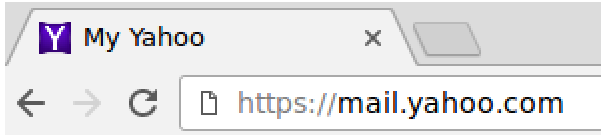
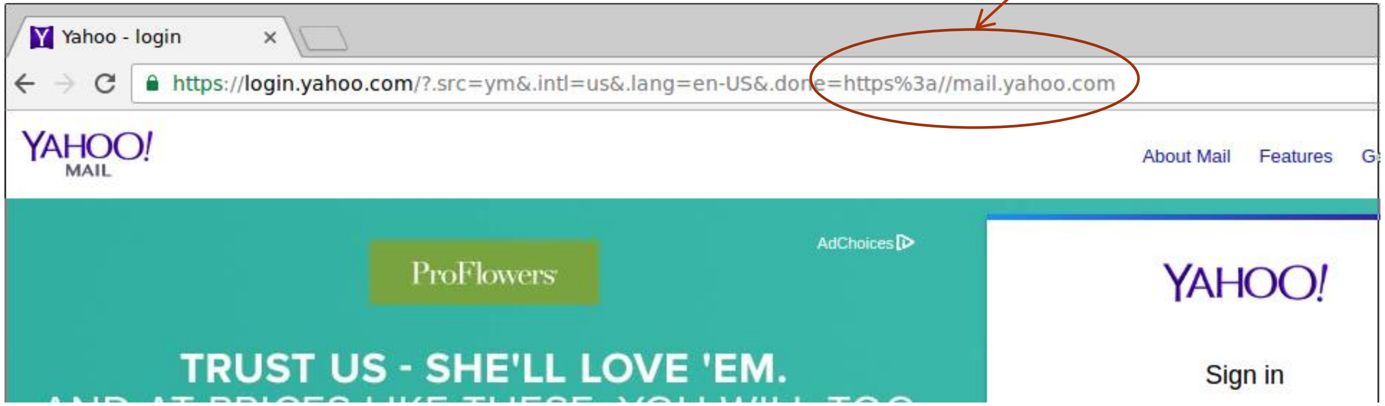
A10: Unvalidated Redirects and Forwards
Axx: Unsolicited Framing

A10: Unvalidated Redirects

- **Web application redirects are very common**
  - Redirect request to a URL-supplied destination
    - User accesses page requiring auth
    - Redirected to login page with URL of origin page as parameter
    - After login, redirected back to URL of origin page



redirect to this URL after login


Yahoo - login
https://login.yahoo.com/?.src=ym&.intl=us&.lang=en-US&.done=https%3a//mail.yahoo.com
YAHOO! MAIL
About Mail
Features
AdChoices
ProFlowers
TRUST US - SHE'LL LOVE 'EM.
YAHOO!
Sign in

    - What if someone screen-scraped Yahoo, found an unvalidated redirect on one of its properties, and phished you with this link in a page/email?

```
https://r.yahoo.com/?.src=ym&.intl=us&.lang=en-US&.done=https%3a//login.yahoo.com.cxx
```

A10: Unvalidated Redirects

- **If not validated, request bounces off of a site that is legitimate and sends victim to a site run by the adversary for phishing or automated malware download**
  - Victim sees something that has the right domain, ends up at a site that looks like it (but controlled by adversary)
  - Podesta perhaps?
- **What attack in the last lecture is this similar to?**

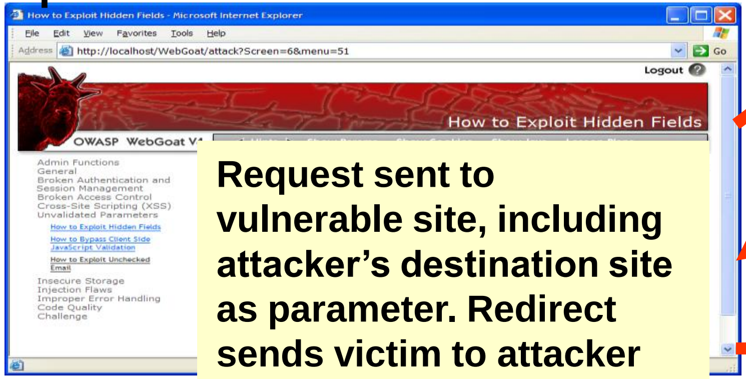
Unvalidated Redirect Illustrated

1 Attacker sends attack to victim via email or webpage

From: Internal Revenue Service
Subject: Your Unclaimed Tax Refund
Our records show you have an unclaimed federal tax refund. Please click here to initiate your claim.

https://www.irs.gov/taxrefund/claim.jsp?year=2006& … &dest=www.evilsite.com

2 Victim clicks link containing unvalidated parameter



How to Exploit Hidden Fields - Microsoft Internet Explorer

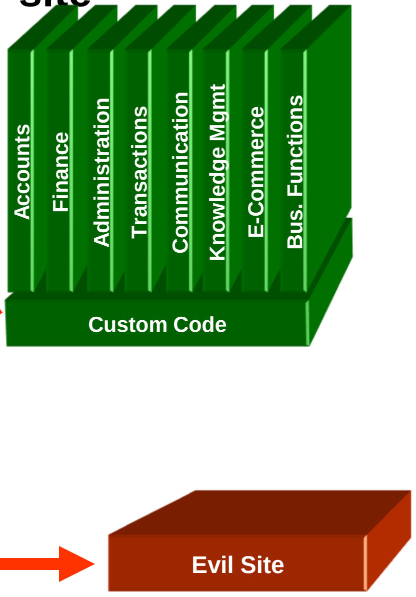
File Edit View Favorites Tools Help

Address http://localhost/WebGoat/attack?Screen=6&menu=51

Go
Logout
How to Exploit Hidden Fields
OWASP WebGoat V4
Admin Functions
General
Broken Authentication and Session Management
Broken Access Control
Cross-Site Scripting (XSS)
Unvalidated Parameters
How to Exploit Hidden Fields
How to Bypass Client Side JavaScript Validation
How to Exploit Unchecked Email
Insecure Storage
Injection Flaws
Improper Error Handling
Code Quality
Challenge

Request sent to vulnerable site, including attacker's destination site as parameter. Redirect sends victim to attacker site

3 Application redirects victim to attacker's site



4 Evil site installs malware on victim, or phish's for private information

A10: Unvalidated Redirects

- Java

```
response.sendRedirect(request.getParameter("url"));
```

- PHP

```
$redirect_url = $_GET['url'];
header("Location: " . $redirect_url);
```

.NET redirect example

```
public ActionResult LogOn(LogOnModel model, string returnUrl) {
   if (ModelState.IsValid) {
      if (MembershipService.ValidateUser(model.UserName, model.Password)) {
         FormsService.SignIn(model.UserName, model.RememberMe);
         if (!String.IsNullOrEmpty(returnUrl)) {
            return Redirect(returnUrl);
         }
         else {
            return RedirectToAction("Index", "Home");
         }
      }
      else {
         ModelState.AddModelError("", "Incorrect user name or password.");
      }
  } // If we got this far, something failed, redisplay form
  return View(model);
}
```

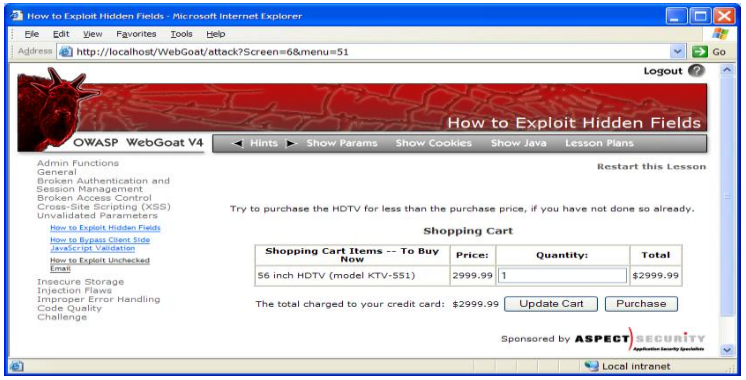
A10: Unvalidated Forwards

- **Forwards similar to redirects, but remain in same web application**
  - Transfer in .NET
  - Internally send the request to a new page in the same application
    - If access to target page not validated, attacker may be able to use unvalidated forward to bypass authentication or authorization checks

Unvalidated Forward Illustrated

1 Attacker sees link in vulnerable, but accessible page that calls the forward

Forwarding code assumes “dest” set via page and has no malicious values



How to Exploit Hidden Fields - Microsoft Internet Explorer

http://localhost/WebGoat/attack?Screen=6&menu=51

How to Exploit Hidden Fields

Admin Functions
General
Broken Authentication and Session Management
Broken Access Control
Cross-Site Scripting (XSS)
Unvalidated Parameters
- How to Exploit Hidden Fields
- How to Bypass Client Side JavaScript Validation
- How to Exploit Unchecked Email

Insecure Storage
Injection Flaws
Improper Error Handling
Code Quality
Challenge

Try to purchase the HDTV for less than the purchase price, if you have not done so already.

Shopping Cart

Shopping Cart Items -- To Buy Now	Price:	Quantity:	Total
56 inch HDTV (model KTV-551)	2999.99	1	$2999.99

The total charged to your credit card: $2999.99 Update Cart Purchase

Sponsored by ASPECT SECURITY Application Security Specialists

2 Application authorizes request, which continues to the forward

Filter

```
public void doPost( HttpServletRequest request,
HttpServletResponse response) {
    try {
        String target = request.getParameter( "dest" ) );
        ...
        request.getRequestDispatcher( target
        ).forward(request, response);
    }
    catch ( ...
```

```
public void sensitiveMethod(
HttpServletRequest request,
HttpServletResponse response) {
    try {
        // Do sensitive stuff here.
        ...
    }
    catch ( ...
```

3 Forwarding pathway fails to validate destination page. Attacker sets target to a page of his/her choosing (potentially an unauthorized page), bypassing access control

JSP forward example

- **Redirect within site via internal fwd parameter**

```
public class ForwardServlet extends HttpServlet
{
  protected void doGet(HttpServletRequest request, HttpServletResponse response)
                 throws ServletException, IOException {
    String query = request.getQueryString();
    if (query.contains("fwd"))
    {
      String fwd = request.getParameter("fwd");
      try
      {
        request.getRequestDispatcher(fwd).forward(request, response);
      }
      catch (ServletException e)
      {
        e.printStackTrace();
      }
    }
  }
}
```

A10 – Prevention

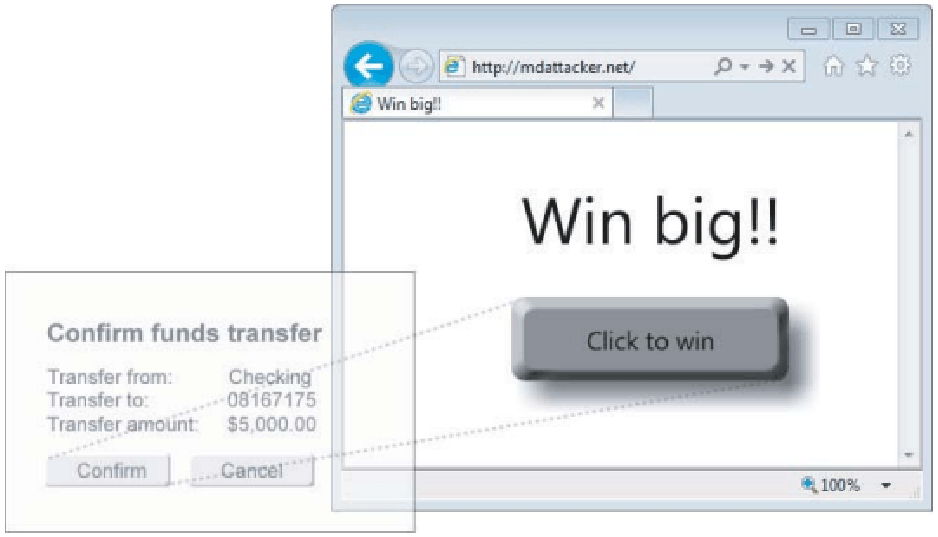
- **Avoid using redirects and forwards**
  - If used, don’t include user input in defining the target URL
  - If you ‘must’ include user input, then, validate each parameter to ensure its valid and authorized access
- **Whitelist redirect locations to ensure it goes to an authorized external site**
- **Force redirects first to a page notifying user of redirect and have them click to confirm**
- **Authorize via access controller before forwarding**
  - Ensure all users who can access the original page are ALL authorized to access the target page when used

OWASP resources

- **OWASP's Guide to Building Secure Web Applications**
  - https://www.owasp.org/index.php/Guide
- **Cheat sheets**
  - https://www.owasp.org/index.php/Cheat_Sheets
- **Application Security Verification Standard**
  - https://www.owasp.org/index.php/ASVS
- **OWASP's ESAPI tools**
    - https://www.owasp.org/index.php/ESAPI

Axx: Unsolicited Framing, UI Redress (Clickjacking)

- **Users visit a malicious website**
  - Malicious site contains an `<iframe>` that loads a legitimate site in a transparent manner
  - Malicious site puts up an enticing button for user to click
  - User clicks on what appears to be button, but button in transparent frame clicked instead



Axx: Clickjacking prevention

- **HTTP header X-Frame-Options**
  - Sites can tell browsers never to load their content in an `<iframe>`
    - X-Frame-Options: DENY
  - Sites can tell browsers to only allow `<iframe>` from same site
    - X-Frame-Options: SAMEORIGIN
  - Sites can tell browsers to only allow `<iframe>` from specific site
    - X-Frame-Options: ALLOW-FROM https://example.com/

Axx: Clickjacking prevention

- **Initial approach**
  - HTTP header X-Frame-Options:
  - Note: ‘X’ means experimental and temporary
  - Sites can tell browsers never to load their content in an `<iframe>`
    - X-Frame-Options: DENY
  - Sites can tell browsers to only allow `<iframe>` from same site
    - X-Frame-Options: SAMEORIGIN
  - Sites can tell browsers to only allow `<iframe>` from specific site
    - X-Frame-Options: ALLOW-FROM https://example.com/
- **Current approach**
  - Content-Security-Policy header
    - `frame-ancestors` directive

Labs and homework

- See previous handout

Questions

- https://sayat.me/wu4f

Clickjacking example

```
mashimaro <~>  8:52PM % telnet google.com 80
Trying 2607:f8b0:400a:800::200e...
Connected to google.com.
Escape character is '^]'.
GET / HTTP/1.1
Host: google.com

HTTP/1.1 301 Moved Permanently
Location: http://www.google.com/
Content-Type: text/html; charset=UTF-8
Date: Mon, 06 Nov 2017 04:52:22 GMT
Expires: Wed, 06 Dec 2017 04:52:22 GMT
Cache-Control: public, max-age=2592000
Server: gws
Content-Length: 219
X-XSS-Protection: 1; mode=block
X-Frame-Options: SAMEORIGIN

<HTML><HEAD><meta http-equiv="content-type" content="text/html;charset=utf-8">
<TITLE>301 Moved</TITLE></HEAD><BODY>
<H1>301 Moved</H1>
The document has moved
<A HREF="http://www.google.com/">here</A>.
</BODY></HTML>
```