CrashLoopBackoff, Pending, FailedMount and Friends

Debugging Common Kubernetes Cluster and Application Issues

About Me

In IT since my first job helping out with computers in my high school in 1994

Past employers: CoreOS, Red Hat, Electronic Arts among many others

Currently a Senior Consultant for OTEEMO GET IT DONE (we're hiring!)

Blood type: Caffeine-positive

Contact info:

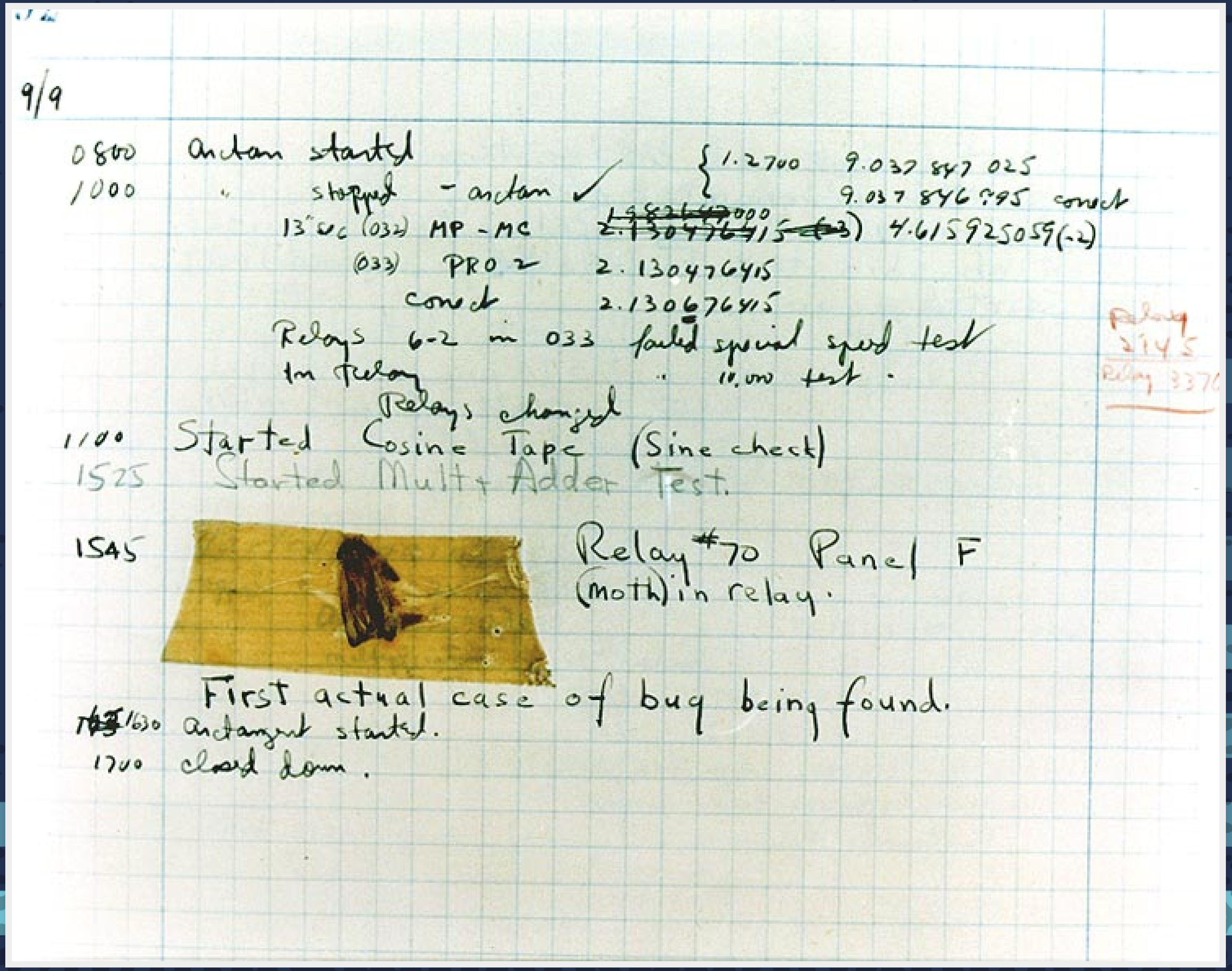
- jthompson@oteemo.com
- Twitter: @caffeinepresent
- Kubernetes Slack: @kensey
- LinkedIn

First thoughts

None of this is rocket science, it's just a new rocket engine

Most of it isn't even really new -- we're just probing the state and outputs of the system. The only new things are:

- *Some* of the tools
- *Some* of the parts you probe



9/9

0800 Anctan started {1.2700 9.037 847 025
1000 " stopped - anctan ✓ 9.037 846 995 const
13" uc (032) MP - MC ~~2.130476415~~ (-3) 4.615925059(-2)
(033) PRO 2 2.130476415
const 2.130676415
Relays 6-2 in 033 failed special speed test
in relay " 11.000 test .
Relays changed
1100 Started Cosine Tape (Sine check)
1525 Started Mult + Adder Test.
1545 Relay #70 Panel F
(moth) in relay.
First actual case of bug being found.
1630 Antangent started.
1700 closed down.

Relay 2145
Relay 3370

Image: the "first bug" log page written by Grace Hopper in 1945
Credit: Wikimedia Commons

Your application deployment just failed

Take a deep breath...

- Don't panic
- Find the Little Book of Calm
- Assorted other advice from classic works of fiction

Now let's fix it

- *Gather info*
- Form a plan
- Test and execute

Gathering info: tools and techniques

Some first steps

Get the lay of the land...

```
kubectl get [-o wide] <nodes, pods, svc...>
kubectl describe [-o yaml] <node, pod, svc...>
```

Often you will spot the issue right here

Let's talk about...

Image credit:
matuska@pixabay

kubectl get events

Provides a summarized view of recently-seen events, e.g:

```
NAMESPACE   LASTSEEN   FIRSTSEEN   COUNT
NAME                                          KIND
SUBOBJECT   TYPE      REASON          SOURCE
MESSAGE
default     6s          6d          39910
data-romping-buffoon-elasticsearch-data-0   PersistentVolumeClaim
            Normal    FailedBinding   persistentvolume-controller
no persistent volumes available for this claim and no storage
class is set
```

kubectl logs

- Note: no `get`

Gets logs from a container in a pod:

```
<probe> INFO: 2017/11/14 21:55:43.738702 Control connection to
weave-scope-app.default.svc starting
<probe> INFO: 2017/11/14 21:55:45.789142 Publish loop for
weave-scope-app.default.svc starting
```

- If the pod has multiple containers, container must be specified too with `-c`

Container logs

Good old-fashioned SSH followed by interacting with the system logs or container runtime

"Didn't we just do that with `kubectl`?" -- If you have really bad cluster issues or you're debugging an issue with a control-plane component, you might not be able to use kubectl

Debugging containers

- SSH to host
- Run a container in an existing container's namespace

"Why?!"

- If the application doesn't provide good logs, or doesn't know what issues it's encountering, at least you can interrogate its environment
- If the host itself has issues and lacks the usual tools, this is often safer and quicker than trying to install them permanently on the host

What to look for

Know what "normal abnormal" behavior looks like

```
<probe> WARN: 2017/12/08 07:39:07.762765 Error collecting weave status,
backing off 10s: Get http://127.0.0.1:6784/report: dial tcp 127.0.0.1:6784:
getsockopt: connection refused
<probe> WARN: 2017/12/08 07:39:07.767862 Cannot resolve 'scope.weave.local.':
dial tcp 172.17.0.1:53: getsockopt: connection refused
<probe> WARN: 2017/12/08 07:39:07.816447 Error collecting weave ps, backing
off 20s: exit status 1: "Link not found\n"
```

In my cluster, this is *not a problem* when I deploy Weave Scope because I don't have Weave networking deployed

Cluster networking issues

- Did you deploy a cluster network?
  - If you did, are the pods for it starting correctly?
  - Are the expected interfaces showing up on the host?
- Are firewalls preventing packets from flowing between hosts?

(Note: no deep dive here, because the topic is vast…)

Pod startup issues

- Are your pods getting scheduled?
- Are your pods starting? If not, why not?
- Are your pods starting but crashing?
  - Container pull/startup issues?
  - Init container failing?
  - Readiness/liveness probes failing?
  - Running out of resources (not only actual usage, but requests)?

Service discovery issues

```
<probe> WARN: 2017/12/08 17:49:35.260659 Cannot resolve
'kubecon2018.default.svc': lookup kubecon2018.default.svc on
10.3.0.10:53: no such host
```

- Cluster DNS issues?
  - Typos in service names?
  - Deployed in wrong namespace?
  - kube-dns not healthy?
- Do your services have endpoints? If not, why not?

Access control issues

```
<probe> WARN: 2017/12/08 07:56:19.104684 Error Kubernetes reflector (pods),
backing off 40s: github.com/weaveworks/scope/probe/kubernetes/client.go:195:
Failed to list *v1.Pod: pods is forbidden: User "system:serviceaccount:default:
kubecon2017-weave-scope" cannot list pods at the cluster scope
<probe> WARN: 2017/12/08 07:56:19.106268 Error Kubernetes reflector (nodes),
backing off 40s: github.com/weaveworks/scope/probe/kubernetes/client.go:195:
Failed to list *v1.Node: nodes is forbidden: User "system:serviceaccount:
default:kubecon2017-weave-scope" cannot list nodes at the cluster scope
```

- NetworkPolicy preventing traffic?
- RBAC preventing reading resources?
  - Need to create a role/service account/binding?
  - Often an issue with things that manage Kubernetes itself or use it for discovery
- TLS issues?

Hey, so how do I fix it?

Usually that's the easy part: Kubernetes is declarative, so just redeclare things correctly:

```
kubectl apply -f ...
```

Don't forget to fix things *before* you clean up old pods/etc. Kubernetes does a lot of cleaning up for you -- don't make work for yourself

Preventive and remedial measures

Application design

- Does your application log diagnostic info? Does it do so *correctly*?
  - BAD: "lp0 on fire"
  - Yes, I'm harping on logs again
  - Does it log *enough*? Are you *sure?*
  - Write more detailed logs anyway
- Does your application have diagnostic *tools*? Do you document them?
- How safe is data and state in case of app failure? Can your application roll back? Have you tested that?

Pre-deployment

- Automate, automate, automate
  - Preaching to the choir, I know
- Factor out redundancy -- repeating yourself is error-prone
- Your environment should not only support both of the above, they should be the obvious path of least resistance -- look at Helm, Draft, etc. for automating/templating app deployments
- Consider the Cluster Autoscaler or other methods of auto-scaling
- Test environments are not optional

Post-deployment

- Application validation tests -- make sure your successful deployment was a *correct* deployment
- Ongoing monitoring -- but avoid "alert fatigue" by choosing your alert conditions well
  - Make sure *your* conditions make sense for *your* environment
- Adopt the chaos monkey (see here or here) -- you show me a server with high uptime and I'll show you a server with unpersisted state

Where you can get more help

Kubernetes Docs

API docs: https://kubernetes.io/docs/api-reference/v1.8/ -- great for resource syntax

Other good info on the main Tasks page: https://kubernetes.io/docs/tasks -- see sidebar under "Monitor, Log and Debug" (especially Troubleshoot Clusters and Troubleshoot Applications)

Kubernetes Slack

kubernetes.slack.com, channels:

#kubernetes-novice: beginner/"how do I get started?" issues

#kubernetes-users: General questions

Look within

A lot of your traditional knowledge is still relevant

- "None of this is new" -- Chen Goldberg, here, yesterday
- RFC 1925 is almost 22 years old but will still give you pertinent advice

Take the time to fully describe problems you encounter (rubber-duck debugging)

Demos

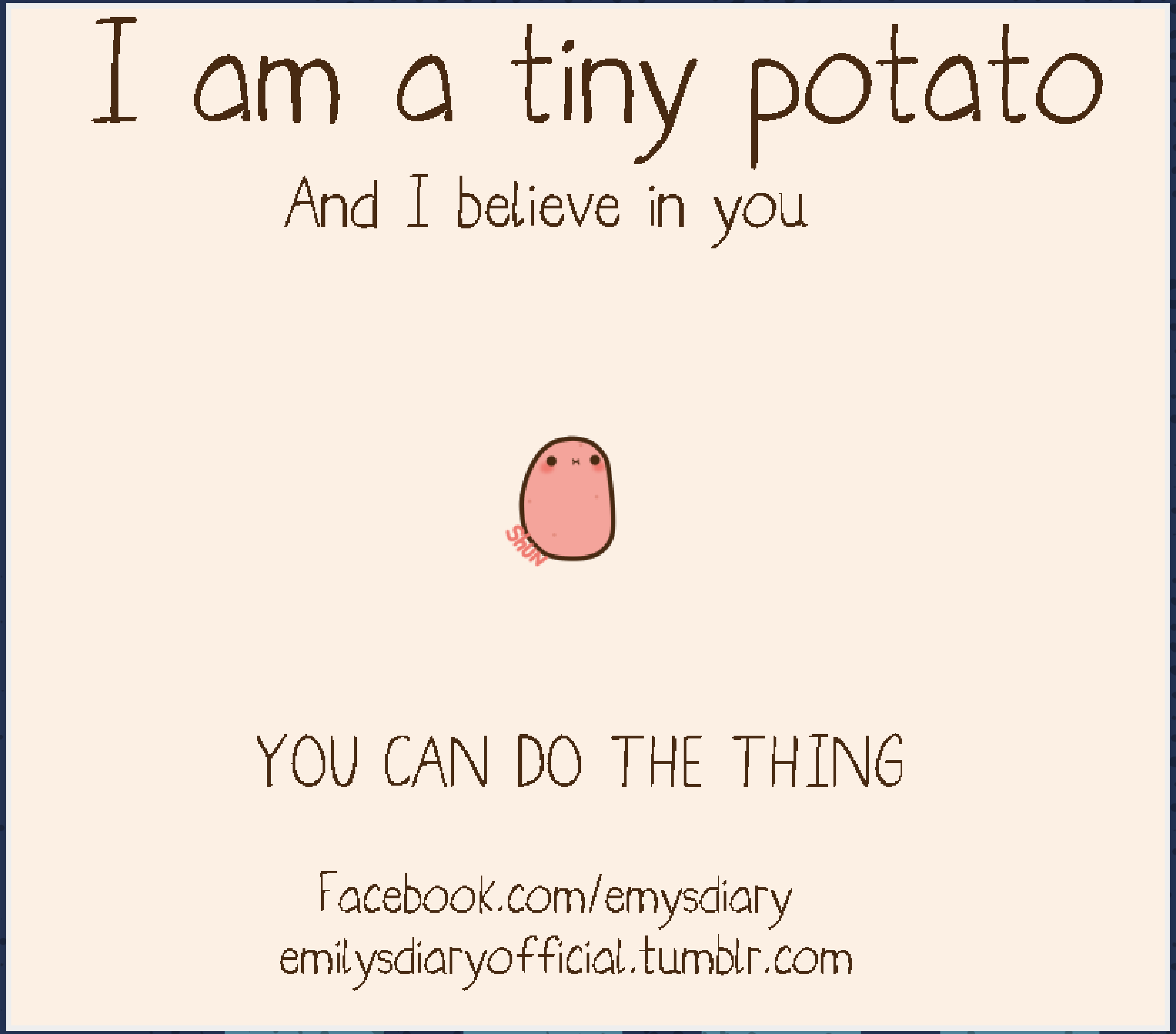
Final thoughts

We've just scratched the surface here

"Aren't there tools for this stuff?" -- yes, but what if deploying them fails? This is about giving you base knowledge to understand what underlies those tools

There *are* a lot of tools out there with advanced capabilities that will help you prevent, debug and fix problems -- find some awesome ones you love and tell us all about them!

Don't Fall for Impostor Syndrome



You know more than you think you do! If you feel like you're drinking from a firehose (especially the last three days!) then it just means you've got a good handle on the state of things

- "Trust Yourself" -- Ilya Chekrygin, here, yesterday

Questions?

Grateful Appreciation To:

Oteemo management (hi Sam!) for getting me here

Justin Garrison and Michelle Noorali for abstract help

Many CoreOS engineers and Red Hat trainers past and present for teaching *me* how to do this stuff and

Thank *you!*

for listening!

Files for demos

Slides:
http://bit.ly/2B81csY+