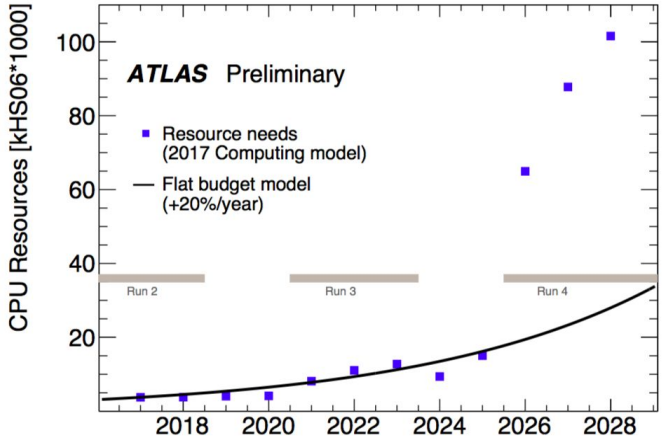
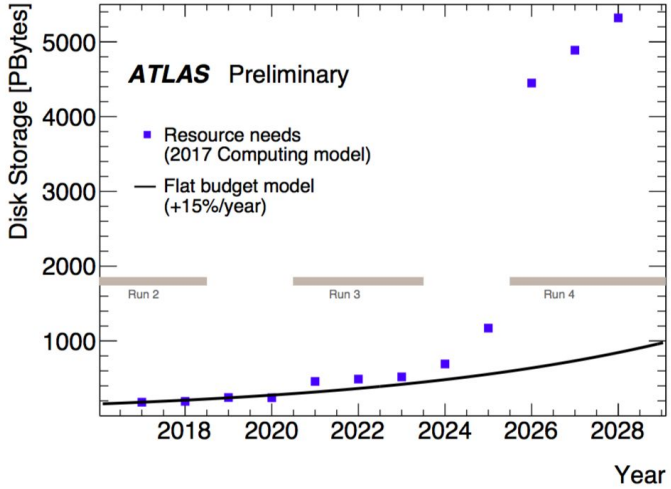
# ATLAS-Google Data Ocean Project

Fernando Barreiro (University Texas at Arlington)
on behalf of ATLAS Distributed Computing
HOW 2019, 20 March, JLAB


# Motivation

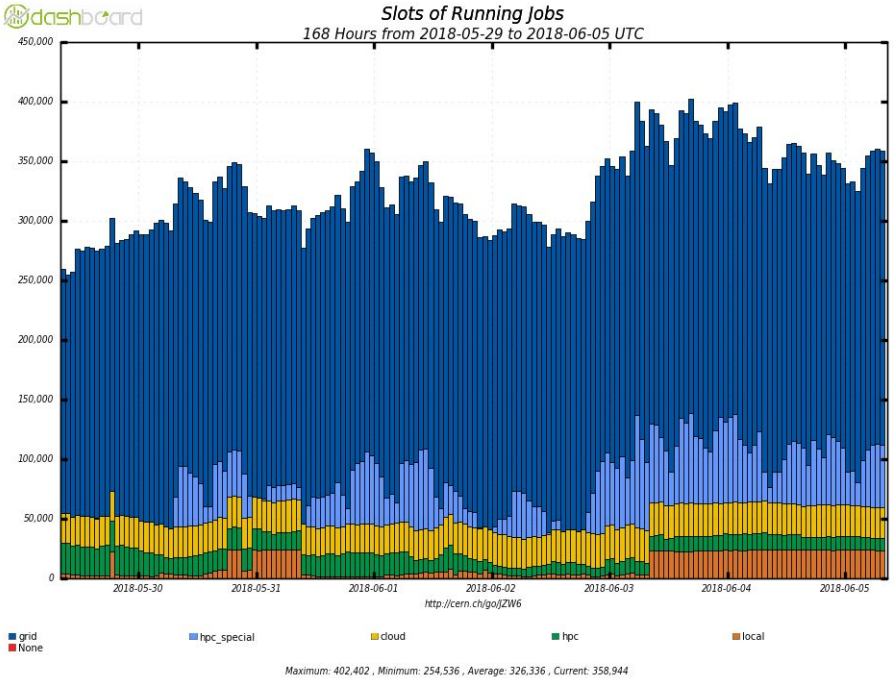



- LHC computing needs keep increasing, while budget will likely remain flat
- Novel computing models with more dynamic use of storage and compute need to be considered
- Commercial sector has evolved since the inception of the WLCG
  - Cloud computing can offer attractive solutions
  - We can learn from industry leaders

# ATLAS usage on non-grid resources

- Cloud, HPC & volunteer resources used successfully for >7 years
- Resources not always tailored for ATLAS: adaptation needed and inherent limitations in suitable workflows
  - Integration was mostly focused on running MC Simulation reading from remote Grid storage



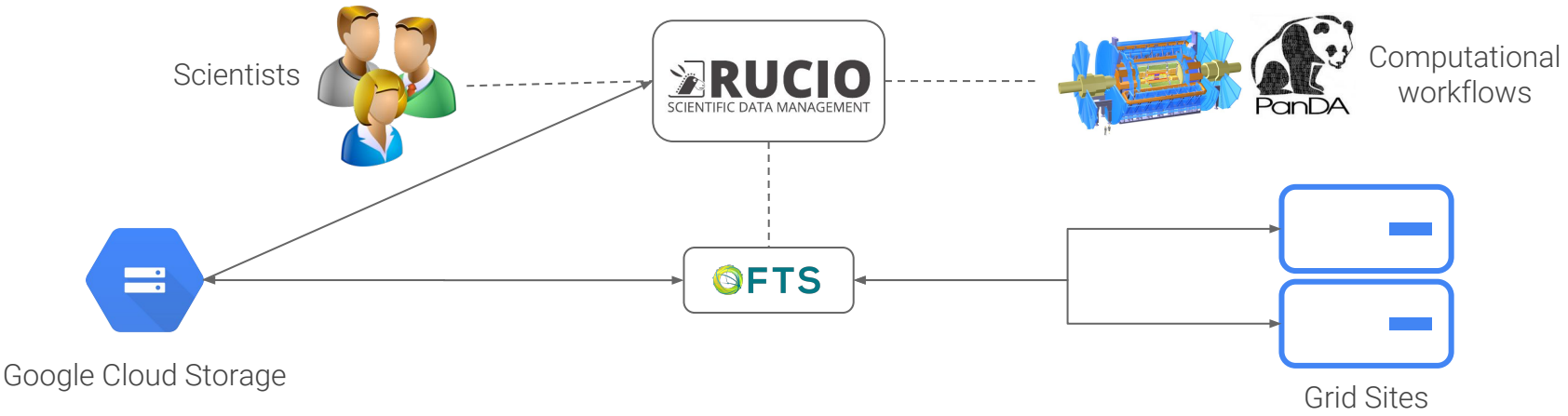
# ATLAS-Google PoC phase

- March-June 2018: integration with ATLAS data & workflow management
  - Data: Google Cloud Storage as a standard Rucio Storage Element
    - 3rd party transfers through FTS, download & upload, accounting
      - No workarounds like Fuse mounting a gridFTP server
  - Jobs: schedule jobs to Google Compute Engine through Harvester
    - Simulation
      - Centralized production with low I/O
      - Standard workload for opportunistic resources
    - End user analysis
      - Ensure 100% input & output availability to improve user experience

M. Lassnig

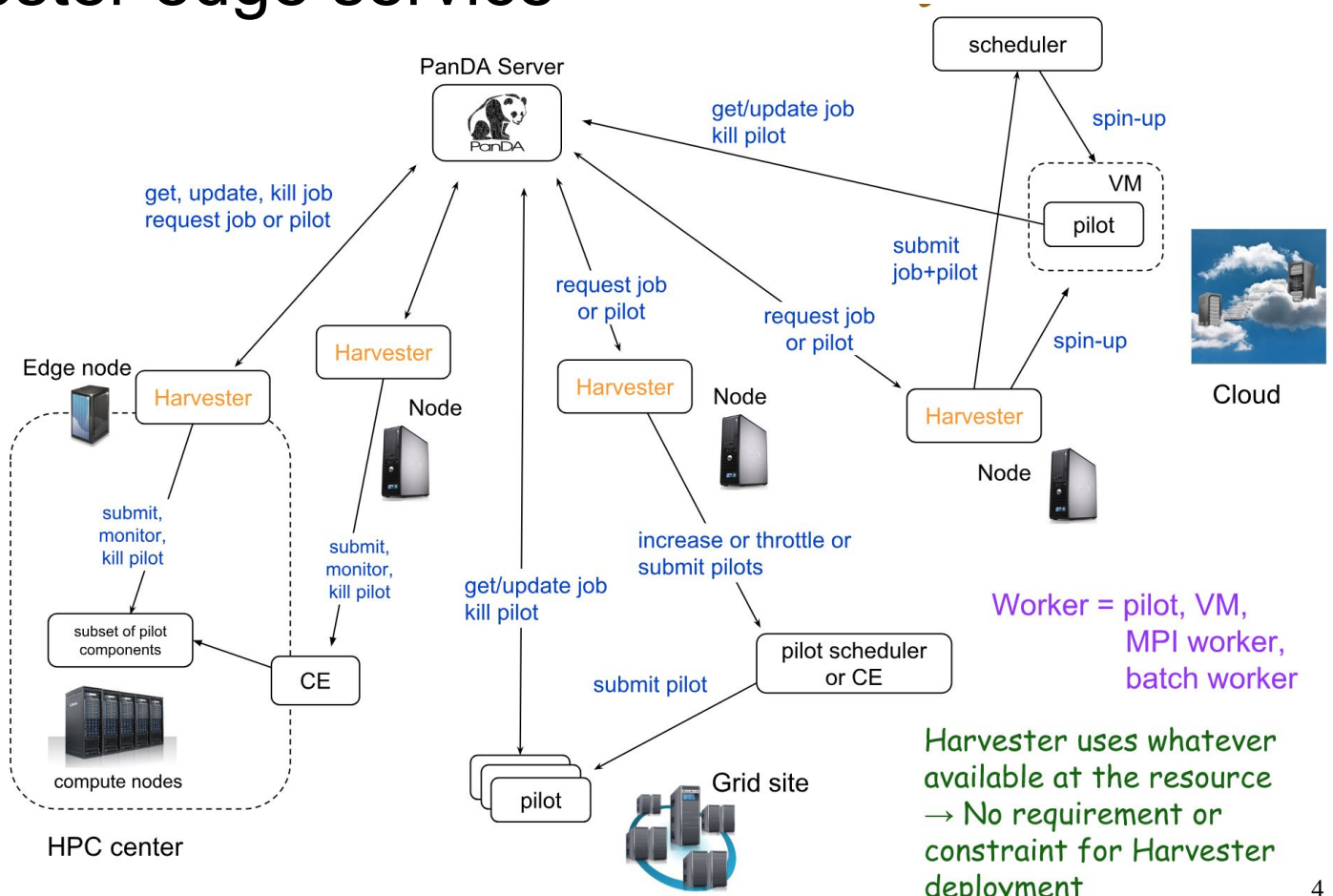
# Getting data into GCS

- The ATLAS Data Management system *Rucio* orchestrates all experiment transfers
  - S3 used in the first iteration, since support is already available from both sides
  - Tests successful, however not usable for client-based access (key distribution, server-side signing)
  - Parallel third-party copy is rate-limited to 100MB/sec because we were not using the native GCS API
- Decision to move to GCP-native client-side signed URLs

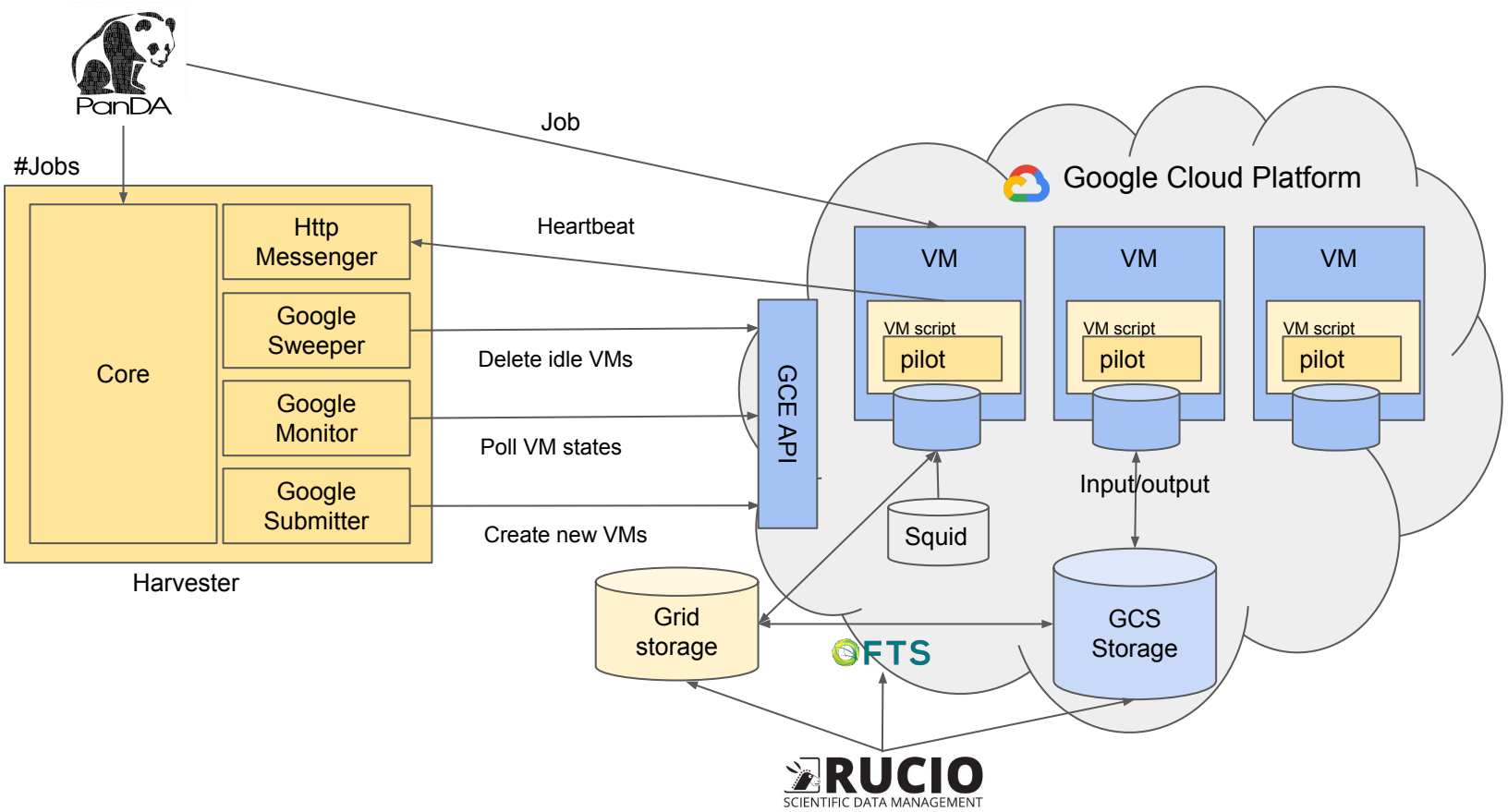


# Harvester edge service

T. Maeno



# Harvester and GCP

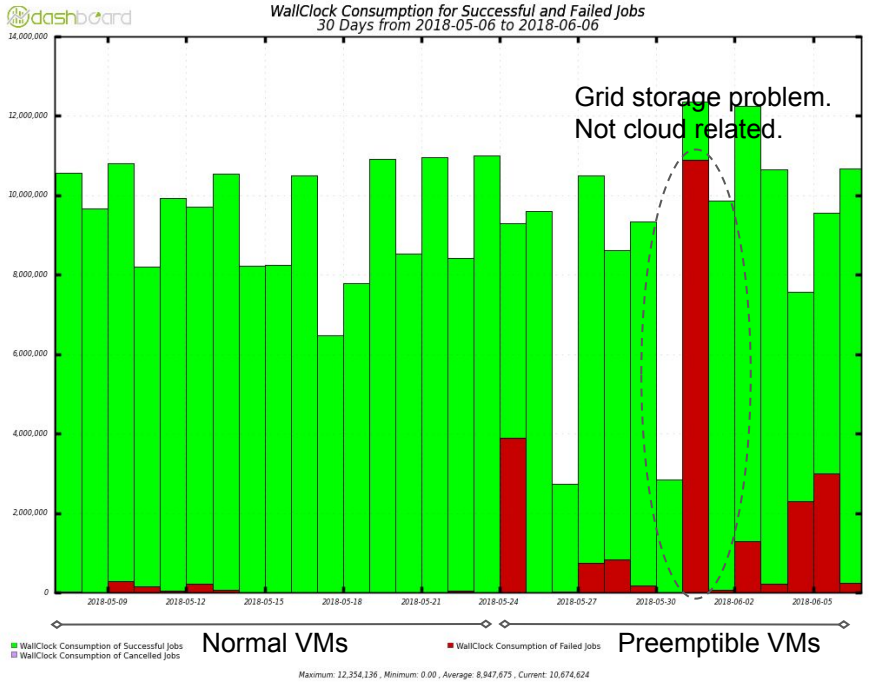


# Harvester integration details

- Purest PanDA-GCE integration: no intermediate layers
  - Plugins talk to GCE via Python API
- CernVM4 image and cloud-init contextualization
  - CVMFS, Squid, Proxy and PanDA configuration
  - VM startup script: start the pilot while sending VM heartbeat/killme messages
  - VMs are recycled ~once per day
  - Squid cache deployed in GCE
- Reducing the cost
  - Custom VMs adjusted to ATLAS simulation (8 vCPUs, 16GB RAM, 50GB disk)
  - Evaluation of preemptible VMs (20% of the cost)
    - Preemptible VM can be evicted any time and the maximum lifetime is 24 hours

# Compute evaluation for simulation

- Operated a 120 core cluster running standard **simulation** jobs for 1.5 months
  - **I/O to CERN storage**
  - Excellent success rate (<<5% errors) using normal VMs
- Preemptible VMs
  - Significantly higher error rate (~20% including a Grid storage outage)
  - Still gain on a $/event basis



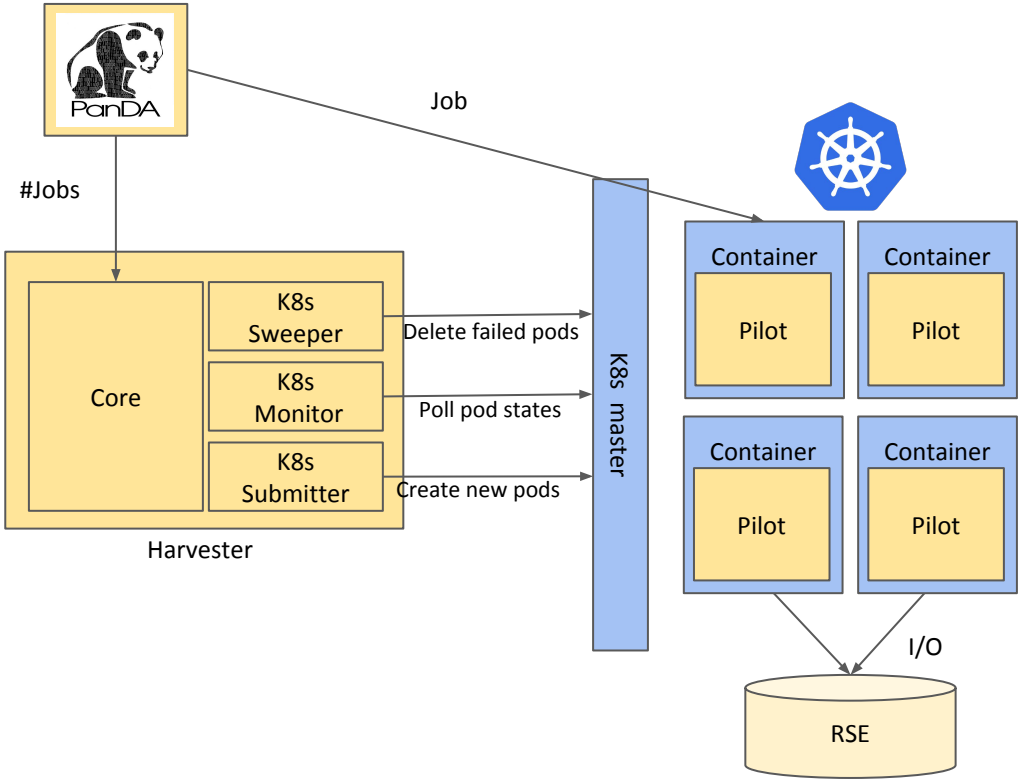

Efficiency of preemptible VMs can be optimized through usage of Event Service.

# Analysis use case

- First cloud exercise with **native use of cloud storage**
  - Pre-placement of datasets to Google Storage
- It was more challenging than simulation
  - Add support for signed URLs in various places
  - Incompatibilities between CernVM4 and Athena pre 21 releases
    - Requires generating and pre-upload of shared libraries
  - Preemptible VMs create confusion
  - Memory leaks in user code
  - File corruption errors through direct I/O: need to stage-in full files
- Full user analysis succeeded (Arnaud Dubreuil under Johannes supervision)
  - ~1M events, 450 GB of input, and 63 GB output

# Harvester and Kubernetes

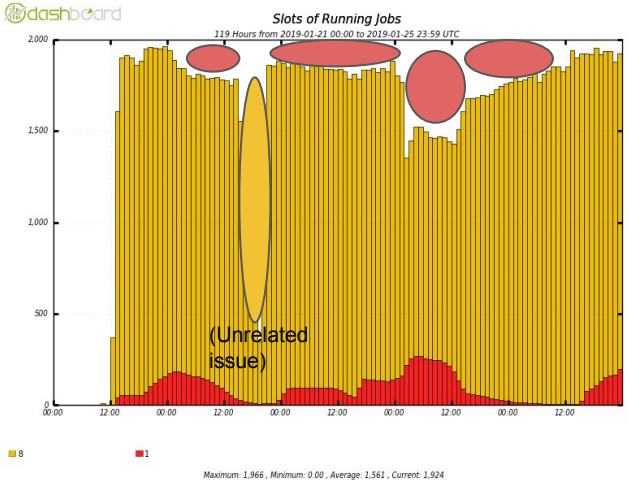
FaHui Lin,
Mandy Yang




- Use Kubernetes as a batch system
  - SLC6 containers + CVMFS-csi driver
- Still room for evolution and more flexibility, e.g. execution of user containers and options

# Harvester and Kubernetes

FaHui Lin




With default K8s scheduler (round robin load balance)


With policy tuning to pack nodes

- Tested at scale for ~ 1 month thanks to CERN IT & Ricardo Rocha
- Default Kubernetes pod orchestration very inefficient for us
  - Need custom scheduling policies
- Need automated node management
  - Lost many nodes during the exercise
  - Ricardo proposed solutions to regenerate lost nodes automatically
- Interest of some sites to evaluate K8s path, e.g. UVic will continue work as qualification task

# Summary and outlook

- Clean, fast integration between ATLAS and Google Cloud during first pilot evaluation
  - Small scale resource usage
- Next steps under discussion: foster long-term collaboration in different disciplines
  - End-user analysis
  - US WLCG sites extension with GCP
  - Optimized I/O and data formats
  - Data management across hot/cold storage
  - Machine learning and quantum computing