Nov 2025

/proc/energy

Didi Hoffmann

WHO AM I?

- Geerd-Dietger Hoffmann = Didi
- CTO at Green Coding Solutions
- Love coding
- Farmer

GREEN CODING;

Good Code is Green

Green Coding Solutions is driving the transition to sustainable software. We provide open-source tools and expertise to help companies, NGOs, and developers measure, understand, and reduce their digital carbon footprint.

P.S. We are also the people that started this conference ;)

The Problem

Resource consumption is rising

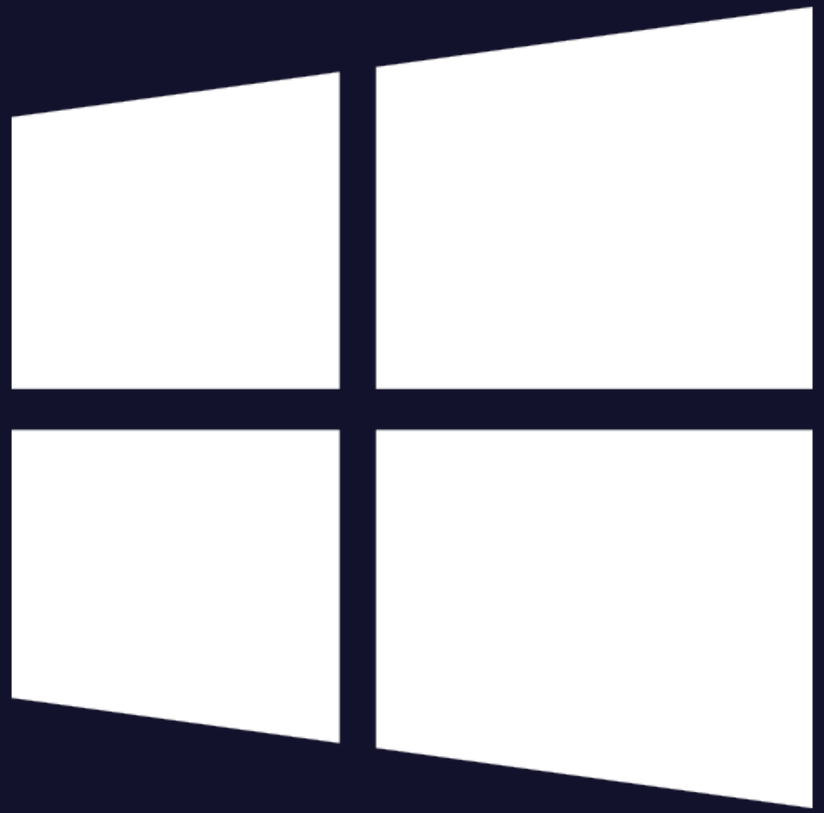
Things are becoming more and more complex

We can’t see the sources

Window

- Windows Energy Estimation Engine (E3)
- Gives you uJ for processes
- Close source

  -> No idea how the values are calculated
- Not well documented
- Only on devices with battery
- No WSL support



macOS

- `powermetrics`
- Gives you energy impact for processes
- Close source

  -> No idea how the values are calculated
- Has been party reverse engineered
- Not well documented
- Sometimes give you totally wrong values
- We use it for some programs

https://blog.mozilla.org/nnethercote/2015/08/26/what-does-the-os-x-activity-monitors-energy-impact-actually-measure/

Linux

- Linux, despite dominating server (77.4%) and mobile (70.8%) markets, lacks effective tools for process-level power measurement.
- We can get measurement for the whole core/ chip with RAPL. But security a problem here.
- We have models that use cpu utilization but that is not a very good value

Making Linux best in class

1

Provide real-time, process-level energy monitoring.

2

Low overhead and applicable on a wide variety of machines

3

Develop an open-source extensible framework

Making Linux best in class

L3E

Linux Energy Estimation Engine:
Flexible, customizable energy models
with fine-grained metrics.

PowerLetrics

Open-Source Implementation: Adapts
powermetrics methodology for Linux using
eBPF.

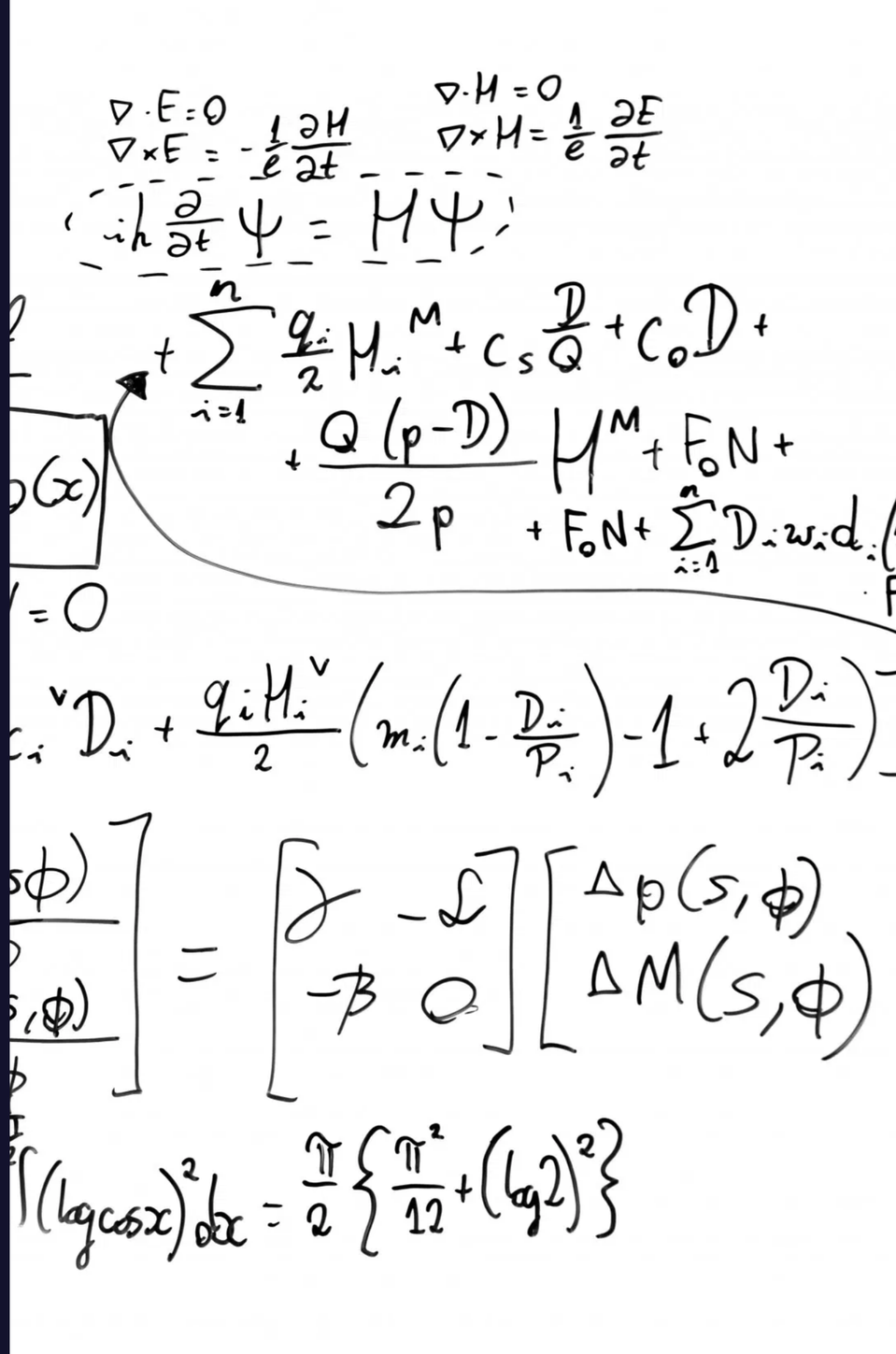
ProcPower

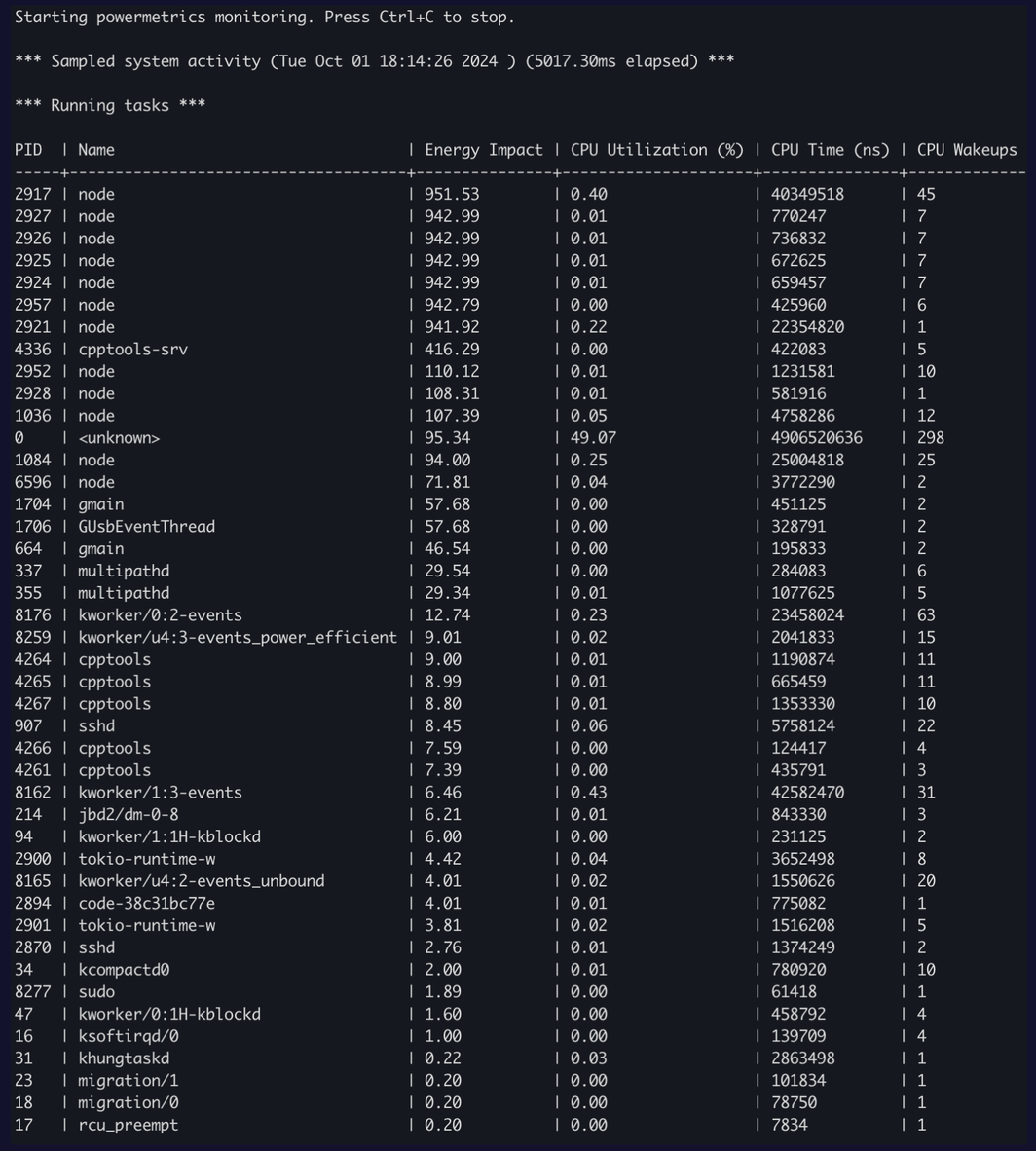
Secure Approach: Innovative /proc integration
eliminates the need for elevated privileges.

L3E

A model that can predict the energy consumption of a process based on hardware counters

- Works in containers (less counters -> less accurate)
- For better accuracy needs calibration
- Graphic cards difficult (not exposed to the kernel)
- Non linear





```
Starting powermetrics monitoring. Press Ctrl+C to stop.

*** Sampled system activity (Tue Oct 01 18:14:26 2024 ) (5017.30ms elapsed) ***

*** Running tasks ***

PID   | Name                                | Energy Impact | CPU Utilization (%) | CPU Time (ns) | CPU Wakeups
------+-------------------------------------+---------------+---------------------+---------------+------------
2917  | node                                | 951.53        | 0.40                | 40349518      | 45
2927  | node                                | 942.99        | 0.01                | 770247        | 7
2926  | node                                | 942.99        | 0.01                | 736832        | 7
2925  | node                                | 942.99        | 0.01                | 672625        | 7
2924  | node                                | 942.99        | 0.01                | 659457        | 7
2957  | node                                | 942.79        | 0.00                | 425960        | 6
2921  | node                                | 941.92        | 0.22                | 22354820      | 1
4336  | cpptools-srv                        | 416.29        | 0.00                | 422083        | 5
2952  | node                                | 110.12        | 0.01                | 1231581       | 10
2928  | node                                | 108.31        | 0.01                | 581916        | 1
1036  | node                                | 107.39        | 0.05                | 4758286       | 12
0     | <unknown>                           | 95.34         | 49.07               | 4906520636    | 298
1084  | node                                | 94.00         | 0.25                | 25004818      | 25
6596  | node                                | 71.81         | 0.04                | 3772290       | 2
1704  | gmain                               | 57.68         | 0.00                | 451125        | 2
1706  | GUsbEventThread                     | 57.68         | 0.00                | 328791        | 2
664   | gmain                               | 46.54         | 0.00                | 195833        | 2
337   | multipathd                          | 29.54         | 0.00                | 284083        | 6
355   | multipathd                          | 29.34         | 0.01                | 1077625       | 5
8176  | kworker/0:2-events                  | 12.74         | 0.23                | 23458024      | 63
8259  | kworker/u4:3-events_power_efficient | 9.01          | 0.02                | 2041833       | 15
4264  | cpptools                            | 9.00          | 0.01                | 1190874       | 11
4265  | cpptools                            | 8.99          | 0.01                | 665459        | 11
4267  | cpptools                            | 8.80          | 0.01                | 1353330       | 10
907   | sshd                                | 8.45          | 0.06                | 5758124       | 22
4266  | cpptools                            | 7.59          | 0.00                | 124417        | 4
4261  | cpptools                            | 7.39          | 0.00                | 435791        | 3
8162  | kworker/1:3-events                  | 6.46          | 0.43                | 42582470      | 31
214   | jbd2/dm-0-8                         | 6.21          | 0.01                | 843330        | 3
94    | kworker/1:1H-kblockd                | 6.00          | 0.00                | 231125        | 2
2900  | tokio-runtime-w                     | 4.42          | 0.04                | 3652498       | 8
8165  | kworker/u4:2-events_unbound         | 4.01          | 0.02                | 1550626       | 20
2894  | code-38c31bc77e                     | 4.01          | 0.01                | 775082        | 1
2901  | tokio-runtime-w                     | 3.81          | 0.02                | 1516208       | 5
2870  | sshd                                | 2.76          | 0.01                | 1374249       | 2
34    | kcompactd0                          | 2.00          | 0.01                | 780920        | 10
8277  | sudo                                | 1.89          | 0.00                | 61418         | 1
47    | kworker/0:1H-kblockd                | 1.60          | 0.00                | 458792        | 4
16    | ksoftirqd/0                         | 1.00          | 0.00                | 139709        | 4
31    | khungtaskd                          | 0.22          | 0.03                | 2863498       | 1
23    | migration/1                         | 0.20          | 0.00                | 101834        | 1
18    | migration/0                         | 0.20          | 0.00                | 78750         | 1
17    | rcu_preempt                         | 0.20          | 0.00                | 7834          | 1
```

powerletrics

- Like powermetrics
- Uses eBPF and RAPL to get values from the OS
- Separates them on a per process level
- Can be used for benchmarking, testing
- => When you are looking/ coding
- Currently in python
- Overhead 2%
- Funded by Catalyst Fund

Green Screen Coalition

$ pip install powerletrics

ProcPower

Kernel Extension

Collects all the metrics in the kernel with very low overhead

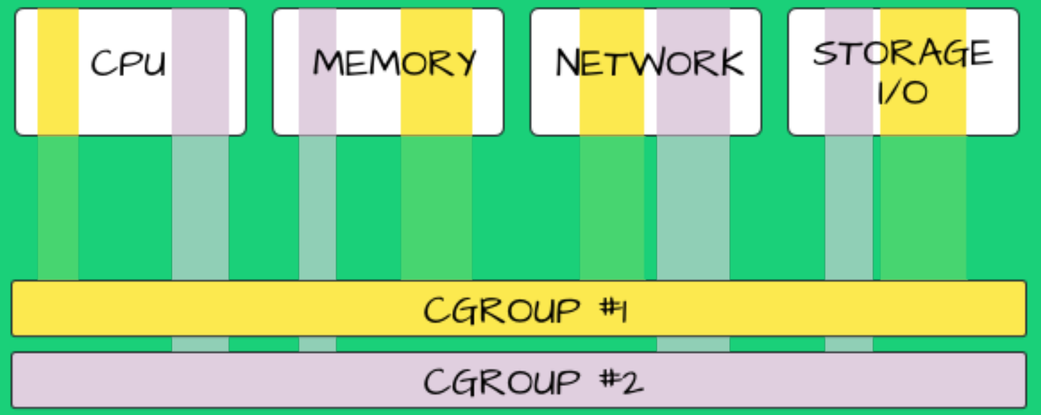
Exposes through /proc

And can be read in cgroups

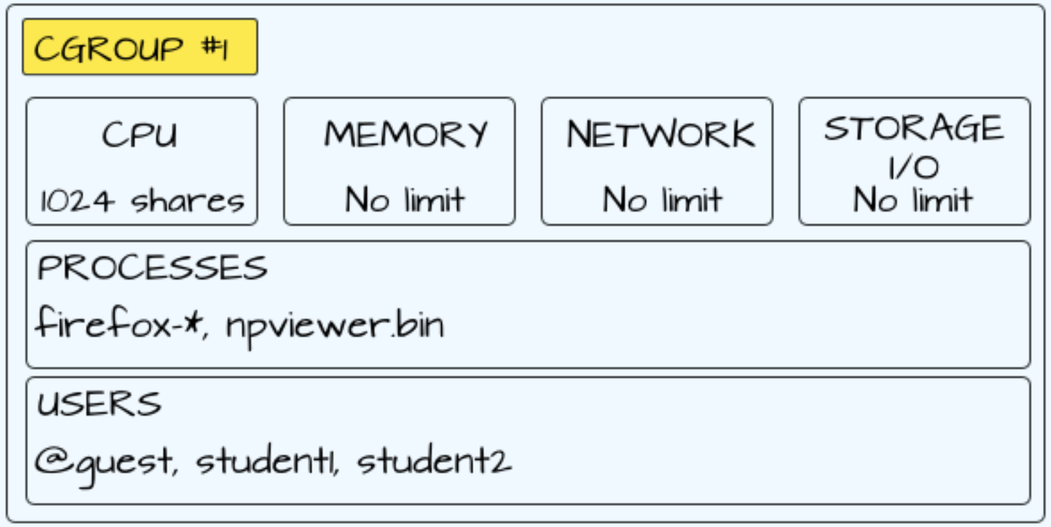
Secure

Windowing approach assures that no sensitive data can be leaked

cgroups



https://mairin.wordpress.com/2011/05/13/ideas-for-a-cgroups-ui/



Data

/proc/energy/cgroup

Shows the energy consumption of the processes in the current cgroup

/proc/energy/all

Shows the energy consumption of all the processes on the system. Requires root

/sys/kernel/debug/energy/all

Gives you everything for model training or debugging

Cool things

As easy as accessing a file

Your standard logging can collect energy metrics

Hosting

Your programs in your hosting environment can access their energy values

Energy becomes A*

Suddenly energy considerations become a first class citizen without much overhead

```
didi@fedora:~$ cat /proc/energy/cgroup
pid=5483 energy=2102001536990 alive=1 kernel=0 cpu_ns=420298267237 mem=6352896 instructions=102040161 wakeups=178863 diski=2293694464 disko=0 rx=0 tx=0 window_time=904474254342 comm=bash
didi@fedora:~$ sudo cat /proc/energy/all | head -n 10
timestamp=908562889618
iterations=7818
sample_ns=100000000
window_ns=1000000000
rapl_core_sum_uj=962972273
rapl_psys_sum_uj=4545448953
pid=0 energy=2416574943495 alive=1 kernel=1 cpu_ns=202584057566 mem=4823293952 instructions=280730931133 wakeups=713029 diski=1107748916 disko=1568890880 rx=27495 tx=2858 comm=*system*
pid=3586 energy=148310882419435 alive=1 kernel=0 cpu_ns=29661922982946 mem=176640000 instructions=253500941 wakeups=32516350 diski=246573465600 disko=0 rx=0 tx=0 comm=gunicorn
pid=3322 energy=63421765950 alive=1 kernel=0 cpu_ns=12684353190 mem=4714496 instructions=0 wakeups=0 diski=0 disko=0 rx=0 tx=0 comm=nginx
pid=50 energy=515245300 alive=0 kernel=1 cpu_ns=103047771 mem=0 instructions=1289 wakeups=0 diski=0 disko=0 rx=0 tx=0 comm=kworker/10:0
didi@fedora:~$
```

```
didi@fedora:~$ cat /proc/energy/cgroup
pid=5483 energy=2102001536990 alive=1 kernel=0 cpu_ns=420298267237 mem=6352896 ins
didi@fedora:~$ sudo cat /proc/energy/all | head -n 10
timestamp=908562889618
iterations=7818
sample_ns=100000000
window_ns=1000000000
rapl_core_sum_uj=962972273
rapl_psys_sum_uj=4545448953
pid=0 energy=2416574943495 alive=1 kernel=1 cpu_ns=202584057566 mem=4823293952 ins
pid=3586 energy=148310882419435 alive=1 kernel=0 cpu_ns=29661922982946 mem=1766400
pid=3322 energy=63421765950 alive=1 kernel=0 cpu_ns=12684353190 mem=4714496 instru
pid=50 energy=515245300 alive=0 kernel=1 cpu_ns=103047771 mem=0 instructions=1289
didi@fedora:~$
```

```
didi@fedora:~$ cat /proc/energy/cgroup
pid=5483 energy=2102001536990 alive=1 kernel=0 cpu_ns=420298267237 mem=6352896 ins
didi@fedora:~$ sudo cat /proc/energy/all | head -n 10
timestamp=908562889618
iterations=7818
sample_ns=100000000
window_ns=1000000000
rapl_core_sum_uj=962972273
rapl_psys_sum_uj=4545448953
pid=0 energy=2416574943495 alive=1 kernel=1 cpu_ns=202584057566 mem=4823293952 ins
pid=3586 energy=1483108 live=1 kernel=0 cpu_ns=29661922982946 mem=1766400
pid=3322 energy=6342176 950 alive=1 kernel=0 cpu_ns=12684353190 mem=4714496 instru
pid=50 energy=5152453 alive=0 ernel=1 cpu_ns=103047771 mem=0 instructions=1289
didi@fedora:~$
```

Next steps

More integrations

Try to get it into some programs

Model DB

Build a database of machines with model data
so that calibration is not needed all the time

More metrics

What is X doing?
What is the graphics card doing?
How about unknown hardware?

ProcPower Sustainability Monitor

Status: OK

File: /proc/energy/cgroup

Latest readings (per PID)

Timestamp: 2025-09-29 13:44:49

	PID	comm	energy
1	426202	code	9084.723
2	426213	code	902.594
3	428485	bash	9.482
4	428860	code	8539.875
5	426100	code	5.716
6	426101	code	5.402
7	428327	code	4234.883

Raw file

```
pid=4899 energy=795105664254053
pid=4966 energy=123675703885430
pid=426097 energy=1355968131648
pid=426100 energy=2057762991853
pid=426101 energy=1944614784766
pid=426103 energy=1945665499995
pid=426118 energy=2576876869200
pid=426131 energy=1102231767053
pid=426139 energy=1402050747098
pid=426187 energy=9995270885062
pid=426202 energy=3270500314865
pid=426208 energy=4816611608597
pid=426213 energy=3249337276198
```

Polling interval: 1000 ms Max points: 300

localhost:8080/wp-admin/admin.php?page=wp-sustainability-monitor

test 1 0 New Howdy, a

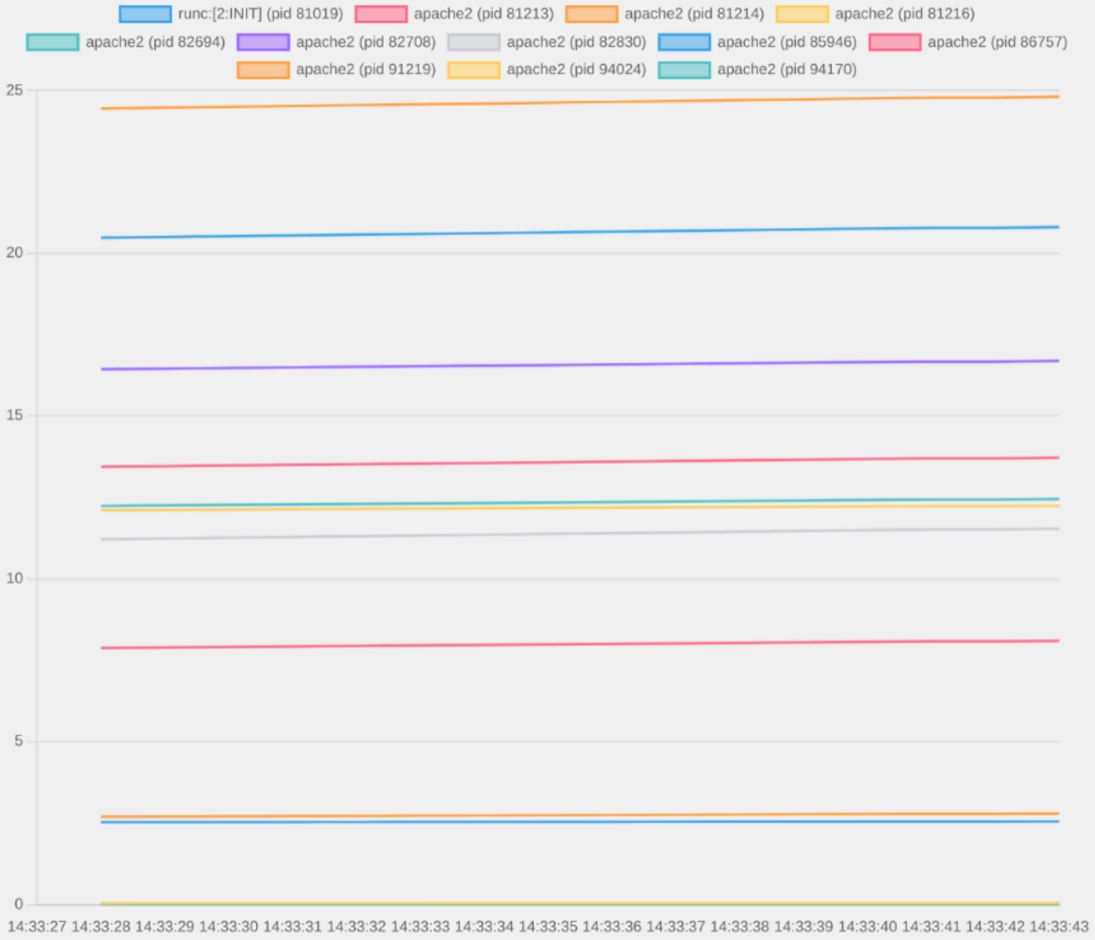
- Dashboard
- Posts
- Media
- Pages
- Comments
- Appearance
- Plugins 1
- Sustainability
- Users
- Tools
- Settings
- Collapse Menu

Sustainability

Your server uses a lot of resources to run WordPress. Monitoring this will give you insights into how to optimize your installation.

You need to install the ProcPower kernel extension on the host system. This page then reads `/proc/energy/cgroup` once per second and graphs per-PID `energy` values.

Status: OK



Latest readings (per PID)

Timestamp: 18/09/2025, 14:33:43

PID	comm	energy
81214	apache2	24.829
85946	apache2	20.824
82708	apache2	16.710
86757	apache2	13.740
82694	apache2	12.464
81216	apache2	12.254
82830	apache2	11.564
81213	apache2	8.109
91219	apache2	2.797

▶ Raw file

Sustainability

Not secure 0.0.0.0:8000/admin/sustainability/monitor/

Django administration

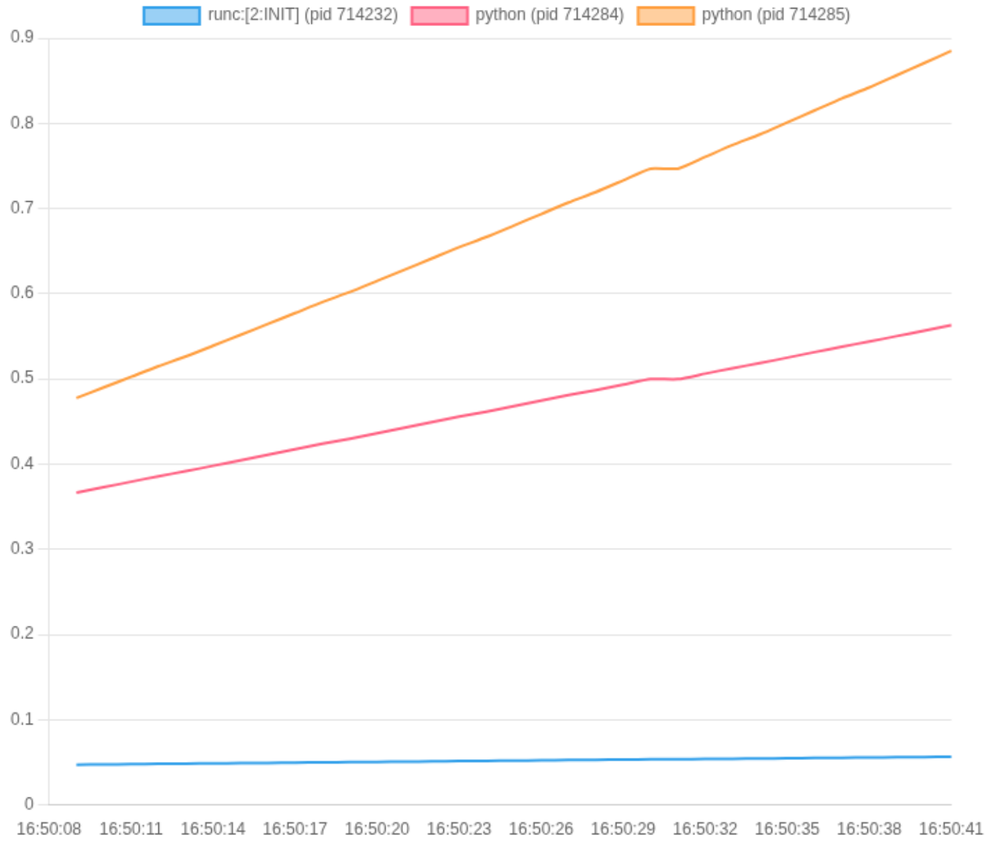
Home

Sustainability

Your server uses resources to run Django & your site. Monitoring helps optimize your installation.

You need the ProcPower kernel extension on the host system. This page reads `/proc/energy/cgroup` once per second and graphs per-PID `energy` values.

Status: OK



Latest readings (per PID)

Timestamp: 29/09/2025, 16:50:41

PID	COMM	ENERGY
714285	python	0.900
714284	python	0.570
714232	runc:[2:INIT]	0.057

▼ Raw file

```
pid=714232 energy=204649818420 alive=1 kernel=0 cpu_ns=16858900278 mem=
pid=714284 energy=2050654073810 alive=1 kernel=0 cpu_ns=396980355866 me
pid=714285 energy=3238467357255 alive=1 kernel=0 cpu_ns=638356645378 me
```

Thank you

Green Screen
Catalyst Fund

Prototype Fund

Green Coding Solutions

QUESTIONS?

Email:

- didi@green-coding.io

Urls:

- https://github.com/green-kernel
- https://www.green-coding.io/