A10: Unvalidated Redirects and Forwards
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

What if someone screen-scraped Yahoo and put this link on a page/email?
A10: Unvalidated Redirects and Forwards

- **Unvalidated Redirect**
  - 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 is this similar to that we just covered?
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.

2. Victim clicks link containing unvalidated parameter


3. Application redirects victim to attacker’s site

4. Evil site installs malware on victim, or phish’s for private information
A10: Unvalidated Redirects and Forwards

- **Java**
  ```java
  response.sendRedirect(request.getParameter("url"));
  ```
- **PHP**
  ```php
  $redirect_url = $_GET['url'];
  header("Location: " . $redirect_url);
  ```
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 RedirectToArray("Index", "Home");
        }
    } else {
        ModelState.AddModelError("", "Incorrect user name or password.");
    }
} // If we got this far, something failed, redisplay form
return View(model);
A10: Unvalidated Redirects and 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
Attacker sees link in vulnerable, but accessible page that calls the forward. Forwarding code assumes “dest” set via page and has no malicious values.

Application authorizes request, which continues to the forward.

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

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

- Redirect within site via internal fwd parameter

```java
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
- Cheat sheets
  - [https://www.owasp.org/index.php/Cheat_Sheets](https://www.owasp.org/index.php/Cheat_Sheets)
- Application Security Verification Standard
  - [https://www.owasp.org/index.php/ASVS](https://www.owasp.org/index.php/ASVS)
- OWASP’s ESAPI tools
  - [https://www.owasp.org/index.php/ESAPI](https://www.owasp.org/index.php/ESAPI)
Homework

- See handout