Data-Dependent Instruction Timing Channels

Ranjit Jhala, Sorin Lerner, Hovav Shacham

An instruction exhibits *data-dependent timing* if its execution time varies with operand values

Floating point instructions on modern processors exhibit data-dependent timing

Data-dependent timing can enable side-channel attacks that reveal program secrets

	Divisor								
Dividend	0.0	1.0	1e10	1e+200	1e-300	1e-42	256	257	1e-320
	Cycle count								
0.0	6.56	6.59	6.58	6.55	6.57	6.58	6.57	6.57	6.59
1.0	6.58	6.58	12.19	12.17	12.22	12.24	6.57	12.24	165.76
1e10	6.58	6.55	12.25	12.20	12.23	12.25	6.57	12.22	165.81
1e+200	6.60	6.60	12.25	12.20	12.22	12.22	6.58	12.24	165.79
1e-300	6.59	6.57	175.22	12.24	12.17	12.22	6.52	12.23	165.83
1e-42	6.60	6.53	12.23	12.22	12.21	12.24	6.58	12.21	165.79
256	6.57	6.55	12.24	12.20	12.20	12.20	6.53	12.22	165.79
257	6.55	6.58	12.24	12.22	12.24	12.23	6.56	12.21	165.80
1e-320	6.56	150.73	165.79	6.59	165.78	165.76	150.66	165.80	165.78

Figure 7: Division timing for double precision floats on Intel i5-4460

What are the timing characteristics of floating-point instructions on modern processors?

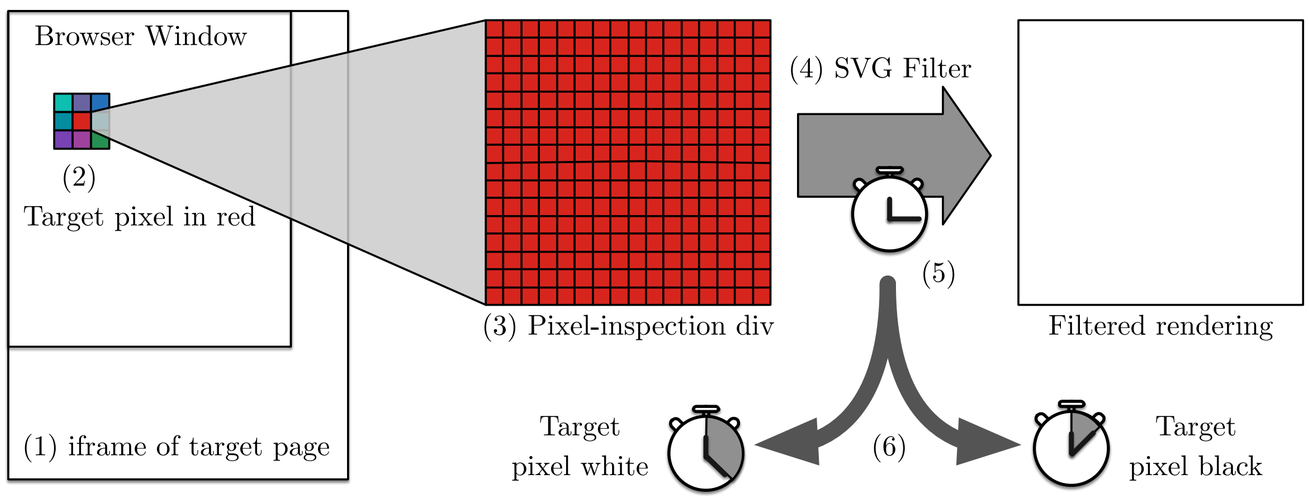
What software is affected by floating-point timing channels?

Can we perform floating- or fixed-point computations in a provably secure way?

Can we mitigate timing channels while using the floating-point unit?

How can processor vendors modify their processors to make floating point instructions safe to use?

We showed that data-dependent instruction timing can lead to side-channel attacks on real software, almost 20 years after the possibility was first hypothesized (Kocher 1996)



Showed that operations on *subnormal numbers* operands are slow [AKMJLS'15]

Characterized floating-point data-dependent instruction timing beyond subnormals [KS'17]

Adapted Paul Stone's SVG filter timing side channel to mount cross-origin pixel-stealing attacks against Firefox, Safari, Chrome [KS'17]

Built libfixedpointfixedtime, a provably constant-time fixed-point math library [AKMJLS'15]

Built CTFP, which transforms floating-point code to mitigate known timing channels; correctness and security verified with SMT solver [ANBJS'18]

Built Iodine, a tool for verifying constant-time execution of Verilog hardware designs [GKSJ'19]

Pixel-stealing vulnerabilities fixed in Firefox (CVE-2017-5407), Safari (CVE-2017-7006), and Chrome (CVE-2017-5107)

Published timing data, measurement code, and mitigation tools as open source

Co-PI Shacham presented data-dependent instruction timing channels to high school students attending UTCS's Hacking and Security Summer Camp (July 2019), and to developers at the ARM Security Summit (September 2019)

Browsers are used by billions of users to access private data

Safari disabled cross-origin SVG filters in response, reducing attack surface

ARM v8.4-A adds PSTATE.DIT flag for data-independent timing of some instructions — but not yet of floating point