I Still Know What You Visited Last Summer

Leaking browsing history via user interaction and side channel attacks

Zachary Weinberg, Eric Y. Chen, Pavithra Ramesh Jayaraman, Collin Jackson

CS558 – Presentation

Natasa Ntagianta
dagianta
Introduction

• Browsers have made a visual distinction between visited and unvisited links to pages from their users
• Combined with JavaScript’s ability to inspect page rendering, exposes Web users’ browsing history
• 2010 – first defense against all known automated techniques for this attack
  - Still possible to probe browsing history
• No effort in blocking interactive attacks
  - A side-channel attack is still possible
Background – the Web platform

- JavaScript made possible for programs to run inside Web pages – need for security
  - applications must not interfere with each other
  - malicious software must not be permitted to exploit the user
- Web’s basic security policy: *same-origin policy* – partitions the Web by its servers

*In principle, a Web site must not be able to determine what other sites its visitors have visited*
**Background – threat model**

*History sniffing* – Illicit inspection of browsing history

- History sniffers can only ask whether particular URLs have been visited – can’t get list of visited URLs

- They control the contents of a site and a DNS domain and can get victims to visit their sites
  - interactive sniffing: victims must be willing to interact with sniffing site same way as with legitimates
Background – attack consequences

• Sites at risk of impersonation can sniff user history for visited phishing sites and warn their users for possible account compromising

• Actual history sniffers are similar to cookie tracking

• History sniffing could enable severe privacy violations, because unlike cookie tracking, it allows the sniffing site to know about visits to other sites with no relationship to it at all

• Knowledge of browsing habit can connect social network identities
Automated Attacks – direct sniffing

• JavaScript programs examine and manipulate pages using DOM – DOM provides access to the computed style of each HTML document

• A malicious site can guess possible visited pages by its users, create links to those and determine whether they were indeed visited by inspecting each link’s computed styles
Automated Attacks – indirect sniffing

Simple defense for direct sniffing: DOM’s computed style could pretend all links being styled as unvisited

- Two classes of indirect techniques for unvisited link detection
  • Make visited and unvisited links take different amount of space
    - causes unrelated elements to move; inspect those elements
  • Make visited and unvisited links cause different images to load
    - certain image loaded only if certain link is visited
    - does not even require JavaScript
Automated Attacks – side-channel sniffing

Side channel attacks exist when a system leaks information, through a mechanism that wasn’t intended to provide that information, bypassing the system’s security policy

• Timing attacks are the most well-known type of side-channel attack
  
  * eg. Make a page take longer to lay out if a link is visited
Automated Attacks – defense

Indirect and side-channel sniffing defense: limited CSS ability to control visited/unvisited link distinction – links are same size and take same amount of time to load

• Ensure that selector matching takes the same amount of time whether or not link are visited
Experiment 1: Interactive attacks – the tasks

• Use visited-link styles to change text color or graphics
• Designed to probe 8 to 100 links each
• Masquerade as a plausible interaction

Such challenges are referred to as CAPTCHAs

Interactive attacks involve placing hyperlinks on the screen – victim actions reveal visited links to the attacker

- can be detected if a victim experiments with the page
Experiment 1: Interactive attacks – the tasks

Please type all the words shown below, then press RETURN.

low hang  we life alone  line cost

Word CAPTCHA: Each word is a hyperlink to a URL the attacker wishes to probe

• If link is visited it is drawn in black
• If unvisited it is drawn same color as the background
Experiment 1: Interactive attacks – the tasks

Please type the string of characters shown below, then press RETURN. You don’t have to match upper and lower case.

FAYA SABA A-65 A9-5

Character CAPTCHA: by clever choice of font and symbols it tests visitedness of 3 links per character
Experiment 1: Interactive attacks – the tasks

Character CAPTCHA: by clever choice of font and symbols it tests visitedness of 3 links per character

• History sniffing defense does not allow visited-link rules to change transparency of a color
  - work around this by making the characters always nearly transparent
Experiment 1: Interactive attacks – the tasks

Chessboard puzzle: Only pawns corresponding to visited sites are made visible

Page can’t tell number of clicks each user makes; attackers must resort to a time-out or explicit “go on” button
Experiment 1: Interactive attacks – the tasks

The large image on the left was assembled from two of the small images on the right: one from the first row and one from the second. Please click on the two small images that make up the large one.

Pattern matching puzzle: select two images that when assembled produce a composite one
Experiment 1: Procedure – Results

• Randomly generated task instances corresponding to known proportions of visited and unvisited links

Overall accuracy rates for the four interactive tasks
Experiment 1: Discussion

• Interactive attacks are feasible, especially if the attacker is interested in a small set of links

• Obvious defense is to prevent link-drawing in the same color as the background
  - merely determining the background color is difficult

• Interactive attacks don’t need to make anything invisible
  - there should at least be a visible difference on the user’s screen
Experiment 1: Discussion

- Limiting the visited-links revelation circumstances might be an appropriate move
  - links are revealed as visited if:
    - they target a document in the same domain
    - link destination has been previously visited from this site
    - current site is on a whitelist of trusted sites
Experiment 1: Discussion

- Attacker can still construct an interactive attack if *any* links are revealed as visited
  - they can draw one pixel of the whitelisted page directly above that link, using *iframes*

Even without a whitelist, attackers can draw one pixel iframes of the sites they want to know about

Private browsing can prevent remote history sniffing attacks with the cost of not distinguishing visited from unvisited links at all
Experiment 2: Side-channel attack – Webcam

• If the color of an area of the screen depends on visited or unvisited link, an attacker can use the camera to detect that color – sophisticated image processing code could detect even small differences

Two major obstacles for this attack:
• Camera activation needs user’s permission
• Necessary to change the color of the screen frequently – screen flash
Experiment 2: Side-channel attack – Webcam

Developed and tested two variants of this attack

• First variant designed to comply with WCAG standard for seizure safety
  - all WCAG requirements make detecting the change in reflected light more difficult, but not impossible

• Second variant made entire browser window flash – used brighter colors for visited/unvisited links
  - in that case it was obvious something unusual was happening
Experiment 2: Results – Discussion

• 100% accuracy achieved in controlled conditions where room was well-lit and computer user remained still
  - dropped to 50% if room was dark

• Most serious obstacle would be to persuade the victims to allow access to their webcams
Related Work

Visited-link state is not the only way to determine whether the user has visited a site – two other techniques involve timing attacks on local caches

• Page cache: measuring the time it takes to load a site, a web page can determine if it was already in the browser’s cache

• DNS cache: can also reveal user’s search queries – some browsers prefetch DNS entries for sites the user is likely to visit in the future

*These techniques work only on the first attempt, because the attack itself causes the information to be cached*
Even if the user clears their cookies, a site may be able to reconstruct them based on other browser state

- Private tools – as Torbutton – aim to mitigate or prevent these attacks at the cost of web functionality

- Private browsing mode can also mitigate some of these attacks

- Ad-blockers prevent many real-world cases of behavior profiling as a side effect
Conclusion

History sniffing as unintended consequence of
• Visited-link indication to the user
• CSS control of all aspects of page appearance
• JavaScript monitoring of page rendering

*Attacks that involve the user remain possible*