REST API

Nate Rini
SchedMD

Copyright 2020 SchedMD LLC
https://schedmd.com

Slurm User Group Meeting 2020

Copyright 2020 SchedMD LLC
https://schedmd.com

Agenda

All times are US Mountain Daylight (UTC-6)

Time	Speaker	Title
9:00 - 9:50	Jason Booth	Field Notes 4: From The Frontlines of Slurm Support
10:00 - 10:25	Brian Christiansen	Cloud and Stuff
10:30 - 10:50	Nate Rini	REST API
11:00 - 11:50	Tim Wickberg	Slurm 20.11 and Beyond, Open Q+A

Copyright 2020 SchedMD LLC
https://schedmd.com

Welcome

- Four separate presentations, four separate streams
- Presentations will remain available for one week after SLUG'20 concludes
- Presentations are available through the SchedMD Slurm YouTube channel
  - https://youtube.com/c/schedmdslurm
- Or through direct links from the agenda
  - https://slurm.schedmd.com/slurm_ug_agenda.html

Copyright 2020 SchedMD LLC
https://schedmd.com

Asking questions

- Feel free to ask questions throughout through YouTube's chat
- Chat is moderated by SchedMD
- Questions will be relayed to the presenter by the moderators
  - Some may be deferred to the end if they cannot be relayed in a timely fashion
- Note there is a ~5 second broadcast delay
  - By the time you've asked your question, the presenter may have moved ahead to a different topic, and may defer the question until the end

Copyright 2020 SchedMD LLC
https://schedmd.com

REST API

Nate Rini
SchedMD

Copyright 2020 SchedMD LLC
https://schedmd.com

Contents

- What is REST API
- What is OpenAPI
- What is the Slurm REST API
- Security
- Authentication
- Changes for Slurm 20.11
- Examples
- How to experiment with Scaleout

Copyright 2020 SchedMD LLC
https://schedmd.com

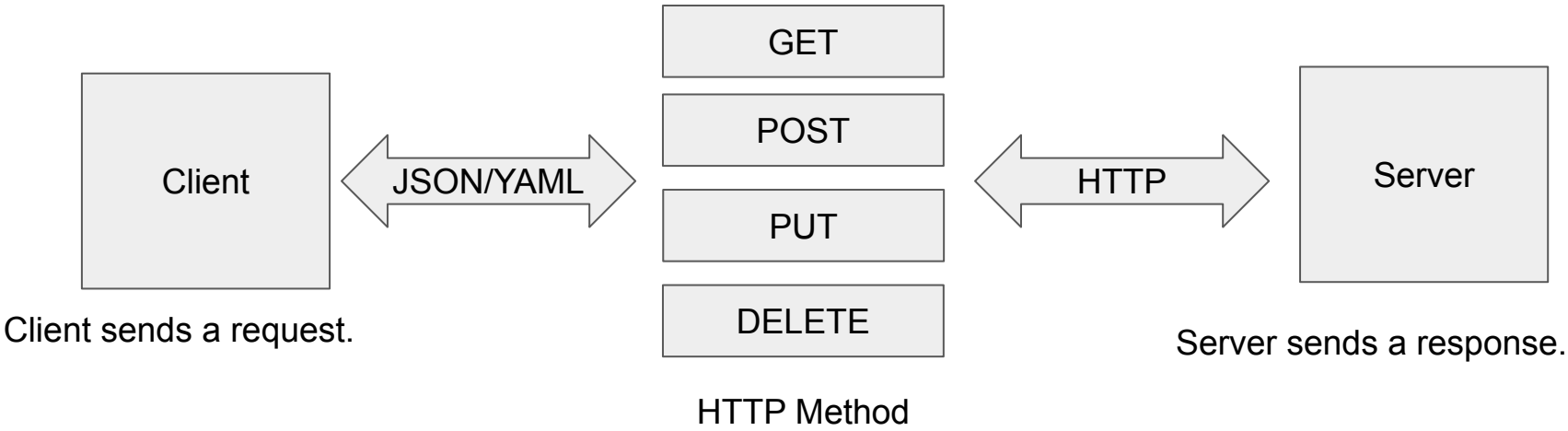
What is REST API

“REST is acronym for REpresentational State Transfer. It is architectural style for distributed hypermedia systems and was first presented by Roy Fielding in 2000 in his famous dissertation.” --https://restfulapi.net/

- Guiding Principles of REST
- Client–server
- Stateless
- Cacheable
- Uniform interface
- Layered system
- Code on demand (optional)

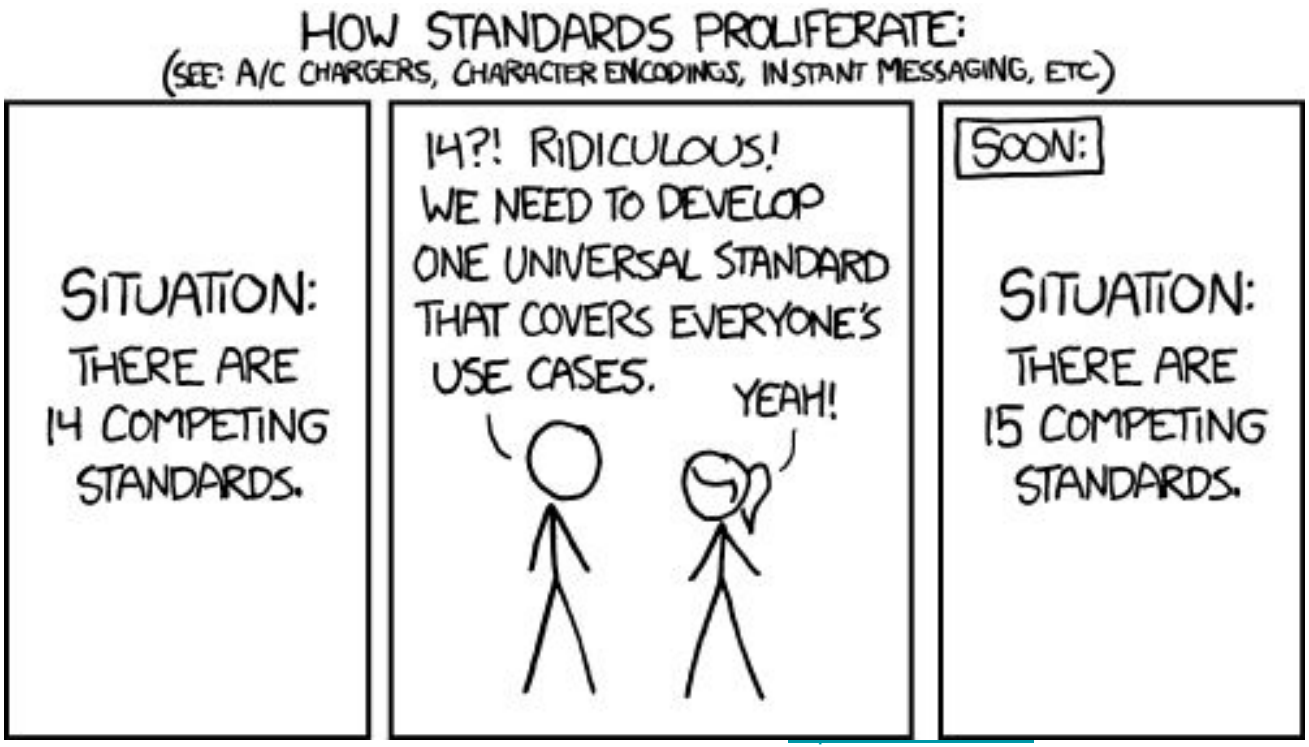
Copyright 2020 SchedMD LLC
https://schedmd.com

REST API Flow



Copyright 2020 SchedMD LLC
https://schedmd.com

REST Standards



https://xkcd.com/927/

Slurm's REST API is designed to be as compliant as possible with the relevant REST and IETF standards.

What is OpenAPI (aka Swagger)

OpenAPI Specification allows you to describe your entire REST API, including:

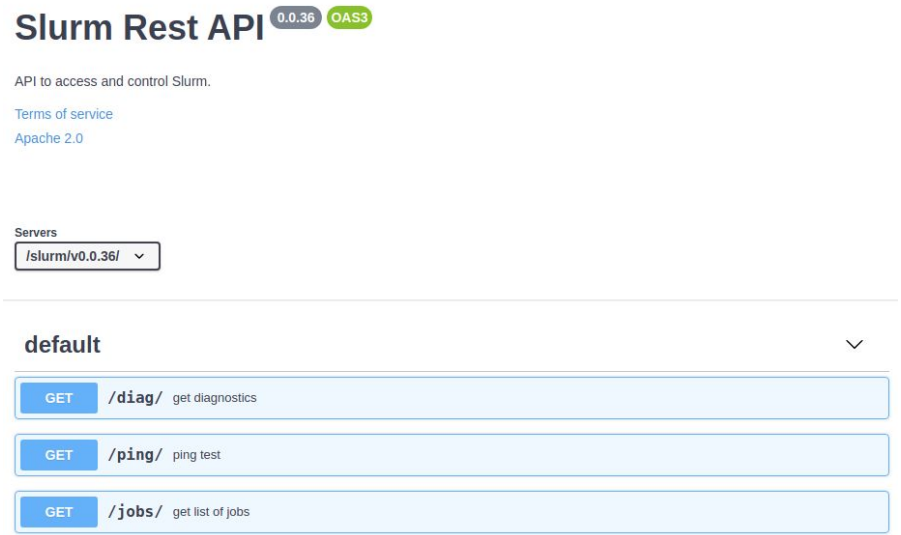
- Available endpoints (/users) and operations on each endpoint (GET /users, POST /users)
- Operation parameters Input and output for each operation
- Authentication methods
- Written in YAML or JSON.
- Readable to both humans and machines.

https://swagger.io/docs/specification/about/

Copyright 2020 SchedMD LLC
https://schedmd.com

Why OpenAPI?

- **Relatively** simple specification for what your program will receive and return
- Free (and some less free) tools that will work with the specification
- Pretty documentation automatically generated:



OpenAPI Client generators

Here are some 3rd party clients that can be used to generate OpenAPI clients:

- https://openapi-generator.tech/
- https://swagger.io/tools/swagger-codegen/

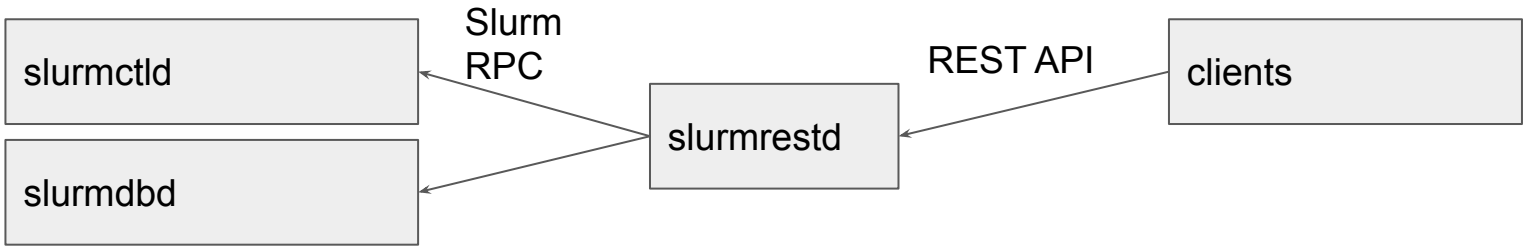
- https://openapi.tools/#sdk (larger list here)

SchedMD does not provide or support
any specific OpenAPI client.

Copyright 2020 SchedMD LLC
https://schedmd.com

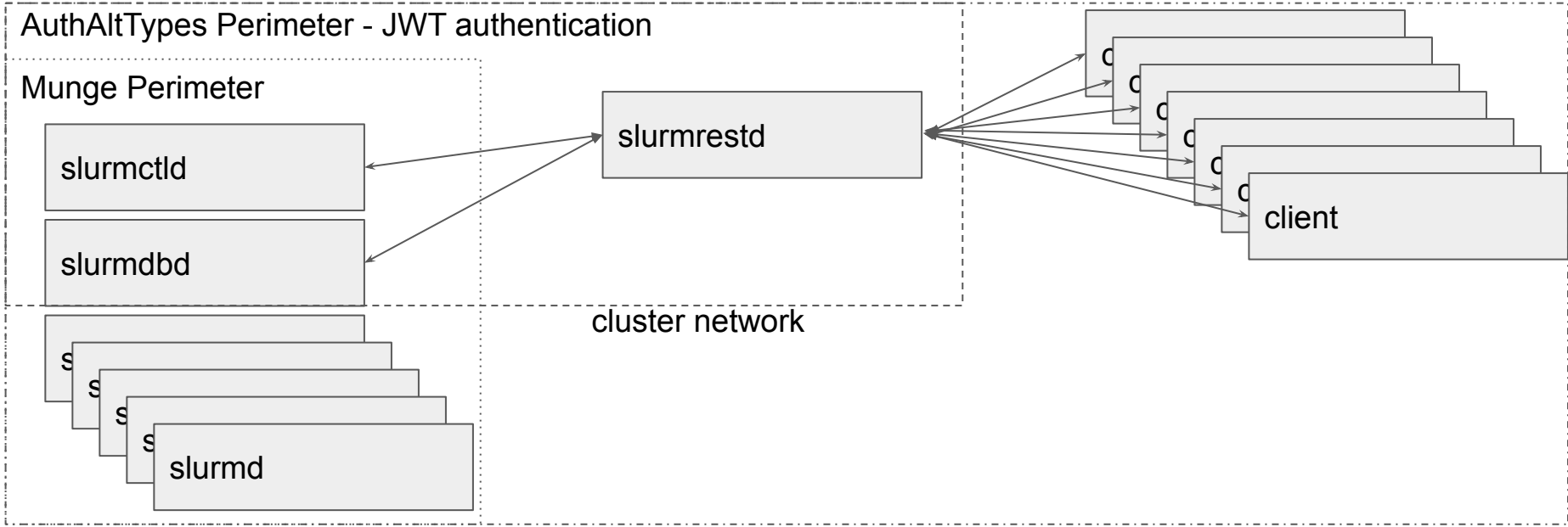
What is the Slurm REST API

A tool that will translate JSON/YAML over HTTP requests into Slurm RPC requests.



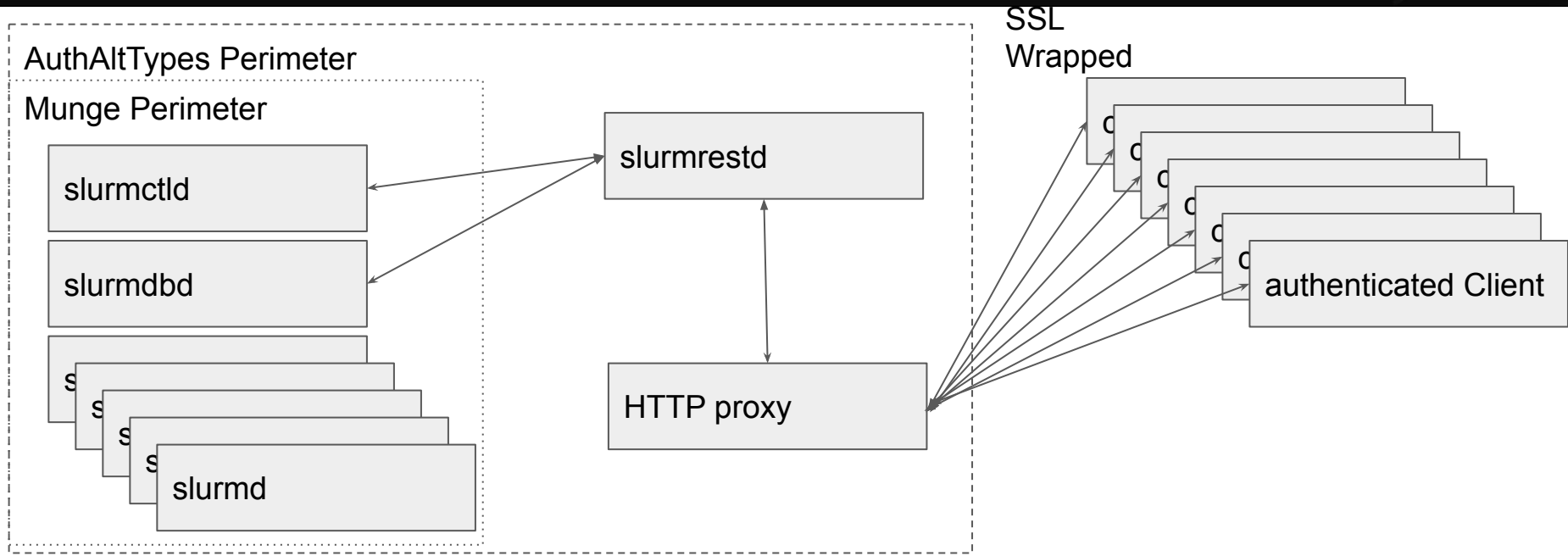
Copyright 2020 SchedMD LLC
https://schedmd.com

Slurm REST API Architecture (rest_auth/jwt)



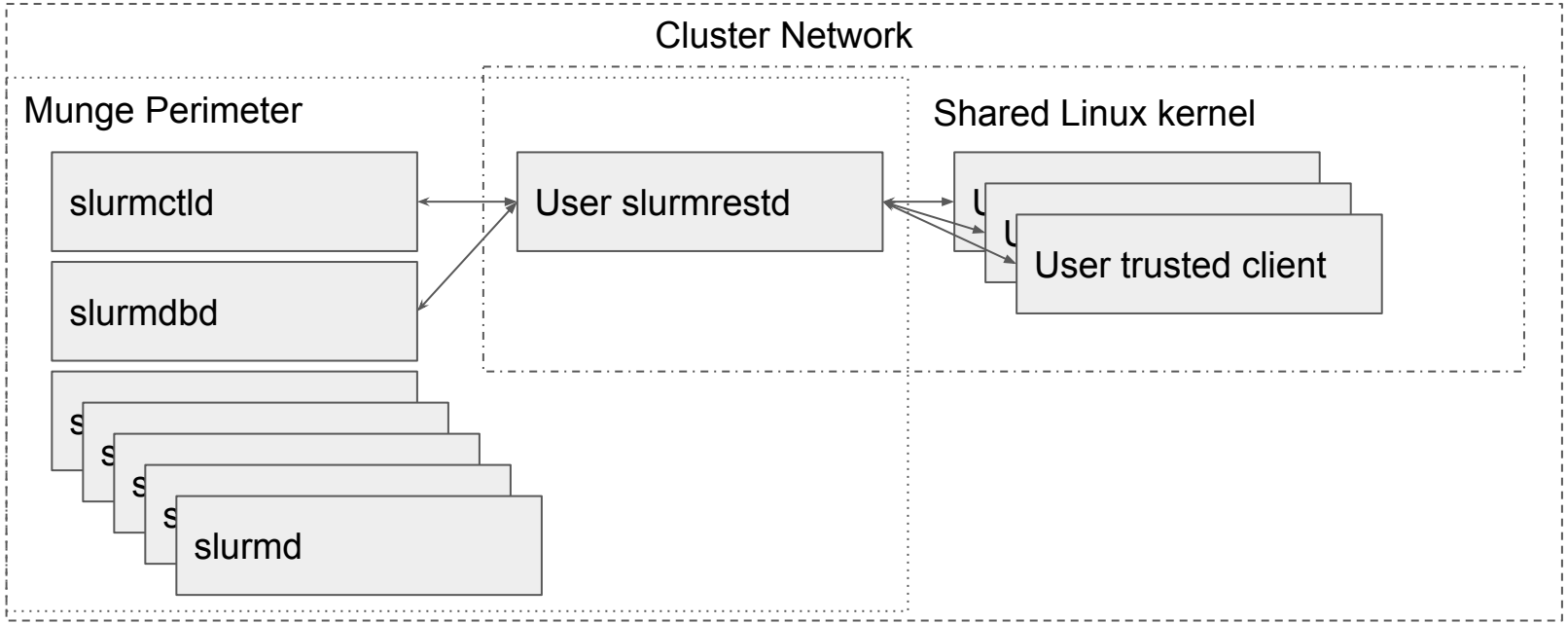
Copyright 2020 SchedMD LLC
https://schedmd.com

Slurm REST API Architecture (rest_auth/jwt + Proxy)



Copyright 2020 SchedMD LLC
https://schedmd.com

Slurm REST API Architecture (rest_auth/local)



Copyright 2020 SchedMD LLC
https://schedmd.com

slurmrestd - REST API daemon

- Daemon Mode
  - Listen on given set of IP:PORTs for client connections.
- Inet Mode
  - Single user connected via STDIN and STDOUT.
    - Started per connection by xinetd, inetd, or systemd socket activation.
  - Script send commands via PIPE directly.
- Any user can run the daemon.
- Execution User:
  - unprivileged - All commands run as execution user
  - privileged user - All commands can run as requested users

Design

1. slurmrestd acts as a framework for requests.
   a. Plugin based addition of new REST methods via registering URL paths.
2. Built in data type to translate JSON and YAML.
3. Built in support for generating OpenAPI specification (per plugin).
4. Conformant HTTP server to handle interfacing with clients and proxies.

Copyright 2020 SchedMD LLC
https://schedmd.com

Inet Mode Example (unprivileged user - rest_auth/local)

- $ echo -e "GET /slurm/v0.0.36/diag HTTP/1.1\r\n" | slurmrestd -a rest_auth/local
- slurmrestd: operations_router: /slurm/v1/diag for pipe:[722494]
- HTTP/1.1 200 OK
- Content-Length: 973
-
- {
- "parts_packedg": 1,
- "req_timeg": 1567712456,

… JSON continues ...

Listen Mode Example (unprivileged user - rest_auth/jwt)

- $ slurmrestd -vvv -a rest_auth/jwt localhost:9997 [::1]:7272 10.10.0.1:38484
- slurmrestd: debug: Reading slurm.conf file: /etc/slurm.conf
- slurmrestd: debug: Interactive mode activated (TTY detected on STDIN)
- slurmrestd: debug: listen: localhost:9997 fd: 3
- slurmrestd: debug: _socket_thread_listen: thread for fd: 3
- slurmrestd: debug: listen: ip6-localhost:7272 fd: 4
- slurmrestd: debug: _socket_thread_listen: thread for fd: 4
- slurmrestd: debug: listen: spheron:38484 fd: 5
- slurmrestd: debug: server listen mode activated
- slurmrestd: debug: _socket_thread_listen: thread for fd: 5

YAML Mode Example

- $ echo -e "GET http://localhost/slurm/v0.0.36/diag HTTP/1.1\r\nAccept: text/yaml\r\n" | slurmrestd -a rest_auth/local
- HTTP/1.1 200 OK
- Content-Length: 1210
- %YAML 1.1
- ---
- !!str parts_packedg: !!int 1

  …. YAML Continues ….

Security

- Built in Authentication plugins
  - rest_auth/local:
    - Only works over local unix named sockets (use unix:/path/to/socket) or pipes.
      - Rejects remote connections
    - Privileged user: use Linux kernel to determine client's user and will run as user.
    - Unprivileged user: use Linux kernel to determine client's user is same user.
      - Rejects connections from different users
      - Be aware of user namespaces
        mimicking same user.
  - rest_auth/jwt:
    - All remote clients must be authenticated
      via HTTP headers:
      - X-SLURM-USER-TOKEN
      - X-SLURM-USER-NAME

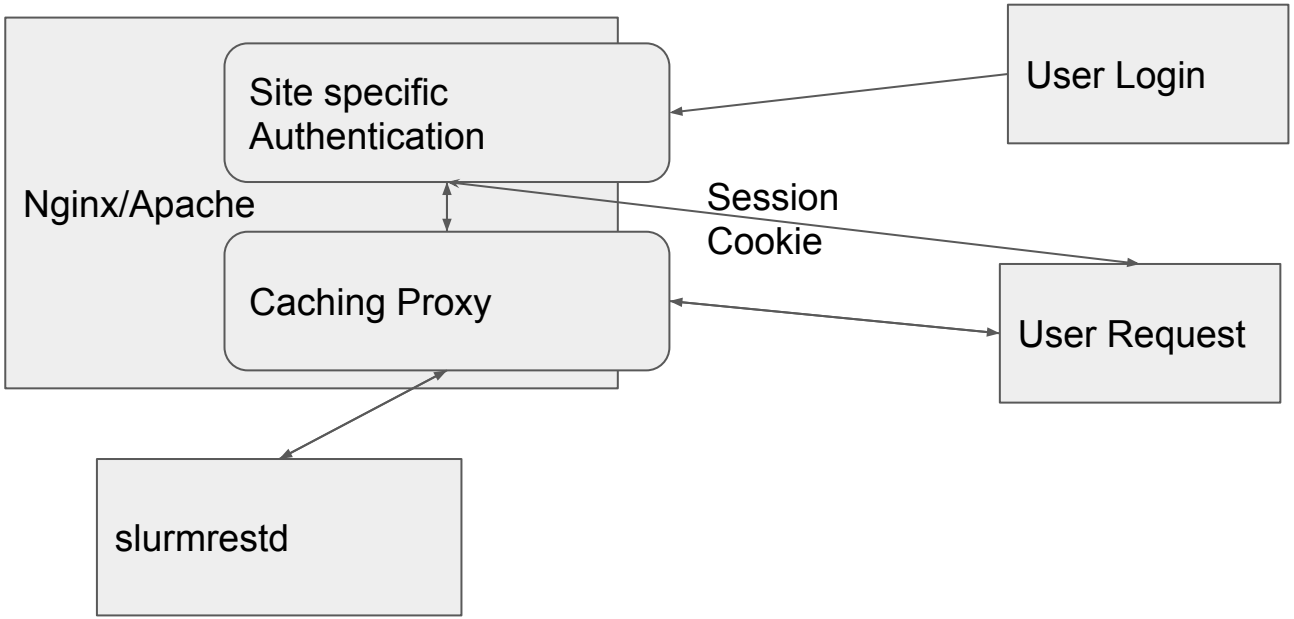
Site specific security policy

- Sites can add their own authentication plugins.
  - Site authentication plugins can be stacked with existing plugins
  - Authentication plugins can examine queries to apply site ACLs
- Sites can offload to an authenticating proxy
  - Apache: mod_proxy_http
  - NGINX: auth_request
    - Working example provided in scaleout
- Sites can add TLS with an authenticating proxy

rest_auth/jwt Authentication Demo (in Scaleout)

- $ make HOST=login bash
- $ su - fred
- $ export $(scontrol token lifespan=9999)
- $ curl -s -H X-SLURM-USER-NAME:$(whoami) -H X-SLURM-USER-TOKEN:$SLURM_JWT -X POST 'http://rest/slurm/v0.0.36/ping'
  - {JSON replied}

Authenticating Proxy Architecture



Copyright 2020 SchedMD LLC
https://schedmd.com

Authenticating Proxy Demo (in Scaleout)

- $ make HOST=login bash
- $ su - fred
- $ curl -c /tmp/cookiejar 'http://proxy:8080/auth/?user=fred'
  - Hello fred. Using slurm user for authentication proxy.
- curl -b /tmp/cookiejar 'http://proxy:8080/slurm/v0.0.36/ping'
  - {JSON response}

Changes for 20.11

- Switch to plugins
  - Authentication via plugins
    - Explicitly choose which plugin instead of attempting to determine at runtime
    - Sites can now stack authentication plugins
      - Site specific authentication plugins can replace or add to existing plugins

```
$ slurmrestd -a list
slurmrestd: Possible REST authentication plugins:
slurmrestd: rest_auth/local
slurmrestd: rest_auth/jwt
```

Copyright 2020 SchedMD LLC
https://schedmd.com

Changes for 20.11

- Switch to plugins (continued)
  - Content via plugins
    - All content is now provided by plugins
    - Sites can now append or replace content plugins
    - Only one plugin can bind to any given URL path at a time (per OpenAPI)

```
$ slurmrestd -s list
slurmrestd: Possible OpenAPI plugins:
slurmrestd: openapi/v0.0.36
slurmrestd: openapi/dbv0.0.36
slurmrestd: openapi/v0.0.35
```

Copyright 2020 SchedMD LLC
https://schedmd.com

Changes for 20.11

- Versioning schema
  - Based on use of plugins
    - Each Slurm builtin plugin is assigned a unique path
      - /slurm/v0.0.35/
      - /slurm/v0.0.36/
      - /slurmdb/v0.0.36/

Copyright 2020 SchedMD LLC
https://schedmd.com

Changes for 20.11

- Add accounting access **[NOT FINAL: currently being added]**
  - View job and step accounting (sacct)
    - Conditional filters supported to avoid downloading whole job database at once.
  - View QOS, accounts, associations, TRES, wckeys, and users (sacctmgr)
  - Add/Remove/Update accounts, associations, and users (sacctmgr)
  - Note: that sreport functionality has **not** been added.

Copyright 2020 SchedMD LLC
https://schedmd.com

Changes for 20.11

- Add slurm/v0.0.36/ plugin (Thanks to Andrew Bruno)
  - Multiple changes made per bug#9131 to make auto generated clients easier
  - Avoid use of “oneOf” in specification to simplify data types
  - Add “operationId” to generate human friendly functions
  - Add content response definitions
    - Allows clients to have built in ability to view results
  - Avoid dictionary for responses
    keyed by name but instead use lists

Copyright 2020 SchedMD LLC
https://schedmd.com

Example Queries

- Job submission (sbatch)
- Heterogeneous Job ("HetJob") submission
- Array Job Submission
- Cancelling job (scancel)
- Viewing jobs (scontrol show job)
- slurmctld diagnostics (sdiag)

HTTP/JSON/YAML have been compacted or even trimmed to fit on the slides. Examples are to show what is possible. Please see generated openapi reference for exact implementation documentation.

Job submission example (sbatch)

- $ cat job.json

{"job":{"tasks":1,"name":"test","nodes":1,"current_working_directory":"/tmp/","environment":{"PATH":"/bin:/usr/bin/:/usr/local/bin/","LD_LIBRARY_PATH":"/lib/:/lib64/:/usr/local/lib"}},"script":"#!/bin/bash\nsrun hostname"}

- $ curl -s -H X-SLURM-USER-NAME:$(whoami) -H X-SLURM-USER-TOKEN:$SLURM_JWT -X POST 'http://rest/slurm/v0.0.36/job/submit' -H "Content-Type: application/json" -d @job.json

{"errors":[],"job_id":3,"step_id":"BATCH","job_submit_user_msg":""}

Job submission request example (sbatch)

- $ cat demo
- POST /slurm/v0.0.36/job/submit HTTP/1.1
- Accept: text/yaml
- Content-Type: application/json
- Content-Length: 348
-
- {"job": {"account": "test", "ntasks": 20, "name": "test18.1", "nodes": [2, 4], "current_working_directory": "/tmp/", "user_id": 1000, "group_id": 1000, "environment": {"PATH": "/bin:/usr/bin/:/usr/local/bin/","LD_LIBRARY_PATH": "/lib/:/lib64/:/usr/local/lib"} }, "script": "#!/bin/bash\necho it works"}

Job submission example (sbatch)

- $ slurmrestd 2>/dev/null < demo
- HTTP/1.1 200 OK
- Content-Length: 131
- %YAML 1.1
- ---
- !!str errors: !!seq []
- !!str job_id: !!int 115
- !!str step_id: !!str BATCH
- !!str job_submit_user_msg: !!null null
- …
- $ squeue -j 115
- JOBID PARTITION USER ST TIME NODES NODELIST(REASON)
- 115 debug nate PD 0:00 4 (Priority)

Job submission request example (sbatch HetJob)

- $ cat demo
- POST /slurm/v1/job/submit HTTP/1.1
- Accept: text/yaml
- Content-Type: application/json
- Content-Length: 654
- {"job": [{"account": "test","ntasks": 20,"name": "test18.1","nodes": [2, 4],"current_working_directory": "/tmp/","user_id": 1000,"group_id": 1000,"environment": {"PATH": "/bin:/usr/bin/:/usr/local/bin/", "LD_LIBRARY_PATH": "/lib/:/lib64/:/usr/local/lib" }}, { "account": "test", "ntasks": 2, "name": "test18.3", "nodes": [2, 4], "current_working_directory": "/tmp/", "user_id": 1000, "group_id": 1000, "environment": { "PATH": "/bin:/usr/bin/:/usr/local/bin/", "LD_LIBRARY_PATH": "/lib/:/lib64/:/usr/local/lib" }}], "script": "#!/bin/bash\nsrun echo it works" }

Job submission example (sbatch HetJob)

- $ slurmrestd < demo
- HTTP/1.1 200 OK
- Content-Length: 131
-
- %YAML 1.1
- ---
- !!str errors: !!seq []
- !!str job_id: !!int 118
- !!str step_id: !!str BATCH
- !!str job_submit_user_msg: !!null null
- ...

Job submission example (sbatch HetJob)

- $ squeue -j 118
- JOBID PARTITION USER ST TIME NODES NODELIST(REASON)
- 118+0 debug nate PD 0:00 4 (None)
- 118+1 debug nate PD 0:00 4 (None)

Job submission request example (sbatch HetJob)

- $ cat demo
- POST /slurm/v0.0.36/job/submit HTTP/1.1
- Accept: text/yaml
- Content-Type: application/json
- Content-Length: 374
- {"job":{"account":"test","array":"1-1000","ntasks":20,"name":"test18.1","nodes":[2,4],"current_working_directory":"/tmp/","user_id":1000,"group_id":1000,"environment":{"PATH":"/bin:/usr/bin/:/usr/local/bin/","LD_LIBRARY_PATH":"/lib/:/lib64/:/usr/local/lib"}},"script":"#!/bin/bash\nsrun echo it works"}

Job submission example (sbatch array)

- $ slurmrestd < demo
- HTTP/1.1 200 OK
- Content-Length: 132
-
- %YAML 1.1
- ---
- !!str errors: !!seq []
- !!str job_id: !!int 1202
- !!str step_id: !!str BATCH
- !!str job_submit_user_msg: !!null null
- ...

Job submission example (sbatch array)

- $ squeue -j 1202
- JOBID PARTITION USER ST TIME NODES NODELIST(REASON)
- 1202_[528-1000] debug nate PD 0:00 4 (Resources)
- 1202_527 debug nate R 0:00 4 host[5-8]
- 1202_526 debug nate R 0:00 4 host[1-4]

Cancelling job example (scancel)

- $ cat demo
- DELETE /slurm/v0.0.36/job/2202 HTTP/1.1
- Accept: text/yaml
- $ slurmrestd < demo
- HTTP/1.1 200 OK
- Content-Length: 41
- %YAML 1.1
- ---
- !!str errors: !!seq []
- ...
- $ scontrol show job 2202|grep JobState
- JobState=CANCELLED Reason=None Dependency=(null)

View job example (scontrol show job)

- $ echo -e 'GET /slurm/v0.0.36/job/2203 HTTP/1.1\r\n' | slurmrestd
- {"account":"test","accrue_time":1568158776,"admin_comment":"","array_job_id":0,"array_task_id":{},"array_max_tasks":0,"array_task_str":"","assoc_id":4,"batch_features":"","batch_flag":true,"batch_host":"host1","bitflags":["JOB_WAS_RUNNING"],"boards_per_node":0,"burst_buffer":"","burst_buffer_state":"","cluster":"linux","cluster_features":"","command":"","comment":"","contiguous":false,"core_spec":{},"thread_spec":{},"cores_per_socket":{},"billable_tres":12,"cpus_per_task":{},"cpu_freq_min":{},"cpu_freq_max":{},"cpu_freq_gov":{},"cpus_per_tres":"","deadline":0,"delay_boot":0,"dependency":"","derived_ec":0,"eligible_time":1568158776,"end_time":1599694776,"exc_nodes":"","execution_node_by_index":[],"exit_code":0,"features":"","fed_origin_str":"","fed_siblings_active":0,"fed_siblings_active_str":"","fed_siblings_viable":0,"fed_siblings_viable_str":"","gres_detail":[],"group_id":1000,"job_id":2203,"job_resources":{},"job_state":"RUNNING","last_sched_eval":1568158776,"licenses":"","max_cpus":0,"

Diagnostics query example (sdiag)

- echo -e 'GET /slurm/v0.0.36/diag HTTP/1.1\r\n' | slurmrestd
- {"parts_packedg":1,"req_timeg":1568158683,"req_time_startg":1568153903,"server_thread_count":3,"agent_queue_size":0,"agent_count":0,"dbd_agent_queue_size":0,"gettimeofday_latency":18,"schedule_cycle_max":12041,"schedule_cycle_last":18,"schedule_cycle_sum":1008716,"schedule_cycle_counter":1138,"schedule_cycle_depth":5109,"schedule_queue_len":0,"jobs_submitted":2130,"jobs_started":2083,"jobs_completed":2082,"jobs_canceled":54,"jobs_failed":0,"jobs_pending":0,"jobs_running":0,"job_states_ts":1568158674,"bf_backfilled_jobs":1,"bf_last_backfilled_jobs":1,"bf_backfilled_pack_jobs":0,"bf_cycle_counter":130,"bf_cycle_sum":37282,"bf_cycle_last":65,"bf_cycle_max":690,"bf_last_depth":1,"bf_last_depth_try":1,"bf_depth_sum":3356,"bf_depth_try_sum":260,"bf_queue_len":0,"bf_queue_len_sum":0,"bf_when_last_cycle":1568158222,"bf_active":0}

Experimentation with Scaleout

- What is Scaleout
  - Docker compose “pod” designed to replicate a working Slurm cluster
  - Simulates a full cluster on your laptop (just don’t try to run real jobs)
  - Cluster is isolated but will have usual outbound NATed communications
  - Setup with suggested configuration for slurmrestd as an example
- Download here: https://gitlab.com/SchedMD/training/docker-scale-out
  - Make sure to pick a branch: master or slurm-20.02
  - Read the README
    - docker-compose can be picky about versions

Copyright 2020 SchedMD LLC
https://schedmd.com

Interacting with Scaleout

- To compile and start cluster (first time is very slow):
  - $ make
- To be root on the controller:
  - $ make bash
- To be root on the login node:
  - $ make HOST=login bash
- To stop scaleout
  - $ make stop

Copyright 2020 SchedMD LLC
https://schedmd.com

Questions?

Copyright 2020 SchedMD LLC
https://schedmd.com

Next Session

- The next presentation is by Tim Wickberg with "Slurm 20.11 and Beyond"
- Starts at 11am Mountain Daylight Time (UTC-6)
- And is on a separate YouTube Live stream
- Please see the SchedMD Slurm YouTube channel for links

Copyright 2020 SchedMD LLC
https://schedmd.com

End Of Stream

- Thanks for watching!

Copyright 2020 SchedMD LLC
https://schedmd.com