Cross-Site Request Forgery

Part 1: Quick Overview

Cross-Site Request Forgery

Quick Overview 1/6

- Misconception #1: CSRF = XSS
  - Fact: CSRF and XSS are completely different attack vectors
  - XSS: Attacker inserts text (for example JavaScript code) onto website by sending the victim a specially prepared link
    - <script>alert('owned')</script>

Cross-Site Request Forgery

Quick Overview 1/6

- Misconception #1: CSRF = XSS
  - Fact: CSRF and XSS are completely different attack vectors
  - XSS: Attacker inserts text (for example JavaScript code) onto website by sending the victim a specially prepared link
    - <script>alert('owned')</script>
  - CSRF: Victim sends attacker's request to the webserver without knowing about it
    - http://www.example.com/admin/deleteuser.php?id=xxx

Cross-Site Request Forgery

Quick Overview 2/6

- Misconception #2: Preventing XSS stops CSRF
  - XSS makes CSRF easier, but it isn't required

Cross-Site Request Forgery

Quick Overview 3/6

- Other Names: CSRF (Sea Surf), XSRF, Session Riding

Cross-Site Request Forgery

Quick Overview 3/6

- Other Names: CSRF (Sea Surf), XSRF, Session Riding
- That's the stuff that goes on in my head

Cross-Site Request Forgery

Quick Overview 4/6

- Possible ways of utilizing CSRF to attack a website
  -

Cross-Site Request Forgery

Quick Overview 4/6

- Possible ways of utilizing CSRF to attack a website
  -
  - Iframes

Cross-Site Request Forgery

Quick Overview 4/6

- Possible ways of utilizing CSRF to attack a website
  -
  - Iframes
  - Automatic Redirects (for example Meta Refresh)

Cross-Site Request Forgery

Quick Overview 5/6

- *Everything* a website allows a customer/user to do can be abused unless this action is specifically protected against CSRF

Cross-Site Request Forgery

Quick Overview 5/6

- *Everything* a website allows a customer/user to do can be abused unless this action is specifically protected against CSRF
- Yes, *everything* !

Cross-Site Request Forgery

Quick Overview 5/6

- *Everything* a website allows a customer/user to do can be abused unless this action is specifically protected against CSRF
- Yes, *everything* !
- Requirement: User is logged in to this website

Cross-Site Request Forgery

Quick Overview 6/6

- Key to understanding CSRF
  - Web applications don't verify that a given *user* is performing a request

Cross-Site Request Forgery

Quick Overview 6/6

- Key to understanding CSRF
  - Web applications don't verify that a given *user* is performing a request
  - They are verifying that a given *browser* is performing said request by checking the cookies

Questions?
arne@aachen-method.com

Next
Part 2: High-Profile Victims