Race Conditions

Time(0.1) vs. Time(0.2)

Race Vulnerability

Typically having thread requests, This is usually how you expect the process to proceed ->

Thread 1	Thread 2		Integer value
			0
read value		←	0
increase value			0
write back		→	1
	read value	←	1
	increase value		1
	write back	→	2

Race Vulnerability

However if two or more threads were to run simultaneously, this is the output ->

Thread 1	Thread 2		Integer value
			0
read value		←	0
	read value	←	0
increase value			0
	increase value		0
write back		→	1
	write back	→	1

Banking Exploit and Race Condition Example

Without Vulnerability

Thread 1	Thread 2	Balance of Account A + B
Check Account A Balance ($500)		500
Add $500 to Account B		1000 (A - 500, B - 500)
Deduct $500 from Account A		500 (A - 0, B - 500)
	Check Account A Balance ($0)	500 (A - 0, B - 500)
	Transfer Failed (Low Balance)	500 (A - 0, B - 500)
		500 (A - 0, B - 500)
		500 (A - 0, B - 500)

With Vulnerability

Thread 1	Thread 2	Balance of Account A + B
Check Account A Balance ($500)		500
	Check Account A Balance ($500)	500
Add $500 to Account B		1000 (A - 500, B - 500)
	Add $500 to Account B	1500 (A - 500, B - 1000)
Deduct $500 from Account A		1000 (A - 0, B - 1000)
	Deduct $500 from Account A	1000 (A - 0, B - 1000)
		1000 (A - 0, B - 1000)

Normal Thread Processing

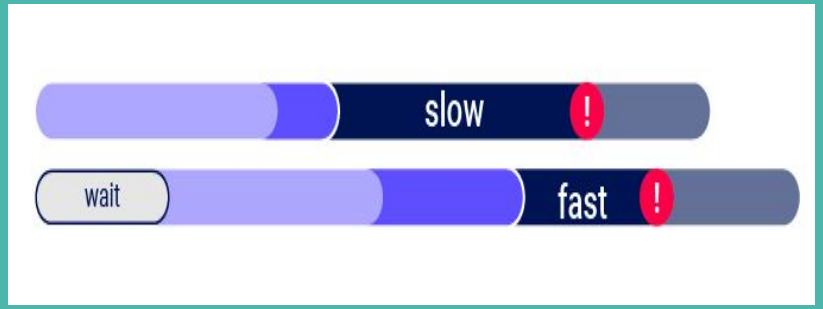
Sometimes 1ms is what keeping this RC vulnerability hidden

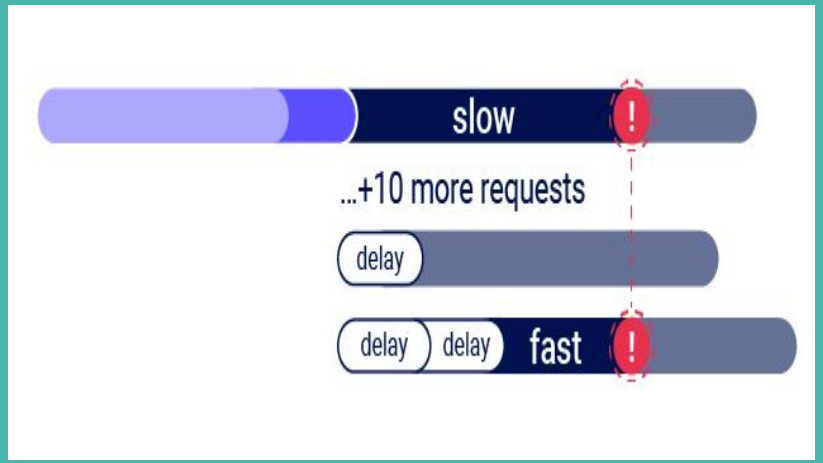
Thread.join() Processing

Activating or Sending all requests at the same time to ensure a Race Condition window.

Adding Delay or Wait

System needs to warm up or too many functions add a 1ms.





Blog Post Cited

Blog Banking Example - https://vickieli.dev/hacking/race-conditions/

Blog Code Example - https://book.hacktricks.xyz/pentesting-web/race-condition