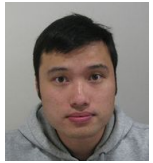
Jenkins Inside Google

David Hoover

Jenkins Inside Google

David Hoover

Kokoro - Jenkins Inside Google










Kokoro - Jenkins Inside Google

Kokoro:

- Build & sign non-standard Google projects
- Replace aging system
- Replace ad-hoc Jenkins use:
  - Identified dozens of 'Jenkins-under-the-desk' instances

Goals:

- Secure
- Reliable
- Easy self-service

Kokoro - Jenkins Inside Google

2015:

- Investigated Jenkins viability
  - Ran stress tests
  - Found/fixed critical bottlenecks
- Presented at Jenkins User Conference with our findings
- Started onboarding Windows customers

2016:

- Launched Linux support
- Launched Git/Gerrit integration
- Launching macOS support

Scale - 2015 Stress Test

- 1 Master
- 500 Build Agents

- 2000 Projects
- 500 Builds per Minute

Scale - 2016 Reality

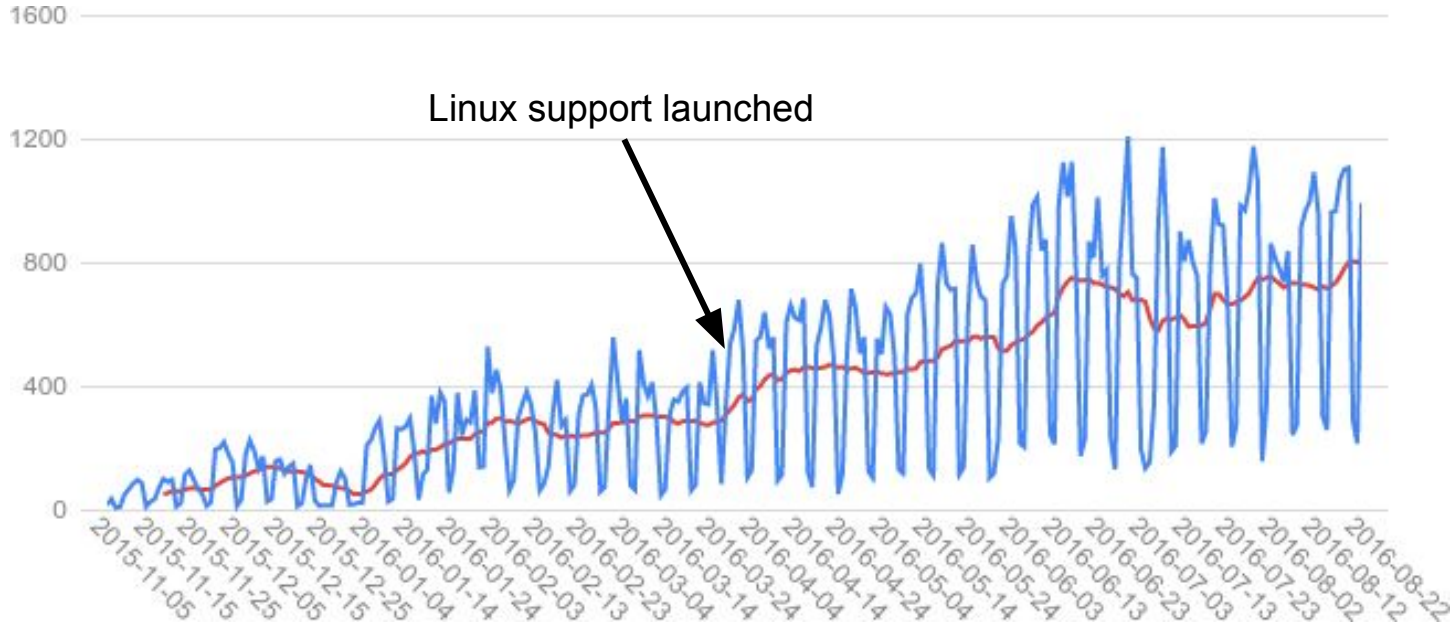
- 1 Master
- 100+ Build Agents

- 200+ Windows Projects (Launched 2015-10-19)
- 200+ Linux Projects (Launched 2016-04-04)
- 1.5k+ Builds per Day
- 250+ Active Users

Mac support launching as we speak:

- 400+ Build Agents
- 750+ Projects (Anticipated)
- 3.5k+ Builds per Day (Anticipated)

Scale - Builds per Day



Reduce Load on Jenkins Master

- Store build artifacts externally
  - One project has nearly 20 million files
- Store logs externally
  - Log output streams directly from Jenkins agent to external log service
  - Jenkins master only stores URL
  - We plan to open-source this work
- Directly install all jars on agents in advance
  - Saves over 500MB traffic per agent
- External service for continuous integration
  - Receives Piper submit notifications (like GitHub webhooks)
  - Polls Git repositories
  - Sends RPC to Jenkins to start build

Reduce Administrative Workload

- Dynamic agent registration
  - Agents started externally, self-register over gRPC
  - Easy to scale pool
  - In process of being open-sourced
- Project configuration stored in source control
  - Full history of all changes
  - Goes through regular code review
- Automatic project creation/deletion
  - Jenkins notices added/removed configurations in source control
  - Always reads configurations from source control

Reliability

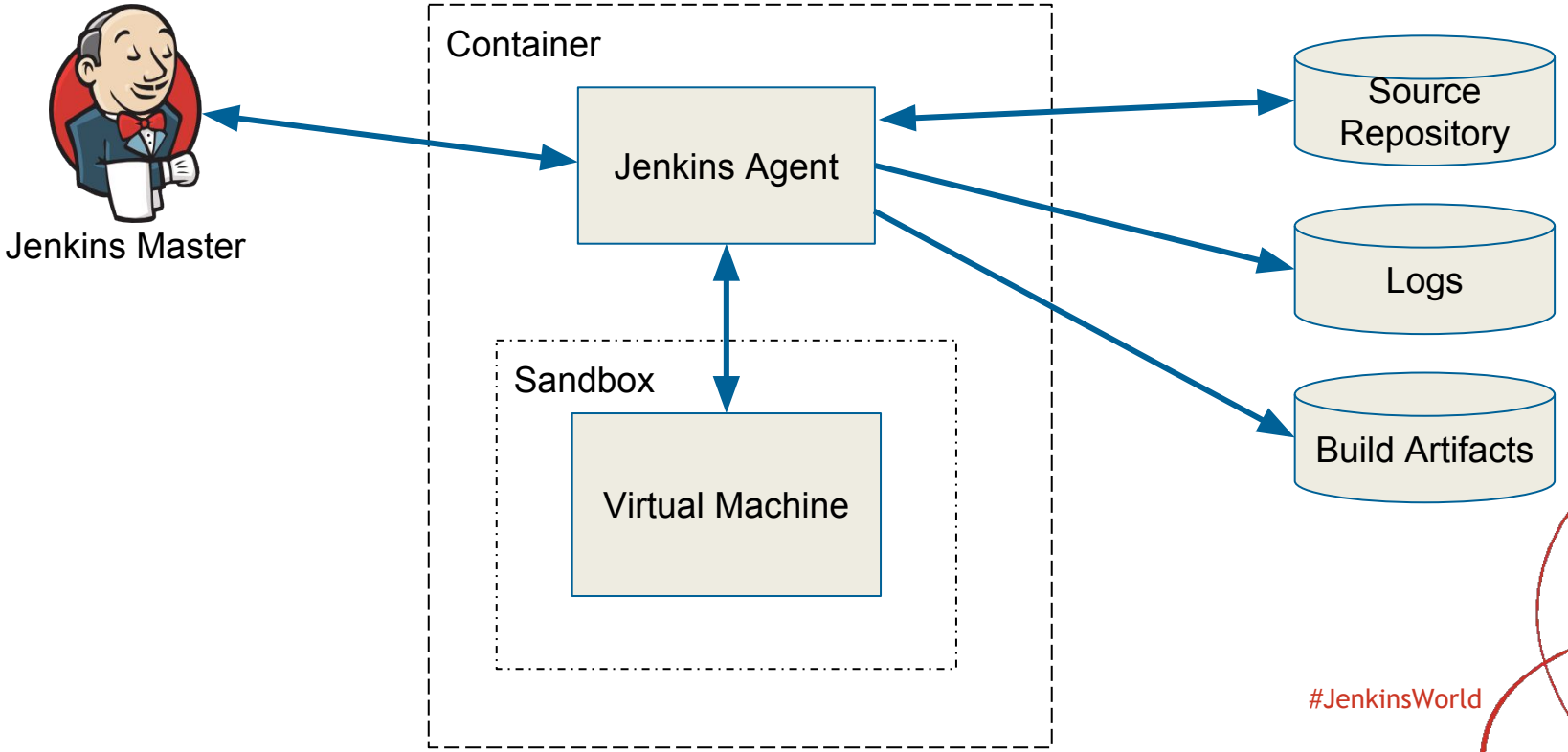
- Keep agents simple
  - One agent = one executor
- Automated Jenkins master failover
  - Spares ready in case anything goes wrong
- Virtually zero administration through UI
  - Track all changes
  - No problems due to misclicks
- Monitoring and alerting
  - Collect metrics
  - Watch trends
  - Alert a human if something seems fishy

Only one hour unplanned downtime so far this year!

Agent Design

- One executor per agent
  - Easy to schedule many small jobs
  - No risk of interactions between builds
  - If something goes wrong, only affects one executor
- Run workload inside sandboxed VM
  - Restart VM with clean OS image after every build
  - Reproducible builds
  - Isolated from network flakiness
- Keep plenty of spares online
  - VMs take minutes to boot

Agent Design - Windows & Ubuntu



Agent Design - macOS

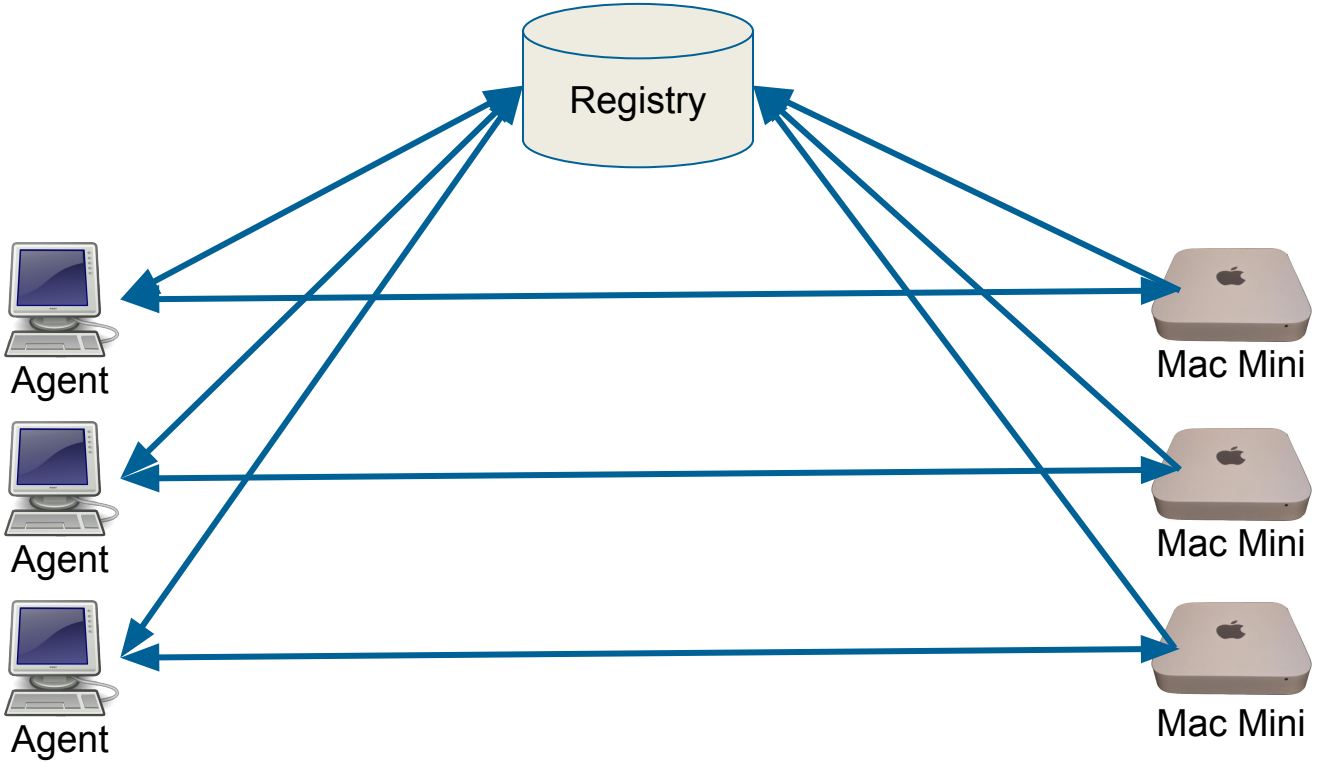
Challenge:

- macOS/iOS builds require Apple hardware
- No Macs in Google production data centers

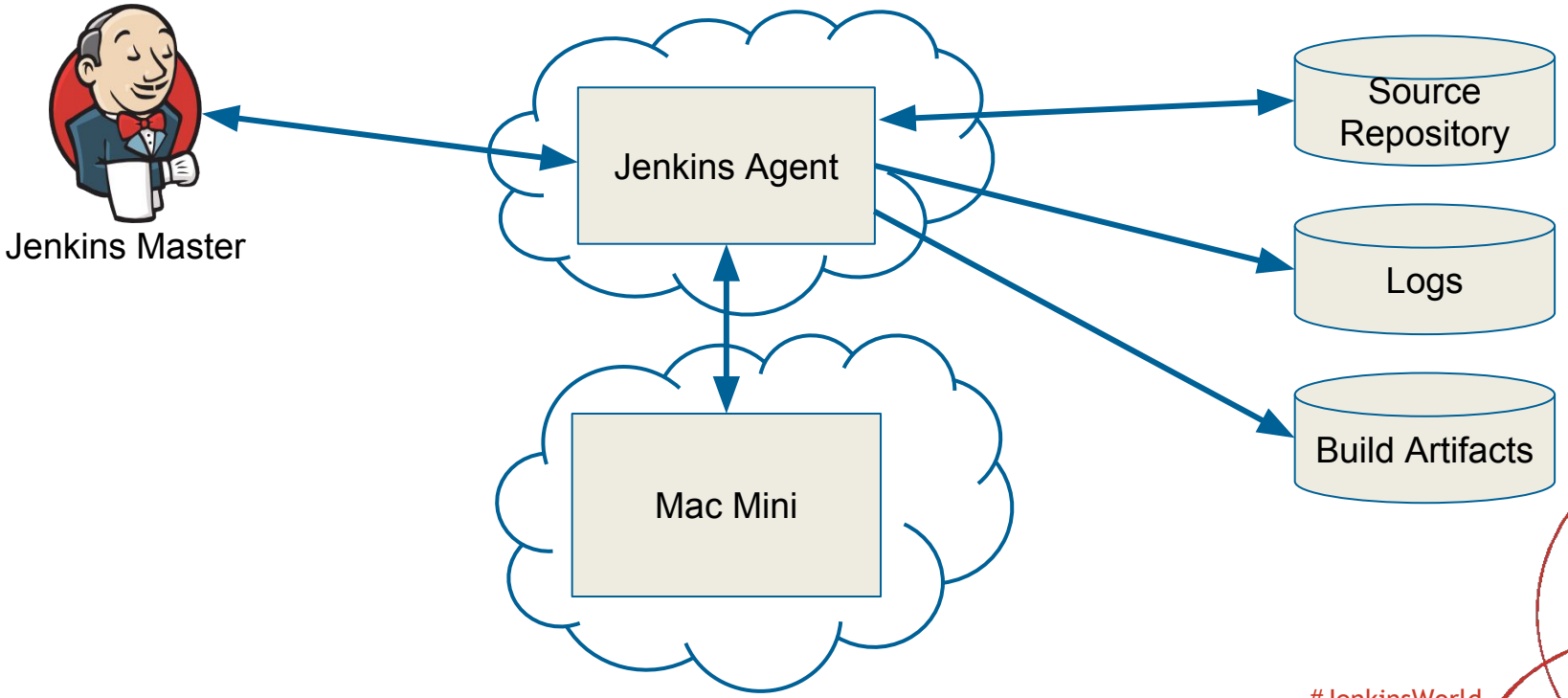
Solution:

- Host Macs in another data center
- Pair Macs with Jenkins Agents
- Netboot Macs with clean OS image after every build

Agent Design - macOS



Agent Design - macOS



Agent Design - GCE/Device Labs

- Just like macOS agents
  - Kokoro runs Jenkins agent in our cloud
  - Customers run build machine in GCE
- Allows completely custom build environments
  - Custom OS
  - Larger-than-normal workloads
  - Specialized hardware (e.g. attached devices)
  - Consistency for performance testing

Project Configuration

Existing Approaches:

- Literate
- YAML
- Pipelines
- DSL

None quite met our needs:

- Project & build configuration tightly coupled
- No shared configurations
- Still modifiable through the UI

Project Configuration

- Split project and build configurations
  - Project config read at head, build config at some other revision
- Generate build steps at runtime
- Automate project registration
  - Configure via RPC when committed
- Nested configurations
  - Share common configuration across projects

Project Configuration


```
# common.cfg
type: CONTINUOUS_INTEGRATION
scm {
  piper_scm {
    depot_path: "//path/to/some-project/..."
    depot_path: "//path/to/some-shared-library/..."
  }
 }
email_address: "some-team@google.com"
```

Project Configuration


```
# windows.cfg
cluster: WINDOWS
build_config_dir: "path/to/some-project/windows"

# ubuntu.cfg
cluster: UBUNTU
build_config_dir: "path/to/some-project/ubuntu"
email_address: "ubuntu-specific-team@google.com"
```

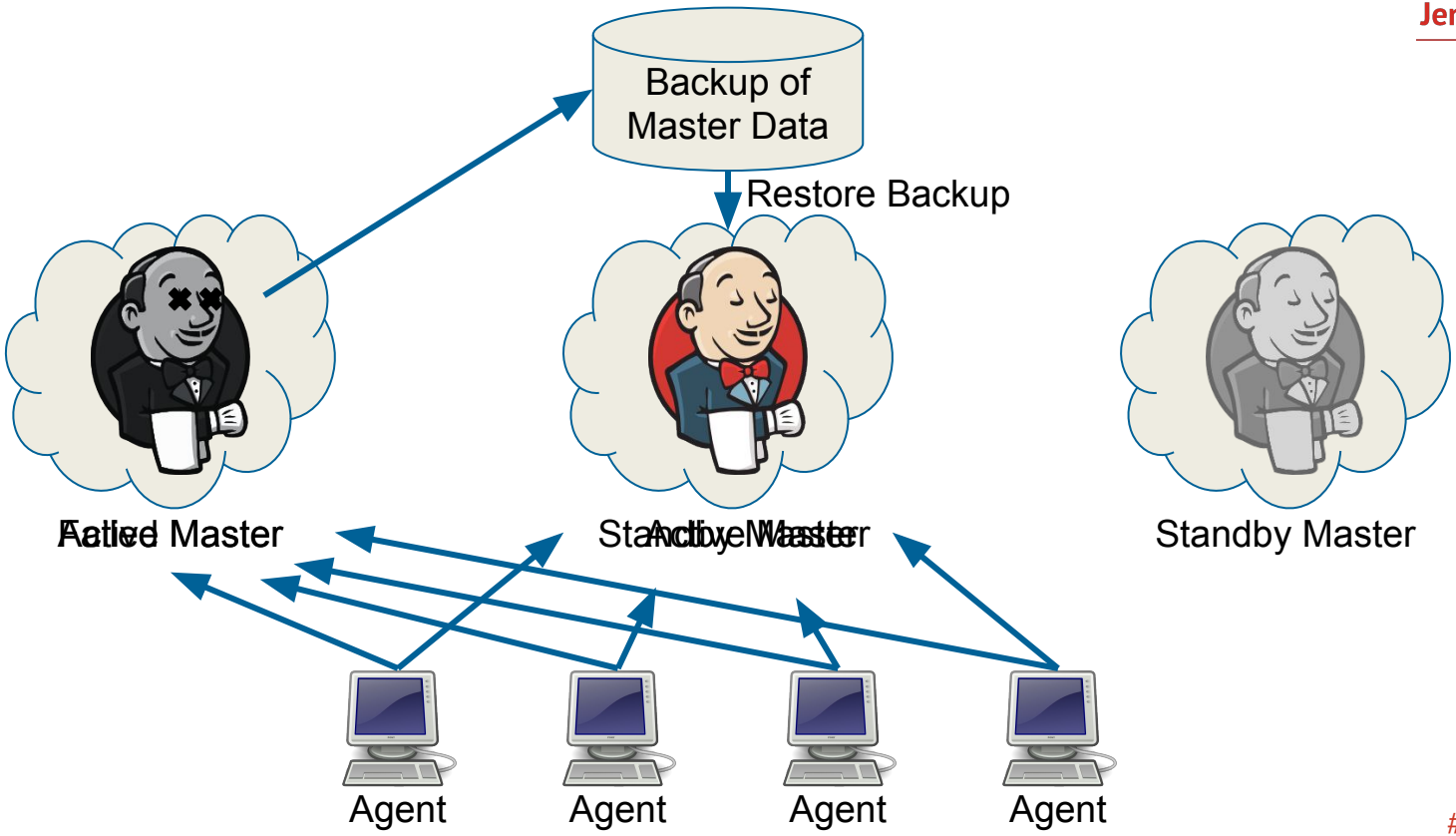
Project Configuration


```
# path/to/some-project/windows/windows.cfg
build_file: "path/to/some-project/windows/build.bat"
timeout_mins: 30
action {
  define_artifacts {
    regex: "**/output/*.exe"
  }
}
```

Project Configuration - Pipelines?

- Plan to generate Pipelines based on our existing configs
  - Similar to Declarative Pipelines
  - Durable builds across master restart
- Investigating more general use
  - Some teams already have complicated Pipelines
  - Must restrict what users can do outside the sandbox

Master Failover



Jenkins Pain Points

- Singleton master
  - Cannot run multiple & spread load
  - Downtime if it fails
  - Downtime when updating
- In-flight builds lost when master restarts
  - Try to push updates during quiet times
  - Looking to Pipelines for durable agents
- Agent management
  - Jenkins UI not geared for 100s of agents
- No project/agent affinity
  - Almost always a clean slate, no incremental source sync

Where Next?

- Onboard macOS builds
  - ***Just*** launched; hopefully my team is still in the audience, not fighting fires
- Pipelines
  - In-flight builds survive master downtime, better parallelism
  - Better visualization with Blue Ocean
- Onboard new teams
  - Support custom VM images
  - New features as needed
- Better integration with other systems
  - Simpler migration for acquisitions, support for Google open source
  - Custom build agents in GCE
- Blue Ocean
  - Preliminary work with CloudBees to use the new UI
- Keep scaling

Questions?

Q & A

Jenkins World 2016

#JenkinsWorld