SoK: In Search of Lost Time: 
A Review of JavaScript Timers in Browsers

Thomas Rokicki\textsuperscript{1}  
Clémentine Maurice\textsuperscript{2}  
Pierre Laperdrix\textsuperscript{2}  
IEEE European Symposium on Security and Privacy - 09 /21

1: Univ Rennes, CNRS, IRISA.  
2: Univ Lille, CNRS, Inria.
JavaScript-based timing attacks

JavaScript Timing Attacks
JavaScript-based timing attacks

JavaScript Timing Attacks
JavaScript-based timing attacks

JavaScript Timing Attacks

Exploit timing differences to infer secrets from the JavaScript sandbox.
JavaScript-based timing attacks

JavaScript **Timing** Attacks

Exploit timing differences to infer secrets from the JavaScript sandbox.

Resolution of 10 -100 ns
JS and timers: A complicated history

Chrome 44
- Resolution: 5 µs

Chrome 64
- Resolution + Jitter: 100 µs

Chrome 72
- Resolution + Jitter: 5 µs

Firefox 41
- Resolution: 5 µs

Firefox 57.0.4
- Resolution: 20 µs

Firefox 59
- Resolution: 2 ms

Firefox 60
- Resolution + Jitter: 1 ms

Firefox 79
- Resolution: 20 µs

& COOP/COEP:

What are the motivations and implications of changing the timers' resolution?
JS and timers: A complicated history

What are the motivations and implications of changing the timers’ resolution?
Classification of JavaScript timing attacks

- Hardware-contention-based attacks
- Transient execution attacks
- Attacks based on system resources
- Attacks based on browser resources
Classification of JavaScript timing attacks

- **Hardware-contention-based attacks**
  
  **Principle:** The attacker infers secrets from timing differences caused by hardware state
  
  **Prerequisites:** High resolution timers & Shared hardware resources
  
  **Examples:** JavaScript Prime+Probe, Rowhammer.js

- Transient execution attacks
- Attacks based on system resources
- Attacks based on browser resources
Classification of JavaScript timing attacks

- Hardware-contention-based attacks
- **Transient execution attacks**
  - **Principle:** The attacker infers secrets from traces of transient execution on the hardware.
  - **Prerequisites:** Transient execution, high resolution timers & shared hardware resources
  - **Examples:** Spectre, RIDL
- Attacks based on system resources
- Attacks based on browser resources
Classification of JavaScript timing attacks

- Hardware-contention-based attacks
- Transient execution attacks

**Attacks based on system resources**

**Principle:** The attacker infers secrets from shared system resources.

**Prerequisites:** High resolution timers & shared system resources.

**Examples:** Keystroke attacks, memory deduplication attacks.

- Attacks based on browser resources
Classification of JavaScript timing attacks

- Hardware-contention-based attacks
- Transient execution attacks
- Attacks based on system resources
- Attacks based on browser resources

**Principle:** The attacker infers secrets from shared browser resources.

**Prerequisites:** High resolution timers & shared browser resources.

**Examples:** History sniffing, fingerprinting.
Built-in timers have a resolution ranging from 5-100 µs.

We have to create our own auxiliary timers:

- by interpolating the low resolution timers
- by exploiting multithreading to build a clock thread

Find out more about these timers and their properties in the paper!

Michael Schwarz et al. “Fantastic timers and where to find them: High-resolution microarchitectural attacks in javascript”. In: International Conference on Financial Cryptography and Data Security. 2017
Reducing the resolution alone is not sufficient because of interpolation.
Reducing the resolution alone is not sufficient because of interpolation.

Add jitter to the measurement.
Reducing the resolution alone is not sufficient because of interpolation.

Add **jitter** to the measurement.

Disable certain multithreading features.
Security vs Practicality

- High resolution timers useful for performance measurements, network, animation
- Multithreading is an important part of the evolution of JavaScript
Security vs Practicality

- High resolution timers useful for performance measurements, network, animation
- Multithreading is an important part of the evolution of JavaScript

Browser vendors want more efficient, less penalizing countermeasures.
Security vs Practicality

- High resolution timers useful for performance measurements, network, animation
- Multithreading is an important part of the evolution of JavaScript

Browser vendors want more efficient, less penalizing countermeasures.

**Isolation-based countermeasures**
Goals of isolation

Different processes means:

- Different address spaces
Different processes means:

- Different address spaces → Prevents Spectre v1 and other attacks that target the same address space
Goals of isolation

Different processes means:

- Different address spaces → Prevents Spectre v1 and other attacks that target the same address space

What site isolation does not prevent:
Goals of isolation

Different processes means:

- Different address spaces → Prevents Spectre v1 and other attacks that target the same address space

What site isolation does not prevent:

- Hardware contention timing attacks.
Goals of isolation

Different processes means:

- Different address spaces $\rightarrow$ Prevents Spectre v1 and other attacks that target the same address space

What site isolation does not prevent:

- Hardware contention timing attacks.
- Cross address space (transient execution) attacks$^1$.

---

$^1$For instance https://leaky.page/ was published a few days after our paper
A change in defense paradigm

With the introduction of these new isolation measures, browser vendors considered the main security issue fixed.
A change in defense paradigm

With the introduction of these new isolation measures, browser vendors considered the main security issue fixed.

Timing-based countermeasures are obsolete:

- Grant higher resolution and lower jitter to built-in timers
- Reallow multi-threading tools
Impact of these changes

- Timing-based countermeasures are efficient against most timing attacks.
Impact of these changes

- Timing-based countermeasures are efficient against most timing attacks.
- New, isolation-based countermeasures are strong countermeasures, but focused on Spectre or software-based timing attacks.
Impact of these changes

- Timing-based countermeasures are efficient against most timing attacks.
- New, isolation-based countermeasures are strong countermeasures, but focused on Spectre or software-based timing attacks.
- Hardware-based timing attacks as well as other transient execution attacks are only mitigated by timing-based countermeasures.
Impact of these changes

- Timing-based countermeasures are efficient against most timing attacks.
- New, isolation-based countermeasures are strong countermeasures, but focused on Spectre or software-based timing attacks.
- Hardware-based timing attacks as well as other transient execution attacks are only mitigated by timing-based countermeasures.
- Recent changes in timers have not been motivated or evaluated.
Impact of these changes

- Timing-based countermeasures are efficient against most timing attacks.
- New, isolation-based countermeasures are strong countermeasures, but focused on Spectre or software-based timing attacks.
- Hardware-based timing attacks as well as other transient execution attacks are only mitigated by timing-based countermeasures.
- Recent changes in timers have not been motivated or evaluated.

What are the security implications of reintroducing high resolution timers?
What are the security implications of reintroducing high resolution timers?

Automated framework to evaluate JavaScript timers using Selenium.

You can find more detailed technical explanations in the paper! The code is available here: https://github.com/thomasrokicki/in-search-of-lost-time
What are the security implications of reintroducing high resolution timers?

Automated framework to evaluate JavaScript timers using Selenium.
Works on Chrome and Firefox, including past and future versions.
What are the security implications of reintroducing high resolution timers?

Automated framework to evaluate JavaScript timers using Selenium.
Works on Chrome and Firefox, including past and future versions.
Our goal is that this analysis can be helpful not only at this point in time, but also in the future.
You can find more detailed technical explanations in the paper!
The code is available here: https://github.com/thomasrokicki/in-search-of-lost-time
Resolution: Smallest operation a timer can measure.
How to evaluate the efficiency of a timer

**Resolution**: Smallest operation a timer can measure.

**Measurement overhead**: Time it takes to make the measurement.

You can find more in-depth details of the experiments and results in the full paper.
On a attacker-controlled website, on Firefox 89 (2021) an attacker can:

• Create a cache covert channel with an ideal bandwidth 800,000 times superior compared to Firefox 78 (2018)

• Compute an eviction set in a matter of seconds, whereas it required tens of minutes on Firefox 78

Timers are more of a threat than two years ago.
On a attacker-controlled website, on Firefox 89 (2021) an attacker can:

- Create a cache covert channel with an ideal bandwidth 800,000 times superior compared to Firefox 78 (2018)
On a attacker-controlled website, on Firefox 89 (2021) an attacker can:

- Create a cache covert channel with an ideal bandwidth 800,000 times superior compared to Firefox 78 (2018)
- Compute an eviction set in a matter of seconds, whereas it required tens of minutes on Firefox 78
On a attacker-controlled website, on Firefox 89 (2021) an attacker can:

- Create a cache covert channel with an ideal bandwidth 800,000 times superior compared to Firefox 78 (2018)
- Compute an eviction set in a matter of seconds, whereas it required tens of minutes on Firefox 78

Timers are more of a threat than two years ago.
Conclusion

- Powerful and fast timers with a 10-100 ns resolution exist.

- Isolation-based countermeasures only apply to Spectre v1 and some system resource attacks.

- Browsers are potentially vulnerable to many hardware or transient execution attacks.

- More viable countermeasures must be found, but it is not particularly suited for browsers.
Conclusion

- Powerful and fast timers with a 10-100 ns resolution exist.
- Isolation-based countermeasures only apply to Spectre v1 and some system resource attacks.
Conclusion

- Powerful and fast timers with a 10-100 ns resolution exist.
- Isolation-based countermeasures only apply to Spectre v1 and some system resource attacks.
- Browsers are potentially vulnerable to many hardware or transient execution attacks.
Conclusion

- Powerful and fast timers with a 10-100 ns resolution exist.
- Isolation-based countermeasures only apply to Spectre v1 and some system resource attacks.
- Browsers are potentially vulnerable to many hardware or transient execution attacks.
- More viable countermeasures must be found, but it is not particularly suited for browsers.
Thank you for your attention

Contact me here: thomas.rokicki@irisa.fr

Feel free to read the paper for more technical details!

Find the code here:
https://github.com/thomasrokicki/in-search-of-lost-time