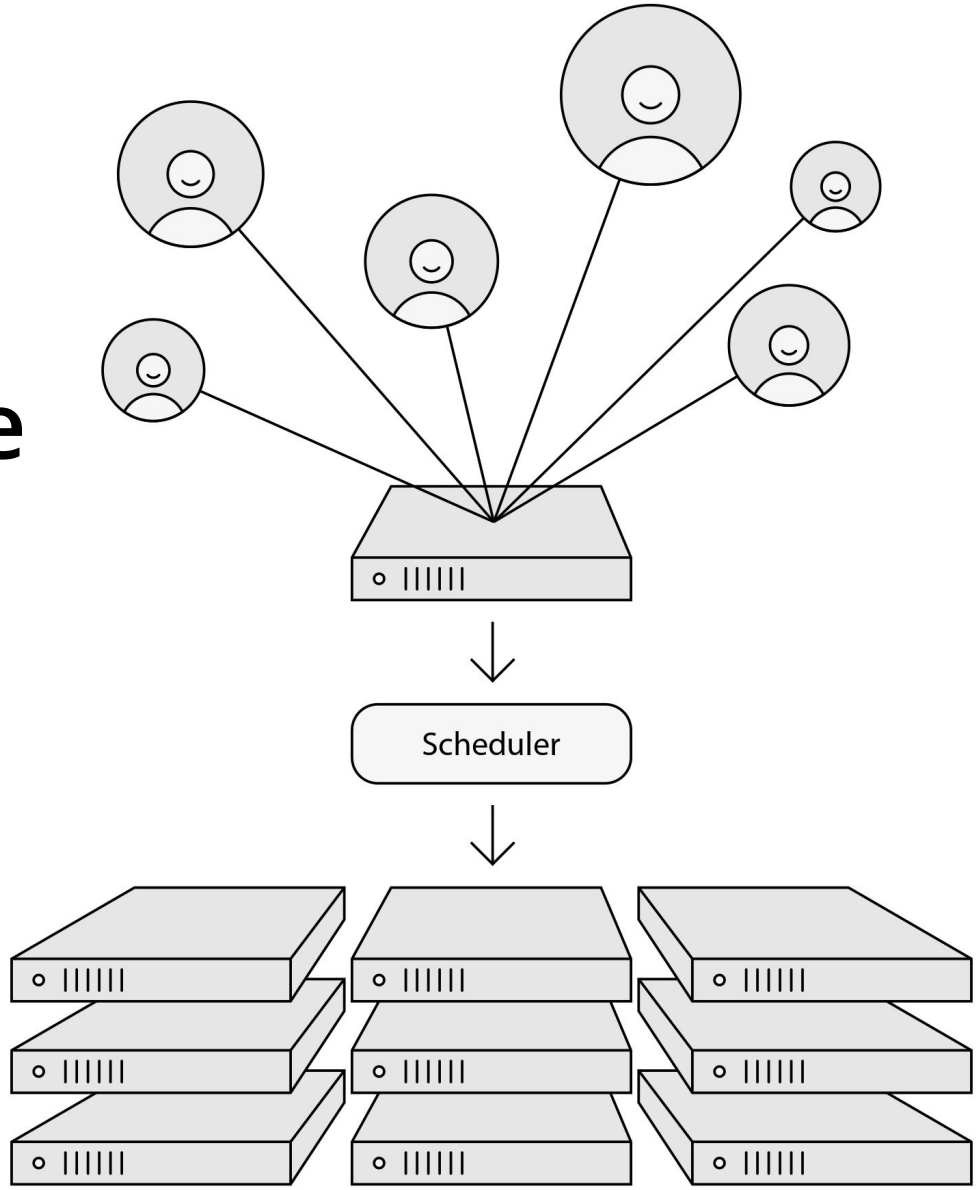
Arbiter: Dynamically Limiting Resource Consumption on Login Nodes

Dylan Gardner, Robben Migacz, and Brian Haymore
University of Utah Center for High Performance Computing

Login nodes are gateways to computational resources.



Our general policy on login nodes

Login nodes should be used for **low-impact** tasks, such as:

- Compiling software
- Preparing scripts
- Transferring small amounts of data
- Some small-scale testing for limited amounts of time

Computational resources need to be used for larger tasks!

The Problem

Inexperienced users and users trying to skip queues *(frequently)* violate login node policies.

This leads to:

- Slow and unresponsive login nodes
- Increased burden on user support teams
- Continued administrator intervention

The *Arbiter* service watches usage on login nodes and dynamically limits the resources available to users who are violating policies.

How it works

- Default hard CPU and memory quota via cgroups
- Scores usage based on soft quota (threshold)
  - Increases "*badness score*" when above soft quota, decreases when below (slower rate)
  - Max badness score is a violation
- Emails users about violations
- Lowers hard quotas on violation
  - Repeated violations result in even lower hard quotas
- Certain processes can use up to hard quota (e.g. compilers)

Configurable to fit different policies

How does the *badness score* work?

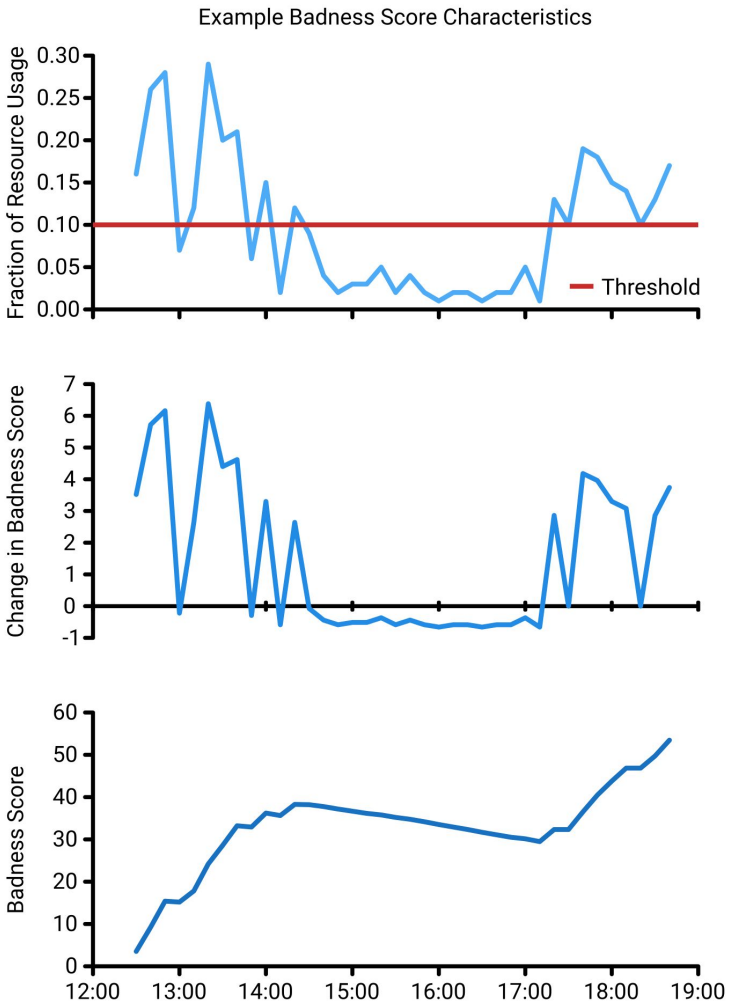
Usage

↓

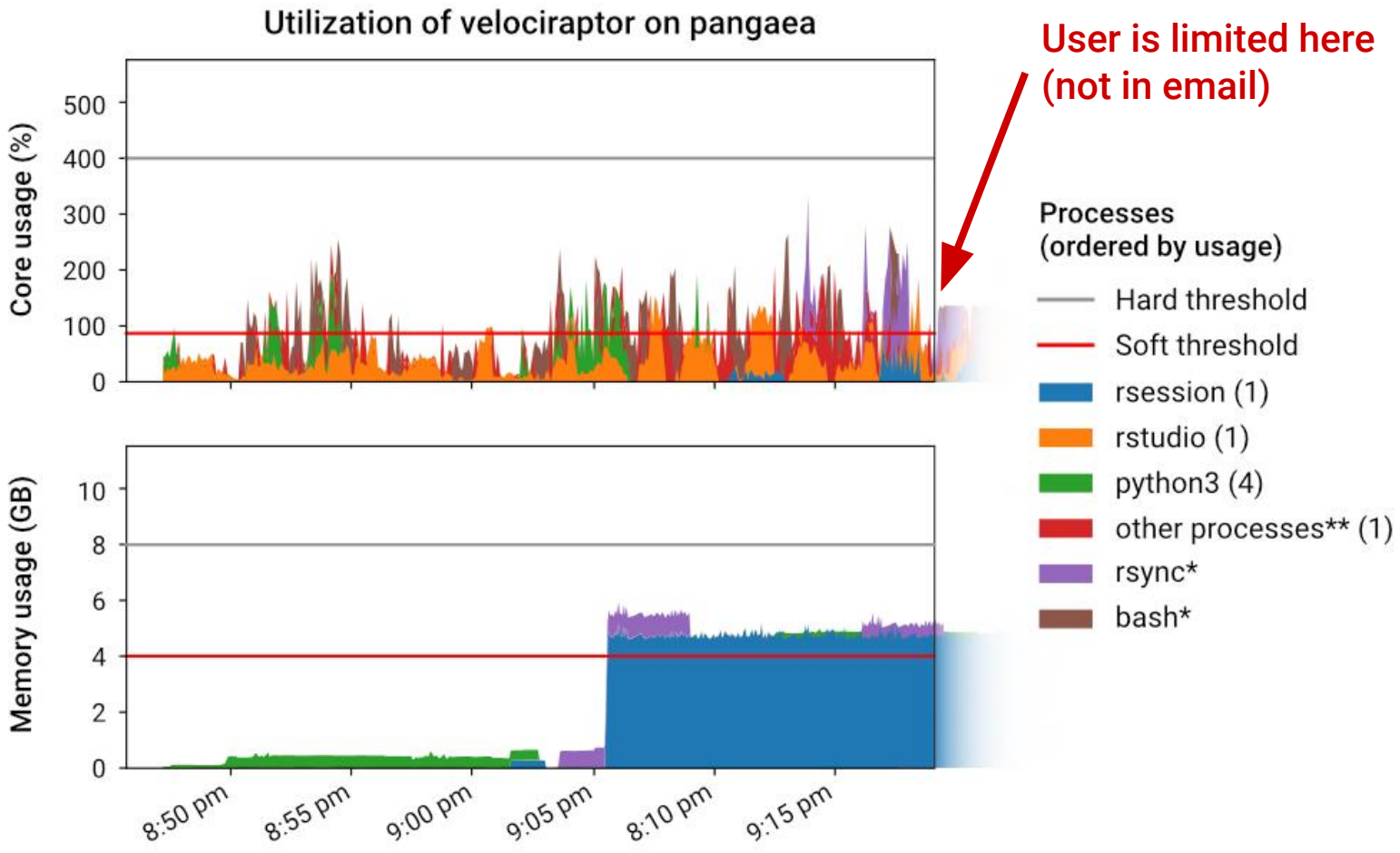
Change in badness

↓

Cumulative badness



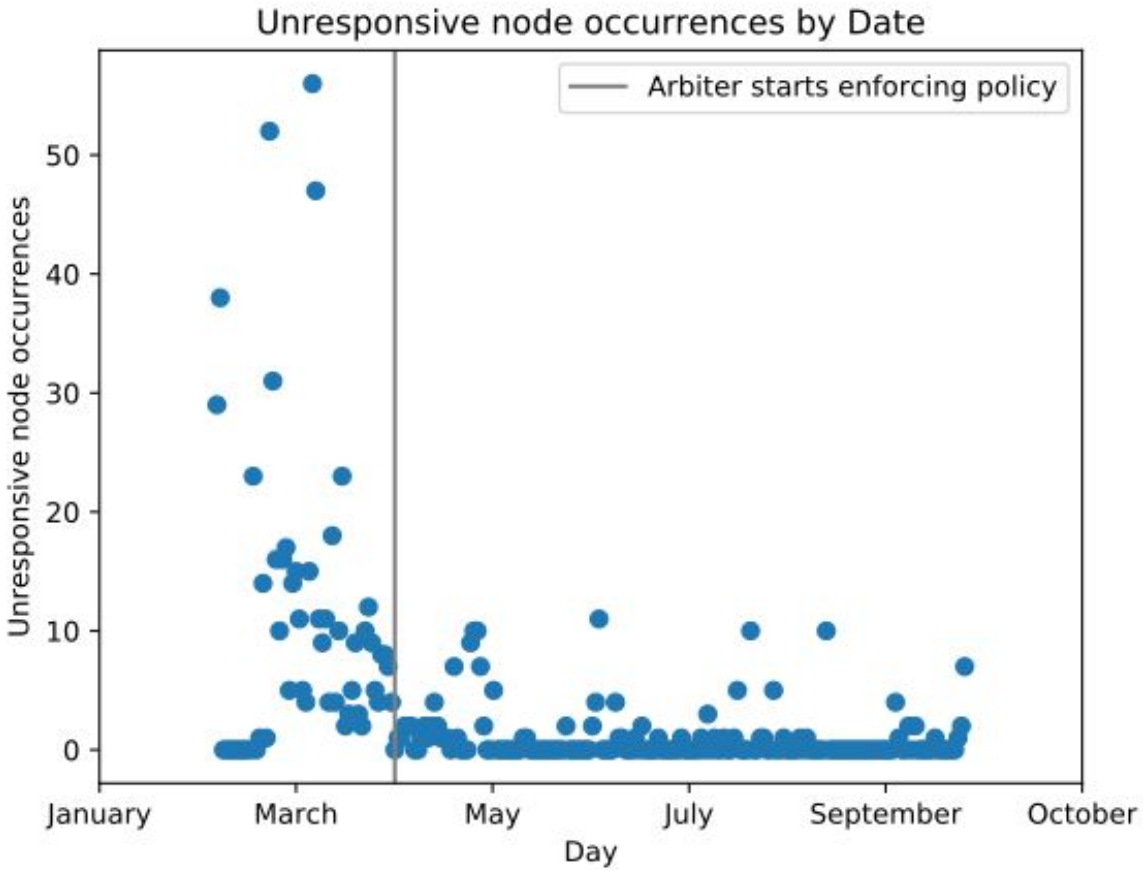
Allows for short-term usage above soft quota (threshold)



Messages to users include plots and a table of resource usage.

**Whitelisted processes (don't count against the users)*

***Unaccounted process usage (i.e. short-lived processes); It is whitelisted.*



The number of unresponsive login node events has been reduced.

Questions or comments?

dylan.gardner@utah.edu
robben.migacz@utah.edu

Code (GPLv2 licensed):

https://gitlab.chpc.utah.edu/arbiter2/arbiter2

Technical Details in Paper:

https://doi.org/10.1145/3332186.3333043

Additional Slides (online viewing)

Technical Details

- Runs as an unprivileged systemd service
  - Uses sudoers to allow for writing to cgroups
  - Optionally uses CAP_SYS_PTRACE capability for more accurate shared memory data via /proc/pid/smaps
- Written with Python 3.6+
- Runs on systemd-based distributions:
  - Developed on CentOS/RHEL 7 (but needs --rhel7-compat flag)
  - Tested on CentOS/RHEL 8, Ubuntu 15.04+
- Uses systemd's user-$UID.slice cgroup

Other solutions (static limits)

Didn’t fit our needs and policy:

- Users can run at the limit forever until admin action
  - Wanted a time component in policy (e.g. Users may use 100% CPU for 15m)
- Doesn’t penalize violators
- Doesn’t email users about violations

e.g. PAM limits, setrlimit/ulimit, static cgroup limits...

Other solutions (cgroups_py)

Solution to same problem by Purdue University.

Didn't fit our needs and policy:

- Doesn't email users about violations
- CPU limits are dependent on the number of users on the machine
  - Wanted a quantifiable and consistent policy
- Doesn't consider processes or historical usage
  - Wanted to allow usage from certain processes (compilers), up to hard default limit

https://sc18.supercomputing.org/proceedings/workshops/workshop_files/ws_hpcsysp113s3-file1.pdf

Example email

A **violation of the usage policy** by **velociraptor (A velociraptor!)** on pangaea was automatically detected starting at 15:53 on 09/17.

This may indicate that you are running computationally-intensive work on the interactive node (when it should be run on compute nodes instead).

You now have the status penalty1 because your usage has exceeded the thresholds set by the policy. Your CPU usage is now limited to 50% of your original limit (2.0 cores) for the next 1 minutes. In addition, your memory limit is 15% of your original limit (2.0 GB) for the same period of time.

High-impact processes

Usage values are recent averages. Instantaneous usage metrics may differ. The processes listed are probable suspects, but there may be some variation in the processes responsible for your impact on the node. Memory usage is expressed in GB and CPU usage is relative to one core (and may exceed 100% as a result).

Process	Avg CPU usage (%)	Avg memory usage (GB)
stress (5)	182.35	0.23
bash (1)	0.00	0.00

Resource plot is typically shown here