin
```

Next steps

- Make the customisation easier for your use cases
- Increase uniformity
- Correlate the different types of option, so set different options in lockstep

Types of role for microservices

- Client, server, or both
- Connects to specified hosts outside local network
- May not connect to certain types of host (finance, production)
- Must use encrypted connections to these hosts
- Document clear contracts about what is allowed

Summary

- First talk about what your application needs to do
- Human readable and understandable
- Machine readable, testable and debuggable.
- Declarative
- Domain specific

Talk!

Questions?

- @justincormack
- justin.cormack@docker.com
- `docker pull justincormack/devopsdays2016`