The Dark Art of Container Monitoring

Gianluca Borello
Gianluca Borello
- Software Engineer at Sysdig
- Core developer of sysdig
- Open source enthusiast

gianlucaborello
In This Talk

• Introduction to containers
• Monitoring containers with traditional open source tools
• Monitoring containers with sysdig
• Use cases
Containers vs VMs

- VM
  - Hypervisor
  - Distro
  - Kernel
  - Server

- App1
  - Distro
  - Kernel

- App2
  - Distro
  - Kernel

- App3
  - Distro
  - Kernel
Containers Are Great…

• less overhead
• faster deployments
• reproducibility of environments
• cost optimizations
• Isolation
• flexibility
...But Seeing Inside Containers Is Not Easy

Containers are:
  • Isolated
  • Self-Contained
  • Simple
  • Lightweight
A slightly complicated containerized environment

• A “distributed” Wordpress web application
• Many containers
Some Things We Want to Monitor

• Resource usage (CPU/Memory/Disk)
• Network activity
• File I/O activity
• Errors/faults
• Application activity
sysdig

- Capture system events, filter them, run useful scripts
- `strace + tcpdump + lsof + htop + Lua`
- Open Source
- Nice curses UI

**Native support for containers!**
sysdig Architecture

Kernel

- App
- Container1 (Docker)
- Container2 (Docker)
- Container3 (LXC)
sysdig Architecture

Instrumentation through kernel module
sysdig Architecture

Capture and analysis
sysdig Architecture

Kernel

- App
- Container1 (Docker)
- Container2 (Docker)
- Container3 (LXC)
- sysdig (Docker)

Save to a trace file: foo.scap

(optionally)
What’s a system call?
Sysdig’s capture architecture

- Command line parsing
- Capture management

- Event parsing
- State engine
- Filtering
- Output Formatting
- Chisel execution

- Capture Control
- Dump files R/W
- OS state collection

- Non-blocking event collection
- Type-based event packing
- Memory mapped buffer handling

Tracepoints
/doc/Documentation/trace/tracepoints.txt