BERNARDO DONADIO

QUANDO A BALEIA AZUL ENCALHA

stone

DEVXPERIENCE

QUEM SOU

- Engenheiro de Automação de TI na Stone Pagamentos
- Membro da equipe de PaaS
- https://bcdonadio.com
- bcdonadio@bcdonadio.com

stone

PORQUE TUDO EVOLUI.

AGENDA

Problemas enfrentados

- REFERENCE LEAK
- DAEMON FREEZE
- BROKEN WHITEOUT
- DEVICEMAPPER MEMORY USAGE
- DNS RACE CONDITION
- TLS INSECURITY

REFERENCE LEAK

O BUG MAIS LADINO QUE EU JÁ ENCONTREI

REFERENCE LEAK

O bug mais ladino que eu já encontrei

"unregister_netdevice: waiting for lo to become free. Usage count = 3"

REFERENCE LEAK

O bug mais ladino que eu já encontrei

PROBLEMA

- Contadores de utilização que não diminuem
- Diversos pontos no kernel Linux
- Difícil de isolar (situação de corrida)

IMPACTO

- Impossível fazer CRUD de containers

WORKAROUND

- Desabilitar IPv6
- Colocar docker0 em modo promíscuo
- Usar poucos containers por nó
- Reiniciar o nó

BUGREPORT EXISTE DESDE MAIO DE 2014

388 COMENTÁRIOS NO BUGREPORT

5 VEZES "CORRIGIDO"

MOBY #5618 BUGREPORT ABERTO ATUALMENTE

DAEMON FREEZE

IRRITANTE, MAS CONTORNÁVEL

DAEMON FREEZE

Irritante, mas contornável

$ docker ps -a

DAEMON FREEZE

Irritante, mas contornável

PROBLEMA

- Docker daemon para de responder requisições
- Aparentemente relacionado ao subsistema de storage

IMPACTO

- Impossível interagir com o daemon

WORKAROUND

- Executar operações a uma taxa baixa
- Detectar “docker ps” demorando mais de 5s e reiniciar o daemon

BUGREPORT EXISTE DESDE JUNHO DE 2015

174 COMENTÁRIOS NO BUGREPORT

SEM TENTATIVAS DE CORREÇÃO

MOBY #13885 BUGREPORT ABERTO ATUALMENTE

BROKEN WHITEOUT

POR QUE NÃO DEVEMOS USAR FILESYSTEMS BLEEDING EDGE

BROKEN WHITEOUT

Por que não devemos usar filesystems bleeding-edge

```
$ echo “Fernando Collor presidente” > arquivo
$ rm –f arquivo
$ cat arquivo
Fernando Collor senador
```

BROKEN WHITEOUT

Por que não devemos usar filesystems bleeding-edge

PROBLEMA

- AUFS whiteout não lida direito com diretórios
- AUFS é o backing storage padrão do Docker CE

IMPACTO

- Arquivos deletados em camadas inferiores voltam a existir
- Quebra apps que carregam arquivos de um diretório inteiro

WORKAROUND

- Usar backing storages alternativos: Overlay2, DeviceMapper ou BTRFS

BUGREPORT EXISTE DESDE DEZEMBRO DE 2014

4 COMENTÁRIOS NO BUGREPORT

1 TENTATIVA DE CORREÇÃO

MOBY #9690 BUGREPORT FECHADO MAS AINDA REPRODUZÍVEL

DEVICEMAPPER MEMORY USAGE

A ROBUSTEZ TEM CUSTO

DEVICEMAPPER MEMORY

A robustez tem custo

DEVICEMAPPER MEMORY

A robustez tem custo

PROBLEMA

- DeviceMapper duplica blocos em memória

IMPACTO

- Uso de memória cresce linearmente com mais instâncias de uso da mesma imagem
- Torna DeviceMapper pouco atrativo para PaaS

WORKAROUND

- Usar backing storages alternativos: Overlay2, AUFS ou BTRFS (com custo de estabilidade)

ESCOLHA CONSERVADORA DA REDHAT

DOCUMENTADO MAS POUCO ÓBVIO NO GUIA DO USUÁRIO

CORREÇÃO IMPOSSÍVEL

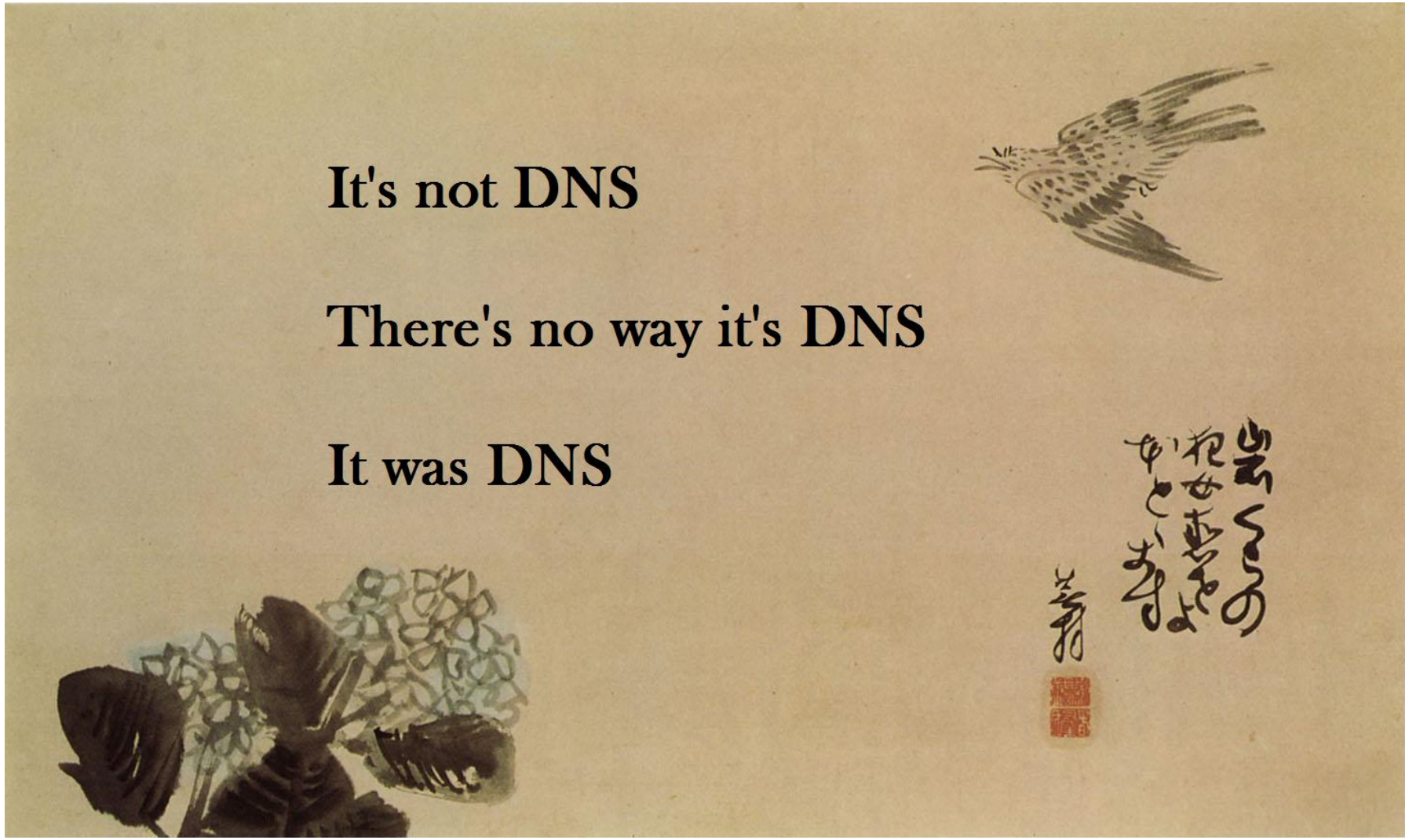
SEM BUGREPORT

DNS RACE CONDITION

A GLIBC TAMBÉM NÃO COLABORA

DNS RACE CONDITION

A Glibc também não colabora



DNS RACE CONDITION

A Glibc também não colabora

PROBLEMA

- Situação de corrida entre query IPv4 e IPv6
- Golang faz uso inseguro da Glibc

IMPACTO

- Resolução errática de nomes tanto no daemon como nos containers

WORKAROUND

- Usar cache DNS local com dnsmasq

BUGREPORT EXISTE DESDE SETEMBRO DE 2013

39 COMENTÁRIOS NO BUGREPORT

1 TENTATIVA DE CORREÇÃO

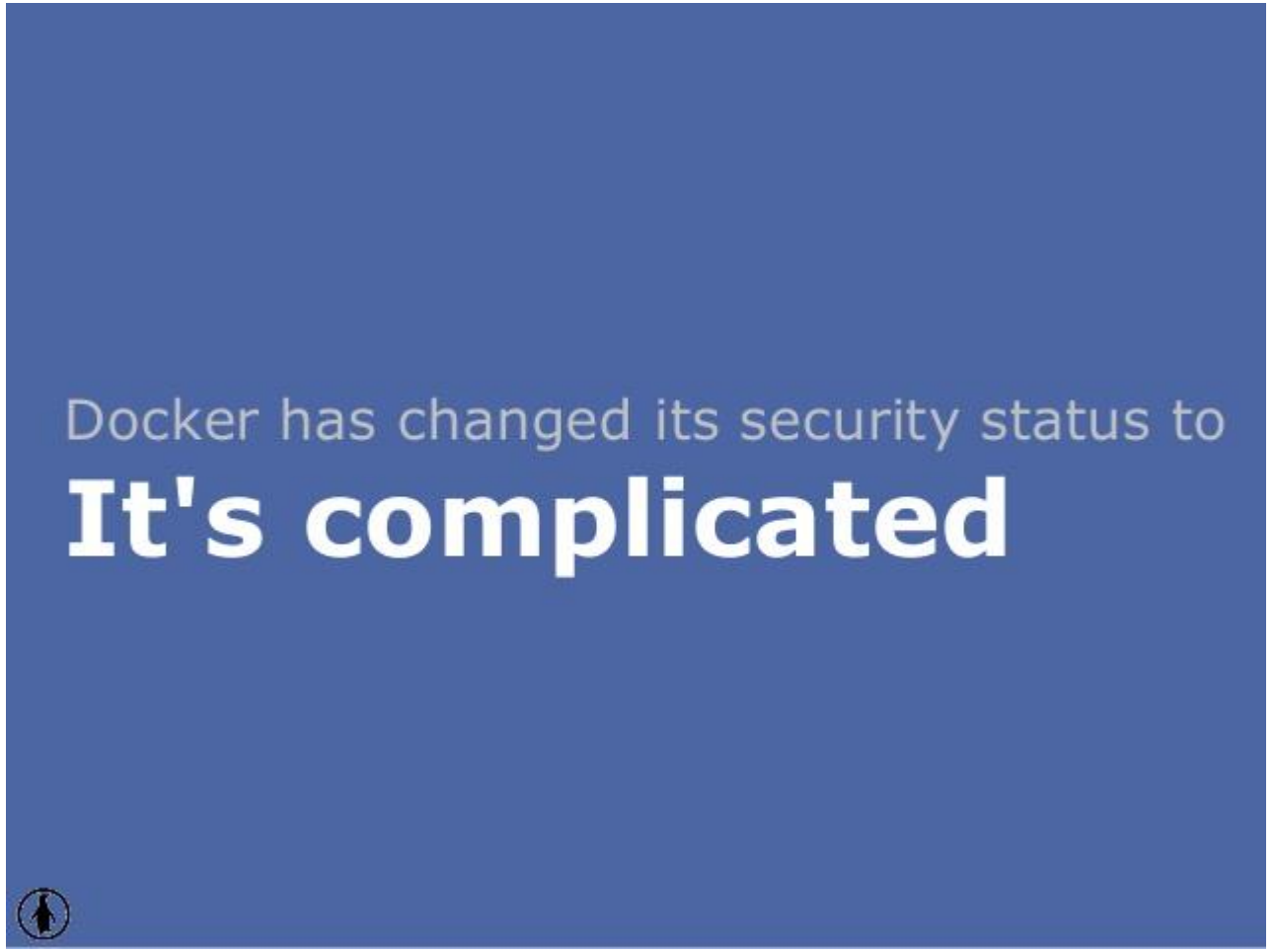
GOLANG #6336 BUGREPORT FECHADO MAS AINDA REPRODUZÍVEL

TLS INSECURITY

NA DÚVIDA, PERMITE TUDO

TLS INSECURITY

Na dúvida, permite tudo



TLS INSECURITY

Na dúvida, permite tudo

PROBLEMA

- Socket de controle do Docker não é seguro por default
- Se campo ExtendedKeyUsage não existe, permite tudo
- Documentação induz ao erro

IMPACTO

- Escalada lateral de privilégios
- Quando um host Docker é comprometido, todos os outros também são

WORKAROUND

- Enforçar obrigatoriedade do campo ExtendedKeyUsage na CA

REPORTADO EM AGOSTO DE 2017

RESPOSTA INSATISFATÓRIA

1 CORREÇÃO NAS DOCS

REPORTADO PRIVADAMENTE AO TIME DE SEGURANÇA

CONCLUSÕES FINAIS

BOAS PRÁTICAS PARA LIDAR COM UMA PLATAFORMA ETERNAMENTE INSTÁVEL

CONCLUSÕES FINAIS

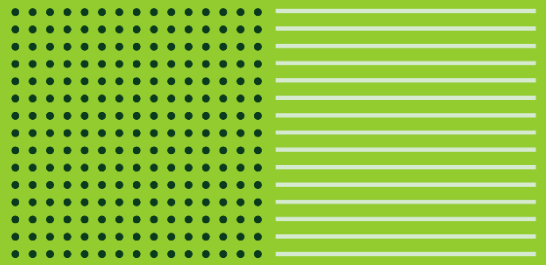
Boas práticas para lidar com uma plataforma eternamente instável

- Não use versões bleeding edge do Docker em produção
- Se agarre em algum vendor (Red Hat, Canonical)
- Ponha apenas apps 12-factor compliant no Docker
- Tenha pelo menos 3 instâncias do mesmo serviço em máquinas diferentes

CONCLUSÕES FINAIS

Boas práticas para lidar com uma plataforma eternamente instável

- Use DeviceMapper ou Overlay2 como backing storage
- Preste muita atenção nas configurações de segurança do Docker
- Use Docker com SELinux ou Apparmor
- Mantenha seus DBs longe do Docker



OBRIGADO.

BERNARDO DONADIO
bcdonadio@bcdonadio.com | https://bcdonadio.com

stone

DEVXPERIENCE