CSE 484: Computer Security and Privacy

Side Channels and Web Tracking

Spring 2021

Tadayoshi Kohno

Thanks to Franzi Roesner, Dan Boneh, Dieter Gollmann, Dan Halperin, David Kohlbrenner, Yoshi Kohno, John Manferdelli, John Mitchell, Vitaly Shmatikov, Bennet Yee, and many others for sample slides and materials ...

Administrivia

- Lab 2 due May 25
- Final Project Checkpoint due May 26
- Lab 3 has become extra credit
- Friday (May 28): Guest Lecture: Charlie Reis (Google)

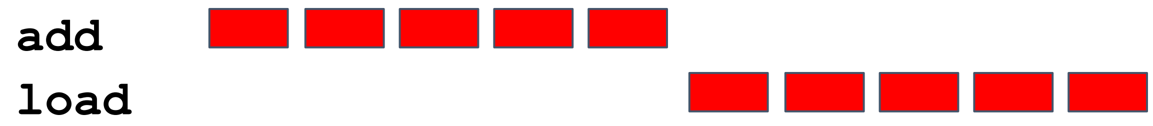
Spectre

- Exploit speculative execution and cache timing information to extract private information from the same process
  - Example: JavaScript from web page trying to extract information from Browser

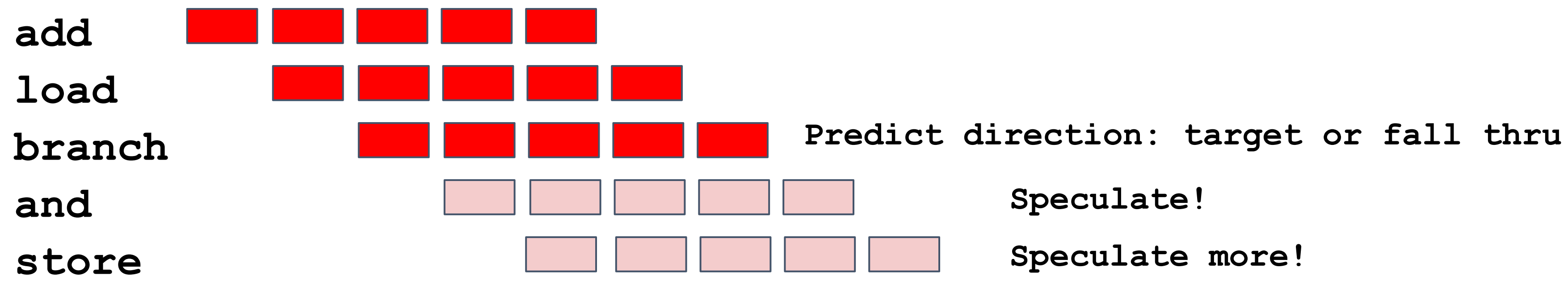
- Architecture Background:
  - Hardware architecture provides “promises” to software
  - Those proposes focus on the functional properties of the software, not performance properties
  - Architectures do a lot to try to increase performance

Instruction Speculation Tutorial

Many steps (cycles) to execute one instruction; time flows left to right →



Go Faster: Pipelining, branch prediction, & instruction speculation



Speculation correct: Commit architectural changes of **and** (register) & **store** (memory) go fast!

Mis-speculate: Abort architectural changes (registers, memory); go in other branch direction

Hardware Caching Tutorial

Main Memory (DRAM) 1000x too slow

Add Hardware Cache(s): small, transparent hardware memory

- Like a software cache: speculate near-term reuse (locality) is common
- Like a hash table: an item (block or line) can go in one or few slots

E.g., 4-entry cache w/ slot picked with address (key) modulo 4

0	--	**12**?	0	12	**07**?	0	12	**12**?	0	12	**16**?	0	16	Note **12**
1	--	Miss	1	--	Miss	1	--	HIT!	1	--	Miss	1	--	victimized
2	--	Insert **12**	2	--	Insert **07**	2	--	No	2	--	Victim **12**	2	--	“early” due
3	--		3	--		3	07	changes	3	07	Insert **16**	3	07	to “alias”

Spectre (Worksheet)

- Consider this code, running as a kernel system call or as part of a cryptographic library.

```
if (x < array1_size)
        y = array2[array1[x] * 256];
```

- Suppose:
  - That an adversary can run code, in the same process.
  - That an adversary can control the value x.
  - That an adversary has access to array2.
  - That the adversary's code cannot simply read arbitrary memory in the process.
  - That there is some secret value, elsewhere in the process, that the adversary would like to learn.

- Can you envision a way that an adversary could use their own code, to call a vulnerable function with the above code, to learn the secret information? Leverage branch prediction and cache structure / timing.

Spectre: Key Insights

- Train branch predictor to follow one branch of a conditional
- After branch predictor trained, make the followed branch access information that the code should *not* be allowed to access
- That access information will be loaded into the cache
- After the hardware determines that the branch was incorrectly executed, the logic of the program will be rolled back *but* the cache will still be impacted
- Time reads to cache, to see which cache lines are read more efficiently

Attacker Steps

- Attacker: Execute code with valid inputs, train branch predictor to assume conditional is true
- Attacker: Invoke code with x outside of array1 , array1_size and array2 not cached, but value at array1+x cached // Attacker goal: read secret memory at address array1+x
- CPU: CPU guesses bounds check is true, speculatively reads from array2[array1[x]*256] using malicious x
- CPU: Read from array2 loads data into cache at an address that depends on array1[x] using malicious x
- CPU: Change in cache state not reverted when processor realizes that speculative execution erroneous
- Attacker: Measure cache timings for array2; read of array2[n*256] will be fast for secret byte n (at array1+x)
- Attacker: Repeat for other values of x

Web Tracking

A topic in flux

- Tracking via cookies
- Tracking via other methods
- Fingerprinting
- FLoC

Ads That Follow You

Advertisers (and others) track your browsing behaviors for the purposes of targeted ads, website analytics, and personalized content.

Third-Party Web Tracking

Browsing profile for user 123:

cnn.com
theonion.com
private-site.com
political-site.com

These ads allow **criteo.com** to link your visits between sites, even if you never click on the ads.

chiefmartec.com April 2020

Marketing Technology Landscape

The Martech 5000

Total Solutions	8,000
Advertising & Promotion	922
Content & Experience	1,936
Social & Relationships	1,969
Commerce & Sales	1,314
Data	1,258
Management	601

Access all the data of this landscape & more at martech5000.com

2019
7,040 solutions

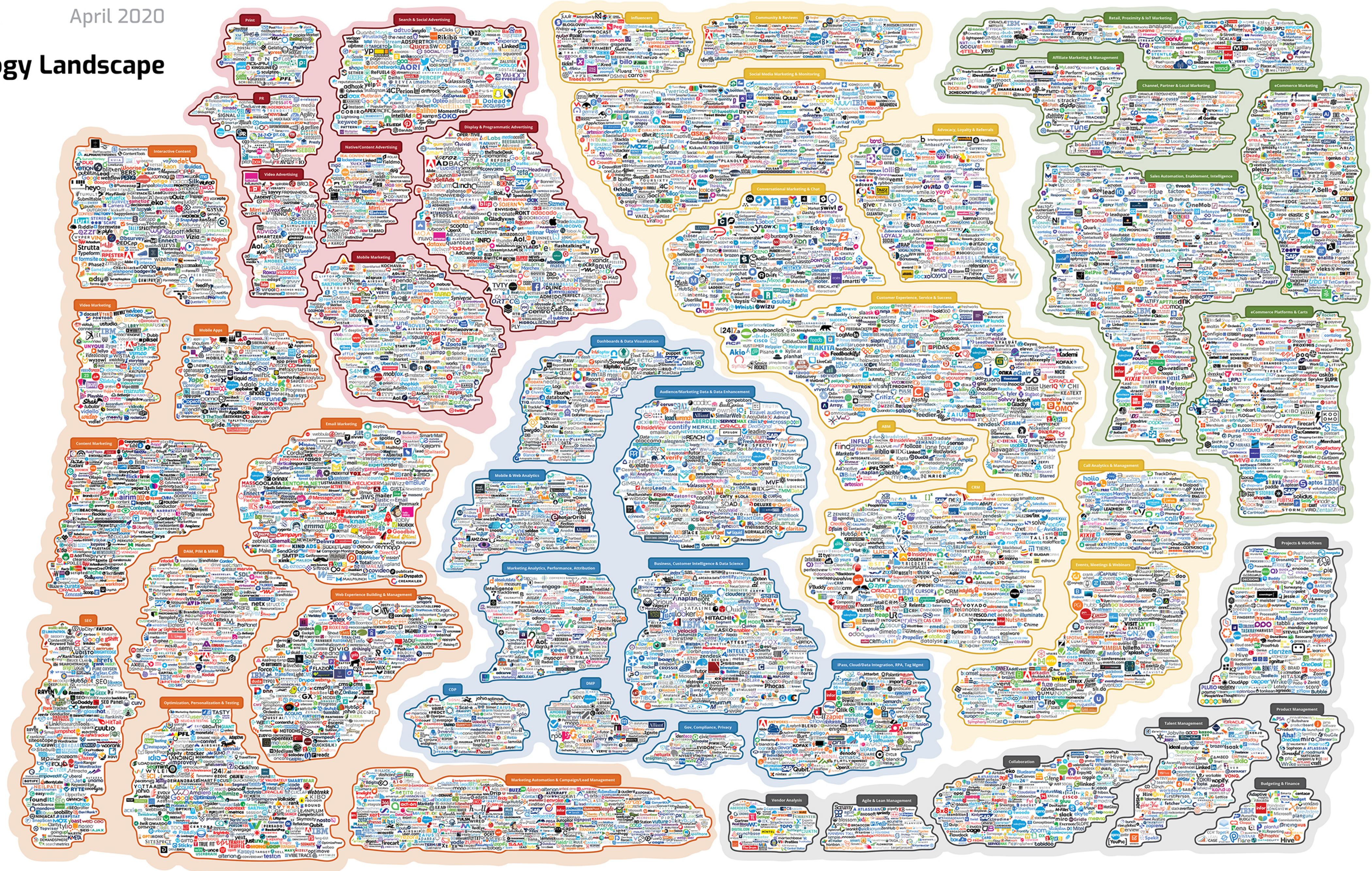
2018
6,829 solutions

2017
5,381 solutions

2016
3,874 solutions

2015
1,876 solutions

2014
947 solutions



Produced by Scott Brinker (@chiefmartec) and Blue Green Brands (@bluegreenbrands).

Concerns About Privacy

THE WALL STREET JOURNAL.

WHAT THEY KNOW | JULY 30, 2010

The Web's New Gold Mine: Your Secrets

A Journal investigation finds that one of the fastest-growing businesse[s] ... business of spying on consumers. First in a series.

Article | Video | Interactive Graphics | Comments

CNN Money | Business | Markets | Tech | Personal Finance | Small Business | Luxury | stock tickers
CNN | U.S. Edition | Log In

Your Privacy, For Sale

Big Data knows you're sick, tired and depressed

WHAT THEY KNOW

Websites Vary Prices, Deals Based on Users' Information

By JENNIFER VALENTINO-DEVRIES, JEREMY SINGER-VINE and ASHKAN SOLTANI

December 24, 2012

The New York Times

May 6, 2011, 5:01 pm | 3 Comments

'Do Not Track' Privacy Bill Appears in Congress

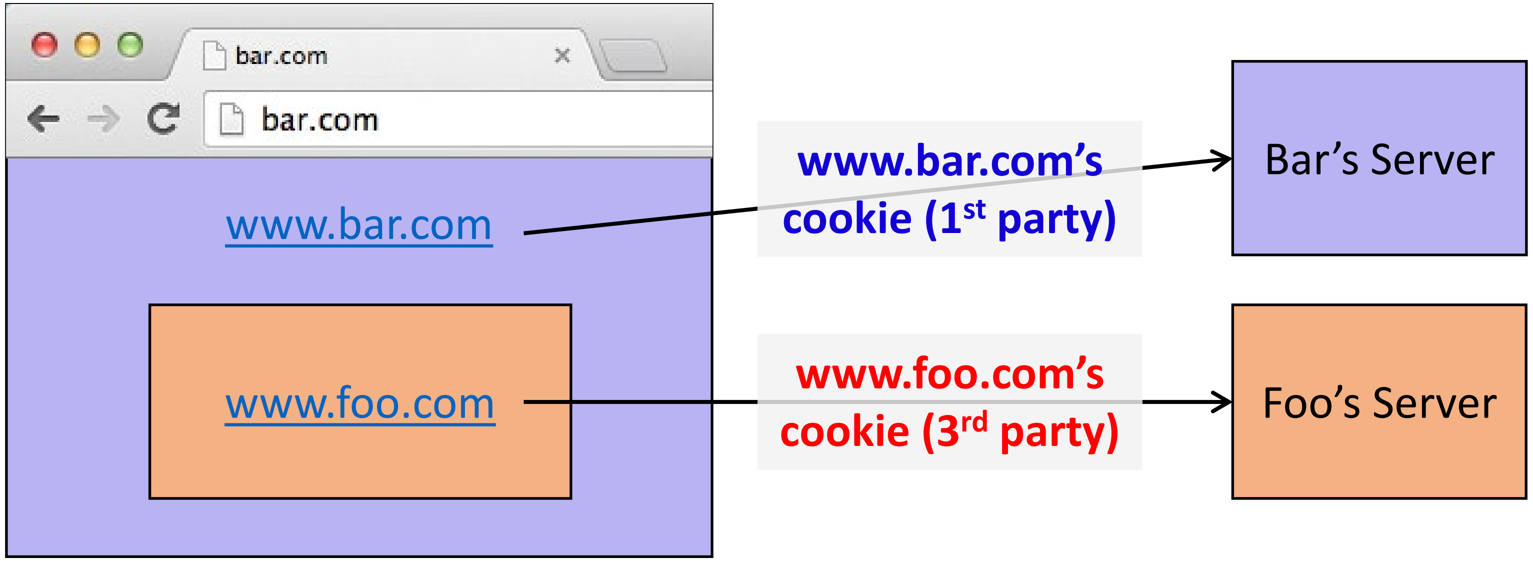
By TANZINA VEGA

And the privacy legislation just keeps on coming.

On Friday, two bills were introduced in Washington in support of a Do Not Track mechanism that would give users control over how much of their data was collected by advertisers and other online companies.

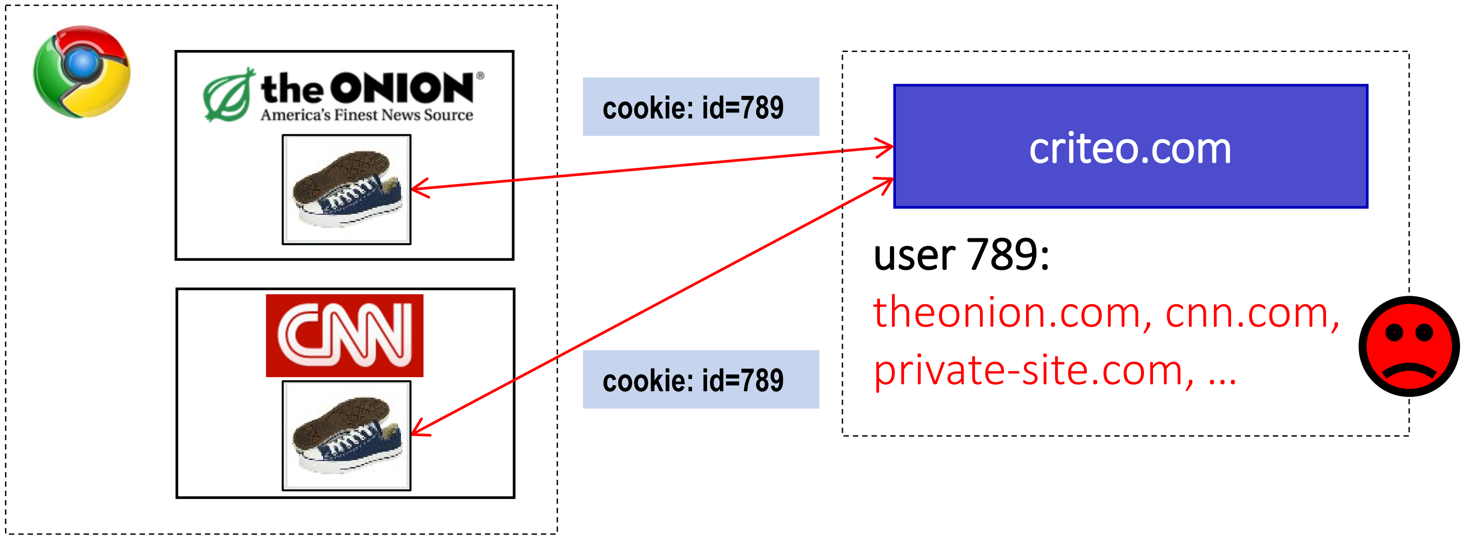
First and Third Parties

- First-party cookie: belongs to top-level domain.
- Third-party cookie: belongs to domain of embedded content (such as image, iframe).



Anonymous Tracking

Trackers included in other sites use third-party cookies containing unique identifiers to create browsing profiles.



Basic Tracking Mechanisms

- Tracking requires:
  (1) re-identifying a user.
  (2) communicating id + visited site back to tracker.

```
▽ Hypertext Transfer Protocol
  ▷ GET /pixel/p-3aud4J6uA4Z6Y.gif?labels=InvisibleBox&busty=2710 HTTP/1.1\r\n
    Host: pixel.quantserve.com\r\n
    Connection: keep-alive\r\n
    Accept: image/webp,*/*;q=0.8\r\n
    User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_9_2) AppleWebKit/537.36
    Referer: http://www.theonion.com/\r\n
    Accept-Encoding: gzip,deflate,sdch\r\n
    Accept-Language: en-US,en;q=0.8\r\n
    Cookie: mc=52a65386-f1de1-00ade-0b26e; d=ENkBRgGHD4GYEA35MMIL74MKiyDs1A2MQI1Q
```

Tracking Technologies

- HTTP Cookies
- HTTP Auth
- HTTP Etags
- Content cache
- IE userData
- HTML5 protocol and content handlers
- HTML5 storage

- Flash cookies (retired)
- Silverlight storage
- TLS session ID & resume
- Browsing history
- window.name
- HTTP STS
- DNS cache

- "Zombie" cookies that respawn (http://samy.pl/evercookie)

History Sniffing: A Side Channel

How can a webpage figure out which sites you visited previously?

- Color of links
  - CSS :visited property
  - getComputedStyle()
- Cached Web content timing
- DNS timing