

Web Timing Attacks

CS487

Today's discussion

- ◆ A new form attack on web users
- ◆ Attack allow any web site to determine whether or not each visitor to the web site has recently visited another web site
- ◆ Since using this attack's one's online habits may be learnt it is an attack on privacy
- ◆ Falls under an important class of attacks called timing based attacks

Why Web Privacy matters

- ◆ Privacy of user activities on the web is a major concern
- ◆ List of web locations visited by a user often conveys detailed information about the user's family, financial or health condition
 - examples
- ◆ Users want to keep this information private
 - visiting a web site leaks information to that web site
- ◆ Ensure that this is not available to 3rd parties

Timing based attacks are problematic

- ◆ These are not due to “bugs” in a browser that are, in theory, fixable
- ◆ They can be carried out without a victim's knowledge
- ◆ Standard “anonymization” networks do not help
- ◆ Disabling browser features such as Java or Javascript doesn't help
- ◆ Only effective remedy requires a slowdown in web operations

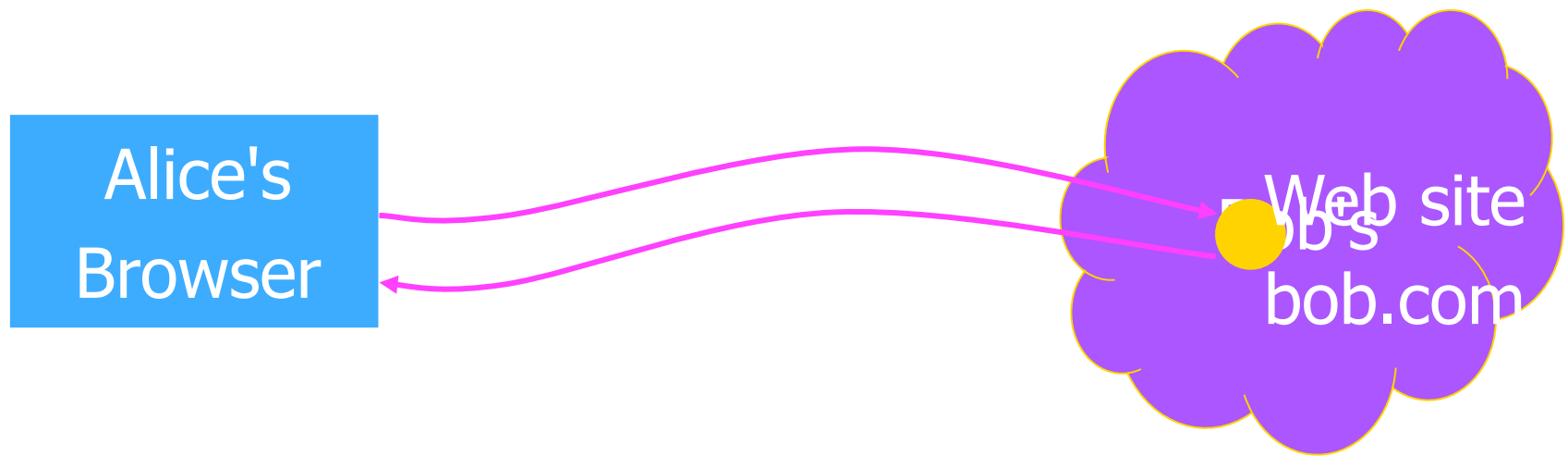
Why would one want this information?

- ◆ An insurance company may want to know if the user visited websites related to a particular medical condition
- ◆ Employer's web site could determine if an employee visited the site of a political organization

Main Idea. Exploit web

- ◆ What is web caching?
caching
- ◆ Accessing web documents takes a long time, so browsers cache this information
 - Save copies of recently-accessed files
 - Makes access to recently accessed documents faster
 - Transparent to the user
- ◆ Attack exploits this fact
 - measures time to access a particular file
 - If cached, this will be fast, otherwise slow
 - Measuring access time will reveal this information

Browser and Network



- ◆ Now, Bob wants to find out if Alice visited Charlie (charlie.com) or not
- ◆ How to do this using timing information ??

The Basic Attack

- ◆ First, Bob looks at Charlie's site and picks a file that any visitor to Charlie's site will see
 - <http://www.charlie.com/logo.jpg>
- ◆ Bob now is going to determine if the logo is present in Alice's web cache
- ◆ If the file is in her cache, Alice must have visited Charlie's site

Attack

- ◆ Bob write's a Java applet that implements this attack, and embeds it in his home page
- ◆ Alice views this web page
 - Java applet downloaded and run in Alice's computer
- ◆ Applet now measures time required to access <http://www.charlie.com/logo.jpg> and reports to Bob
- ◆ If time is less than some threshold, Bob learns that Alice visited Charlie's site

How to measure access time?

- ◆ Main component involves measuring access time by Bob
- ◆ Java and Javascript have facilities to do this
 - ◆ javascript: onload event (before setting src attribute of image)
- ◆ Could we prevent attack by disabling Java or Javascript?
 - Answer is no

Measuring time without Java or Javascript

- ◆ Write a web page that loads three files in a sequence
 - 1. A dummy file on the attacker's side
 - 2. A file whose access time is to be measured
 - 3. another dummy file on the attacker's side
- ◆ Attacker's web server measures difference between 1 and 3
- ◆ All this can be done without knowledge of victim

Improving accuracy

- ◆ Threshold crucial in determining attacks
- ◆ Use known hits and known misses
 - known misses can be generated by non-existent URLs
 - known hits can be the main measurement
- ◆ Combine results of several measurements
 - e.g., several images from Charlie's page

How to deliver attack?

- ◆ Induce victim to view web page
 - Similar to cross site scripting
 - Unscrupulous organization could put measurement code on its web site
 - Web advertising agency could add measurement code to banner ads
 - Attacker could have access to a web page with a high page rank
 - Victim can be lured by an email message
 - HTML email can be used to perform attack

Un-cacheable files

- ◆ Not all files are cacheable
 - dynamically created files created by POST responses
 - approx. 60% of files on web are cacheable files
- ◆ Often uncacheable files refer to cacheable files, e.g., logo files
- ◆ To carry out attack, need only one successful file

Anonymization does not help

- ◆ Since accesses made through anonymity systems are subject to caching, none of these systems can prevent timing attacks
- ◆ Some may block Java / Javascript, but the three-file strategy will still work

Exploiting DNS caching

- ◆ End-user machines most often cache DNS query results
- ◆ Most machines have a DNS cache
- ◆ Measure times made to query DNS and determine if this was cached
- ◆ Using the same delivery methods, the attacker can mine whether the victim visited a particular web site