# Reactive Monitoring

Sam Clements

# Context

Being a systems administrator sucks.
Everything is on fire, all the time.

So we have tools to monitor infrastructure.
They tell us when something has gone wrong.

# Alerting

When a problem arises that needs fixing, these tools can send alerts.

A systems administrator can then ‘log on’, and manually take steps to resolve the issue.

# On-call

Companies usually have a rotating ‘on-call’ position or team that is available 24/7.

If an on-call team is small, or only located in one timezone, this means getting woken up.

# Disadvantages

This is fine when there are major issues that **need** to be manually fixed.

But lots of problems are much smaller, and have simple fixes.

# This is not fun

Being woken up at night to fix unimportant alerts leads to grumpy sysadmins.

# An improvement

I’m working on a system that can **react** to certain problems and attempt to fix them.

This means not having to get up to deal with problems that don’t require complex fixes.

# How will it work

The system will load a configuration file defining **resources**, **conditions**, and **actions**.

*When RESOURCE is in CONDITION, do ACTION.*

# Examples

*When service X is using no CPU time, restart it.*

*When website A has stopped responding, clear these logfiles and restart service B.*

*When service Y is not running, make a dump of the database.*

# But actually…

"…shouldn't you fix the programs that are being monitored?"

Which is fine until you use an unmaintained program that no-one knows how to fix, or the developer is on holiday, or the developers ran out of coffee, or no-one knows how to fix the problem, or they can't publish another release until at least next week, or they have more important projects to work on, or they all get hit by a metaphorical bus.

# But actually…

“…shouldn’t services be automatically restarted?”

Which is fine until you discover the service is running, but not using any CPU or memory.