universität freiburg

HTC Accounting with AUDITOR

Dirk Sammel
on behalf of the AUDITOR team

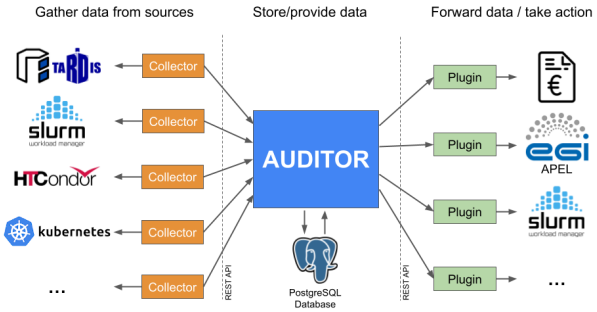
17.09.2025
HTCondor Workshop Prague

Bundesministerium
für Bildung
und Forschung

IDIUM

- What is AUDITOR?
- HTCondor collector
- Current usage
- Latest features
- What's next?

- **AUDITOR** is a new, multi-purpose accounting ecosystem
- Main usage: accounting of HTC resources for WLCG (ATLAS, CMS, ...)
- Wide range: "small" Tier-2 sites (e.g. Freiburg) $\rightarrow$ CERN
- Solved: accounting for sub-clusters in Overlay Batch System



Current version: 0.10.0

https://github.com/ALU-Schumacher/AUDITOR/releases/tag/v0.10.0

also available in WLCG repository

Modular accounting ecosystem

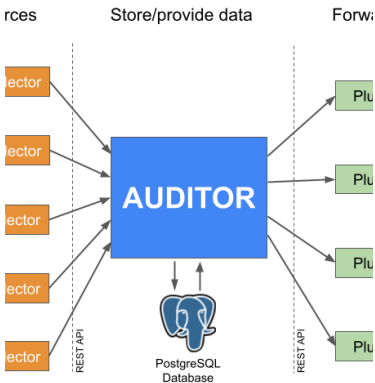
- **Collectors**
  - Accumulate data
- **Core component**
  - Accept data
  - Store data
  - Provide data
- **Plugins**
  - Take action based on stored data

Code & Documentation:

https://github.com/ALU-Schumacher/AUDITOR
Documentation
AUDITOR. *Eur. Phys. J. C* **85**, 265 (2025)

Demo: accounting pipeline from scratch

https://github.com/ALU-Schumacher/auditor-demo



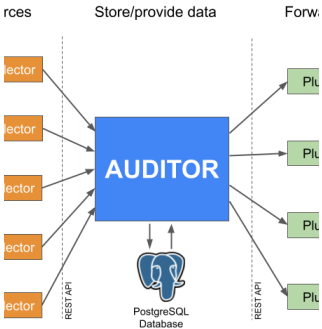
- Implemented in **Rust**
  - Access via REST interface
- Data stored in PostgreSQL
- Completely stateless
  - No dataloss
  - Suitable for high availability setups
- Provided as **RPM** or **Docker container**
- Client libraries in Rust and Python
  - pyauditor documentation: https://alu-schumacher.github.io/AUDITOR/pyauditor
- Unit of accountable resources: **Record**

- `record_id`: uniquely identifies the record
- `meta`: multiple key-value pairs of the form `String -> [String]`
- `components`: arbitrary number of resources that are to be accounted for (CPU, RAM, Disk, GPU, ...)
  - `scores`: (multiple) benchmark scores supported
- `start_time`, `stop_time`: datetime in UTC
- `runtime`: calculated as `stop_time - start_time`

- `meta` & `component` fields allow for maximal flexibility
- In environments with mixed hardware:
  individual accounting of different CPU types/corepower values

```
{
    "record_id": "hpc-4126142",
    "meta": {
      "group_id": [ "atlpr" ],
      "site_id": [ "hpc" ],
      "user_id": [ "atlpr001" ]
    },
    "components": [
      {
        "name": "Cores",
        "amount": 8,
        "scores": [
          {
            "name": "HEPSPEC06",
            "value": 10.0
          },
          {
            "name": "HEPScore23",
            "value": 10.0
          }
        ]
      },
      {
        "name": "Memory",
        "amount": 16000,
        "scores": []
      }
    ],
    "start_time": "2023-02-24T00:27:58Z",
    "stop_time": "2023-02-24T03:41:35Z",
    "runtime": 11617
},
```

- **TARDIS Collector**
  - Collects drone information
- **SLURM Collectors**
  - Collects information about SLURM jobs via SLURM CLI commands
- **Kubernetes Collector** (dev @ Wuppertal)
  - Accounts resources from kubernetes clusters
- **HTCondor Collector** (dev @ KIT)
  - Collects information from job ClassAds via condor_history



- **Priority plugin**
  - Adjusts user/group priority in batch system
- **APEL accounting plugin**
  - Reports accounting data to the WLCG accounting service (APEL)

Most important collector for WLCG accounting: **HTCondor collector**

- Developed at KIT, main developer Florian von Cube
- Written in Python → uses pyauditor client library
- Get job ClassAds and convert into AUDITOR record structure
- Constructs and executes condor_history command
- New features introduced with v0.10.0
  - constraint: ClassAd filter expression for condor_history, collect only jobs for which this expression is true (Example: exclude some VOs)
  - history_file: read HTCondor history from specified file
- Upcoming feature: -search to read multiple history files
- Why not HTCondor Python bindings?
  - Was planned, but can't use with new history_file feature (?)

```yaml
addr: localhost
port: 8000
timeout: 10
state_db: /var/lib/auditor_htcondor_collector/htcondor_history_state.db
record_prefix: htcondor
interval: 900 # 15 minutes
pool: htcondor.example.com
schedd_names:
  - schedd1.example.com
  - schedd2.example.com
job_status: # See https://htcondor-wiki.cs.wisc.edu/index.cgi/wiki?p=MagicNumbers
  - 3 # Removed
  - 4 # Completed
```

- state_db: SQLite database used for persistent storage of the job ids last processed by the HTCondor collector
- list of schedds → multiple invocations of condor_history

```yaml
components:
  - name: "Cores"
    key: "CpusProvisioned"
    scores:
      - name: "HEPSPEC"
        key: "MachineAttrApelSpecs0"
        matches: HEPSPEC\D+(\d+(\.\d+)?)  # This regex matches the value of HEPSPEC in the corresponding ClassAd
        only_if:
          key: "LastRemoteHost"
          matches: ^slot.+@(?:site1)-.{10}@.+$  # This score is only attributed to the component on site1
      - name: "HEPscore23"
        key: "MachineAttrApelSpecs0"
        matches: HEPscore23\D+(\d+(\.\d+)?)
        only_if:
          key: "LastRemoteHost"
          matches: ^slot.+@(?:site1)-.{10}@.+$
  - name: "Memory"
    key: "MemoryProvisioned"
  - name: "UserCPU"
    key: "RemoteUserCpu"
```

- name: name in AUDITOR record
- key: name of job ClassAd

```yaml
meta:
  user:
    key: Owner
    matches: ^(.+)$
  group:
    key: VoName
    matches: ^(.+)$
  submithost:
    key: "GlobalJobId"
    matches: ^(.*)#\d+.\d+#\d+$  # As this regex contains a group, the value for 'submithost' is set to the matching group.

  # For `site` the first match is used.
  site:
    - name: "site1"  # This entry
      key: "LastRemoteHost"
      matches: ^slot.+@site1-.+$
    - key: "LastRemoteHost"
      matches: ^slot.+@(site2)-.+$  # This regex contains a group, the value for 'site' is set to the matching group ("site2").
    - name: "UNDEF"  # If no match is found, site is set to "UNDEF"
```

- key: name of job ClassAd

```
-h, --help            show this help message and exit
-c CONFIG_FILE, --config CONFIG_FILE
                      Path to config file.
-j <CLUSTERID>[.<PROCID>], --job-id <CLUSTERID>[.<PROCID>]
                      ID of the job, condor_history to invoke with.
-n SCHEDD, --schedd-names SCHEDD
                      Name of the schedd, condor_history to invoke with.
-k HISTORY_FILE, --history-file HISTORY_FILE
                      Path to history file, to read condor_history from.
-l {DEBUG,INFO,WARNING,ERROR,CRITICAL}, --log-level {DEBUG,INFO,WARNING,ERROR,CRITICAL}
                      Log level. Defaults to INFO.
-f LOG_FILE, --log-file LOG_FILE
                      Log file. Defaults to stdout.
-i INTERVAL, --interval INTERVAL
                      Interval in seconds between queries. Defaults to 900.
-1, --one-shot        Run once and exit.
```

- CLI arguments override values from config file
- -j: start with this job
- -k: path to history file

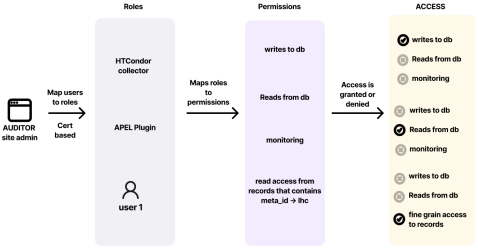
Current/Future Users

BERGISCHE
UNIVERSITÄT
WUPPERTAL

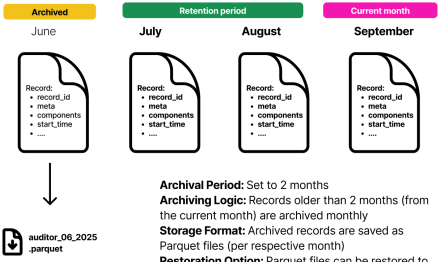
After extensive testing, WLCG came to the conclusion:

"..WLCG Operations Coordination will encourage WLCG sites to adopt AUDITOR as a generic site-level accounting solution.." (WLCG/HSF Workshop 2025)

Role Based Access Control

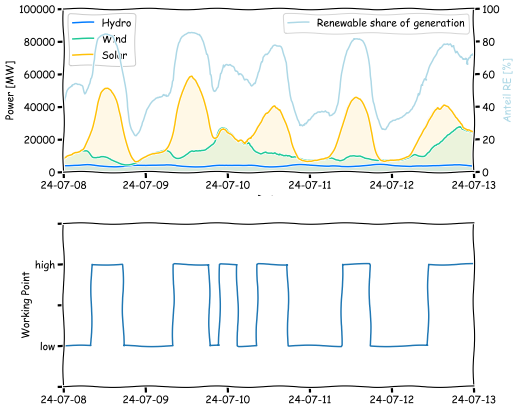


Archiving



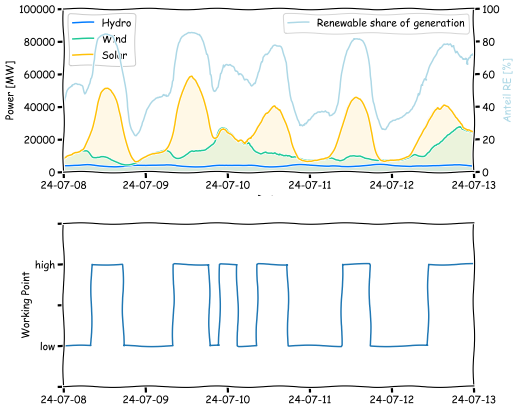
Archival Period: Set to 2 months
Archiving Logic: Records older than 2 months (from the current month) are archived monthly
Storage Format: Archived records are saved as Parquet files (per respective month)
Restoration Option: Parquet files can be restored to the Auditor database using provided scripts

Poster at ACAT 2025 Hamburg



Possible scenario to safe energy: CPU throttling

- Define 2 or more working points for high- and low performance $\widehat{=}$ energy consumption
- Switch working points according to available energy mix



Possible scenario to safe energy: CPU throttling

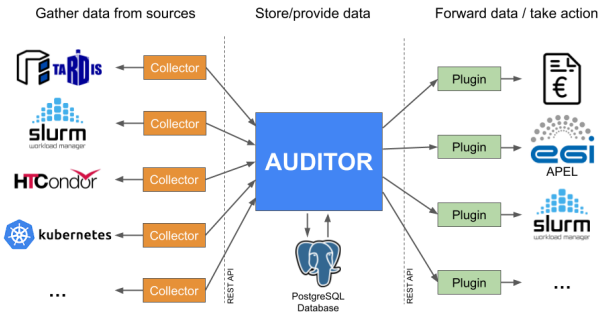
- Define 2 or more working points for high- and low performance $\widehat{=}$ energy consumption
- Switch working points according to available energy mix

- Different working points will have different corepower values
- Store switching points with different corepower values in records
- Derive proper normalised CPU time

- Current status: file on disk with W/vCore value
- Updated every 5 minutes
- How to get/extend lists of timestamps and W/vCore values into job ClassAd?
- condor_chirp? Would need to hack condorjob.sh script created by ARC-CE
- Any other/better options?

Conclusion

- AUDITOR provides an accounting ecosystem for various use cases
  - Individual T1/T2 sites
  - Sub-clusters in OBS
  - Already in production!
- HTCondor collector: very important part of the AUDITOR ecosystem
- Open questions
  - Python bindings and history file
  - Dynamically update job ClassAd with value from file



References

Website: https://alu-schumacher.github.io/AUDITOR/

GitHub: https://github.com/ALU-Schumacher/AUDITOR/

FIDIUM: https://fidium.erumdatahub.de

Email: auditor@physik.uni-freiburg.de

Dirk Sammel
Albert-Ludwigs-Universität Freiburg
dirk.sammel@physik.uni-freiburg.de