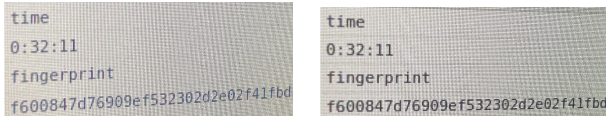


Poster: CPU-Print: From Multiplying Matrices to Uniquely Identifying Devices using DVFS

Kaustav Goswami, Hari Venugopalan, Ryan Swift[†], Chen-Nee Chuah, Jason Lowe-Power, Zubair Shafiq
University of California, Davis [†]University of Southern California
{kggoswami,hvenugopalan,chuah,jlowepower,zshafiq}@ucdavis.edu, ryanswif@usc.edu

Privacy concerns have led to increasing restrictions on stateful identifiers [3], [11], [18]. Consequently, the research community is paying significant attention towards developing novel *stateless* device fingerprinting techniques [5], [15], [20], [21], [23]. Fingerprinting, which uniquely identifies devices, users, or browsers, is applied in fraud detection, bot detection [24], user tracking [9], [22] *etc.*.

An ideal fingerprinting technique must be unique, stable and robust [8]. This means that the technique needs to capture sufficient differences in terms of software and/or hardware to distinguish homogeneous devices, with and without background noise. In addition, the fingerprint should remain stable over a sufficiently long time period despite background noise.



```
time
0:32:11
fingerprint
f600847d76909ef532302d2e02f41fbd
```

(a) Fingerprint.js unique identifier string on system A. (b) Fingerprint.js unique identifier string on system B.

Fig. 1. Demonstration of generating the same fingerprint on two identically configured systems.

Fingerprint.js [9] is a widely used software-based fingerprinting technique, that aggregates both software and hardware parameters to uniquely identify devices. However, it may generate identical identifiers in homogeneous settings, as shown in Figure 1.

To overcome the issue mentioned above, researchers have proposed various hardware-based device fingerprinting techniques [5], [15], [20], [21], [23] based on side-channels. These techniques leverage stable hardware properties until the hardware is changed. Sanchez-Rola *et al.* exploited a high-resolution timing-based side channel to fingerprint CPUs [20]. Laor *et al.* introduced DRAWNAPART [21] to fingerprint browsers using the underlying GPU via the WebGL library, with timing side-channel signals as fingerprints. Standalone power side-channels on the other hand, have been exploited to leak cryptographic keys [16], [25], [26]. Researchers have combined this channel for fingerprinting, notably website [19] and app [4] fingerprinting. DF-SCA [6] proposed the idea of using DVFS for website fingerprinting.

Building on previous works, we hypothesize the existence of a fingerprinting signal based on the CPU’s power draw, specifically the frequency state of the CPU core. Toward this, we create *power-viruses* to exploit this signal from the user-

space. We define a power-virus in our work as a program that can draw various amount of power. Changes in CPU cores’ power draw alter the frequency state. Variations in CPU fabrication [14] manifest as timing differences when controlled by the fingerprinter. Therefore we present CPU-Print that correlates DVFS-based power side-channels with device fingerprinting. Even though we hype “*power-viruses*”, however, in reality, it is a simple matrix multiplication program. We perform some ALU operation (load) on the CPU cores and then force the CPU cores to sleep. We then change the load per unit interval to follows a certain periodic function $f(x + P) = f(x); P > 0$, to maximize the signal. We hypothesize that the power draw of our power-virus needs to be instantaneous but long enough to trigger the frequency scaling mechanism.

Operating systems (OS) today use power profiles like power governors in Linux to manage the utilization-to-power ratio of the CPU cores [2], [7], [17]. From our testing, we found that the default power governor in Linux and Android is `ondemand` from the pool of `performance`, `ondemand` and `powersave`. The `ondemand` governor samples CPU load and switches between frequency states, while `performance` maintains maximum frequency, scaling down during temperature throttling. Conversely, `powersave` keeps the frequency at the minimum possible frequency state. Changing the power governor in Linux requires root privileges. However, an unprivileged user can read the frequency state file (`scaling_cur_freq`).

There are three primary challenges for us to address:

- Although Linux covers the majority of devices, CPU-Print must be OS-agnostic.
- A JavaScript-based web implementation is needed for broader reach, but JavaScript’s sandbox restricts access to system files like `scaling_cur_freq`.
- Overcoming low-resolution clocks in browsers, which were patched because of privacy concerns [1], [21].

We address all the challenges comprehensively. Measuring the relationship between the DVFS governor and power and frequency data is not trivial. We design our *custom-clock* for CPU-Print to measure elapsed time of a fixed computation, as this is directly related to the frequency and power usage of the CPU, and is measurable within a browser at millisecond resolution [10], [13]. We use a constant function in a loop to approximate a fixed interval of time. We run the loop 1000

TABLE I
RESULTS OBSERVED ACROSS DIFFERENT CLASSIFIERS

Base Model	# of model params (incl. classifier)	Accuracy	F1 Score	Precision	Recall
DTW	N/A	0.6914	0.6918	0.6909	0.6926
MLP	2,181,505	0.8800	0.9474	0.9000	1.0000
FCN	33,090,433	0.8500	0.9455	0.8966	1.0000
ResNet [12]	8,255,297	0.5250	0.6885	0.5385	0.9545

times to get around the low resolution timer problem. The clock runs concurrently with the power virus, collecting timing values into an array, as the power-virus induces state switches in DVFS. The power-virus forces scaling to higher frequency as load per unit increases. The CPU cores draw more power at higher frequency which leads to increased temperature and faster computation. When the load per unit starts decreasing, CPU cores starts drawing less power and the governor scales to a lower frequency. There will be throttling due to the CPU core's thermal limitations. This forces the governor to scale. All the scaling values adds to the fingerprint signal.

We position ourselves as the fingerprinter whose goal is to extract unique and stable fingerprints from both homogeneous and heterogeneous devices. In our threat model, the fingerprinter can run the power-virus on the user's device via the web, producing a series of timing values per sampling interval up to 1 minute.

In this abstract, we show results from our in-the-wild deployment. We collected 50,000 traces of multiple power-viruses: step, sine curve, saw-tooth, inverse-saw-tooth and delta with the same time period for each of the periodic function. There were 225 unique devices with two or more traces, which we used for fingerprint training and later for testing. We collected a cookie as the ground truth. Table I shows different statistics when using different classifiers. A random classifier is our baseline as it will always have a maximum accuracy of 0.500. We use DTW, 1D CNN (MLP), FCN and ResNet to measure the similarity between the two traces. While DTW is sufficient for *device-type* fingerprinting, we use a ML-based classifier for device fingerprinting. The accuracy of 1D CNN and FCN is fairly high, up to 88% for the former, compared to a simple distance-based technique like DTW and a more complex technique like ResNet.

Concretely, our contributions in this abstract are:

- A DVFS-based device fingerprinting technique.
- We collected in-the-wild data via a large-scale deployment of our technique and collected 50,000 traces across 225 unique devices and saw up to 88% fingerprinting accuracy.

As for future work, our plan is to perform extensive in-lab experiments for testing the robustness of CPU-Print.

REFERENCES

- [1] [Online]. Available: https://developer.mozilla.org/en-US/docs/Web/API/Performance_API/High_precision_timing
- [2] Apple. [Online]. Available: <https://support.apple.com/en-us/101613>
- [3] —, "User Privacy and Data Use," <https://developer.apple.com/app-store/user-privacy-and-data-use/>.
- [4] Y. Chen, X. Jin, J. Sun, R. Zhang, and Y. Zhang, "Powerful: Mobile app fingerprinting via power analysis," in *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*, 2017, pp. 1–9.
- [5] Y. Cheng, X. Ji, J. Zhang, W. Xu, and Y.-C. Chen, "Demicpu: Device fingerprinting with magnetic signals radiated by cpu," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '19. ACM, 2019, p. 1149–1170.
- [6] D. R. Dipta and B. Gulmezoglu, "Df-sca: Dynamic frequency side channel attacks are practical," in *Proceedings of the 38th Annual Computer Security Applications Conference*, ser. ACSAC '22. ACM, 2022, p. 841–853.
- [7] Dominik Brodowski, Nico Golde, Rafael J. Wysocki and Viresh Kumar, "Linux CPUFreq CPUFreq Governor," <https://www.kernel.org/doc/Documentation/cpu-freq/governors.txt>.
- [8] P. Eckersley, "How unique is your web browser?" in *Proceedings of the 10th International Conference on Privacy Enhancing Technologies*, ser. PETS'10. Springer-Verlag, 2010, p. 1–18.
- [9] FingerprintJS, "FingerprintJS," <https://github.com/fingerprintjs/fingerprintjs>.
- [10] M. Firefox, "Performance: now() method," Tech. Rep., 2024. [Online]. Available: <https://developer.mozilla.org/en-US/docs/Web/API/Performance/now>
- [11] Google, "Advertising ID," <https://support.google.com/googleplay/android-developer/answer/6048248>.
- [12] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.
- [13] P. Irish, "When milliseconds are not enough - performance.now," Google Chrome, Tech. Rep., 2012. [Online]. Available: <https://developer.chrome.com/blog/when-milliseconds-are-not-enough-performance-now/>
- [14] J. Lee, D. Lim, B. Gassend, G. Suh, M. van Dijk, and S. Devadas, "A technique to build a secret key in integrated circuits for identification and authentication applications," in *2004 Symposium on VLSI Circuits. Digest of Technical Papers (IEEE Cat. No.04CH37525)*, 2004, pp. 176–179.
- [15] D. Li, D. Liu, Y. Ren, Z. Wang, Y. Sun, Z. Guan, Q. Wu, and J. Liu, "Fphammer: A device identification framework based on dram fingerprinting," 2023.
- [16] M. Lipp, A. Kogler, D. Oswald, M. Schwarz, C. Easdon, C. Canella, and D. Gruss, "Platypus: Software-based power side-channel attacks on x86," in *2021 IEEE Symposium on Security and Privacy (SP)*, 2021, pp. 355–371.
- [17] Microsoft, "Processor power management options overview," Tech. Rep., 2024. [Online]. Available: <https://learn.microsoft.com/en-us/windows-hardware/customize/power-settings/configure-processor-power-management-options>
- [18] Mozilla, "Using HTTP cookies," <https://developer.mozilla.org/en-US/docs/Web/HTTP/Cookies>.
- [19] Y. Qin and C. Yue, "Website fingerprinting by power estimation based side-channel attacks on android 7," in *2018 17th IEEE International Conference On Trust, Security And Privacy In Computing And Communications/ 12th IEEE International Conference On Big Data Science And Engineering (TrustCom/BigDataSE)*, 2018, pp. 1030–1039.
- [20] I. Sanchez-Rola, I. Santos, and D. Balzarotti, "Clock around the clock: Time-based device fingerprinting," in *Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security*, ser. CCS '18. ACM, 2018, p. 1502–1514.
- [21] Tomer Laor and Naif Mehanna and Antonin Durey and Vitaly Dyadyuk and Pierre Laperdrix and Cl  mentine Maurice and Yossi Oren and Romain Rouvoy and Walter Rudametkin and Yuval Yarom, "DRAWN APART : A Device Identification Technique based on Remote GPU Fingerprinting," in *Proceedings 2022 Network and Distributed System Security Symposium*. Internet Society, 2022.
- [22] A. Vastel, P. Laperdrix, W. Rudametkin, and R. Rouvoy, "Fp-stalker: Tracking browser fingerprint evolutions," in *2018 IEEE Symposium on Security and Privacy (SP)*, 2018, pp. 728–741.
- [23] H. Venugopalan, K. Goswami, Z. A. Din, J. Lowe-Power, S. T. King, and Z. Shafiq, "Fp-rowhammer: Dram-based device fingerprinting," 2024.
- [24] H. Venugopalan, S. Munir, S. Ahmed, T. Wang, S. T. King, and Z. Shafiq, "Fp-inconsistent: Detecting evasive bots using browser fingerprint inconsistencies," 2025.
- [25] Y. Wang, R. Paccagnella, E. T. He, H. Shacham, C. W. Fletcher, and D. Kohlbrenner, "Hertzbleed: Turning power Side-Channel attacks into remote timing attacks on x86," in *31st USENIX Security Symposium (USENIX Security 22)*, Aug. 2022, pp. 679–697.
- [26] L. Yan, Y. Guo, X. Chen, and H. Mei, "A study on power side channels on mobile devices," in *Proceedings of the 7th Asia-Pacific Symposium on Internetworking*, ser. Internetworking '15. ACM, 2015, p. 30–38.

CPUPrint: From Multiplying Matrices to Uniquely Identifying Devices using DVFS

Kaustav Goswami¹, Hari Venugopalan¹, Ryan Swift², Chen-Nee Chuah¹, Jason Lowe-Power¹, Zubair Shafiq¹
¹University of California, Davis ²University of Southern California



Problem Statement

- There is a growing restriction against stateful identifiers due to privacy concerns [1].
- We need to design a new hardware-based fingerprinting techniques with:
 - Stateless
 - Robust
 - Stable
 - Accessible from the user-space

Motivation

- Software-based fingerprinting techniques are fast and but less stable.
- However, they can have a high false-positive rate in homogeneous settings.

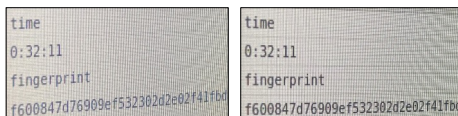


Figure 1: Identical fingerprints using Fingerprint.js across 2 homogeneous systems.

Objective

- Exploit CPU frequency readings as the fingerprint signal.

Challenges

- Platform-agnostic design.
- A JavaScript-based web implementation
- Overcoming low-resolution clocks in browsers.

Background

- Modern-day operating systems have power governors.
- Default power governor on Linux is **ondemand** or **schedutil**.
- The frequency of the CPU cores is scaled based on the utilization.
- Higher utilization draws more power.
- This changes the frequency state of the CPU via voltage scaling.

Power-Virus

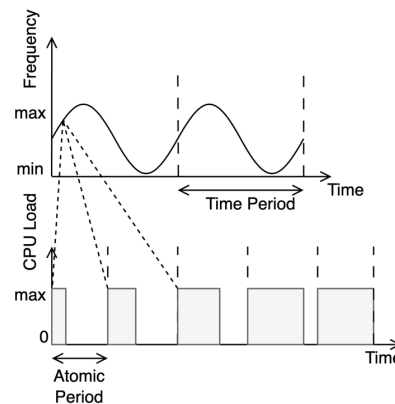


Figure 2: Theoretical demonstration of a power-virus.

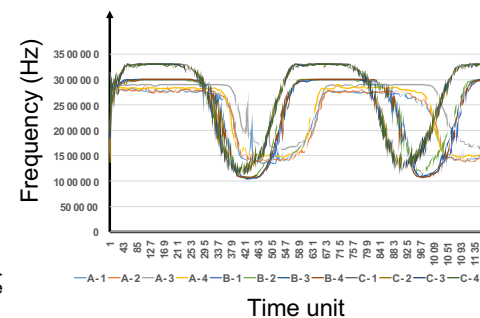


Figure 3: A plot with frequency values of 3 different systems when sine power-virus was being executed.

Results

Model	Params	Accuracy	F1 Score	Precision	Recall
DTW	N/A	0.6914	0.6918	0.6909	0.6926
MLP	2,181,505	0.8800	0.9474	0.9000	1.0000
FCN	33,090,433	0.8500	0.9455	0.8966	1.0000
ResNet	8,255,297	0.5250	0.6885	0.5385	0.9545

Table 1: Statistical results across different classifiers.

Related Works

- **Hardware-based fingerprinting:**
 - Clock-around-the-clock (CPU) [2]
 - DRAWNAPART (GPU) [3]
 - DF-SCA (DVFS) [4]
- **Power side-channels**
 - PLATYPUS [5]
- **Power side-channel + Fingerprint**
 - Qin *et al.* [6]: Website fingerprinting
 - POWERFUL [7]: App fingerprinting

Contributions

- A new device fingerprinting technique.
- Large-scale in the wild deployment with 50,000 traces across 225 unique devices.
- *Hierarchical fingerprinting.*

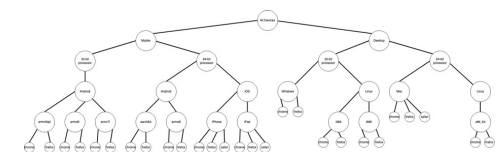


Figure 4: A demonstration of a hierarchical device type fingerprint tree.

References

- [1] Mozilla, "Using HTTP cookies"
- [2] I. Sanchez-Rola *et al.*, "Clock around the clock: Time-based device fingerprinting", ACM CCS 2018
- [3] Tomer Laor *et al.*, "DRAWNAPART: Device Identification Technique based on Remote GPU Fingerprinting", NDSS 2022
- [4] D. R. Dipta *et al.*, "Df-sca: Dynamic frequency side channel attacks are practical", ACM ACSAC 2022
- [5] M. Lipp *et al.*, "Platypus: Software-based power side-channel attacks on x86", IEEE S&P 2021
- [6] Y. Qin *et al.*, "Website fingerprinting by power estimation based side-channel attacks on android 7", IEEE TrustCom 2018
- [7] Y. Chen *et al.*, "Powerful: Mobile app fingerprinting via power analysis", IEEE INFOCOMM 2017